Prof. Dr. Dieter Kranzlmüller
Dr. Nils gentschen Felde
Dr. Karl Führlinger
Stephan Reiter
Christian Straube

IT-Sicherheit

Workshop im Rahmen des Informatik-Probestudiums 2012

Überblick/Agenda

- Tag 1 (Fr):
  - Grundlagen der technischen Informatik mit Übungen
  - Einführung in die Bedienung von Linux
  - Übungen
- Tag 2 (Mo):
  - Übungen
  - HPC und paralleles Rechnen
- Tag 3 (Di): Grundlagen der Computergrafik
  - POV-Ray
  - Ausblick und Demo
- Tag 4 (Mi): Grundlagen der IT-Sicherheit
  - Grundlagen der IT-Sicherheit mit Übungen
  - Demonstration "Passwörter unter Windows knacken"

Zeitplan für heute

- 10:00 – ca. 11:15 Uhr:
  - Grundlagen der IT-Sicherheit
  - Demonstration: Passwörter knacken unter Windows
- kurze Pause
- Bis ~13:00 Uhr
  - Übungen zur IT-Sicherheit
    - Firewalls
    - Passwörter knacken unter Linux
  - Weitere Arbeit an den Übungen zu
    - Shell-Programmierung
    - Schaltungsentwurf
- Nachmittags: Abschluss im Plenum
  - Raum B001
  - Vorstellung der Projektergebnisse

Angriffe & „Malware“

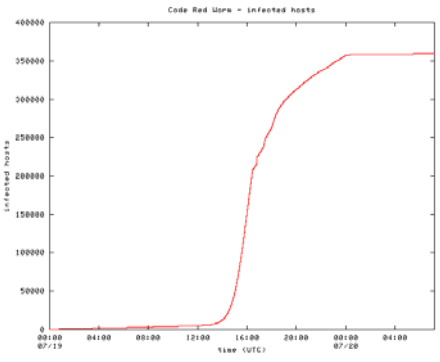
- Malware oder auch Schadsoftware
  - Speziell entwickelt um Schaden anzurichten
  - Viren, Würmer, Trojaner
  - Nutzt oftmals sog. „Buffer Overflows“
- Möglichkeiten zur Klassifikation eines Angriffs:
  - Unterscheidung nach Ausgangsort des Angriffs
    - Intern:
      - Angriffe auf ein System aus dem internen Netz
      - Angreifer…
        - » … muss keine Firewalls mehr passieren
        - » … kann interne Dienste nutzen
      - Schwer zu erkennen → Intrusion Detection Systeme (IDS)
    - Extern:
      Angriffe aus einem nicht vertrauenswürdigen Netz
  - Unterscheidung nach Art des Angriffs
    - Passiv bzw. automatisch
    - Aktiv bzw. manuell

Passive Angriffe

- Viren
  - Verbreiten sich **nicht** selbstständig
  - Verbreitung passiv
    - Email
    - Diskette
    - …
- Würmer
  - Nützen Software-Schwachstellen zur Ausbreitung
  - Suchen automatisch nach neuen Angriffszielen
  - Diverse Ausbreitungsstrategien denkbar
- Trojanische Pferde
  - Tarnung als harmloses Programm, Dokument o.ä.
  - Eingebaute Schadensroutinen

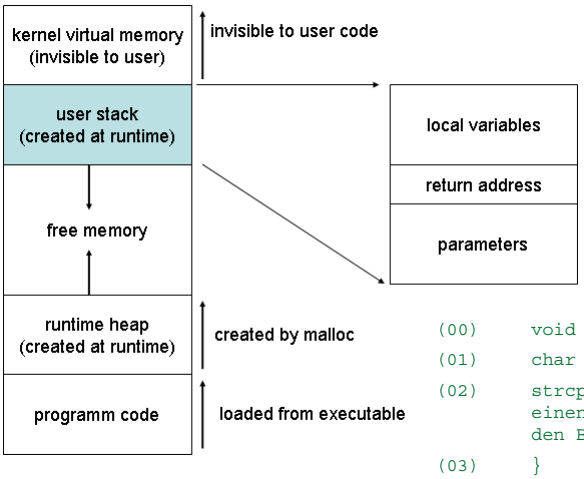
CodeRed v2

- Exemplarisch, da…
  - gut erforscht
  - effiziente Ausbreitungsstrategie
- Nutzt Sicherheitslücke im Microsoft IIS
- Ausbreitungsstrategie:
  - Aktiver Wurm
  - Wurminstanz generiert zufällig IP-Adressen
  - Versuch entsprechendes System zu infizieren:
    >359.000 infizierte Hosts in <14 Stunden
- Ziel:
  distributed DoS Angriff gegen www1.whitehouse.gov



