Hálózatba kapcsolt erőforrás platformok és alkalmazásaik

Simon Csaba
TMIT
2017

Anonimitás – elosztott módon

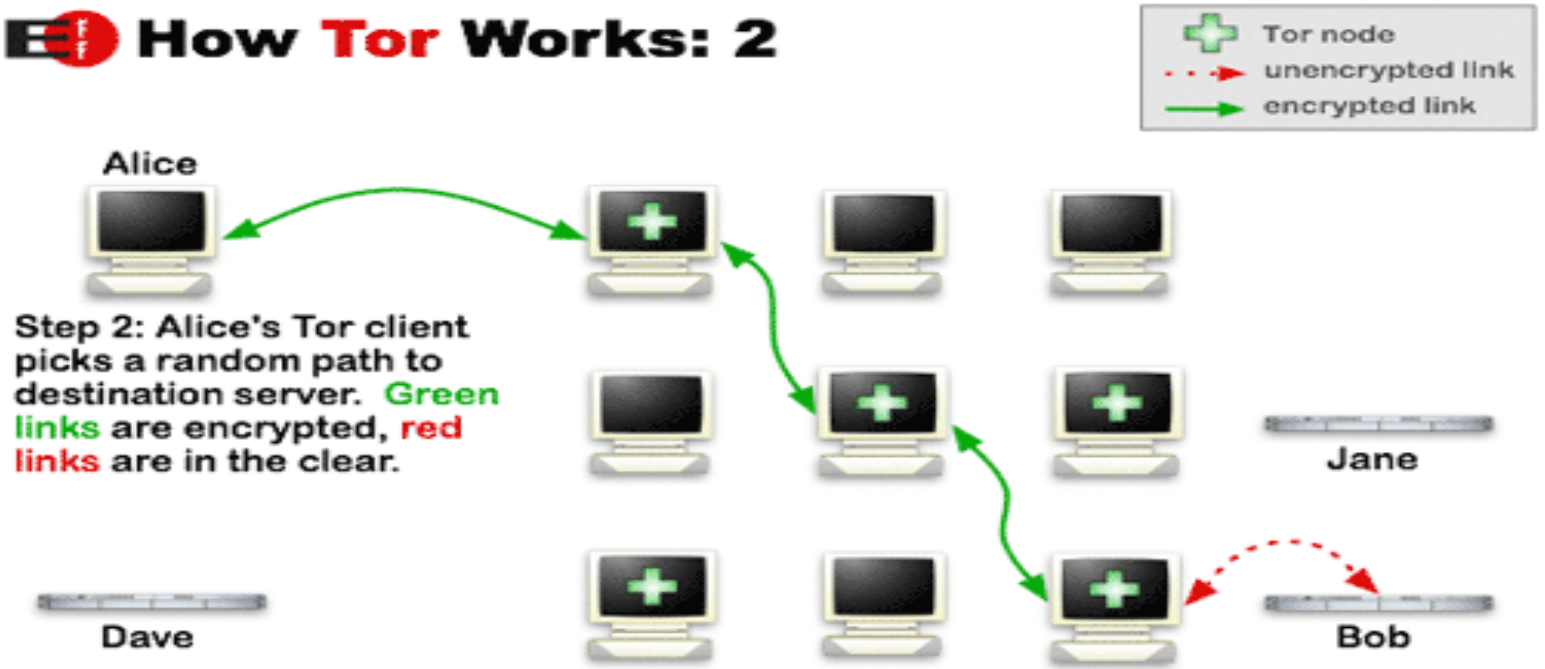
Anonimitás

- Első lépés
  - Követők, lehallgatók összezavarása
  - Nem tudják, hogy ki is tölt le
  - P2P módon együttműködnek a felhasználók
    - „relay”

