TOR The Onion Router

im Rahmen des Informatik-Seminars „IT-Sicherheit“ bei Dr. Gerd Beuster im Sommersemester 2012

Michael Gaida

Oktober 30, 2012

Inhaltsverzeichnis

1 Einführung

Obwohl wir die Anonymität im Alltag als selbstverständlich empfinden, bewegen wir uns im Internet nur beschränkt anonym. Mindestens der Provider kann Ihre IP-Adresse jederzeit einer Person zuordnen. Auch auf Webseiten werden statistische Daten in umfangreichen Logfiles erfasst, um diese weiter zu verarbeiten oder sie zu verkaufen. Die informelle Selbstbestimmung ist im Internet stark gefährdet. Ist es im Alltag also selbstverständlich, sich nicht jeder fremden Person auszuweisen, ist genau das Interesse vieler Staaten und Organisationen. Neben anderen Projekten, die es erlauben Nachrichten mehr oder weniger anonym zu versenden, bzw. sie zu empfangen, entwickelte ein Entwicklerteam[1] im Jahre 1995/96 das *Onion Routing*. Finanziert wurde die Entwicklung vom *Office of Naval Research*[1][11]. Die Bedrohung des *Electronic Warfare*, also der

[1] Abteilung der United States Navy, zuständig für Koordination und Ausführung wissenschaftlicher Forschung

elektronischen Kriegsführung, machte es nötig eine sichere Methode zu entwickeln, Nachrichten aus Kommandozentralen zu den entsprechenden Einheiten zu schicken, ohne den eigenen Standort dem Gegner zu offenbaren[10]. Im Jahre 1998 wurde das *Freedom Network*, als erstes nicht militärische Installation in betrieb genommen. Weitere Sponsoren waren *Defence Advanced Research Projects Agency* und die *Electronic Frontier Foundation*. Tor entstand aus der Generation 2 des Onion-Routing und wurde im Oktober 2003 in Betrieb genommen und ist bis jetzt fortwährend erreichbar gewesen. Das Projekt wird unter der freien MIT-Lizenz[2] entwickelt und der Quellcode ist über ein öffentliches Versionskontrollsystem[3] erreichbar. Des weiteren gibt es verschiedene Mailinglisten um die Transparenz des Projektes zu erhalten.

2 Anonymität im Internet

2.1 Interessengruppen für Anonymität

Um zu verstehen warum Anonymität im Internet wichtig ist, muss man die potentiellen Interessengruppen einzeln betrachten:

2.1.1 Private Bürger

Für den privaten Bürger ist das natürliche Verlangen der Privatsphäre das entscheidende Argument. Im Alltag ist es unnatürlich sich vor fremden Personen auszuweisen ohne, dass ein berechtigtes Interesse besteht, wie z.B. am Flughafen. Diese natürliche Vorsicht schützt Personen im Alltag vor ungewollter Werbung, erhält die nötige Privatsphäre und schützt somit auch einen Teil der Freiheitsrechte[4]. Im Internet lauern die gleichen, wenn nicht sogar mehr Gefahren, wenn man sich unüberlegt und ohne Einschränkung offenbart. Die Recherche nach Krankheiten z.B. sollte jedem möglich sein ohne, dass eine dritte Person, wie der Provider, dies mitschreibt.

2.1.2 Gewerblich

Im Gewerbe ist die Privatsphäre nicht der entscheidende Punkt. Hier wird sogar versucht so viele privaten Daten wie möglich von Kunden und Konkurrenten zu speichern, um diese weiter zu verarbeiten oder gar zu verkaufen. So ist es Unternehmen, wie ebay und Google möglich Kunden gezielte Artikel anzubieten und Werbung zu schalten. Für Firmen ist jedoch der Punkt der Netzwerksicherheit interessant. Anonyme Kommunikation ist entscheidend, wenn es darum geht sich den Konkurrenten nicht zu offenbaren, hinsichtlich des Standorts oder der Kommunikationspartner[2].

[2] http://www.opensource.org/licenses/mit-license.php
[3] https://git.torproject.org
[4] Meinungsfreiheit (Art. 5 Abs. 1 GG), Religionsfreiheit (Art. 4 Abs. 1 und 2 GG), Berufsfreiheit (Art. 12 Abs. 1 GG)

2.1.3 Regierung

Regierungen versuchen inzwischen so viele Daten wie möglich von den Bürgern zu sammeln. Gründe hierfür sind die Strafverfolgung und das Überwachen der Bürger. Diese zweifelhaften Argumente jedoch machen es schwierig Regierungen von der Wichtigkeit der Anonymität der Bürger zu überzeugen. In vielen Staaten ist es der Regierung wichtiger Inhalte zu zensieren um die Bürger vor Informationen zu schützen. Es sind jedoch Länder wie z.B. die USA, die ihrerseits wiederum Interesse für die Anonymität zeigen und das Projekt erst ins Leben gerufen haben[11]. Kommunikation zwischen Regierungen oder interne Kommunikationen müssen vor *Traffic Analyse*(3.4.5 Seite 13) geschützt werden. So wurde die Entwicklung des *Onion Routing* von der amerikanischen Regierung voran getrieben, um dem Electronic Warefare vorzubeugen[12]. Regierungen haben also auch einen hohen Bedarf an Anonymität, wenn man es aus dem richtigen Blickwinkel betrachtet.

