

./easter -h -egg
Düsseldorf 2002

The Hijacker's Guide to the Galaxy

Grundlagen: TCP, TCP-Hijacking, ARP-Spoofing/
-Relaying, UDP, ICMP-Redirection, geswitchte
Umgebungen, Hijacking-Angriffe auf verschiedene
Protokolle, Social-Hijacking



Stefan Krecher
Chaos Computer Club
Mail: stefan@ccc.de

| Einführung |

Hijacking in elektronischen Netzen

Infiltrieren, Übernehmen, Stören, Umlenken jeglicher client–server oder peer–to–peer Netzwerkverbindungen.

Hijacking auf verschiedenen Ebenen

- Layer–2 Hijacking
 - ARP–Spoofing
 - ARP–Relaying
- Layer–3/4 Hijacking
 - TCP–Hijacking
 - ICMP–Redirection
- Layer–5 Hijacking
 - Kompromittieren von Protokollen der Application–Layer (z.B. HTTP, FTP, POP3, DNS usw.)
- Layer–8 Hijacking: Social–Hijacking ;–)

| Einführung | Hijacking–Ziele |

- Verbindungen Übernehmen
 - Abhängen eines Kommunikationspartners
 - Aufrechterhaltung der Verbindung
 - Einschleusen von Anweisungen
- einen existierenden Rechner/ Service imitieren
 - Router, Gateway
 - Webserver, DNS–Server
- Antworten eines existierenden Rechners/
Services imitieren
 - HTTP–Redirection
 - ICMP–Redirection
 - POP3–/ DNS–Spoofing usw.
 - ...

| Einführung | Vorraussetzungen |

Hardwaremäßig

- Angreifer muß in der Lage sein, Pakete von/ für mindestens eines der Opfer empfangen zu können
- Idealerweise: Rechner sind über einen Hub oder Switch miteinander verbunden
- Netzwerkadapter des Angreifers muß in den sog. Promiscuous–Mode schaltbar sein

Softwaremäßig

- Sniffer (z.B. tcpdump, ethereal, sniffit)
- Paketgenerator (z.B. spak, ippacket, tkipconstructor)
- ggf. ARP–Flooder, RST–Daemon, SYN–Flooder
- Fertige Hijacking–Software (z.B. hunt, dsniff, juggernaut, ettercap)

TCP – Verbindungsaufbau (three-way handshake, RFC 793)

Client

Server

SYN,SEQ=1000:1000 →

← SYN|ACK,SEQ=5000:5000,
ACK=1001,WIN=100

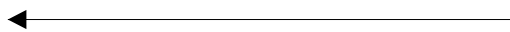
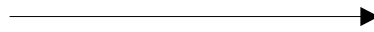
ACK,SEQ=1001:1001,
ACK=5001,WIN=100 →

TCP – Datenaustausch

Client

Server

ACK,SEQ=1001:1011,
ACK=5001,WIN=100



ACK,SEQ=5001,
ACK=1012,WIN=90

Der Client schickt 10 Byte Payload an der Server.

TCP – Beenden einer Verbindung

- Rechnerausfall
- time-out
- beidseitiger Abbau via FIN-Flag
- einseitiger Abbau via RST-Flag (s.u.)
- Beendigung als Resultat eines Hijacking-Angriffs

Client

Server

← ACK,SEQ=5001,
ACK=1012

RST=1012

→

TCP – desynchronized state

Def.: Die Sequenznummer eines eingehenden Paketes liegt außerhalb des Empfangsfensters

- Die Sequenznummer des eingehenden Paketes ist kleiner als die erwartete Sequenznummer (ACK)
- Die Sequenznummer des eingehenden Paketes ist größer als die erwartete Sequenznummer plus Größe des Empfangsfensters
- Das Paket ist nicht akzeptabel und wird gedroppt
- Es gibt einseitige und beidseitige Desynchronisation
- ACK–Storm kann resultieren
- Resynchronisation ist nur durch »Mithilfe« des Clients möglich

| TCP | Grundlagen |

TCP – desynchronized state

Client

Server

← ACK,SEQ=5001,
ACK=1015,WIN=100

ACK,SEQ=1001:1011,
ACK=5005,WIN=100 →

← ACK,SEQ=5001,
ACK=1015,WIN=100

ACK,SEQ=1001,
ACK=5005,WIN=100 →

ACK,SEQ=1001:1011,
ACK=5005,WIN=100 →

Usw. usf. ...

