Decentralized Public Key Infrastructures

Christian Grothoff

Berner Fachhochschule

24.5.2019

Learning Objectives

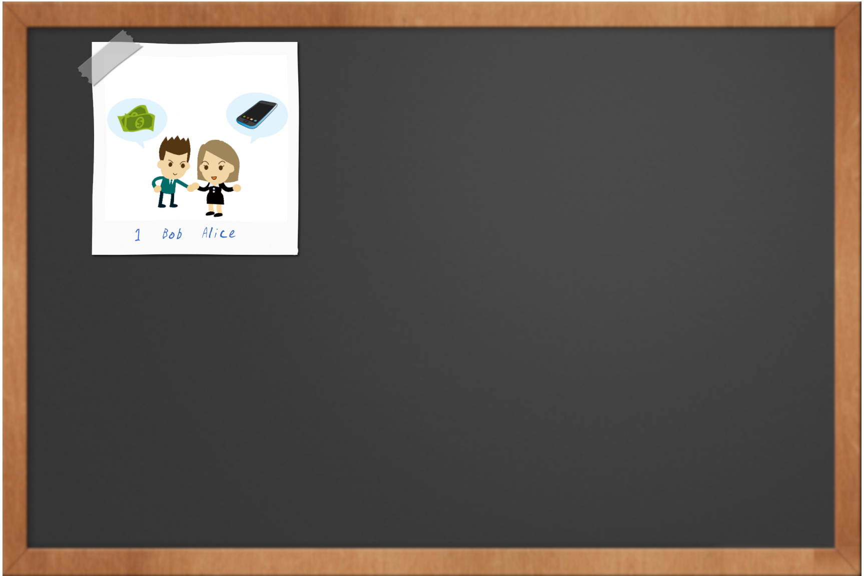
Learn about:

- Blockchain Blockchain Blockchain!
- Ideas behind the Web of Trust
- Using GnuPG
- Goals and theory behind Fog of Trust
- Modern public key infrastructures

Blockchain[1]

[1]Illustrations by Alexandra Dirksen, IAS, TUBS [1]

Blockchain

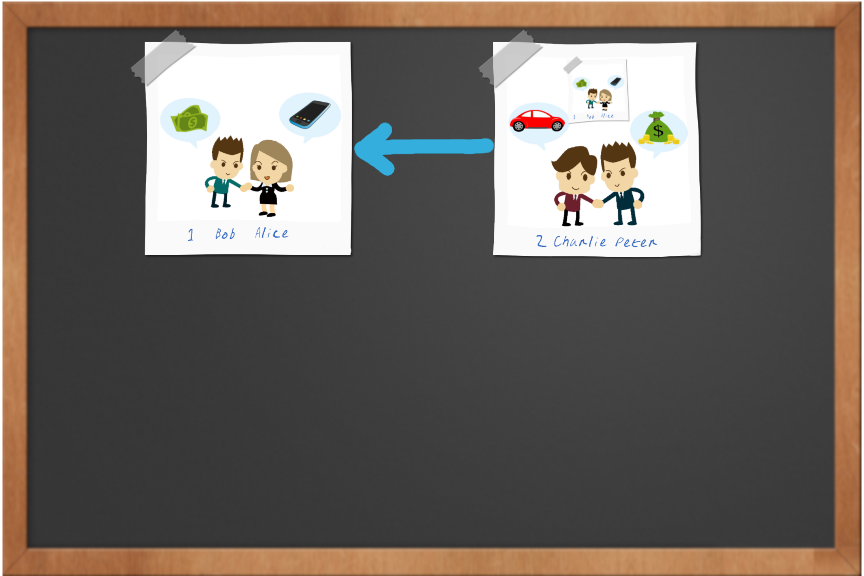


Blockchain

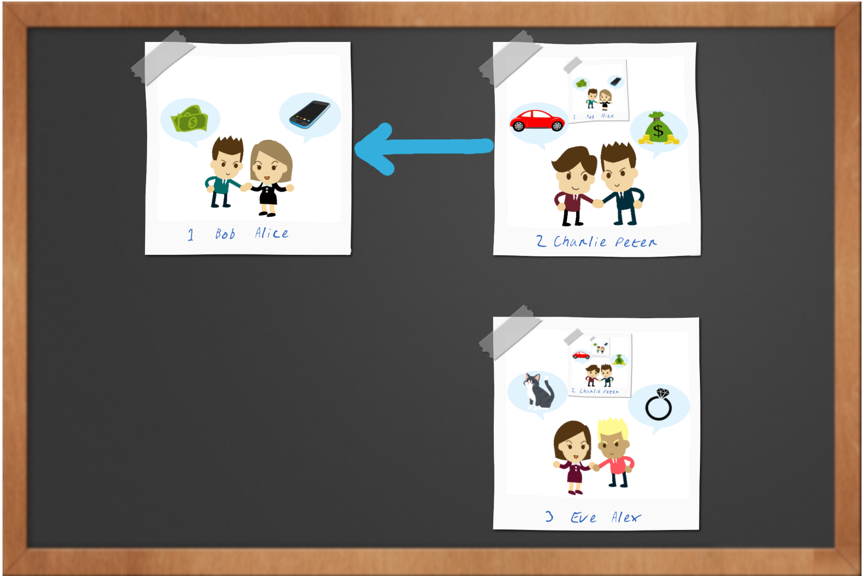
Blockchain

Blockchain

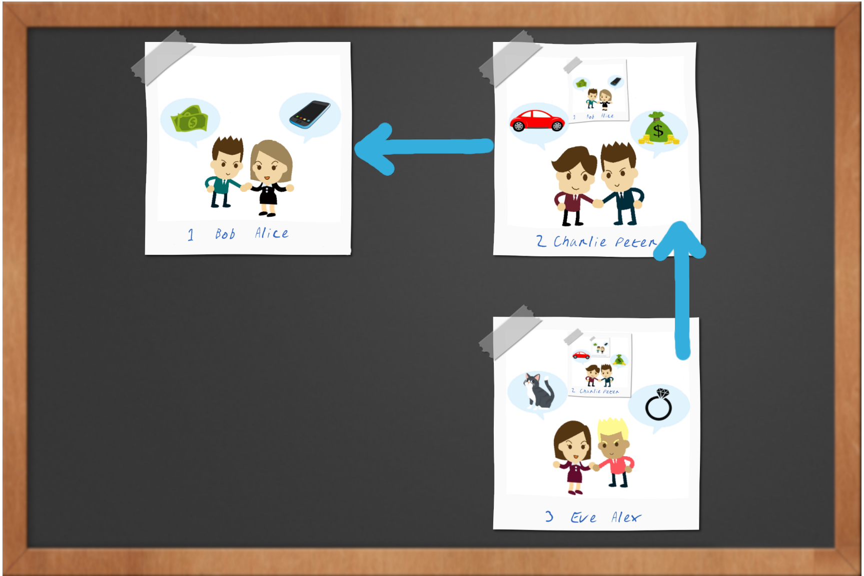
Blockchain



Blockchain

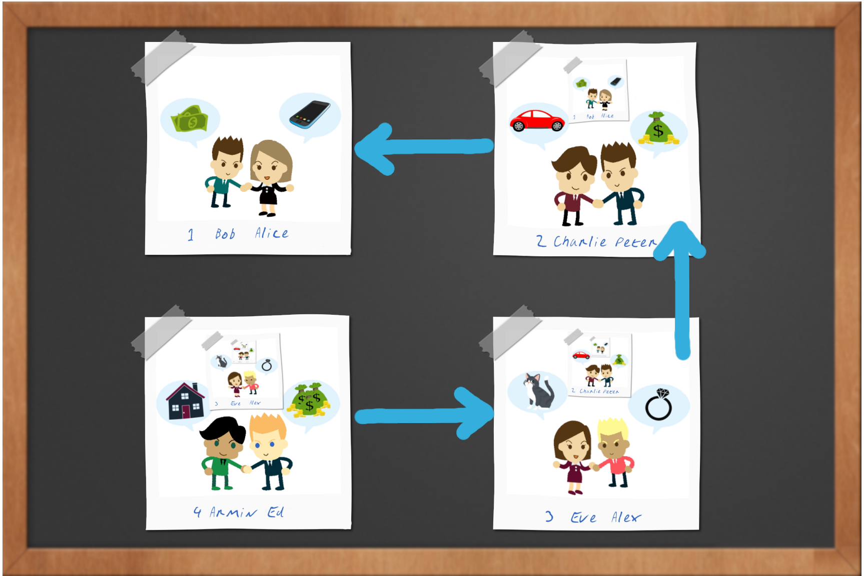


Blockchain

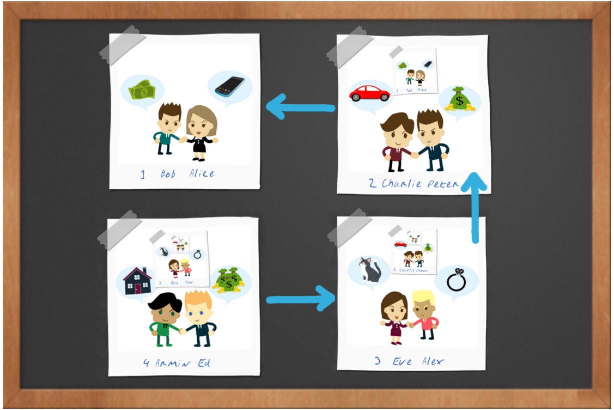


Blockchain

Blockchain

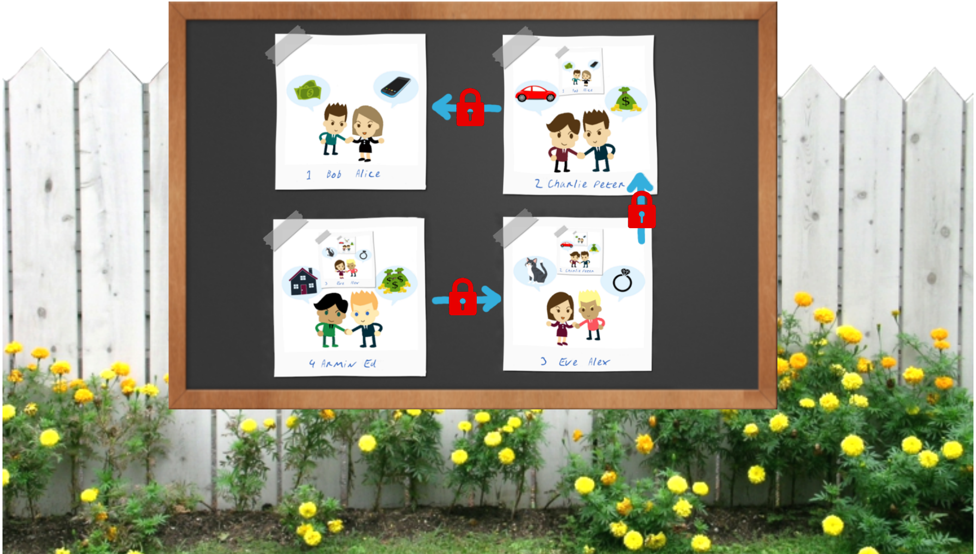


Advertised Blockchain "properties"