2.1.4 Blockierte Nutzer

Die Gruppe der blockierten Benutzer ist die Gruppe, bei der das Interesse der Anonymität mit am höchsten und ohne negative Seiteneffekten besteht. Sie leiden unter der Zensur ihrer Regierung und Informationen aus der Welt werden ihnen vorenthalten. Wenn die Regierung nicht weiß, auf welche Seite eine Person will, kann sie auch nicht aktiv blockiert werden. Diese Gruppe liegt im vordergründigen Interesse des TOR Projekts. Nur so können Journalisten in Ländern wie China ihre Arbeit verrichten ohne blockiert zu werden oder sich sogar in Gefahr zu begeben.

2.2 Politische Sicht

2.3 Internetzensur

Internetzensur wird in vielen Ländern der Welt betrieben. Hier unterscheidet man vier Gruppen:

- Freier Zugang
- Teilweise zensiert
- Überwacht
- Zensiert

Eine vollständige Zensur besteht derzeit in folgenden Ländern[5]

- China
- Nord Korea
- Burma

[5] http://en.rsf.org/

- Vietnam
- Iran
- Turkmenistan
- Usbekistan
- Saudi Arabien
- Ägypten
- Kuba

In Deutschland gab es bisher nur teilweise Zensur. Beispiele sind hier rotten.com und youporn.com[6]. Beide Seiten sind nach Rechtsstreiten heute wieder erreichbar[7]. Im Jahre 2007/2008 gab es Überlegungen Google zu sperren, da von dort aus auf Seiten wie eben youporn und rotten verwiesen wird. Dies wurde aber nie durchgesetzt[8]. Am 17. April 2009 trat das Zugangserschwerungsgesetz[6] in Kraft. Es sieht eine Sperrung von allen Webseiten vor, die sich auf einer vom BKA erstellten Liste befinden. Diese Liste sollte jeden Tag aktualisiert werden. Bedenklich ist jedoch, dass die Liste ausschließlich von BKA erstellt und herausgegeben wird. Dies wurde jedoch nie in die Praxis umgesetzt und das Zugangserschwerungsgesetz wurde im April 2011 aufgehoben[9].

3 Technische Methoden des Onion Routings

3.1 Begriffsdefinitionen

Relay Ein drittes System, welches bei einer Kommunikation als Zwischenstelle dient. Daten werden von Alice an das Relay geschickt und von diesem an Bob weitergeleitet. Damit gibt es keine direkte Verbindung zwischen Alice und Bob

Onionproxy Client, welcher sich mit dem TOR-Netzwerk verbindet.

Verzeichnisserver Netzwerkknoten mit hohem Vertrauensstatus. Sie verwalten die Daten zu weiteren Knoten des TOR-Netzwerks

Entry-Node Knoten über den in das Tor-Netzwerk eingestiegen wird

Middle-Node Knoten der Daten von einem anderen Knoten bekommt und diese weiterleitet

Exit-Node Knoten über den das Tor-Netzwerk verlassen wird und die Nachricht an den Empfänger sendet

[6] Beschluss von 17.10.2007, Frankfurt am Main, Aktenzeichen 2-06 O 477/07
[7] Urteil von 08.02.2008, Aktenzeichen 3-12 O 171/07
[8] OLG Frankfurt am Main, Beschluss vom 22.1.2008, Aktenzeichen 6 W 10/08
[9] http://www.tagesschau.de/inland/internetsperren118.html

Onionrouter Knoten im TOR-Netzwerk. Dieser kann ein Entry-Node, Middle-Node, Exit-Note sein, dieses ist aber vom Benutzer zu konfigurieren

Bridge-Relay Wie ein Entry-Node, nur nicht automatisch vergeben. Wird durch die Bridge-Authority vergeben 3.4.4 Seite 13

Entry-Guard Entry-Nodes, die sich durch hohe Zuverlssigkeit ausgezeichnet haben. Sie werden nicht bei jeder Route neu gewhlt, sondern ber lngere Dauer genutzt 3.4.5 Seite 13

Circuit Eine Verbindung zwischen mehreren Rechnern ber das Tor Netzwerk z.B. zwischen einem Onionproxy und einem Onionrouter.

