Buffer Overflow for Beginners
19C3 Berlin

" .. und konnte Root-Rechte auf dem System erlangen, indem ein Buffer Overflow ausgenutzt wurde ..."

Hintergrund, Auswirkung und Vermeidung von Buffer Overflows

Dipl.-Inform. Viola Braeuer

Buffer Overflow Content

Zum Einstieg: Nette Beispiele

Was ist ein Buffer Overflow?

Wie kann er ausgenutzt werden? Exploit

Wege zur Vermeidung:
Sicherheitsbewusste Programmierung

Buffer Overflow Examples

Stammgast: MS Web-Server IIS Code Red

1988: fingerd: Internet-Wurm

Ja, auch unter Unix:

- Name-Server BIND (named)
- FTP-Server wu-ftp

Auch Sicherheits-Software:

- Firewall Gauntlet (Network Associates)

Buffer Overflow
How does it work?

Ueberschreiben der Ruecksprungadresse:

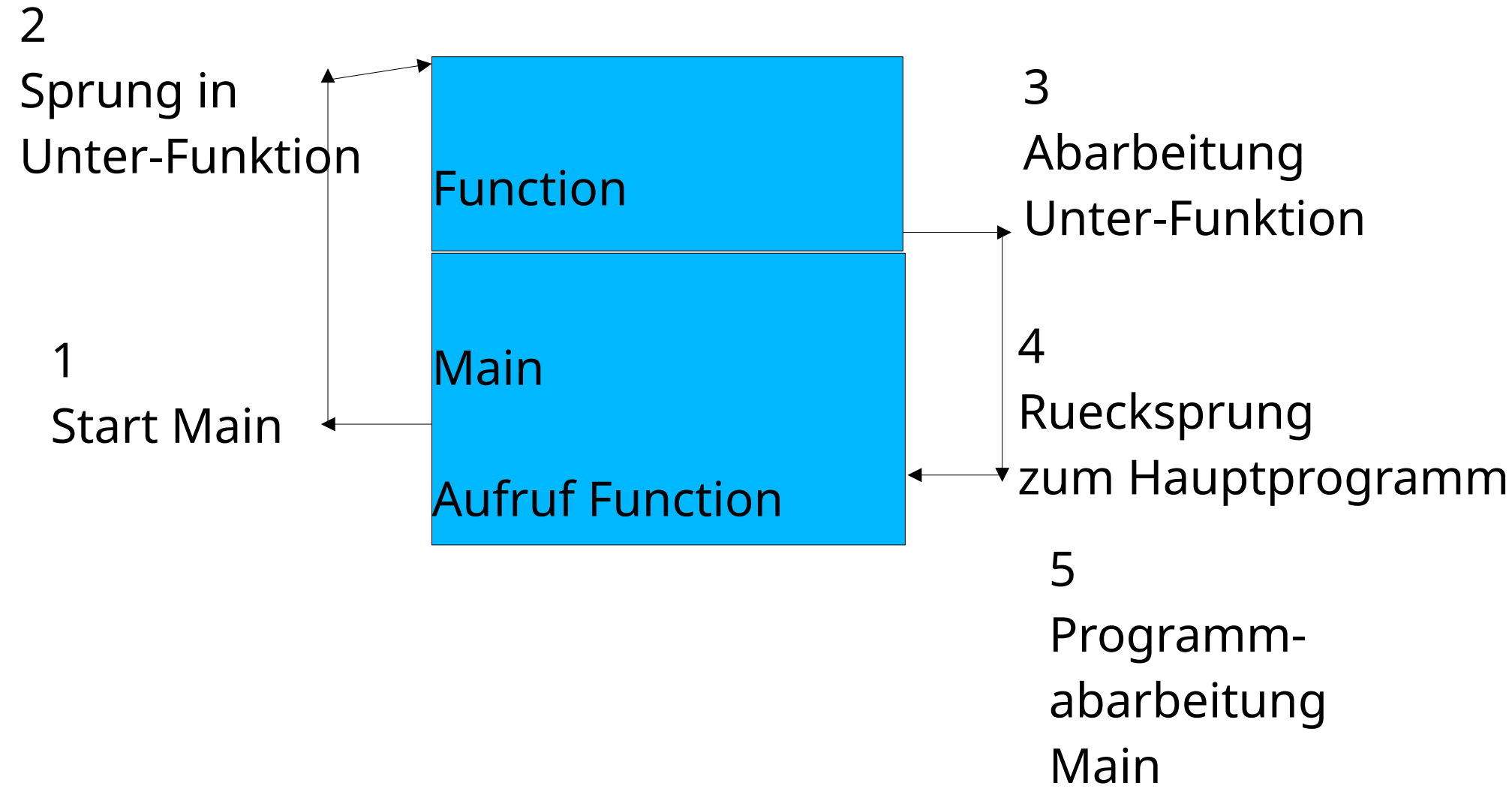
... auf dem Weg nach Hause die falsche Abzweigung erwischt--
Und wo GANZ anders gelandet..