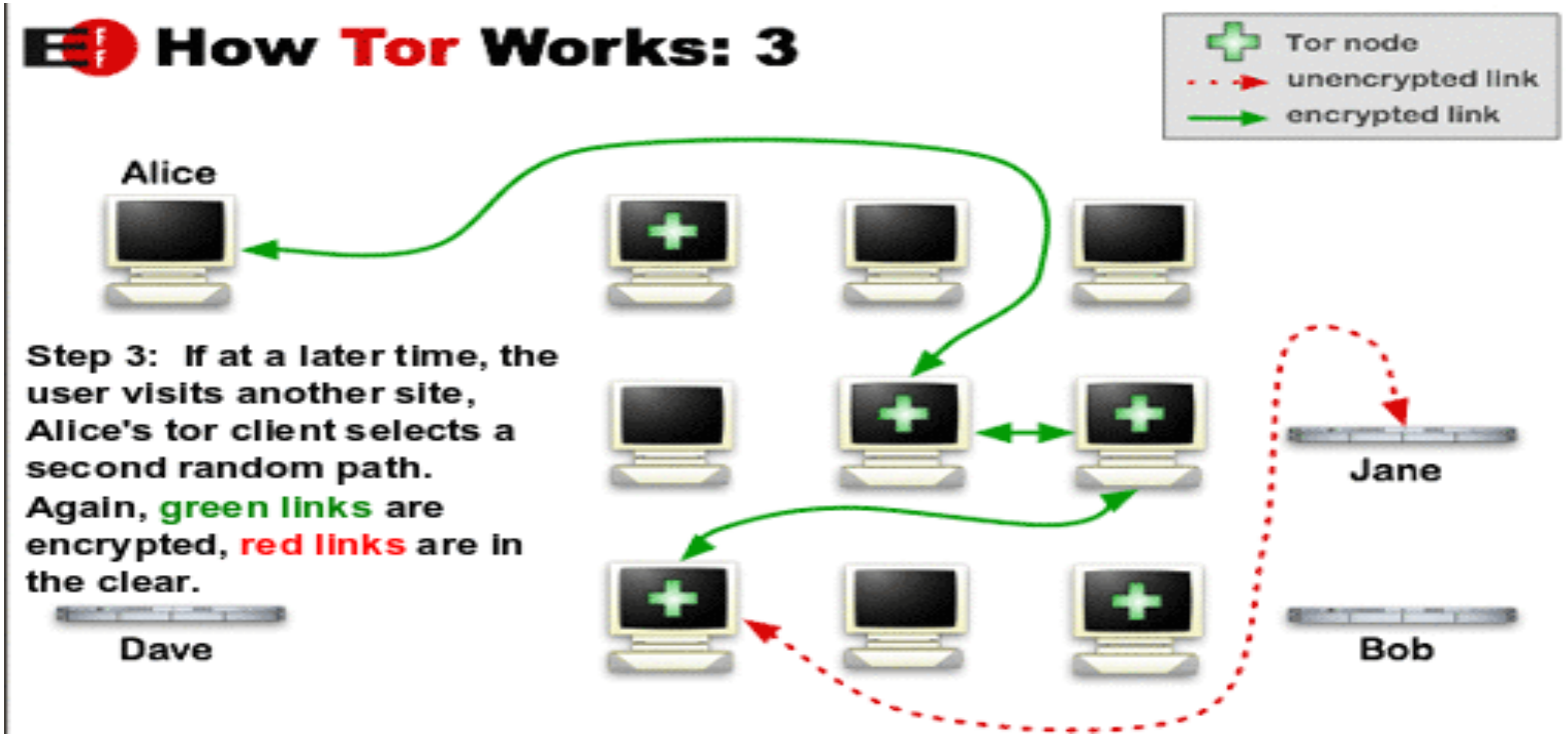
Tor


How Tor Works: 1
Tor node
unencrypted link
encrypted link
Alice
Step 1: Alice's Tor client obtains a list of Tor nodes from a directory server.
Jane
Dave
Bob

Tor


How Tor Works: 2
Tor node
unencrypted link
encrypted link
Alice
Step 2: Alice's Tor client picks a random path to destination server. Green links are encrypted, red links are in the clear.
Jane
Dave
Bob

Tor


How Tor Works: 3
Tor node
unencrypted link
encrypted link
Alice
Step 3: If at a later time, the user visits another site, Alice's tor client selects a second random path. Again, green links are encrypted, red links are in the clear.
Dave
Jane
Bob

Anonimitás – elosztott módon

Anonimitás

- Második lépés
  - Követők, lehallgatók összezavarása
  - P2P rendszerben történik a routolás is
  - Adatot is szétosztjuk
  - ELOSZTOTT ADATBÁZIS
    - Igazából egy elosztott cache
    - Keresés

Freenet

- Ian Clarke, végzős hallgató
- University of Edinburgh

- Teljesen elosztott, egyenrangú hálózat

- A peer-ek tárolófelületet és sávszélességet „ajánlanak” fel a rendszernek
  - Nem tudják a rendszer mit tárol ott
  - Minden adat kódolva van

- Freenet – *„földrajzilag elosztott nagyméretű virtuális merevlemez, névtelen (anonymous) hozzáféréssel”*

