Lecture 15

Internet Protokolle II

Fairness & Anonymity

Thomas Fuhrmann

Network Architectures
Computer Science Department
Technical University Munich

PlanetLab

- Ca. 350 Standorte weltweit, meist Universitäten.
- Min. 2 Linux Maschinen vor Ort.
- Mittels VServer können dort (beliebig) viele virtuelle Linux Maschinen erzeugt werden.
- Verwaltungswerkzeuge erlauben es Forschergruppen, dort eigene Programme laufen zu lassen.

- Beispiele:
  - Internet-Messungen
  - Early Deployment von neuen Anwendungen
  - Aber nicht: massiv verteiltes Rechnen → Grid-Computing
  - Jedoch: Verteilte Ressourcen-Verwaltung für z.B. Grid-Computing.

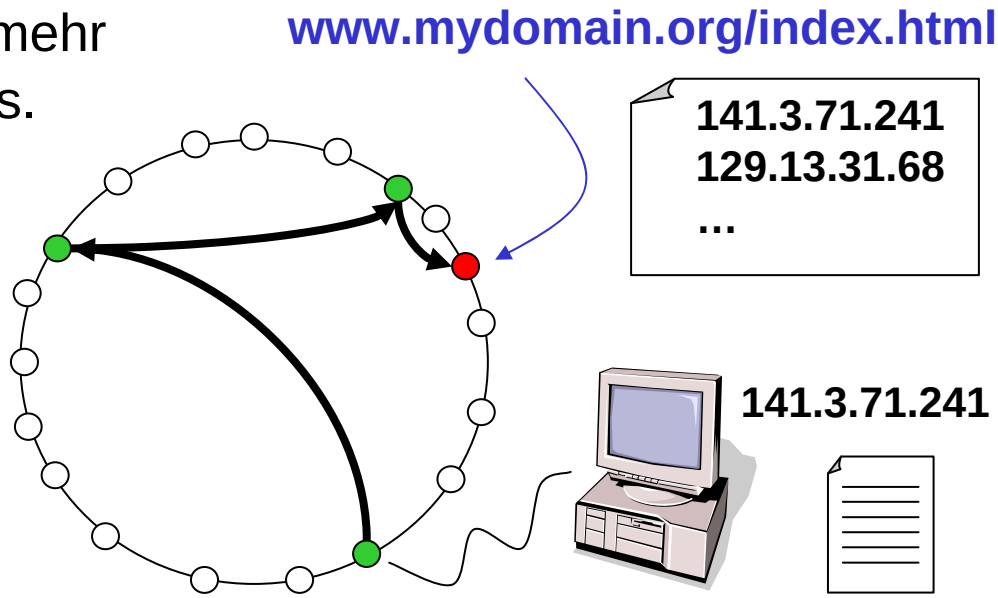
Coral CDN (1)

- Content Distribution System für WWW Inhalte
- Links zeigen auf das Coral System, dort werden sie auf die jeweiligen Web-Server aufgelöst
- Trotz Slash-Dot Effekt werden die Web-Server nicht belastet:
  - Mehr Nachfrage erzeugt mehr Kopien in den Coral-Peers.
  - DHT verwaltet Listen von Peers die Kopien speichern.
- Coral arbeitet momentan auf Basis des PlanetLab Testbetts.

"You make it fun; we'll make it run"

CORAL

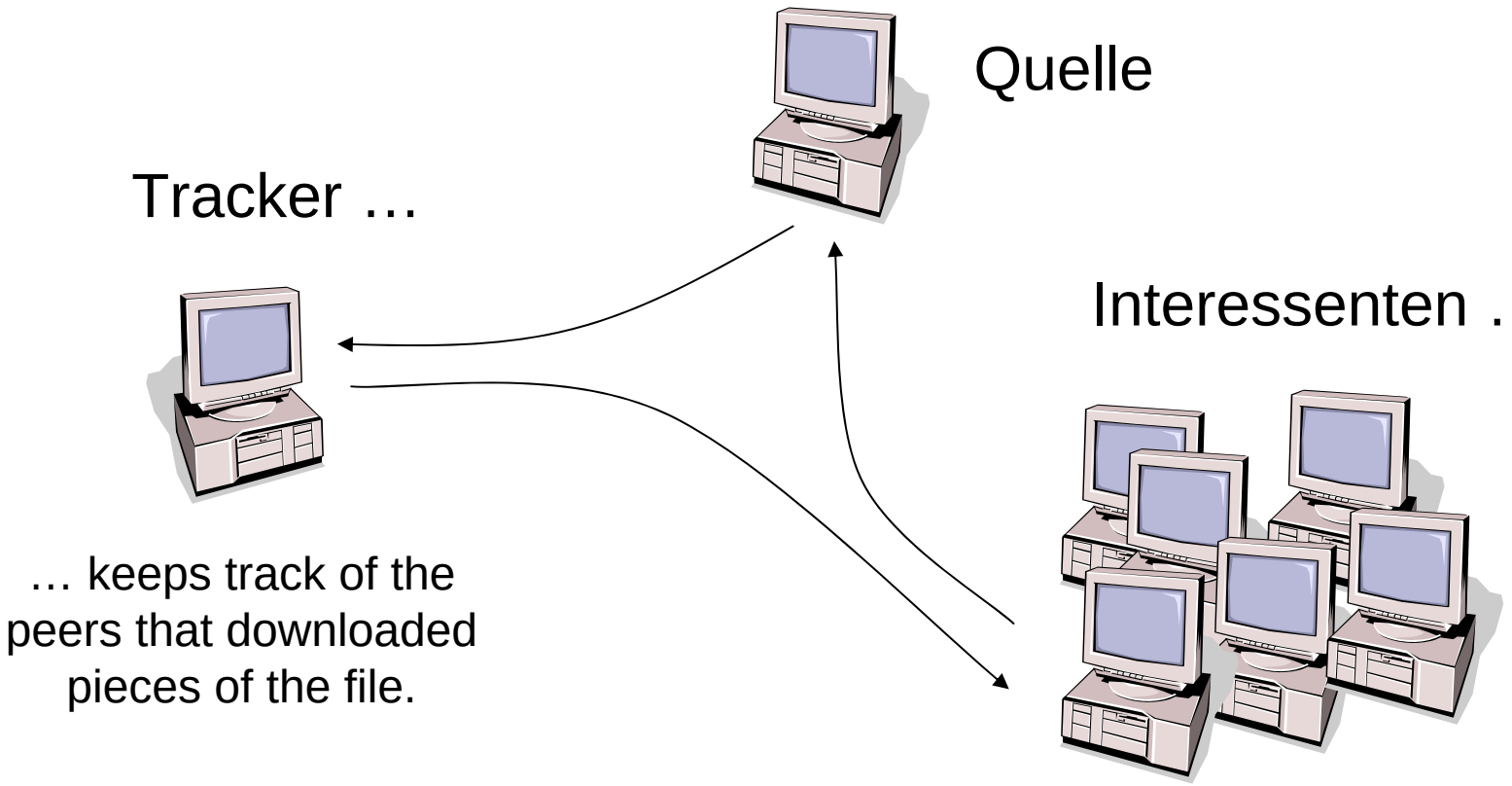
The Coral Content Distribution Network



Coral CDN (2)

- Design Überlegungen:
  - Content soll dort liegen, wo er tatsächlich gebraucht wird bzw. wurde (Original-WWW-Server, Coral-Proxy-Caches)
  - DHT verwaltet nur Listen dieser Maschinen.
  - Begrenzung des Anfrageverkehrs durch Caching der Anfragen in der DHT.
  - Begrenzung des Publikationsverkehrs durch Ausdünnung der Listen auf dem Weg zur Hash-Wurzel.
- Beispiel:
  - „www.mydomain.org/index.html„ wird von Browser via Coral angefragt.
  - Dazu ist eine Coral-Maschine entweder als Proxy eingetragen, oder wird via DNS-Tricks angesprochen.
  - Diese Coral-Maschine bilden den Hash der URL und fragt in Richtung auf diesen Hash-Wert nach anderen Coral-Maschinen die diese URL im Cache haben.
  - Die jetzt fragende Coral-Maschine wird dabei in die Listen eingetragen.