Buffer Overflow
How does it work?

Ueberschreiben der Ruecksprungadresse durch ueberlange Eingabe so, dass sie auf die Start-Adresse des Mal-Codes zeigt

Einfuegen von Mal-Code: Erzeugung einer Root-Shell

Buffer Overflow
How does it work? Funktion



Buffer Overflow
How does it work?

Einfaches Beispiel in C:

```
void function(int a, int b) {
      char buffer[]=”ABCABCABC”
      }

void main() {
      ...
      function(3,4);
      }
```

Buffer Overflow

How does it work? Memory

Arbeitsspeichers zur Ausfuehrung eines Programms

wird zur Laufzeit zugewiesen und

Besteht aus drei Teilen:

- Stack Segment:
  - Zur Parameteruebergabe
- Daten-Segment (Heap):
  - Fuer Variablen und dynamischen Speicher: malloc()
- Code-Segment: Maschienen-Code

Buffer Overflow
How does it work? Memory

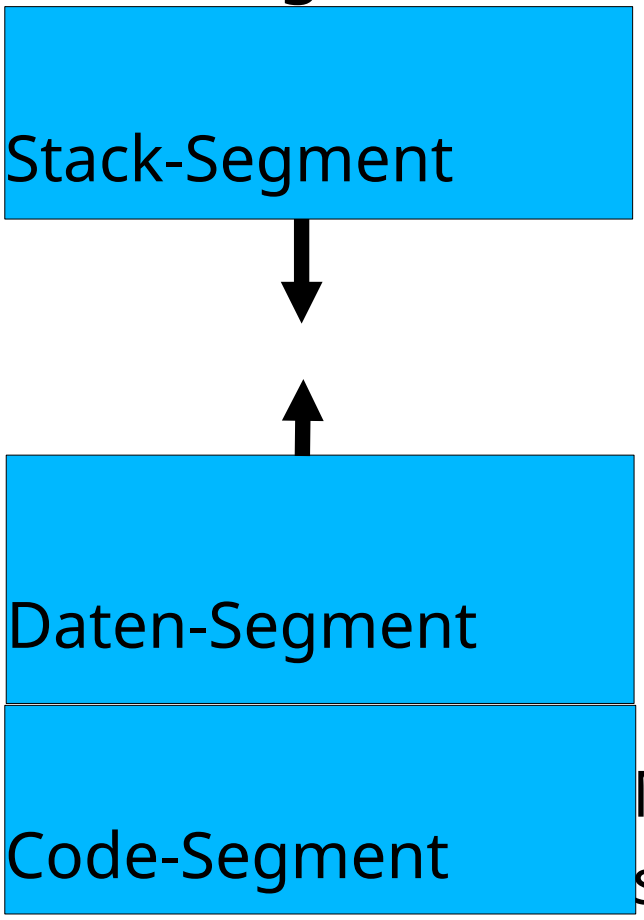
Zur Laufzeit des Programms werden

- Lokale Varibalen
- Uebergabeparameter fuer Funktionen
- Ruecksprungadressen fuer Funktionen

