Peer-to-Peer Systems and Security

Incentives

Christian Grothoff

Technische Universität München

April 13, 2013

Why Incentives?

Client-server:

- Server in control & trusted
- Faulty clients are dropped

$\Rightarrow$ clients forced to obey

$\Rightarrow$ focus on protocol spec

P2P:

- Users may deviate from protocol spec
- rational nodes:
  - maximize own utility
  - can try to be "tricky"
- irrational nodes:
  - altruistic
  - disruptive

Example

Gnutella — Ader & Huberman (2000):

- 70% of peers provide no files
- 1% of peers provide 37% of all files

Maze — Yang & Zhang & Li & Dai (2005) [2]:

- 22% of peers with one account are free-riders
- 77% of peers with eight or more accounts are free-riders

Reasons

Reasons for free-riding:

- Motivation (better performance, lower risk, ...)
- Opportunity (no authority, anonymity/pseudonymity, lack of group cohesiveness)

Reasons for cooperation:

- Inherent generosity (self-esteem, political motivations, ...)
- Monetary payment schemes
- Reciprocity-based schemes — Incentives

Incentives

Trust is no assurance that the other entity will cooperate.

Incentives are mechanisms to make a peer cooperate by giving it benefits from cooperation

Basics of Game Theory: The Prisoner's Dilemma

- two suspects are arrested
- asked to testify against each other
- If both testify, both serve 7 years
- If one testifies, only other serves 10 years
- If neither testifies, both serve 2 years

	A talks	A silent
B talks	A: 7, B: 7	A: -10, B: 0
B silent	A: 0, B: 10	A: -2, B: -2

Solution

- Best social strategy: no one testifies
- Nash equilibrium:
  - for **constant choice** of the other party, each player optimizes his benefit

Solution

- Best social strategy: no one testifies
- Nash equilibrium:
  - for **constant choice** of the other party, each player optimizes his benefit
  - if both talk, then there is a **Nash equilibrium**

A strategy is **dominant** if is always better than every other strategy.

The Prisoner's Dilemma of Filesharing

	U shares	U rejects
D downloads	U: -, D: ++	U: 0, D: 0

Note that "D" doesn't really have a choice here.

BitTorrent: reward uploader

If U is also downloading and the download for U becomes faster if it uploads, then:

	U shares	U rejects
D downloads	U: $+$, D: $++$	U: 0, D: 0

New rational strategy and Nash equilibrium: uploading!

The Game of Game Theory: Mechanism Design

- Define rules of the game
- Rules must be enforced:
  - central control
  - non-free, hacker-proof software
  - cryptography
- Design rules such that desired outcome occurs

$\Rightarrow$ rational behaviour $\equiv$ good behavior

Mechanism Design: Common Concerns

- calculating optimal strategy might be hard
- do all players have the required information to do so?
- does asymmetric knowledge create unfairness?
- what is “the best” rule system?
- what is best for society as a whole?

HashCash

Adam Back proposed *HashCash* as a solution to stop unsolicited mass E-mailing (also known as spam). Key idea:

- the sender pays per E-mail

HashCash

Adam Back proposed *HashCash* as a solution to stop unsolicited mass E-mailing (also known as spam). Key idea:

- the sender pays per E-mail
- instead of money, use CPU time (BitCoin anyone?)

Dot-bit (aka "namecoin") uses the same idea to limit ".bit" domain registrations.

HashCash: protocol

- In order to send an E-mail, the sender must find a collision in a hashcode.
- The hashcode can be provided by the receiver (challenge) or be derived from the E-mail with the receiver address and time for a non-interactive version.
- The number of bits that must match in the two hashcodes can be used to make it more or less expensive for the sender.

HashCash: problems

- Cost applies also for legitimate mass-mailings (aka mailinglists)
- CPU time is wasted, increased transaction costs
- Cost must be adjusted to match current CPUs, thus the protocol never benefits as better hardware becomes available!