Freenet

- Biztosítja a „szerzők” és az „olvasók” anonimitását
  - Nem lehet megállapítani az adatok forrását és célállomását
  - Az adatokat *kulcsok* azonosítják
  - A peer-ek nem tudják mit tárol a rendszer a gépükön
    - Nem (vagy nehezen) vonhatóak felelősségre
- A keresést abba az irányba továbbítja, ahol a legvalószínűbb a találat
  - Nincs központi szerver (Napster)
  - Nincs elárasztás (Gnutella)
- A fájlok azonosítása tárolási helyüktől független

Freenet

- Az adatok dinamikus elosztása a rendszerben

- A hálózati topológia (gráfstruktúra) folyamatosan változik
  - Új kapcsolatok jönnek létre a peer-ek között
  - Az állományok vándorolnak a rendszerben
  - Dinamikus, adaptív útválasztás

Freenet fejléc

- UniqueID (64 bit)
  - Azonosítja az üzenetet
  - A hurkok elkerülésére szolgál
- Hops To Live
  - Még hányszor lehet továbbküldeni a csomagot
- Depth
  - Hányat ugrott már a küldés óta
- Source
  - Az üzenet forrását azonosítja

Üzenet típusok

- HandshakeRequest
  - Kapcsolatot kezdeményez egy peer-el
- HandshakeReply
  - Válasz a HandshakeRequest-re
- DataRequest
  - A megadott kulcsnak megfelelő adatot kéri
- DataReply
  - Válasz a DataRequest-re
  - A közbeeső peer-ek cache-ben tárolják az adatot
  - A következő hasonló kérésre már tudnak közvetlenül válaszolni
- RequestFailed
  - Sikertelen keresés esetén

Üzenet típusok

- InsertRequest
  - Megegyezik a DataRequest-el
  - Azt ellenőrzi, hogy a megadott kulcsnak megfelelő fájl megtálalható-e a rendszerben
  - Ha igen, egy DataReply érkezik
  - Ha nem, egy InsertReply üzenet

- DataInsert
  - Kérés a megadott adat és kulcs tárolására a rendszerben
  - Az InsertRequest útvonalán halad
    - Ott tárolja ahol majd keresni fogják
  - A közbeeső peer-ek cache-ben tárolják az adatot

Data Store

- Minden peer-nek tudnia kell…
  - Merre továbbítani egy kérést
  - Merre továbbítani egy választ
  - Meddig tárolni egy adott dokumentumot

Data Store

Key	Data	Address
8e47683isdd0932uje89	ZT38hwe01h02hdhgdzu	tcp/125.45.12.56:6474
456r5wero04d903iksd0	Rhweui12340jhd091230	tcp/67.12.4.65:4711
f3682jkjdn9ndaqmmxia	eqwe1089341ih0zuhge3	tcp/127.156.78.20:8811
wen09hjfdh03uhn4218	erwq038382hjh3728ee7	tcp/78.6.6.7:2544
712345jb89b8nbopledh		tcp/40.56.123.234:1111
d0ui43203803ujoejqhh		tcp/128.121.89.12:9991

Lista tetején – „friss”, gyakran kért fájlok

- **Kulcs, adat, származási cím**

Lista alján – régi, ritkán kért fájlok

- **Kulcs, elérhetőségi cím**

Ha egy kulcs lefele mozog a listán, egy idő után az adatok törlődnek

Keresés

- A peer-ek által tárolt kulcsok alapján
  - Korlátozott számú ugrás, de nagyobb mint a Gnutella esetében
    - Tipikusan 500
  - Az üzenetek azonosítója alapján szűri a hurkokat