Im Arbeitsspeicher abgelegt (konkret im Stack)

Buffer Overflow
How does it work? Memory

Zur Laufzeit eines Programms:



Zwischenspeicher fuer lokale Variablen, Uebergabe-Parameter, Prozessorregister
= First in Last Out (FILO)

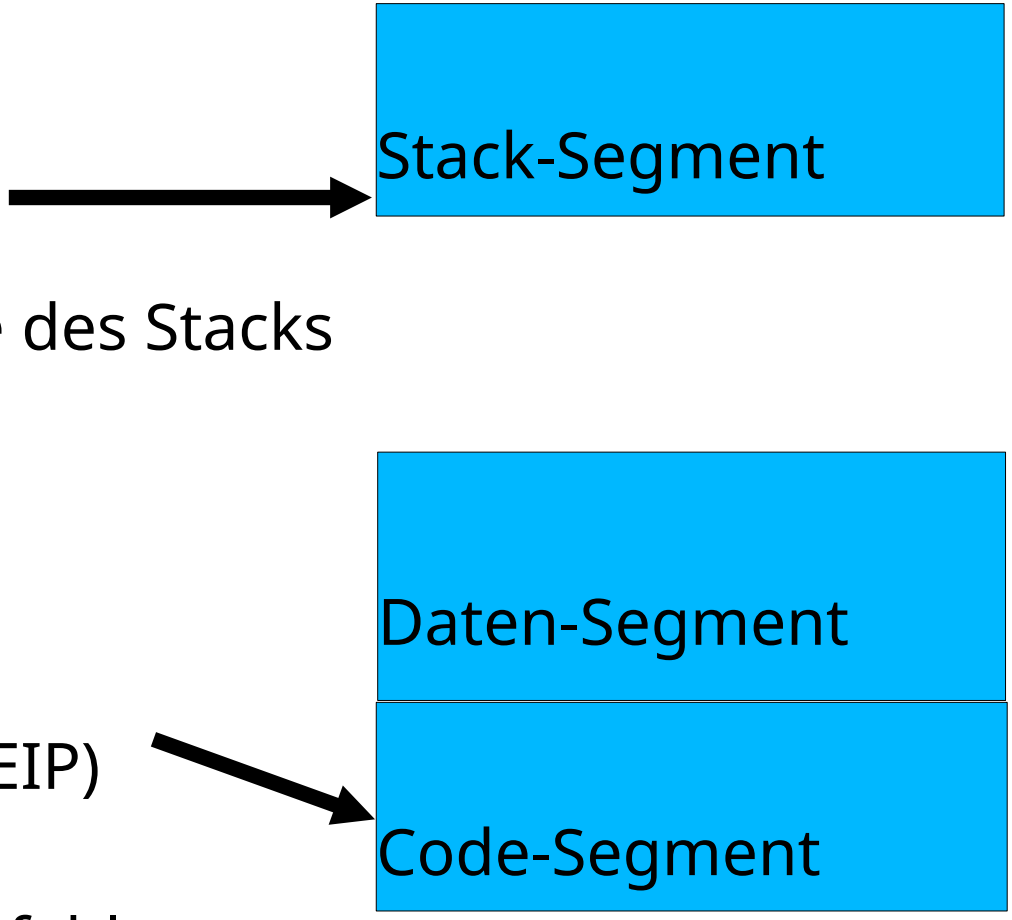
=Heap, fuer globale Daten und Konstanten,
Platz fuer malloc()

Maschienenprogramm
schreibgeschuetzt
(Segmentation violation)

0 Byte

Buffer Overflow
How does it work?

Pointer:



Buffer Overflow
How does it work? Frame

Frame-Konzept:

Base-Pointer (EBP)
zeigt auf den lokalen Speicher-Bereich der aktuellen Unter-Funktion auf dem Stack

Offset zur Navigation innerhalb des Frames

Buffer Overflow
How does it work?

