Script **generated by TTT**

Title: Grundlagen_Betriebssysteme (30.01.2013)

Date: Wed Jan 30 13:15:00 CET 2013

Duration: 45:00 min

Pages: 14

Beispiel: Pufferüberlauf

Durch einen künstlich herbeigeführten Pufferüberlauf kann ein Angreifer die Ausführung seines eigenen Programms veranlassen und oft auch noch die Systemadministrator-Berechtigung (root) erlangen.

Hintergrund

Veränderung der Rücksprungadresse

Gegenmaßnahmen

- Unterscheidung
  - sichere Programmierung.
  - Maßnahmen zur Übersetzungszeit.
  - Maßnahmen zur Laufzeit.

Die meisten C-Compiler und Laufzeitsysteme überprüfen nicht die Einhaltung der Feldgrenzen.

```
int i;
char c[256];
i = 12000;
c[i] = 0;
```

String Library in C

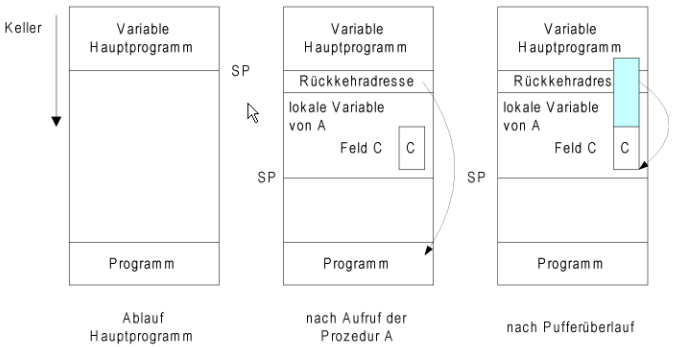
Implementierung der Unix Funktion `gets` (get string from stdin)

- keine Möglichkeit zur Spezifikation der Anzahl der zu lesenden Zeichen

```
char *gets(char *dest) {
    int c = getc();
    char *p = dest;
    while (c != EOF && c != '\n') {
        *p++ = c; c = getc();
    }
    *p = '\0';
    return dest;
}
```

- ähnliche Probleme auch bei anderen Unix Funktionen
  - strcpy: kopiert einen String beliebiger Länge
  - scanf, fscanf, sscanf: mit %s Konvertierungsspezifikation.

Angreifbarer Buffer Code



Falls das attackierte Programm mit root-Berechtigung (setuid root in Unix) abläuft, läuft das aufgerufene Programm im Puffer auch mit root-Berechtigung.

Angreifer kann seiner aufrufenden Shell root-Berechtigung verleihen.

Durch einen künstlich herbeigeführten Pufferüberlauf kann ein Angreifer die Ausführung seines eigenen Programms veranlassen und oft auch noch die Systemadministrator-Berechtigung (root) erlangen.

Hintergrund

Veränderung der Rücksprungadresse

Gegenmaßnahmen

Unterscheidung

sichere Programmierung.

Maßnahmen zur Übersetzungszeit.

Maßnahmen zur Laufzeit.

Sicherheit in Rechensystemen

Fragestellungen

Dieser Abschnitt behandelt die Sicherheitsproblematik in zentralen Rechensystemen. Dazu werden verschiedene Schutzmechanismen auf Betriebssystemebene vorgestellt.

Zugriffsschutz in Rechensystemen.

Schutzmatrix, insbesondere Zugriffskontrolllisten und Capability-Listen.

Mobiler Code.

Motivation

Schutzmechanismen

Mobiler Code

Für einen Schutzmechanismus gelten die folgenden Anforderungen

alle Objekte eines Systems müssen eindeutig und fälschungssicher identifiziert werden.

externer Benutzer eines Systems muss eindeutig und fälschungssicher identifiziert werden ⇒ Authentifizierung.

Zugriff auf Objekte sollte nur über zugehörige Objektverwaltung geschehen.

Zugriff auf Objekte nur, wenn Zugreifer die nötige Rechte hat.

Rechte müssen fälschungssicher gespeichert werden; Weitergabe von Rechten darf nur kontrolliert erfolgen.