3.2 Verstecken einer Kommunikation

Der einfachste Weg eine Kommunikation zu verstecken ist durch ein einfaches Relay. Ein bekannter Anonymisierungsdienst der so arbeitet ist der Anonymizer[10]. Wenn Alice mit Bob kommunizieren möchte, schickt sie ihre Datenpakete an ein Relay und dieses schickt die Daten weiter an Bob. Auf dem gleichen Weg kommuniziert Bob mit dem Relay und dieses schickt die Daten wieder zurück an Alice. Das hat zur Folge, dass der Provider nicht weiß, dass Alice sich mit Bob unterhält, sondern nur sieht, dass Alice mit dem Relay kommuniziert. Der Provider von Bob weiß ebenfalls nichts von Bobs eigentlichem Kommunikationspartner, sondern nur davon, dass Bob mit dem Relay Daten austauscht. So weit betrachtet ist die Kommunikation nun erstmal anonym. Das Relay, z.B. ein

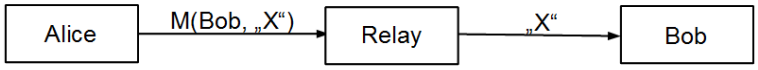

Abbildung 1: Einfaches Relay zwischen Alice und Bob, z.B. ein kommerzieller Proxy

kommerzieller Proxy, kann uns zwar versprechen den Traffic nicht zu lesen, zu loggen oder gar weiter zu geben, kontrollieren lässt sich das jedoch nicht. Alice und Bob müssen dem Relay vertrauen. Die komplette Anonymität liegt also in den Händen des Relays und kann ebenso alleine durch diesen gebrochen werden. Um dieses Problem zu lösen, kann man den Ansatz des verteilten Vertrauens wählen[8]. Wenn also das Relay zwischen den beiden Kommunikationspartnern, nicht ein Relay ist, sondern mindestens drei, die in Reihe geschaltet sind, weiß keines der Relays von Alice und von Bob. Das erste Relay (Entry-Node) kennt nur Bob und hat das zweite Relay (Middle-Node) als Empfänger. Das zweite Relay schickt diese Nachricht weiter an das dritte Relay (Exit-Node), welches die Informationen dann an Bob sendet.

[10] http://anonymizer.com

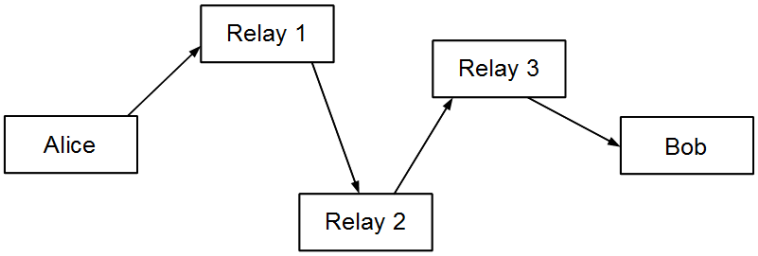

Abbildung 2: Mehrere Relays, zwischen Alice und Bob

Dies lässt sich in 4 relevante Fälle unterteilen, in denen entsprechende Relays überwacht werden oder korrupt sind:

1. Relay 1 oder 2 oder 3: keine Gefahr
2. Relay 1 und 2: keine Gefahr
3. Relay 2 und 3: keine Gefahr
4. Relay 1 und 3: in diesem Fall ist es möglich, durch Traffic Analyse der Daten, eine Nachricht eindeutig einem Sender und Empfänger zu zuordnen. Wird behandelt in 3.4.5 Seite 13

3.3 Onion Routing

Um Informationen zu schützen und sicher zu stellen, dass sie nur von entsprechenden Personen gelesen werden können, wurde seit 1995, durch Paul Syverson, David Goldschlag und Michael Reed für das Naval Research Laboratory, an dem Verfahren des *Onion Routing* gearbeitet[11][9]. Die Funktionsweise des *Onion Routing* ist leicht anhand eines analogen Beispiels zu erklären:

Wir nehmen an, dass Alice Bob einen Brief schicken will. Die an Bob adressierte Nachricht selber wird in drei Umschläge gepackt. Von aussen nach innen gesehen stehen die Adressen von Carol, Dave und Tom auf den Umschlägen. Wenn Alice den Brief nun abschickt, wird dieser an Dave geschickt. Da der Brief an Dave adressiert ist, kann er diesen öffnen und erhält somit einen Brief adressiert an Carol. Carol erhält den Brief, öffnet diesen und hat nun den Brief an Tom. Tom erhält den Brief, öffnet ihn und hat nun die Nachricht von Alice in den Händen und leitet diese weiter an Bob. In die Informatik übertragen müssen wir noch zwei Annahmen beachten:

1. Briefe sind verschlüsselte Nachrichten, der Umschlag also die Verschlüsselung auf dem der Empfänger steht.
2. Nur der Empfänger, an den der Brief adressiert ist, kann diesen öffnen. Um die beinhaltende Nachricht zu entschlüsseln, braucht man also den entsprechenden Key.