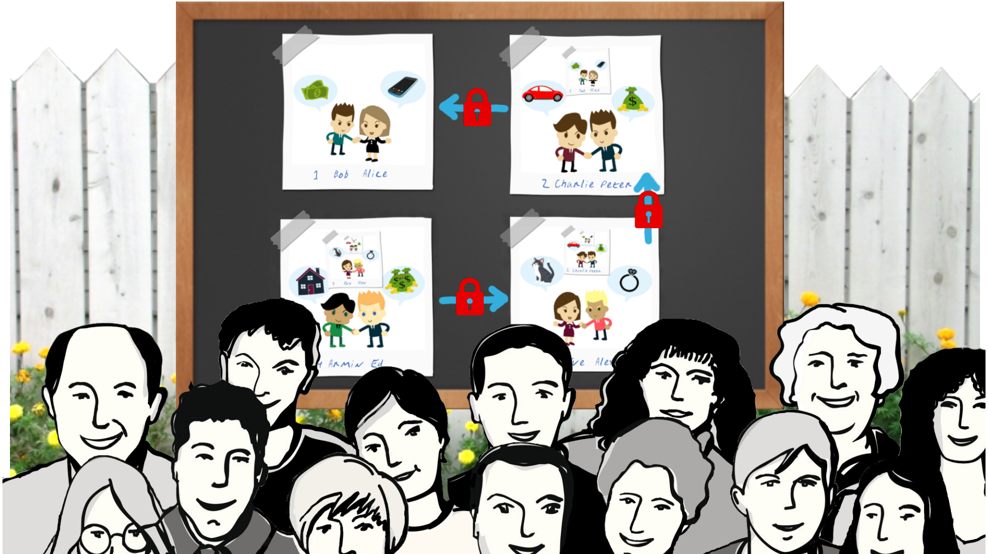


Immutability

Transparency

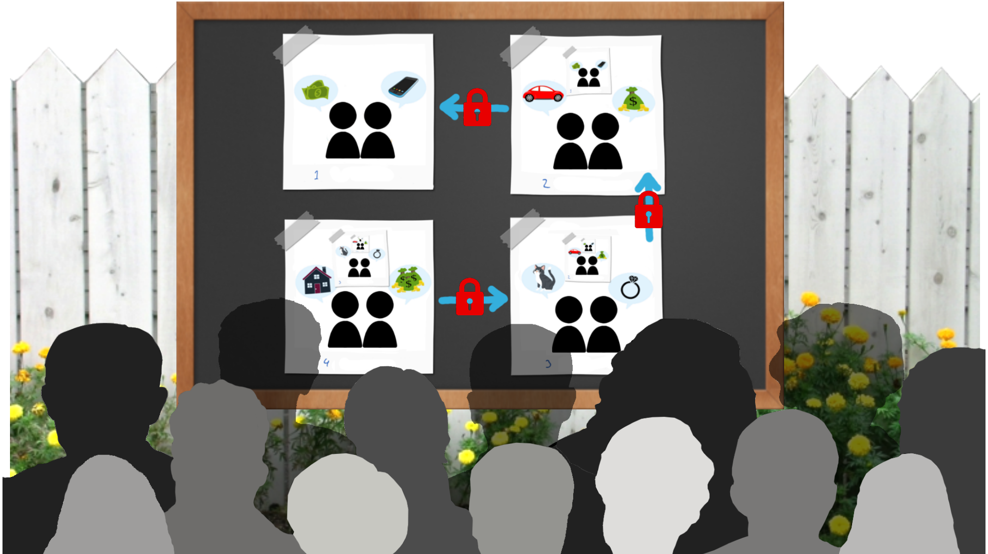


Decentralisation



Autonomy

Anonymity

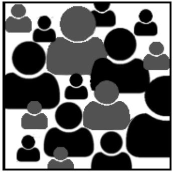


Blockchain “properties”[2]

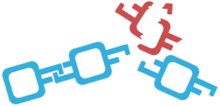
Immutability

Transparency

Anonymity



Decentralisation



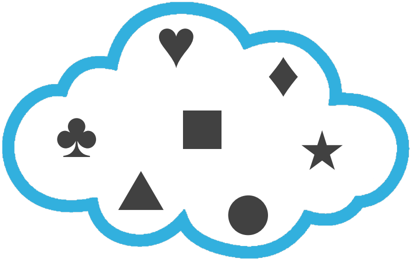
Irreversibility

Autonomy

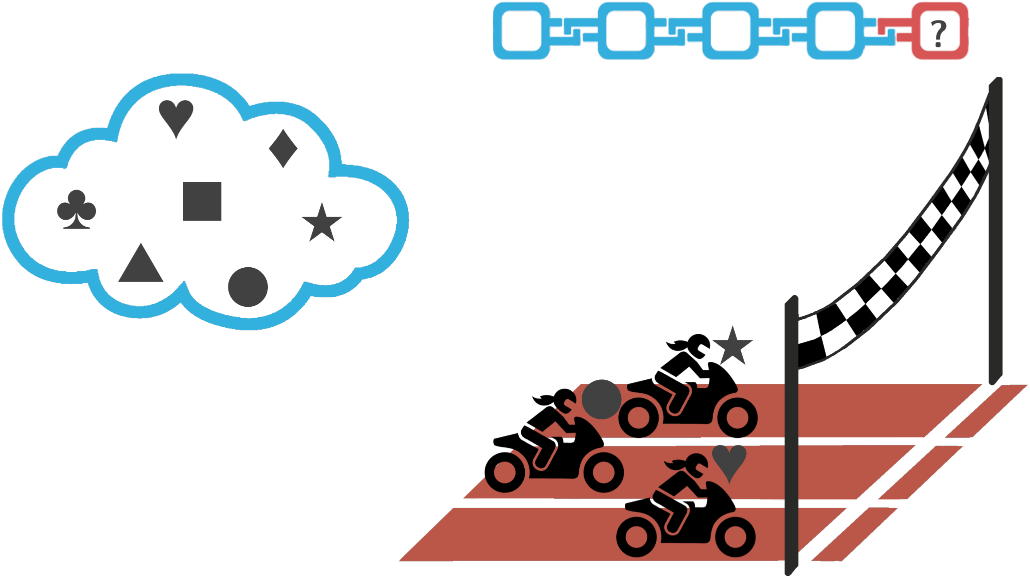
[2]These **only** hold with many **significant caveats**!

Who gets to append the next block?