- ACK–Storm
- Retransmissions

Angriff: Infiltration

Def.: Desynchronisation durch Einschleusen von Daten in den Strom

- Der Angreifer schickt als Client getarnt ein Payloadpaket an den Server
- Der Server führt die Anweisung aus
- Kein direkter Datenaustausch möglich, Verbindung ist desynchronisiert
- Vermittlung/ Paket-forwarding durch den Angreifer
- light-variante: simple/ simplex hijack
- ultra-light: »Take-no-prisoners«
- (early desynchronization)

| TCP | Angriffe |

Infiltration

Client

Server

← ACK, SEQ=5000,
ACK=1000

Client(Hacker)

Server

ACK|PSH, SEQ=1000:1010, →
ACK=5000

← ACK, SEQ=5000,
ACK=1011

Client

Server

ACK|PSH, SEQ=1000:1005, →
ACK=5000

← ACK, SEQ=5000,
ACK=1011

Take–No–Prisoners

Def.: Einschleusen von Anweisungen, ohne Rücksicht auf Verluste, keine Vertuschung.

- simple hijack, simplex hijack
- Effektive Anweisung einfügen (no second chance)
- Schnelles Handeln nach dem Angriff
- Leichtgläubige, reaktionsschwache Opfer
- Angriff auch nur mit Sniffer und Paketgenerator möglich

Das RST–Mysterium

- Gründe für RST (nach RFC 793)
 - Antwort auf Pakete einer nicht existierenden Verbindung
 - Antwort auf das Acknowledgen von etwas noch nicht Gesendetem
 - unpassender security level/ compartment/ precedence
 - Firewall
- »Simple Active Attacks Against TCP« beschreibt einen Hijacking–Angriff mittels »RST–Reopen«
- Berichte über sich wiederaufbauende Connections nach einem RST–Paket
- Protokollstacks verhalten sich unterschiedlich/ merkwürdig

| TCP | Angriffe |

Nach dem Angriff

- Abhängen des Clients mit FIN oder RST (vor oder nach dem Angriff)
- den Client »benachrichtigen«
- den Client bewegungsunfähig machen (RST–Daemon)
- Verbindung resynchronisieren
- weglaufen

| TCP | Angriffe |

Beispiel: einfache Infiltration einer Telnet-Session

- Netzwerkadapter in promiscuous-mode schalten
- tcpdump starten
- Notwendige IP-Adressen, Port- und Sequenznummern herausuchen

```
Eterm Eterm-0.8.10
joshua:/home/stefan#
joshua:/home/stefan# ifconfig eth0 promisc
joshua:/home/stefan# tcpdump -i eth0 -S -n -t src port 23 or dst port 23
tcpdump: listening on eth0
192.168.0.23.1039 > 192.168.0.1.23: P 1054711469:1054711470(1) ack 4223612893 wi
n 17520 (DF) [tos 0x10]
192.168.0.1.23 > 192.168.0.23.1039: P 4223612893:4223612894(1) ack 1054711470 wi
n 5840 (DF)
192.168.0.23.1039 > 192.168.0.1.23: . ack 4223612894 win 17520 (DF) [tos 0x10]
192.168.0.23.1039 > 192.168.0.1.23: P 1054711470:1054711471(1) ack 4223612894 wi
n 17520 (DF) [tos 0x10]
192.168.0.1.23 > 192.168.0.23.1039: P 4223612894:4223612895(1) ack 1054711471 wi
n 5840 (DF)
192.168.0.23.1039 > 192.168.0.1.23: . ack 4223612895 win 17520 (DF) [tos 0x10]
192.168.0.23.1039 > 192.168.0.1.23: P 1054711471:1054711473(2) ack 4223612895 wi
n 17520 (DF) [tos 0x10]
192.168.0.1.23 > 192.168.0.23.1039: P 4223612895:4223612897(2) ack 1054711473 wi
n 5840 (DF)
192.168.0.23.1039 > 192.168.0.1.23: . ack 4223612897 win 17520 (DF) [tos 0x10]
192.168.0.1.23 > 192.168.0.23.1039: P 4223612897:4223613103(206) ack 1054711473
win 5840 (DF)
192.168.0.23.1039 > 192.168.0.1.23: . ack 4223613103 win 17520 (DF) [tos 0x10]
```