Reputation

R. Dingledine, N. Mathewson and P. Syverson wrote about *Reputation in Privacy Enhancing Technologies*:

- Reputation is a way to track past performance and reward (Freehaven: you stored 1k for a week, I store 7k for a day).
- If reputation is global, claims must be verified, which can be very hard.
- If reputation is local, servers must *risk* resources to new nodes to keep the network open

$\Rightarrow$ Vulnerability: "screw every server once" attack

Problems with Reputation Systems

- Time-dependency of behaviour (Ebay attack)
- Whitewashing (Sybil attack)
- Collusion of attackers

Global Trust: Who watches the Watchers?

R. Dingledine and P. Syverson wrote about *Reliable MIX Cascade Networks through Reputation*:

- traditional approach uses external trusted witnesses that probe the mix

Global Trust: Who watches the Watchers?

R. Dingledine and P. Syverson wrote about *Reliable MIX Cascade Networks through Reputation*:

- traditional approach uses external trusted witnesses that probe the mix

Key idea — allow a mix-cascade to monitor itself:

- nodes send test-messages to monitor their own cascade
- nodes announce the failure of their own cascade, damaging the reputation of all nodes in the cascade
- nodes that misbehave by incorrectly reporting cascade failure damage their own reputation!

Problems

- All-bad cascade would never report failure and have perfect reputation
- Creeping death: adversary fails cascade if good nodes lose more than bad ones
- Still does not detect failure instantly (loss)
- Adversary could create fresh identities (need strong identities / WoT)

Reputation & Currency

R. Dingledine, N. Mathewson and P. Syverson ask in *Reputation in Privacy Enhancing Technologies*:

- Reputation as currency? Transitivity?
- Does reputation expire?
- Multiple currencies and convertability?
- **Where does currency come from?**

Marx & the Origins of Capital

Capital is money that can be used to make more money.

Marx & the Origins of Capital

Capital is money that can be used to make more money.

Historically, **original accumulation** needed to create capital.

Marx & the Origins of Capital

Capital is money that can be used to make more money.

Historically, **original accumulation** needed to create capital.

Original accumulation = resource extraction, spoils of war.

Where does Money come from?

Where does Money come from?

Money is loaned into existence.

Where does Money come from?

Money is loaned into existence.

- This requires a central bank and/or strong identities

Where does Money come from?

Money is loaned into existence.

- This requires a central bank and/or strong identities
- Resource extraction requires some kind of mining...

Where does Money come from?

Money is loaned into existence.

- This requires a central bank and/or strong identities
- Resource extraction requires some kind of mining...
- Transaction costs are important for an efficient economy

The Excess Based Economy

C. Grothoff proposed an *Excess Based Economy* [1]:

- use respect as a "private currency"
- but trust no one except your resource allocation algorithm

Goals:

- Support accounting for fair resource allocation
- ⇒ Reward contributing nodes with better service
- Fully decentralized, no trusted authority
- New nodes must be able to join the network,
  Sybil attack must be ineffective
- Efficient in bandwidth and CPU consumption

Excess Based Economy: Security Model

Adversary model:

- Everybody else is malicious and violates the protocols
- Everybody can make-up a new identity at any time (without being detected)
- Public keys are identities

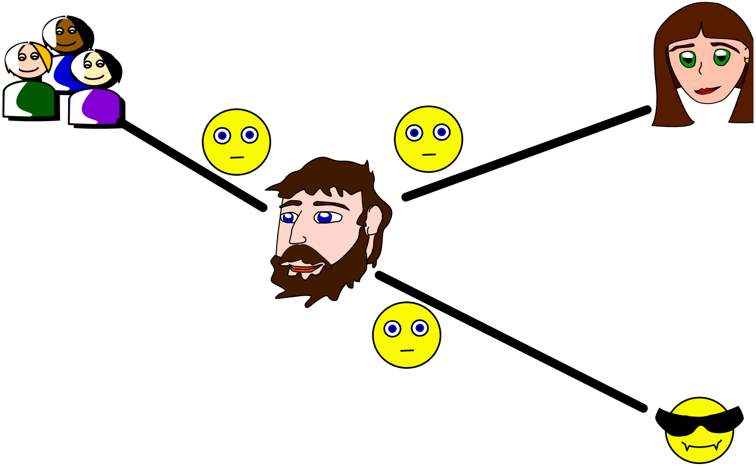
Threat model:

