

Datenspuren - Privatsphäre war gestern
ein Symposium des CCC Dresden

Mixmaster & Co.

Ergebnisse der Diskussionsrunde zu Anonymisierungsdiensten

Jens Kubieziel (jens@kubieziel.de)

2005-05-08

Inhaltsverzeichnis

| | | |
|----------|--|----------|
| 1 | Mixmaster & Co. | 3 |
| 1.1 | AN.ON bzw. JAP | 3 |
| 1.1.1 | Funktionsweise und Benutzung | 3 |
| 1.1.2 | Betrieb eines Servers | 4 |
| 1.1.3 | Sonstiges | 4 |
| 1.1.4 | Weitere Informationen | 4 |
| 1.2 | Tor | 4 |
| 1.2.1 | Funktionsweise und Benutzung | 4 |
| 1.2.2 | Probleme beim Betrieb eines Servers bzw. der Nutzung | 8 |
| 1.2.3 | Sonstiges | 8 |
| 1.2.4 | Weitere Informationen | 9 |
| 1.3 | Mixmaster und Mixminion | 9 |

1 Mixmaster & Co.

Im Rahmen des Symposiums „Datenspuren - Privatsphäre war gestern“ fand eine Diskussionsrunde zu Anonymisierungsdiensten statt. Hier sollten Probleme und Lösungsvorschläge beim Betrieb und der Nutzung von anonymisierenden wie auch pseudonymisierenden Diensten besprochen werden. Die Runde wurde vom Autor des Dokuments geleitet.

Das Dokument stellt eine Zusammenfassung und weitere Diskussionsgrundlage dar. Sollte der Leser Fehler finden oder andere Hinweise haben, würde ich mich über eine Nachricht an jens@kubieziel.de freuen.

Hauptgegenstand der zweistündigen Runde waren JAP/AN.ON, Tor und die Remailer Mixmaster bzw. Mixminion.

1.1 AN.ON bzw. JAP

1.1.1 Funktionsweise und Benutzung

AN.ON entstand aufgrund einer Zusammenarbeit der TU Dresden mit dem Unabhängigen Landeszentrum für Datenschutz Schleswig-Holstein (ULD) sowie der FU Berlin. Ziel des vom BMWA geförderten Projektes war die Schaffung einer Plattform für die anonyme Kommunikation.

Die Software basiert auf der Programmiersprache Java und ist plattformunabhängig. Der Nutzer installiert auf seinem Rechner ein Clientprogramm namens JAP. Dieses arbeitet als Proxy und nimmt über das Internet Verbindung zum Dienst auf.

AN.ON besteht dann aus verschiedenen so-genannten Mixen bzw. Mixkaskaden. Das Prinzip wurde 1981 von David Chaum erstmalig beschrieben. Ein Datenpaket wird durch verschiedene Mixe geschickt und mit den öffentlichen Schlüsseln aller Mixe verschlüsselt. Jeder Mix kann dann nur den Teil entschlüsseln und einsehen, der mit seinem Schlüssel chiffriert wurde. Eine genaue Beschreibung findet sich auf den Seiten des Projekts: <http://anon.inf.tu-dresden.de>

Für die Vertrauenswürdigkeit der Kommunikation ist insbesondere wichtig, dass die Mixe in der Hand unabhängiger Betreiber liegt. Hierin lag auch die Ursache für den erfolgreichen Angriff gegen den Dienst. Mitte 2003 fragte das BKA Daten eines Nutzers mittels eines gerichtlichen Beschlusses an. Die Projektbetreiber bauten in die Kaskade Dresden-Dresden eine Überwachungsfunktion ein. Gleichzeitig legte das ULD Beschwerde gegen den Beschluss ein und bekam auch Recht. Jedoch erzwangen Beamte die Herausgabe des betreffenden Datensatzes. In einem späteren Verfahren wurde der obige Beschluss komplett aufgehoben.