| TCP | Angriffe |

Beispiel: einfache Infiltration einer Telnet-Session

- Paketgenerator starten (perl tkipconstructor.pl)
- IP-Adressen, Port- und Sequenznummern eintragen
- möglichst 1337'en Infiltrationsstring eingeben
- Paket abschicken



| ARP | Übersicht |

- Physikalische Adressierung im lokalen Netz über das Address Resolution Protocol (ARP)
- Identifizierung der Netzwerkadapter durch MAC-Adressen (Medium Access Control)
- Versand von Daten die in Ethernet-Frames gekapselt und an MAC-Adressen adressiert sind
- Umsetzung von IP-Adressen auf MAC-Adressen mit ARP-Requests (Broadcasts) und -Replies
- Der Protokollstack hält im ARP-Cache die ihm bekannten MAC-Adressen vor
- MAC-Table-Overflow im Switch mit macof möglich, ARP-Cache-Angriff mit gespoofen ICMP-Echorequests und gespoofen ARP-Replies
- Bei einem Request wird die eigene MAC-Adresse gleich mitgegeben
- Switches merken sich welche MAC-Adresse zu welchem Port gehört

| ARP | Spoofing |

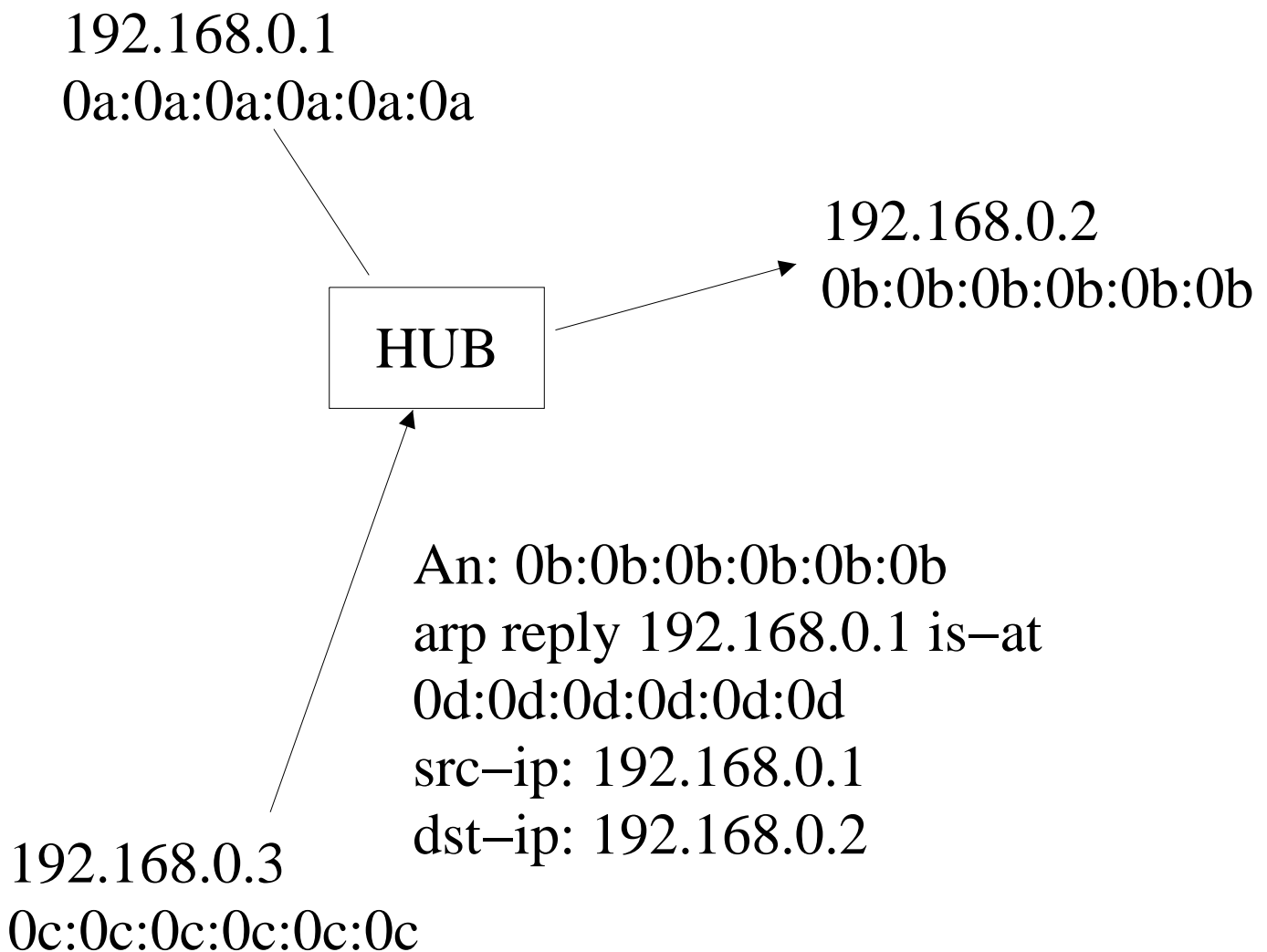
Fälschen der eigenen MAC–Adresse

```
ifconfig eth0 down  
ifconfig eth0 hw ether 0a:0a:0a:0a:0a:0a  
ifconfig eth0 192.168.0.23  
ifconfig eth0 up
```

