



Vorlesung

Datenschutz und Privatheit in vernetzten Informationssystemen

Kapitel 4: Datenschutz im Internet
Teil 2: Verbergen der IP-Adresse

Erik Buchmann
buchmann@ipd.uka.de



Agenda für heute

[Motivation](#)

Mixe

JAP

TOR

Freenet

- Einführung
- Mixe nach Chaum
- JAP
- TOR
- Freenet





Problem: IP-Adresse

[Motivation](#)

Mixe

JAP

TOR

Freenet

- Rechner kommunizieren im Internet über IP-Adressen
- Im Internet kann jeder
 - besuchte Webserver,
 - jeder Router “unterwegs”,
 - der Internet-Zugangspvoder,
 - Dienste wie Proxies, Firewalls, Cachesermitteln,
 - dass eine Kommunikation stattgefunden hat,
 - wer die Kommunikationspartner waren,
 - welcher Art diese Kommunikation war, und
 - den Inhalt (unverschlüsselter) Kommunikation.





Angreifermodell

[Motivation](#)

[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- schwache Angreifer
 - betreibt Webserver, Router, oder Analysedienst (vgl. Web-Bugs)
 - kann Informationen von wenigen ausgewählten Rechnern zusammenführen
- starker Angreifer
 - z.B. Geheimdienst mit starken Ressourcen, Betreiber eines Internet-Backbones
 - kann sehr große Teile der Internet-Kommunikation belauschen



Anonymität bei schwachen Angreifern

[Motivation](#)

[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- Trusted third Party, z.B. <http://www.anonymizer.com>
 - leitet eingehende Verbindungen unter der eigenen IP-Adresse weiter
 - löscht Cookies, identifizierende Attribute, Referer etc.



Anonymizer®
Trusted / Proven / Secure

Consumer Enterprise Government About Us

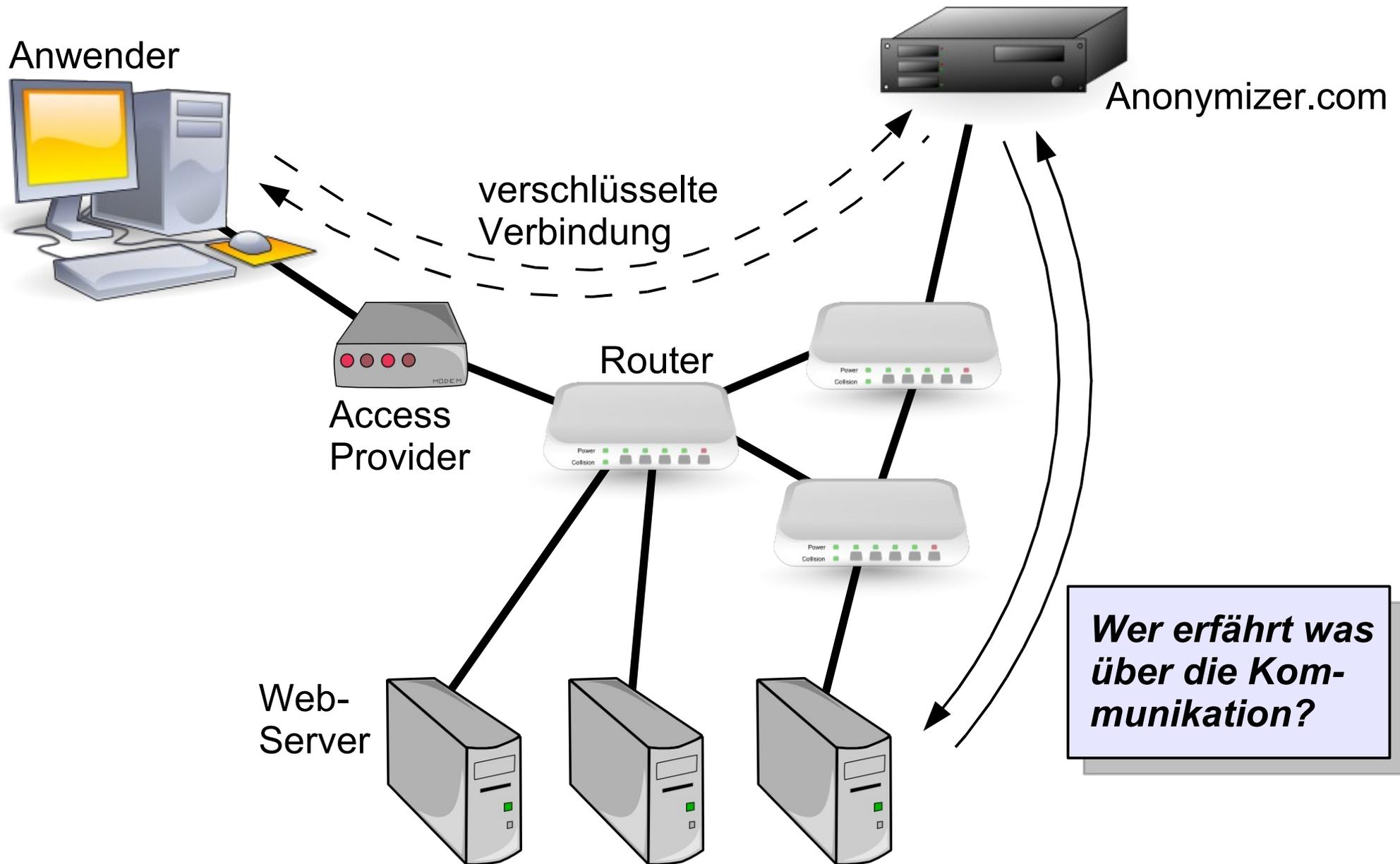
Anonymous Surfing™

..and it's free to try

Go way beyond simply hiding your computer's IP address with Anonymizer Anonymous Surfing™

- Safe Online Shopping
- Secure Internet Banking
- Phishing & Pharming Alerts
- Wireless PC Security

Anonymizer.com



Wer erfährt was über die Kommunikation?

Privoxy

- Privacy-Proxie zum selber-aufsetzen
 - d.h., benötigt einen Rechner, um darauf die Software zu installieren
- Leistet dasselbe wie Anonymizer.com
 - Cookie-Management
 - Filter zum säubern von HTTP-Headern (Referer, Browser-Informationen, etc.)
 - Filter zum entfernen von Web-Bugs, Scripten
- Open Source, <http://www.privoxy.org/>



Anonymität bei starken Angreifern

[Motivation](#)

Mixe

JAP

TOR

Freenet

- Es werden gebraucht:
 - starke **Public-Key-Verschlüsselung**
 - möglichst **viele unabhängige Rechner**, die Nachrichten für den Nutzer weiterleiten
- Idee:
 - Nutzer verbinden sich nicht direkt mit dem Server, sondern über mehrere Zwischenstationen
 - Kommunikation zwischen zwei Stationen erfolgt verschlüsselt
 - Anfragen eines Nutzers werden zwischen denen anderer Nutzer versteckt
- Ansätze im folgenden: JAP, TOR





Grundlagen: Mixe und Mix-Kaskaden



Mixe

Motivation

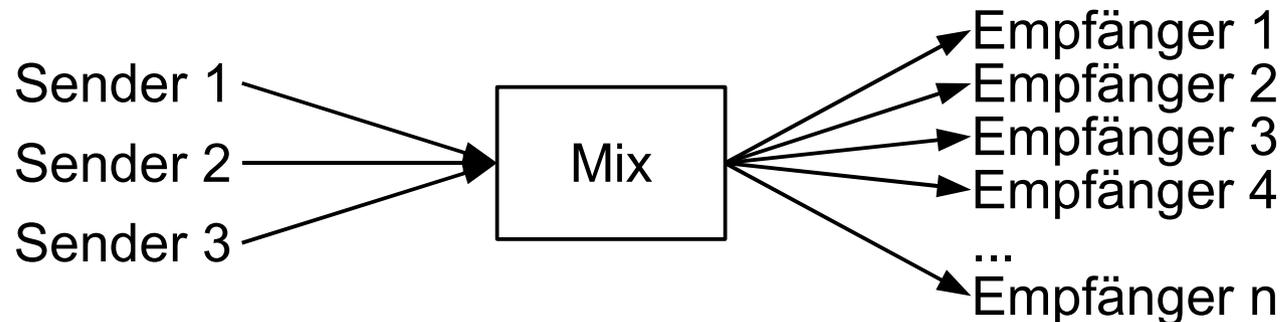
[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- D. Chaum: *Untraceable Electronic Mail*, Comm. ACM'81
- Mix vermittelt zwischen vielen Sendern und Empfängern



- verschlüsselte Nachrichten,
- kryptografische Transformation im Mix,
Eingabe-Nachricht != Ausgabenachricht
(*nächste Folie*)
- weiterleiten zum Ziel (oder weiterem Mix)
in zufälliger Reihenfolge (nicht Eingangsreihenfolge!)





Verschlüsselung in Mix-Kaskaden

Motivation

[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- Symbole:
 - K öffentlicher Schlüssel
 - R Zufallszeichenfolge
 - A Adresse der nächsten Station
- Arbeitsweise eines Mix:
 - Mix n erhält
$$K_n(R_n, K_{n-1}(R_{n-1}, K_{n-2}(R_{n-2} \dots \text{Msg}, A_0 \dots A_{n-2}), A_{n-1}), A_n)$$
 - mit K_n verschlüsselte Nachricht auspacken,
 R_n löschen, Adresse A_n lesen
 - sendet $K_{n-1}(R_{n-1}, K_{n-2}(R_{n-2} \dots \text{Msg}, A_0 \dots A_{n-2}), A_{n-1})$ an A_n
- Letzter Mix sendet Msg direkt an Adressaten A_0