Dieser Vorgang führte zu vielerlei Diskussionen. Einige Nutzer sind der Meinung, dass die Kaskade Dresden-Dresden nicht mehr vertrauenswürdig ist. Weiterhin kann damit festgestellt werden, Betreiber von Anonymisierungsdiensten aufgrund eines gerichtlichen Beschlusses (nach Grundlage von §100 a und b StPO) eventuell gezwungen werden können, die Anonymisierung aufzuheben.

1.1.2 Betrieb eines Servers

In der Diskussion wurde schnell klar, dass der Betrieb eines solchen Mixes für die meisten Nutzer nicht machbar ist. Denn der entstehende Netzverkehr liegt im Terabyte-Bereich. Somit kämen als Betreiber derzeit nur größere Institutionen in Frage. Dies wurde von den Teilnehmern ebenfalls so gesehen.

Weiterhin wäre der oben genannte rechtliche Aspekt ein Hinderungsgrund. In der Regel sind Organisationen mit eigenen Rechtsabteilungen bzw. geeigneter rechtlicher Beratung besser gegen staatliche Eingriffe gewappnet als private Nutzer.

1.1.3 Sonstiges

In der Diskussion kam die Frage auf, ob die Clientsoftware weiterentwickelt wird. Laut der Projektseite stammt die letzte Version vom 2005-05-08. Somit kann davon ausgegangen werden, dass die Entwicklung nicht eingestellt wurde.

1.1.4 Weitere Informationen

Weitergehende Informationen kann man auf folgenden Seiten finden:

- <http://anon.inf.tu-dresden.de/>
- http://de.wikipedia.org/wiki/Java_Anon_Proxy
- <http://www.datenschutzzentrum.de/projekte/anon/>
- <http://www.jurpc.de/aufsatz/20040140.htm>
- <http://www.inf.tu-dresden.de/~hf2/anon/>

1.2 Tor

1.2.1 Funktionsweise und Benutzung

In der Diskussion stellte sich heraus, dass die meisten wenig oder nichts vom Torprojekt gehört haben. Daher will ich zunächst etwas genauer auf das Design eingehen.

Tor ist ein Akronym und steht für „The Onion Router“ oder auch „Tor Onion Router“. Seit 2004 wird das Projekt von der Electronic Frontier Foundation gefördert und ist unter der URL <http://tor.eff.org/> erreichbar. Zu Beginn der Entwicklung erfolgte die Förderung durch die Defense Advanced Research Project Agency (DARPA) bzw. US Navy.

Die Software kann von der Projektseite heruntergeladen werden. Die Installation und Anpassung ist sehr einfach und ebenfalls detailliert auf den Webseiten beschrieben. Tor kann als SOCKS-Proxy im Browser (oder auch anderen Anwendungen) eingerichtet werden.¹

Wenn nun jemand beispielsweise die Seite <http://www.kubieziel.de/> besuchen will, bezieht der Torclient zunächst eine Liste von Torknoten (Onionrouter) vom Directoryserver. Dieser Server hält Informationen vor, welche Knoten online sind, wo sie sich befinden, deren Schlüssel und weitere Informationen vor. Gleichzeitig kontrolliert er auch, welche Knoten sich zum Netzwerk verbinden können. Um hierbei Engpässe zu vermeiden, werden die Daten in der Regel von anderen Servern zwischengespeichert (Cache).

Zellen

Die Kommunikation der Torknoten untereinander erfolgt mit Paketen fixer Größe von je 512 Byte. Diese werden als *Zellen* („cells“) bezeichnet. Die Zellen bestehen aus einem Kopf („Header“) und den eigentlichen Daten („Payload“). Im Header steht ein so-genannter Kanalbezeichner² (circID) gefolgt von einem Kommando. Der Kanalbezeichner legt fest, zu welcher Verbindung diese Zelle gehört. Das Kommando beschreibt, was mit dem Payload zu tun ist.