[CAIDA, http://www.caida.org]

Buffer Overflows




```
(00)    void bufferOverflow() {
(01)    char buffer[16];
(02)    strcpy(buffer, "Ich kopiere
        einen zu langen String in
        den Buffer!");
(03)    }
```


Beispiel: Bufferoverflow

```
#include <stdio.h>

int test() {
   int i[1] = {42};
   i[2] += 32;
   printf("test: i[0] = %i\n\n", i[0]);
   return i[0];
}

int main() {
   int x = test();
   printf("Dies ist ein Testprogramm.\n");
   printf("Es demonstriert einen Puffer-Überlauf.\n\n");
   x++;
   printf("main: x = %i\n", x);
   return 0;
}
```

Aktive Angriffe

- Vorsätzliche und kontrollierte Angriffe
- Beispiele
  - Session Hijacking:
    Übernahme bereits autorisierter Verbindungen
  - DNS-Spoofing
  - IP-Spoofing
  - Dictionary Attacks auf Passwörter von Nutzern
  - TCP Sequence Prediction
  - Denial of Service (DoS)
  - …
- Nach erfolgreichem Angriff werden meist Backdoors installiert

Denial of Service (DoS)

- Meist aktiver Angriff
- Ziel
  - Blockieren von Diensten
  - Außerbetriebsetzung von Systemen
- Nutzt i.d.R. Fehler in Programmen (z.B. Buffer Overflows) oder Betriebssystemen
- Beispiele
  - Ping of Death
  - (Win)Nuke
  - Snork
  - Teardrop
  - SYN-Flood
  - …
- Angriff durch Vielzahl von Rechnern
  → Distributed Denial of Service (DDoS)

Paketfilter-Firewalls

Konfigurationsansätze

- „Es ist alles erlaubt, was nicht explizit verboten ist“
  - Sog. „Blacklist“
  - Sehr benutzerfreundlich
  - Problematik: vergessene Kommunikationsbeziehungen belieben erlaubt
- „Es ist alles verboten, was nicht explizit erlaubt ist“
  - Sog. pessimistischer Ansatz
    oder „Whitelist“
  - Genau definierte Kommunikationsbeziehungen werden explizit freigeschaltet
  - Letzte Zeile im Regelwerk lautet immer „deny all“

Paketfilterung

- Ähnlich einem IP-Router
- Filtert Pakete anhand vorgegebenem Regelwerk
- Arbeitet typischerweise auf den OSI-Schichten 3 & 4
- Mögliche Paket-Eigenschaften zur Filterung:
  - IP-Adresse: Quell- bzw. Ziel-Adresse
  - Protokoll-ID: TCP, UDP, ICMP
  - Flags: für korrekten Verbindungsauf- bzw. -abbau
  - Ports: Quell- bzw. Ziel-Ports (z.B. für HTTP, SSH, SMTP, …)
  - ICMP-Code
  - …
- Durch TCP-Flags ist u.a. Richtung des Verbindungsaufbaus zu erkennen

Das Regelwerk

- Regeln (rules) definieren Umgang mit Paketen
- Gängig: Abarbeitung der Regeln „von oben nach unten“
- Möglicher Umgang mit Paketen:
  - ACCEPT
    - Paket darf passieren
    - Weiterrouten des Datagramms
  - REJECT
    - Zurückweisen des Pakets
    - Benachrichtigung des Absenders (z.B. ICMP-Meldung oder TCP-Paket mit gesetztem RST-Flag)
  - DROP
    - Zurückweisen des Pakets
    - Keine Benachrichtigung des Absenders
- Anwendung von Regeln kann protokolliert werden (LOG)
- Problematisch: Konsistenz des Regelwerks

Statische vs. Dynamische Paketfilterung

- Statische Paketfilterung
  - Zustandslos, d.h. unabhängig von vorangegangenen Paketen
  - Für jedes Paket gilt gleicher Regelsatz
- Dynamische Paketfilterung
  - Auch „Stateful Inspection“
  - Zustandsabhängig
  - Erweitert Regelwerk temporär
  - Beispiele:
    - Rückrichtung für erlaubte Verbindung zulassen
    - Überwachen der TCP-Sequenznummern