Entschlüsselung in Mix-Kaskaden

Motivation

Mixe

JAP

TOR

Freenet

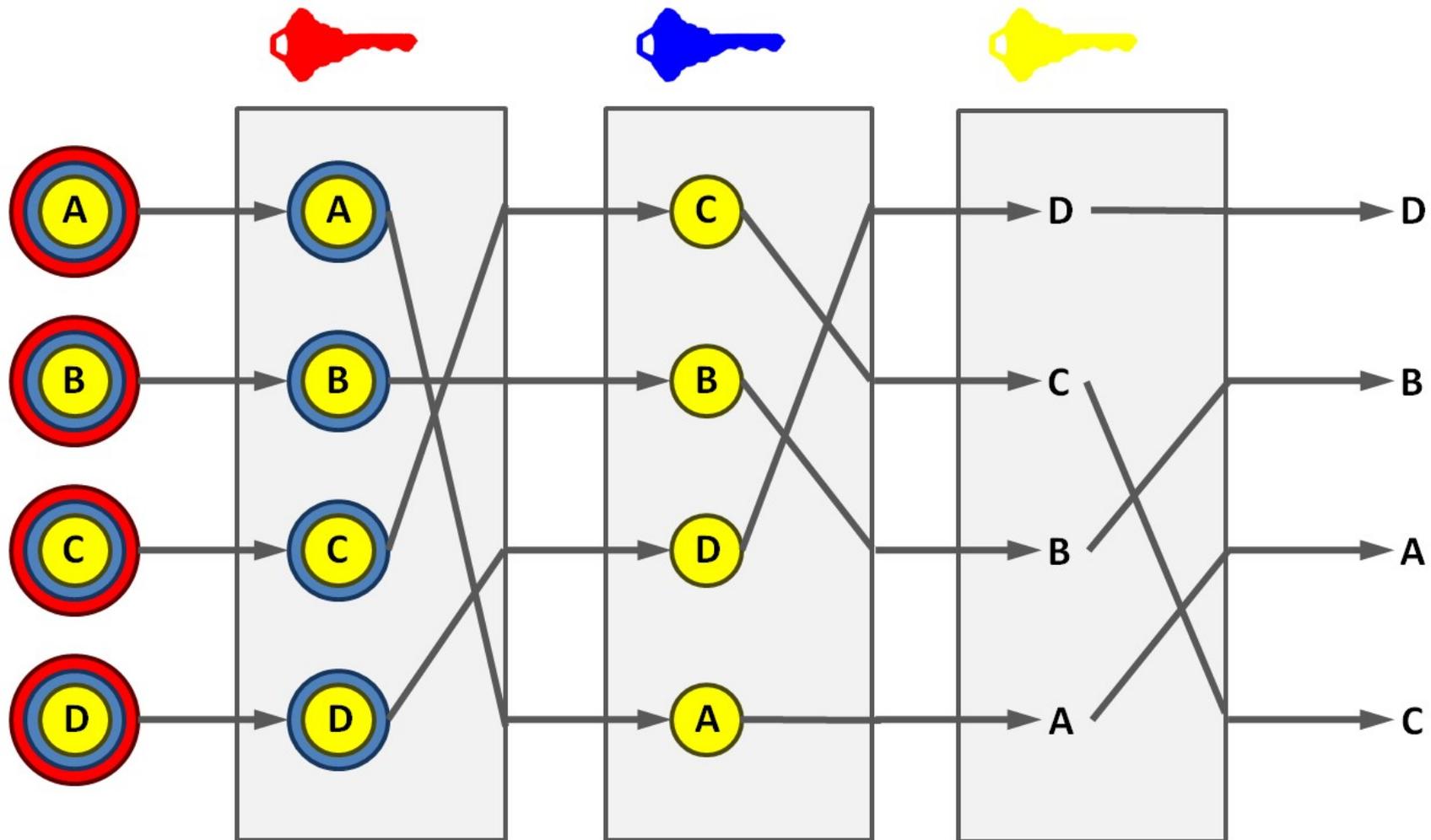


Bild: Wikimedia CC





Was ist mit der Antwort?

Motivation

[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- Empfänger soll nicht den Absender feststellen, muss aber trotzdem antworten können
- Prinzip: Adresse für Antwort verschlüsselt mitsenden, R dient jeweils als Schlüssel für die Antwort
 - Absender X , Einmalschlüssel K_x , Empfänger Y
Antwortadresse $K_1(R_1, A_x)$, K_x wird mitgesendet
 - Y kann Adresse A_x nicht entschlüsseln, sondern nur der letzte Mix in der Kaskade
 - Mix 1 erhält von Y Antwort $K_1(R_1, A_x)$, $K_x(R_0, \text{Msg})$ und sendet $R_1(K_x(R_0, \text{Msg}))$ an A_x
 - Verfahren lässt sich ebenfalls kaskadieren





Fazit

Motivation

[Mixe](#)

[JAP](#)

[TOR](#)

[Freenet](#)

- Nur öffentliche Informationen werden übertragen
 - Absender muss kennen:
 - öffentliche Schlüssel $K_0 \dots K_n$ aller Mixe
 - Jeder Mix m kennt
 - eigenen privaten Schlüssel K_m^{-1}
- Aber: einige ungelöste *praktische* Probleme, z.B.
 - Public-Key-Verschlüsselung ist *langsam*
 - statistische Angriffe über die Zahl der ein- und ausgehenden Pakete je Mix
 - Replay-Angriffe
(Wiedereinspielen von gültigen Paketen)





Verschleierung der IP-Adresse mit JAP



JAP Anon Proxy



Motivation

Mixe

JAP

TOR

Freenet

- Entwickelt seit 2000 vom AN.ON-Projekt
 - Uni Regensburg, TU Dresden, gefördert mit Mitteln der Deutschen Forschungsgemeinschaft und vom Bundesministerium für Wirtschaft und Technologie
 - <http://anon.inf.tu-dresden.de>
- Ziele:
 - Senderanonymität beim Abruf von Webseiten (*Anm.: Unterschied zu TOR o.ä., welche die gesamte Internet-Kommunikation verschlüsseln können*)
 - Einfache Benutzbarkeit; leichte Installation, Konfiguration und Bedienung
 - Dienstgüte; sowohl bzgl. Netzparametern (Durchsatz, Latenzzeit etc.) als auch Sicherheit (Anonymität)



JAP Anon Proxy

Motivation

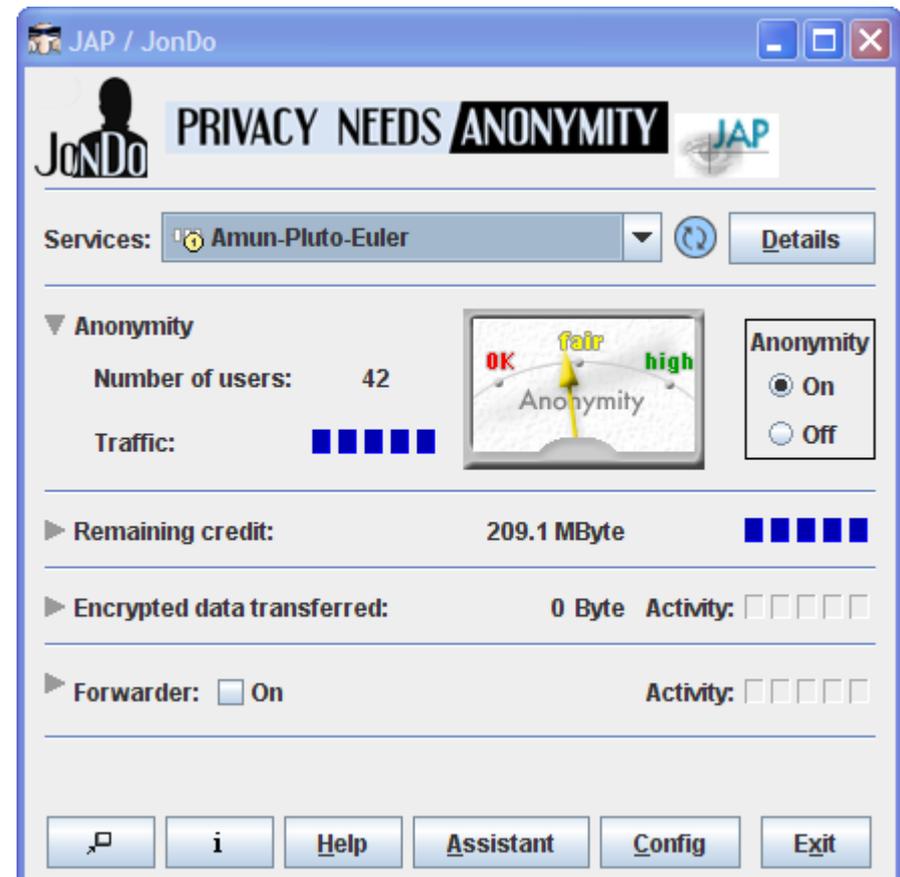
Mixe

JAP

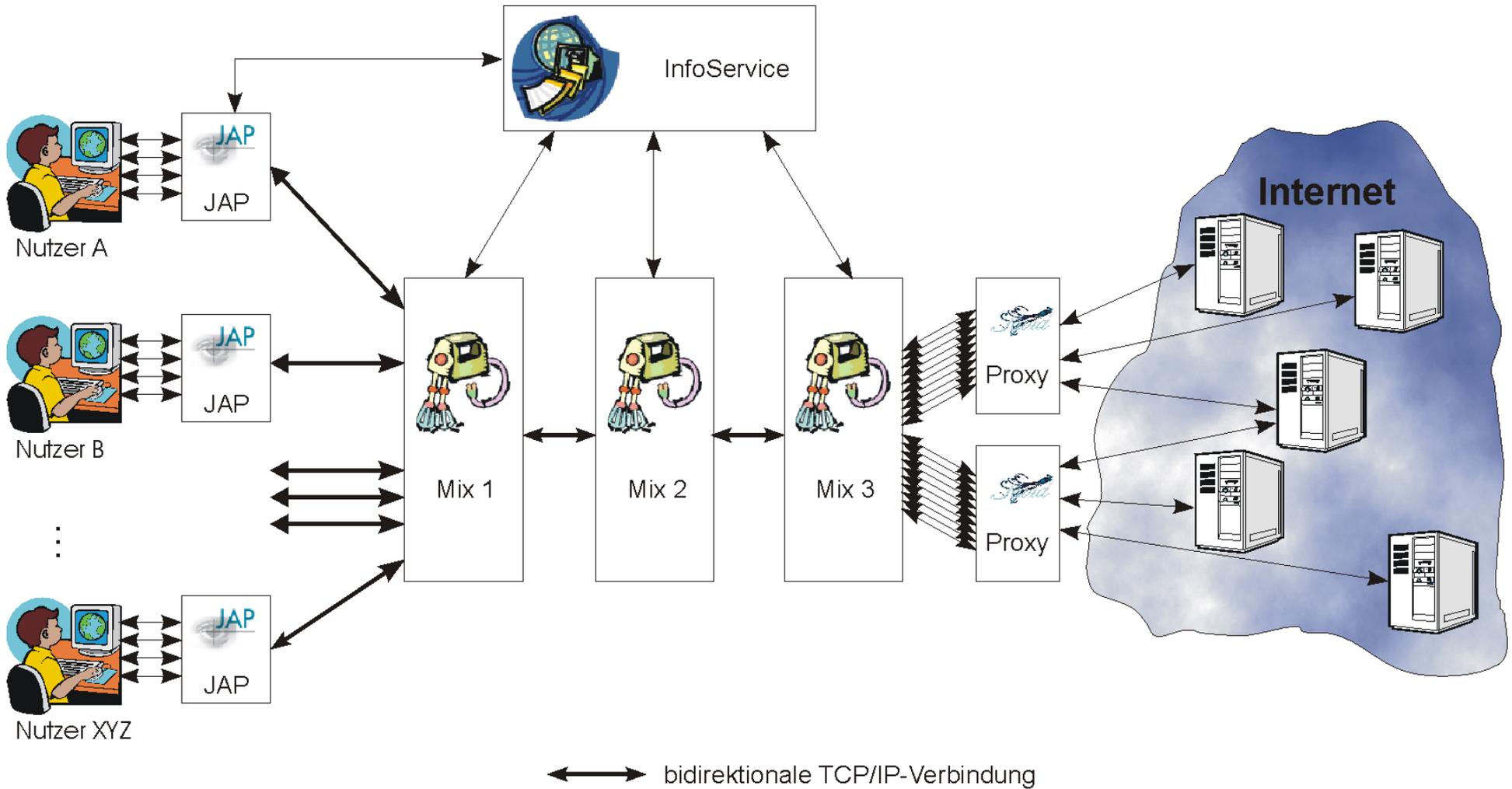
TOR

Freenet

- Im Gegensatz zu Chaum **feste Kanäle** mit symmetrischer Verschlüsselung
→ Problem mit der Rücksendeadresse entfällt, symm. Verschlüsselung ist effizienter als Public-Key
- grafische Client-Komponente
 - Proxy im Webbrowser
- Zusätzlich: Info-Service informiert über Schlüssel und aktive Mixe



Architektur von JAP



Quelle: S. Köpsell, <http://anon.inf.tu-dresden.de>



Mix-Kanäle

Motivation

Mixe

[JAP](#)

[TOR](#)