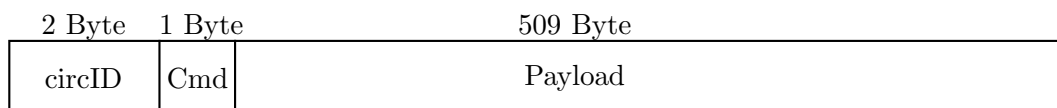


Abbildung 1.1: Aufbau einer Zelle bzw. eines Pakets

Je nach Kommando unterscheidet man *Kontrollzellen* und *Weiterleitungszellen*.

Kontrollzellen werden immer von dem Knoten interpretiert, der sie empfängt. Die Kommandos, die Kontrollzellen tragen können sind:

padding derzeit nur für Keepalive genutzt, wäre auch für Linkpadding nutzbar

create, created wird zum Aufbau eines neuen Kanals genutzt

destroy Schliessen eines Kanals

relay Daten für Weiterleitung (Ende-zu-Ende-Kommunikation)

create_fast, created_fast Aufbau eines Kanals ohne public keys

¹Viele Anwendungen machen zuerst DNS-Anfragen, bevor sie diese an den Proxy weitergeben. Daher ist die Nutzung eines filternden HTTP-Proxy, wie Privoxy, sinnvoll.

²Im Original heißt dies Circuit Identifier. Meiner Meinung nach ist dies passendste Übersetzung.

Weiterleitungszellen tragen Ende-zu-Ende-Daten. Diese Art von Zellen haben einen größeren Header. Neben den bereits bekannten Daten werden hier noch eine StreamID, Checksumme, Länge des Payloads und spezielle Kommandos übertragen. Der Aufbau einer solchen Zelle stellt sich wie folgt dar:

| | | | | | | |
|--------|--------|----------|------------|--------|--------|----------|
| 2 Byte | 1 Byte | 2 Byte | 6 Byte | 2 Byte | 1 Byte | 498 Byte |
| circID | Relay | StreamID | Checksumme | Länge | Cmd | Payload |

Abbildung 1.2: Aufbau einer Weiterleitungszelle

Die StreamID ist der Bezeichner des Datenstroms, die Checksumme dient der Verifikation der Integrität der Daten und das Kommando ist ein spezifisches Relaykommando. Während sich die Zelle durch den Kanal bewegt, wird der spezifische Header und der Payload ver- und entschlüsselt. Als Algorithmus kommt AES (128 Bit) im Countermode zur Verwendung.

Das letzte Feld im Header kann folgende Kommandos tragen:

- relay data
- relay begin
- relay end
- relay teardown
- relay connected
- relay extend(ed)
- relay sendme
- relay drop

Aufbau von Verbindungskanälen

Das Proxyprogramm beim Nutzer baut inkrementell eine Verbindung auf und verhandelt mit jedem Knoten im Kanal einen symmetrischen Schlüssel. Zunächst sendet der Proxy einen **create**-Aufruf an den ersten Knoten. Die Authentifizierung erfolgt hierbei mittels Diffie-Hellman. Wenn ein Verbindungskanal etabliert ist, erfolgt alle Kommunikation mit dem ausgehandelten Schlüssel.

Um den Kanal weiter auszubauen, wird nun ein **relay extend**-Aufruf an den ersten Knoten geschickt. Hierin ist die Adresse des zweiten Knoten angegeben und die erste Hälfte des Diffie-Hellman-Austauschs wird mit dem Schlüssel des zweiten Knoten chiffriert. Wenn der erste Knoten dies erhält, kopiert er diese Angaben in einen **create**-Aufruf und gibt diesen an den neuen Knoten weiter. Dieser antwortet mit **created**, der

zweite kopiert das in einen `relay extended`-Aufruf und gibt das an den Knoten des Nutzers zurück. Nunmehr teilen sich der Nutzer und der zweite Knoten einen gemeinsamen Schlüssel. Für jeden weiteren Knoten wird ebenso verfahren.