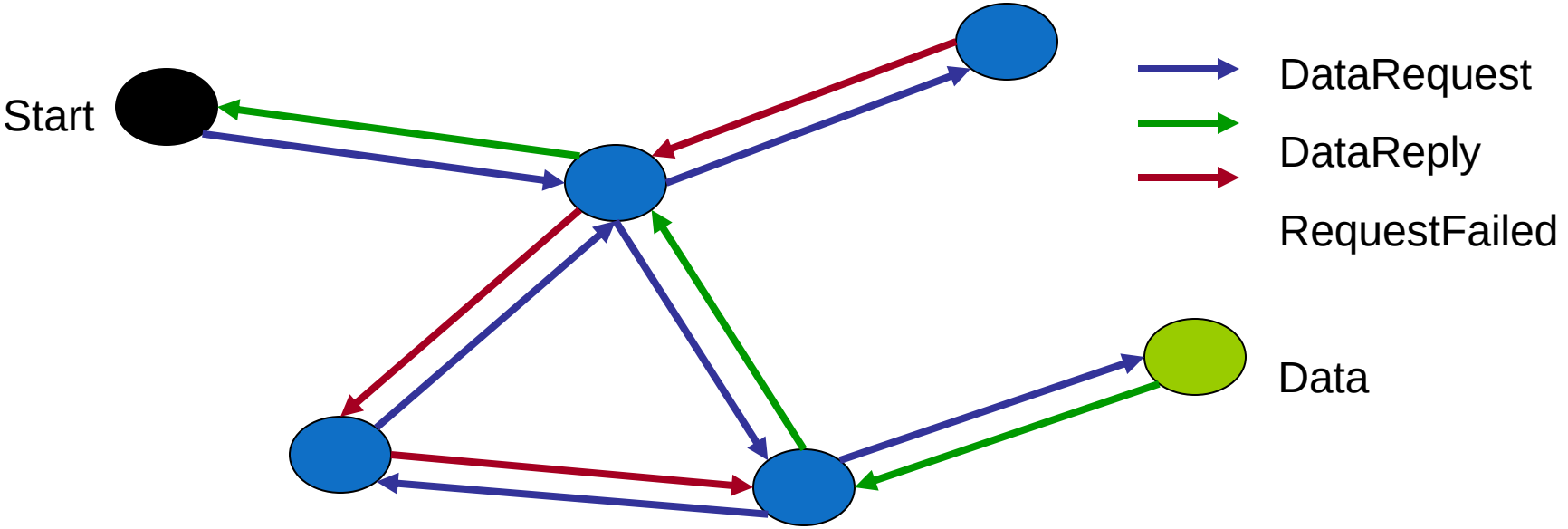
- Egy kérés érkezésekor
  - Ha már kezelte a kérést (hurok), visszaválaszol hogy a küldő próbálja másfelé továbbítani („next-best choice”)
  - Ha rendelkezik az állománnyal, válaszol
  - Ha nem, továbbküldi a keresett kulcshoz legközelebb álló kulcsnak megfelelő cím felé

Keresés (II)

- Ha egy válasz érkezik
  - A kulcs és az adat bekerül a cache-be
  - A legrégebbi kulcs kikerül a data store-ból

- Találat esetén az állomány küldése nem pont-pont közötti
  - A keresés útvonalán terjed vissza
  - Segíti a cache működését
  - Növeli az anonimitást