- detect flooding / abusive behaviour / excessive free-loading
- but allow *harmless* amounts of free-loading

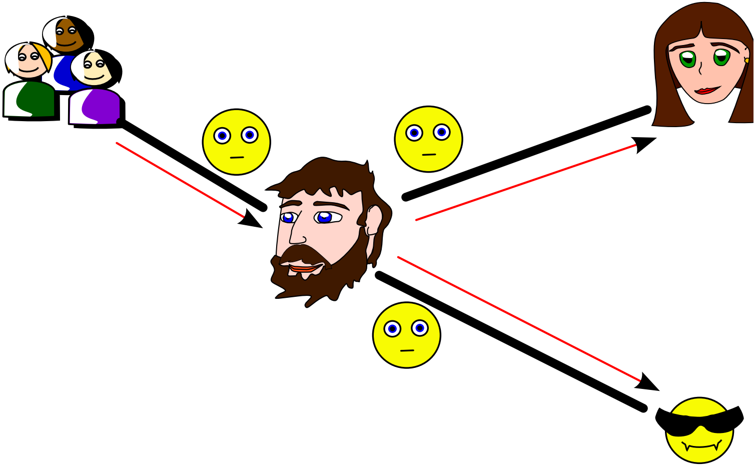
Excess Based Economy: Human Relationships

- We do not have to *respect* anybody to form an opinion.
- Opinions are formed on a one-on-one basis, and
- may not be perceived equally by both parties.
- We do *not* charge for every little favour.
- We *are* grateful for every favour.
- There is no guarantee in life, in particular Alice does not have to be kind to Bob because he was kind to her.

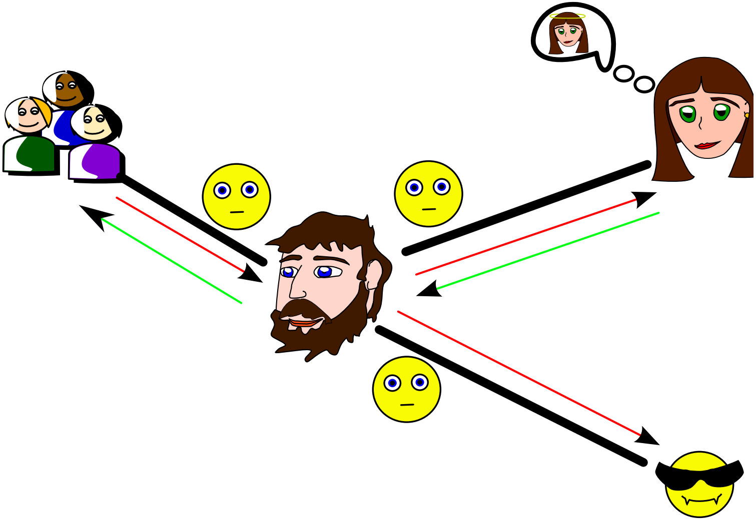
Excess-based Economy Illustrated (1/8)



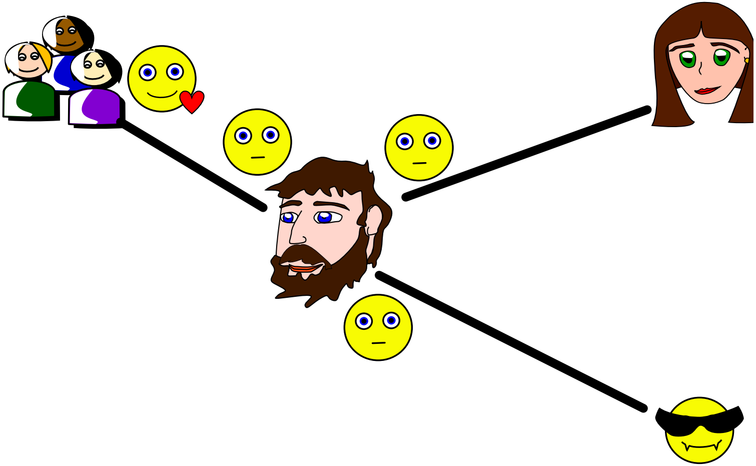
Excess-based Economy Illustrated (2/8)