Um nun Verzögerungen im Netzverkehr zu vermeiden, baut der Torclient vorsorglich verschiedene solche Kanäle auf. Immer wenn ein Kanal beendet wurde, wird gleichzeitig automatisch ein neuer eröffnet. Weiterhin sollen die Clients Kanäle im Abstand von einer Minute rotieren.

Nutzung von Weiterleitungszellen

Wenn einmal ein Kanal eröffnet ist, können Weiterleitungszellen versandt werden. Dazu wählt der Proxy des Nutzers einen offenen Kanal zu einem Exitknoten und verschlüsselt den Payload mit den symmetrischen Schlüsseln der beteiligten Knoten. Jeder Knoten, der dann diese Zelle entgegennimmt, entschlüsselt diese und überprüft, ob diese zu ihm gehört. Falls nicht, leitet er diese weiter. Andernfalls wird die betreffende Anweisung im Payload ausgeführt.

Wenn nun der Exitknoten Daten aus der geöffneten, nichtanonymen Verbindung bekommt, verschlüsselt er diese mit dem Schlüssel des Torproxys vom Nutzer und leitet die Zelle an den nächsten Torknoten weiter. Dieser prüft wieder, ob das Paket zu ihm gehört. Falls nicht, verschlüsselt auch dieser die Zelle mit dem ausgehandelten Schlüssel. Der Proxy des Nutzers kann dann wieder alle Daten entschlüsseln und weiterverarbeiten.

Verbindungsaufbau

Zusammenfassend kann der Verbindungsaufbau wie folgt dargestellt werden: Beim Aufbau einer Verbindung zu einer externen Adresse (Bsp.: `http://www.kubieziel.de/`) wird der Torproxy des Nutzers damit beauftragt. Dieser öffnet den neuesten Kanal (oder öffnet einen neuen) und selektiert eine Strecke zum Exitknoten. Danach wird eine `relay begin`-Zelle mit einer neuen frei gewählten StreamID geschickt. Wenn sich der Exitknoten dann mit der externen Seite verbindet, sendet er `relay connected` zurück. Der Proxy antwortet hierauf wiederum mit einer SOCKS-Antwort und packt nunmehr alles in `relay data`-Zellen, die er dann auf den Weg schickt.

Einige Anwendungen machen zunächst DNS-Anfragen, bevor sie die Daten an den Torproxy weitergeben. Für anonyme Kommunikation ist es wenig erfreulich, wenn erst nach einer IP-Adresse gefragt wird und dann anonym gesurft wird. Daher sollte man zusätzlich noch einen filternden HTTP-Proxy wie z.B. Privoxy installieren. Bei anderen Anwendungen sollte man die IP-Adresse direkt eingeben oder die Anwendung per SOCKS-Wrapper nutzen.³

Konfiguration

Eine wichtige Frage in der Diskussion war, unter welchen Nutzerrechten Tor läuft bzw. ob hier Administratorrechte notwendig sind. Die Software benötigt zum Betrieb nur

³Siehe hierzu auch das Torify-HOWTO auf <http://wiki.noreply.org/noreply/TheOnionRouter/TorifyHOWTO>

Nutzerrechte und wird von den Linuxdistributionen als Nutzer „Tor“ oder auch „debian-tor“ installiert. Auch sind für Tor keinerlei Kernelmodule o.ä. notwendig.

Die Konfiguration erfolgt in der Datei `torrc`. Diese befindet sich in `/etc/torrc` bzw. `~/tor/torrc` (unter Windows auch `Anwendungsdaten\tor\torrc`).

Die Teilnehmer der Diskussion interessierten sich weiter, ob man bestimmte Exit- oder auch Entryknoten fest vorgeben kann. Dazu muss in der Konfigurationsdatei die Variable `exitnodes` bzw. `entrynodes` belegt sein. Als Werte steht eine Liste von Namen der Torservers, die man gern nutzen möchte (Bsp.: `entrynodes foo, bar, quux`). Daneben existiert noch die Konfigurationsvariable `excludenodes`. Sie bestimmt, welche Knoten nicht benutzt werden dürfen.