BitTorrent Overview (1)



… erhalten vom Tracker Adressen anderer Peers, die Stücke des Files bereits haben. Dann laden sie diese Stücke von dort, prüfen mittels kryptographischem Hash, ob der Inhalt korrekt ist, und bieten mittels Tracker das Stück für Dritte an.

BitTorrent Überblick (2)

- Bittorrent nutzt das Grundprinzip selbstorganisierender Peer-to-Peer Systeme:
  - Mehr Nachfrage erzeugt mehr Angebot!
- Bittorrent ist nicht völlig dezentral, da pro Quelle ein Tracker angegeben werden muss.
  - D.h. Quellen sind keine Peers! (vgl. Napster)
- Bittorrent ist darauf angewiesen, dass Peers geladenen Inhalt weiter verteilten:
  - Peers geben bevorzugt solchen Peers ihren Inhalt, von denen sie selbst Inhalt bekommen (=„Tit for tat“).
- Verschiedene Strategien für die Auswahl der Inhaltsstückchen führen zu verschiedener Effizienz:
  - Oft nachgefragte Stücke sind wertvoll, sowohl für die Gemeinschaft, als auch für den jeweiligen Peer, denn so ist die Chance hoch dass man damit andere Stücke erhalten kann.
  - Freizügige Peers, die ohne Gegenleistung Inhaltsstücke abgeben stabilisieren das Gesamtsystem.

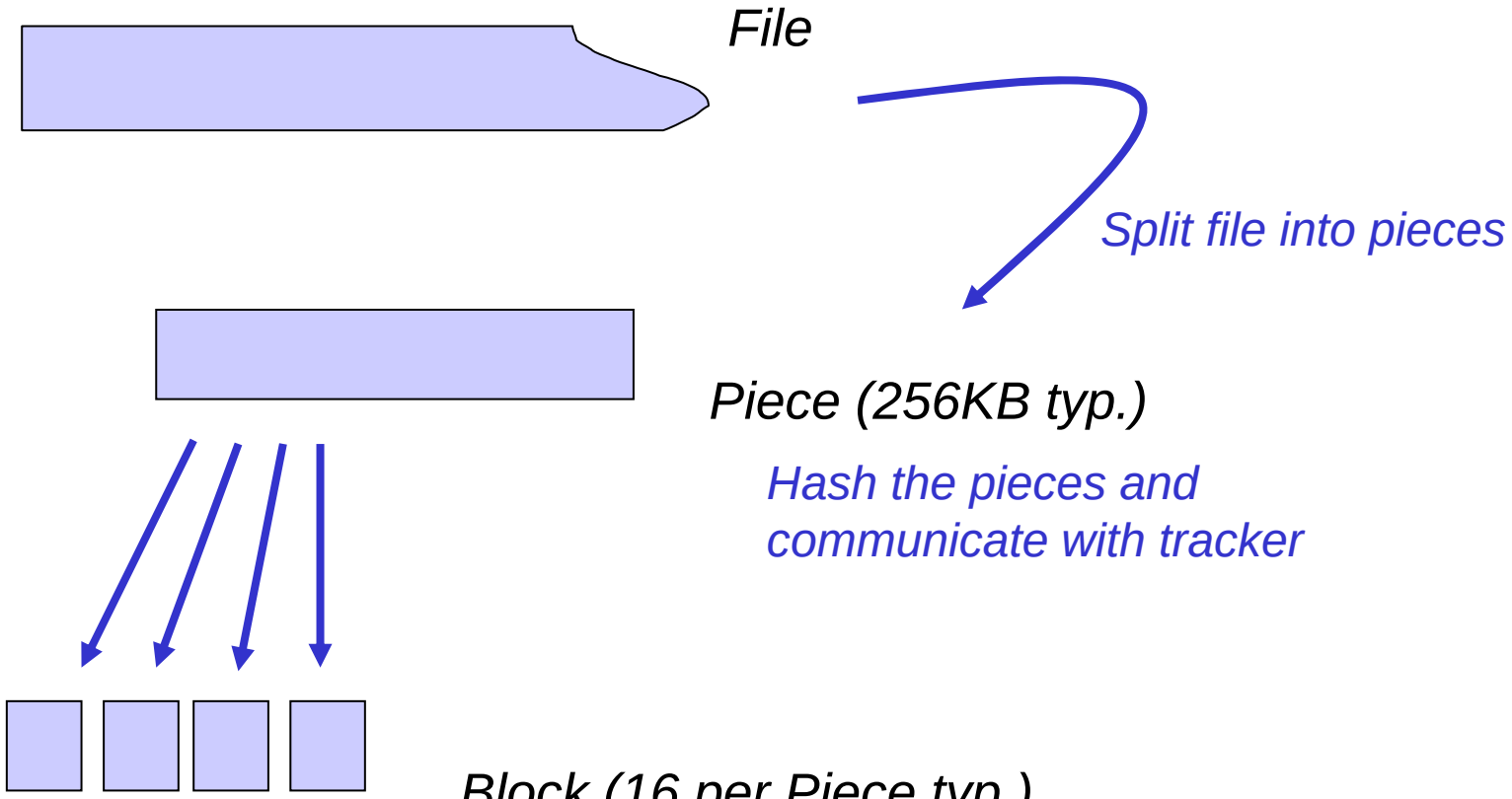
BitTorrent

- Peer-To-Peer File Sharing Protokoll
- Entwickelt von Bram Cohen
- Protokoll Design im April 2001
- 37% des deutschen Datenaufkommens, u.a. zu Verbreitung von freier Software und auch „World-of-Warcraft“
- Skaliert in der Anzahl der teilnehmenden Peers
- Ursprünglich mit Trackern zur Unterstützung bei der Koordination der Datei-Downloads
- Neuerdings mit Erweiterungen für den reinen P2P-Betrieb
- Benutzt ausgefeilte Algorithmen zur Lastverteilung zwischen den Peers:
  - Tit-for-Tat
  - Choking
  - Swarming

BitTorrent Begrifflichkeiten

- Chunks: Splitten einer Datei in mehrere Teile
  - + Schnellere Weiterverteilung
  - + Häufigeres Prüfen
  - + Schnelleres Reseed
  - + Verteilungsstrategien:
    Welcher Chunk wird geladen?
- Leecher - Knoten der das File downloadet
- Seeder - Knoten der das File an anderer uploadet
- Tracker - Server der Informationen über andere Peers bereitstellt
- .torrent – Tracker-URL und Hash-Werte für Chunks
- Swarm - Seeder und Leecher zu denen ein Peer verbunden ist

BitTorrent – Blocks and Pieces



BitTorrent Arbeitsweise

- Tracker ermöglicht zunächst das Auffinden anderer Peers
- Peers, die einen Chunk (i.e. „Piece“) der Ressource geladen haben melden dies dem Tracker
- Peers erhalten vom Tracker eine Liste anderer Peers, die Teile oder die gesamte Ressource bereits besitzen (=Seeder) oder noch herunterladen (=Leecher).
- Diese Liste von anderen Clients wird als Schwarm („Swarm”) bezeichnet.

