

Eine erste Einführung in die Sicherheit von (verteilten) Systemen

Dozent: Prof. Dr. Michael Eichberg

Kontakt: michael.eichberg@dhw-mannheim.de

Version: 2024-02-26

Die Folien basieren in weiten Teilen auf einem Foliensatz von Prof. Dr. Henning Pagnia.

Alle Fehler sind meine eigenen.



Themen

- Transmission Control Protocol (TCP)
- Einmal-Passwörter
- Secure Shell (SSH)
- Firewalls
- The Onion Router (TOR)

1. TRANSMISSION CONTROL PROTOCOL (TCP)

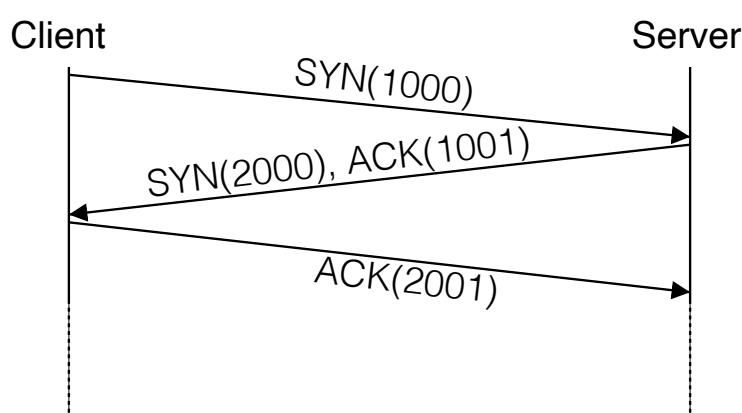
Prof. Dr. Michael Eichberg

TCP Grundlagen

- Protokoll der Schicht 4 (Transport Layer) basiert auf IP
- verbindungsorientierte Kommunikation zweier Rechner im Internet zuverlässig und geordnet:
 - Verwerfen von Duplikaten und fehlerhaft übertragener Pakete
 - automatisches Wiederversenden fehlender Pakete
 - Nachrichtenpuffer: Daten werden in korrekter Reihenfolge an Applikation zugestellt
- Verbindungsaufbau immer zwischen zwei Sockets (Socket-Adresse: IP Adresse und 16 Bit-Port-Nummer)

Aufbau einer TCP Verbindung

Dreifacher Handshake:



5

Terminologie:

SYN:	同步 (session establishment)
ACK:	acknowledge
RST:	reset

Verbindungsauftbau - Ablauf:

1. Client sendet SYN Paket mit initialer Sequenznummer 1000 an den Server.
2. Server sendet ein SYN-ACK Paket mit einem SYN mit seiner initialen Sequenznummer 2000 und ein ACK mit der Sequenznummer 1001 an den Client
3. Client sendet ein ACK Paket mit Sequenznummer 2001 an den Server; danach ist die Verbindung aufgebaut.

Das Betriebssystem sollte die initialen Sequenznummern zufällig wählen, so dass ein Angreifer diese nicht leicht vorhersagen kann. Beide Seiten haben eigene Sequenznummern, die unabhängig voneinander sind.

Bei einer laufenden Verbindung werden die Sequenznummern inkrementiert und es ist nicht (mehr) erkennbar wer die Verbindung aufgebaut hat.

Ports bei TCP

- Port-Nummern werden für die Kommunikation zwischen zwei Diensten/Prozessen verwendet
- Ports sind 16 Bit Zahlen (0-65535)
- (Unix) Ports < 1024 sind privilegiert (nur root kann diese öffnen)
- einige Port-Nummern sind Standarddiensten zugeordnet

Port-Nummern einiger Standarddienste [1]

Ungeschützte Dienste

Protokoll	Dienst	Portnummer
ftp	Dateitransfer	21
smtp	Simple Mail Transfer Protocol	25
dns	Domain Name System	53
http	Hypertext Transfer Protocol	80
login	Login auf entfernte Rechner	513

Geschützte Dienste

Protokoll	Dienst	Portnummer
ssh	Secure Shell	22
https	HTTP über Secure Socket Layer	443
smt�ps	SMTP über Secure Socket Layer	465
imaps	IMAP über Secure Socket Layer	993
pop3s	POP3 über Secure Socket Layer	995

[1] Port numbers assigned by IANA

Angriffe auf TCP - Motivation

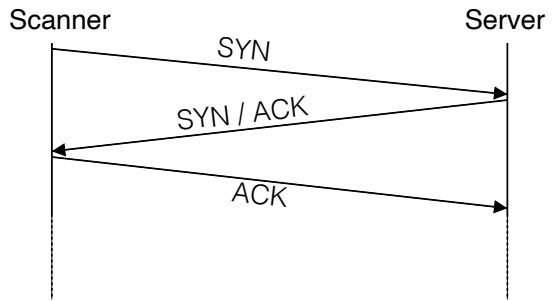
- Netzwerkprogrammierung mit TCP ist relativ komfortabel.
- Viele Dienste sind mit TCP implementiert.
- Angreifer nutzen Schwachstellen in TCP Diensten aus.
- Server haben heutzutage i. Allg. alle nicht verwendeten Dienste geschlossen.
Angreifer muss verwundbare Dienste zum Beispiel durch Port Scans finden.

Port Scans: TCP Connect Scan

- vollständiger Verbindungsauftbau zu allen bzw. zu ausgewählten Ports

Bewertung:

- simpelster Port Scan
- große Entdeckungsgefahr (Scan selbst ist kein Angriff)
- mögliche Verbesserung: zwischen dem Scannen mehrerer Ports Pausen einstreuen (Wie lange?)

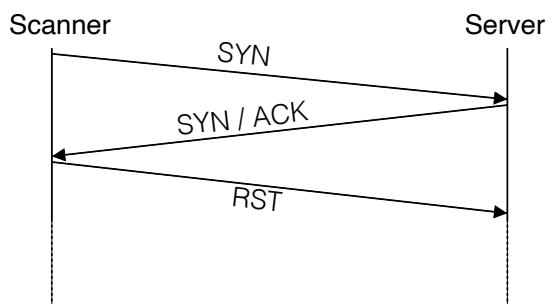


Port Scans: TCP SYN Scan

1. Senden eines TCP-Segments mit gesetztem SYN-Flag an einen Port
2. falls der *Port offen* ist, kommt SYN/ACK zurück danach RST senden
3. falls der *Port nicht offen* ist, kommt RST (oder nichts) zurück

Bewertung:

- kein vollständiger Verbindungsaufbau
- meist nicht protokolliert
- geringe(re) Entdeckungsgefahr



Port Scans: Stealth Scans

Versenden eines für den Verbindungsaufbau ungültigen TCP-Segments an einen Port:

- NULL-Scan (keine Flags)
- ACK-Scan (ACK-Flag)
- FIN-Scan (FIN-Flag)
- XMAS-Scan (alle Flags)

Laut RFC kommt RST zurück, falls Port offen. (Reaktion aber abhängig vom Betriebssystem)

Bewertung:

- Zugriff wird meist nicht protokolliert
- Scan bleibt unbemerkt

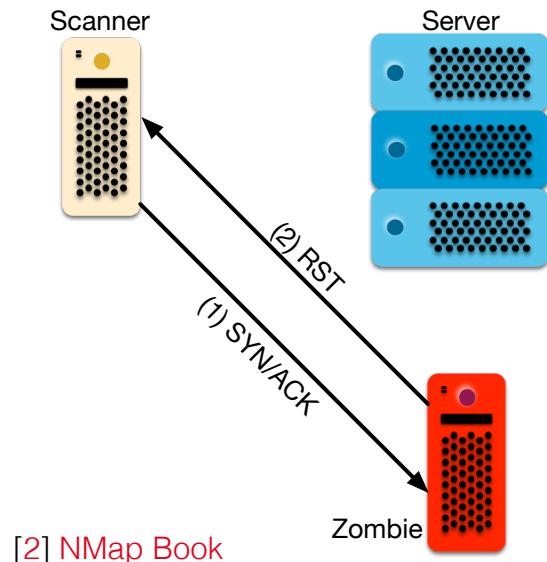
XMAS-Scan:

Bei diesem Scan sind alle Flags gesetzt; ein XMAS-Scan wird auch als Christmas-Tree-Scan bezeichnet, da das Paket erleuchtet ist wie ein Weihnachtsbaum.

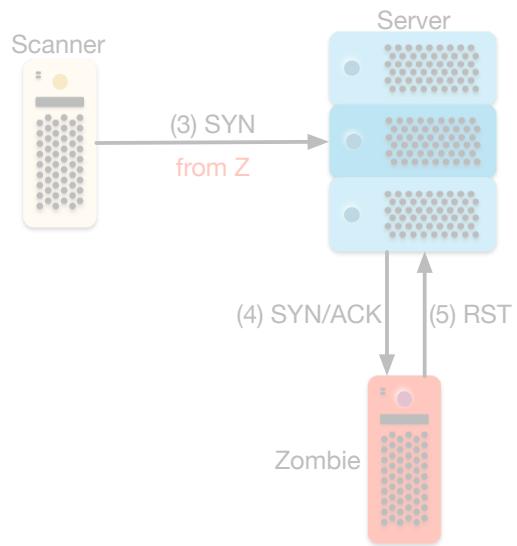
Port Scans: Idle Scan [2]

Bei allen bisher betrachteten Scans kann der Scanner prinzipiell identifiziert werden. Unter Verwendung eines sog. Zombies geht es auch anders:

Sondiere IP ID des Zombies:



Starte Scan:



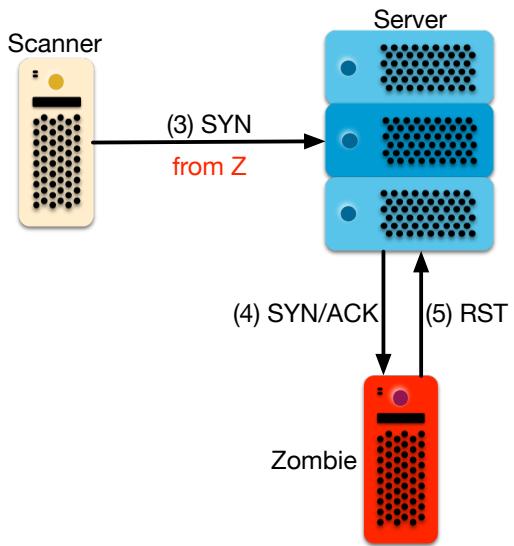
Zombies: ein Rechner (Computer, Drucker oder anderes IoT Gerät) im Internet *möglichst ohne eigenen Netzverkehr* und mit „altem“ Betriebssystem, bei dem die IP ID in vorhersehbarer Weise inkrementiert wird.

Sollte ein Intrusion Detection System vorhanden sein, so wird dieses den Zombie als Angreifer identifizieren.