- Bekanntmachen der Adresse z.B. durch ICMP–echo–requests (ping) an einen Rechner oder Broadcast (die angepingten Protokollstacks vermerken für die entsprechende IP–Adresse die neue MAC–Adresse)
- Übernahme einer strategisch wichtigen Position im Netz (Gateway)
- Betrügen eines Switches
- Stören einer bestehenden Verbindung

| ARP | Spoofing |

Spoofen fremder MAC–Adressen



- 192.168.0.3 generiert gefälschten ARP–Reply
- 192.168.0.2 ändert den Eintrag im ARP–Cache
- 192.168.0.1 ggf.: »0d:0d:0d:0d:0d:0d is using my ip«

| ARP | Spoofing |

Spoofen fremder MAC-Adressen mit spak

```
makearp -di 192.168.0.2 -si 192.168.0.1  
-sm 0d:0d:0d:0d:0d:0d -dm 0b:0b:0b:0b:0b:0b  
-op 2 | makeeth -s 0d:0d:0d:0d:0d:0d  
-d 0b:0b:0b:0b:0b:0b -i - -t 0x806  
| sendeth -i -
```

spak (Send PAcKet) hat Tools, um ip, tcp, udp, arp und ethernet Pakete zu generieren und zu verschicken

Die Pakete können aufeinander aufbauend via Pipes miteinander verbunden und verschickt werden

| ARP | Spoofing |

Geswitchte Umgebung

192.168.0.1

0a:0a:0a:0a:0a:0a

192.168.0.2

0b:0b:0b:0b:0b:0b

SWITCH	
port	mac
0	0a:0a:0a:0a:0a:0a
1	0b:0b:0b:0b:0b:0b
2	?