Verschiedene Strategien sind möglich, die bestimmen, welcher Chunk als erstes geladen werden soll:

- Random First Piece: Lade einen Chunk, egal welchen
- Rarest First: Versuche den seltensten Chunk zu laden
- Strict Priority: Sobald ein Chunk geladen wurde, lade alle anderen Teile auch
- Endgame mode: Fordere ein Chunk von mehreren Peers gleichzeitig an

BitTorrent Spieltheorie

Gefangenendilemma

- A und B werden eines gemeinsamen Verbrechens beschuldigt
- Beide werden getrennt eingesperrt und beiden wird ein Vorschlag unterbreitet

	A schweigt	A verrät
B schweigt	A: 2 Jahre B: 2 Jahre	A: 1 Jahr B: 5 Jahre
B verrät	A: 5 Jahre B: 1 Jahr	A: 4 Jahre B: 4 Jahre

BitTorrent Tit for Tat

Vier Eigenschaften

- Klarheit
- Freundlichkeit
- Provozierbarkeit
- Nachsichtigkeit

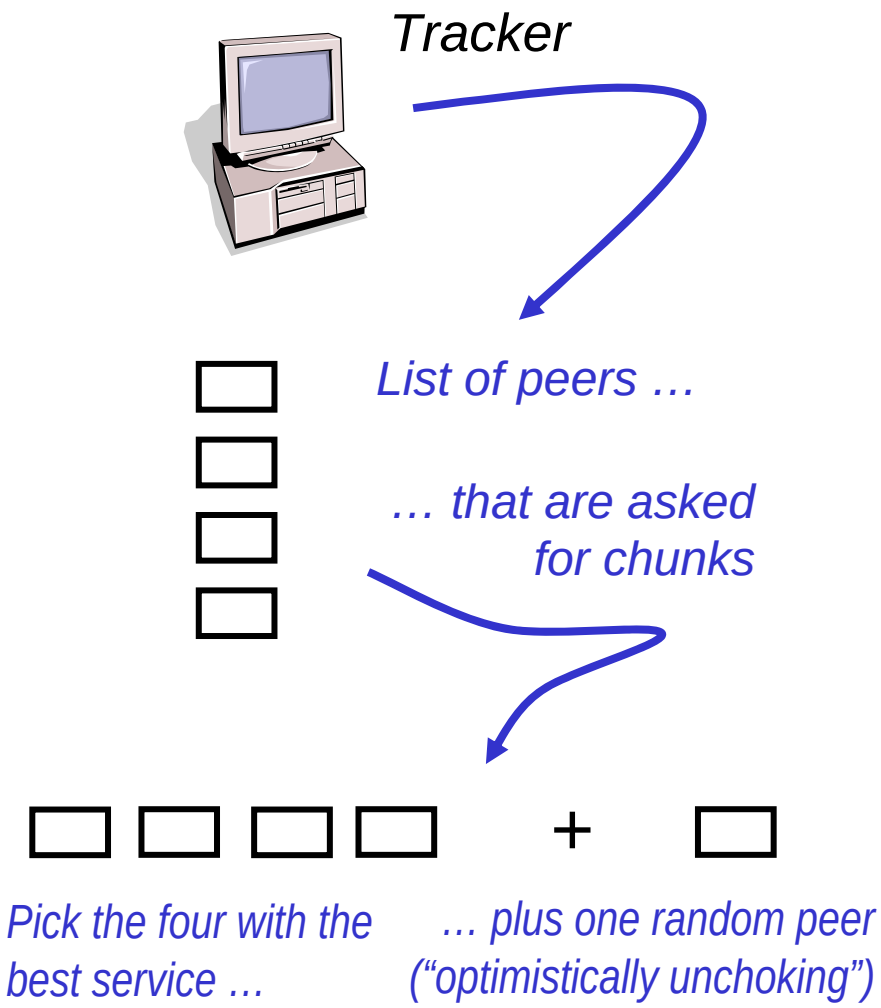
1. Der Agent kooperiert immer, solange er nicht provoziert wird
2. Wenn er provoziert wurde, wird er sich rächen
3. Der Agent vergisst schnell (nur Vorrunde)
4. Es muss dem Agenten möglich sein mehr als einmal gegen seinen Mitspieler anzutreten
5. Je langer das Spiel lauft (mehr Runde) desto besser funktioniert die Strategie.

BitTorrent ist ein wiederholtes spielen des Gefangenendilemmas

- Handlung von A ist auch gut für B -> Handlung von B wird auch für A gut sein
- Handlung von A ist auch gut für B -> Handlung von B wird auch für A gut sein

BitTorrent Choking

- Verschiedene Zustände eines Anfragenden
  - Ein interessierter Knoten will einen Chunk vom lokalen Peer.
  - Ein gedrosselter (“choked”) Knoten bekommt keine Daten vom lokalen Peer.
  - Ein abgewiesener (“snubbed”) Knoten hat in der letzten Zeit keine Daten gesendet.
- Das Protokoll läuft in Runden ab. Eine neue Runde wird nach Ablauf der Rechoke-Periode gestartet



BitTorrent Choking

```
choke_round(int: round_nr) {
    // Optimistic Unchoke
   if ((round_nr % 3) == 0) {
        opt_unchoke = select_snubbed();
        opt_unchoke += select_choked();
   }
   sort(interested_clients, bandwith)
   for (int: i = 0; i < nr_uploadslots; i++) {
        unchoke(interested_clients[i]);
   }
   unchoke(opt_unchoke);
}
```

Der Seeder Algorithmus unterscheidet sich in einigen Versionen von BitTorrent leicht vom angegebenen, um Free-Riding zu verhindern. Ein Client konnte nur 4 mal unchoked werden. Diese Verbesserung wurde aber inzwischen wieder kassiert.

