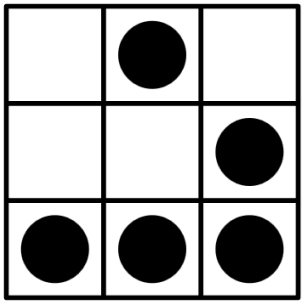
Hacking
in C
(Windows)



Reinhard Oertel

Agenda

- Einleitung
- Techniken
  - Hintergrundwissen
  - Buffer-Overflow
- Zusammenfassung

Meldungen 2011

Attacke auf Playstation-Netzwerk

Hacker stehlen Millionen Sony-Kundendaten

Es könnte einer der größten Datendiebstähle der Geschichte sein: Bei einem Hackerangriff auf das Playstation-Netzwerk von Sony sind Daten von mehr als 70 Millionen Nutzern gestohlen worden - möglicherweise auch Kreditkarteninformationen. Der Konzern rät, Bankabrechnungen genau zu kontrollieren.

Cyberwar

Nato-Staaten rüsten für das fünfte Schlachtfeld

Schlachten werden auch über Facebook und Twitter geschlagen - das haben die Revolutionen Arabiens bewiesen. Cyber-Krieger im Pentagon haben nun Software bestellt, mit der sie Meinung im Netz manipulieren können - in Farsi, Arabisch, Urdu und Paschtu. In den USA selbst wäre das illegal.

COMPUTERSICHERHEIT

Deutsche Stromkonzerne von Stuxnet befallen

Sicherheitslücke

Siemens bestätigt Schwachstellen in Industrie-Software

Es geht um Kraftwerke, Fabriken, Stahlhütten: US-Forscher haben Sicherheitslücken in Siemens-Steuersoftware für Industrieanlagen entd

Computerviren

IT-Manager warnen vor Cyber-Attacken auf Stromnetze

Der Computerwurm Stuxnet zeigt, wie massiv Infrastruktur manipuliert werden kann. Strom-, Gas- und Wasserversorgung sind besonders angreifbar. Eine Befragung unter IT-Verantwortlichen aus 14 Staaten offenbart gravierende Sicherheitslücken.

Mutmaßlicher "Comodo-Hacker"

"Ich bin nicht zu stoppen, also fürchtet euch"

Es war ein spektakulärer, bedrohlicher Hack, nun gibt es ein Bekennerschreiben: Ein junger Iraner rühmt sich, Zertifikate der Firma Comodo gestohlen zu haben, eine Art Sicherheitsbasis des Webs. Er warnt Dissidenten vor Lauschangriffen - und droht mit Rache für den Stuxnet-Virus.

Quelle: Spiegel.de

Hacker

Gebräuchliche Definition

- Jemand, der über das Netzwerk in ein Computersystem eindringt (um sich ggf. Informationen zu beschaffen).

Definition

- Experimentierfreudige Personen, die mit Fachkenntnissen eine Technologie beliebiger Art außerhalb ihrer normalen Zweckbestimmung oder ihres gewöhnlichen Gebrauchs benutzen bzw. anpassen

Abgrenzung

- eine allgemeine Definition ist schwer
  - es gibt viele Strömungen innerhalb der Subkultur
  - Strömungen haben jeweils eigene Überzeugung

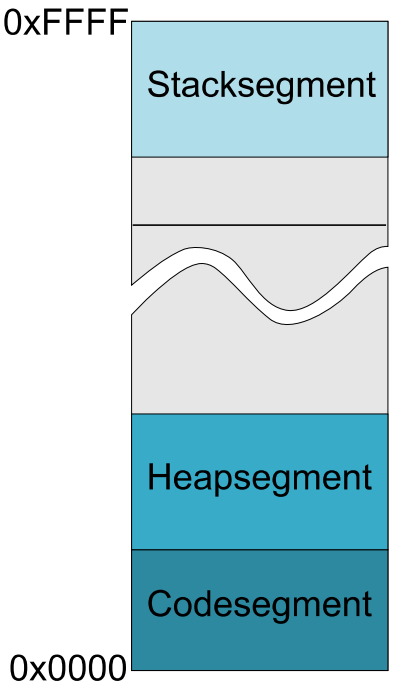
- „Jargon File“ versucht Subkultur zu erfassen
  - eine Art Enzyklopädie des Hackertums
  - seit 1975 von freiwilligen Autoren kontinuierlich aktualisiert und erweitert