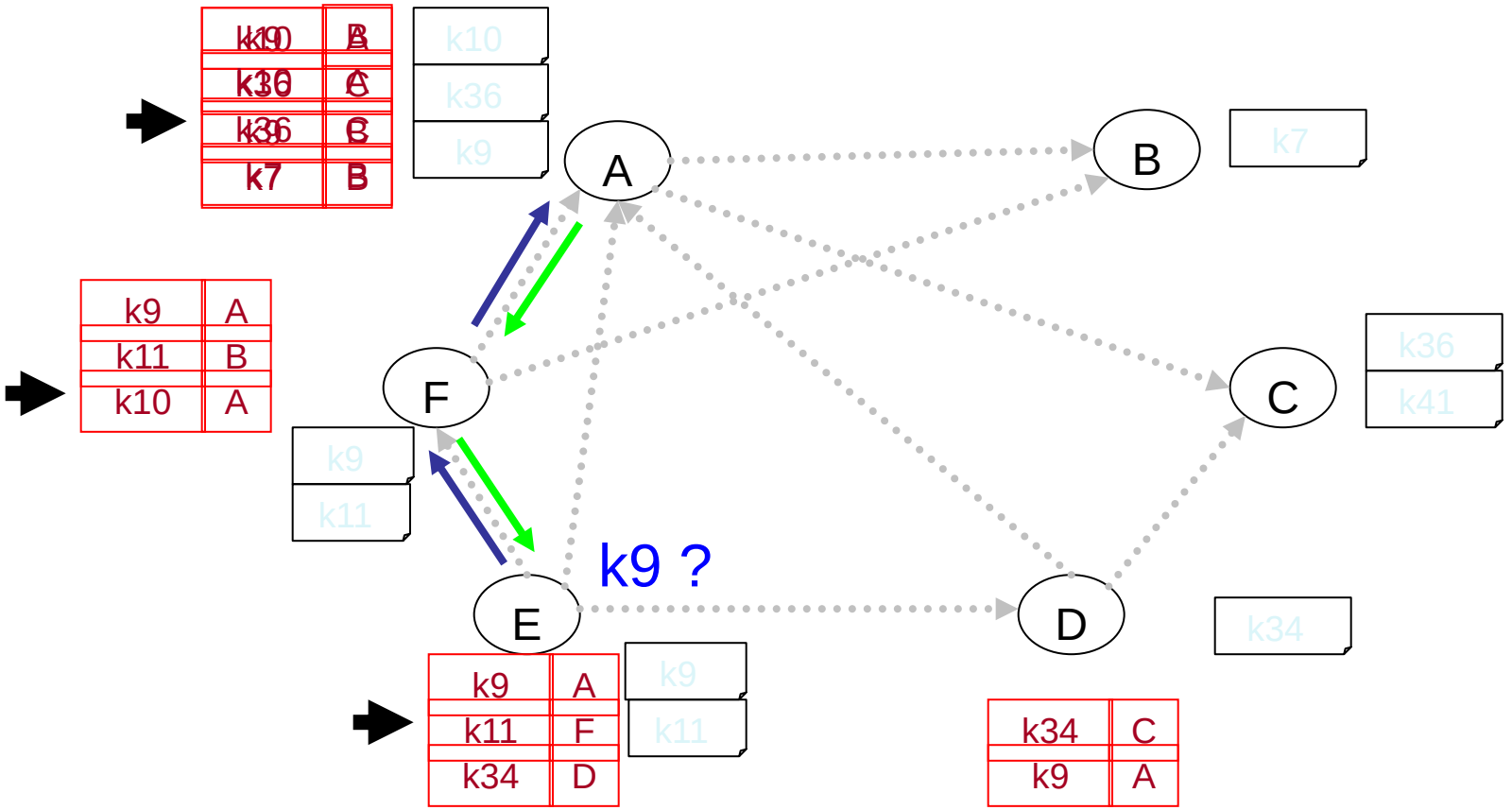
Keresés - példa



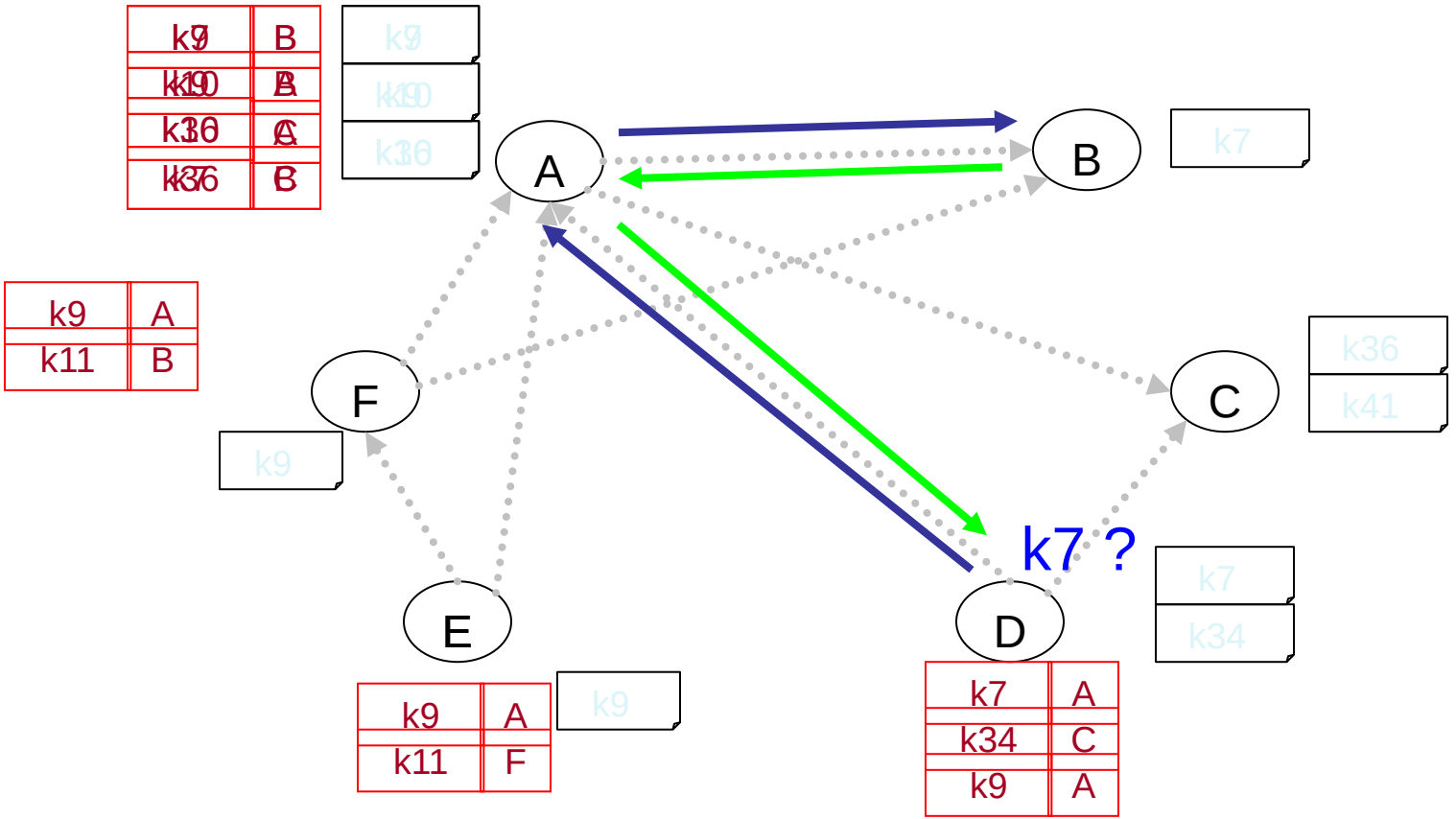
Keresés

- Az elején a Data Store üres
  - A peer véletlenszerűen választja ki merre küldje a keresési, illetve adatbeviteli kéréseit
  - Az adatok eloszlása is véletlenszerű lesz
- A keresés (útválasztás) minősége idővel javul
  - Lavina effektus
  - Egy tárolt fájl alapján a peer bekerül más peer-ek adatbázisába, a megfelelő kulccsal
  - Hasonló kulcsok iránti kérések elkezdenek érkezni
  - A válaszok bekerülnek a cache-be
  - Egyre több hasonló kulcsú állományt tárol
- A peer nem döntheti el milyen kulcstartományra „specializálódik”
  - A többi peer által tárolt kulcsoktól függ
  - Növeli a rendszer „ellenőrizhetetlenségét”

Freenet keresés - példa



Freenet keresés - példa



Lavina effektus példa

Animáció, klikkelj a „vonalkódra”

Keresések eloszlása

- Slashdot effektus (http://slashdot.org/)
  - Népszerű számítástechnikai hírportál
    - „News for nerds, stuff that matters”
  - Ha egy érdekes hír jelenik meg rajta, mindenki ráklikkel a linkre
  - A hirtelen terhelés megbéníthatja a kisebb kapacitású gépeket
- A Freenet elkerüli a Slashdot effektust
  - A cache-elés miatt a keresések eloszlanak
  - A források nem kerülnek túl nagy terhelés alá
- A kulcsok hash-ek