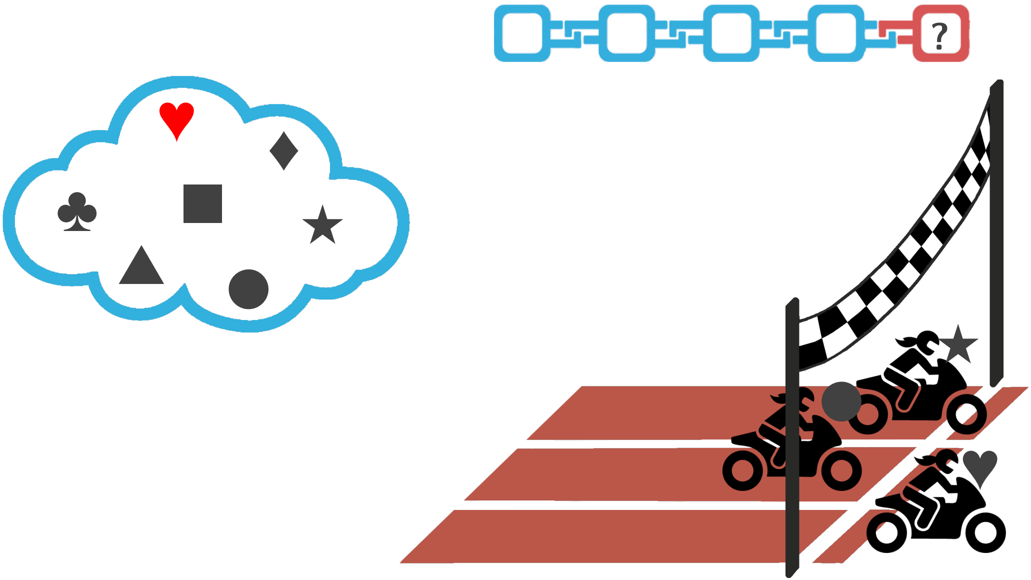
Proof of Work



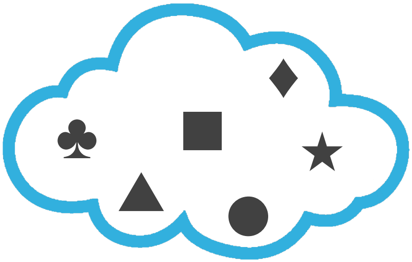
Proof of Work

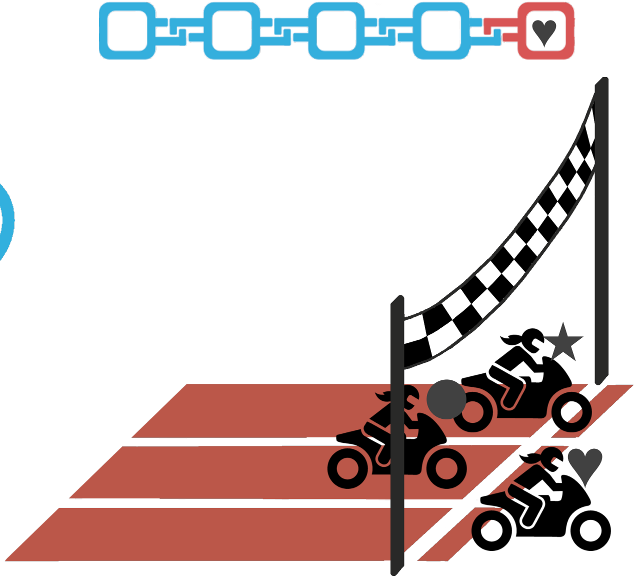


Proof of Work

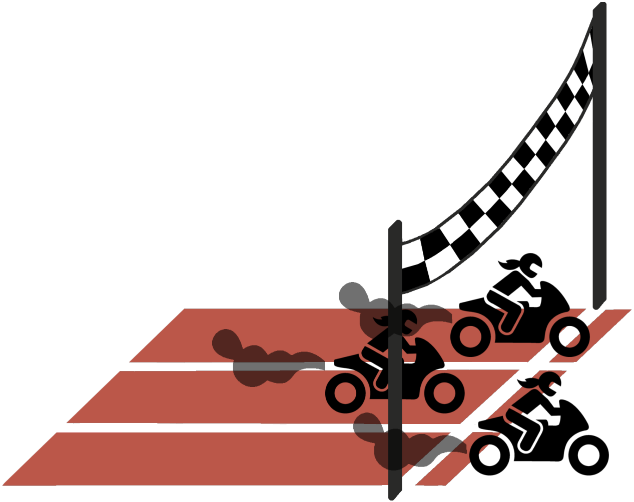


Proof of Work

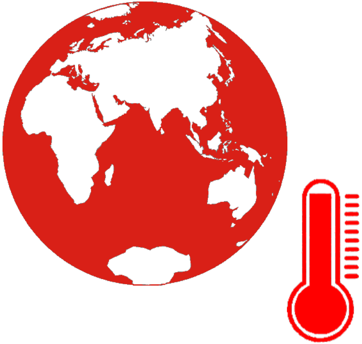


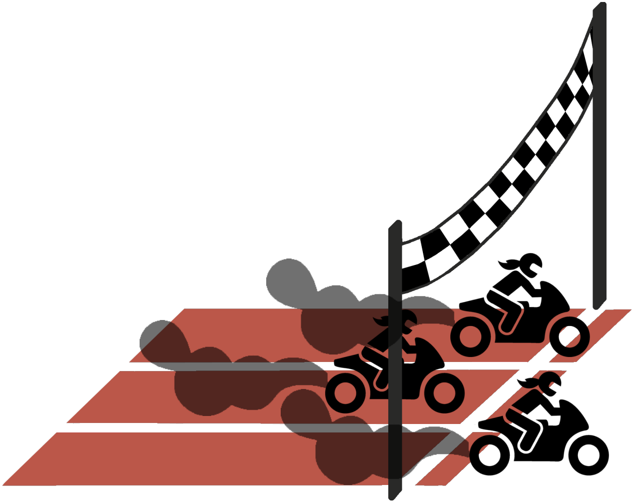


Proof of Work



Proof of Work





Break

GnuPG

- Free version of PGP, with library (libgcrypt)
- Provides common cryptographic primitives
- Provides implementation of OpenPGP (RFC 2440)
- Commonly used for secure E-mail
- Provides web of trust

Using GnuPG

- $ gpg –gen-key
- $ gpg –export
- $ gpg –import FILENAME
- $ gpg –edit-key EMAIL; > fpr > sign > trust
- $ gpg –clearsign FILENAME

The Web of Trust

Problem:

- Alice has certified many of her contacts and *flagged* some as *trusted* to check keys well.
- Bob has been certified by many of his contacts.
- Alice has **not** yet certified Bob, but wants to securely communicate with him.

The Web of Trust

Problem:

- Alice has certified many of her contacts and *flagged* some as *trusted* to check keys well.
- Bob has been certified by many of his contacts.
- Alice has **not** yet certified Bob, but wants to securely communicate with him.

Solution:

- Find paths in the certification graph from Alice to Bob.
- If sufficient number of short paths exist certifying the same key, trust it.

Excercise: Explore

`http://pgp.mit.edu`

Pairing-based cryptography

Let G_1, G_2 be two additive cyclic groups of prime order q, and G_T another cyclic group of order q (written multiplicatively). A pairing is an efficiently computable map e:

$$e : G_1 \times G_2 \to G_T \tag{1}$$

which satisfies $e \neq 1$ and bilinearity:

$$\forall_{a,b \in F_q^*},\ \forall_{P \in G_1, Q \in G_2} :\ e\left(aP, bQ\right) = e\left(P, Q\right)^{ab} \tag{2}$$

Examples: Weil pairing, Tate pairing.

Hardness assumption

Computational Diffie Hellman:

$$g, g^x, g^y \Rightarrow g^{xy} \tag{3}$$

remains hard on G even given e.

Boneh-Lynn-Sacham (BLS) signatures

Key generation:

Pick random $x \in \mathbb{Z}_q$

Signing:

$\sigma := h^x$ where $h := H(m)$

Verification:

Given public key g^x:

$$e(\sigma, g) = e(h, g^x) \tag{4}$$

Boneh-Lynn-Sacham (BLS) signatures

Key generation:

Pick random $x \in \mathbb{Z}_q$

Signing:

$\sigma := h^x$ where $h := H(m)$

Verification:

Given public key g^x:

$$e(\sigma, g) = e(h, g^x) \tag{4}$$

Why:

$$e(\sigma, g) = e(h, g)^x = e(h, g^x) \tag{5}$$

due to bilinearity.

Fun with BLS

Given signature $\langle \sigma, g^x \rangle$ on message h, we can *blind* the signature and public key g^x:

$$e(\sigma^b, g) = e(h, g)^{xb} = e(h, g^{xb}) \tag{6}$$

Thus σ^b is a valid signature for the *derived* public key $(g^x)^b$ with blinding value $b \in \mathbb{Z}_q$.

Break

The Fog of Trust

Problem:

- Publishing who certified whom exposes the social graph.
- The "NSA kills based on meta data".

The Fog of Trust

Problem:

- Publishing who certified whom exposes the social graph.
- The "NSA kills based on meta data".

Solution:

- Do not publish the graph.
- Have Alice and Bob collect their certificates locally.
- Use SMC protocol for

  private set intersection cardinality with signatures!

We will only consider paths with **one** intermediary.

Straw-man version of protocol 1

Problem: Alice wants to compute $n := |\mathcal{L}_A \cap \mathcal{L}_B|$

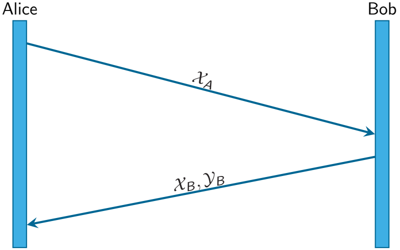
Suppose each user has a private key c_i and the corresponding public key is $C_i := g^{c_i}$ where g is the generator

The setup is as follows:

- $\mathcal{L}_A$: set of public keys representing Alice trusted verifiers
- $\mathcal{L}_B$: set of public keys representing Bob's signers
- Alice picks an ephemeral private scalar $t_A \in \mathbb{F}_p$
- Bob picks an ephemeral private scalar $t_B \in \mathbb{F}_p$

Straw-man version of protocol 1

$$\mathcal{X}_A := \left\{ C^{t_A} \;\middle|\; C \in \mathcal{L}_A \right\}$$



$$\mathcal{X}_B := \left\{ C^{t_B} \;\middle|\; C \in \mathcal{L}_B \right\}$$

$$\mathcal{Y}_B := \left\{ \overline{C}^{t_B} \;\middle|\; \overline{C} \in \mathcal{X}_A \right\} = \left\{ C^{t_B \cdot t_A} \;\middle|\; C \in \mathcal{L}_B \right\}$$

$$\mathcal{Y}_A := \left\{ \hat{C}^{t_A} \;\middle|\; \hat{C} \in \mathcal{X}_B \right\} = \left\{ C^{t_A \cdot t_B} \;\middle|\; C \in \mathcal{L}_A \right\}$$

Alice can get $|\mathcal{Y}_A \cap \mathcal{Y}_B|$ at linear cost.

Attack against the Straw-man

If Bob controls two trusted verifiers $C_1, C_2 \in \mathcal{L}_A$, he can:

- Detect relationship between $C_1^{t_A}$ and $C_2^{t_A}$
- Choose $K \subset \mathbb{F}_p$ and substitute with fakes:

$$\mathcal{X}_B := \bigcup_{k \in K} \left\{ C_1^k \right\}$$

$$\mathcal{Y}_B := \bigcup_{k \in K} \left\{ (C_1^{t_A})^k \right\}$$

so that Alice computes $n = |K|$.

Cut & choose version of protocol 1: Preliminaries

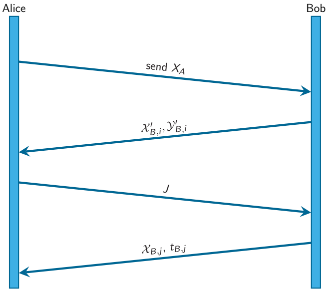
Assume a fixed system security parameter $\kappa \geq 1$.

Let Bob use secrets $t_{B,i}$ for $i \in \{1, \ldots, \kappa\}$, and let $\mathcal{X}_{B,i}$ and $\mathcal{Y}_{B,i}$ be blinded sets over the different $t_{B,i}$ as in the straw-man version.

For any list or set Z, define

$$Z' := \{h(x) | x \in Z\} \tag{7}$$

Cut & choose version of protocol 1



Protocol messages:

1. Alice sends:
   $$\mathcal{X}_A := \texttt{sort}\,[\, C^{t_A} \mid C \in \mathcal{A} \,]$$
2. Bob responds with commitments:
   $$\mathcal{X}'_{B,i}, \mathcal{Y}'_{B,i} \quad \textit{for} \;\; i \in 1, \ldots, \kappa$$
3. Alice picks a non-empty random subset $J \subseteq \{1, \ldots, \kappa\}$ and sends it to Bob.
4. Bob replies with $\mathcal{X}_{B,j}$ for $j \in J$, and $t_{B,j}$ for $j \notin J$.