Einteilung

- Die „wahren“ Hacker
  - in der Linux / Open Source / Free Software Szene tätig
- Die „aufklärerischen“ Hacker
  - hauptsächlich im CCC, für metasploit & Co, etc. tätig
- Die „Cracker“
  - hauptsächlich in der Demo / Warez / Viren Szene tätig
- Die „Polithacker“
  - Beipiel: Virus (*Stuxnet*) gegen iranisches Atomprogramm

Techniken

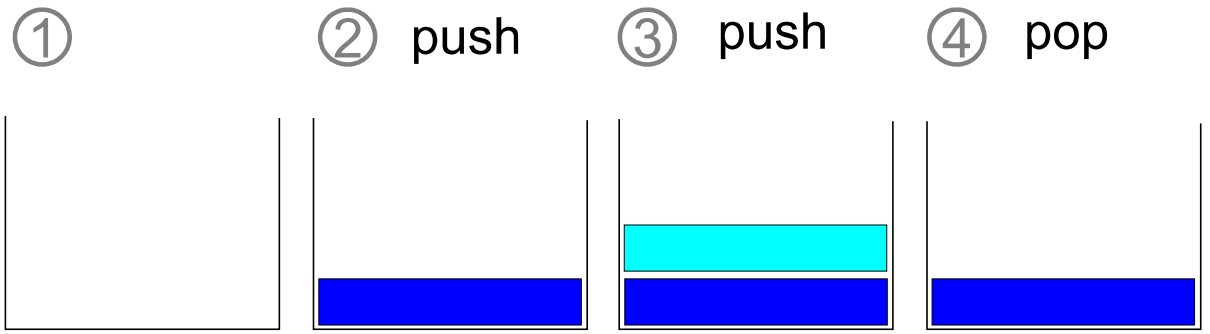
- Buffer-Overflow (Pufferüberlauf)
  - **Stack-Overflow**
  - **Heap-Overflow**



Hintergrundwissen

Stack

- auch Stapelspeicher / Kellerspeicher
- Last-In First-Out Prinzip (LIFO)
- Basisoperationen:
  - **push** legt Objekt auf den Stapel
  - **pop** nimmt das oberste Objekt vom Stapel

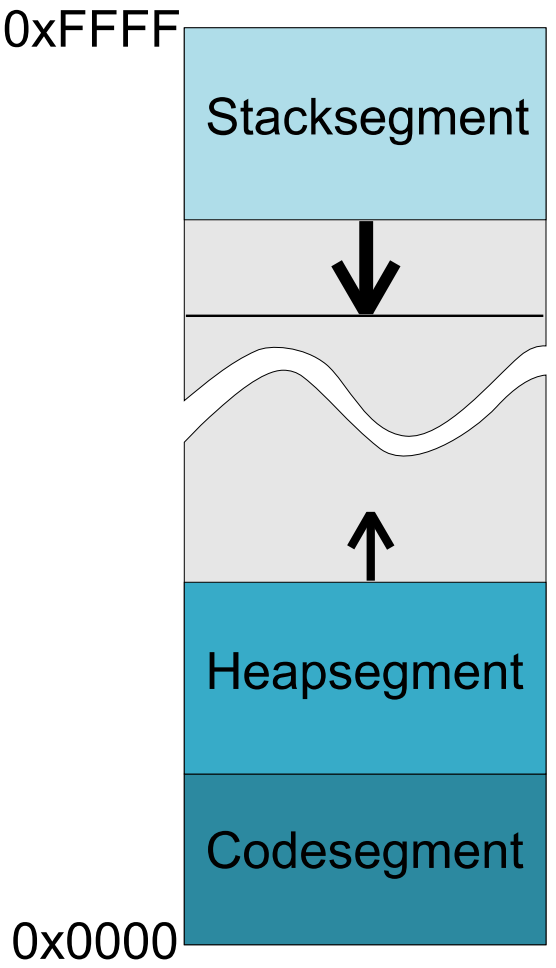


Stack in C (I)

- Ablage für
  - lokale Variablen
    - wenn nicht `malloc()` verwendet wird
  - Parameter für Funktionen
  - Rücksprungadressen für Unterprogramme
  - gesicherte Prozessorregister
    - Stichwort: *Caller save* / *Callee save*

Stack in C (II)

- Stack beginnt am oberen Ende des Adressraums
- wächst nach unten



Funktionen & Stack

- Jede Funktion hat „eigenen“ Stack-Ausschnitt
- begrenzt durch
  - EBP *(Base Pointer)*
  - ESP *(Stack Pointer)*

- EBP → Basis des Ausschnitts
- ESP → Zeiger auf das oberste Stackelement
  - wird bei **push** & **pop** angepasst