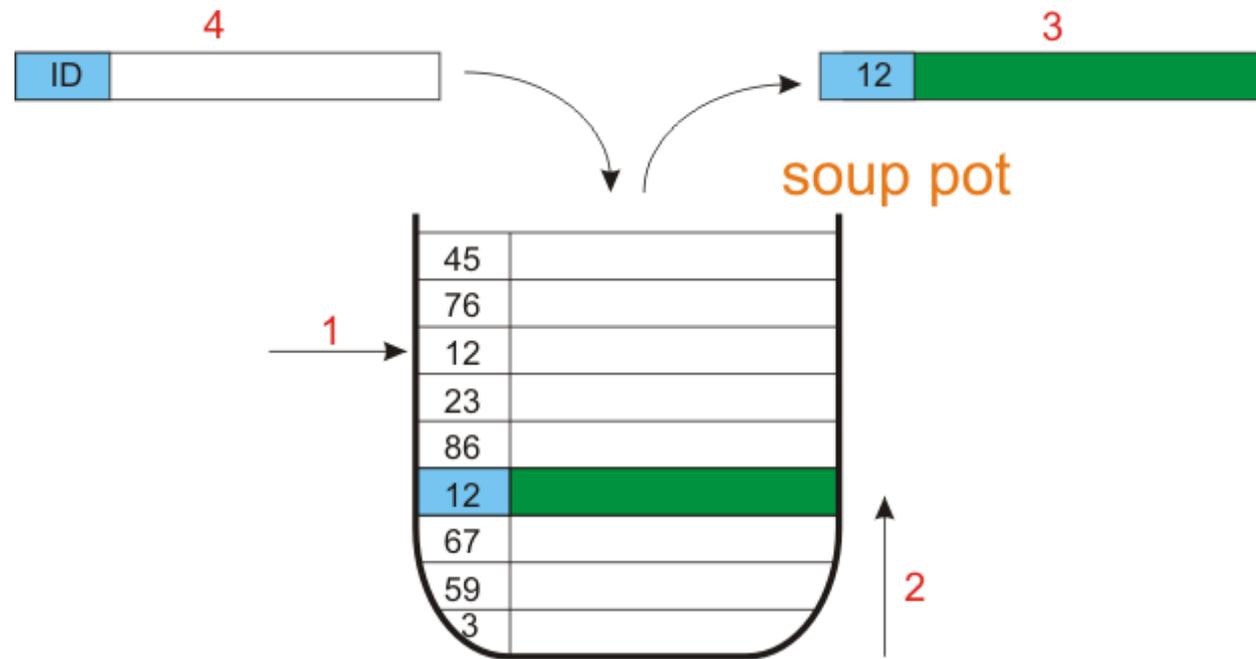
[Freenet](#)

- Statische Verbindung zwischen zwei Kommunikationsendpunkten über mehrere Mixe hinweg
 - Verbindungsorientierter Vollduplex-Betrieb (*Gegensatz zu paketorientiert bei Chaum*)
 - vom Sender einer Nachricht initiiert
- Sicherheitsfeatures
 - Datenstrom wird auf mehrere MIX-Pakete aufgeteilt, die alle gleich groß sind (auffüllen mit Zufallszahlen)
 - Zufällige Pakete an beliebige Empfänger, um Zuordnung zu verhindern
 - Symmetrische Verschlüsselung; (langsame) Public-Key-Verschlüsselung für Aushandeln der Public-Key-Schlüssel





Umsortieren von Paketen: Mix-Pool



- Eintreffen eines MixPaketes:
 1. Zufällige Auswahl eines MixPaketes (hat *Kanal-ID*)
 2. Suchen nach dem ältesten MixPaket mit *Kanal-ID* im Pool
 3. Ausgabe des MixPaketes
 4. Hinzufügen des empfangenen MixPaketes

Quelle: S. Köpsell, <http://anon.inf.tu-dresden.de>



Kommunikation mit dem Info-Service (1/2)

Motivation

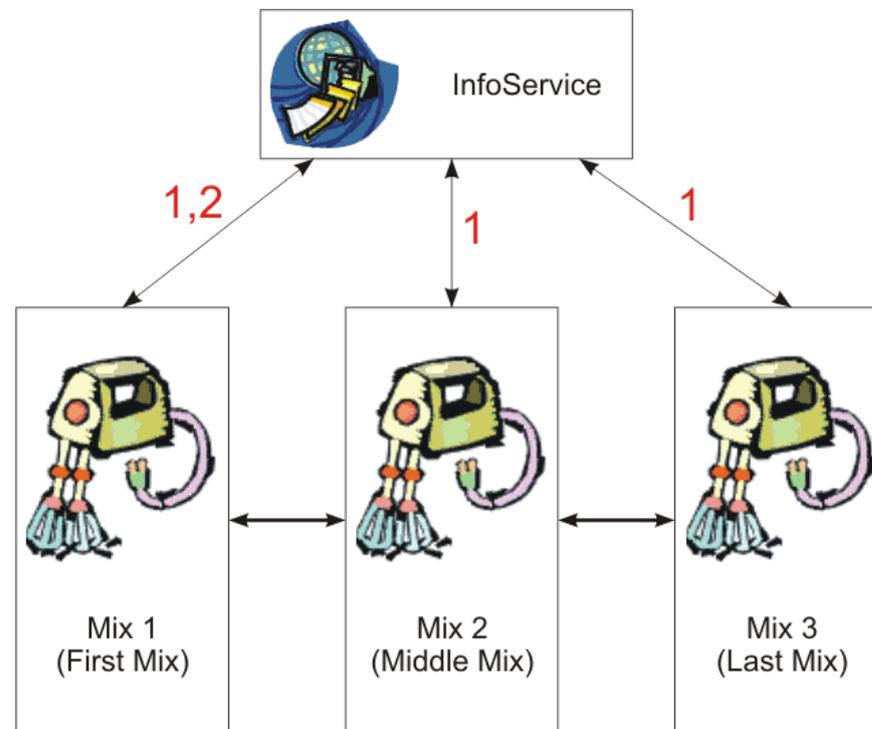
Mixe

JAP

TOR

Freenet

1. jeder Mix sendet Informationen über sich alle 10 Minuten an den InfoService (Name, Betreiber, Standort etc.)
2. erster Mix einer Kaskade sendet Statusinformationen (Benutzer, gemixte Pakete) jede Minute an InfoService



Quelle: S. Köpsell, <http://anon.inf.tu-dresden.de>



Kommunikation mit dem Info-Service (2/2)

Motivation

Mixe

[JAP](#)

TOR

Freenet

- Infoservice informiert über
 - Status-Daten zum Mix
 - Mix-Betreiber
 - aufgebaute Mix-Kaskaden
- Dient als Verzeichnis:
 - Auslieferung der JAP-Software
 - Verzeichnis der aktiven Mixe
 - Verzeichnis der öffentlichen Schlüssel
- JAP funktioniert generell aber auch ohne
 - jedenfalls sofern der Nutzer selbst Adresse und Schlüssel vom ersten Mix in einer Kaskade kennt





Starten einer Mix-Kaskade

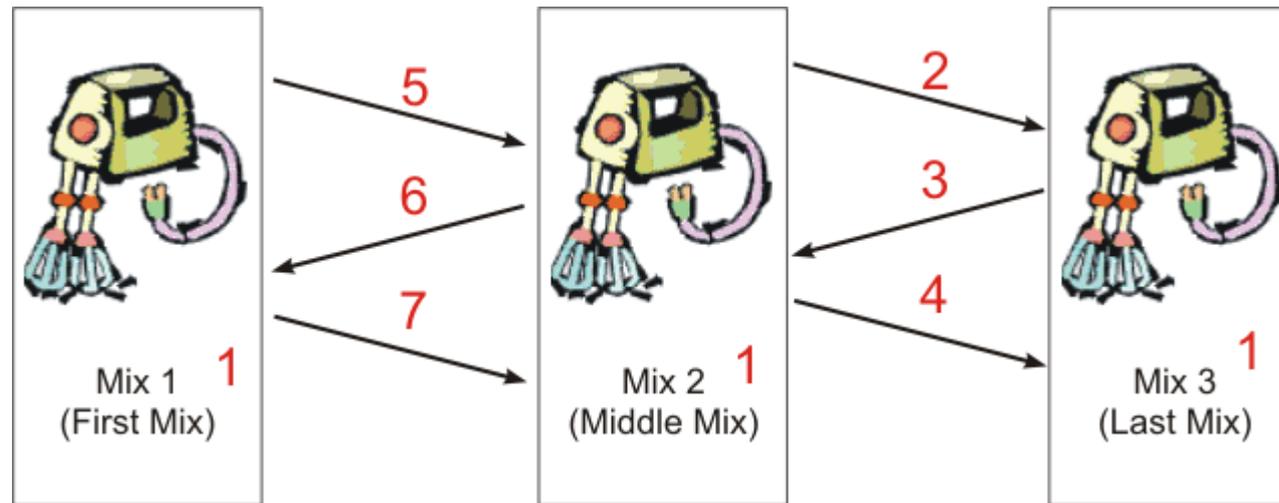
Motivation

Mixe

JAP

TOR

Freenet



1. Mix wird gestartet
2. Mix 2 verbindet sich zu Mix 3
3. Mix 3 sendet öffentlichen Schlüssel
4. Mix 2 sendet symmetrischen Schlüssel (verschlüsselt und signiert)
5. Mix 1 verbindet sich zu Mix 2
6. Mix 2 sendet seinen öffentlichen Schlüssel (signiert von Mix2) und den von Mix 3 (signiert von Mix 3)
7. Mix 1 sendet symmetrischen Schlüssel (verschlüsselt und signiert)

Quelle: S. Köpsell, <http://anon.inf.tu-dresden.de>





Verbindungsaufnahme mit einer Kaskade

Motivation

Mixe

JAP

TOR

Freenet

1. Client etabliert TCP/IP-Verbindung zum ersten Mix einer Kaskade
2. Mix sendet signierte Liste mit je einem Eintrag pro Mix der Kaskade an Client:
 - XML Struktur, jeder Eintrag enthält:
 - öffentlichen RSA Schlüssel des Mixes
 - ID des Mixes
 - Signatur geleistet von Mix
3. Client sendet symmetrischen Schlüssel für Verbindung JAP ↔ erstem Mix der Kaskade an Mix (verschlüsselt mit öffentlichem Schlüssel des Mix)
4. Verbindung kann benutzt werden





Die JAP-Mixe

Motivation

Mixe

[JAP](#)

TOR

Freenet

- Wenigstens 3 Mixe für JAP notwendig
- Daten vom 13.5.2009, 9 Uhr morgens:

| Name | Nutzer | Verkehr | Pakete |
|----------------------|--------|-------------|---------------|
| Dresden-Dresden | 3130 | 38 (normal) | 2,849,573,135 |
| JAP00.10.070 | 1323 | 41 (normal) | 479,116,379 |
| CookieCooker.de | 62 | 64 (hoch) | 14,902,585 |
| GalaxyConsult | 84 | 100 (hoch) | 93,655,608 |
| SurfSky-SpeedPartner | 62 | 39 (normal) | 66,984,625 |
| Pimenidis-Behrens | 22 | 80 (hoch) | 1,012,180 |
| Opossum-Unsu-Saturn | 31 | 33 (normal) | 1,722,136 |
| Amun-Pluto-Euler | 87 | 100 (hoch) | 23,736,734 |
| SpeedPartner-ULD | 1201 | 68 (hoch) | 9,704,103 |
| Euklid-Rose-ExarKun | 79 | 42 (normal) | 90,973,571 |





Mix-Betreiber

Motivation

Mixe

[JAP](#)

TOR

Freenet

- Mix-Betreiber gibt eine Selbstverpflichtung ab
 - keine Logfiles über die in JAP transportierten Verbindungen speichern
(Anm.: aber Logfiles für eigene Verbindungen, vgl. Vorratsdatenspeicherung)
 - kein Datenaustausch mit anderen Mix-Betreibern
- Solange nicht alle Betreiber einer Mix-Kaskade miteinander kooperieren, ist Anonymität sichergestellt
 - jedenfalls wenn der Benutzer keine Fehler macht





Wer sind die Nutzer von JAP?