Cut & choose version of protocol 1: Verification

For $j \notin J$, Alice checks the $t_{B,j}$ matches the commitment $\mathcal{Y}'_{B,j}$.

For $j \in J$, she verifies the commitment to $\mathcal{X}_{B,j}$ and computes:

$$\mathcal{Y}_{A,j} := \left\{ \hat{C}^{t_A} \,\middle|\, \hat{C} \in \mathcal{X}_{B,j} \right\} \tag{8}$$

To get the result, Alice computes:

$$n = |\mathcal{Y}'_{A,j} \cap \mathcal{Y}'_{B,j}| \tag{9}$$

Alice checks that the n values for all $j \in J$ agree.

Protocol 2: Private Set Intersection with Subscriber Signatures

- Naturally, signers are willing to *sign* that Bob's key is Bob's key.
- We still want the identities of the signers to be private!
- BLS (Boneh et. al) signatures are compatible with our blinding.

$\Rightarrow$ Integrate them with our cut & choose version of the protocol.

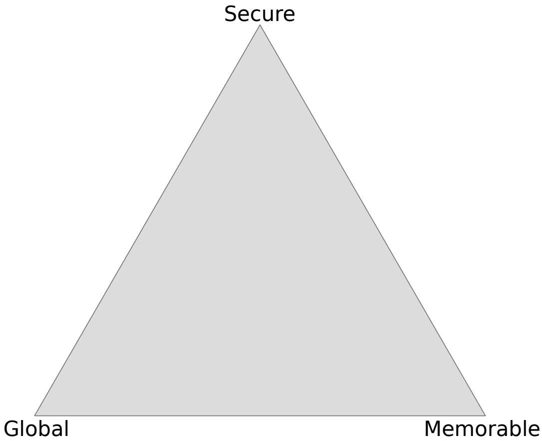
Costs are linear in set size. Unlike prior work this needs no CA.

Break

Security Goals for Name Systems

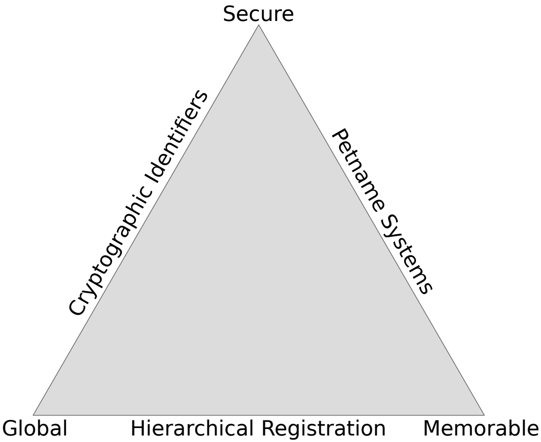
- Query origin anonymity
- Data origin authentication and integrity protection
- Zone confidentiality
- Query and response privacy
- Censorship resistance
- Traffic amplification resistance
- Availability

Zooko's Triangle



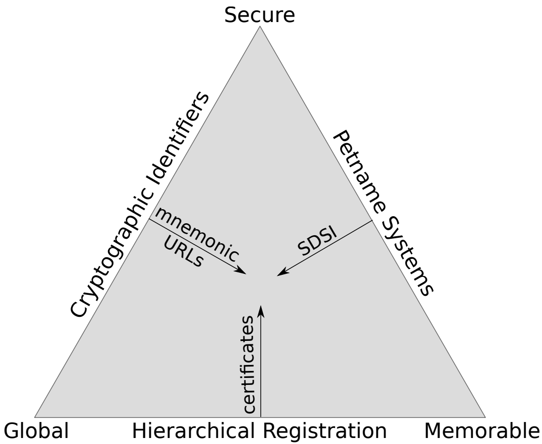
A name system can only fulfill **two**!

Zooko's Triangle



DNS, ".onion" IDs and `/etc/hosts/` are representative designs.

Zooko's Triangle



Approaches Adding Cryptography to DNS

- DNSSEC
- DNSCurve
- DNS-over-TLS
- DNS-over-HTTPS
- RAINS

Case study: DoH

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"The IETF is standardizing DNS over HTTPS (DOH), where all DNS queries are sent over the HTTPS protocol to some well-known HTTPS server (such as Google's 8.8.8.8 or Cloudflare's 1.1.1.1). This will prevent local governments from manipulating DNS traffic and improve the user's privacy with respect to their ISPs and governments. However, Google or Cloudflare will see the DNS queries and replies of the users, and they must be expected to have weak privacy policies and are subject to US law which includes secret rules and court orders. The NSA has a history of snooping on (MORECOWBELL) and manipulating (QUANTUMDNS) DNS traffic."*

Discuss virtues and vices affected.

Case study: RAINS

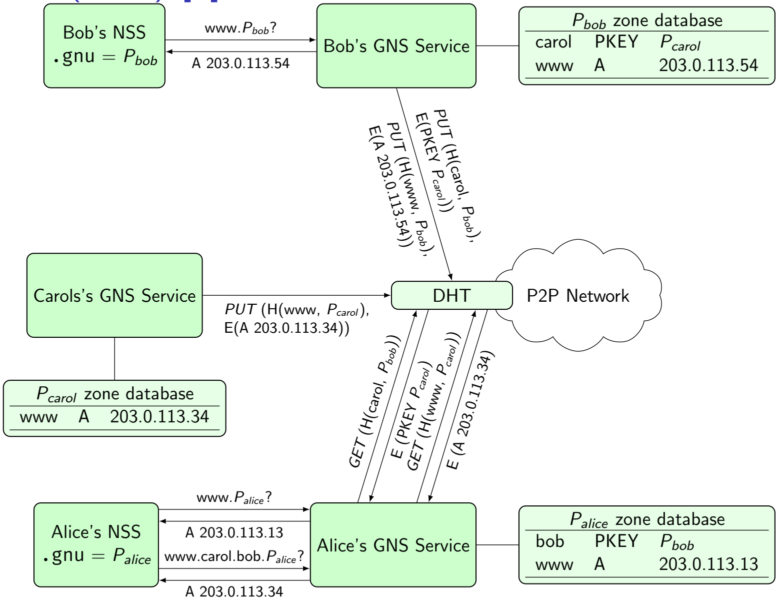
DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"The ETH Zurich is developing a new name system called RAINS with a new trust anchor operated by the regional Internet service provides, aka the local Isolation Service Domain (ISD). RAINS does not change the privacy of DNS (provides can continue to monitor traffic, all zone data becomes public) and allows the local authorities to block Web sites to improve public safety and enforce local laws (see also: "Glücksspielgesetz in Switzerland"). At the same time, foreign censorship efforts are less likely to be effective (unless they foreign government forces the DNS authority to alter the authoritative records)."*

Discuss virtues and vices affected.

Break

The GNU Name System (GNS) [2]



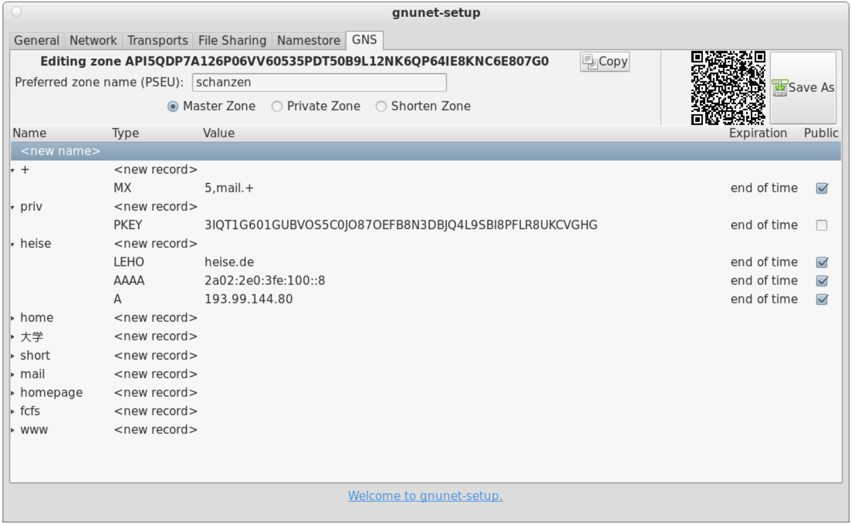
The GNU Name System[3]

Properties of GNS

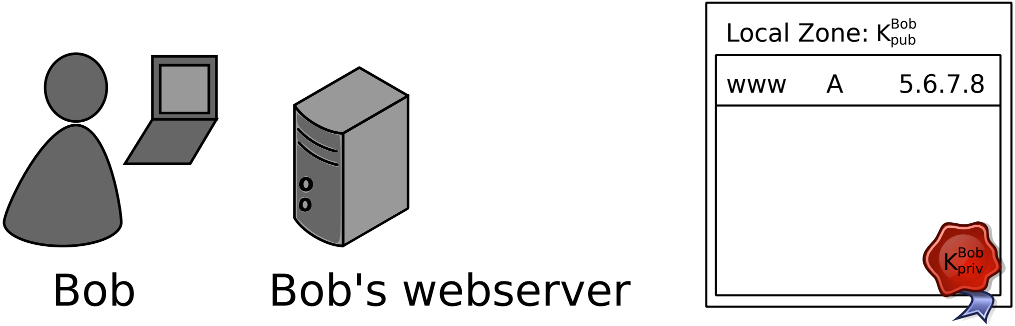
- Decentralized name system with secure memorable names
- Delegation used to achieve transitivity
- Also supports globally unique, secure identifiers
- Achieves query and response privacy
- Provides alternative public key infrastructure
- Interoperable with DNS

[3] Joint work with Martin Schanzenbach and Matthias Wachs

Zone Management: like in DNS



Name resolution in GNS

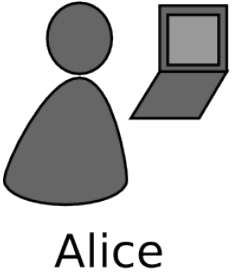


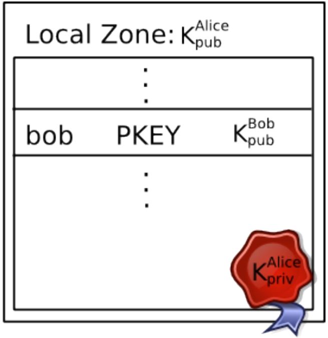
Secure introduction

Bob Builder, Ph.D.