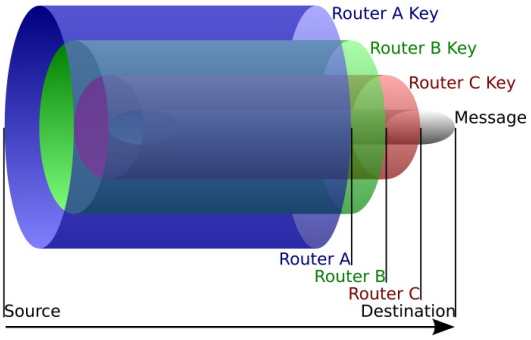

Abbildung 3: Onion Routing, Quelle: http://en.wikipedia.org/wiki/Onion_routing

Es werden drei Schlüsselpaare generiert. Mit den privaten Schlüsseln verschlüsselt Alice die Informationen in drei Schichten. Die entsprechenden Empfänger können die Verschlüsselungen von aussen nach innen entschlüsseln und das Paket weiterschicken.

3.4 TOR

3.4.1 Anonyme Verbindung über TOR

Senden Um eine Verbindung zwischen zwei Kommunikationspartnern zu ermöglichen, braucht der Sender (Alice) einen Onionproxy[11]. Der entsprechende Client, z.B. der Browser muss so konfiguriert sein, dass er seine Verbindung zum Onionproxy weiterleitet. Dieser arbeitet als Socks-Proxy. Das heißt, dass diese Konfiguration mit allen Clients möglich ist, die die entsprechenden Sockskonfigurationen für ausgehende TCP Verbindungen zulassen. Für den Datentransfer zwischen den Teilnehmern eines Tor Circuits werden 512 Byte groe Kontroll- und Relay-Pakete genutzt. Die Kontroll-Pakete werden für den Verbindungsaufbau und den Schlüsselaustausch genutzt[8]:

- circID: Circuit Identifier, sagt aus, auf welchen Circuit sich das Packet bezieht
- CMD:
  - *padding*: Verbindung aufrecht erhalten
  - *create*: neuen Circuit erstellen
  - *destroy*: die Verbindung zu unterbrechen

Die Relay-Pakete werden für den eigentlichen Datenverkehr genutzt. Sie haben noch zustzliche Felder im Paketheader[8]:

- streamID: Identifiziert den Stream, da mehrere ber einen Circuit multiplexed werden knnen
- Digest: end-to-end Checksumme um die Paketintegritt zu kontrollieren

[11] https://www.torproject.org/about/overview.html.en

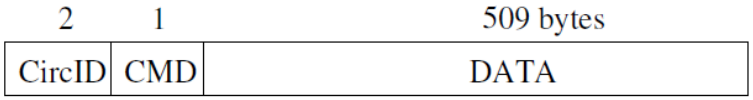

Abbildung 4: Kontroll-Paket

- Len: Länge der Daten
- CMD:
  - *padding*: Verbindung aufrecht erhalten
  - *relay data*: normal Daten senden
  - *relay end*: einen Daten Stream beenden
  - *relay teardown*: Verbindung beenden
  - *relay connected*: Onionproxy informieren, dass das *relay begin* erfolgreich war
  - *relay truncated*: um nur einen teil des Circuit zu beenden
  - *relay sendme*: für den erhalt des Packetflusses

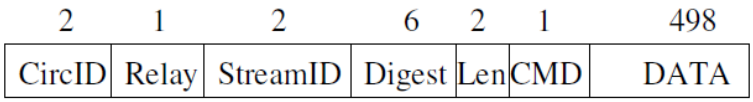

Abbildung 5: Relay-Paket

Der Onionproxy baut eine Verbindung zum Verzeichnissserver auf und eine mögliche Route, die von Alice zum Empfänger (Bob) durch das Tor-Netzwerk führt. Dies sind normalerweise drei Onionrouter. Nach der Routenauswahl verbindet sich der Onionproxy von Alice mit dem ersten Onionrouter. Über einen Diffie-Hellman-Schlüsselaustausch[5] zwischen dem Onionproxy von Alice und dem ersten Onionrouter wird das erste Schlüsselpaar ausgetauscht. Über den ersten Onionrouter wird daraufhin eine Verbindung zum zweiten Onionrouter aufgebaut und auch hier wird ein Schlüsselpaar über einen Deffi-Hellman-Schlüsselaustausch generiert. Das Gleiche passiert dann über den ersten und zweiten Onionrouter zum dritten Onionrouter. Daraufhin hat Alice für jeden der drei Onionrouter einen Schlüssel, mit dem sie Nachrichten verschlüsseln kann und jeder Onionrouter einen Schlüssel, mit dem die Nachrichten wieder entschlüsselt werden können. Die bisherige kommunikation wurde ber die Kontroll-Pakete durchgefhrt. Alice verschlüsselt nun die Nachricht in entsprechender Reihenfolge und sendet ihr Datenpaket in einem Relay-Paket an den ersten Onionrouter, der als Entry-Node fungiert. Wie schon beim Onionrouting beschrieben, wird nun die Nachricht schichtweise entschlüsselt