BitTorrent Swarming

- Jeder Peer hat eine eigene Sicht auf den Schwarm
- Die Liste aller Clients wird vom Tracker verwaltet
- Der Tracker liefert eine Teilmenge dieser Liste an den anfragenden Client zurück
- Clients können via Peer Exchange Protokoll ebenfalls bekannte Clients austauschen
- Ansatzpunkt für verschiedene Optimierungen:
  - Azureus: Vivaldi Netzkoordinaten
  - Azureus: Ono Plugin
- Spannender: Trackerless Betrieb, reiner P2P Modus.
- Leider derzeit keine einheitliche P2P Implementierung
  - Lässt scheinbar einige wichtige Bestandteile von Kademlia weg ;)
- Implementierung von Swarming in unsrer DHT
- Zwei Ansätze:
  - Mehrere Schwarme als global eindeutige Untermengen aller beteiligten Knoten an einem File
  - Ein Schwarm mit zufälligen Knoten aus der Gesamtknotenmenge unabhängig für jeden Peer

BitTorrent P2P Swarming

Schwarmmanagement - Ansatz „eindeutiger Schwarm“

- Schwarmliste in **DHT** verwalltet
- Seeder teilt einem anfragendem Peer die SchwarmID des kleinsten Schwarmes mit falls er nicht schon eine SchwarmID gesetzt hat
- Falls alle Schwarme zu groß sind um noch einen Peer aufzunehmen ➔ Aufteilung des Schwarmes
  - Neue SchwarmID erstellen und die hälfte der Knoten eines Schwarms in den neuen Schwarm verschieben
  - Benachrichtung der Knoten mit neuer SchwarmID
  - Wenn ein Schwarm zu klein wird müssen dessen Knoten wieder auf andere Schwärme aufgeteilt werden bzw. verschmolzen

+ 1:1 Abbildung auf Multicast Gruppen möglich
+ Effizienter Nachrichtenaustausch in dieser Untermenge

- Hoher Verwaltungsaufwand
- Merge / Split Operationen
- Wer bestimmt, ob der Schwarm zu groß ist

BitTorrent P2P Swarming

Schwarmmanagement - Ansatz „pro Peer swarm“

- Schwarmliste wird nicht global eindeutig verwalltet
- Bei jeder Anfrage nach Chunks merkt sich der Seeder den gesetzten Swarm
- Falls ein Client keinen gesetzten Swarm hat, schickt der Seeder neben daten auch noch eine Liste mit Swarm/Client Tupeln mit
- Clients können dann den Swarm kontaktieren und beitreten

+ 1:1 Abbildung auf Multicast Gruppen möglich
+ Effizienter Nachrichtenaustausch in dieser Untermenge

- Wenig Verwaltungsaufwand
- Chaotische Verwaltung
- Nicht immer beste Gruppenverteilung

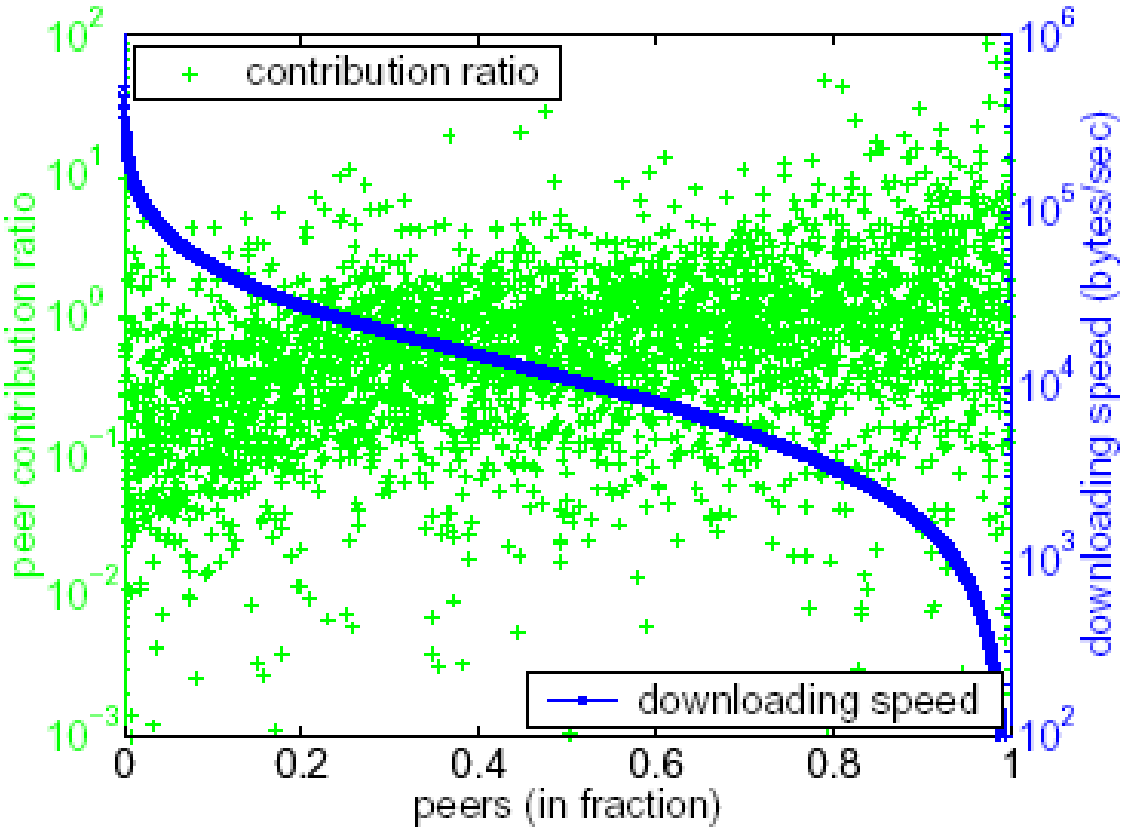
BitTorrent P2P Swarming

Lust auf Simulationen zu dem Thema?

elser@so.in.tum.de

Wir haben kekse ☺

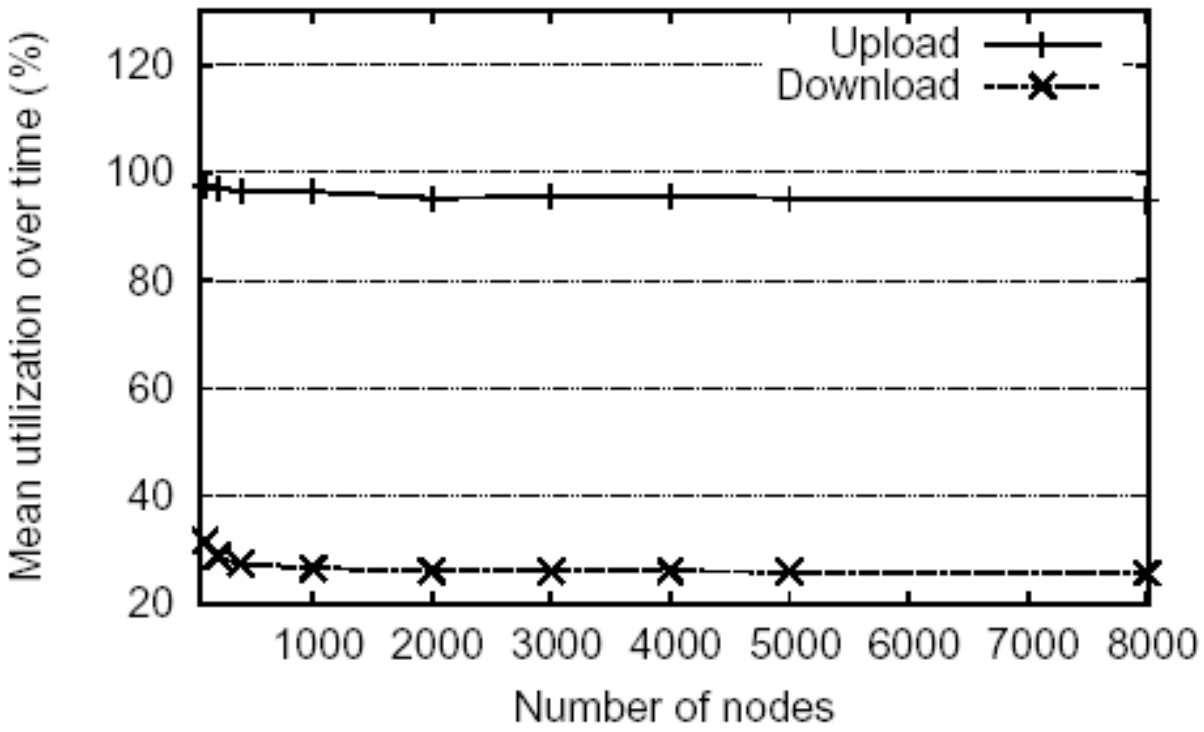
BitTorrent Ergebnisse



Teils unfair: Mit steigender Downlaodrate nimmt die Contribution Rate ab.