Motivation

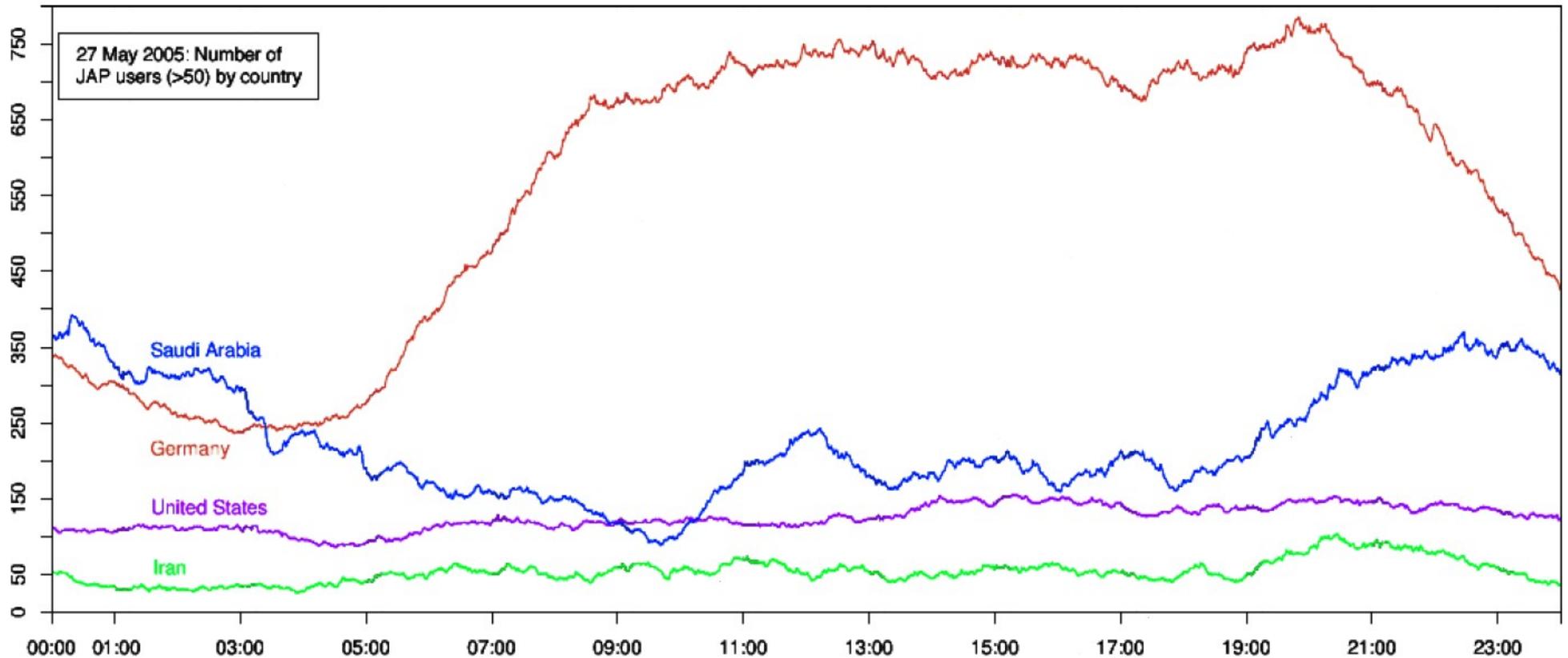
Mixe

JAP

TOR

Freenet

- Europa 60%
- Asien 27%
- USA 12%
- Rest 1%



Quelle: <http://anon.inf.tu-dresden.de>





Und wie verhalten sie sich?

Motivation

Mixe

JAP

TOR

Freenet

- Anfragen von Strafverfolgern und Privatpersonen zwecks Rückverfolgung einer IP-Adresse

| Jahr | Anfragen gesamt | Anfragen von Strafverfolgungsbehörden | Anfragen von Privatpersonen | |
|------|-----------------|---------------------------------------|-----------------------------|-------------------|
| 2001 | 13 | 7 | 6 | Projektstart |
| 2002 | 40 | 15 | 25 | |
| 2003 | 58 | 21 | 37 | |
| 2004 | 61 | 38 | 23 | |
| 2005 | 42 | 27 | 15 | |
| 2006 | 7 | 4 | 3 | Bis 31. März 2006 |

Quelle: <https://www.datenschutzzentrum.de/projekte/anon>





Grenzen des Ansatzes

Motivation

Mixe

JAP

TOR

Freenet

- Verschleiert nur die IP-Adresse
 - Cookies, Identifizierung durch Quasi-Identifizier in Suchmaschinen- oder Formulardaten möglich
 - Aufmerksamkeit, Kenntnisse vom Nutzer gefordert
- Statische Kaskaden
 - Wenn Angreifer errät, welche Kaskade eine Zielperson nutzt, genügt der Angriff von 3 Rechnern
- Beschränkt auf Webseiten-Zugriff
 - allerdings eher aus Performanz-Gründen; kein Problem des Konzepts
- Performanz, Verfügbarkeit
 - Mix-Server einer Kaskade sind Single Point of Failure und Bottleneck für viele Nutzer





Vorratsdatenspeicherung und JAP

Motivation

Mixe

[JAP](#)

TOR

Freenet

- 01.01.2009: Vorratsdatenspeicherung (§113a TKG)
 - Speicherung von Verkehrsdaten (nicht: Inhalte)
- Umgesetzt in JAP
 - der erste Mix speichert IP-Adresse, Zeit, Kanalnummer zum nächsten Mix
 - mittlere Mixe speichern eingehende und ausgehende Kanalnummern, Zeit
 - ausgehende Mixe speichern die eingehende Kanalnummer, Zeit, Portadresse im Internet
- Nicht gespeichert: URL, IP des kontaktierten Servers
 - *Behörden müssten Daten von Webserver und allen Mixen einer Kaskade kombinieren; es genügt ein Mix im Ausland*





Verschleierung der IP-Adresse mit TOR



TOR



Motivation

Mixe

JAP

TOR

Freenet

- Projekt der Electronic Frontier Foundation, entwickelt etwa ab 2004
- Im Vergleich zu JAP modernerer Ansatz
 - *Onion-Routing* (nichts anderes als das Mix-Modell)
 - Hidden Services: Anonyme Serveradressen!
 - Unterstützt beliebige TCP-basierte Protokolle
 - http, P2P-Protokolle oder ICQ
 - Zufällige Verbindungen für jede Netzwerkverbindung
 - d.h. Kanal wird neu gewählt, sobald Server die Verbindung schließt.
 - von der Browserimplementierung abhängig, ggf. wird jedes Bild und jede HTML-Seite scheinbar von anderem Rechner angefordert
 - keine statischen Mix-Kaskaden



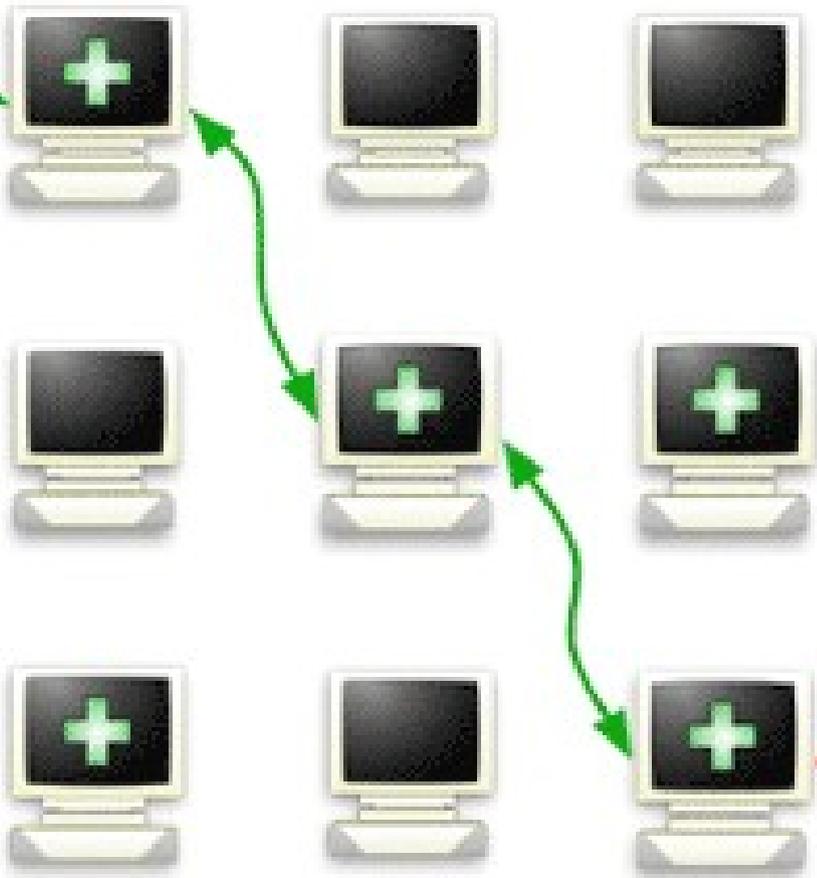
Wie Tor funktioniert: 1



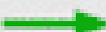
Wie Tor funktioniert: 2

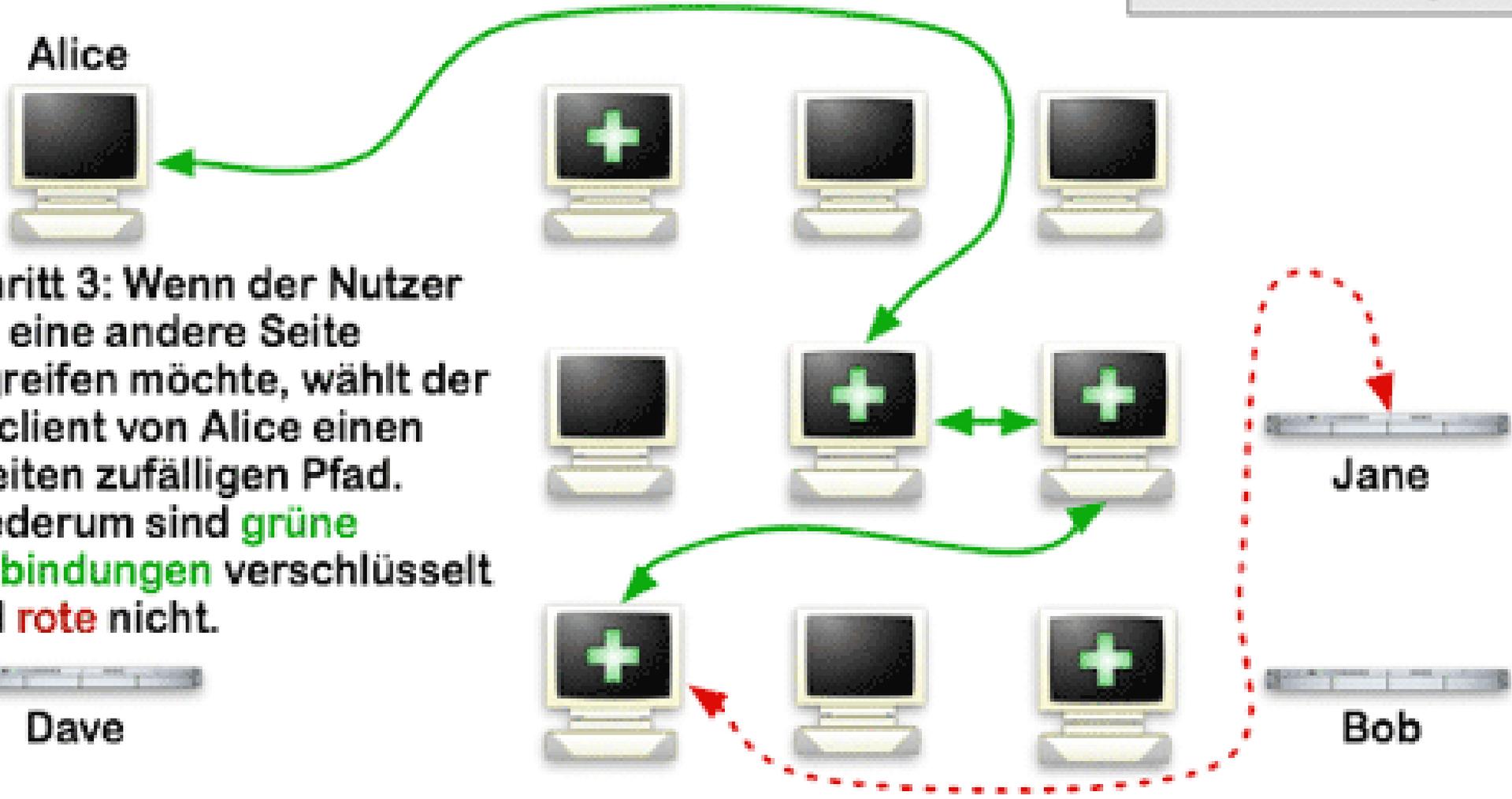


Schritt 2: Der Torclient von Alice wählt einen zufälligen Pfad zum Zielserver.
Grüne Verbindungen sind verschlüsselt, rote Verbindungen nicht.