Address: Country, Street Name 23
Phone: 555-12345
Mobile: 666-54321
Mail: bob@H2R84L4JIL3G5C.zkey

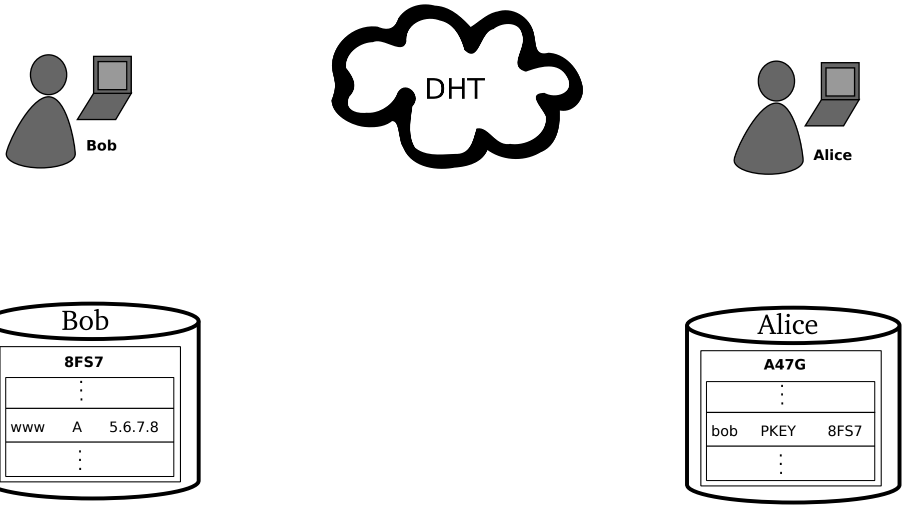
Delegation



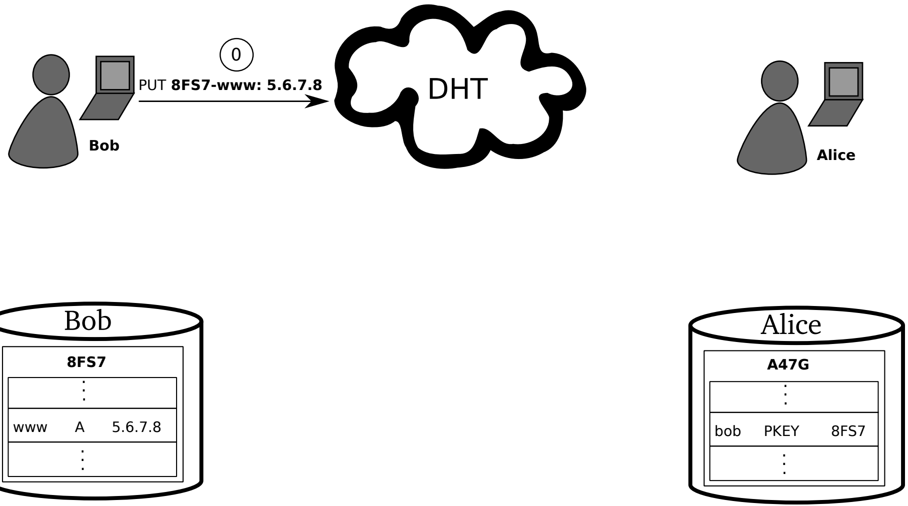


- Alice learns Bob's public key
- Alice creates delegation to zone K_{pub}^{Bob} under label **bob**
- Alice can reach Bob's webserver via **www.bob.gnu**