192.168.0.3

0c:0c:0c:0c:0c:0c

2

Beim ersten arp-reply oder connect/ transfer-Versuch von/ zu 192.168.0.3 merkt sich der Switch die MAC-Adresse

| ARP | Spoofing |

ARP-Relaying (1)

Ziel: Sniffen/ Hijacking in geswitchten Umgebungen

Variante 1:

- Spoofen der fremden MAC-Adressen mit der Adresse des Angreifers
- alle Pakete kommen beim Angreifer an
- Der Angreifer muss weitervermitteln (ip_forwarding)
- Bsp. dsniff, der Angreifer hat die IP 192.168.0.3:
echo '1' > /proc/sys/net/ipv4/ip_forward
arp spoof -i eth0 -t 192.168.0.1 192.168.0.2
(alle Pakete von 192.168.0.1 für 192.168.0.2 kommen bei 192.168.0.3 an und werden von dort via Kernel-ip_forwarding weitervermittelt)
- Der Angreifer kann alle Pakete sehen und normal sniffen oder hijacken

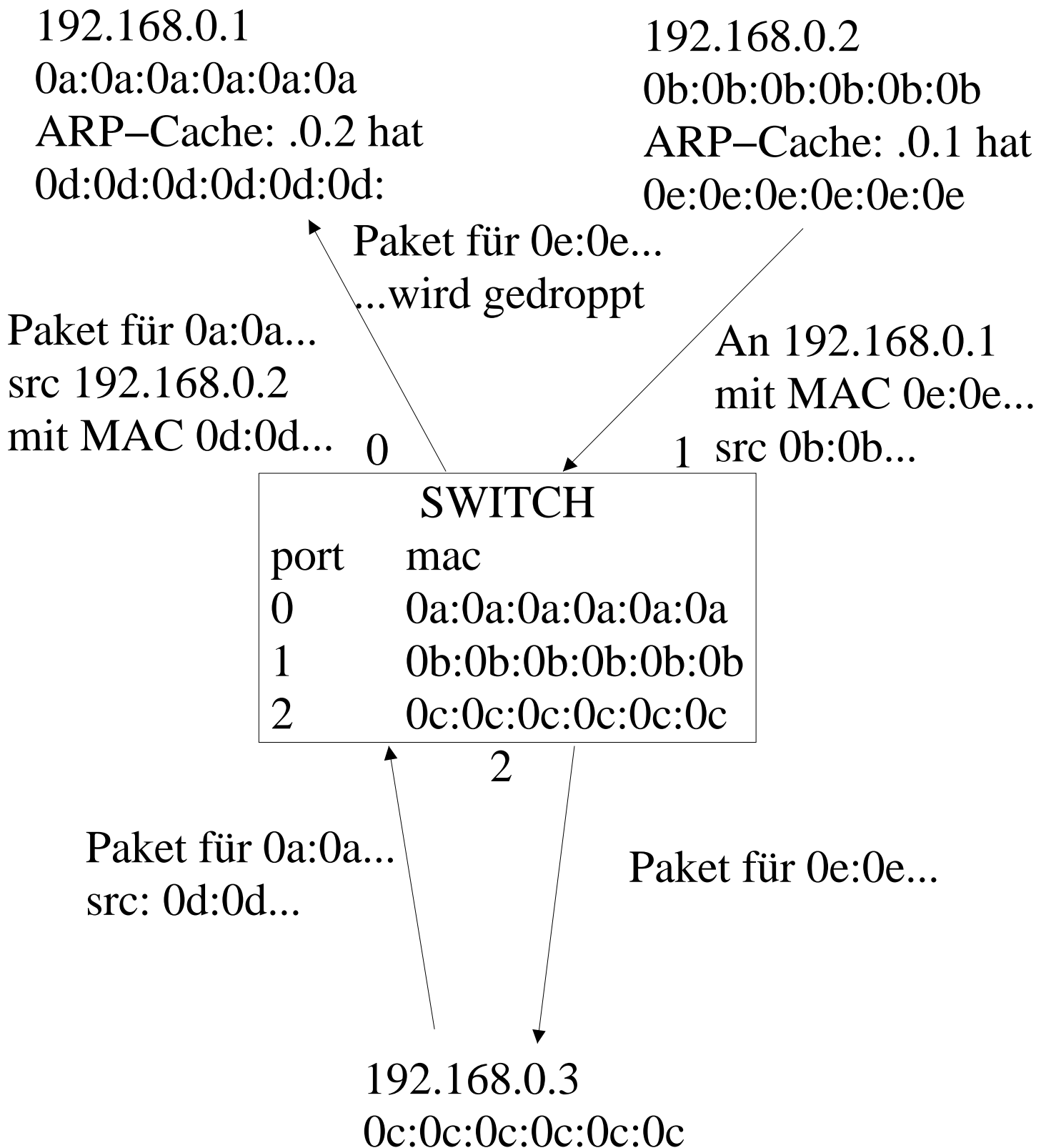
ARP–Relaying (2)