Wie Tor funktioniert: 3

-  Torknoten
-  unverschlüsselte Verbindung
-  verschlüsselte Verbindung



Schritt 3: Wenn der Nutzer auf eine andere Seite zugreifen möchte, wählt der Torclient von Alice einen zweiten zufälligen Pfad. Wiederum sind grüne Verbindungen verschlüsselt und rote nicht.



Hidden Services

Motivation

Mixe

JAP

TOR

Freenet

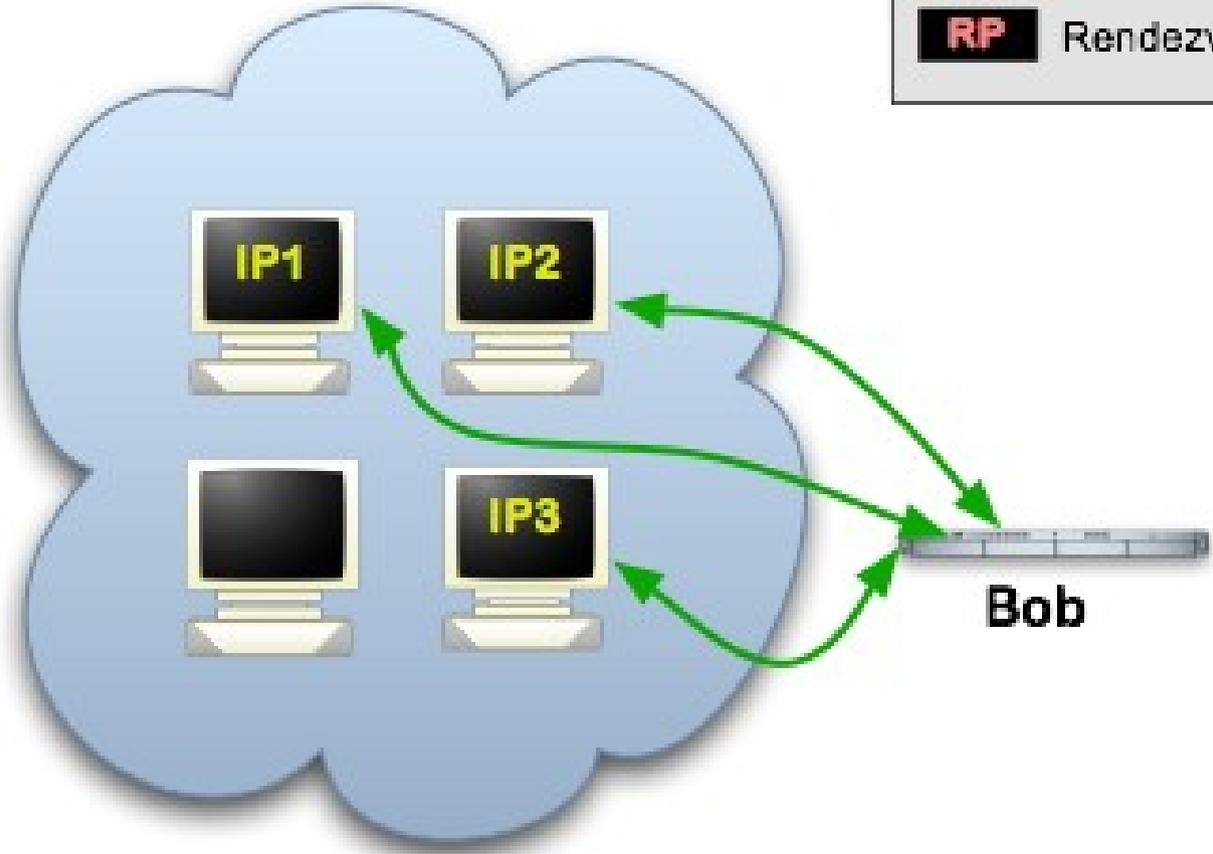
- Idee: Server (z.B. Blogs, News etc.) anbieten, ohne die eigene Identität preiszugeben
 - herkömmliches WWW: whois-Anfrage verrät Betreiber
- Funktionsweise:
 - einige TOR-Knoten als Introduction Points, die Verbindungen zum Server herstellen
 - TOR-Knoten kennen IP-Adresse vom Server nicht, da über TOR verborgen
 - Descriptor auf Verzeichnisserver, der Public Keys und Introduction Points enthält
 - über Descriptor können TOR-Nutzer anonyme Verbindung zum Server aufbauen



Tor Hidden Services: 1

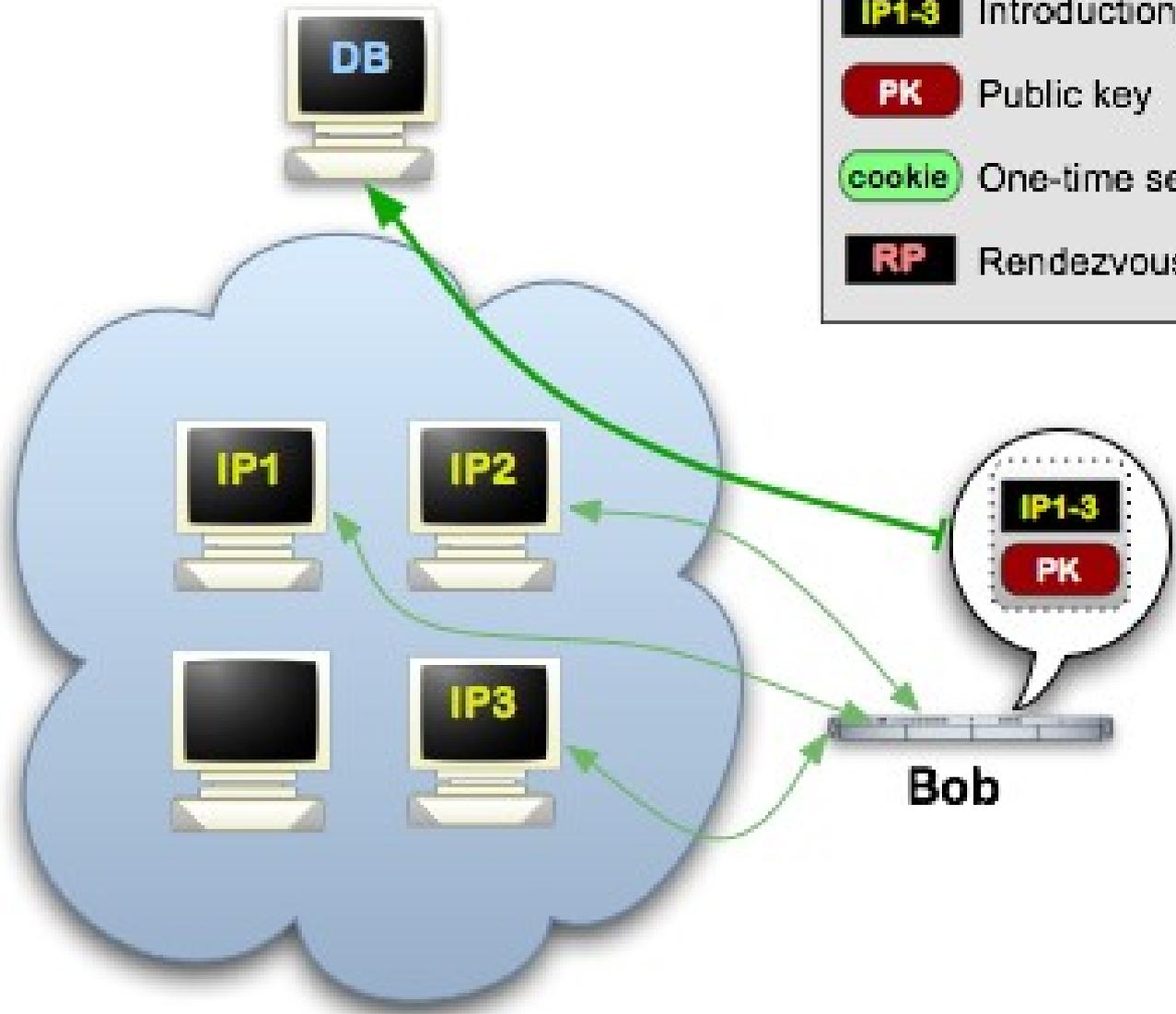
-  Tor cloud
-  Tor circuit
-  Introduction points
-  Public key
-  One-time secret
-  Rendezvous point

Step 1: Bob picks some introduction points and builds circuits to them.



Tor Hidden Services: 2

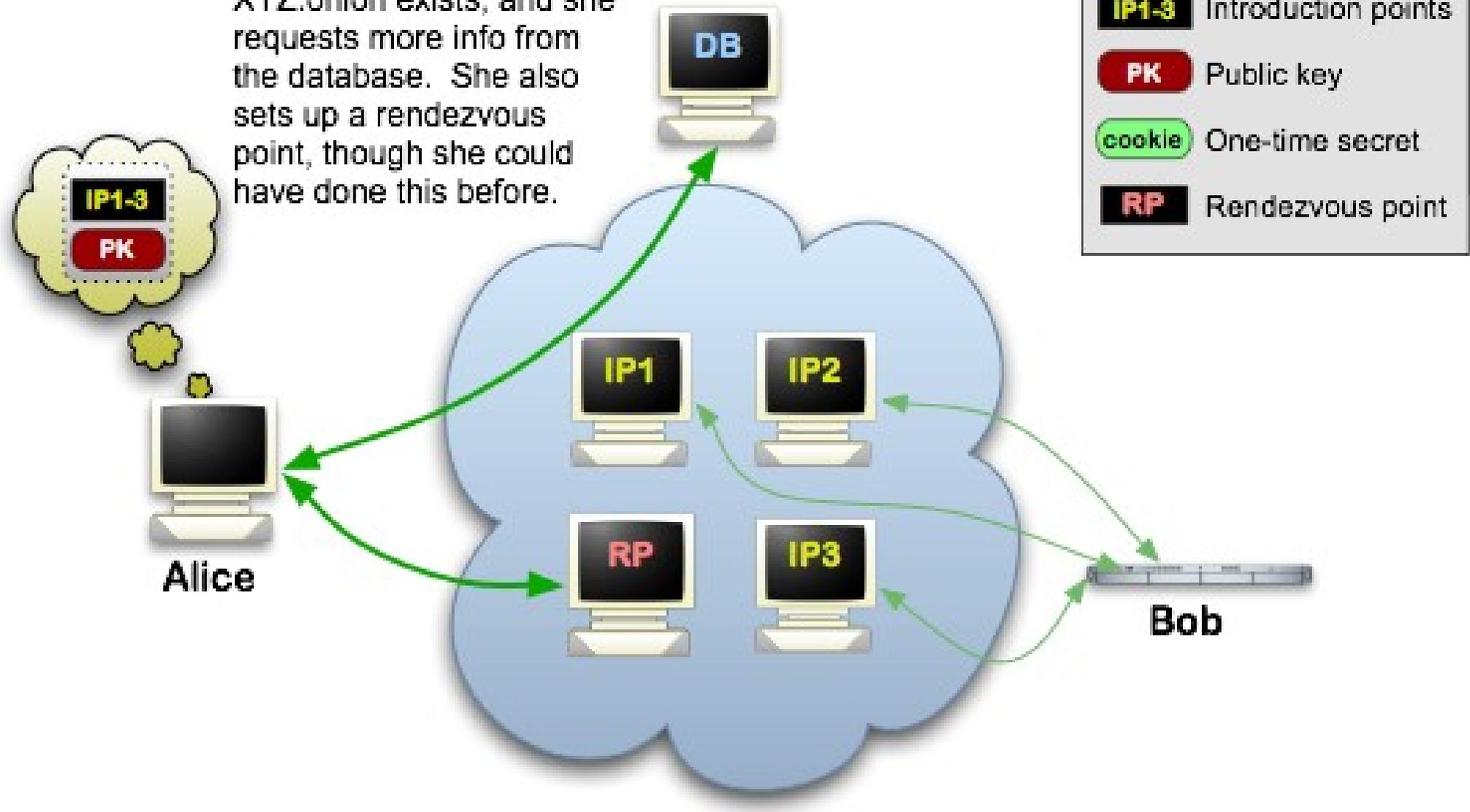
Step 2: Bob advertises his hidden service -- XYZ.onion -- at the database.