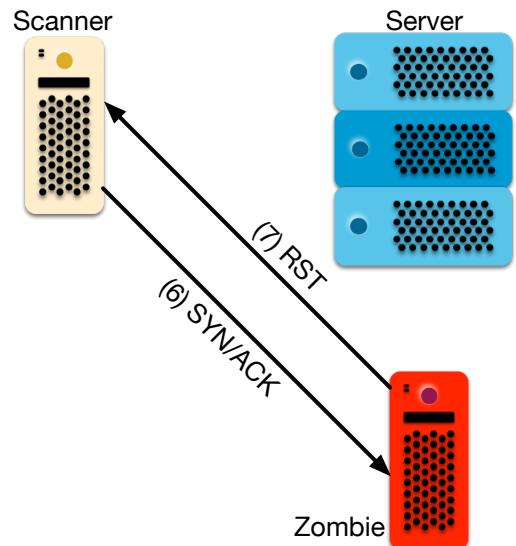
Grundlegende Idee: Der Zombie sendet ein RST Paket zurück, da er kein SYN gesendet hat und kein SYN/ACK erwarte. Dadurch erfährt der Angreifer die aktuelle IP ID des Zombies. Über diesen Seitenkanal - d.h. die Veränderung der IP ID des Zombies - kann der Angreifer nun den Zustand des Ports auf dem Zielrechner ermitteln.

Port Scans: Idle Scan

Starte Scan:



Sondiere IP ID des Zombies:



Port Scans: Idle Scan - Zusammenfassung

- Angreifer sendet SYN/ACK Paket an Zombie
- der Zombie antwortet mit RST und enthüllt seine IP ID ( *IP Fragment Identification Number*).
- Angreifer sendet SYN ("vom" Zombie) an Port des Servers
- **[Port offen]** Der Zielrechner antwortet mit SYN/ACK an den Zombie, wenn der Port offen ist.
[Port geschlossen] Der Zielrechner antwortet mit RST an den Zombie, wenn der Port geschlossen ist. Dies wird vom Zombie ignoriert.
- **[Port offen]** Der Zombie antwortet mit RST, da er kein SYN gesendet hat und kein SYN/ACK erwartet und erhöht seine IP ID.
- Der Angreifer sendet wieder ein SYN/ACK an den Zombie, um die IP ID zu erfahren.

Mit einem IDLE Scan kann nicht unterschieden werden, ob der Port geschlossen oder gefiltert ist.

Port Scans mit nmap

- alle Arten von Port-Scans möglich
- auch OS fingerprinting
- u. U. sogar Ermittlung der Versionsnummern von Diensten

```
$ nmap 192.168.178.121 -Pn
Starting Nmap 7.94 ( https://nmap.org ) at 2023-12-14 13:16 PST
Nmap scan report for Michaels-MacBook-Pro (192.168.178.121)
Host is up (0.0056s latency).

Not shown: 995 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
445/tcp   open  microsoft-ds
5000/tcp  open  upnp
7000/tcp  open  afs3-fileserver
```

15

OS-Fingerprinting

Beim OS-Fingerprinting werden Datenpakete analysiert, die aus einem Netzwerk stammen, um Informationen für spätere Angriffe zu gewinnen. Durch die Erkennung des Betriebssystems, mit dem ein Netzwerk arbeitet, haben Hacker es leichter, Schwachstellen zu finden und auszunutzen. OS-Fingerprinting kann auch Konfigurationsattribute von entfernten Geräten sammeln. Diese Art von Aufklärungsangriff ist in der Regel (einer) der erste(n) Schritt(e).

Es gibt zwei Arten von OS-Fingerprinting: (1) Aktiv und (2) passiv.

1. Bei einem aktiven OS-Fingerprinting-Versuch senden die Angreifer ein Paket an das Zielsystem und warten auf eine Antwort, um den Inhalt des TCP-Pakets zu analysieren.
2. Bei einem passiven Versuch agieren die Angreifer eher als "Schnüffler", der keine absichtlichen Änderungen oder Aktionen im Netzwerk vornimmt. Passives OS-Fingerprinting ist ein unauffälligerer, aber wesentlich langsamerer Prozess.

Port Knocking

- Ein Knock-Daemon versteckt offene Ports auf dem Server.
- Zugriffe auf alle Ports werden im Log-File protokolliert.
- Knock-Daemon beobachtet das Log-File.
- Erst nach Erkennen einer vordefinierten (Einmal-)Klopfsequenz öffnet der Knock-Daemon den gewünschten Port für diesen Client.
- Client kann nun die Verbindung aufbauen.

16

Weiterführend

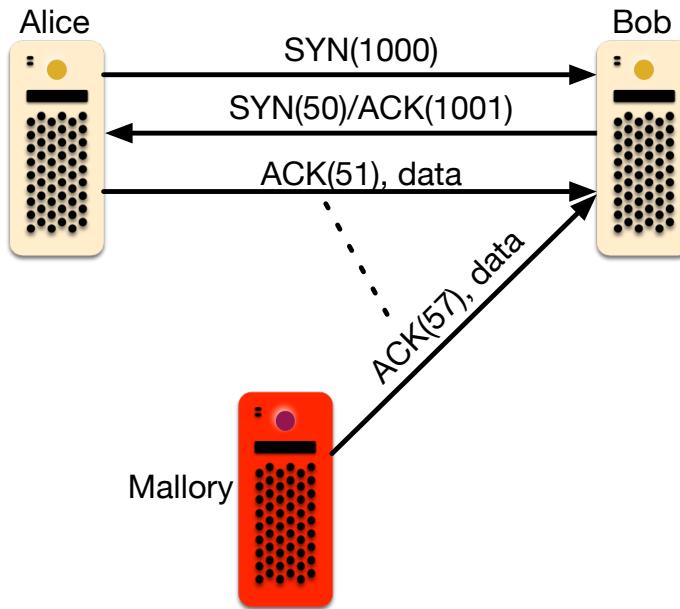
Alternativen zu einer Knock-Sequenz ist zum Beispiel, dass der Port nur dann als offen gilt, wenn die IP ID eine bestimmte Sequenznummer aufweist.

M. Krzywinski: Port Knocking: Network Authentication Across Closed Ports in SysAdmin Magazine 12: 12-17. (2003)

TCP Stealth

Connection Hijacking

Angreifer übernimmt eine bestehende - zum Beispiel eine bereits durch (Einmal-)Passwort authentisierte - Verbindung.



17

TCP/IP-Hijacking ist eine Form eines Man-in-the-Middle-Angriffs. Der Angreifer bestimmt erst die IP-Adressen der beiden Sitzungsteilnehmer.

Danach gibt es mehrere Möglichkeiten:

- Der Angreifer schickt ("in einer Pause") ein Paket mit der passenden Sequenznummer an den Server.
(Dies kann dann in einem ACK-Storm enden, was ggf. unterbunden werden muss (zum Beispiel durch das Senden eines RSTs), oder ignoriert werden kann.)
- Der Angreifer macht einen Client mit einem DoS-Angriff unerreichbar, um sich dann mit dem Anderen zu verbinden, indem er die Netzwerk-ID des ausgeschalteten Clients nutzt.

Denial-of-Service (DoS) Angriffe

Ziel des Angreifers: Lahmlegen eines Dienstes oder des ganzen Systems ...

- durch Ausnutzen von Schwachstellen (FLAG *vulnerabilities*, z.B. Buffer Overflow)
- durch Generierung von Überlast (Ausschöpfen von RAM, CPU, Netzwerkbandbreite, ...)

Beispiel: Ping-of-Death

(Historisch: aus dem Jahr 1997)

Ein **ping** verwendet Internet Control Message Protocol (ICMP) üblicherweise kleine Nachrichten, verwendete Länge ist aber einstellbar.

Falls zu groß ⇒ Buffer Overflow ⇒ Systemabsturz!

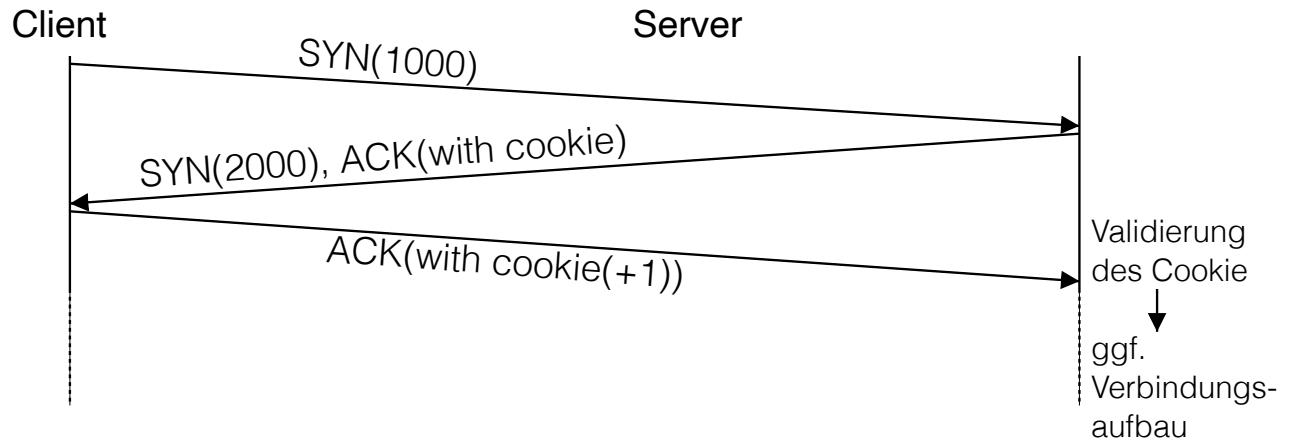
Variante: mittels Fragmentierung ließen sich generell übergroße IP-Pakete (>65,536 Byte) erstellen.

Denial-of-Service: SYN-flooding Angriff

- Angriff auf Design
- Angreifer sendet eine Verbindungsaufbauanforderung (gesetztes SYN-Flag) an Zielmaschine
- Server generiert eine halboffene TCP-Verbindung
- Angreifer wiederholt in schneller Folge dieses erste Paket zum Verbindungsauftakt
⇒ vollständiges Füllen der internen Systemtabelle
⇒ Anfragen normaler Benutzer werden zurückgewiesen
- Angreifer verwendet i. Allg. IP-Spoofing weswegen Firewalls wirkungslos sind.
- Abwehr: SYN-Cookies

SYN-Cookies - D J. Bernstein

SYN-Cookies sind speziell konstruiert initiale Sequenznummern.

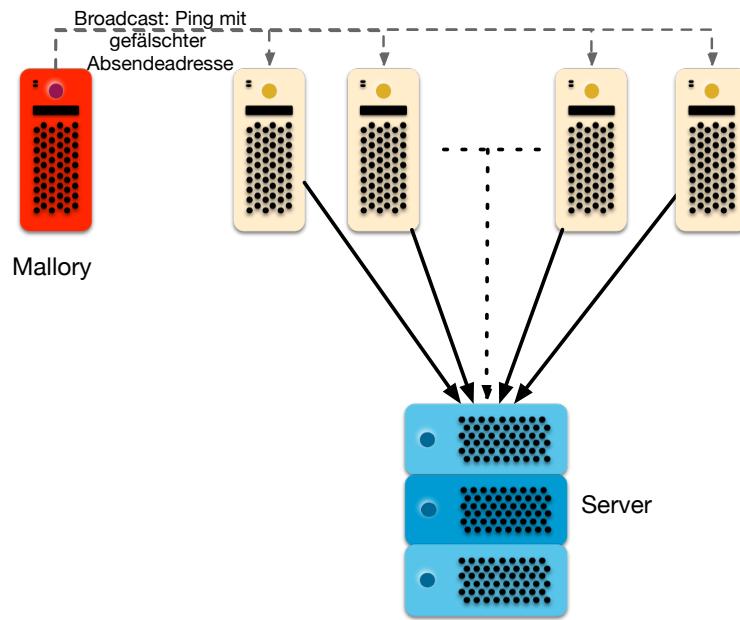


Der Cookie ermöglicht es, dass keine Informationen im Speicher gehalten werden müssen.
Der Cookie encodiert die Informationen, die der Server benötigt, um die Verbindung aufzubauen: Client IP, time window, etc.