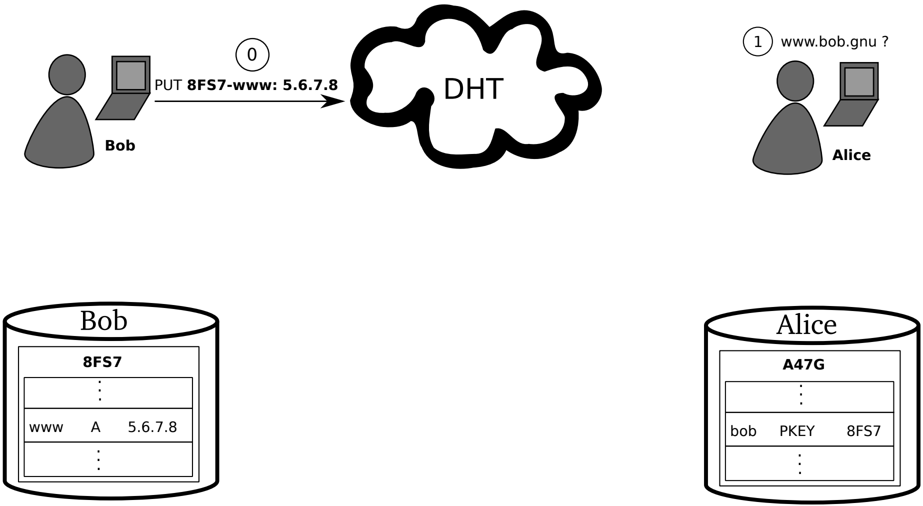
Name Resolution



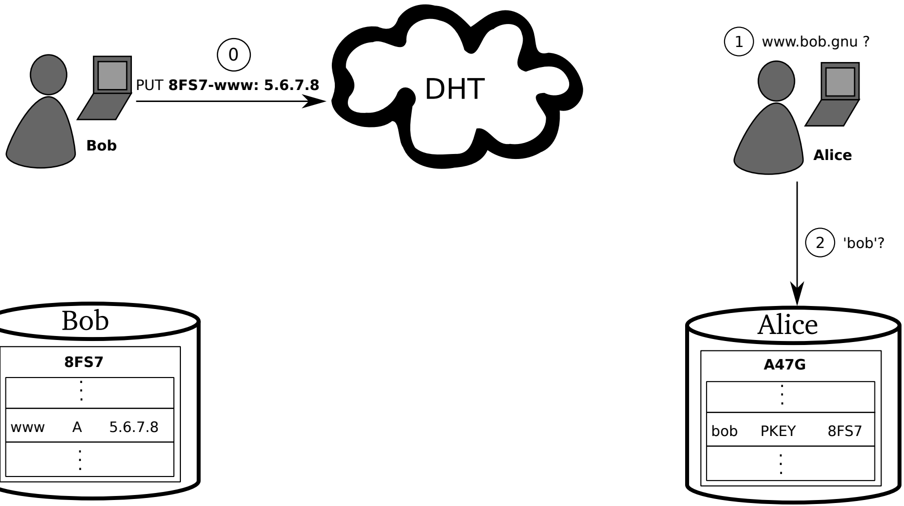
Name Resolution



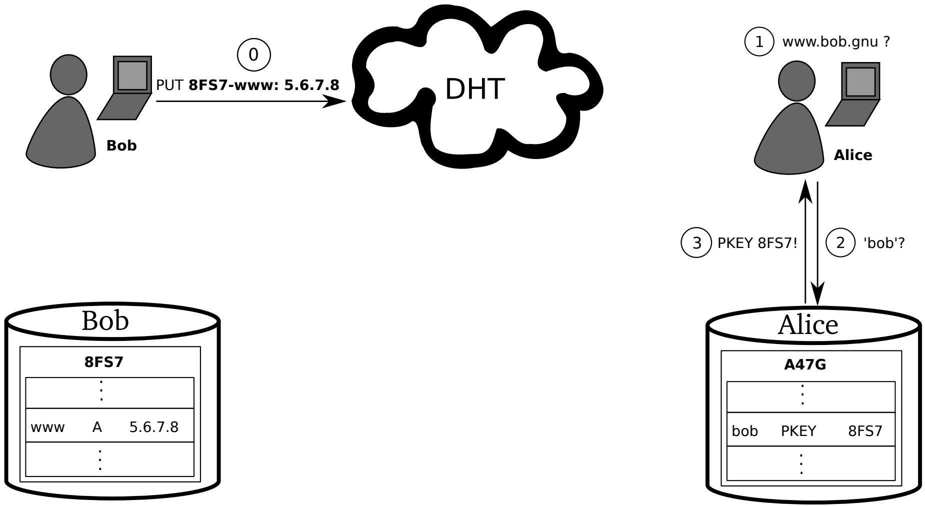
Name Resolution



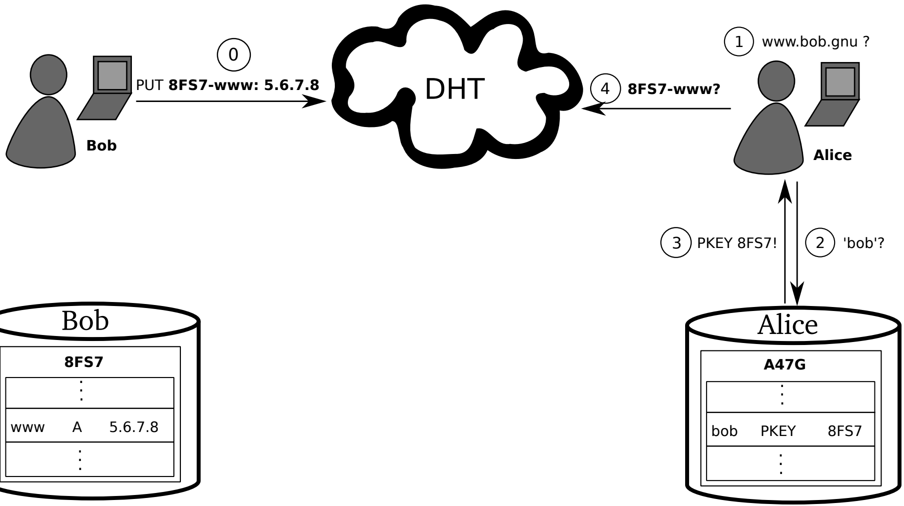
Name Resolution



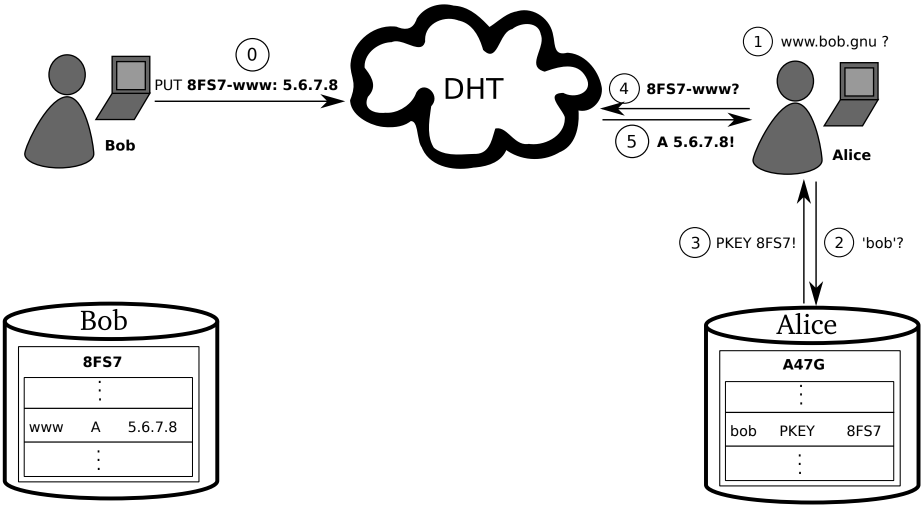
Name Resolution



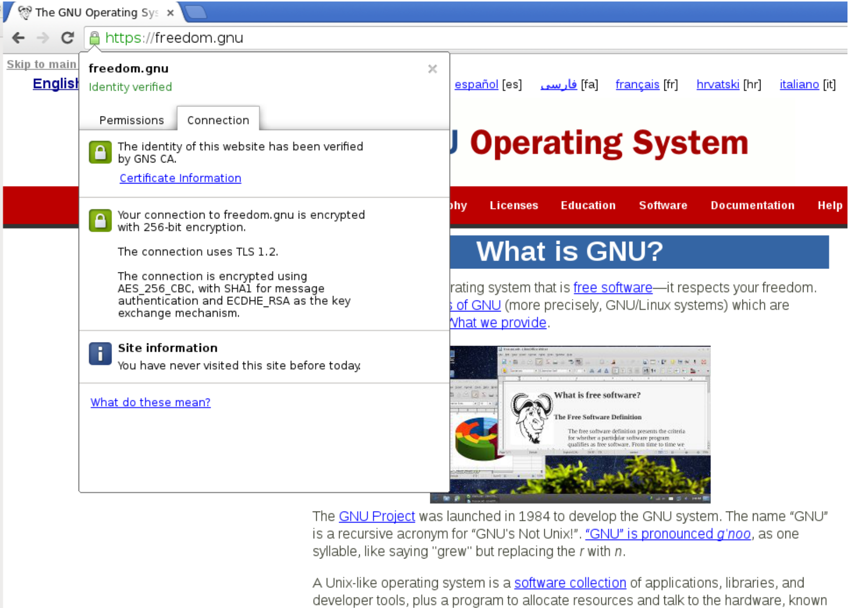
Name Resolution



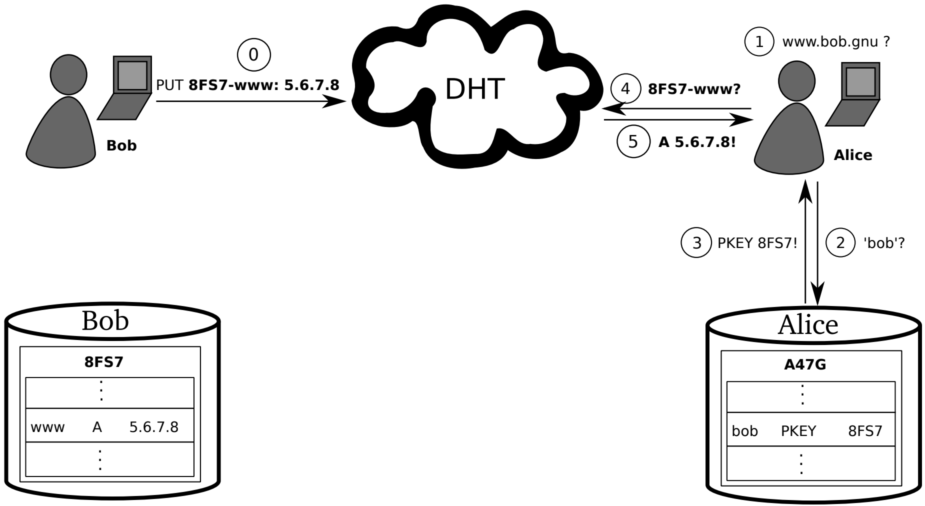
Name Resolution



GNS as PKI (via DANE/TLSA)



Privacy Issue: DHT



Query Privacy: Terminology

G	generator in ECC curve, a point
o	size of ECC group, $o := \|G\|$, o prime
x	private ECC key of zone ($x \in \mathbb{Z}_o$)
P	public key of zone, a point $P := xG$
l	label for record in a zone ($l \in \mathbb{Z}_o$)
$R_{P,l}$	set of records for label l in zone P
$q_{P,l}$	query hash (hash code for DHT lookup)
$B_{P,l}$	block with encrypted information for label l in zone P published in the DHT under $q_{P,l}$

Query Privacy: Cryptography

Publishing records $R_{P,l}$ as $B_{P,l}$ under key $q_{P,l}$

$$h := H(l, P) \tag{10}$$

$$d := h \cdot x \mod o \tag{11}$$

$$B_{P,l} := S_d(E_{HKDF(l,P)}(R_{P,l})), dG \tag{12}$$

$$q_{P,l} := H(dG) \tag{13}$$

Query Privacy: Cryptography

Publishing records $R_{P,l}$ as $B_{P,l}$ under key $q_{P,l}$

$$h := H(l, P) \tag{10}$$
$$d := h \cdot x \mod o \tag{11}$$
$$B_{P,l} := S_d(E_{HKDF(l,P)}(R_{P,l})), dG \tag{12}$$
$$q_{P,l} := H(dG) \tag{13}$$

Searching for records under label l in zone P

$$h := H(l, P) \tag{14}$$
$$q_{P,l} := H(hP) = H(hxG) = H(dG) \Rightarrow \texttt{obtain } B_{P,l} \tag{15}$$
$$R_{P,l} = D_{HKDF(l,P)}(B_{P,l}) \tag{16}$$

Using cryptographic identifiers

- Zone are identified by a public key
- "alice.bob.*PUBLIC-KEY*" is perfectly legal in GNS!

$\Rightarrow$ Globally unique identifiers

Key Revocation

- Revocation message signed with private key (ECDSA)
- Flooded on all links in P2P overlay, stored forever
- Efficient set reconciliation used when peers connect
- Expensive proof-of-work used to limit DoS-potential
- Proof-of-work can be calculated ahead of time
- Revocation messages can be stored off-line if desired

Efficient Set Union

(based on "What's the difference? Efficient Set Reconciliation without Prior Context", Eppstein et al., SIGCOMM'11)

- Alice and Bob have sets A and B
- The sets are very large
- ...but their symmetric difference $\delta = |(A - B) \cup (B - A)|$ is small
- Now Alice wants to know $B - A$ (the elements she's missing)
- ...and Bob $A - B$ (the elements he's missing)
- How can Alice and Bob do this efficiently?
  - w.r.t. communication and computation

Bad Solution

- Naive approach: Alice sends A to Bob, Bob sends $B - A$ back to Alice
- . . . and vice versa.

- Communication cost: $O(|A| + |B|)$:(
- Ideally, we want to do it in $O(\delta)$.
- First improvement: Don't send elements of A and B, but send/request hashes. Still does not improve complexity :(

- We need some more fancy data structure!

Bloom Filters

Constant size data structure that "summarizes" a set.

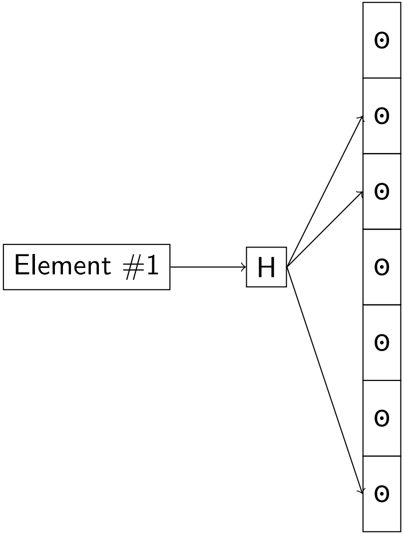
Operations:

$d = NewBF(size)$ Create a new, empty bloom filter.

$Insert(d, e)$ Insert element e into the BF d.