-  Tor cloud
-  Tor circuit
-  Introduction points
-  Public key
-  One-time secret
-  Rendezvous point

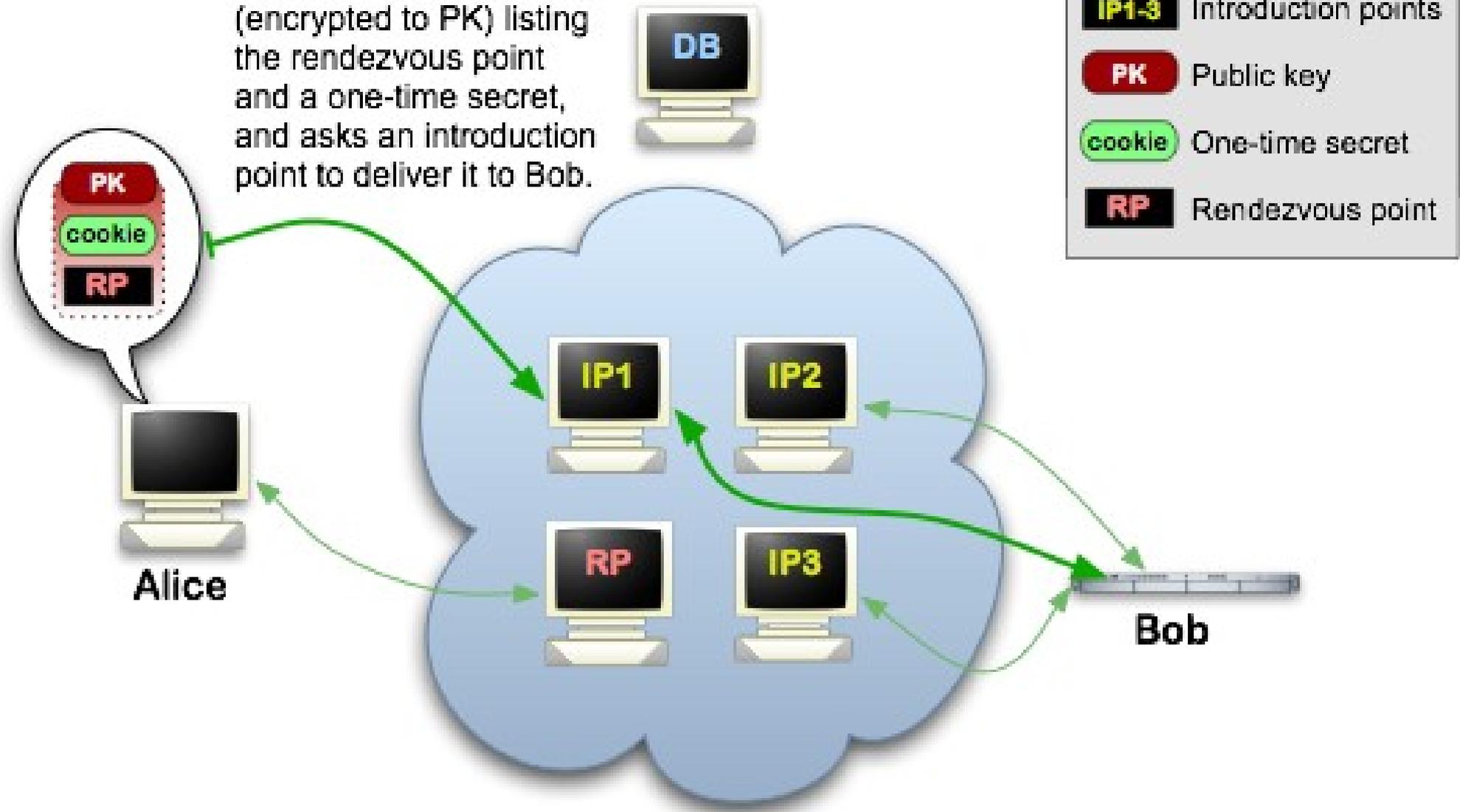
Tor Hidden Services: 3

Step 3: Alice hears that XYZ.onion exists, and she requests more info from the database. She also sets up a rendezvous point, though she could have done this before.



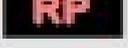
Tor Hidden Services: 4

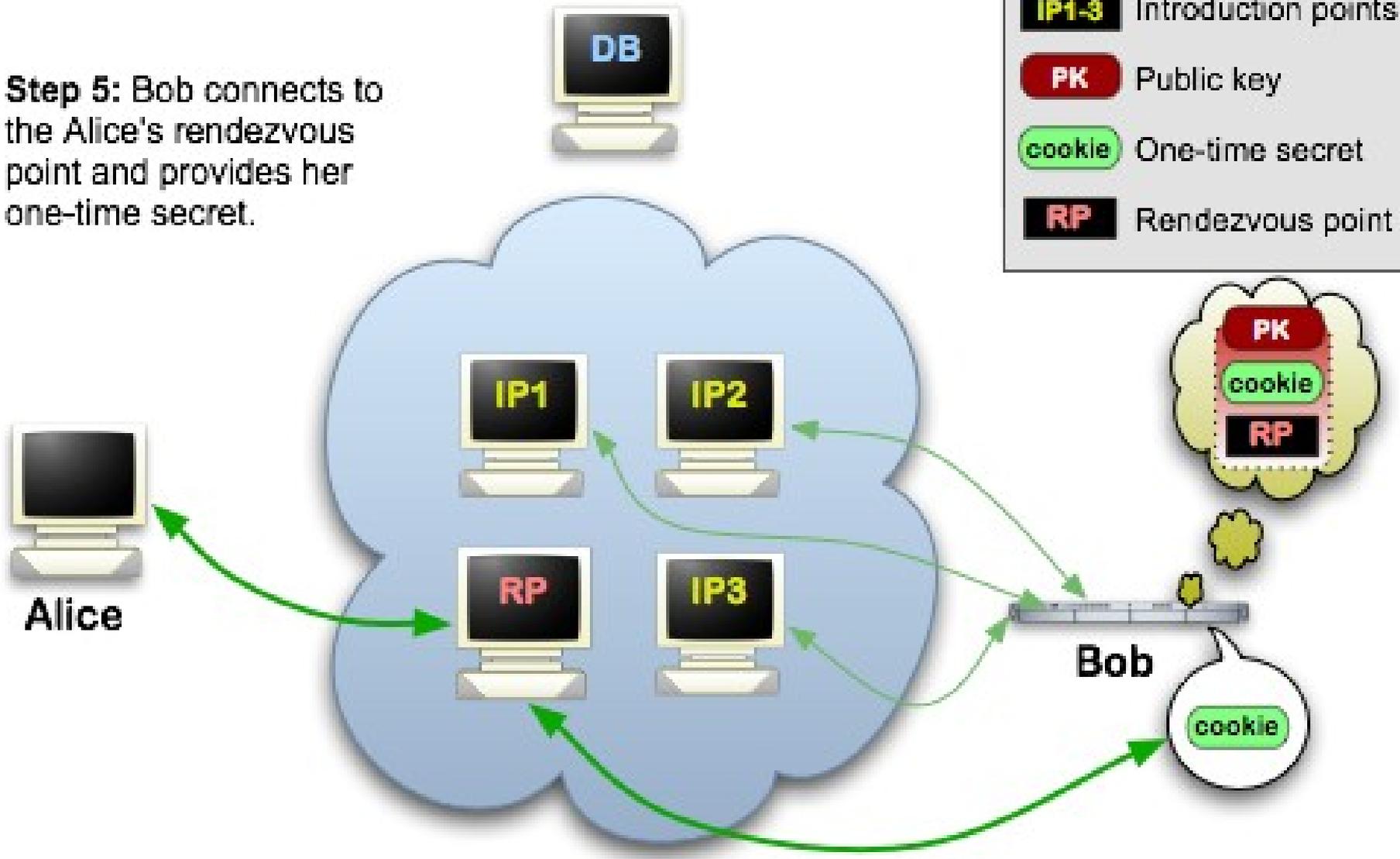
Step 4: Alice writes a message to Bob (encrypted to PK) listing the rendezvous point and a one-time secret, and asks an introduction point to deliver it to Bob.



Tor Hidden Services: 5

Step 5: Bob connects to the Alice's rendezvous point and provides her one-time secret.

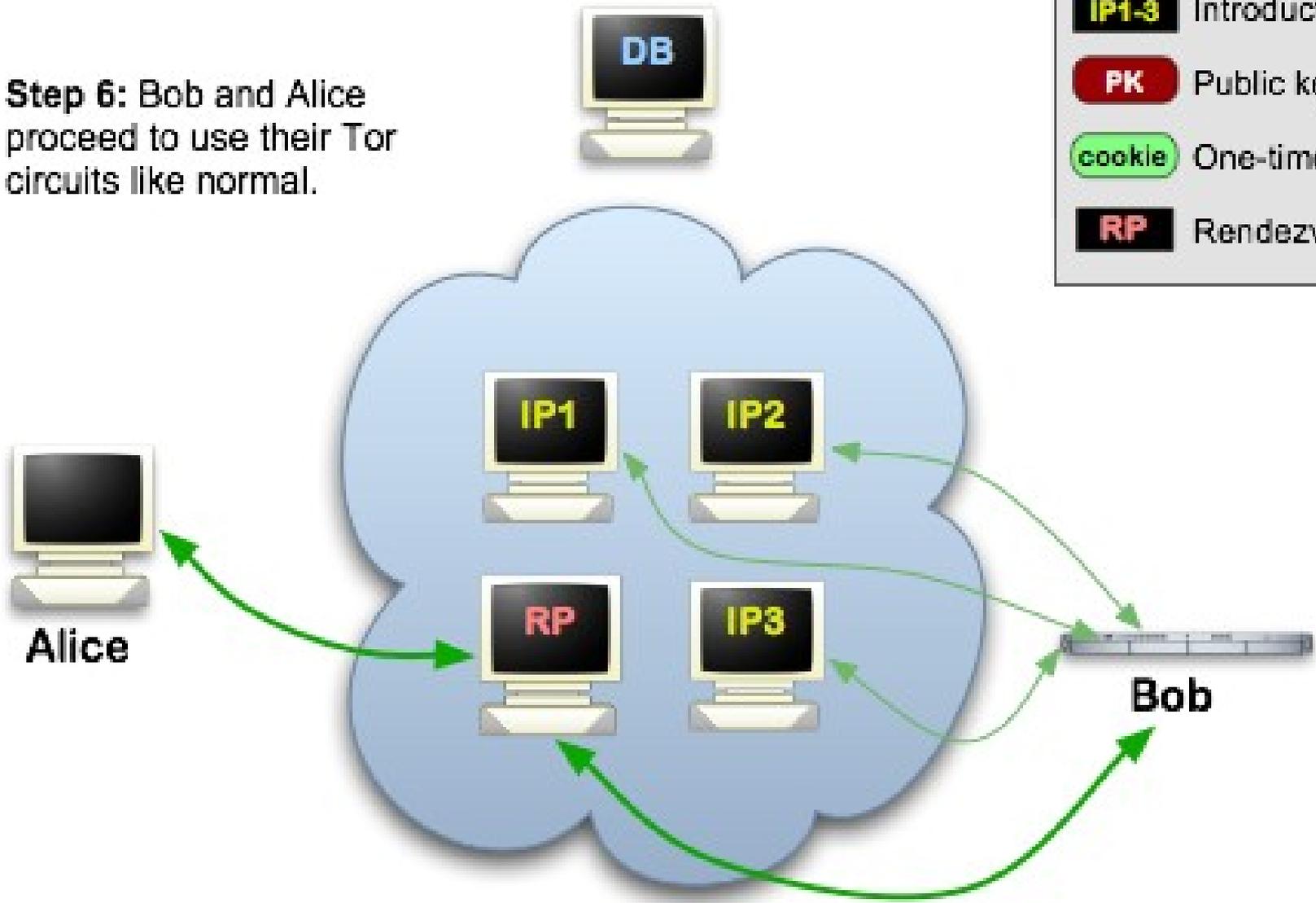
-  Tor cloud
-  Tor circuit
-  Introduction points
-  Public key
-  One-time secret
-  Rendezvous point



Tor Hidden Services: 6

Step 6: Bob and Alice proceed to use their Tor circuits like normal.