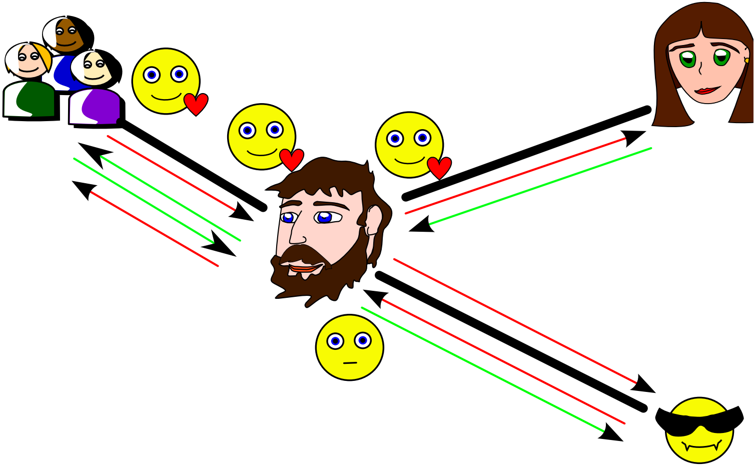
Excess-based Economy Illustrated (3/8)



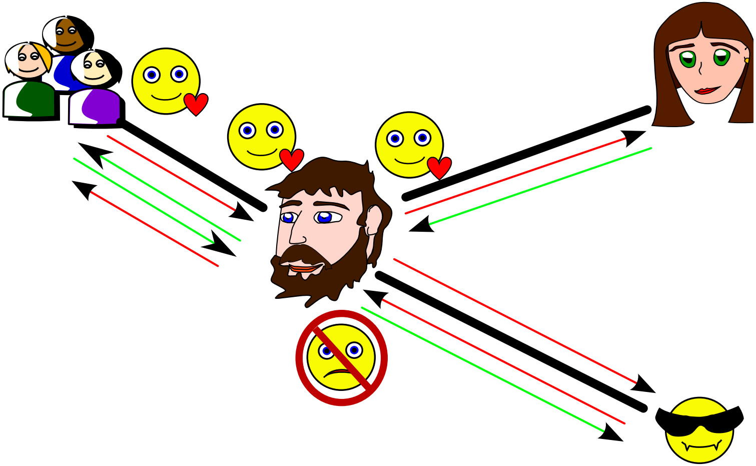
Excess-based Economy Illustrated (4/8)



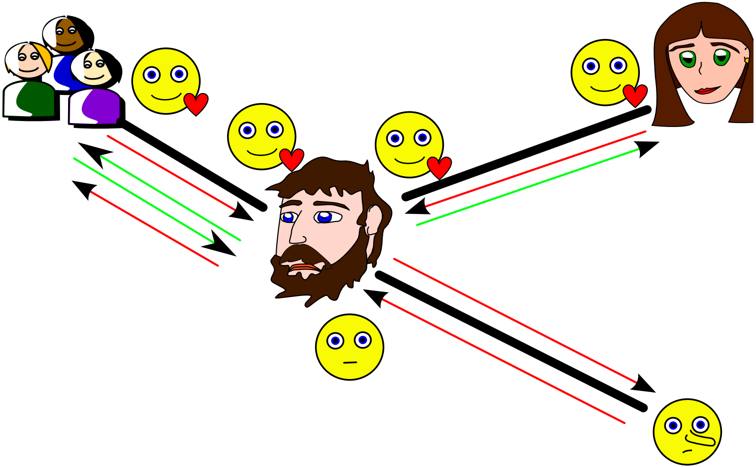
Excess-based Economy Illustrated (5/8)