Distributed Denial-of-Service (DDoS) Angriff

Opfer wird von sehr vielen Angreifern mit Nachrichten überflutet.

Ein Beispiel: Smurf-Angriff:



Distributed Denial-of-Service (DDoS) Angriff

- Bot-Netze (Botnetze) werden verwendet, um DDoS-Angriffe durchzuführen.
- Bot-Netze können viele 10.000 Rechner umfassen.
- IoT Geräte sind besonders beliebt (z.B. IP-Kameras, Smart-TVs, Smart-Home Geräte, ...), da diese oft nicht ausreichend geschützt sind und trotzdem permanent mit dem Internet verbunden sind.
- Beliebte Ziele:
 - Onlinespieleserver
 - Banking-Portale
 - politische Webseiten
- Firewalls und Intrusion Detection Systeme sind meist wirkungslos, da die Angriffe von vielen verschiedenen IP-Adressen kommen.

Distributed-Reflected-Denial-of-Service (DRDoS)

Angriff

- Idee:
 - Es wird eine Anfrage an einen Server gesendet, die eine große Antwort auslöst. (z.B. hat(te) der NTP Monlist Befehl eine Antwort, die ca. 200 Fach größer ist als die Anfrage!)
 - Mittels IP-Spoofing wird die IP-Adresse des Opfers als Absenderadresse verwendet.
 - Es werden insbesondere Dienste basierend auf UDP verwendet, da hier keine Verbindung aufgebaut werden muss.
- Nehmen einen signifikanten Teil aller DDoS-Angriffe ein.
- Die Tatsache, dass die Sender legitime Server sind, erschwert die Abwehr.
-  *Egress Filtering* kann helfen, die Verwendung von IP-Spoofing zu verhindern.

23

Bereits im Jahr 2018 wurde ein Angriff mit einer Bandbreite von 1,7 TBit/s beobachtet.

Egress Filtering: Der Router verwirft alle Pakete, die eine Absenderadresse verwenden, die nicht aus dem eigenen Netzwerk stammt.

Distributed Denial-of-Service (DDoS) Angriffe - Beispiel

[...] Google's DDoS Response Team has observed the trend that distributed denial-of-service (DDoS) attacks are **increasing exponentially in size**. Last year, we blocked the largest DDoS attack recorded at the time. This August [2023], we stopped an even larger DDoS attack — 7½ times larger — that also used new techniques to try to disrupt websites and Internet services.

This new series of DDoS attacks reached **a peak of 398 million requests per second (rps)**, and relied on a novel HTTP/2 “Rapid Reset” technique based on stream multiplexing that has affected multiple Internet infrastructure companies. By contrast, last year's largest-recorded DDoS attack peaked at 46 million rps.

Distributed Denial-of-Service (DDoS) Angriffe

Beispiele:

- TCP Stack Attacks (SYN, FIN, RST, ACK, SYN-ACK, URG-PSH, other combinations of TCP Flags, slow TCP attacks)
- Application Attacks (HTTP GET/POST Floods, slow HTTP Attacks, SIP Invite Floods, DNS Attacks, HTTPS Protocol Attacks)
- SSL/TLS Attacks (Malformed SSL Floods, SSL Renegotiation, SSL Session Floods)
- DNS Cache Poisoning
- Reflection Amplification Flood Attacks (TCP, UDP, ICMP, DNS, mDNS, SSDP, NTP, NetBIOS, RIPv1, rpcbind, SNMP, SQL RS, Chargen, L2TP, Microsoft SQL Resolution Service)
- Fragmentation Attacks (Teardrop, Targa3, Jolt2, Nestea)
- Vulnerability Attacks
- Resource Exhaustion Attacks (Slowloris, Pyloris, LOIC, etc.)
- Flash Crowd Protection

Schutz vor DDoS-Angriffen - On-Site

Robustheitsmaßnahmen

- Aufrüsten der Ressourcen (z.B. Bandbreite, CPU, RAM, ...)
- Exemplarische Sofortmaßnahmen bei aktivem Angriff:
 - Whitelisting von IP-Adressen von besonders wichtigen Clients
 - Blacklisting von IP-Adressen aus bestimmten Bereichen
 - Captchas
 - Überprüfung der Browser-Echtheit
- Anti-DDoS Appliances

Achtung

Diese Maßnahmen sind häufig teuer und ggf. begrenzt effektiv; wenn der Angriff die verfügbare Bandbreite übersteigt, sind diese Maßnahmen wirkungslos.

Schutz vor DDoS-Angriffen - Off-Site

Robustheitsmaßnahmen

- Einbinden des ISP
- Einbinden spezialisierter Dienstleister (im Angriffsfall wird mittels BGP-Rerouting der Traffic an den Dienstleister umgeleitet, der dann die DDos Attacke filtert.)
- Content-Delivery-Networks (CDNs) für statische Inhalte (z.B. Cloudflare, Akamai, ...)
- Distributed Clouds

Password Sniffing

In der Anfangszeit:

unverschlüsselte Übertragung von Passwörtern (telnet, ftp, ...)

In der Übergangszeit (bzw. in bestimmten Szenarien auch heute):

Verwendung von Einmal-Passwörtern (S/Key, ...)

Heute:

Passwörter werden verschlüsselt übertragen (ssh, https, ...)

Unverschlüsselte Passwörter können leicht mittels eines Sniffers, der den Netzwerkverkehr mitschneidet (z.B. Wireshark), abgefangen werden.

Einmal-Passwörter

Die Idee ist, dass Passwörter nur genau einmal gültig sind und nicht wiederverwendbar sind.

- Tokens (z.B. RSA SecurID)
- Codebuch: Liste von Einmal-Passwörtern, die das gemeinsame Geheimnis sind.
- S/Key: Passwort "wird mit einem Zähler kombiniert" und dann gehasht.

Das S/Key Verfahren

Prinzip

Einmal-Passwort-System nach Codebuch-Verfahren, dass im Original auf der kryptographischen Hashfunktion MD4 basiert.

Initialisierung

1. Der Nutzer gibt sein Passwort W ein; dies ist der geheime Schlüssel. (Sollte W bekannt werden, dann ist die Sicherheit des Verfahrens nicht mehr gewährleistet.)
2. Eine kryptografische Hash-Funktion H wird n -mal auf W angewandt, wodurch eine Hash-Kette von n einmaligen Passwörtern entsteht. $H(W), H(H(W)), \dots, H^n(W)$
3. Das initiale Passwort wird verworfen.
4. Der Benutzer erhält die n Passwörter, die in umgekehrter Reihenfolge ausgedruckt werden: $H^n(W), H^{n-1}(W), \dots, H(H(W)), H(W)$.
5. Nur das Passwort $H^n(W)$, das an erster Stelle der Liste des Benutzers steht, der Wert von n und ggf. ein Salt, wird auf dem Server gespeichert.

1

Anmeldung

Identifizierte n das letzte verwendete Passwort.

- Der Server fragt den Nutzer nach dem Passwort $n - 1$ (d.h. $H^{n-1}(W)$) und übermittelt ggf. auch den Salt.
- Der Server hasht das Passwort und vergleicht es mit dem gespeicherten Passwort $H^n(W)$.
- Ist das Passwort korrekt, dann wird der Nutzer angemeldet und der Server speichert das Passwort $H^{n-1}(W)$ als neues Passwort $H^n(W)$ und dekrementiert n .

2

30

Intern verwendet S/KEY 64-bit Zahlen. Für die Benutzbarkeit werden diese Zahlen auf sechs kurze Wörter, von ein bis vier Zeichen, aus einem öffentlich zugänglichen 2048-Wörter-Wörterbuch ($2048 = 2^{11}$) abgebildet. Zum Beispiel wird eine 64-Bit-Zahl auf "ROY HURT SKI FAIL GRIM KNEE" abgebildet.

Secure Shell (SSH)

Verschlüsselte Verbindung

SSH ermöglicht die sichere Fernanmeldung von einem Computer bei einem anderen (typischerweise über TCP über Port 22). Es bietet mehrere Optionen für eine starke Authentifizierung und schützt die Sicherheit und Integrität der Kommunikation durch starke Verschlüsselung

Ablauf

1. Authentisierung des Server-Rechners
2. Authentisierung des Benutzers (bzw. des Clients) mittels
 - a. Passwort
 - b. ~~.rhosts-Eintrag~~
 - c. privatem (RSA-)Key (hauptsächlich verwendete Methode)
3. Kommunikation über symmetrisch verschlüsselte Verbindung

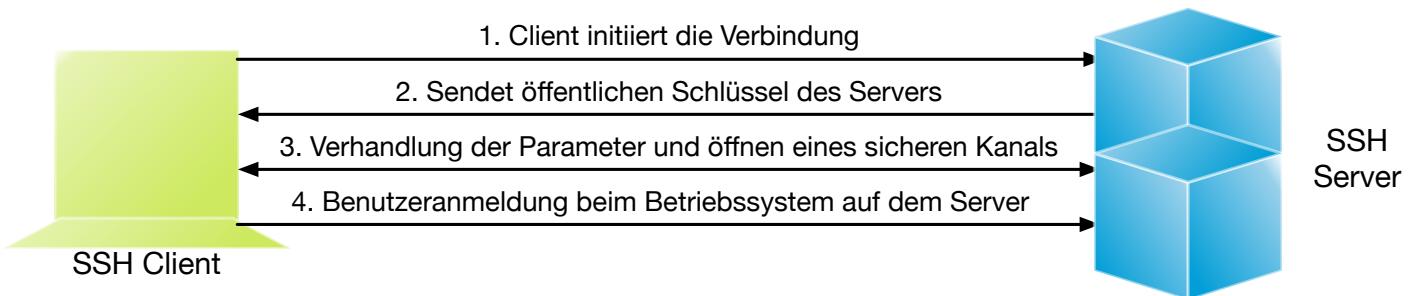
31

Die Authentifizierung mittels eines Schlüsselpaares dient primäre der Automatisierung (dann wird auch keine „Schlüsselphrase“ zum Schutz des Passworts verwendet). Auf jeden Fall ist effektives Schlüsselmanagement erforderlich:

[...] In einigen Fällen haben wir mehrere Millionen SSH-Schlüssel gefunden, die den Zugang zu Produktionsservern in Kundenumgebungen autorisieren, wobei 90 % der Schlüssel tatsächlich ungenutzt sind und für einen Zugang stehen, der zwar bereitgestellt, aber nie gekündigt wurde.