-  Tor cloud
-  Tor circuit
-  Introduction points
-  Public key
-  One-time secret
-  Rendezvous point





Zahlen zu TOR

Motivation

Mixe

JAP

TOR

Freenet

- Zahlen von 2007, Quelle: <http://www.heise.de>
 - ca. 700 TOR-Knoten, welche die Verschlüsselung und den Weitertransport übernehmen
 - ca. 250 Exit-Knoten mit mehr als 20kB/s, die Daten ins Internet weiterleiten
 - Performanz bei DSL (1024kBit/s Downstream)
 - ohne TOR: Downloadgeschwindigkeit 112kB/s
 - mit TOR: Downloadgeschwindigkeit 15kB/s





Grenzen des Ansatzes

Motivation

Mixe

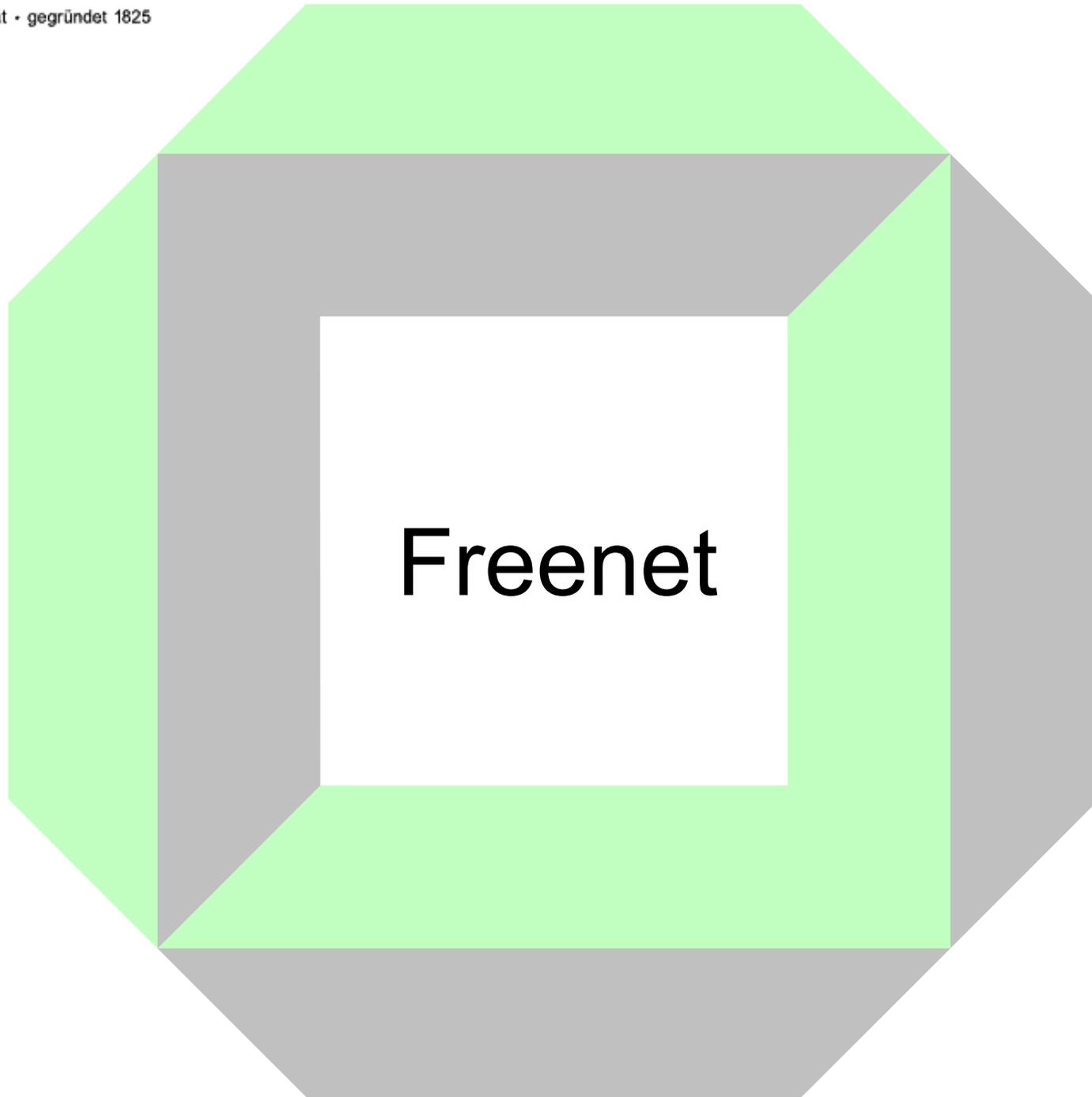
JAP

TOR

Freenet

- Ebenso wie bei JAP:
 - Datenspuren auf verschiedenen 'logischen Ebenen'; IP-Adresse ist nicht alles
 - Web-Bugs, Cookies, Scripte etc. können den Nutzer identifizieren
- einige praktische Probleme
 - einschleusen von TOR-Knoten mit sehr vielen freien Ressourcen (oder Manipulation dieser Angaben), diese werden vom Netz bevorzugt genutzt
 - durch Kontrolle weniger Rechner große Teile des Verkehrs *vollständig* belauschen
 - TCP-Zeitstempel werden von jedem Betriebssystem anders gesetzt, hängen von der Rechnerhardware ab
 - statistische Angriffe möglich







Freenet

Motivation

Mixe

JAP

TOR

Freenet

- Ziel: **anonymer Datenaustausch, Zensurresistenz**
 - TOR Hidden Services: es existiert immernoch ein Server im Netz, der Daten speichert und einer Person zugeordnet ist
 - Freenet: *Abstreitbarkeit* für Informationsanbieter
 - Daten anonym und verteilt speichern, Peer-to-Peer Ansatz
 - keine Quelle, die man vom Netz nehmen könnte → Löschung und Zensur unmöglich
- entwickelt seit 2000, Open Source
<http://freenetproject.org>
- *seit 2008 "Darknet" Feature, direkte Kommunikation nur mit vertrauenswürdigen Knoten*





Das Peer-to-Peer Modell

Motivation

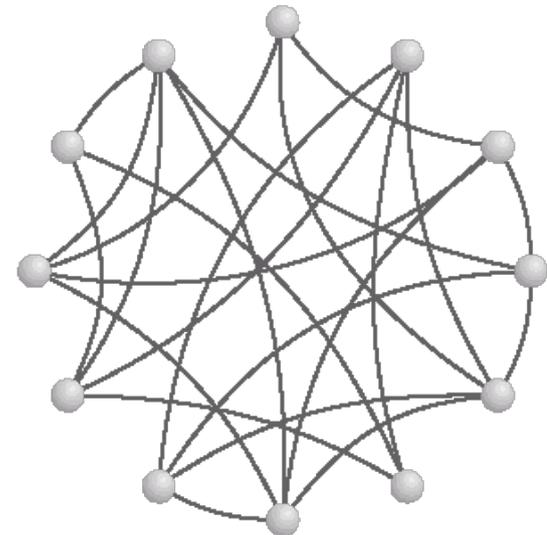
Mixe

JAP

TOR

[Freenet](#)

- viele *unabhängige* Knoten
- jeder Knoten kennt einige ausgewählte (strukturierte P2P-Systeme) oder zufällige (unstrukturierte Systeme) andere Knoten
 - lokales, unvollst. Wissen über Netzwerktopologie
- jeder Knoten leitet Nachrichten an einen anderen weiter, der in der Netzwerktopologie näher am Ziel ist
 - Small-World-Eigenschaft (vgl. Milgram-Experiment '69 6 degrees of separation)





Freenet-Architektur

Motivation

Mixe

JAP

TOR

Freenet

- Jeder Knoten speichert
 - Daten
 - Routing-Tabelle mit ausgewählten Knoten
 - IP-Adresse und gespeicherte Daten des Knotens
- Daten werden über ortsunabhängige Schlüssel referenziert
 - Daten dürfen zwischen den Knoten “umziehen”
- Anfragen nach diesen Schlüsseln werden über Ketten von Freenet-Knoten weitergeleitet
 - Verschlüsselung zwischen den Knoten
 - Routing-Topologie lernt über die Zeit





Referenzierung über Schlüssel

Motivation

Mixe

JAP

TOR

[Freenet](#)

- Zwei relevante Schlüsseltypen (keyword-signed keys sind nicht mehr im Einsatz)
 - **Content-hash key**, krypt. Signatur der Daten
 - vergleichbar mit Inodes im Dateisystem
 - **Signed-subspace key**, Namespaces, in denen jeder lesen, aber nur der Ersteller schreiben kann
 - vergleichbar mit Datei- und Verzeichnisnamen
- Verbreitung der Schlüssel als Lesezeichen, in öffentlichen Directories oder über Freenet-Suchmaschinen





Daten abrufen (1/2)

Motivation

Mixe

JAP

TOR

[Freenet](#)

1. Schlüssel beschaffen
2. Anfrage mit Schlüssel an einen Freenet-Knoten senden
3. Knoten erhält Anfragen
 - merken, von wem die Nachricht kam
 - zum Schlüssel passende Daten lokal gespeichert?
 - Ja: Antwort zurücksenden
 - Nein: Anfrage an den Knoten senden, dessen Schlüsselbereiche dem gesuchten Schlüssel am ähnlichsten sind (*nächste Folie*)
4. Knoten leitet Antwort zurück
 - ggf. Daten im eigenen Repository zwischenspeichern, anderen mitteilen, dass er nun diese Daten kennt
 - ggf. Routingtabelle updaten





Daten abrufen (2/2)