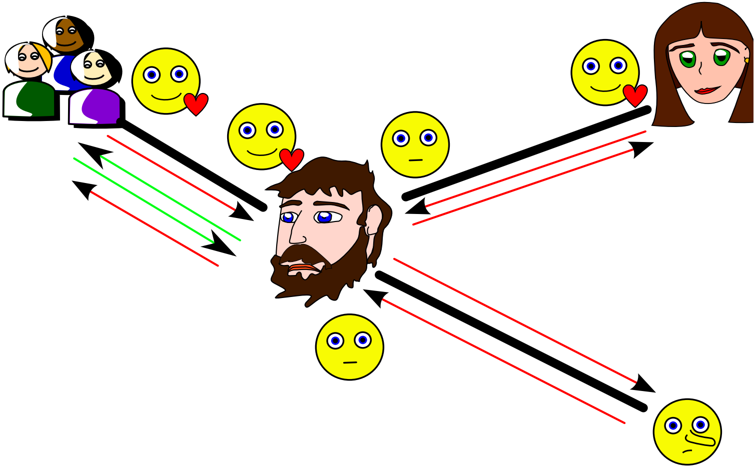
Excess-based Economy Illustrated (6/8)



Excess-based Economy Illustrated (7/8)



Excess-based Economy Illustrated (8/8)



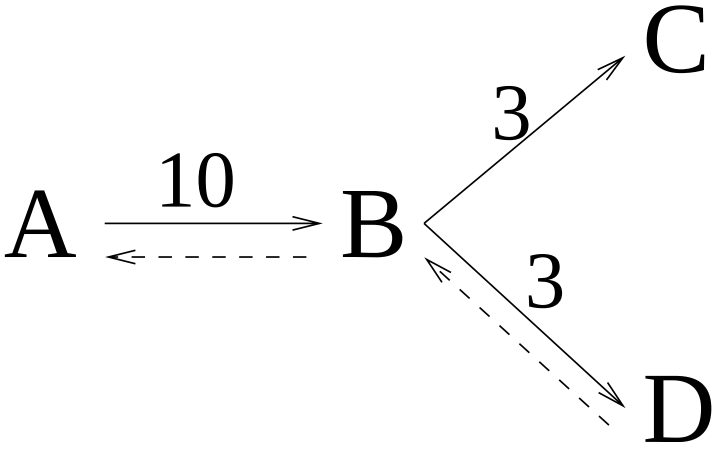
Excess-based Economy

The Excess-based economy is based on the following principals:

- if you are *idle*, doing a favour for free does not cost anything;
- if somebody does you a favour, remember it;
- if you are *busy*, work for whoever you like most, but remember that you paid the favour back;
- have a *neutral* attitude towards new entities;
- never disrespect anybody (they could create a new identity anytime).

Excess Based Economy: Transitivity

If a node acts on behalf on another, it must ensure that the sum of the charges it may suffer from other nodes is lower than the amount it charged the sender:



Excess Based Economy: Open Issues

- If a node is idle, it will not charge the sender; if a node delegates (indirects), it will use a lower priority than the amount it charged itself; if an idle node delegates, it will always give priority 0. A receiver can not benefit from answering a query with priority 0.
- If the priority is 0, content will not be marked as valuable.
- under heavy use and long attacks, all respect may disappear

Excess Based Economy: Achievements

We have presented an economic model, that:

- solves the problem of primitive accumulation
- does not rely on trusted entities
- can be used for resource allocation
- requires link-to-link authenticated messages, but no other cryptographic operations
- does not require a global view of the transaction and can thus be used with anonymous participants

Economy: Requirements for Encoding

- Need content encoding that makes cheating not viable!

The Encoding for Censorship Resistant Sharing (ECRS)