—SSH.com (Dez. 2023)

Secure Shell (SSH) - Protokoll



Beide Seiten haben einen Public-private Key Schlüsselpaar zur Gegenseitigen Authentifizierung

- User Keys:**
- **Authorized keys** - Datei mit den öffentlichen Schlüsseln der Nutzer, gespeichert auf Serverseite
 - **Identity keys** private Schlüssel der Nutzer
- Host keys:** benötigt für die Authentifizierung von Servern, um Man-in-the-Middle-Angriffe zu verhindern.
- Session Keys:** werden für die symmetrische Verschlüsselung der Daten in einer Verbindung verwendet. Session Keys (*Sitzungsschlüssel*) werden während des Verbindungsbaus ausgehandelt.

Im Falle von SSH gibt es kein initiales Vertrauen zwischen Server und Client.

Secure Shell (SSH) - Verbindungsauflaufbau - Beispiel

```
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to example.org [1.2.3.4] port 22.
debug1: Connection established.
debug1: identity file /home/user/.ssh/id_rsa type -1
debug1: identity file /home/user/.ssh/id_rsa-cert type -1
debug1: identity file /home/user/.ssh/id_dsa type -1
debug1: identity file /home/user/.ssh/id_dsa-cert type -1
debug1: Remote protocol version 1.99, remote software version OpenSSH_5.8
debug1: match: OpenSSH_5.8 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_5.5p1 Debian-6
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: server->client aes128-ctr hmac-md5 none
debug1: kex: client->server aes128-ctr hmac-md5 none
debug1: SSH2_MSG_KEX_DH_GEX_REQUEST(1024<1024<8192) sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_GROUP
debug1: SSH2_MSG_KEX_DH_GEX_INIT sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_REPLY
debug1: Host 'example.org' is known and matches the RSA host key.
debug1: Found key in /home/user/.ssh/known_hosts:1
debug1: ssh_rsa_verify: signature correct
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS

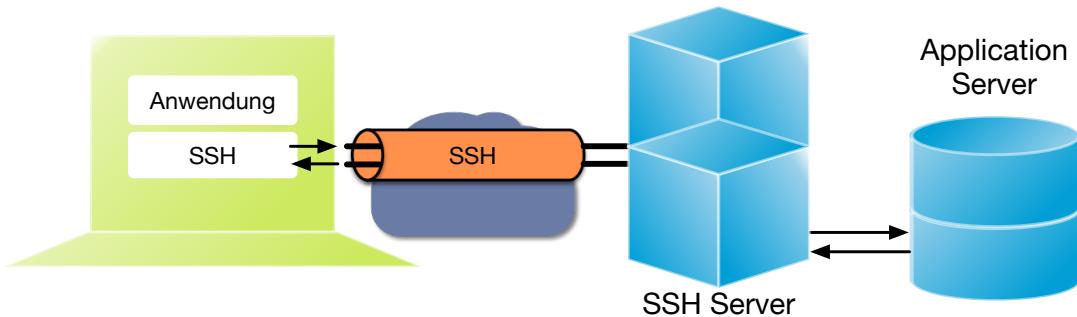
debug1: SSH2_MSG_NEWKEYS received
debug1: Roaming not allowed by server
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password,keyboard-interactive,hostbased
debug1: Next authentication method: publickey
debug1: Trying private key: /home/user/.ssh/id_rsa
debug1: Trying private key: /home/user/.ssh/id_dsa
debug1: Next authentication method: keyboard-interactive
debug1: Authentications that can continue: publickey,password,keyboard-interactive,hostbased
debug1: Next authentication method: password
user@example.org's password:
debug1: Authentication succeeded (password).
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
debug1: Sending environment.
```

Secure Shell (SSH) - Risiken durch mangelnde Schlüsselverwaltung

- Schlüssel werden nicht regelmäßig ausgetauscht
- Schlüssel werden nicht gelöscht, wenn sie nicht mehr benötigt werden
- viele (die meisten) Schlüssel werden nicht verwendet
- Es ist oft nicht bekannt, wer Zugriff auf welche Schlüssel hat(te)
- Es ist nicht bekannt, welche Schlüssel auf welche Systeme Zugriff haben
- Malware kann SSH-Schlüssel stehlen
- SSH Keys können ggf. privilegierten Zugriff gewähren
- SSH Keys können benutzt werden, wenn um Backdoors zu verstecken
- Server keys erlauben ggf. Man-in-the-Middle-Angriffe

SSH Tunneling

- ermöglicht die Übertragung beliebiger Netzwerkdaten über eine verschlüsselte SSH-Verbindung. z.B.
 - um ältere Anwendungen zu verschlüsseln.
 - um VPNs (Virtual Private Networks) zu implementieren
 - um über Firewalls hinweg auf Intranetdienste zuzugreifen.
- ermöglicht auch Port-forwarding (lokale Ports werden auf entfernten Rechner weitergeleitet)



SSH und „Back-tunneling“

- Der Angreifer richtet einen Server außerhalb des Zielnetzwerks ein
- Nach Infiltration des Zielsystems verbindet der Angreifer sich von innen mit dem externen SSH-Server.
- Diese SSH-Verbindung wird so eingerichtet, dass eine TCP-Port-Weiterleitung von einem Port auf dem externen Server zu einem SSH-Port auf einem Server im internen Netzwerk möglich ist.
- Die meisten Firewalls bieten wenig bis gar keinen Schutz dagegen.

Es ist in diesem Fall besonders interessant für den Angreifer den SSH Server zum Beispiel bei einem Cloud-Anbieter zu betreiben, welcher von dem Unternehmen standardmäßig verwendet wird (am Anfang steht immer die Aufklärung!). In diesem Fall wird die Firewall keine ausgehenden SSH-Verbindungen dorthin blockieren.

Schwachstellen in SSH

Nearly 11 million SSH servers vulnerable to new Terrapin attacks

[...] It [The Terrapin attack] manipulates sequence numbers during the handshake process to compromise the integrity of the SSH channel, particularly when specific encryption modes like ChaCha20-Poly1305 or CBC with Encrypt-then-MAC are used. [...]

By Bill Toulas

—January 3, 2024 10:06 AM

Übung: Port Scans - IDLE Scan

- Warum kann mit einem IDLE Scan nicht festgestellt werden warum ein Port geschlossen oder gefiltert ist?
- Welchen Wert hat die IP ID des Zombies, der einem IDLE Scan durchführt, wenn der Zielport offen bzw. geschlossen ist wenn der Scanner diesen wieder abfragt?

1. Welche Vorteile bieten Einmalpasswortsysteme gegenüber Systemen mit mehrfach zu verwendenden Passworten?
2. Welchen Angriffen sind Einmalpasswortsysteme weiterhin ausgesetzt?
3. Generieren Sie eine Liste von Einmalpassworten mit Initialwert $r = 769$. Generieren Sie $h(r)$ bis $h_6(r)$ wenn die Einwegfunktion hier der Einfachheit halber $h(x) = x^2 \bmod 1000$ ist.
4. Wie oft kann sich der Benutzer anmelden? Wie sieht seine Liste aus?
5. Welchen Wert speichert der Server vor dem jeweiligen Anmeldevorgang?
6. Spielen Sie zwei Anmeldevorgänge durch.
7. Wenn ein Passwort $H^L(W)$, $1 < L < N$ bekannt ist, welche Auswirkungen hat dies auf die Sicherheit des Verfahrens?

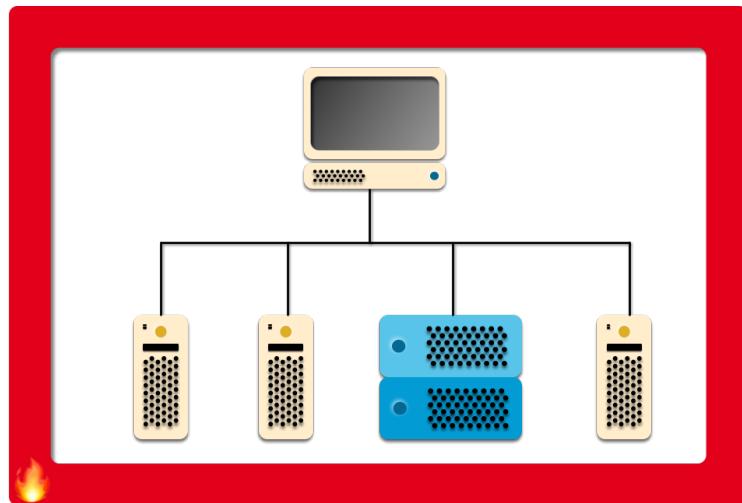
Übung: DDoS

1. Welches Problem entsteht wenn zum Schutze vor Angriffen auf die Verfügbarkeit die Ressourcen von IT-Systemen und deren Internet-Anbindung erhöht werden?
2. Recherchieren Sie was ein "Low and Slow Angriff" ist.
3. Wo kann überall "Egress filtering" statt finden.

2. FIREWALLS

Prof. Dr. Michael Eichberg

Unabhängiges Netz - „Ideale Situation“



Vorteile:

- keinerlei Angriffsmöglichkeiten von außen
- kein Schutz gegen Insider
- kein Zugang zum Internet

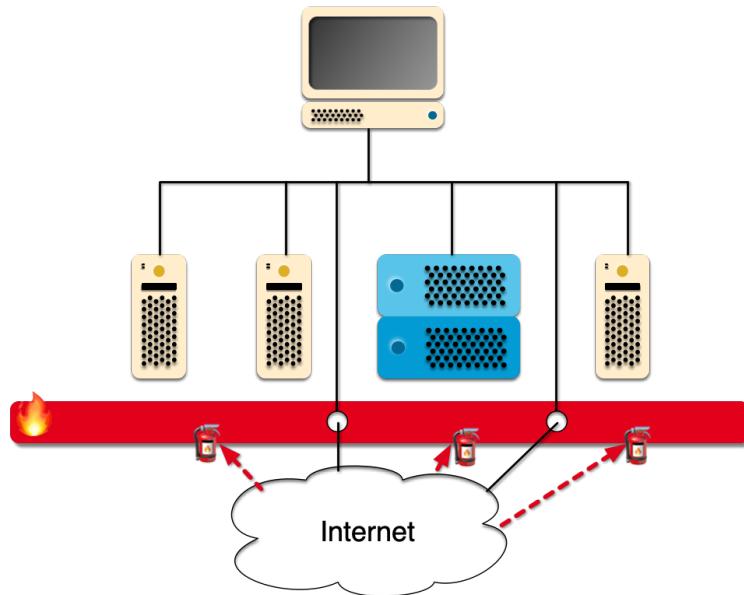
Nachteile:

Von der Notwendigkeit des Schutzes von Rechnern

[...] Züger und sein Team hätten [...] erst kürzlich ein Experiment durchgeführt, [...]. Sie hätten einen Computer "ohne jeglichen Schutz" mit dem Internet verbunden, um zu sehen, wie lange es dauere, bis er befallen sei. Konkrete Details zur Konfiguration dieses Systems werden zwar nicht genannt, angeblich war der Rechner aber schon nach 20 Minuten infiltriert.