Motivation

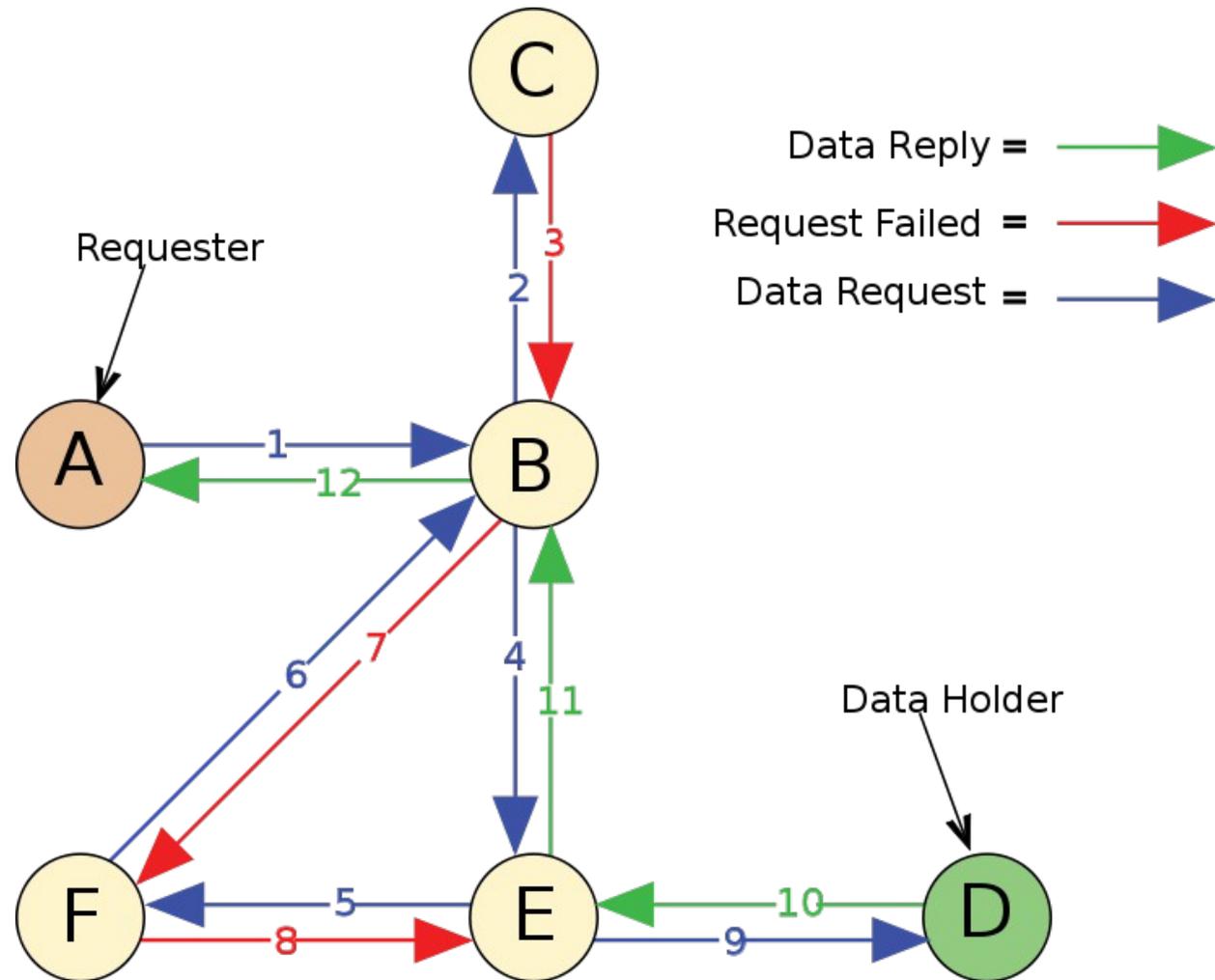
Mixe

JAP

TOR

[Freenet](#)

- Query Routing in Freenet: Hillclimbing + Backtracking





Anonymität in Freenet

Motivation

Mixe

JAP

TOR

Freenet

- Anfrager bleibt anonym
 - Nachricht sucht sich ihr Ziel anhand Schlüssel selbst,
 - Verbindungen nur jeweils zwischen zwei Knoten,
 - dem Schlüssel ist nicht anzusehen, was er bezeichnet
- Ersteller der Daten bleibt anonym
 - Daten werden durch ihre Schlüssel adressiert, und an beliebiger Stelle ins Freenet eingespeist
- Zensurresistenz
 - Daten können prinzipiell an beliebigen Stellen gespeichert werden
 - beliebte Daten häufig in den Caches der Knoten
 - Daten werden nur verschlüsselt abgespeichert, d.h. Freenet-Betreiber weiß nicht welche Daten er hostet





Das Darknet-Feature in Freenet

Motivation

Mixe

JAP

TOR

Freenet

- Grundsätzliches Problem in allen anonymen Netzen: das Einschleusen von manipulierten Knoten
 - mit wenigen Rechnern/Ressourcen gezielt angreifen, wenn “die richtigen Stellen” in der Netztopologie bekannt/erraten
- Darknet
 - abgeschlossenes, privates Friend-to-Friend-Netzwerk
 - keine Kommunikation mit unbekanntem Knoten, daher Einschleusen erschwert
 - Herausforderung: trotzdem ein verbundenes Netzwerk schaffen





Umsetzung von Darknet

Motivation

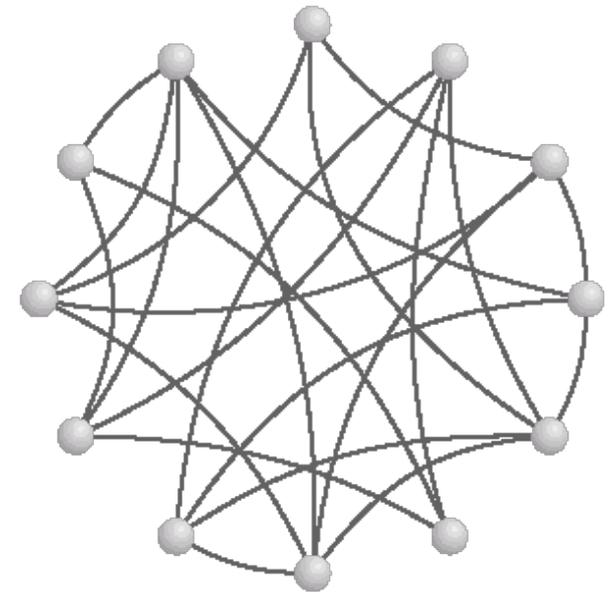
Mixe

JAP

TOR

[Freenet](#)

- Routingtabellen werden nicht mehr selbstlernend von den Knoten angelegt, sondern von Hand
 - Betreiber kennen sich
- Implementierung eines strukturierten Overlays in Form eines Ringes, auf das Schlüssel im Interval $[0, 1]$ abgebildet werden
- Knoten können Plätze im Ring tauschen
 - es bilden sich Cluster von Knoten, die
 - effizient kommunizieren und
 - ähnliche Daten speichern





Grenzen des Ansatzes (1/2)

Motivation

Mixe

JAP

TOR

[Freenet](#)

- Keine Garantien
 - Routing-Verfahren kann Daten nicht finden, obwohl sie im Netz gespeichert sind
 - selten gesuchte Daten können verschwinden (Least-Recently-Used-Ansatz)
 - der erste gefragte Knoten kann gleichzeitig Ersteller und Speicherort der Daten sein
 - erfährt unmittelbar, wer etwas wissen will
- Skalierbarkeit bzgl. Zahl der Knoten, Pfadlänge
 - Small-World-Ansatz ist wenig effizient; strukturierte P2P-Systeme wären besser
 - Darknet-Erweiterung löst das Problem teilweise
- Darknet erfordert, andere Darknet-Nutzer zu kennen





Grenzen des Ansatzes (2/2)

Motivation

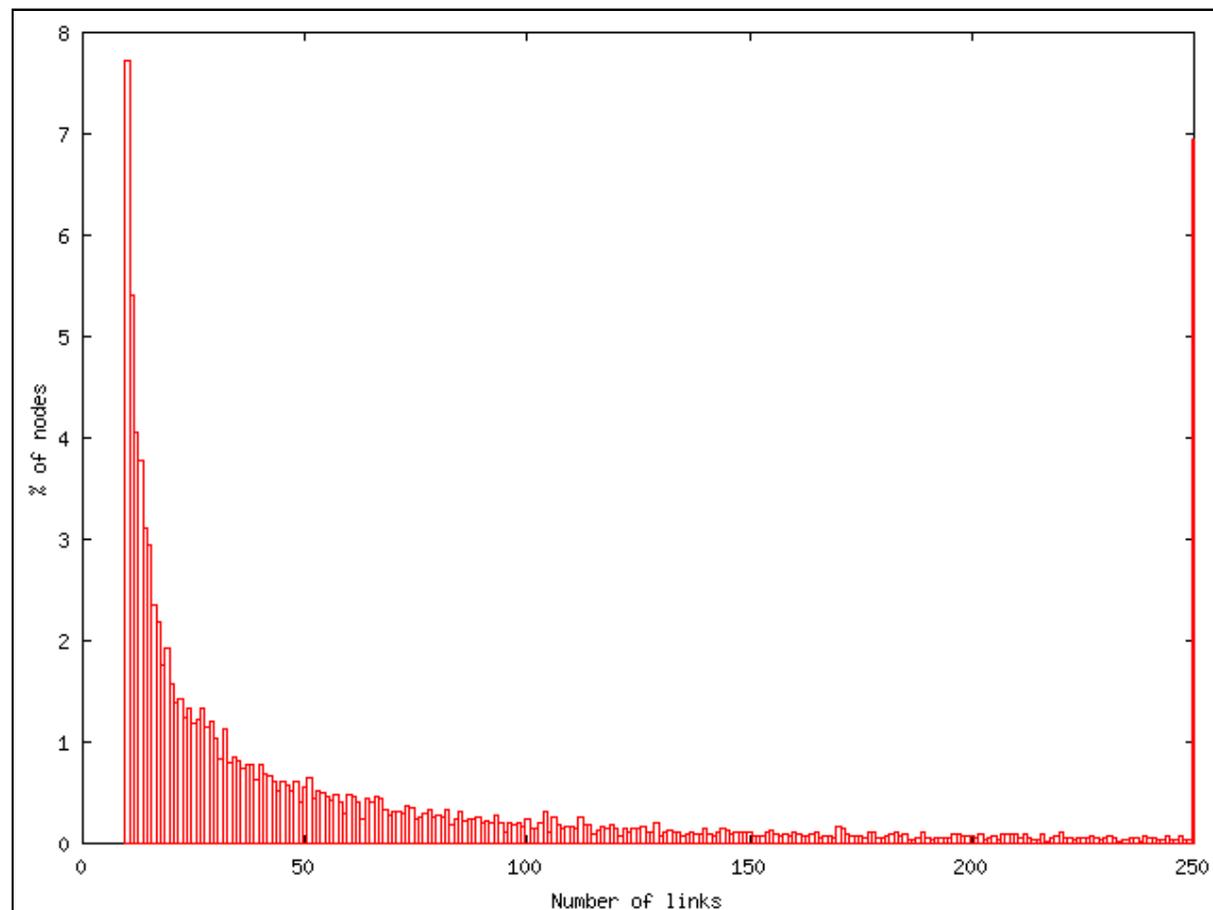
Mixe

JAP

TOR

[Freenet](#)

- Skalenfreies Netz, einige wenige Knoten wichtiger als andere → Angriffspunkte!



Quelle: Topics in Reliable Distributed Computing, Yoav Levy 2004





Zusammenfassung



Zusammenfassung

- Datenspuren im Internet
 - auf Rechner des Nutzers, Internet-Anbieter, Weiterleiter, (Web-, DNS-, sonstige) Server, P2P, Analysedienste von Dritten
 - basieren auf Log-Informationen des HTTP-Protokolls
Cookies, Web-Bugs
- Bisher betrachtete Dienste und Protokolle
 - WWW, TCP/IP
- Bisher betrachtete Datenschutzansätze
 - P3P, JAP, TOR, Freenet





Literatur

- [1] Roger Dingledine et al.: *Tor: The Second-Generation Onion Router*, Proceedings of the SSYM 2004
<http://freehaven.net/tor/tor-design.pdf>
- [2] Ian Clarke et al.; *Freenet: A Distributed Anonymous Information Storage and Retrieval System*, LNCS 2009
<http://citeseer.ist.psu.edu/old/clarke00freenet.html>