und vom dritten Onionrouter in Urform an Bob gesendet. Die Kommunikation zwischen den Onionroutern und dem Onionproxy findet in einem Cipher Stream[3] basierend auf einem 128 Bit Schlüssel statt. Da, die letzte von Alice erstellte Verschlüsselungsschale vom dritten Relay entschlüsselt wurde, wird die Kommunikation zwischen dem dritten Relay und Bob unverschlüsselt bertragen. Hier empfiehlt sich eine weitere end-to-end Verschlüsselung, wenn diese von Bob unterstüzt wird.

Empfangen Bob sendet seine Antwort dann an den Exit-Node, von dem er die Nachricht erhalten hat. Der Onionrouter verschlüsselt dann die Nachricht mit dem Schlüssel, den dieser mit Alice (über die anderen beiden Onionrouter) ausgehandelt hat und sendet die Nachricht an den Middle-Node. Dieser Onionrouter und der Entry-Node von Alice verschlüsseln die Nachricht wieder und vom Entry-Node wird die dreifach verschlüsselte Nachricht wieder an Alice geschickt. Alice, die alle drei Schlüssel hat, entschlüsselt die Nachrichten wieder und kann die Antwort lesen.

Folgend eine Tabelle[12], die darstellt, welche Informationen welcher Node bei der Datenübertragung hat.

	Benutzer	Bridge oder Entry-Guard	Middle-Node	Exit-Node
IP des Tor Users	ja	ja	nein	nein
IP der Bridge/des Entry-Guards	ja	ja	ja	nein
Nachricht an Bridge/Entry-Guard	ja	ja	nein	nein
IP des Middle-Node	ja	ja	ja	ja
Nachricht an Middle-Node	ja	nein	ja	nein
IP des Exit-Node	ja	nein	ja	ja
Nachricht an Exit-Node	ja	nein	nein	ja
IP des Empfängers	ja	nein	nein	ja
Nachricht an den Empfänger	ja	nein	nein	ja

Anhand dieser tabellarischen Auflistung der Informationen kann man entnehmen, dass der Exit-Node die Nachricht in der Urfassung sieht. Die schalenweise Verschlüsselung des Tor-Netzwerks reicht also nur bis zum Exit-Node. Das eröffnet einerseits die Möglichkeit nahezu jeden Service über das Tor Netzwerk zu erfragen, doch muss für eine vollständige Verschlüsselung zwischen Sender um Empfänger die Nachricht vom Empfänger explizit verschlüsselt werden. Dafür muss vorher ein Schlüsselaustausch zwischen Sender und Empfänger auf einem anderen Weg erfolgt sein.

3.4.2 Hidden Service

sonstiges vorteil fr bob durch die extraschicht des RP: er kann Verbindungen Annehmen und andere Ablehnen

Bereitstellen Das Tor-Netzwerk bietet ebenso die Funktionalität mit wenig Aufwand seinen Service anonym anzubieten[13]. Das Einrichten[14] des *Hidden Service* geschieht mit Hilfe

[12] https://trac.torproject.org/projects/tor/wiki/doc/TorFAQ
[13] https://www.torproject.org/docs/hidden-services.html.en
[14] https://www.torproject.org/docs/tor-hidden-service.html.en

des Tor-Clients. Im folgenden Beispiel gehen wir davon aus, dass Alice auf einen *Hidden Service* zugreifen will, der von Bob angeboten wird. Die Kommunikation findet auch hier wieder mit den Kontroll-Pakten für den Aufbau, die Steuerung und Relay-Paketen für den eigentlichen Datenaustausch statt. Im ersten Schritt werden von Bob *Circuits* zu zufällig gewählten Relays aufgebaut. Diese werden im folgenden als *Introduction Points* agieren. Bob gibt in diesem Schritt auch seinen Public-Key an die *Introduction Points*. Sein generierter *Public Key* wird als Identifikation für den Service genutzt. Im nächsten Schritt baut Bob den *Service Descriptor* zusammen. Dieser beinhaltet seinen *Public Key*, eine Beschreibung des Dienstes und eine Zusammenfassung von den gewählten *Introduction Points*. Der *Service Descriptor* wird nun zu einem Datenbank-Server gesendet, welcher Einträge über *Hidden Services* verwaltet. Der Datenbank-Server ist eine *Distributed-Hash-Table* die dem *Cooperative File System*(CFS [7])ähnlich ist. Um die Einträge abzufragen, wird eine 16-Stellige Adresse aus dem *Public Key* generiert. Die Adresse hat dann die Form [XYZ].onion. Auch wenn dies vorerst nach einer umständlichen Adresse aussieht, hat sie zwei entscheidende Vorteile:

1. Es ist keine zentrale Vergabestelle nötig
2. Alle Beteiligten, inklusive der *Introduction Points* und des Clients, können den Hidden Service anhand des *Public Key* verifizieren.