Prinzip der minimalen Rechte.

grundlegenden Schutzmechanismen sollen ohne großen Aufwand überprüft werden können.

Man unterscheidet die folgenden Ebenen des Zugriffsschutzes.

Maschinenschutz : Kontrolle des physischen Zugangs zum Rechensystem.

Zugangskontrolle : Kontrolle des logischen Zugangs zum Rechensystem, d.h. Ausführung von Aufträgen im Rechensystem.

Berechtigungskontrolle : Kontrolle des Benutzerzugriffs auf einzelne Datenbestände und die Ausführung einzelner Dienste.

Systemschutz : Gewährleistung der Integrität der Schutzmechanismen.

Schutz von gespeicherter Information vor Diebstahl, unerwünschter Manipulation und Verletzung der Vertraulichkeit ist ein zentrales Anliegen in allen Mehrbenutzersystemen.

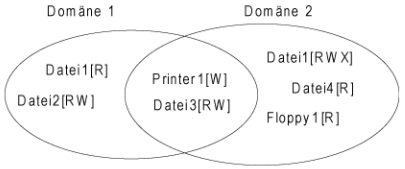
Anforderungen

Ebenen des Zugriffschutzes

Schutzmatrix

Authentifizierung

Definition: Eine **Schutzdomäne** ist eine Menge von (Objekt, Rechte) Paaren.



R = read, W = write, X = execute

Verknüpfung eines Prozesses mit einer Schutzdomäne.

zu jedem Zeitpunkt wird ein Prozess in einer Schutzdomäne ausgeführt.

Beispiel Unix: bei Ausführung eines Systemaufrufs wechselt der Prozess vom Benutzermodus in den Systemmodus ("kernel mode") ⇒ entspricht einem Wechsel der Schutzdomäne.

Das Paar (Prozess P, Schutzdomäne D) wird als ***Subjekt*** bezeichnet.

Der Zugriffswunsch eines Subjektes S auf ein Objekt o ist definiert als (D, o, a), wobei D die Schutzdomäne und a die Zugriffsart ist.

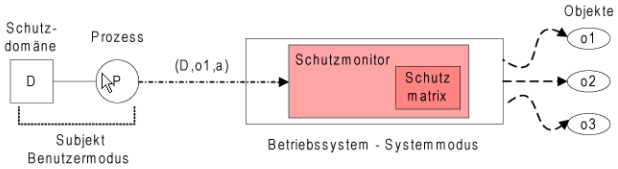
Matrix-Datenstruktur

Konzeptuell verwendet ein Betriebssystem eine Matrix-Datenstruktur, um die Zuordnung Objekt-Schutzdomäne zu verfolgen.

	Objekt					
Domäne	Datei1	Datei2	Datei3	Datei4	Printer1	Floppy1
1	read	read write	read write		write	
2	read write execute		read write	read	write	read

Schutzmonitor

Jeder Zugriff (D, o, a) eines Subjektes S wird mit Hilfe eines Schutzmonitors überprüft.



der Schutzmonitor ist vertrauenswürdig.

Subjekte können in keinem Fall auf Objekte unter Umgehung des Schutzmonitors zugreifen.

neue Prozesse müssen sich gegenüber dem Schutzmonitor authentifizieren.

Konzeptuell verwendet ein Betriebssystem eine Matrix-Datenstruktur, um die Zuordnung Objekt-Schutzdomäne zu verfolgen.

	Objekt					
Domäne	Datei1	Datei2	Datei3	Datei4	Printer1	Floppy 1
1	read	read write	read write		write	
2	read write execute		read write	read	write	read

Das Konzept der Schutzmatrix wurde von B. Lampson eingeführt. Es verknüpft Schutzdomänen mit den zu schützenden Objekten.

Schutzdomänen

Schutzmonitor

Schutzmatrix ist typischerweise sehr groß und dünn besetzt ⇒ eine direkte Implementierung ist deshalb nicht sinnvoll.

Zugriffskontrollliste

Capability-Liste

Zusammenfassung: Zugriffskontrolllisten und Capability-Listen haben in gewisser Weise komplementäre Eigenschaften

ACLs erlauben das selektive Zurücknehmen von Rechten.

Capabilities können weitergegeben werden.