Variante 2:

- Spoofen der fremden MAC–Adressen mit nicht existierenden Adressen
- Ein Paket für die jeweilige Gegenseite kommt bei allen angeschlossenen Rechnern an, da die Ziel–MAC–Adresse nicht bekannt ist.
- Das Paket wird überall gedroppt
- Der Angreifer baut das Paket um und schickt es mit der (nicht existierenden) Absender–MAC an den vermeintlichen Empfänger
- Diese Variante ist weniger auffällig
- z.B. hunt: arp–spoof/ relay daemon

| ARP | Spoofing |

ARP-Relaying, Variante 2



ICMP–Überblick

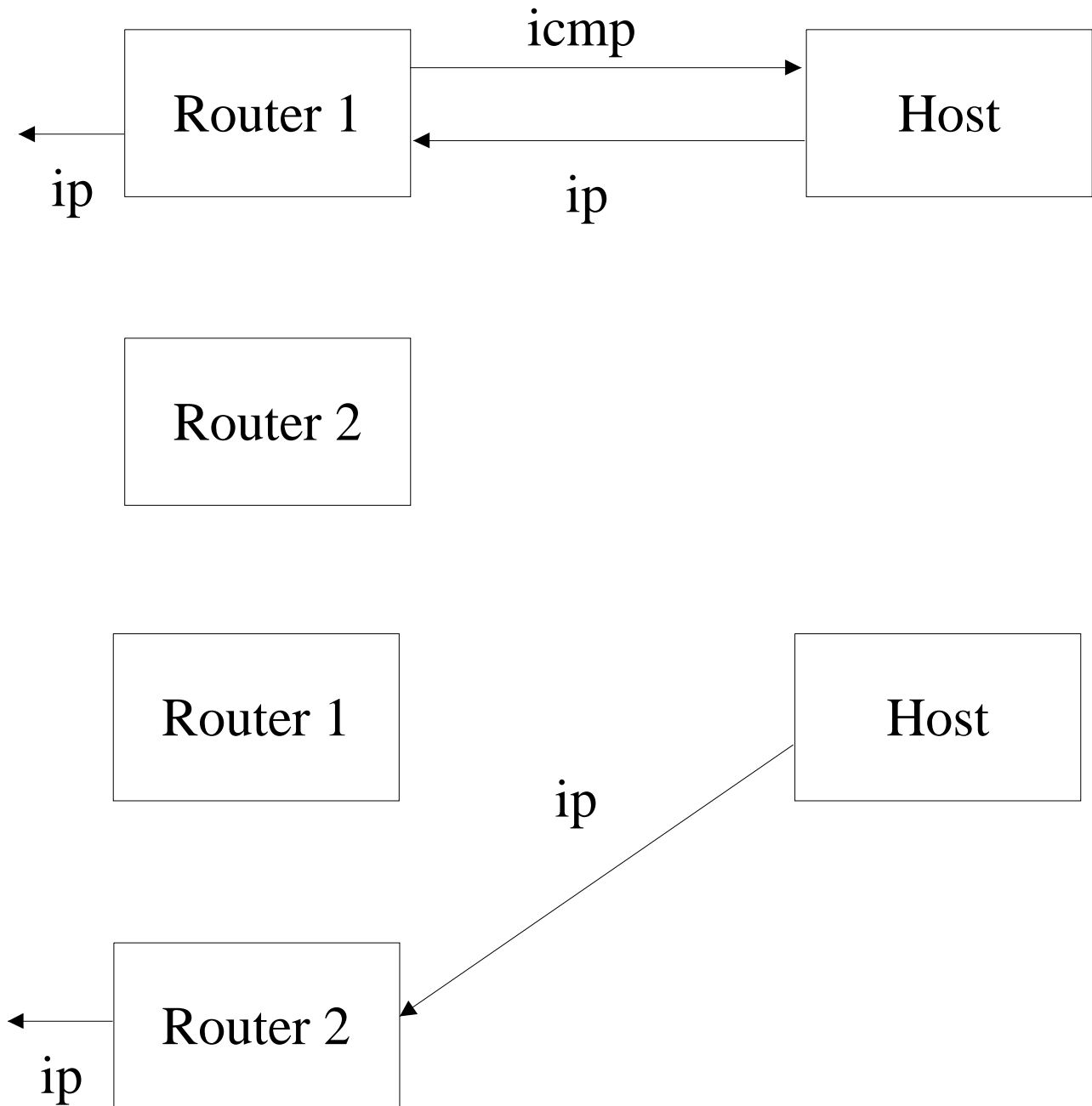
- Internet Control Message Protocol, RFC 792
- Statusmeldungen über Netzzustand
- Versand in einem IP–Paket
- Unterschiedliche Statusmeldungen – z.B.
 - Echo Reply (Ping)
 - Destination unreachable
 - Source Quench
 - Redirect
 - Timestamp
 - Router Advertisement
 - ...
- Router broadcasts
- Linux ICMP bug
 - Daten aus freigegebenem Speicher werden vom Kernel u.U. in ein TTL–Exceeded–Paket gepackt
 - 2.2er Kernel bis Version 2.2.18