—Golem.de 6.2.2024

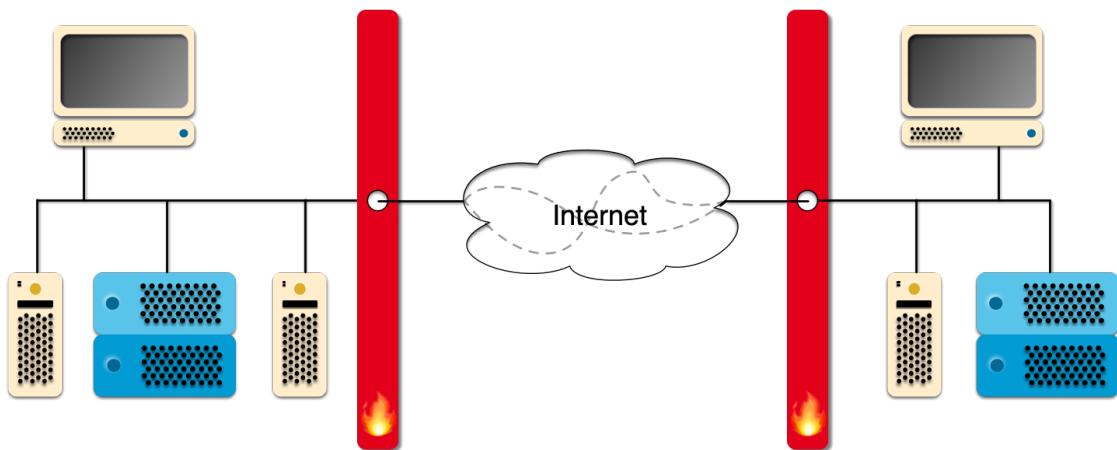
Schutzschicht zwischen internem und externem Netz



- Kontrolle des Nachrichtenverkehrs durch Filterung
- begrenzte Isolation mit begrenztem Schutz

Eine Firewall schafft zwischen verbundenen Netzen Sicherheitsdomänen mit unterschiedlichem Schutzbedarf. Eine wichtige Teilaufgabe ist das Ausarbeiten von Sicherheitsrichtlinien.

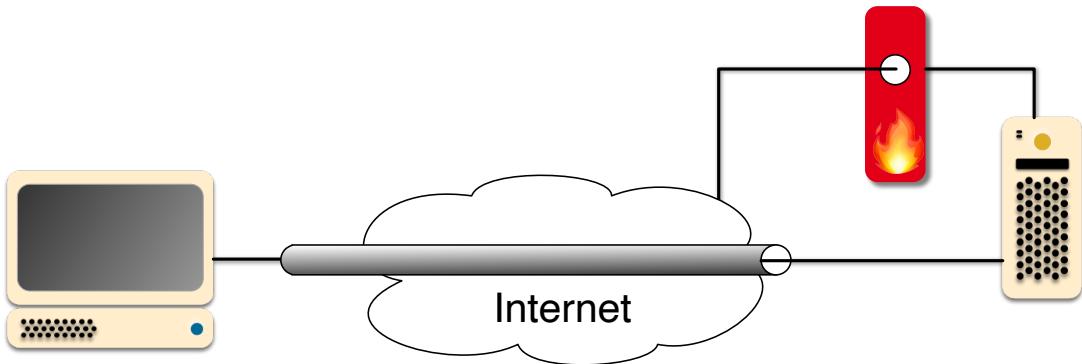
Realisierung von Virtual Private Networks (VPN)



- Aufbau einer scheinbar privaten Verbindung von Firmenteilnetzen über das (öffentliche) Internet
- Zusätzliche Verbindungsverschlüsselung zwischen den Firewalls.

Ziel ist es aktive und passive Angriffe zu unterbinden. Selbst bei verschlüsselten Verbindungen kann die Verkehrsflussanalyse noch Informationen liefern über die Verbindungen liefern.

Kommerzielle VPNs für Endnutzer



46

Motivation

- Schutz der Privatsphäre; der ISP kennt nicht mehr die Webseiten, die man aufruft
- Die IP-Adresse des Nutzers ist den aufgerufenen Webseiten nicht mehr bekannt und kann deswegen der Umgehung von Geo-Blocking dienen.

Nachteile?

- Vertrauen in den VPN-Anbieter muss vorhanden sein. Insbesondere, beim Einsatz zum Stärken der Privatsphäre, muss der VPN-Anbieter vertrauenswürdig sein und sollte ein so genannter "no-log" Anbieter sein. Es gibt auch (kostenlose) VPN-Anbieter, die die Daten der Nutzer verkaufen (ehemals: [Facebook Onavo](#)).

Schutz auf den Schichten des TCP/IP Stacks

Zentraler Schutz des gesamten internen Netzwerks durch:

- Paket Filter ( *Packet Filtering*)
 - Blockieren bestimmter IP-Empfänger-Adressen (extern / intern)
 - Blockieren bestimmter IP-Absender-Adressen (extern / intern)
(z.B. aus dem Internet mit internen IP-Absender-Adressen.)
 - Blockieren bestimmter Dienste; ggf. nur für bestimmte IP-Adressen
- Filter auf Anwendungsebene ( *Application-level Filtering*)
 - inhaltsbezogene Filterung der Verkehrsdaten eines Dienstes
 - z.B. Virenfilter
 - wirkungslos bei verschlüsselten Verkehrsdaten
- Protokollierungsmöglichkeit der Kommunikation von / nach extern

47

Firewalls (alleine) können die Struktur des Netzwerks nicht verbergen.

DoS Attacke auf Anwendungsebene

[...]

Angriff auf die Kleinen

Waren bei früheren Spamangriffen massenhaft Accounts auf der größten Mastodon-Instanz mastodon.social angelegt worden, die dann von dort ihre Inhalte verbreiteten, trifft es nun nicht die größte, sondern die kleinsten. Automatisiert werden dabei Instanzen ausgesucht, auf denen eine Registrierung ohne Überprüfung und sogar ohne ein Captcha möglich ist. Das können etwa solche mit wenigen Accounts sein, die von Enthusiasten etwa für eine Gemeinde betrieben werden. Waren die Verantwortlichen in den vergangenen Tagen nicht aufmerksam, wurden diese Instanzen dann regelrecht überrannt. Die Spam-Accounts verschickten massenhaft Nachrichten mit einem Bild des namensgebenden Frühstücksfleischs und Links zu Discord-Servern, die wohl lahmgelegt werden sollten.

—Mastodon: *Spamwelle zeigt Schwächen auf [...]*

Realisierungsmöglichkeiten von Firewalls

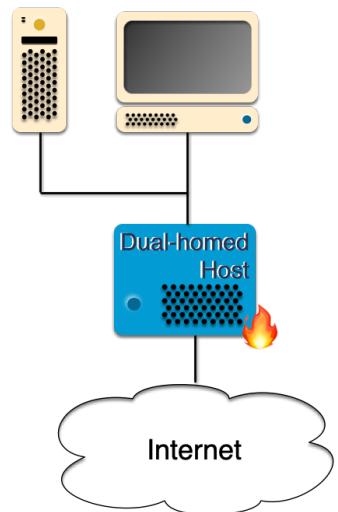
- Hardware-Firewall
 - Screening Router
 - Application Gateway (auch Bastion Host)
 - Proxy-Server für bestimmte Dienste
 - Client-Software (HTTP-Browser, telnet, ftp, ...)
 - Server-Software
- Software-Firewall (*Personal Firewall*)

Im Falle eines  *Bastion Host*, ist dies der einzige unmittelbar erreichbare Rechner.

Dual-Homed Host

Aufbau

- zwei Netzwerkkarten: ggf. private interne Adressen
- Screening Router & Gate: Packet Filter und Application-Level Filter
- Proxy-Dienste installieren
- Benutzer-Logins von extern
- Konf. der Netzwerkkarten: IP-Pakete nicht automat. weiterleiten



Screening Router

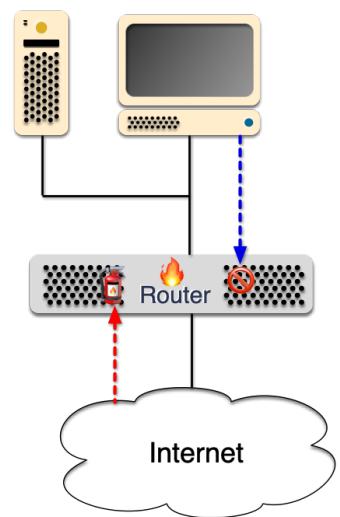
Aufbau

- programmierbarer Hardwarerouter
- simple Filterfunktionen:
 - nur Paket-Header prüfen
 - schnelle Auswertung ermöglicht hohen Durchsatz
- Realisierung eines Packet Filters

Bewertung

- ✓ einfach und billig
- ✓ flexibel

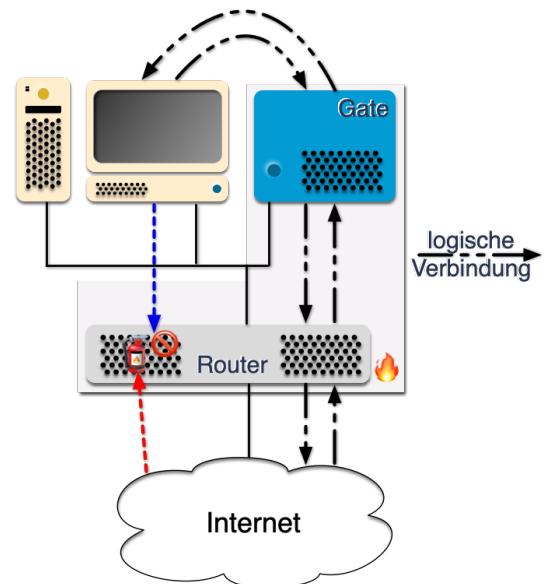
- ! schwer zu testen
- ! Protokollierung
- ! Fernwartung
- ! keine Inhaltenfilterung



Screened Host

Aufbau

- Screening Router blockiert:
 - Pakete von / an interne Rechner (nicht Gate)
 - Source-Routed Pakete
- von extern nur Gate sichtbar
- Pakete von intern nur via Gate
- Gate bietet Proxy-Server (z.B. für E-Mail)



52

Gibt es für eine bestimmte Anwendung kein Application-level Proxy, dann kann auf einen für TCP/UDP generischenen Proxy zurückgegriffen werden. Dieser arbeitet auf dem Session Layer und kann nur die Header-Informationen auswerten. Es handelt sich dann um ein *Circuit-level Proxy/Gateway*. Im Vergleich zu einem Application-level Proxy ist die Sicherheit geringer, da der Circuit-level Proxy nicht in der Lage ist, die Daten zu interpretieren.

Ein allgemeines Problem ist, dass viele Anwendungen auf generische Protokolle wie HTTP aufsetzen. Weiterhin betreiben einige Anwendungen „Port Hopping“, d.h. sie wechseln den Port wenn der Standardport nicht offen ist.

Eine Anforderung an „Next-generation Firewalls“ ist, dass diese die Analyse von den Daten einer Anwendung unabhängig vom Port und Protokoll ermöglichen.