Beim Beenden der Funktion wird

der Speicher auf dem Stack wieder freigegeben

Der “alte” Frame-Pointer wiederhergestellt

Die gespeicherte Ruecksprungadresse zum Instruktion Pointer (EIP)

Buffer Overflow
How does it work?

Stack-Operationen: **PUSHL** & **POPL**:

PUSHL = auf dem Stack ablegen

pushl %ebp

Ablage des Basepointers auf dem Stack

POPL = vom Stack entfernen

popl %eax

oberster Stack-Wert nach Register eax

Stack arbeitet nach LIFO-Prinzip (Last in First out)

Buffer Overflow
How does it work?

Altes Beispiel in C:

```
void function(int a, int b) {
      char buffer[]=”ABCABCABC”
      }

void main() {

      function(3,4);

      }
```

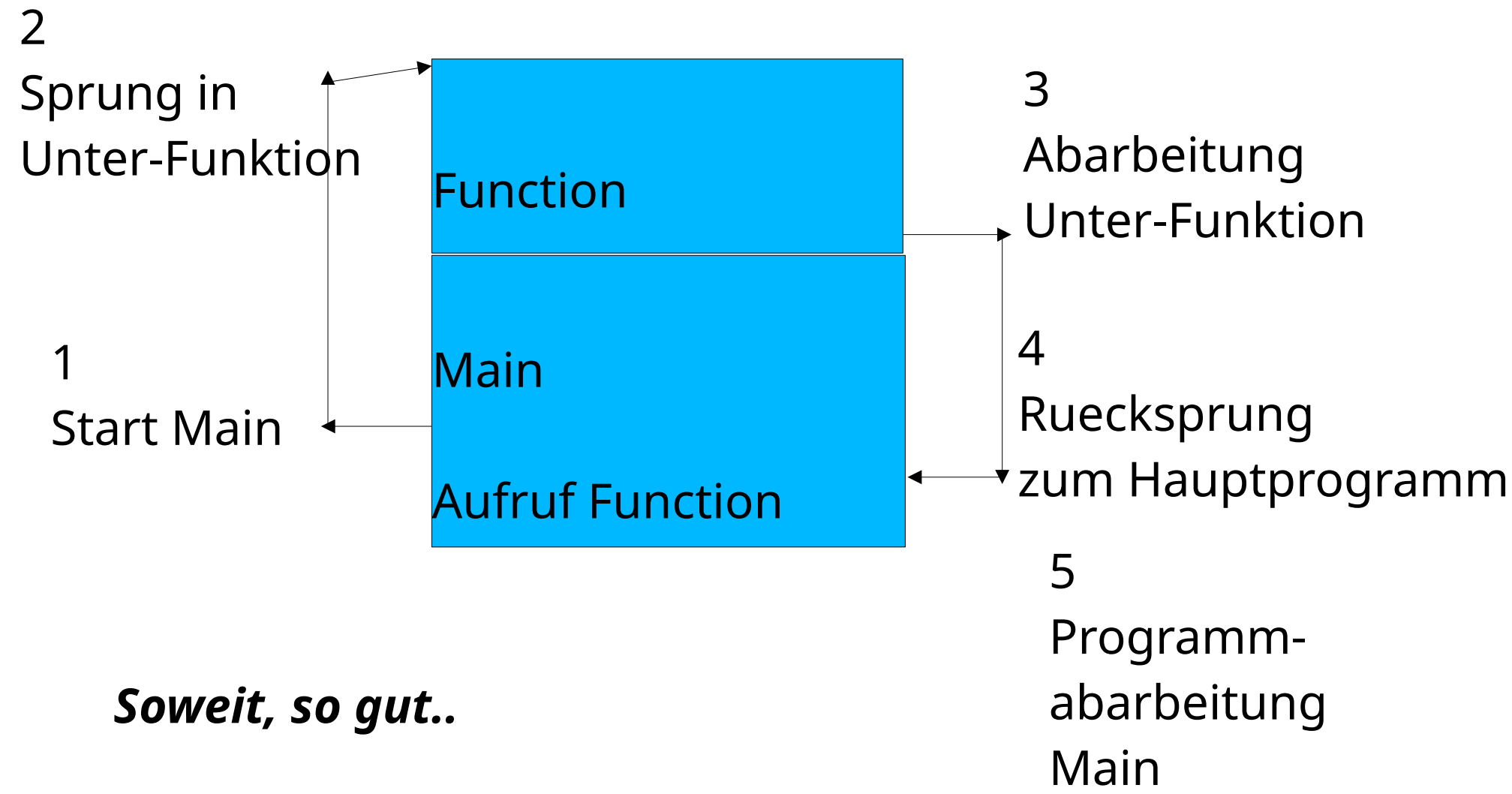
Buffer Overflow
How does it work?

```
;sieht in Assembler so aus:
;function
pushl %ebp
            ; alten BasePointer (FramePointer) auf dem
            ; Stack ablegen
movl %esp, %ebp
            ;alter StackPointer wird FramePointer
subl $10, %esp
            ;Speicher reservieren
..
leave       ;FramePointer wiederherstellen
ret         ;Ruecksprung
```