Measurements, Analysis, and Modeling of BitTorrent-like Systems (IMC 2005) L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang

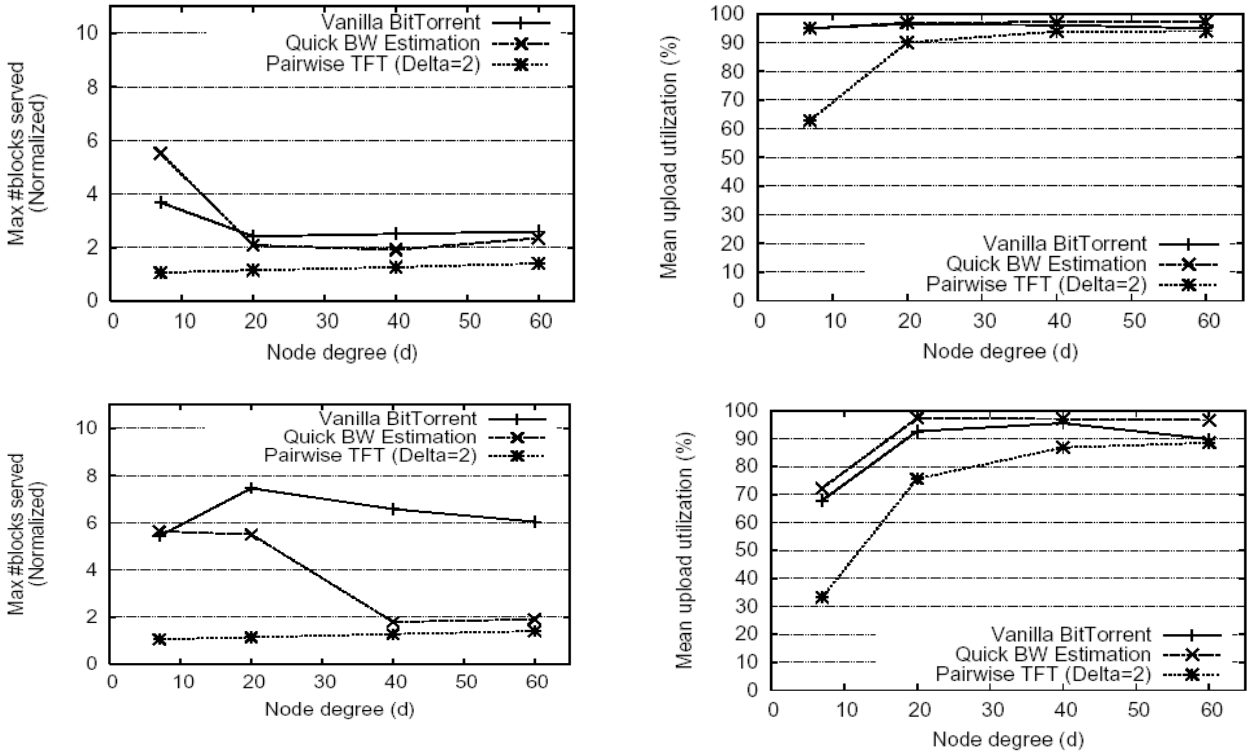
BitTorrent Ergebnisse



Uplink wird nahezu 100% ausgenutzt

Analyzing and Improving BitTorrent Performance (INFOCOM 2006)
A. Bharambe, C. Herley and V Padmanabhan

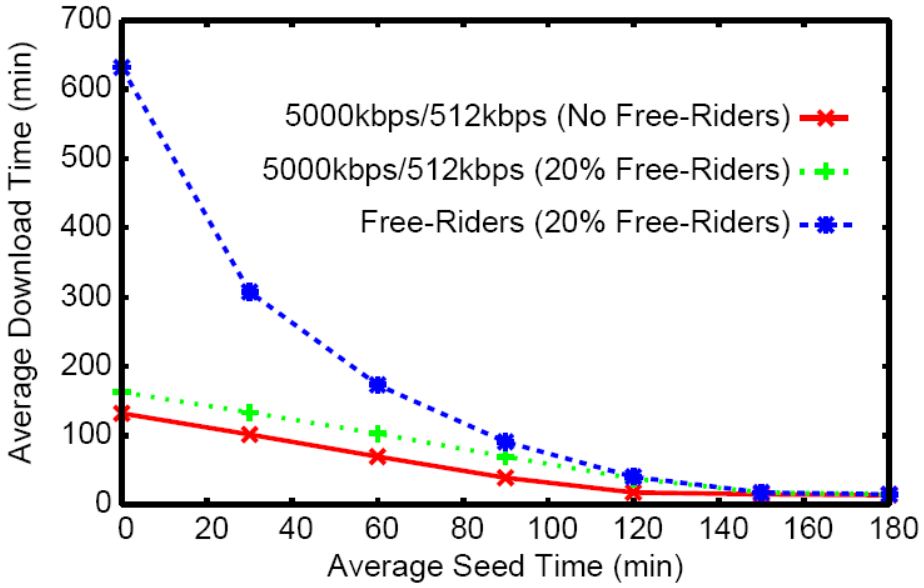
BitTorrent Ergebnisse

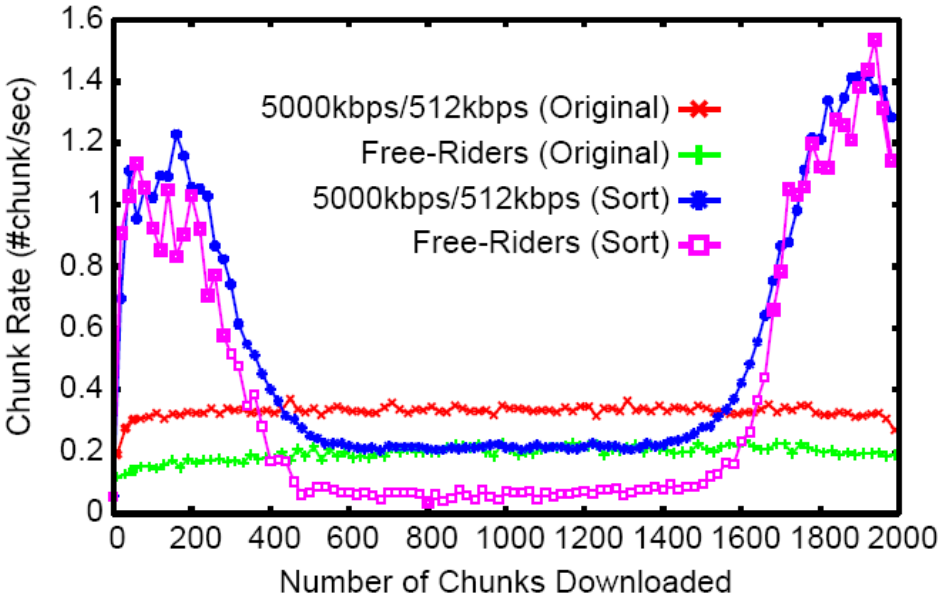


Tracker Policies, die nach Bandbreite sortieren, schlage alle anderen Policies.