Beispiele

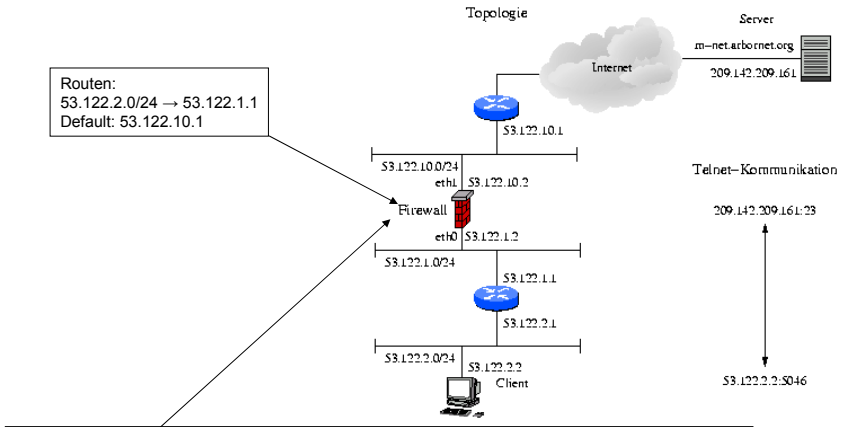
Nr.	Quelle	Ziel	Protokoll	Src-Port	Dst-Port	Flags	Action	LOG
1	Client	Server	TCP	>1023	23	any	ACCEPT	X
2	Server	Client	TCP	23	>1023	!syn	ACCEPT	-
3	any	any	any	any	any	any	DROP	X

Beispiel: Filtertabelle für Telnet bei statischer Paketfilterung

Nr.	Quelle	Ziel	Protokoll	Src-Port	Dst-Port	Action	LOG
1	Client	Server	TCP	>1023	23	ACCEPT	X
2	any	any	any	any	any	DROP	X

Beispiel: Filtertabelle für Telnet bei dynamischer Paketfilterung

Beispiel: Telnet-Kommunikation



Nr.	Quelle	Ziel	Protokoll	Src-Port	Dst-Port	Flags	Action	LOG
1	53.122.2.0/24	any	TCP	>1023	23	any	ACCEPT	X
2	any	53.122.2.0/24	TCP	23	>1023	!syn	ACCEPT	-
3	any	any	any	any	any	any	DROP	X

Network Address Translation (NAT)

- Fähigkeit Quell- und Ziel-Adressen und Ports auszutauschen
- Verbergen von physischen IP-Adressen hinter logischen
- Eingriff in die Client-Server-Kommunikation!
  - Logische Verbindung bleibt bestehen
  - Kann Probleme mit sich bringen
    - Serveradressierung
    - Verschlüsselung
    - … (siehe auch spätere Vorlesungen)
- Unterteilung
  - Statisches NAT
  - Dynamisches NAT

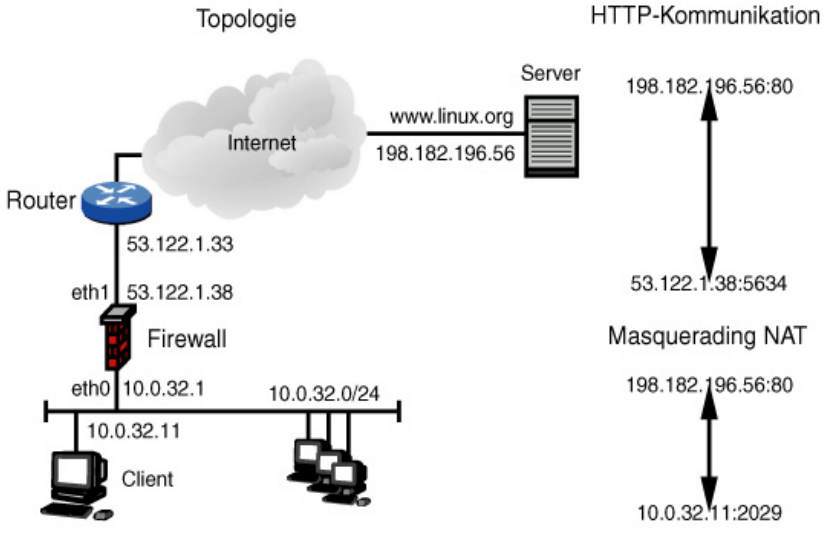
Dynamisches NAT

- Auch „IP-Masquerading“
- Verbergen eines gesamten Netzes hinter einer externen IP-Adresse
- Typisch für private Netze (z.B. DSL-Router)
- Firewall-Rechner ist dann auch Default Gateway aller internen Rechner
- Bei ausgehenden Verbindungen i.d.R.
  - Umsetzung der Quell-IP
  - Umsetzung des Quell-Ports