Buffer Overflow
How does it work?

```
;Assembler
;main
..
pushl $4        ; Parameteruebergabe via Stack
pushl $3
call function   ; Aufruf Unterfunktion sichern
                ; incl. Sicherung des Instruction Pointer
..
```

Buffer Overflow
How does it work?



Soweit, so gut..

Buffer Overflow Exploit

Ueberschreiben der Ruecksprungadresse durch ueberlange Eingabe, so dass sie auf die Startadresse des Mal-Codes zeigt

Einfuegen von Mal-Code: Erzeugung einer Root-Shell

Buffer Overflow Exploit

Ueberschreiben der Ruecksprungadresse durch ueberlange Eingabe: strcpy

```
function(char*a) {
        char buffer[4];
        strcpy(buffer,a);
        }


main() {
                function("dieser String ist zu lang");
          }
```

Buffer Overflow Exploit

Prinzip:

Gezieltes Ueberschreiben der Ruecksprungadresse,

So, dass sie auf den “neuen” Code zeigt.

Buffer Overflow Exploit

Gezieltes Ueberschreiben der Ruecksprungadresse:

Gesucht: Adresse, an der die Ruecksprungadresse gespeichert ist

Auf dieser die “neue” Ruecksprungadresse schreiben

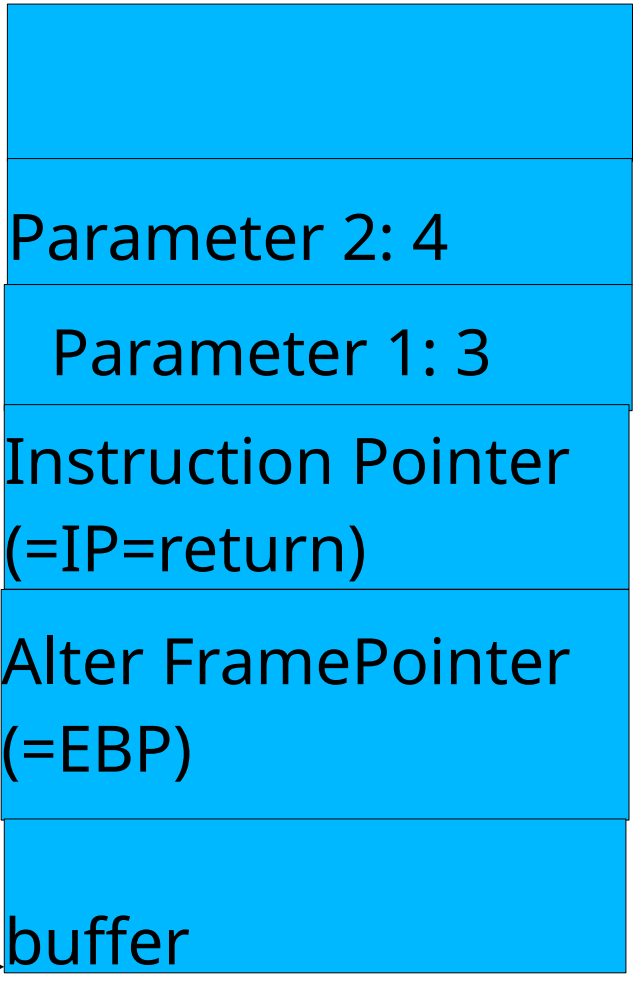
Buffer Overflow
How does it work?