Analyzing and Improving BitTorrent Performance (INFOCOM 2006)
A. Bharambe, C. Herley and V Padmanabhan

Improving BitTorrent





- Free Riders depend more on the seeds than benevolent peers.
- All peers have difficulties downloading the first chunks or the last chunks of a file. (Assume strict order.)
- Idea: Seeders serve preferably the files‘ first and last chunks.

Source: Chow et al. Improving BitTorrent: A Simple Approach, IPTPS 2008

Bekannte Clients der verschiedenen Netze

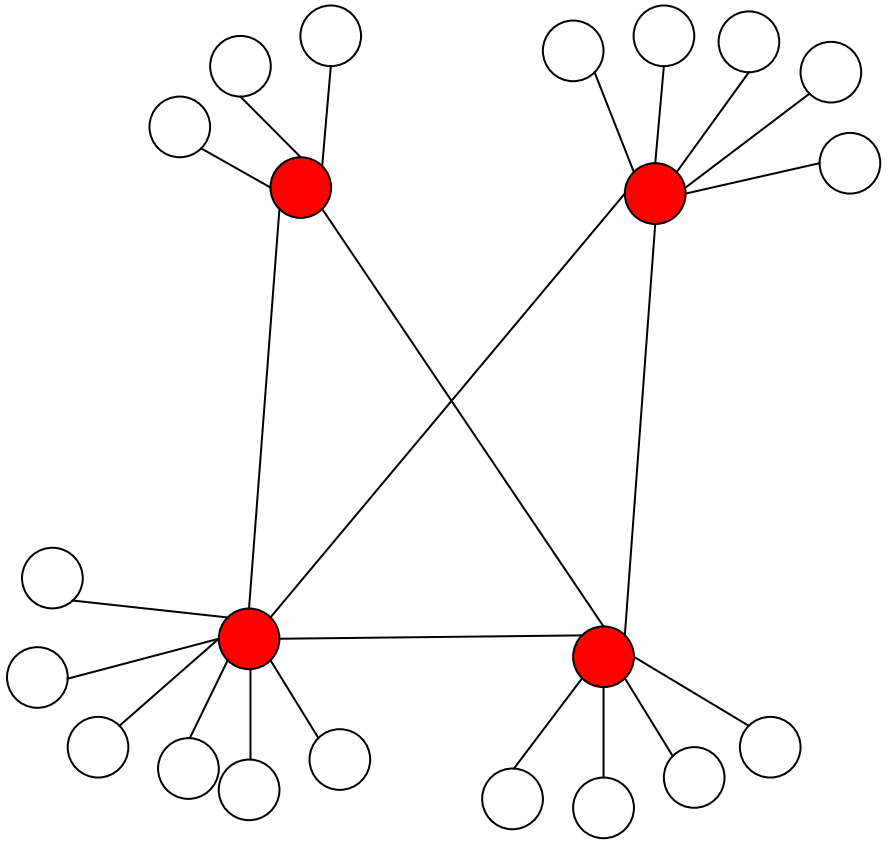
- FastTrack
  - KaZaA — offizieller Client – oft spywareverseucht
  - KaZaA Lite — gepatchter Client, nuetzt Netz aus
  - iMesh
  - MLDonkey — vorzeigeproject fuer Objective CAML
- eMule
  - eMule — meistgenutztester Client
  - MLDonkey
- BitTorrent
  - Azureus — DER BitTorrent Client in Java
  - BitComet — Client der das Netz oft ausnuetzte indem er z.B. falsche Daten an Tracker meldete

Über KaZaA

- Kommerzielles P2P System von Sharman Networks Inc.
- KaZaA war um 2002 das verbreitetste P2P Netz überhaupt
- Traffic großer Teil des Internettrafics
  - Univ. of Washington: 37% des TCP Traffics, mehr als 2x der HTTP Traffic.
  - 76% des US P2P Traffics
- Verkehr wurde verschlüsselt übertragen
- Darunterliegendes P2P-Netz: FastTrack
  - Netz besteht aus normalen Nodes und Supernodes

Supernode Architektur

- Supernodes sind Rechner mit mehr Bandbreite/Rechenzeit als andere Rechner
  - Keine Serverarchitektur
- Jeder normale Node hängt an einem Supernode, Supernodes haben auch Verbindungen untereinander
- Supernodes pflegen Liste der geshareten Dateien der angeschlossenen Nodes
- Suchanfragen gehen an Supernode, Supernode kann Anfrage an anderen SN weiterleiten



KaZaAs Niedergang

- Ca. Mitte 2004 wurde KaZaA von anderen Netzen überholt
- Probleme von KaZaA:
  - Hashing Algorithmus (UUHash) sehr schnell, aber nicht kollisionsresistent
  - Viele Korrupte Dateien im Netz (Pollution attacks)
  - Verschiedene Clients haben das Netz „missbraucht“: KaZaA lite hat sich bei suchen Nacheinander mit vielen Supernodes verbunden um mehr Suchergebnisse zu finden. Schlecht für Netzstruktur