Nachteile von IP-Masquerading

- Verbindungen von außen nach innen sind nur sehr eingeschränkt möglich
  - Beispiel: Server im lokalen Netz
  - Lösung: Port-Mapping
- Applikationen, die auf der Verwendung fester Quell-Ports basieren, können nicht verwendet werden
- Applikationen, die nur eine bestimmte Anzahl von Verbindungen pro Quell-IP-Adresse zulassen, können zu Problemen führen
- Die Anzahl der insgesamt möglichen Verbindungen nach außen ist beschränkt

Beispiel: dynamisches NAT



Aufgaben für heute Nachmittag:

- Firewalls mit iptables
- Passwörter knacken unter Linux
- Umfrage ausfüllen:

  http://probestudium.ifi.lmu.de

John the Ripper

- http://www.openwall.com/john/
- Verschiedene Modi
  - Wordlist mode
    - Wörterbuchangriff auf Passwörter
    - Einige Wörterbücher werden mitgeliefert
  - "Single crack" mode
    - login names, "GECOS" / "Full Name" fields und users' home directory names als Passwortkandidaten
    - Variationen dazu (sog. mangling)
  - "Incremental" mode
    - Brute Force
  - External mode
    - Externe Quelle für Passwortkandidaten, die John the Ripper auf Passwortliste anwedet

Live-Demo:
Passwörter knacken unter Windows

Dr. Nils gentschen Felde

Grundlagen: Hash Funktionen

- Hash Funktion
  - x ϵ X Menge aller möglichen Eingaben
  - z ϵ Z Menge aller möglichen Hashes
  - H(x) =z Hash Funktion
  - In der Regel gilt |X|>>|Z|
    -> H(x) ist nicht injektiv
    -> H(x) ist nicht bijektiv
  - Beispiel einer Hashfunktion: Quersumme
- Kryptografische Hash Funktion
  - Zusätzliche Anforderungen:
    - preimage resistance
    - collision resistance
  - Beispiele für kryptografische Hash Funktionen:
    MD5, SHA

LM Hashes

- Verwendet in Windows 95, 98, Me, um Passwort Hashes für die Authentisierung im Netz zu speichern
- Aus Kompatibilitätsgründen ebenfalls in Windows NT, 2000, XP, Vista, … zusätzlich zum neuen NTLM Hash gespeichert
- LM Hash Algorithmus:
  1. Das Passwort des Benutzers in Form eines OEM-String wird zu Großbuchstaben umgeformt.
  2. Dieses Passwort wird entweder auf 14 Bytes mit Nullen gefüllt oder gekürzt.
  3. Das Passwort mit der festen Länge wird in zwei 7 Byte-Hälften aufgeteilt.
  4. Aus jeder Hälfte wird durch hinzufügen eines NON-Parity-BITs ein 64-BIT langer DESSchlüssel erzeugt.
  5. Jeder dieser Schlüssel wird dazu genutzt, den Konstanten ASCII-String "KGS!@#$%"zu DES-verschlüsseln, woraus zwei 8 Byte Chiffretext-Werte resultieren.
  6. Diese beiden Chiffretext-Werte werden verbunden, um einen 16 Byte-Wert zu bilden, der den LM hash darstellt.

Sicherheitsmängel des LM Verfahrens

- Komplexität:
  - Komplexität für 14-stellige Passwörter: 95^14 = 4.876.749.791.155.298.590.087.890.625 (ca. 4,9 Quadrilliarden)
  - Aufteilung in zwei Hälften: 95^7
  - Konvertierung in Großbuchstaben: 69^7 = 7.446.353.252.589 (ca. 7,5 Billionen)
- Weitere Mängel:
  - DES ist als schwach anzusehen
  - Öffentlich bekannter und zu kurzer Klartext-String bietet Angriffsfläche für Kryptoanalyse

Mögliche Angriffe

- Wörterbuch Attacken
  - Finden nur Wörter, die im Wörterbuch stehen
- Brutforce Attacken
  - Dauern sehr lange
  - Tool: „John the Ripper“
- Rainbowtable Verfahren
  - Kompromiss aus Speicherbedarf und Rechenzeit
  - Vorberechnung von Tabellen, die Chains enthalten
  - Chain kann mehrere Millionen Hashes repräsentieren
  - Gespeichert werden nur Anfangs- und Endwert der Chain
  - Findet u.U. nicht alle möglichen Passwörter
  - Tool: „ophcrack“

Let's go...