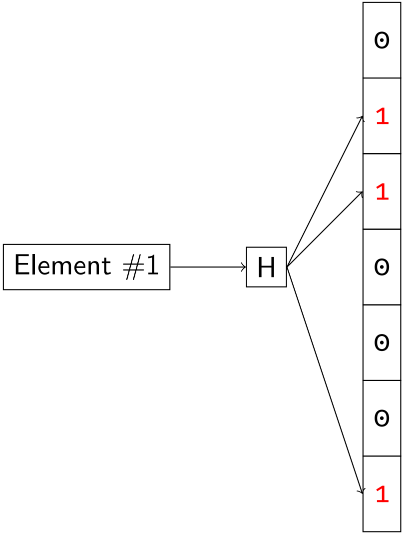
$b = Contains(d, e)$ Check if BF d contains element e.
$b \in \{$"Definitely not in set", "Probably in set"$\}$

BF: Insert



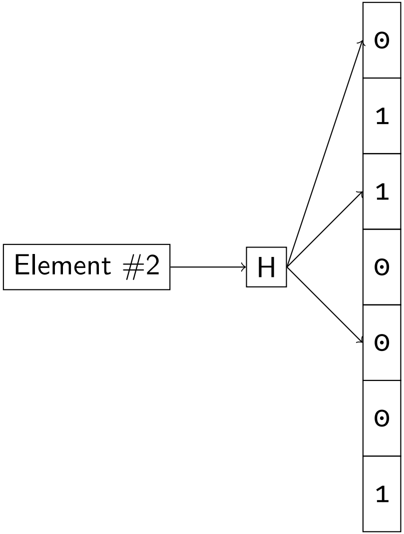
$$H(\text{Element } \#1) = (2, 3, 7)$$

BF: Insert



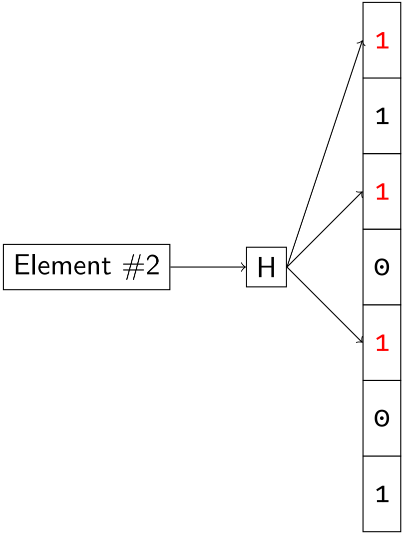
$$H(\text{Element } \#1) = (2, 3, 7)$$

BF: Insert



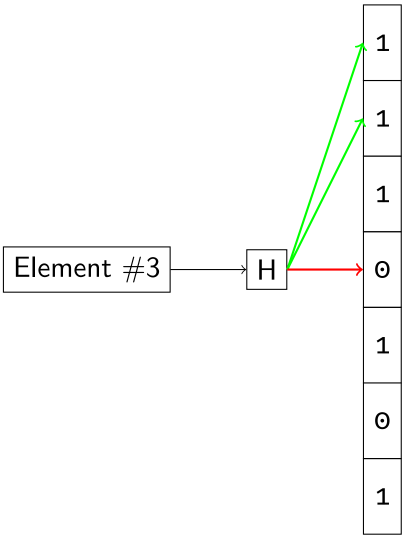
$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H(\text{Element } \#2) = (1, 3, 5)$$

BF: Insert



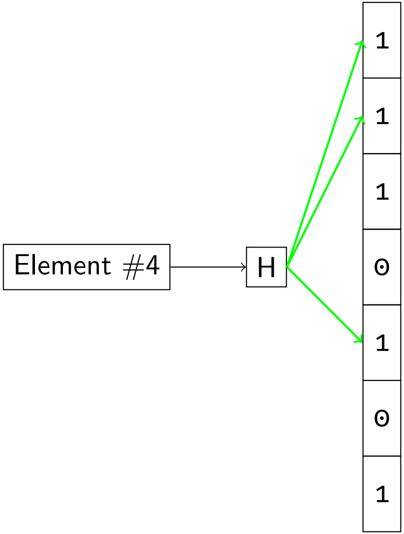
$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H(\text{Element } \#2) = (1, 3, 5)$$

BF: Membership Test



$H(\text{Element } \#1) = (2, 3, 7)$
$H(\text{Element } \#2) = (1, 3, 5)$

BF: Membership Test (false positive)



$H(\text{Element } \#1) = (2, 3, 7)$
$H(\text{Element } \#2) = (1, 3, 5)$

Counting Bloom Filters

BF where buckets hold a **positive integer**.

Additional Operation:

$Remove(d, e)$ Remove element from the CBF d.

$\Rightarrow$ False negatives when removing a non-existing element.

Invertible Bloom Filters

Similar to CBF, but

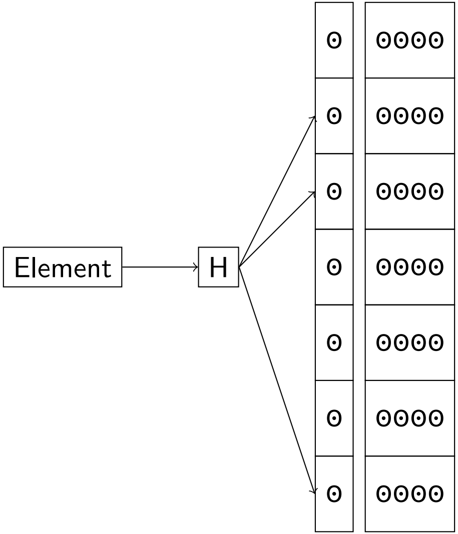
- Allow **negative counts**
- Additionaly store **(XOR-)sum of hashes** in buckets.

Additional Operations:

$(e, r) = \mathit{Extract}(d)$ Extract an element (e) from the IBF d, with result code $r \in \{\mathit{left}, \mathit{right}, \mathit{done}, \mathit{fail}\}$

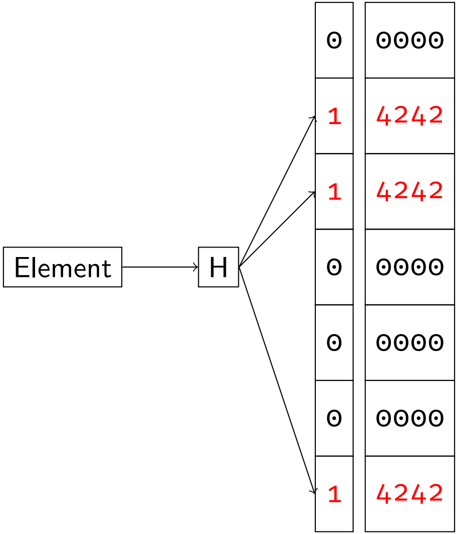
$d' = \mathit{SymDiff}(d_1, d_2)$ Create an IBF that represents the symmetric difference of d_1 and d_2.

IBF: Insert



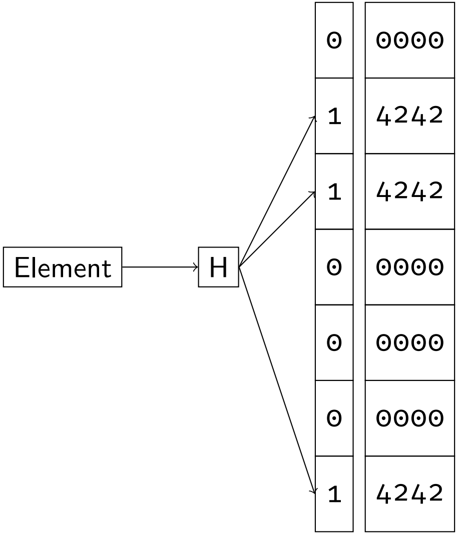
$H(\text{Element } \#1) = (2, 3, 7)$
$H'(\text{Element } \#1) = 4242$

IBF: Insert



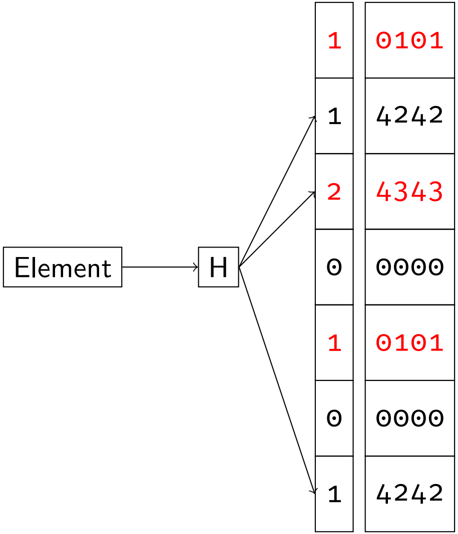
$H(\text{Element }\#1) = (2, 3, 7)$
$H'(\text{Element }\#1) = 4242$

IBF: Insert



$$H(\text{Element }\#1) = (2,3,7)$$
$$H'(\text{Element }\#1) = 4242$$
$$H(\text{Element }\#2) = (1,3,5)$$
$$H'(\text{Element }\#2) = 0101$$

IBF: Insert



$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H'(\text{Element } \#1) = 4242$$
$$H(\text{Element } \#2) = (1, 3, 5)$$
$$H'(\text{Element } \#2) = 0101$$

IBF: Extract

1	0101	pure bucket
1	4242	
2	4343	
0	0000	
1	0101	
0	0000	
1	4242	

- Pure bucket $\Rightarrow$ extractable element hash
- Extraction $\Rightarrow$ more pure buckets (hopefully/probably)
- Less elements $\Rightarrow$ more chance for pure buckets

Symmetric Difference

We can directly compute the symmetric difference without extraction.

- Subtract counts
- XOR hashes

The Set Union Protocol

1. $\Rightarrow$ Create IBFs
2. Compute SymDiff
3. Extract element hashes

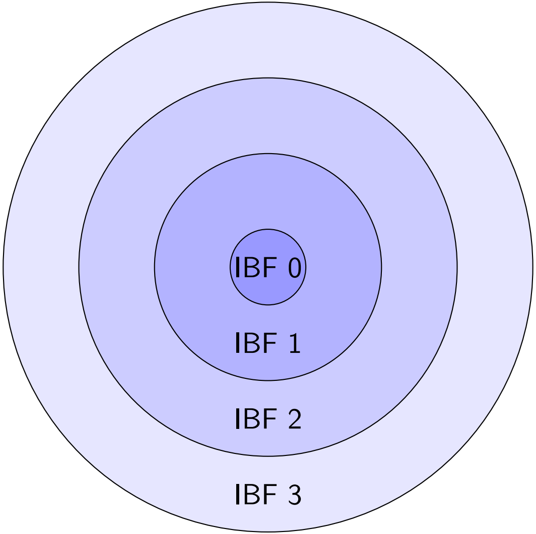
- Amount of communication and computation only depends on δ, not $|A| + |B|$:)
- How do we choose the initial size of the IBF?
- $\Rightarrow$ Do difference estimation first!