Unter Umständen ist es auch notwendig, den Verkehr, der über den Torservers geleitet wird zu begrenzen. Hierzu dienen die Optionen `BandwidthRate` und `BandwidthBurst`.

1.2.2 Probleme beim Betrieb eines Servers bzw. der Nutzung

Wie auch schon beim Anonymisierungsdienst der TU Dresden angesprochen, kann ein Hinderungsgrund der entstehende Traffic sein. Viele private Nutzer besitzen lediglich DSL-Zugänge bzw. Server mit begrenztem Freivolumen. Zur Begrenzung des Traffics kann die oben genannte Option genutzt werden. Somit ist eine Kontrolle des Traffic möglich.

Obwohl die Zahl der Server weiter steigt, sind immer noch zu wenige Knoten vorhanden. Dadurch sperren viele Betreiber bestimmte Dienste bzw. Ports. Gerade Tauschbörsen sind oft wegen des großen Volumens gesperrt. Hier könnte eine höhere Zahl an Torknoten Abhilfe schaffen.

Ein weiteres Problem, dass auch von der geringen Zahl an Exitknoten herrührt, ist, dass der entstehende Traffic aus Sicht des externen Serverbetreibers einen Denial-of-Service-Angriff beinhalten kann. Daher sperren einige Betreiber, wie auch die englische Wikipedia, die Exitknoten.

Als Betreiber eines Exitknotens ist man auch mit dem Missbrauch des Dienstes konfrontiert. Daher kann man sowohl in das Blickfeld staatlicher Ermittlungsbehörden wie auch anderer Angreifer geraten. Eine Möglichkeit, dies zu umgehen, ist den Server als Mittelknoten einzustellen. Hier wird der Verkehr einfach zu anderen Torknoten geleitet. Der Server bildet jedoch nie einen Ausgangspunkt aus dem System. Eine weitere Variante wäre, in der Exitpolicy nur bestimmte Dienste zuzulassen. Damit wird auch eine Einschränkung der Missbrauchsfälle zugelassen.

Als rechtliche Unterstützung wurden für die Betreiber von Exitknoten Disclaimer erstellt. Leider wurden diese bislang noch nicht gerichtlich gewürdigt. Insofern fällt eine Aussage über deren Gültigkeit schwer.

1.2.3 Sonstiges

Einer der Teilnehmer war am Lebenslauf des Projektleiters interessiert. Dieser kann auf <http://www.freehaven.net/~arma/cv.html> gefunden werden.

Das Projekt selbst ist seit Oktober 2003 in Betrieb und es sind derzeit ca. 140 Server in Betrieb (Ich hatte hier in der Veranstaltung 200-300 geschätzt. Das wird dann hoffentlich bald werden.). Nach Angaben des Projektleiters gibt es immer mal wieder einen Bericht auf Slashdot, was dann dazu führt, dass die Zahl der Knoten sprunghaft ansteigt und dann auf dem Niveau verharrt. Die Zahl der Nutzer wird auf einige (Zehn-?)Tausende geschätzt. Wegen der Natur des Dienstes kann man nichts genaueres sagen. Der Traffic pro Tag und Knoten liegt bei 1-20 GB.

Ein weiterer wichtiger Punkt war die Geschwindigkeit. Einige Teilnehmer hatten in der Vergangenheit schlechte Erfahrungen mit anderen Diensten gemacht. Der Autor kann bei Verbindungen mit bis zu ISDN-Geschwindigkeit keine Nachteile feststellen. Erst bei höheren Raten ist z.T. ein langsamerer Seitenaufbau zu beobachten. Es ist auch der (subjektive) Eindruck entstanden, dass bis ca. 15 Uhr höhere Geschwindigkeiten zu erzielen sind. Später sind regelmäßig Verlangsamungen zu beobachten. Der Grund hierin dürfte in der verstärkten Aktivität der amerikanischen Bevölkerung. Eventuell haben auch einige Knoten Probleme mit der Namensauflösung, da hin und wieder derartige Meldungen im Browser auftauchen. Generell ist es offensichtlich ein Trade-Off zwischen Anonymität und Geschwindigkeit. Für mehr Geschwindigkeit muss man dann die Anonymität aufgeben.