Konfiguration eines Gateways

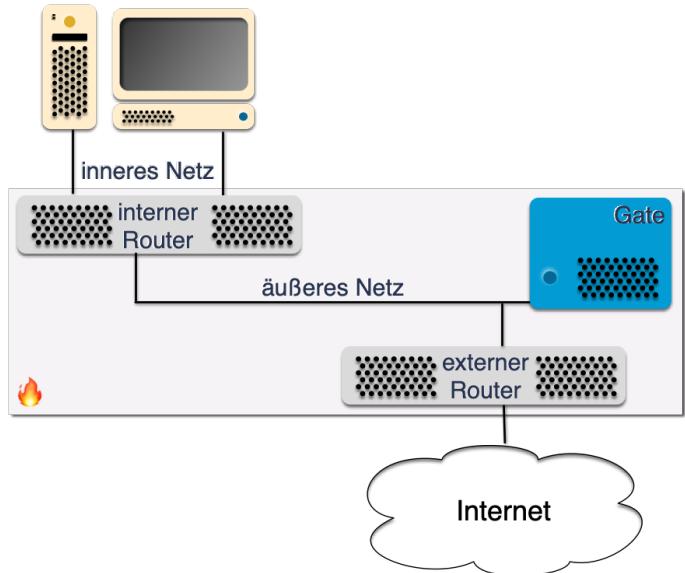
Das Ziel der Konfiguration muss eine minimale angreifbare Oberfläche sein.

- Abschalten aller nicht-benötigten Netzdienste
- Löschen aller nicht benötigter Programme
- Rechte von /bin/sh auf 500 setzen
- Rechte aller Systemverzeichnisse auf 711 setzen
- keine regulären Benutzerkennungen
- root-Login mit Einmal-Passwortsystem
- setzen von Platten- und Prozess-Quotas
- volle Protokollierung, möglichst auf Hardcopy-Gerät
- möglichst sichere, stabile und regelmäßig aktualisierte Betriebssystemversion einsetzen

Screened Subnet

Aufbau

- interner Screening Router als weiterer Schutzwall
 - blockiert Dienste, die nicht einmal bis zum Gate gelangen sollen
 - lässt nur Pakete zum / vom Gate durch
- äußeres Netz realisiert Demilitarisierte Zone (DMZ) für HTTP-Server, Mail-Server, ...



Intrusion Detection Systeme (IDS)

Definition

Ein IDS ist ein Gerät (meist ein speziell konfigurierter Rechner), das vielfältige Techniken zur Erkennung von Angriffen anwendet und Angriffe meldet und ggf. abwehrt, in dem (z.B.) die Firewall automatisch umkonfiguriert wird.

Motivation

- Firewalls alleine sind zu statisch und deswegen häufig nicht ausreichend
- bessere Aufzeichnung und flexiblere Erkennung notwendig
- angepasste Reaktion notwendig

Umsetzung

An verschiedenen, neuralgischen Stellen werden spezielle Sensoren platziert, die (hier) den Netzwerkverkehr überwachen und verdächtige Aktivitäten melden.

55

Miteinander verwandt bzw. typischerweise in einem Produkt zu finden:

- Intrusion Detection (IDS)
- Intrusion Response (IRS)
- Intrusion Prevention (IPS)

IDS-Erkennungstechniken

- Signaturerkennung
- statistische Analyse
- Anomalieerkennung

Probleme

- Fälschlicherweise gemeldete Angriffe (false positives)
- nicht gemeldete Angriffe (false negatives) (insb. bei neuartigen Angriffen)
- Echtzeitanforderung, insb. bei Hochgeschwindigkeitsnetzen
- Aufzeichnung bei Netzwerken mit Switches (⇒ spez. SPAN Port)
- Sensoren sollen unbeobachtbar sein (stealth)

Übung: Firewalls

1. Was sind Vorteile eines Dual Homed Host gegenüber einem Paketfilter? Was sind die Nachteile?
2. Benennen Sie zwei konzeptionelle Grenzen von Firewalls. d.h. zwei Szenarien gegen die Firewalls nicht schützen können.
3. Für welche der folgenden Cybersicherheitsstrategien können Firewalls eingesetzt werden:
 1. Angriffe vermeiden
 2. Angriffe erkennen
 3. Angriffe abwehren/Angriffen entgegenwirken
 4. Reaktion auf Angriffe
4. Sie werden beauftragt die Firewall so einzurichten, dass Mails mit Schadsoftware nicht durchgelassen werden. Wie reagieren Sie?

3. THE ONION ROUTER (TOR)

Prof. Dr. Michael Eichberg

Tor (The Onion Router)

- Anwendungsunabhängiger **low-latency Anonymisierungsdienst für TCP-Verbindungen**, der den Standort und die IP des Nutzers verschleiert
- Typische Anwendung: anonymes Surfen im Internet und Instant Messaging (z.B. Briar)
- Frei und Open Source
- gegründet 2002, öffentlich nutzbar seit 2003, Code seit 2004 frei verfügbar
- Baut ein *Overlay-Netzwerk* auf
- Grundlegendes Prinzip: Onion Routing

low-latency: Die Verzögerung durch die Anonymisierung ist so gering, dass Tor für Instant Messaging und das Surfen im Internet verwendet werden kann.

Overlay-Netzwerk: Tor baut ein eigenes Netzwerk auf, welches auf dem Internet aufsetzt. Die Verbindungen zwischen den Tor-Knoten werden von Tor zusätzlich verschlüsselt.

Tor - Verwendung und Sicherheit

- legale/intendierte Nutzungen: Nutzer mit allg. Datenschutzbedürfnissen, *Whistleblowers*, Dissidenten, Journalisten, ...
- illegale Zwecke (Darknet).

Es wird geschätzt, dass etwa 80% des Datenverkehrs im Zusammenhang mit dem Zugriff auf Kinderpornografie steht. Solche Schätzungen sind allerdings mit Vorsicht zu genießen!

- Mehrere Sicherheitslücken wurden in der Vergangenheit gefunden und geschlossen. Die Angriffe [3] waren:
 - DoS Attacken
 - Deanonymisierungsattacken
 - Identifikation von *Onion Services* (aka *Hidden Services*)

[3] Aufstellung von Angriffen auf Tor.

60

[Sicherheitslücke gefunden in 2013] Wenn ein einzelner Nutzer Tor über einen längeren Zeitraum [3 bis 6-Monate, abhängig von einigen Faktoren] regelmäßig nutzt, ist es fast sicher, dass er de-anonymisiert werden kann.

[Übersetzt mit DeepL.]

—<https://www.infosecurity-magazine.com/news/tor-is-not-as-safe-as-you-may-think/>

Surface web vs. Deep web vs. Dark web

Tor - Hintergrund

- Die grundlegende Idee ist es eine Trennung zwischen der Quelle und dem Ziel des Datenverkehrs zu schaffen.
- Der Datenverkehr wird über *mehrere Knoten (Relays)* umgeleitet, die jeweils nur den vorherigen und den nächsten Knoten kennen. Der Weg den ein Datenpaket nimmt, wird als *Circuit* oder *Path* bezeichnet.
- Der Pfad wird dazu vorher ausgewählt und der gesamte Datenverkehr entsprechend des Pfades verschlüsselt.
- Tor bietet Anonymität auch für die Serverseite durch *Onion Services* (auch *Hidden Services*), die nur über eine von Tor vergebene Onion-Adresse erreicht werden können.

Tor - Bedrohungsmode

Tor bietet Schutz für folgenden Angreifern: Einem Angreifer dem es gelingt ...

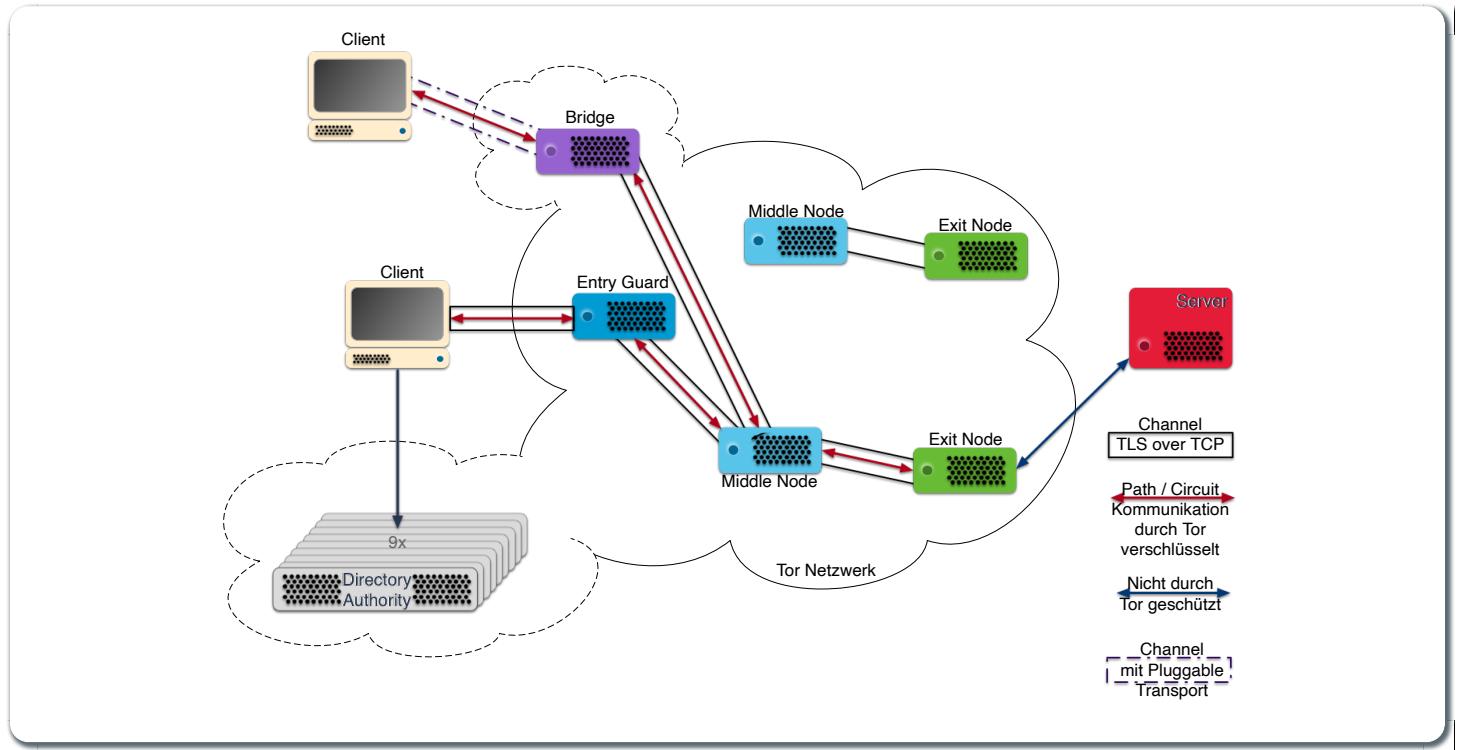
- einen Teil der Kommunikation zu beobachten und
- nur einen Teil der Tor-Knoten zu kontrollieren, indem er entweder einen eigenen Tor-Knoten (*Relay*; früher *Onion-Router*) betreibt oder einen bereits laufenden Knoten kompromittiert.