Beispiel: Funktionen & Stack (I)

```
void function(int a, int b, int c)
{
    int* pReturn;
    // ...
}

int _tmain(int argc, _TCHAR* argv[])
{
    function (1, 2, 3);
}
```

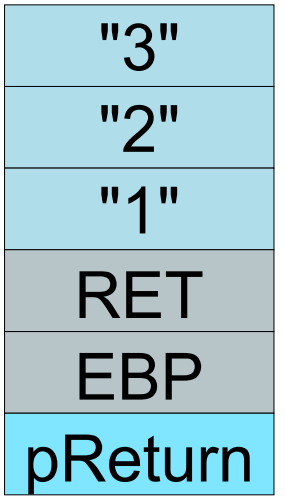
Beispiel: Funktionen & Stack (II)

```
int _tmain(int argc, _TCHAR* argv[])
{
 push         ebp
 mov          ebp,esp
 sub          esp,0C0h
 push         ebx
 push         esi
 push         edi
 lea          edi,[ebp-0C0h]
 mov          ecx,30h
 mov          eax,0CCCCCCCCh
 rep stos     dword ptr es:[edi]
    function(1,2,3);
 push         3
 push         2
 push         1
 call         function (135113Bh)
 add          esp,0Ch
}
```

```
void function(int a, int b, int c)
{
 push         ebp
 mov          ebp,esp
 sub          esp,0CCh
 push         ebx
 push         esi
 push         edi
 lea          edi,[ebp-0CCh]
 mov          ecx,33h
 mov          eax,0CCCCCCCCh
 rep stos     dword ptr es:[edi]
    int* pReturn;
    // ...
}
```

Stack

hohe Adressen



niedrige Adressen

RET → Rücksprungadresse

Techniken

Stack-Overflow

Stack-Overflow (I)

Definition

- Zu große Datenmengen werden in einen zu kleinen reservierten Speicherbereich (Puffer) im Stack geschrieben
- Speicherstellen hinter dem Puffer werden überschrieben

Stack-Overflow (II)

Ursache:

- Von-Neumann-Architektur
  - Code und Daten nicht getrennt
- C kennt keine Strings
  - Strings als `char` Arrays realisiert
- Viele C-Funktionen (z.B. `strcpy()`) kopieren Daten ohne Längenüberprüfung
- Stack und Puffer-Adressen wachsen entgegengesetzt

Stack-Overflow (III)

Möglichkeiten:

- lokale Variablen können überschrieben werden
- Änderung der Rücksprungadresse einer Funktion
- Überschreiben von Funktionspointern
- Überschreiben von Exception Handler

→ Änderung des Programmverhaltens

Stack-Overflow (IV)

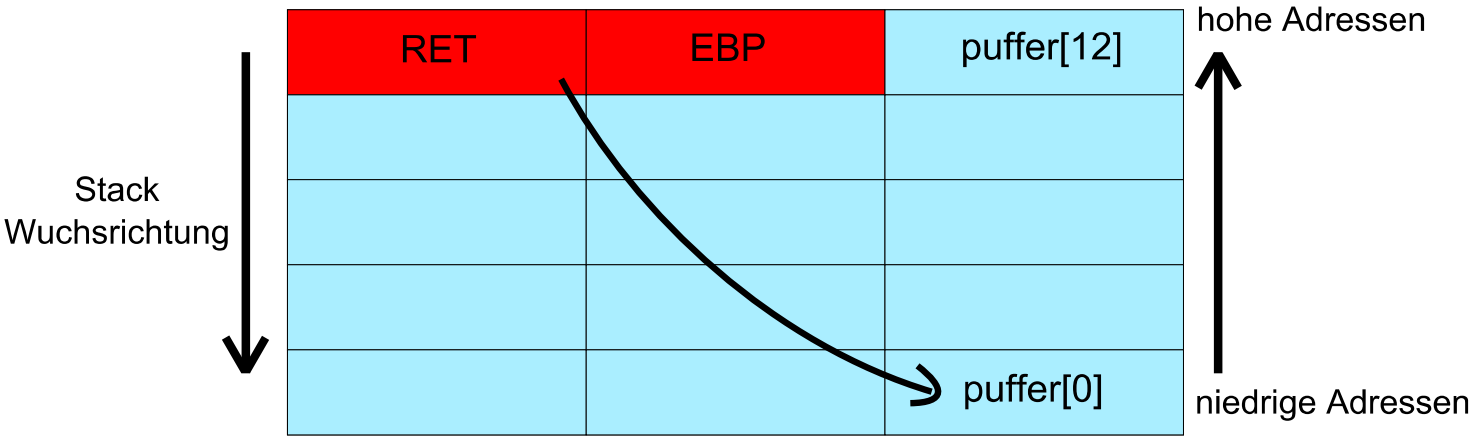
Beispiel:

Änderung der Rücksprungadresse

Stack-Overflow (V)

- Änderung der Rücksprungadresse ermöglicht:
  - Ausführung der Daten im Puffer

    → Ausführung von beliebigen Code



Stack-Overflow (VI)

- Problem:
  - „0“ markiert Ende einer Zeichenkette

    → Fremdcode darf keine „0“ enthalten

  → Verwendung äquivalenter Ausdrücke

  z.B. `mov eax, 0` → `xor eax, eax`
- Schadcode kann alle Funktionen der Windows API verwenden, die das Programm selbst einbindet
  - Mit LoadLibrary beliebige Funktionen nachladbar

Prominentes Beispiel

- „*Twilight Hack*“ für die Wii
  - durch angepasstes Savegame für das Spiel *The Legend of Zelda: Twilight Princess*
  - dem Pferd „*Epona*“ wurde langer „Name“ gegeben
    → Stack Overflow
  - beliebiger Schadcode konnte ausgeführt werden

Schutzmaßnahmen

- Array-Verwendung genaustens prüfen
  - Schon das Vergessen der abschließenden „0“ bei Strings kann eine Sicherheitslücke bedeuten
- Benutze die sicheren C-Funktionen
  - Vermeide `strcpy(), strcat(), strlen()`
  - Nutze `strlcpy(), strlcat(), strnlen()`
- Überprüfen der Daten in zwei Stufen
  1. Einlesen und Länge begrenzen
  2. Formatieren und Filtern
- (Nutze dynamischen Speicher im Heap)

Techniken

Heap-Overflow

Heap-Overflow (I)

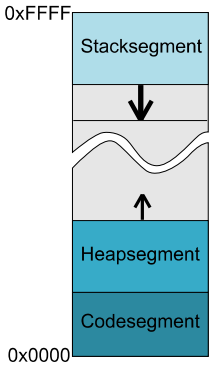
- genauer: Buffer-Overflow im Heap
- 2. Generation der Buffer-Overflow Technik
- Grund für jede 2. Sicherheitslücke in Software
- deutlich komplizierter
- meist über zu ladende Dateien realisiert
  - Bilder, PDFs, etc.