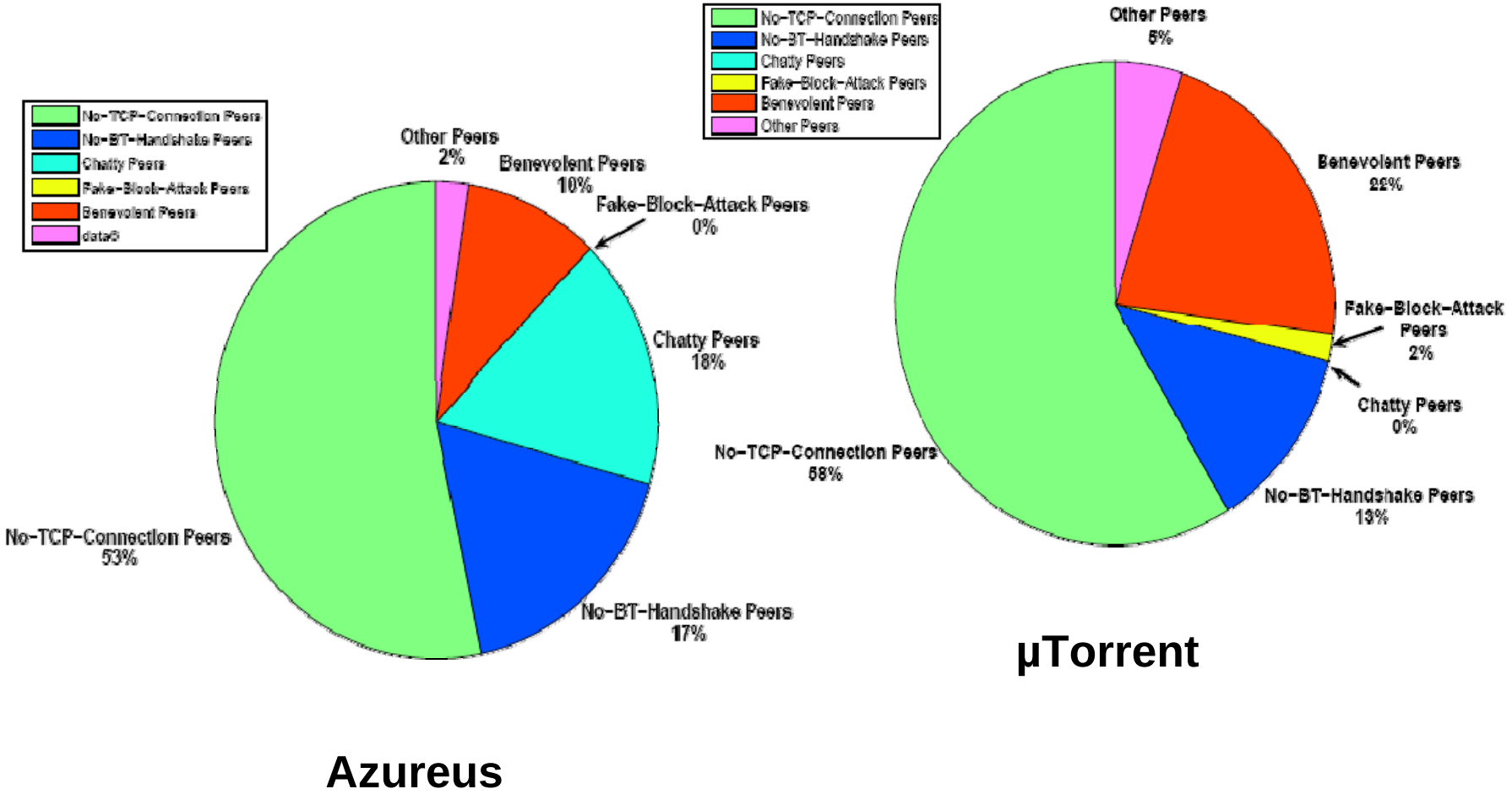
Attacks in P2P Systems (1)

- Poisoning attacks
  - Provide files whose contents are different from the description.
  - Often used by copyright owners to poison illegally sharing in popular file sharing platforms.
  - Forces the users to spend more downloading effort (for the poisoned content).
  - Challenge: Hard to detect automatically ($\rightarrow$ Is this really Madonna singing?)
- Pollution attacks
  - Insert "bad" chunks into an otherwise valid file on the network
  - Variant of poisoning: less effort for polluter (only insert minor defects, but spoil entire file)
  - Only possible in (badly engineered) chunk based systems, i.e. typically easy to detect automatically. Wasted effort limited to polluted chunk.
- Denial of service attacks
  - Exploit badly engineered or badly implemented protocol to bring down the service.
  - Attacks the service by making overly many valid requests to the system.
  - Distributed DoS attacks are hard to fight.

Attacks in P2P Systems (2)

- Besides the described attacks which can be used to keep a P2P system from working properly, copyright holders can also team with network providers:
  - Filter P2P ports (or protocols) to prevent users from using peer-to-peer file-sharing
  - Easy to implement, but easy to work around, too. → Arms race!
  - Often, providers (including universities) do not distinguish between illegal and legal uses of P2P systems (which would be hard indeed); they just block everything.
- Finally, as long as the P2P system does not provide anonymity, users can be prosecuted “out-of-band”, i.e. by the respective local legal system.
- However, be aware that attacks via P2P networks can be harmful for valid, legal users of P2P systems, too:
  - Downloaded or forwarded files may be infected with viruses or malware.
  - Peer-to-peer network software is itself malware, e.g. contains spyware

Attacks in BitTorrent



Source: Dhungel et al. A Measurement Study of Attacks on BitTorrent Leechers, IPTPS 2008

Free Riders in P2P Systems

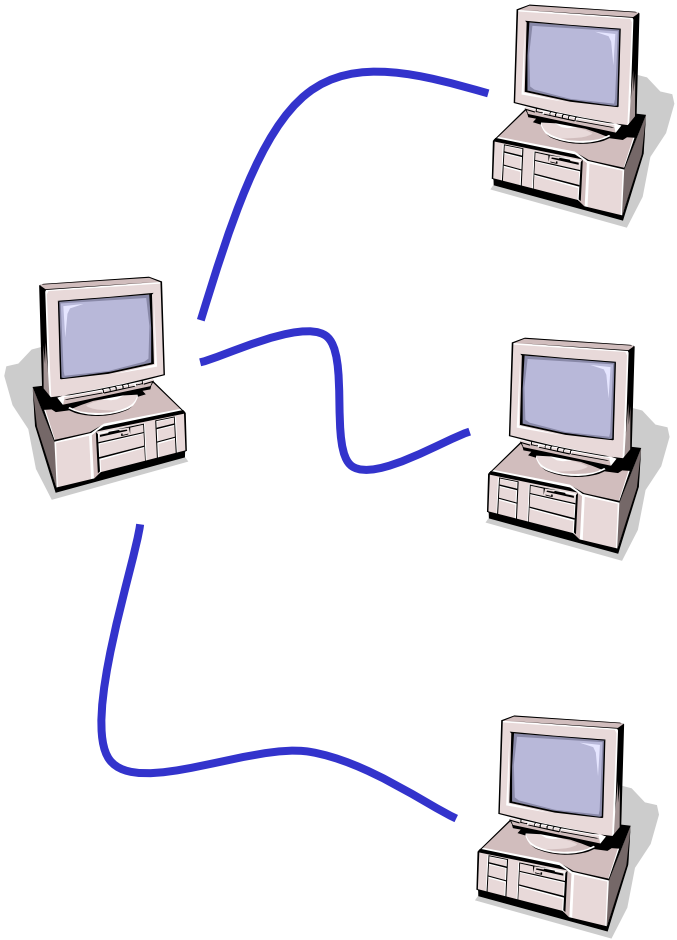
- P2P systems rely on peers providing resources to the system.
- Free-Riders make use of the network without contributing resources to it.
- Possible solution:
  - Force users to contribute resources that are equivalent to the consumed resources.
  - But: Chicken-and-Egg Problem; system must give initial credit to each peer, so that it can join the service.
  - But: Sybil attack; peers always pretend to be new comers.
- Fighting free riders in P2P systems is hard without strong identities.
  - Try to limit negative impact of free riders.
  - Hope for enough altruistic peers to provide their resources.

The Sybil Attack

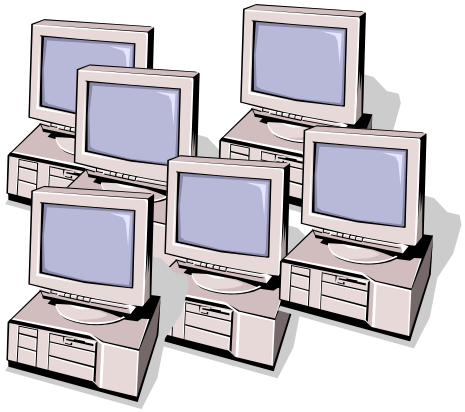
- "Sybil" by Flora Rheta Schreiber (1973) describes the life of Sybil Dorsett, a woman with multiple personality disorder (cf. Wikipedia for details about the case).
- On the Internet, nobody knows your are a dog (P. Steiner, The New Yorker, 1993).
- In P2P networks, nobody knows how many personalities you have.
- Strong identities: Use an external mechanism to prevent a natural person from obtaining an arbitrary number of identities:
  - Central authority checks persons' identity an issues cryptographic certificate
  - Central authority requires initial (or monthly) fee
  - Check for IP address, but what about dynamic IP space and NAT!?!