  - Szemantikailag egymáshoz közel álló tartalmak teljesen különböző kulcsokat kaphatnak
  - Egy „hot topic”-hoz tartozó különböző állományok eloszlanak a rendszerben

Hash függvény

- Nagy értelmezési tartományt képez le egy „szűk” értékkészletre
  - Változó hosszúságú x paramétert képez le fix hosszúságú h = H(x) értékre
- Kriptografikus hash függvények
  - Ha adott x, H(x) relatív egyszerűen kiszámítható
  - H(x) egyirányú
    - Ha adott h hash érték, túlzottan számításigényes olyan x-et találni, amire H(x) = h
  - H(x) ütközésmentes
    - H „gyengén ütközésmentes” ha egy adott x-re, túlzottan számításigényes egy olyan y megtalálása ($x \neq y$), melyre H(x) = H(y)
    - H „erősen ütközésmentes” ha túlzottan számításigényes bármilyen olyan x és y értékeket találni ($x \neq y$), melyre H(x) = H(y)

Hash függvény

- Népszerű algoritmusok
  - MD5 – (Message Digest Algorithm 5)
    - Ronald Rivest (MIT), 1991
    - http://www.ietf.org/rfc/rfc1321.txt
  - SHA-1 (Secure Hash Algorithm)
    - http://www.itl.nist.gov/fipspubs/fip180-1.htm

Kulcsok

- Minden állományt egy kulcs azonosít
- A kulcs egy globális azonosító része
  - Uniform Resource Identifiers (URIs): freenet:*keytype@data*
- Több fajta kulcs létezik
  - Keyword Signed Key (KSK)
  - Signature Verification Key (SVK)
  - SVK Subspace Key (SSK)
  - Content Hash Key (CHK)