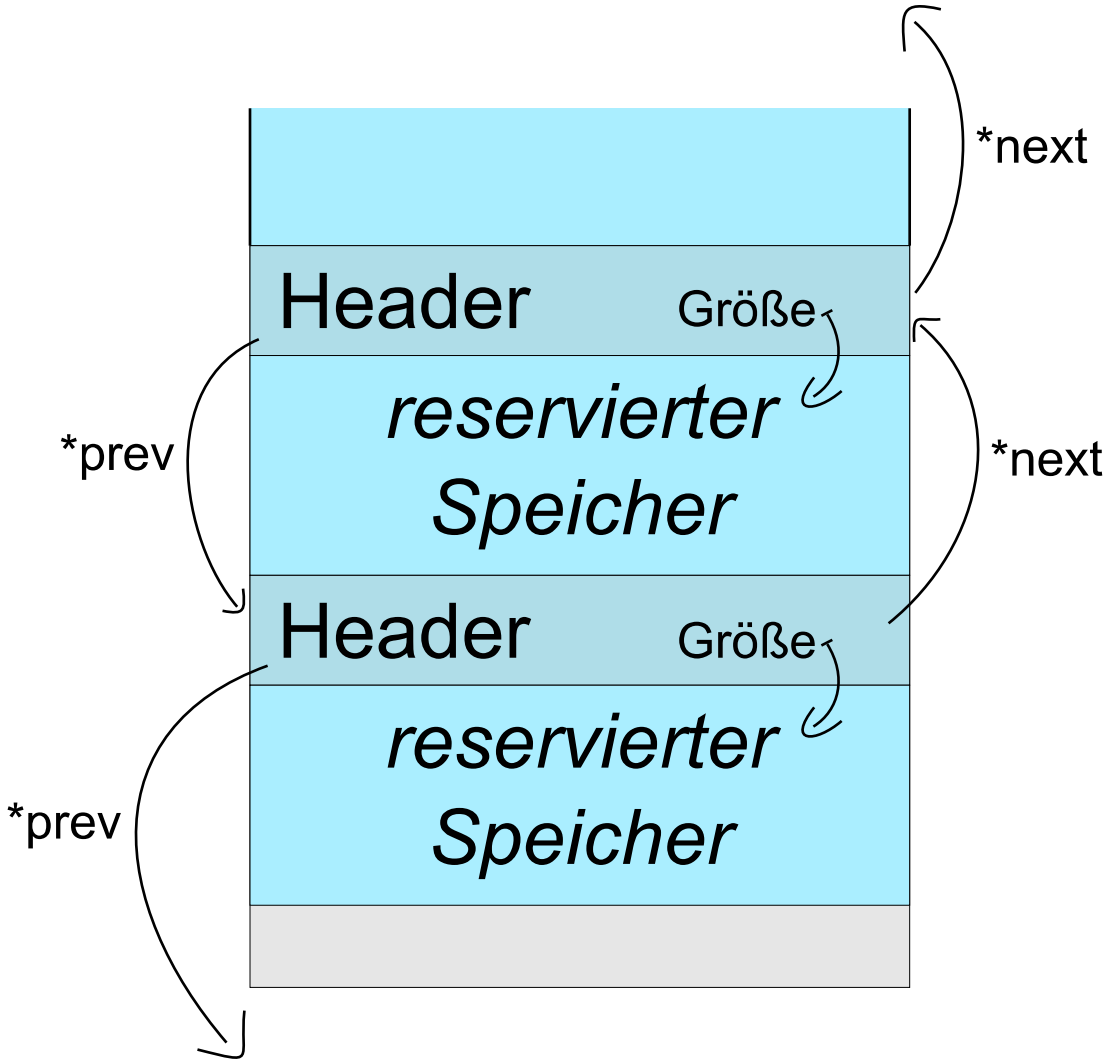
Hintergrundwissen

Heap Verwaltung (I)

- Speicher im Heap wird dynamisch reserviert
  - durch `malloc()`, `calloc()`, `realloc()`
- Verwaltung durch Systembibliothek
- Implementierung über doppelt verkettete Liste
  - Elemente bestehen aus:
    - Header für Verwaltung
    - reservierter Speicherblock
  - von Laufzeitumgebung abhängig

Heap Verwaltung (II)





Heap Verwaltung (III)

- freigewordene Speicherblöcke müssen zusammengefasst werden
  - sonst Fragmentierung

- Realisierung durch Änderung von **next* & **prev*

Heap-Overflow (II)

Definition

- Überschreiben von Speicherbereichen aus denen später eine Sprungadresse vom System geladen wird

- Ausnutzung der Defragmentierung
  - Überschreiben von *next der bestehenden Header
  - *next zeigt dann auf neuen Header mit Fremdcode
  - Zusammenlegung freier Blöcke schreibt angepasste Rücksprungadresse auf Stack

Prominente Beispiele

- Nachfolger des *„Twilight Hack“* : Bannerbomb
  - Nutzt Schwachstelle in der JPEG Dekomprimierung
- iPhone Jailbreak
- PS3 Exploit

Schutzmaßnahmen

- Konsistenz Check der einzulesenden Daten
- *GNU libc* prüft Pointer Konsistenz
- Ausführung von Heap-Speicher verhindern
  - Data Execution Prevention des BS
  - Address Space Layout Randomization
    - vergibt zufällige Adressbereiche

Zusammenfassung

Zusammenfassung

- Stack-Overflow
  - Ausnutzung der Vermischung von Programmdaten und Verwaltungsinformation (*RET-Pointer*)
  - durch Pufferüberlauf auf dem Stack
    → Änderung der Rücksprungadresse
  - Anpassung des Programmverlaufs / Ausführung Schadcode

- Heap-Overflow
  - 2. Generation des Stack-Overflows
  - Ausnutzung der Heap-Verwaltung & -Defragmentierung
  - System lädt schädliche Rücksprungadresse eigenständig

Quellen

- Smashing The Stack For Fun And Profit

  http://www.phrack.org/issues.html?id=14&issue=49

- Vermeiden von Sicherheitslöchern beim Entwickeln einer Applikation - Teil 3: Buffer Overflow

  http://www.linuxfocus.org/Deutsch/May2001/article190.shtml

- w00w00 on Heap Overflows
  - http://packetstormsecurity.org/files/view/13877/w00w00-heap-overflows.txt
- Eingelocht: Buffer-Overflows und andere Sollbruchstellen
  - http://www.heise.de/security/artikel/Eingelocht-270148.html
- Buffer overflow
  - http://en.wikipedia.org/wiki/Buffer_overflow
- Ein Haufen Risiko
  - http://www.heise.de/security/artikel/Ein-Haufen-Risiko-270800.html