Warnung

Folgendes Szenario ist nicht abgedeckt: Ein Angreifer, der beide Enden der Kommunikation, den **Entry Guard** und den **Exit Node** überwachen kann.

Gegen solche Angreifer bietet Tor keine Anonymität.

Tor - Aufbau



63

Tor-Knoten: Rechner, die das Tor-Netzwerk bilden. Es gibt drei Arten von Tor-Knoten:

- *Entry Nodes* (auch *Guard Nodes*): Diese Knoten sind die ersten Knoten in der Kette. Sie kennen die IP-Adresse des Clients. Sie können den Datenverkehr nicht entschlüsseln. Sie können aber sehen, dass der Datenverkehr von einem bestimmten Client kommt.
- *Middle Nodes*: Diese Knoten sind die mittleren Knoten in der Kette. Sie kennen weder die IP-Adresse des Clients noch die IP-Adresse des Ziels. Sie können den Datenverkehr nicht entschlüsseln. Sie können aber sehen, dass der Datenverkehr von einem bestimmten Entry Node kommt und an einen bestimmten Exit Node geht.
- *Exit Nodes*: Diese Knoten sind die letzten Knoten in der Kette. Sie kennen die IP-Adresse des Ziels. Sie können den Datenverkehr entschlüsseln. Sie können aber nicht sehen, von welchem Entry Node der Datenverkehr kommt.
- *Bridge Nodes*: Diese Knoten sind *Entry Nodes*, die nicht bzw. nicht vollständig öffentlich bekannt sind. Diese dienen ggf. dazu in Ländern, in denen Tor blockiert wird, den Zugang zu Tor zu ermöglichen. Sollte eine Verbindung zu einer Bridge nicht hergestellt werden können, aufgrund der Struktur der Nachrichten - zum Beispiel aufgrund der Verwendung von *Deep Packet Inspection* - dann ist es möglich diese mit Hilfe von *Pluggable Transports* zu verschleiern.

Tor-Netzwerk: besteht aus mehreren tausend Tor-Knoten. Viele Knoten sind freiwillig betriebene Knoten.

Circuit/Path:

Ein Circuit besteht typischerweise aus drei Knoten: *Entry Node*, *Middle Node* und *Exit Node*. Mehr Knoten sind möglich, haben jedoch nur einen geringen Einfluss auf die Sicherheit. Die Übertragung der Daten zwischen diesen Knoten erfolgt verschlüsselt. In welcher Form die Daten vom *Exit Node* zum Ziel übertragen werden, ist nicht Teil von Tor. Hat der Client eine verschlüsselte Verbindung initiiert (HTTPS), dann ist auch der Datenverkehr zwischen dem Exit Node und dem Ziel (noch) verschlüsselt ansonsten nicht und der Exit Node kann den Datenverkehr lesen.

Directory Authority:

Knoten, die die Liste der aktiven Tor-Knoten verwalten. Diese Liste wird von allen Tor-Knoten regelmäßig in Hinblick auf das *Consensus Document* bzgl. der Knoten und deren Eigenschaften sowie Zustand abgefragt. Das *Consensus Document* wird von den *Directory Authorities* einmal pro Stunde gemeinsam erstellt und beschreibt die relevanten Eigenschaften jedes Tor-Knotens. Die Authentizität des *Consensus Document* wird durch die Signaturen der *Directory Authorities* nachgewiesen.

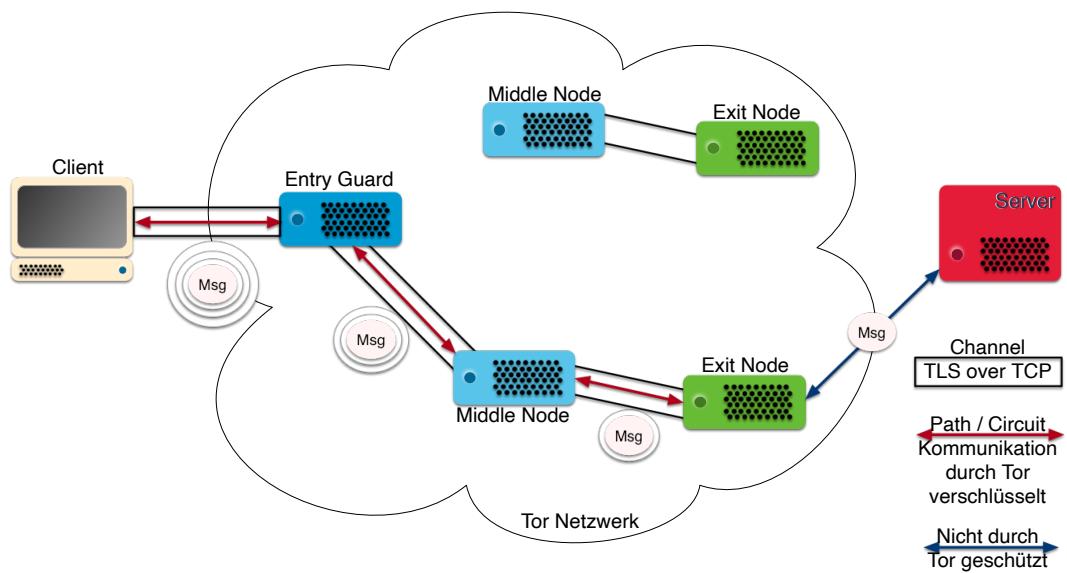
Es gibt (Stand 2023) 9 *Directory Authorities*.

Spezifikation

Hinweis

In älteren Dokumenten wird der *Client* auch als *Onion Proxy (OP)* bezeichnet und die Tor-Knoten als *Onion Router (OR)*. Die Tor-Knoten ( *Nodes*) werden auch als *Onion Relay* bezeichnet.

Onion Routing

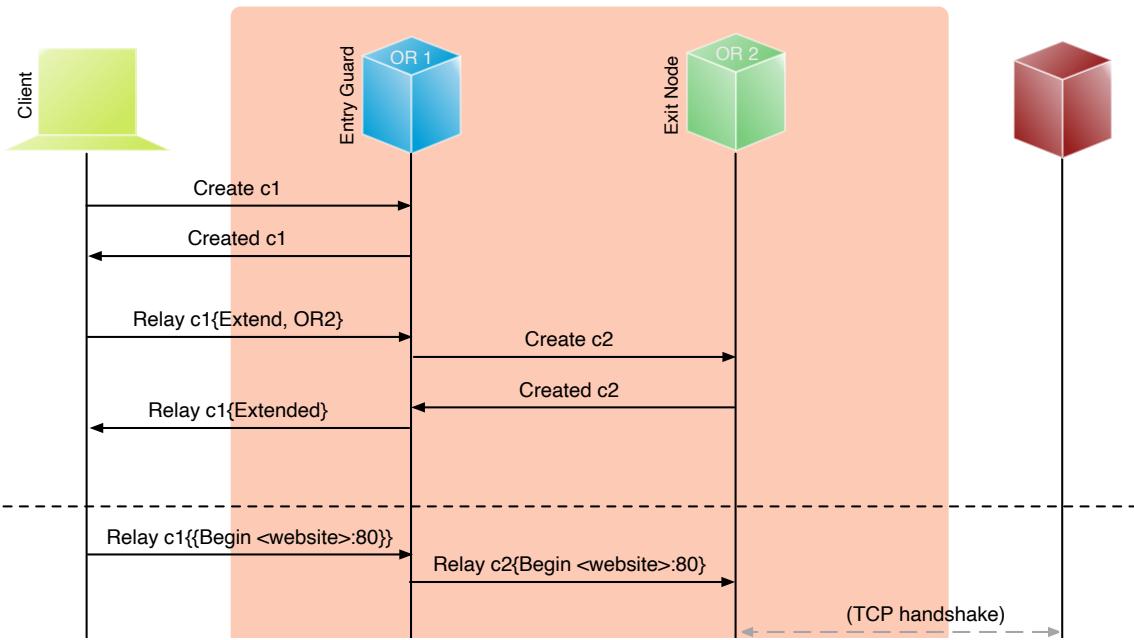


64

■ **Onion Routing:** bedeutet, dass die Datenpakete mehrfach verschlüsselt werden. Jeder Tor-Knoten kann nur die Verschlüsselungsschicht entfernen, für die er den Schlüssel hat. Die Schlüssel werden mit dem Client während des Aufbaus des Circuits ausgehandelt. Es gibt für jeden Tor-Knoten einen eigenen Schlüssel und die Nachrichten werden in umgekehrter Reihenfolge der Tor-Knoten entlang des Pfades verschlüsselt. d.h. die Verschlüsselung für den Entry Node wird als letztes angewendet, da diese als erstes entfernt wird.

Cells: sind die Datenpakete, die zwischen den Tor-Knoten ausgetauscht werden. Cells sind immer 512Byte groß, um es unmöglich zu machen anhand der Größe der Datenpakete Rückschlüsse auf die Daten zu ziehen.

Initiierung eines Circuits (konzeptionell)



65

Jeder Tor-Knoten verfügt über mehrere Keys. Für den Aufbau der Verbindung werden die *Onion Keys* verwendet. Mit Hilfe dieser werden die initialen Datenpakete mittels Public-Key Kryptografie verschlüsselt. Dies wird benötigt, um den AES Key - einer pro Knoten - der für den eigentlichen Versand benötigt wird, auszuhandeln und sicher zu übertragen.

In der Grafik wird der Aufbau eines Circuits mit zwei Tor-Knoten dargestellt. Der Client kennt die Onion Keys der Tor-Knoten (**OR1** und **OR2**). Die Onion Keys werden verwendet, um die *Create* Zelle zu verschlüsseln. Der Entry Node verwendet diese Onion Keys um die *Create* Zelle zu entschlüsseln und den gemeinsamen Schlüssel zu erzeugen. Um einen längeren Pfad aufzubauen, muss der Client ggf. einfach mehrere **Extend** Nachrichten versenden. Erhält ein Knoten eine *Relay* Nachricht, dann kann der Knoten diese mit dem mit ihm ausgehandelten AES Key entschlüsseln und die Nachricht weiterleiten. Er kann den Inhalt (zum Beispiel eine weitere *Relay* Nachricht oder eine *Extend* Nachricht) nicht lesen.