Keresés

- A kereséshez tudni kell az adott kulcsot
- A publikáló félnek ezt nyílvánosan meg kell hírdetni
  - Anonymous hirdetési lehetőség Freenet portálokon
    - Rövid leírás, kulcs
  - Freenet levelező listákon
  - Freenet IRC csatorna
    - irc.freenode.net #freenet
  - Privát e-mail, web
  - Graffiti, hőlégballonon, repülő által húzott reklámszalagon

Anonimitás

- A peer-ek véletlenszerűen „hazudhatnak” a kérésekkel kapcsolatban
  - A Depth és a Hop-To-Live értékek változtathatóak
- Lehetetlen kideríteni kitől kapod meg a dokumentumot
  - A forrás címet bármely csomópont az útvonalon átírhatja a sajátjára
- Lehetetlen kideríteni ki publikált először a rendszerben egy adott fájlt
- Igyekeznek a réseket is betömni
  - „the Javascript exploit mentioned would not have worked on Freenet because Freenet removes Javascript by default. ”
- Nem biztosít igazi anonimitást
  - A nem triviális támadások ellen valószínűleg nem képes védekezni
    - Forgalomanalízis, nagyszámú csomópont kompromittálása

Előnyök

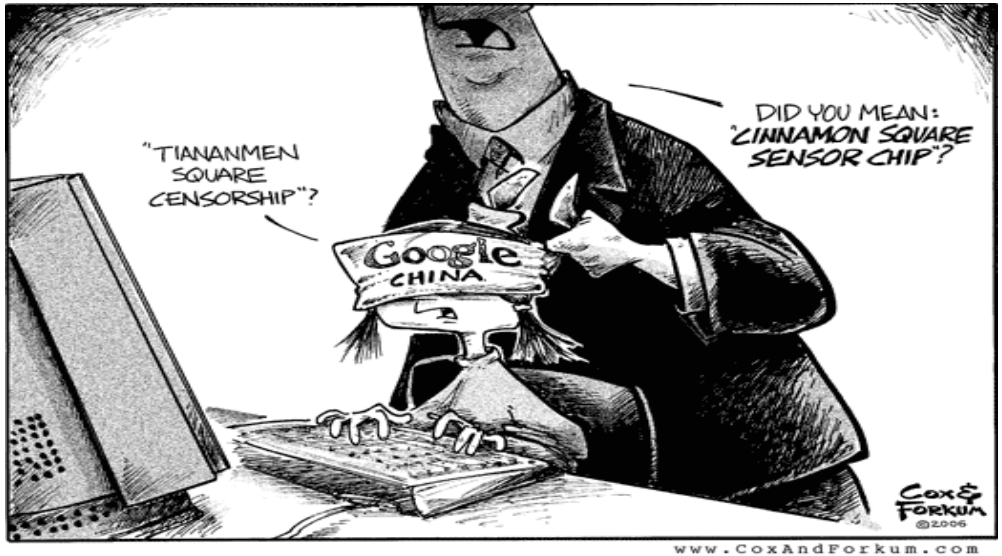
- Teljesen elosztott rendszer
- Nagy hibatűrés
- Robusztus, skálázható
- Másolatok automatikus generálása
- Jól alkalmazkodik a felhasználók dinamizmusához
- Adaptív, hatékony útválasztás
- Anonim szerzők és olvasók
- Freeriding nem lehetséges

Hátrányok

- Nem garantálható a keresések ideje
- Nem garantálható a keresések sikere
- Ismeretlen topológia
  - A kereső algoritmusok nem tudják ezt kihasználni
- Nem biztosítja a tartós tárolást
  - Egy ma publikált fájl holnapra lehet hogy „kihal” a rendszerből
- Nagyméretű állományoknál nem előnyös egy hosszú útvonal mentén küldeni a választ
  - Nagy a hibalehetőség
  - „Elpocsékolt” sávszélesség

Anonimitás vs. hatékonyság

- A Freenet fő célja a cenzúra elkerülése és az anonimitás,
- nem a hatékonyság
  - Főként kisméretű fájlok, szöveges dokumentumok tárolására és terjesztésére kiváló
  - Freenet China
    http://freenet-china.org/