Altes Beispiel in C:

```
void function(int a, int b) {
      char buffer[]=”ABCABCABC”
      }

void main() {

      function(3,4);

      }
```

Buffer Overflow
How does it work?

Stack:



Buffer Overflow Exploit

Gezieltes Ueberschreiben der Ruecksprungadresse:

“Einpacken” des Mal-Codes in NOPs (No Operation Anweisungen)

--> hoehere Wahrscheinlichkeit Mal-Code zu treffen

--> s. Literatur

Buffer Overflow Exploit

Einfuegen von Mal-Code auf dem Stack

Ueberlanger String beinhaltet Mal-Code:

Erzeugung einer Root-Shell:
“/bin/sh”

--->s. Literatur

Buffer Overflow
Exploit Shell-Code

char shellcode[]= “\xe8\x3f\x1f\xfd\xb4\x23\x03\
xe8\x60\x60\x3c\x61\x0b\x39\x02”
“\x99\x34\x1a\x3c\x53\x0b\x43\x06\x1a\x20\
x20\x08\x01\x34\x16\x03”

“\xe8\xe4\x20\xe0\x08\x96\xd6\x03\xfe/**bin/sh**";

Buffer Overflow
Exploit Auswirkungen

Root-Rechte auf dem Rechner (wenn exploitetes Programm mit Root-Rechten laeuft)

Das Uebliche:

- Sammeln von Passwoertern
- Einbau von Hintertueren (Backdoors)
- Verwischen der Spuren (Root-Kits)
  - Logfiels loeschen
  - Binaries ersetzten
- Als Plattform fuer weitere Angriffe

Buffer Overflow Gegenmassnahmen

Prinzipiell einfach ..

Sicherheitsbewusste Programmierung,
Erst denken, dann programmieren
Weiterbildung (Eigenheiten der benutzten Programmiersprache, bes. C)
Kampf der Featuritis
Zeit fuers Testen nehmen

Buffer Overflow Gegenmassnahmen

Konkrete Programmiersprache:

Sprache C:

beim Kopieren von Strings Zeichen begrenzen mit strncpy() statt strcpy()

Und andere C-Funktionen (s. Lit.)

Strcat, sprintf, scanf, gets,..

Java benutzen:

Grenzen der Speicherbereiche werden ueberwacht

Buffer Overflow Gegenmassnahmen

Konkret Returnadresse sichern:

StackShield fuer Linux: Sichert Return-Adresse und korrigiert sie bei Bedarf

StackGuard: Wenn Return-Adresse geaendert, Meldung und Programm-Stop

Andere Ansaetze:

Nicht-ausfuehrbare Stack (im Betriebssystem)

Security Enhanced Linux (NSA)

Detailierte Rechte-Vergabe

Buffer Overflow
Literatur

"Shamshing the stack for fun and profit"

Aleph One,
http://phrack.org/show.php?p=49&a=14

Assemblerprogrammierung unter Linux:
http://linuxassembly.org/

Secure Programming for Linux and Unix

HOWTO:www.dweehler.com/secure-pragrams/

StackShield:www.angelfire.com/sk/stackshield/
StackGuard:http://immunix.org

Ct 23/2001 S.216 ff.

Buffer Overflow Fazit

Fazit + These:

Sicherheitsbewusste Programmierung

Akzeptanz von Fehlern:

Es ist immer einfacher, ein Loch zu finden, als ein komplexes System zu entwerfen, das sicher sein soll.

These:

Alle lebendigen komplexen Systeme sind vulnerable.

Buffer Overflow
Exploit

Einfuegen von Mal-Code - Fallstricke:

“0” beendet strcpy
-> Eleminierung von Nullen