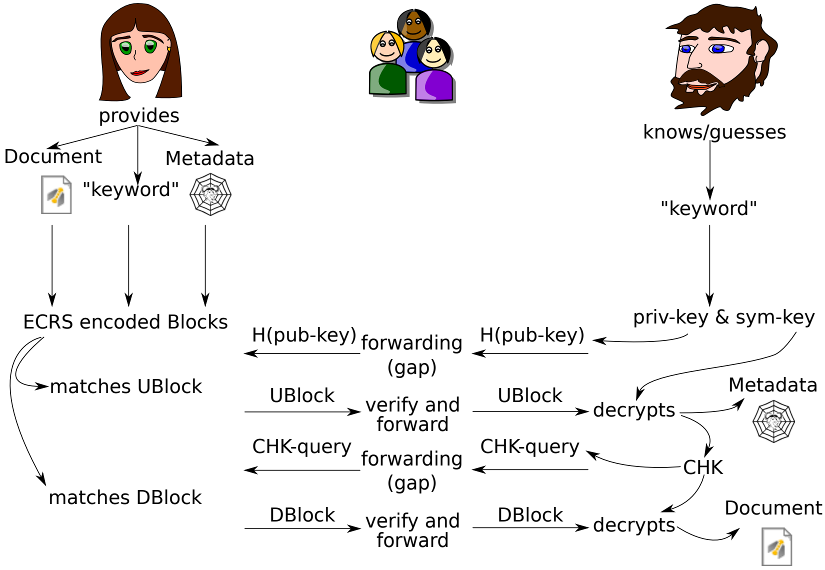
GNUnet file-sharing uses ECRS to:

- prevent cheating
- preserve privacy
- support search

Problems with Other Encoding Mechanisms

- Content distributed in plaintext (e.g. gnutella) facilitates censorship and may void deniability
- Content must be inserted into the network and is then stored twice, in plaintext (by the originator) and encrypted (by the network – e.g. Freenet)
- Independent insertions of the same file result in different copies in the network (e.g. Publius)
- Verification of content integrity can only occur after download is complete (most systems)

ECRS Overview



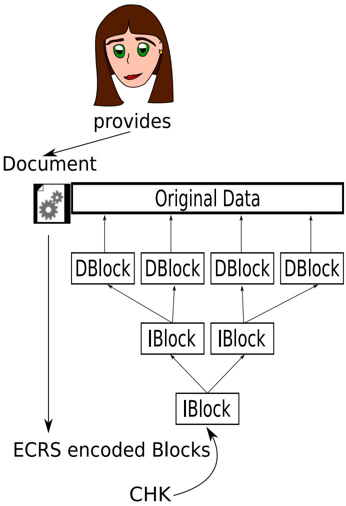
Properties of ECRS

- Breaks files into small blocks
  $\Rightarrow$ operations performed by routers are $O(1)$
- Operations performed by responders are $O(\log n)$ where n is the size of the datastore
- All receiver operations have (amortized) runtime $O(n)$ where n is the size of the result set or the size of the file; memory use for files is $O(\log n)$

Properties of ECRS

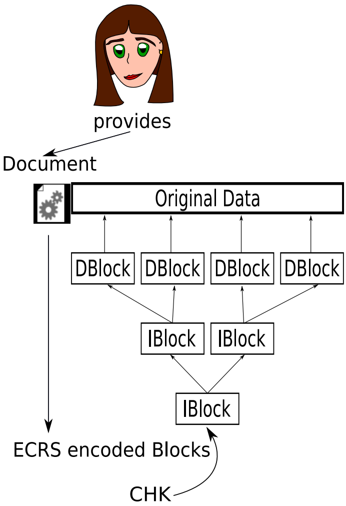
- Breaks files into small blocks
  $\Rightarrow$ operations performed by routers are $O(1)$
- Operations performed by responders are $O(\log n)$ where n is the size of the datastore
- All receiver operations have (amortized) runtime $O(n)$ where n is the size of the result set or the size of the file; memory use for files is $O(\log n)$
- Intermediaries *cannot* view content or queries
  $\Rightarrow$ Peers can send replies to queries and plausibly deny having knowledge of their contents
- Intermediaries *are* able to verify validity of responses
  $\Rightarrow$ Enables swarming, even in the presence of malicious peers trying to corrupt files

ECRS Details: Document Encoding



- Split content into 32k blocks B
- AES-256 encrypt B with $H(B)$
- Store $E_{H(B)}(B)$ under $H(E_{H(B)}(B))$
- Build tree containing up to 256 CHK pairs: $H(B), H(E_{H(B)}(B))$