Im Rahmen von Tests wurden vom Projekt selbst eine 60 MB große Datei insgesamt 108 mal heruntergeladen. Dabei lag die durchschnittliche Downloadzeit um etwa 50 % höher als ohne die Verwendung von Tor. Beim Aufbau einer einzelnen Webseite lag der Median bei 2,7 Sekunden verglichen mit 0,3 Sekunden ohne die Software (Bestwert mit Tor: 0,6 Sekunden).

Für die Nutzer des Browsers Firefox existiert eine Erweiterung namens „Switchproxy“. Hier kann der Nutzer einen oder mehrere Proxys voreinstellen und mit einem Klick zwischen denen wechseln. Dies ermöglicht es, schnell und problemlos Tor auszuschalten oder zu nutzen.

1.2.4 Weitere Informationen

- <http://tor.eff.org/> — Homepage des Projektes
- <http://tor.eff.org/cvs/tor/doc/design-paper/tor-design.html> — Design des Dienstes
- <http://tor.eff.org/cvs/tor/doc/tor-spec.txt> — Protokollspezifikation
- <http://www.cl.cam.ac.uk/users/sjm217/papers/oakland05torts.pdf> - Low-Cost traffic analysis of Tor
- <http://wiki.noreply.org/noreply/TheOnionRouter/TorFAQ> — FAQ zu Tor
- <http://kai.iks-jena.de/bigb/asurf.html> — Informationen zu Tor

1.3 Mixmaster und Mixminion

Die Diskussion zu Tor nahm den größten Teil der Zeit ein. Daher wurde über die Remailer nur kurz gesprochen. Anfänglich gab ich eine kurze Einführung in die historische Entwicklung und bestehenden Remailerprogramme.

Hierauf folgte dann die Frage eines Teilnehmers, warum denn Remailer notwendig wären. Schließlich könnte man auch mit einem Konto bei einem Freemailer Anonymität erreichen. Dies wurde vom Auditorium einhellig verneint, da mind. staatliche Ermittlungsbehörden Zugriff auf die Logdateien der Freemailer erhalten können. Somit wird hier maximal Pseudonymität erreicht.

Mixmaster ist nach meiner Meinung und der des Auditoriums eine ausgereifte und sehr gut nutzbare Software. Lediglich die Integration in Mailprogramme mit grafischer Oberfläche lässt noch Wünsche offen bzw. ist z.T. nicht existent. Hier wäre es wünschenswert, wenn es hier auch Lösungen gäbe.

Die Teilnehmer waren auch einhellig der Meinung, dass die Clientprogramme für Windows, Jack B. Nymble, Quicksilver und John Doe, wenig benutzbar bis unbenutzbar sind. Die Entwicklung ist angabegemäß auf dem Stand von DOS bzw. Windows 3.11 stehen geblieben. Eventuell sollte hier auch ein Projekt für eine Neuentwicklung oder Verbesserung der bestehenden Software initiiert werden.

Mixminion stellt eine Weiterentwicklung der Remailerprotokolle (Type-III-Remailer) dar. Leider wurde die Entwickler des Projektes zu Tor abgezogen, so dass die Entwicklung seit 2004 stillsteht. Gerade aufgrund der kurzen Entwicklungszeit und der fehlenden Projektbetreuung ist noch von Bugs auszugehen. Auch hier wäre es wünschenswert, wenn sich ambitionierte Entwickler fänden, die Beiträge zu dem Projekt leisten.

An diesem Punkt endete die Diskussion. Ich möchte alle Teilnehmern für die angeregte Diskussion danken und hoffe, dass einige den Betrieb eines Servers für sich in Betracht ziehen.