Ist dies geschehen, ist der Service komplett konfiguriert.

Aufrufen Um einen Hidden Service aufzurufen, muss der Aufrufer den Tor-Client installiert haben und ihm muss die .onion-Adresse bekannt sein. Entsprechende Adressen bekommt man z.B. durch einen Direktkontakt oder nach ein wenig Recherche im Internet. Folgend eine kleine Auswahl von Hidden-Services:

1. Wikileaks: http://isax7s5yooqgelbr.onion
2. keys.indymedia.org: http(s)://qtt2yl5jocgrk7nu.onion
3. TorMail[15] http://jhiwjjlqpyawmpjx.onion
4. TorPM http://4eiruntyxxbgfv7o.onion/pm/
5. Jabber(IM via XMPP): ch4an3siqc436soc.onion Port: 5222

Beim Aufruf der entsprechenden Adresse, wird der *Service Descriptor* aus der Datenbank geladen. Der Client kennt nun die vom *Hidden Service* gewählten *Introduction Points* und den entsprechenden *Public Key*. Gleichzeitig wird vom Cient ein *One-Time-Secret* generiert und ein zufälliger Server im Tor-Netzwerk gewählt. Diesem wird das *One-Time-Secret* zugeschickt und damit abgefragt, ob er als *Rendezvous Point* agieren kann. Über diesen von Alice gewählten *Rendezvous Point* wird die eigentliche Kommunikation zwischen Alice und Bob stattfinden. Die neuen Informationen werden nun zu einer sogenannten *Introduce Message* zusammengefasst. Sie umfasst die Adresse vom *Rendezvous Point*, dass *One-Time-Secret* und ist mit dem *Public Key* des *Hidden Service* verschlüsselt. Nun wird die Nachricht

[15] http://tormail.org/

an einen der *Introduction Points*, mit der Aufforderung den *Hidden Service* zu kontaktieren, gesendet. Der *Hidden Service* dekodiert die *Introduce Message* und baut einen *Tor Circuit* zum *Rendezvous Point* auf. Das in der *Introduce Message* beinhaltete *One-Time-Secret* wird an den *Rendezvous Point* geschickt. Diese Nachricht wird *Rendezvous Message* genannt. Im letzten Schritt informiert der *Rendezvous Point* Alice, dass die Verbindung zwischen ihm und Bob aufgebaut ist. Der *Rendezvous Point* dient nun als Relay zwischen Alice und Bob und reicht die *end-to-end* Verschlüsselte Nachricht weiter. Im ersten Moment scheint dieses Modell sehr umständlich, doch bei näherer Betrachtung wird verständlich, warum nicht direkt ein *Introduction Point* für die Verbindung genutzt wird. Einmal wird nicht jedes zur Kommunikation benötigtes Relay von einer Seite ausgesucht und der *Rendezvous Point* weiß vorher nichts von diesem *Hidden Service*. Dies garantiert eine deutlich höhere Sicherheit für die Kommunikation und den *Hidden Service*.

3.4.3 Angriffe

Staat

1. Eine der verwundbarsten Stellen im Tor-Netzwerk sind die derzeit 10 Verzeichnisserver. Denkbar wäre z.B. die Manipulation der Verzeichnissdaten, oder die Abschaltung eines Verzeichnisservers. Letztere Möglichkeit ist in sofern schwierig, da erst bei einem Ausfall von 50 % aller Verzeichnisserver nicht mehr genügend Signaturen vorhanden sind und die Onionrouter über das Vertrauen der Relays entscheiden[10]. Die Verzeichnisserver werden aus diesem Grund von Vertrauten und kompetenten Mitarbeitern des Tor-Projekts gehostet. Die Adressen der Verzeichnisserver findet man unter src/or/config.c seiner Tor Installation.
2. Ein alternativer Angriff könnte sein, dass ein Angreifer versucht alle *Sessionkeys* einer Strecke zu brechen, doch wird die Route alle 10 Minuten neugewählt.
3. Eine weitere denkbare Strategie eines Angreifers könnte sein, eine hohe Anzahl Onionrouter zu kontrollieren und dabei die nicht kontrollierten Onionrouter anzugreifen. Dies würde das Verhältniss von kontrollierten zu nicht kontrollierten Onionroutern zu seinen Gunsten anpassen. Hier spricht aber die Anzahl, die Vielfalt und die Robustheit der unterschiedlichen Systeme gegen eine derzeit realistische Chance.
4. Des Weiteren ließen sich in der Vergangenheit Muster im HTTP-Verkehr erkennen (*GET /tor/*), was aber schon seit einigen Versionen behoben ist[2]

Beispiel China Große Anstrengung gingen in der Vergangenheit von China aus. Durch IP-Blockaden, Deep Packet Instpection und Zensierung durch Suchmaschinen[13].