ECRS Details: Document Encoding

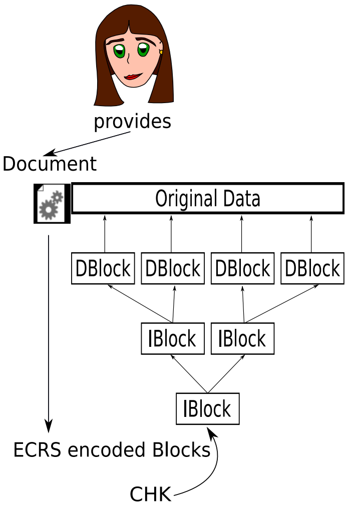


- Encryption of blocks independent of each other
- Inherent integrity checks
- Multiple (independent) insertions result in identical blocks
- Small blocksize makes traffic more uniform
  $\Rightarrow$ traffic analysis is harder

ECRS Details: Document Encoding Limitations

- If the exact data can be guessed... participating hosts can match the content.
  Intended to reduce storage costs!

ECRS Search Design Requirements



- Retrieve content with simple, natural-language keyword
- Guard against malicious hosts:
  prevent attackers from providing useless replies!
- Do not expose actual keyword used
- Do not expose CHK or metadata:
  encrypt CHK and metadata as well!

ECRS Searching: UBlocks

Let R be the (plaintext) metadata and CHK.

- For each keyword K use $d := H(H(K)) \mod n$ to generate ECDSA key (d, Q) with $Q := dG$.
- Store $E_{H(K)}(R), Q, r, s$ under $H(Q)$ where (r, s) is ECDSA signature with d
- User searching also computes Q from K and sends query $H(Q)$
- Intermediates match Q against $H(Q)$ and verify signature

Benefits and Limitations of UBlocks

- + Malicious peer cannot learn d or $H(K)$ without guessing the keyword
- + Malicious peer must guess keyword to generate valid reply
- + Malicious peer cannot modify reply without being detected
- - Requies public key cryptography

Pseudonyms, Namespaces and Updates

Let x be the private ECDSA key for some pseudonym and $h := H(H(K))$ for each identifier K.

- Let $Q := xG$ be the public key of the namespace.
- Let $d := x + h \mod n$ and $V := dG$.
- Sign $E_{H(K) xor Q}(R)$ with ECDSA key d and store under $H(V)$.
- User searching has Q and K, computes $V = Q + hG$ and sends query $H(V)$
- Intermediates match V against $H(V)$ and verify signature

Meta data R can include information about updates.

Benefits and Limitations of UBlocks

- \+ Only pseudonym owner knows x
- \+ Only pseudonym owner can publish in namespace
- \+ Malicious peer cannot generate valid reply
- \+ Malicious peer cannot modify reply without being detected
- \+ Malicious peer cannot distinguish keyword search from namespace search (without guessing keyword)
- \- Requies public key cryptography

The Multiple Search Result Problem

- Responder can not send "fake" response (ECRS)
- Responder can send same response again and again

⇒ No incentive to look for alternative responses!

⇒ First (few) responses to keyword spread far and wide, others will never be displayed!

⇒ Need to use creative keywords (but in that case, caching is much less effective!)

Solution (1/2)

- As part of the query, communicate what replies are *not* acceptable
- Can not include full replies (too big)

$\Rightarrow$ Use bloomfilter of hash codes of encrypted replies

Solution (2/2)

- Bloomfilter is probabilistic
- Even relatively generous bloomfilters would filter approximately $1{:}2^{10}$ valid replies
- Solution: add random 32-bit nounce to hash function, change nounce (sometimes) when repeating requests

⇒ False-positives less than $1{:}2^{42}$

Open Issues

- Approximate queries

References

Christian Grothoff.
An Excess-Based Economic Model for Resource Allocation in Peer-to-Peer Networks.
Wirtschaftsinformatik, 3-2003, June 2003.

Mao Yang, Zheng Zhang, Xiaoming Li, and Yafei Dai.
An empirical study of free-riding behavior in the maze p2p file-sharing system.
In *Proceedings of the 4th international conference on Peer-to-Peer Systems*, IPTPS'05, pages 182–192, Berlin, Heidelberg, 2005. Springer-Verlag.