Irodalom

- I. Clarke, O. Sandberg, B. Wiley, T. W. Hong, , "*Freenet: A distributed anonymous information storage and retrieval system*", in Workshop on Design Issues in Anonymity and Unobservability, Berkeley, CA, July 2000.
  - http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.10.4919
  - 2000 legtöbbször hivatkozott cikke a Citeseer-ben
    - Jelenleg 773 hivatkozás a Citeseer-ben, 2236 hivatkozás a Google Scholar-ban

- The Free Network Project
  - http://www.freenetproject.org/
  - http://freenetproject.org/papers.html

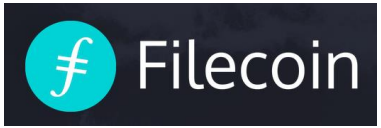
Blockchain alapú P2P megoldások

IPFS

- Interplanetary File System
  - combines the concept of linking the distributed file sharing to a file system
  - Hashing individual files

- Ethereum
  - a single shared computer that is run by the network of users and on which resources are parceled out and paid for by *Ether*
  - *finance, the internet-of-things, farm-to-table produce, electricity sourcing and pricing, and sports betting*

- Filecoin
  - incentive for storage, on top of IPFS
  - distributed electronic currency similar to Bitcoin
  - proof-of-retrievability component, which re-quires nodes to prove they store a particular file



Zeronet

- Dynamic Web
- P2P
- Integrated BitCoin

- ZeroNet and IPFS: uncensorable auto-scaling BitTorrent powered websites
- ZeroNet is created for fast, dynamic websites, IPFS is more like a storage solution

http://bitcoinist.com/interview-developer-zeronet/

SWARM

SWARM

SERVERLESS HOSTING INCENTIVISED PEER-TO-PEER STORAGE AND CONTENT DISTRIBUTION

- Distributed **storage** platform and content distribution service
  - a native base layer service of the ethereum web 3 stack
- Features
  - DDOS-resistant, zero-downtime, fault-tolerant, censorship-resistant
  - self-sustaining due to a built-in incentive system
    - uses peer to peer accounting
    - allows trading resources for payment

http://www.epointsystem.org/~nagydani/homepage

Összefoglalás

- P2P > fájlcserélés
  - Lehetőség van a résztvevők identitásának elrejtésére
  - DHT alapú
- Blockchain alapú új javaslatok (2014/15-től kezdődően)
  - Distributed storage
  - Cenzorship
  - Incentive
  - Dynamic

Egyéb fóliák

CAP

Legal Download in EU

- http://index.hu/kultur/2013/10/30/uhd/

CAP tétel

- *CAP = Consistency, Availability, Partition Tolerance*
  - Elosztott rendszerekben
  - Egyszerre nem lehet mind a három célt elérni

- http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

CAP tulajdonságok

- CAP - mozaikszó
  - Konzisztencia (consistency)
  - Rendelkezésre állás (availability)
  - Partíció tolerancia (partition tolerance)
- Elosztott rendszerekben egyidejűleg legfeljebb két fenti tulajdonság garantálható teljesen

- A. Fox, E. Brewer, „Harvest, yield, and scalable tolerant systems”. *Workshop on Hot Topics in Operating Systems, 1999.*

http://shark-collection.googlecode.com/svn/docs/Technique/Architect/%5B8%5DHarvest,%20Yield,%20and%20Scalable%20Tolerant%20Systems.pdf

CAP tétel

- „C” - Konzisztencia: bármely időpontban, bármely csomóponttól lekérdezhetjük bármelyik „atomi” adatot ugyanaz az érték
- „A” - Rendelkezésre állás: minden működő csomóponthoz érkező kérésre tud válaszolni
- „P” - Partíció tolerancia: adott kérésre hálózati partíció esetén is végre lehet hajtani az írás vagy olvasás műveletet
- **CAP tétel**: elosztott rendszerben a hálózat partíciója esetén a rendszer műveletei nem lesznek atomiak és/vagy az adategységei elérhetetlenek lesznek

CAP elvárások a gyakorlatban

- CAP tétel szerint nem valósítható meg egy működő elosztott adatbázis
  - Ugyanis az nagyszámú, egymástól független elemből állna
  - Ez meg „behozza” a particionálás veszélyét a rendszerbe
- Ugyanakkor globális cégek működtetnek ilyen rendszereket. Hogyan?
  - A három feltétel egyikének lazításával
  - Pl: ha a formális konzisztencia követelménynél gyengébb elvárás megengedése
    - Amazon Dynamo - „A”, „P” tulajdonságokat garantálja
  - Pl. ha a késleltetés minimalizálását helyezik előtérbe
    - Yahoo PNUTS – „P” tulajdonságot minden esetben garantálja