3.4.4 Bridge Relays

Ein Problem ergibt sich durch die Sperrung der Tor-Nodes auf Seiten der Internet Service Provider (ISP). Die Liste aller Tor-Nodes ist öffentlich[16]. In einigen Ländern wurden die ISPs dazu aufgefordert Tor-Nodes zu sperren und somit den Personen, die ihren Zugang über diesen ISP haben, zu verwehren. Da Tor auch genutzt werden kann um Zugriffssperren zu umgehen, da man indirekt mit der zensierten Adresse kommuniziert, ist Tor gerade für zensierende Staaten ein Problem. Um diese Zensurresistenz aufrecht zu erhalten und es Personen zu ermöglichen, trotz der Sperrung von Tor-Nodes das Tor-Netzwerk zu nutzen, wurde die *Bridge-Funktionalität* entwickelt. *Bridge-Relays* sind normale Tor-Nodes, die jedoch nicht in einer öffentlichen Liste gehalten und auch nicht vom Verzeichnisserver vergeben werden. Somit ist es für Staaten schwerer die Tor-Nodes zu identifizieren und zu sperren[2]. Aus Sicherheitsgründen werden die Bridge-Relays auf drei verschiedene Wegen[17] durch die *Bridge Authority* vergeben und getrennt gehalten:

1. Website[18] besuchen und über das Formular Bridge-Relays anfordern. Man erhält 2 IPv4 und 2 IPv6 Adressen.

2. E-Mail an bridges@torproject.org. Die E-Mail muss entweder von einer gmail.com oder einer yahoo.com Domain verschickt werde. Somit wird auf den vorhandenen Schutz vor Massenkontenerstellung der beiden Anbieter vertraut. Man erhält 3 IPv4 Bridge-Relays zur Auswahl.

3. Direktkontakte zu Entwicklern oder Personen, die mit dem Tor Projekt in Verbindung stehen.

Angriffe Durch China wurde im September 2009 der 1. Vergabepool(Website) und im März 2010 der 2. Vergabepool(E-Mail) angegriffen[19]. Dabei wurden aus dem 1. Pool 176 und aus dem 2. Pool 201, von zu der Zeit insgesamt 542 Bridge-Relays, herausgegeben. Alle Bridge-Relays wurden daraufhin von China gesperrt. Hier zeigt sich schon der Sinn der Aufteilung in verschiedene Pools, denn im 3. Pool waren noch 165 Bridge-Relays, mit denen man arbeiten konnte, da sie nicht gesperrt waren. Dennoch ist diese Lösung noch nicht die beste Vergabestrategie, da immer auf mehrere Faktoren zu vertrauen ist.

3.4.5 Entry Guards

Eine Schwachstelle des Tor-Netzwerkes wurde mit der Einführung der *Entry Guards* gemindert. Wenn ein Angreifer den Entry- und Exit-Node unter Kontrolle hat, kann er aufgrund des Eingangs- und des Ausgangs-Traffics Schlüsse ziehen, von welchem Sender welches Paket an welchen Empfänger geht. Diese *Traffic Analyse* basiert auf der Korrelation von Größen und Zeitattributen an beiden Enden des Tor-Netzwerks. Da Tor-Routen

[16] http://proxy.org/tor.shtml
[17] https://www.torproject.org/docs/bridges.html.en
[18] https://bridges.torproject.org/
[19] https://blog.torproject.org/blog/research-problems-ten-ways-discover-tor-bridges

kurzlebig sind und immer wieder neu gewählt werden, ergibt sich bei regelmäßiger Nutzung eine Wahrscheinlichkeit von 100%, dass eine Verbindung über das Tor-Netzwerk deanonymisiert werden kann. Das Angreifer, die einen Tor-Node unter Kontrolle haben eine Routenneuwahl erzwingen können, wenn sie nicht die nötigen Knoten für die *Traffic Analyse* besitzen, macht diese Methode noch effektiver. Wenn p das Verhältnis der vom Angreifer kontrollierten Knoten zur Anzahl aller Knoten ist, können wir sagen, dass es eine Wahrscheinlichkeit von $p \cdot p = p^2$ gibt, dass der Angreifer Entry- und Exite-Node unter Kontrolle hat. Da der Angreifer jedoch, wenn er einen Middle-Node unter Kontrolle hat, eine Routenneuwahl erzwingen kann, nehmen wir r als aufgebaute Routen an: $(1-p^2)^r$. Das bedeutet, dass bei steigendem r die Wahrscheinlichkeit eine Route nicht zu deanonymisieren gegen 0 läuft[4]. Um dem entgegen zu wirken, gibt es das Konzept der *Entry Guards*[20]. Entgegen des eigentlichen Konzepts des *Onion Routings*, wird der Entry-Node nicht jedes Mal dynamisch aus den Daten des Verzeichnisservers gewählt, sondern der Onion-Proxy erhält eine Anzahl (normalerweise drei) Entry-Nodes. Diese werden dann über eine längere Dauer (mehrere Wochen) vom Sender genutzt, um das Tor-Netzwerk zu betreten. Nur beim Ausfall des gewählten Entry-Guards, wird vorzeitig eine Ersatzauswahl getroffen. *Entry Guards* sind in erster Linie ganz normale Entry-Nodes die ein hohes Vertrauen genießen und zuverlässig laufen. Entry-Nodes werden zu *Entry Guards*, wenn sie bereits längere Zeit laufen und in dieser Zeit eine hohe Verfügbarkeit und eine überdurchschnittliche Übertragungskapazität zeigen[21].