Tor Relays in Deutschland

The New York Times | Settings | torbrowser | Set TOR Ext. | Congratulations! | tor configuration | tor network | TOR Nodes | TOR Project | Relay Search | Onion Available

<https://metrics.torproject.org/rs.html#search/country/de>

Tor Metrics

News Sources Services Development Research About

Home Users Servers Traffic Performance Onion Services Applications

Home > Services > Relay Search > Search for country:de

Relay Search

country:de

Show 10 entries

Nickname [†]	Advertised Bandwidth	Uptime	Country	IPv4	IPv6	Flags	Add. Flags	ORPort	DirPort	Type
• sn0rlax (1)	70.83 MiB/s	49d 21h	DE	176.9,85.41	2a01:4f8:151:2324::2	•	•	8080	0	Relay
• pointy (1)	70.78 MiB/s	2h 58m	DE	46.4,20.30	2a01:4f8:221:39a2::2	•	•	443	0	Relay
• StayStrongRelay01 (1)	68.17 MiB/s	29d 13h	DE	162.55.91.19	-	•	•	443	0	Relay
• setsun (1)	67.19 MiB/s	15d 12h	DE	144.76.200.80	-	•	•	9001	0	Relay
• magic48relay (1)	67 MiB/s	14h 54m	DE	144.7,62.23.174	-	•	•	9001	0	Relay
• b69 (0)	63.63 MiB/s	19d 21h	DE	144.7,64.31.99	-	•	•	9001	0	Relay
• relayanon1172 (1)	62.99 MiB/s	31d 15h	DE	185.220.101.172	-	•	•	11172	0	Relay
• m4rs (1)	62.16 MiB/s	1y 4d0	DE	142.132.204.112	2a01:4f8:261:5099::2	•	•	4443	0	Relay
• as21250 (9)	59.59 MiB/s	16d 13h	DE	185.177.206.67	2a05:4741:25:8000:fefe::130	•	•	443	0	Relay
• FreeBSD2324 (1)	58.66 MiB/s	12d 18h	DE	148.251.125.117	2a01:4f8:210:2073::	•	•	9001	0	Relay
Total	28134.98 MiB/s									

Showing 1 to 10 of 1,0992 entries

Previous [1](#) [2](#) [3](#) [4](#) [5](#) ... [200](#) Next

[†]The number shown in parentheses is the total effective family size for the relay including the relay itself. A size of 1 indicates that the relay does not have any other effective family members. Note that bridge relays do not advertise family members in their descriptors and so there is no value shown for bridge relays.

Information for relays was published: 2024-01-02 07:00:00 UTC.

Information for bridges was published: 2024-01-02 06:56:30 UTC.

Onionoo version: 8.0/d2c1281

66

Flags

Beschreibung jedes Tor-Knotens in Hinblick auf die Rolle des Knotens im Tor-Netzwerk. Zum Beispiel: kann der Knoten als Entry Node verwendet werden? Ist der Knoten schnell genug um als Exit Node verwendet zu werden?

Auszug wichtiger *Flags*:

- | | |
|-----------------|---|
| HSDir: | Ein Router ist ein <i>v2 Hidden Service Directory</i> |
| Running: | Eine Authority konnte sich innerhalb der letzten 45 Minuten mit dem Router verbinden. |
| Stable: | die gewichtete Zeit zwischen zwei Fehlern (<i>weighted MTBF</i>) ist größer als 7 Tage oder größer als der Median aller aktiven Router. |
| Valid: | eine Version von Tor wird ausgeführt, die von den Authorities als aktuell angesehen wird und keine bekannten Schwachstellen aufweist. |

Informationen über Tor Relays

67

Pfade, die über die ganze Welt gehen verhindern, dass der **Entry**– und **Exit**–node beim gleichen Anbieter liegen.

Circuit for nytime... sciiyd.onion

Tor Circuit

- This browser
-  **Finland (guard)** 65.21.94.13, 2a01:4f9:3b:468e::13
-  **France** 163.5.121.253, 2a04:ecc0:8:a8:4567:906:0:1
-  **United States** 82.165.215.73
- ⋮ Onion site relays
- nytime... sciiyd.onion

New Tor circuit for this site
Your guard node may not change

Jan. 2024 - zu vermeidende Hoster:

Frantech / Ponynet / BuyVM (AS53667)
OVH SAS / OVHcloud (AS16276)
Online S.A.S. / Scaleway (AS12876)
Hetzner Online GmbH (AS24940)
IONOS SE (AS8560)
netcup GmbH (AS197540)
Psychz Networks (AS40676)
1337 Services GmbH / RDP.sh (AS210558)

Tor Exit Nodes

Die Anzahl der Exit nodes ist deutlich kleiner (2. Jan. 2024 - 1314 Einträge) als die Anzahl der Knoten. Dies liegt daran, dass die technischen Anforderungen höher sind (z.B. stabile IP Adressen) und insbesondere daran, dass die Betreiber der **Exit nodes** darauf vorbereitet sein müssen ggf. (zahlreiche) Anfragen von den Behörden zu bekommen. [4]

Geolocation data from IPRegistry.co (Product: API, real-time)

 IP ADDRESS:	130.149.80.199	 ISP:	Verein Zur Foerderung Eines Deutschen Forschungsnetzes E.V.
 COUNTRY:	Germany 	 ORGANIZATION:	Tu Berlin, Campus Network
 REGION:	Berlin	 LATITUDE:	52.53126
 CITY:	Berlin	 LONGITUDE:	13.38782

[4] Tor Exit Node Guidelines

68

Reverse IP Lookup für 130.149.80.199 durchgeführt mit [IP Location Service](#).

Tor Relays: Exit Policy

Jeder **Node** legt in seiner **Exit Policy** genau fest welchen Datenverkehr weiterleiten möchte:

- Es gibt offene Exit Nodes, die alle Anfragen weiterleiten.
- Es gibt Knoten, die die Daten nur an weitere Tor-Knoten weiterleiten.
- Es gibt Knoten, die nur bestimmte Dienste (z.B. HTTPS) weiterleiten.
- Es gibt „private Exit Nodes“, die nur zu einem bestimmten Netz Verbindungen aufbauen.

Onion Services/Hidden Services

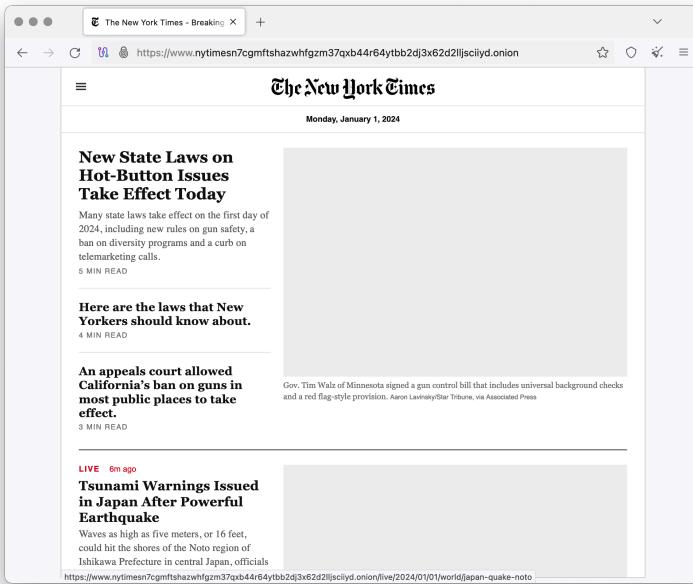
- Server, die Anfragen nur aus dem Tor-Netzwerk annehmen, werden als *Onion Services* (bzw. *Hidden Services*) bezeichnet.
- **.onion** ist eine *Pseudo-Top-Level-Domain*, die für Onion Services verwendet wird.
- Onion Services können nur über das Tor-Netzwerk erreicht werden.

Onion-Adresse der New-York-Times im Tor Netzwerk:

<https://nytimesn7cgmftshazwhfgzm37qxb44r64ytbb2dj3x62d2lljsciyyd.onion> (Aus Deutschland faktisch nicht nutzbar.)

Tor Browser

Standardanwendung für den Zugriff auf das Tor-Netzwerk.



Ergebnis nach mehreren Minuten Wartezeit und zwei Versuchen überhaupt eine Verbindung aufzubauen.

Das Tor-Netzwerk erlaubt ggf. das Setzen des **Exit Nodes**, um zum Beispiel geografische Sperren zu umgehen. Entsprechende Dienstanbieter können dies jedoch leicht erkennen, da die Knoten des Tor Netzwerkes bekannt sind (<https://check.torproject.org/torbulkexitlist>) und verweigern dann den Zugriff.

Tor

- ✓ Schützt vor der Analyse des Datenverkehrs.

Von **SecureDrop** wird zum Beispiel für Whistleblower empfohlen sich mit dem SecureDrop Service über Tor zu verbinden und erst dann Dokumente hochzuladen.

- ✓ Tor Browser schützt relativ effektiv vor Website-Fingerprinting.
- ❗ Teilweise sehr langsam (insbesondere bei Onion Services).
- ❗ Monitoring des Netzwerks ist an den Grenzen möglich.
- ❗ Ende-zu-Ende Korrelation von Datenverkehr ist möglich.
- ❗ Die Anonymität hängt auch von der Anzahl der Nutzer ab.

72

Website Fingerprinting

Website Fingerprinting ermöglicht es die besuchten Websites anhand des Datenverkehrs zu identifizieren. Dabei wird nicht der Inhalt der Datenpakete analysiert, sondern die statistischen Eigenschaften des Datenverkehrs. Wie groß sind die Datenpakete (d.h. die ausgelieferten Dateien)? Wie viele Datenpakete werden wann verschickt? Wie lange dauert es bis ein Datenpaket verschickt wird (d.h. Geschwindigkeit der Webseite)? Wie lange dauert es bis ein Datenpaket ankommt?

(Cross-)Browser Fingerprinting

Durch das Sammeln vieler (auch kleiner) Informationen über den/die Browser und das Betriebssystem kann ein für praktische Zwecke hinreichend eindeutiger Fingerabdruck erstellt werden. Dieser kann dann zur Identifikation des Nutzers verwendet werden.

Kleiner Auszug aus den möglichen Informationen:

- System Fonts
- Werden Cookies unterstützt?
- Betriebssystem
- Betriebssystem Sprache
- Keyboard layout
- Art/Version des Browsers
- verfügbare Sensoren: Beschleunigungssensor, Näherungssensor, Gyroskop
- verfügbare Browser Plugins
- HTTP-Header Eigenschaften
- CPU Klasse
- HTML 5 Canvas Fingerprinting
- Unterstützung von Multitouch

Monitoring des Netzwerks an den Grenzen

Hat in der Vergangenheit dazu geführt, dass Nutzer von Tor-Netzwerken identifiziert werden konnten.

Ende-zu-Ende Korrelation von Datenverkehr

Auch als *Traffic Confirmation* bekannt. Diese Art von Attacke ist möglich, wenn *Relays* am Anfang und am Ende der Verbindung kontrolliert werden. Die Angreifer können dann den Datenverkehr an beiden Enden beobachten und die Datenpakete korrelieren z.B. basierend auf statistischen Informationen über die Zeitpunkte und Volumen von Datenflüssen.

- Ist es für *Onion Services* (.onion) notwendig auf HTTPS zu setzen oder reicht HTTP für eine sichere Kommunikation? Ist die Verwendung von HTTPS ggf. sogar problematisch?
- Warum führt der Tor Browser keine DNS Lookups durch? Warum ist dies wichtig und wer kann/muss es stattdessen machen?
- Warum hätte das Abschalten von TOR auf kriminelle Aktivitäten im Internet vermutlich nur einen geringeren Einfluss?
- Wie vergleichen sich Proxies und Tor-Knoten?
- Wie unterscheidet sich Tor von einem VPN?
- Macht es Sinn ein VPN über Tor oder anders herum zu betreiben?
- Was passiert wenn eine Angreifer in der Lage ist $50\% + 1$ der **Directory Authority** Server zu kontrollieren?