| ICMP | Redirection |

Redirection

- Redirect–Nachricht: ICMP–Code 5
- Nachricht von Routern an Hosts, das Pakete für ein bestimmtes Ziel besser über einen anderen Router/Gateway verschickt werden sollen
- Der Router leitet empfangene Pakete trotzdem zum Ziel weiter
- der Host ändert seine System–Routing–Table (wenn er ICMP–Code 5 Nachrichten annimmt)
- die Routen expiren nicht (Vorteil gegenüber ARP–Cache–Poisoning)
- Theoretisch: attack from anywhere
- icmp_redir.c (»Playing redir games with ARP and ICMP«, yuri volobuev)

| ICMP | Redirection |



Das ICMP-Paket kann problemlos von einem Angreifer immitiert werden

HTTP–Hijacking

- Eine bestehende Verbindung entführen und Passwörter loggen oder Eingaben manipulieren
- Sich beim Verbindungsaufbau als Zielrechner ausgeben
- HTTP–Requests mit gefälschten HTTP–Replies umlenken
 - HTTP–GET–Requests identifizieren
 - Beantworten mit HTTP–Statusmeldung »301« (Moved Permanently) und der neuen Location
 - z.B. httpredir

POP3–Hijacking

- Nachrichten manipulieren, löschen
- Standard–Mails verschicken

| **UDP** | Überblick |

- User Datagram Protocol, RFC 768
- »besseres IP–Paket«, Header: Ports, Length, Checksumme
- connectionless, kein Handshake
- Handhabung/ Programmierung wie bei TCP (Sockets)
- Verwendet u.a. für NFS, streams, SNMP, RIP, DNS, Netbios
- Server kann mehr Anfragen bedienen
- Spoofing sehr einfach
- SMB–Hijacking

| Layer 5 | ausgewählte Angriffe |

DNS-Spoofing

- Fälschen von Antworten auf eine DNS-Query (Paketkonstruktion trivial)
- UDP/ TCP
- Die richtige Antwort vom DNS-Server kommt auch, aber meist zu spät
- Bsp. dsniff: dnsspoof
 - Verwendung eines Hostfiles möglich
 - Mit pcap-Filter-Expressions kann der Spoofing-Angriff auf bestimmte Services beschränkt werden