4 TOR Projekt

Abbildung 6: Offizielles Logo des Tor-Projekt, Quelle: https://www.torproject.org/

Den Tor Client gibt es in verschiedenen Ausführungen und Bundles. Die vorgefertigten Bundles ermöglichen es dem Nutzer ohne umfangreiche Kenntnisse z.B. einen Entry-Node zu betreiben oder ganz normal den Tor Client als Einstieg zum Tor-Netzwerk zu nutzen. Weitere interessante Projekte, wie die Tor Cloud[22], in

[20] https://www.torproject.org/docs/faq#EntryGuards
[21] https://blog.torproject.org/blog/research-problem-better-guard-rotation-parameters
[22] https://cloud.torproject.org

Verbindung mit der Amazon Cloud werden ebenfalls auf der Projektseite[23] vorgestellt. Als grafische Oberfläche für den Client, bietet sich Vidalia[24] an.

5 Fazit

Das Tor-Projekt ist eines der fortschrittlichsten Wege, um das Internet anonym zu nutzen. Jedoch gibt es einige Punkte, die man bei der Benutzung bedenken sollte. Darunter fällt nicht nur die korrekte Installation und Konfiguration, sondern auch das Wissen, dass z.B. die Nachricht zwischen dem Exit-Node und dem Empfänger nicht verschlüsselt ist. Das dezentrale und dynamische Routing bringt ein hohes Maß an Sicherheit, doch ist die Übertragungsgeschwindigkeit bei jeder Route auf die Kapazität des schwächsten Nodes begrenzt. Somit ist das Netzwerk nicht darauf ausgelegt, große Datenmengen zu verschieben, jedoch z.B. Texte auszutauschen oder zu bloggen. Es gibt viele offene Punkte, an denen man sich im Tor-Projekt beteiligen kann, um es weiter zu entwickeln oder den Nutzerkreis und somit die Anzahl der Relays zu erweitern.

[23] https://www.torproject.org/projects/projects.html.en
[24] https://www.torproject.org/projects/vidalia.html.en

Literatur

[1] Ross J. Anderson, editor. *Information Hiding, First International Workshop, Cambridge, U.K., May 30 - June 1, 1996, Proceedings*, volume 1174 of *Lecture Notes in Computer Science*. Springer, 1996.

[2] Jacob Applebaum and Roger Dingledine. How governments have tried to block tor. 28 Chaos Communication Congress, 2011.

[3] Côme Berbain and Henri Gilbert. On the security of iv dependent stream ciphers. In *FSE*, pages 254–273, 2007.

[4] Nikita Borisov, George Danezis, Prateek Mittal, and Parisa Tabriz. Denial of service or denial of security? How attacks on reliability can compromise anonymity. In *Proceedings of CCS 2007*, October 2007.

[5] Emmanuel Bresson, Olivier Chevassut, David Pointcheval, and Jean-Jacques Quisquater. Provably authenticated group diffie-hellman key exchange. In Mike Reiter, editor, *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS '01)*, pages 255–264, Philadelphia, Pennsylvania, 2001. ACM Press.

[6] Bundestag. Gesetz zur erschwerung des zugangs zu kinderpornographischen inhalten in kommunikationsnetzen. *BGB*, 2010.

[7] Frank Dabek, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01)*, Chateau Lake Louise, Banff, Canada, October 2001.

[8] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[9] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding routing information. In Anderson [1], pages 137–150.

[10] Jens Kubieziel. *Anonym im Netz*. Open Source Press, München, 2007.

[11] Paul Syverson. A peel of onion. Washington DC, USA, 2011. Center for High Assurance Computer Systems U.S. Naval Research Laboratory.

[12] Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an analysis of onion routing security. Technical report, 2000.

[13] Philipp Winter and Stefan Lindskog. How china is blocking tor. *CoRR*, abs/1204.0447, 2012.