"On the Internet, nobody knows you're a dog."

Eclipse Attack



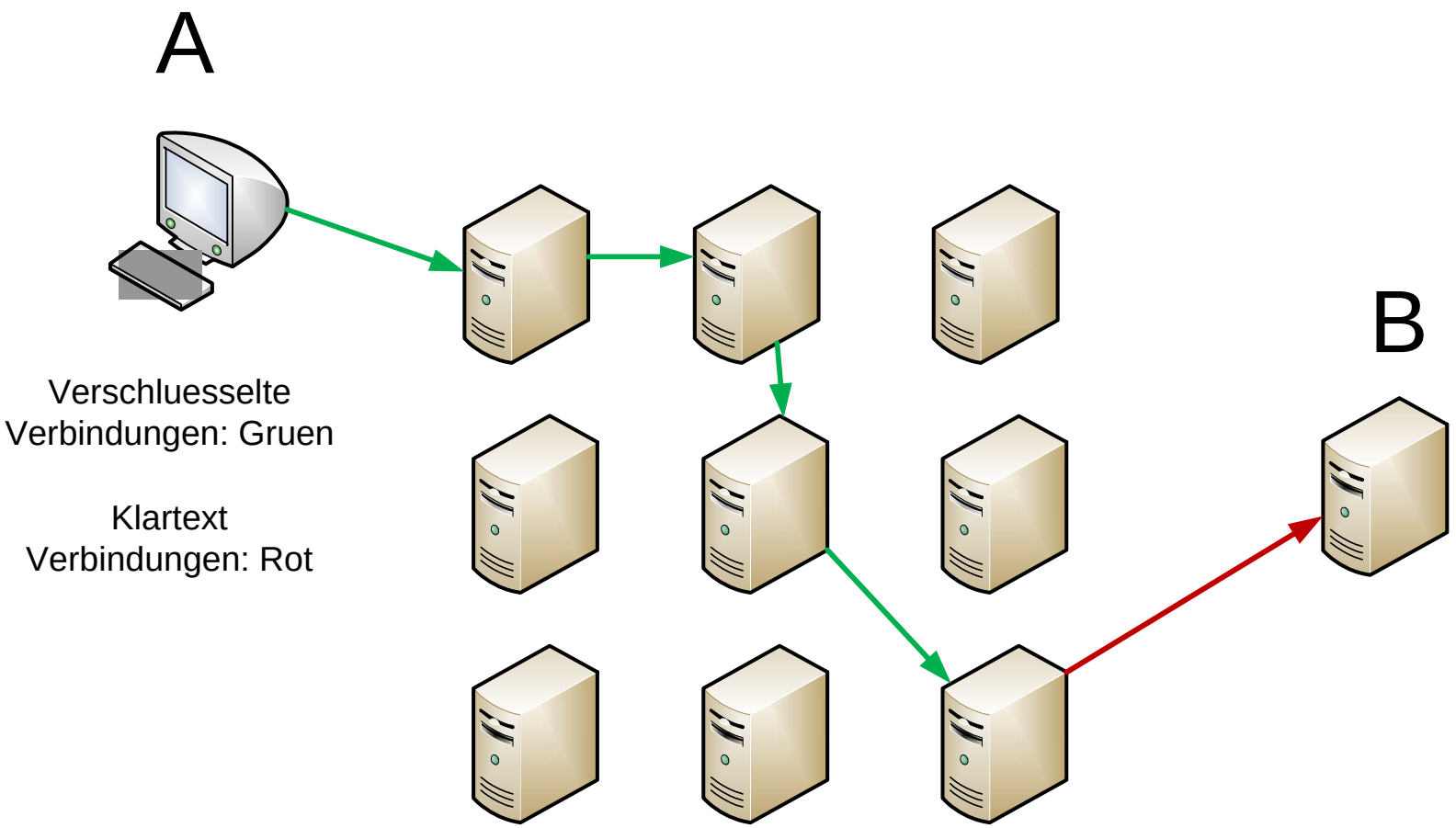
A set of malicious nodes can block a peer from the overlay or otherwise manipulate a peer, if it controls all the peer‘s connections.



Onion Routing (1)

- Name von engl. Onion (Zwiebel)
- Nachricht wird mehrfach verschlüsselt und dann über eine Serie von Onion-Routern geschickt
- Jeder der Router kann eine Verschluesslungsschicht entfernen, der letzte Router schickt die Klartextnachricht zum Ziel
- Jeder Router kennt nur die Adressen seiner direkten Nachbarn
  - Der Router der die Nachricht entschlüsselt und weiterleitet kennt also nie die wirkliche Nachrichtenquelle
- Antwortnachrichten werden in exakt umgekehrter Reihenfolge wieder verschlüsselt

Onion Routing (2)



TOR – The Onion Router

- Tor ist eine C-Implementierung des Onion Routings
- Jeder Benutzer lässt einen lokalen SOCKS-Proxyserver auf seinem Rechner laufen
- Über diesen Proxy werden dann die eigentlichen Verbindungen aufgebaut
- Der Tor Node auf dem lokalem Rechner sucht sich einen Weg durch das Netz und verschlüsselt die Nachricht
- Liste aller im Moment aktiven Tor Nodes wird über Zentralen Server verteilt

Probleme von TOR

- In letzter Zeit ist ein auffälliges Wachstum der Exit-Nodes in China und den USA zu beobachten
- Anscheinend wird der Traffic einiger Exit-Nodes überwacht
  - In Klartext übertragene Passwörter und Usernamen können so ohne Probleme abgegriffen werden
- In Deutschland stand einige Zeit ein Exit-Node, der bei SSL-Verbindungen sein eigenes Zertifikat statt dem Serverzertifikat angeboten hat
  - Wenn man nicht auf die Browserwarnungen geachtet hat, konnte dieser Node mittels einer „Man-in-the-Middle“ Attacke trotz Verschlüsslung alle Informationen abhören
- Bei falscher Konfiguration werden DNS-Anfragen immer noch über die lokale Internetverbindung weitergeleitet
  - Provider kann u.U. trotzdem Kontaktadressen nachvollziehen

Summary

- BitTorrent is a widely used P2P filesharing application
  - Originally based on centralized trackers
  - Tit-for-tat algorithm tries to ensure fairness
  - Hard to ban free riders, but seeders can try to preferably support benevolent peers
- Kazaa has been a widely used P2P filesharing application 2002 – 2004
  - Super peer system like later Gnutella systems
  - Law suits and dubious spyware made it unpopular in the end
  - Kazaa authors continued and created Skype
- All P2P overlays are prone to attacks, e.g. poisoning or Sybil attacks
- Tor is a rarely used implementation of onion routing

Questions?

Thomas Fuhrmann

Department of Informatics
Self-Organizing Systems Group
c/o I8 Network Architectures and Services
Technical University Munich, Germany

fuhrmann@net.in.tum.de