HTTPS/SSH-1 man-in-the-middle

- Angreifer reagiert auf https-Anfragen, schickt sein eigenes Zertifikat an den Client und baut selber eine Client-Verbindung zum https-Server auf
- Der Angreifer vermittelt nun, und kann die Verbindungsdaten entschlüsseln

| Layer 5 | ausgewählte Angriffe |

Szenario: HTTPS–Hijacking mit DNS–Spoofing und ARP–Relaying in einer geswitchten Umgebung

192.168.0.1 ist Router/ Gateway

192.168.0.2 ist der ahnungslose Surfer

192.168.0.3 der Angreifer

Verbindung zwischen 192.168.0.1 und 2
»sichtbar« machen:

- hunt starten, mac–discovery–daemon starten
- ...0.1 und ...0.2 anpingen um deren
MAC–Adressen zu erfahren
- mac–discovery–daemon stoppen
- »add host to host arp spoof« wählen, ...0.1 und
...0.2 als zu spoofende hosts angeben, imaginäre
MAC–Adressen angeben, relayer–daemon
starten, wir können nun alle Pakete sehen

DNS–Spoofing

- hostfile für dnsspoof erstellen, z.B.
»192.168.0.3 *« alle Anfragen mit ...0.3
beantworten
- dnsspoof –f hostfile

| Layer 5 | ausgewählte Angriffe |

Szenario, Teil 2

Man-In-The-Middle

- webmitm starten
- ein neues Zertifikat wird generiert (bzw. ein Vorhandenes benutzt)
- Wenn der Client jetzt eine https-Seite aufruft, erscheint der Hinweis auf ein neues Zertifikat, wenn der Client dies akzeptiert, kann die Verbindung mitgelesen werden
- webmitm loggt Passwörter und Eingaben in HTML-Formulare

| Exkurs | Social–Hijacking, nicht ganz ernst |

- Pferdewetten–Hijacking
 - zwei Varianten
- Zeitungs–Hijacking
- Post–Hijacking
- Identitäts–Hijacking
- Flugzeug–Hijacking
- Lieferservice–Hijacking
 - u.U. Win–Win–Situation
- Einkaufswagen–Hijacking

| Gegenmaßnahmen und Erkennung |

Gegenmaßnahmen

- Verhindern, daß Pakete von anderen mitgelesen werden können
- Switches statt Hubs bieten geringen zusätzlichen Schutz (ARP–spoofing/ –relaying)
- Weniger anfällige Netztopologien wählen
- ipsec u.ä.
- Verschlüsselte und authentifizierte Verbindungen
- arpswatch, IDS
- statische ARP–Tabellen

Erkennung

- ACK–Storm, Retransmissions
- Identifikation von Karten im Promiscuous–Mode
- Ungewöhnlichkeiten (connections werden beendet, neue Zertifikate usw.)

| Links |

[http://www.usenix.org/publications/library/proceedings/securit
y95/full_papers/joncheray.txt](http://www.usenix.org/publications/library/proceedings/securit
y95/full_papers/joncheray.txt)

Simple Active Attack Against TCP, Laurent Joncheray

<http://lin.fsid.cvut.cz/~kra/index.html#HUNT>

Pavel Krauz, HUNT Project

<http://ettercap.sourceforge.net/>

Alberto Ornaghi, Marco Valleri. Ettercap

http://staff.washington.edu/dittrich/papers/arp_fun.txt

Playing redir games with ARP and ICMP, Yuri Volobuev

<http://phrack.infonexus.com/search.phtml?view&article=p50-6>

Phrack 50-6, route: Juggernaut

<http://packetstorm.securify.com/sniffers/>

u.a. Chad Renfro, Packet Sniffer Construction

<http://www.fefe.de/switch/index.html>

Felix von Leitner, Switching and VLAN Security FAQ

<http://www.monkey.org/~dugsong/dsniff>

Dug Song. dsniff (u.a. arpspoof, dnsspoof, macof, webmitm)

<http://www.krecher.de/>

u.a. Sean Harney: ippacket, Stefan Krecher: TCP-Spy, tkipconstructor, easterjack, easterrst, httpredir

| ./easter -h -egg | hijacking | Stefan Krecher | CCC 2002 |