Difference Estimation

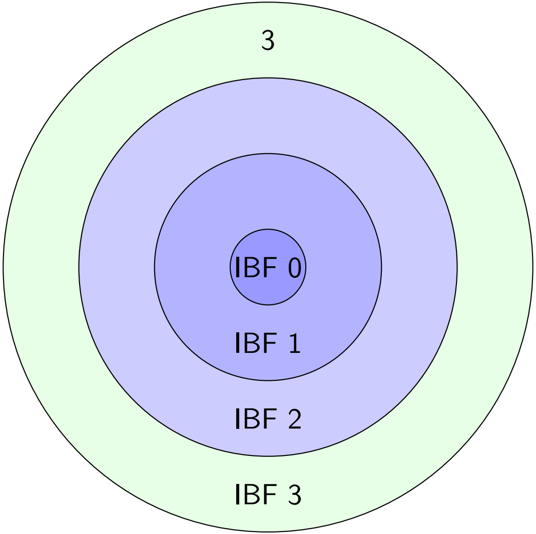
- We need an estimator that's accurate for small differences
- Turns out we can re-use IBFs for difference estimation:

1. Alice and Bob create fixed number of constant-size IBFs by sampling their set. The collection of IBFs is called a Strata Estimator (SE).
   - Stratum 1 contains $1/2$ of all elements
   - Stratum 2 contains $1/4$ of all elements
   - Stratum n contains $1/(2^n)$ all elements
2. Alice receives Bob's strata estimator
3. Alice computes $SE_{diff} = SymDiff(SE_{\mathrm{Alice}}, SE_{\mathrm{Bob}})$
   - by pair-wise *SymDiff* of all IBFs in the SE
4. Alice estimates the size of SE_{diff}.

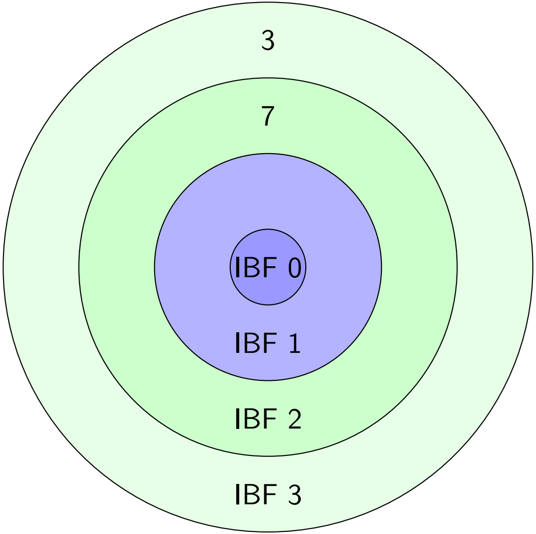
Strata Estimator



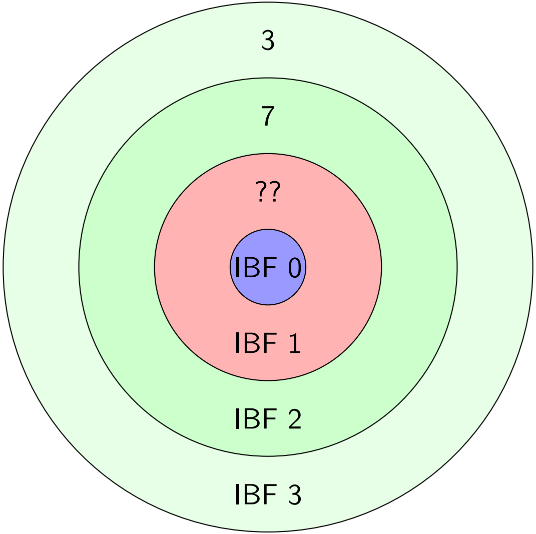
Strata Estimator



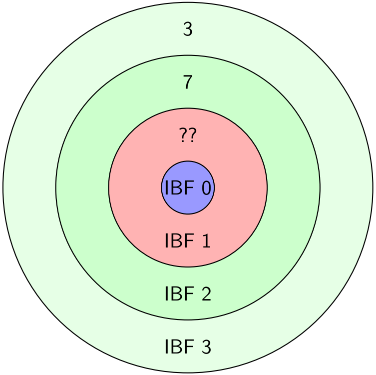
Strata Estimator



Strata Estimator



Estimation



Estimate as $(3 + 7) \cdot 2^4$.
(Number of extracted hashes scaled by expected number of elements in the remaining IBFs)

The Complete Protocol

1. Alice sends SE_{Alice} to Bob
2. Bob estimates the set difference δ
3. Bob computes IBF_{Bob} with size δ and sends it to Alice
4. Alice computes IBF_{Alice}
5. Alice computes $IBF_{diff} = SymDiff(IBF_{\text{Alice}}, IBF_{\text{Bob}})$
6. Alice extracts element hashes from IBF_{diff}.
   - $b = left \Rightarrow$ Send element to to Bob
   - $b = right \Rightarrow$ Send element request to to Bob
   - $b = fail \Rightarrow$ Send larger IBF (double the size) to Bob, go to (3.) with switched roles
   - $b = done \Rightarrow$ We're done ...

Summary

- Interoperable with DNS
- Globally unique identifiers with ".PUBLIC-KEY"
- Delegation allows using zones of other users
- Trust paths explicit, trust agility
- Simplified key exchange compared to Web-of-Trust
- Privacy-enhanced queries, censorship-resistant
- Reliable revocation

Case study: GNS

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"The GNU Name System (GNS) establishes a new name system using cryptography where zone data, queries and replies are private. The use of a distributed hash table (DHT) implies that resolution costs are comparable to those of DNS. However, states and ISPs cannot monitor or block queries, limiting their ability to protect the public from malicious Web sites. Names are not globally unique, allowing multiple anonymous users to lay claim to the same name. However, the system includes some well-known mappings by default, which users are unlikely to change. Trademarks, copyrights anti-fraud or anti-terrorism judgements can only be enforced against those well-known mappings, which users are able to bypass."*

Discuss virtues and vices affected.

Break

Namecoin

Let's just put the records into the Blockchain!

Namecoin

Let's just put the records into the Blockchain!

Or rather, put the public key of the owner and signed updates into it.

Namecoin

Let's just put the records into the Blockchain!

Or rather, put the public key of the owner and signed updates into it.

And let's have some expiration rules.

Case study: Namecoin

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"Namecoin establishes a new name system on the blockchain (where thus zone data is also public), but where public authorities cannot block information. Queries are performed against a local copy of the blockchain and thus also private. There is no WHOIS, so the owner of a name can also be anonymous. However, Namecoin uses much more bandwidth and energy as blockchain payments are used for registration and name resolution. Names are registered on a first-come, first-served basis. Trademarks, copyrights anti-fraud or anti-terrorism judgements cannot be used to force owners of names to relinquish names."*

Discuss virtues and vices affected.

Break

Ethereum Name System[4]

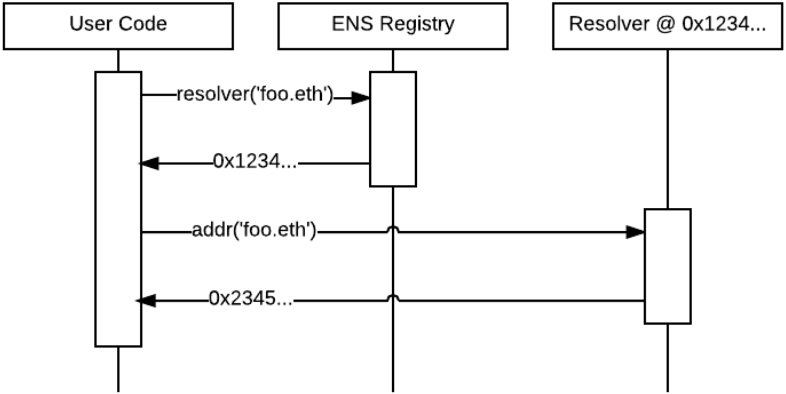
Let's have a smart contract in the Blockchain manage naming!

Blockchain contains smart contract and data who controls which name.

Contract allocates names under `.eth` using auctions.

[4] https://ens.domains/

Ethereum Name System[5]



[5]https://ens.domains/

Privacy summary

Method	Defense against MiTM	Zone privacy	Privacy vs. network	Privacy vs. operator	Traffic amplification resistance	Censorship resistance	Ease of migration
DNS	✗	✓	✗	✗	✗	✗	✓
DNSSEC	✓	✗	✗	✗	✗	✗	✗*
DNSCurve	✓	✓	✓	✗	✓	✗	✗
DNS-over-TLS	✓	n/a	✓	✗	✓	✗	✗
Namecoin	✓	✗	✓	✓	✓	✓	✗
RAINS	✓	✗	✓	✗	✓	✗	✗
GNS	✓	✓	✓	✓	✓	✓	✗

*EDNS0

Key management summary

	Suitable for personal use	Memorable	Decentralised	Modern cryptography	Understandable	Exposes metadata	Transitive
DNS	✗	✓	✗	✗	✗	✗	✓
DNSSEC	✗	✓	✗	✗	✗	✗	✓
DNSCurve	✗	✓	✗	✓	✗	✗	✓
DNS-over-TLS	✗	✓	✗	✗	✗	✗	✓
TLS-X.509	✗	✓	✗	✗	✗	✗	✓
Web of Trust	✓	✗	✓	✗	✗	✗	✓
TOFU	✓	✗	✓		✓	✓	✗
Namecoin	✗	✓	✗	✓	✓	✗	✓
RAINS	✗	✓	✗	✓	✓	✗	✓
GNS	✓	✓	✓	✓	✓	✓	✓

Possible Future Work (Project 2, BS thesis)

- Implement & evaluate **bounded** Eppstein set reconciliation
- Implement Fog-of-Trust (ideally in Rust)

Conclusion

DNS	globalist
DNSSEC	authoritarian
Namecoin	libertarian (US)
RAINS	nationalist
GNS	anarchist

In which world do you want to live?

Exercise

```
# apt-get install git autoconf automake autopoint gettext
# apt-get install libunistring-dev libgnutls28-dev
# apt-get install openssl gnutls-bin libtool libltdl
# apt-get install libcurl-gnutls-dev libidn11-dev
# apt-get install libsqlite3-dev
$ git clone git://gnunet.org/libmicrohttpd
$ git clone git://gnunet.org/gnunet
$ git clone git://gnunet.org/gnunet-gtk
$ for n in libmicrohttpd gnunet gnunet-gtk do;
    cd $n ; ./bootstrap ; ./configure --prefix=$HOME ...
    make install
    cd ..
  done
```

Exercise

```
$ gnunet-setup  # enable TCP transport only
$ gnunet-arm -s # launch peer
$ gnunet-namestore-gtk # configure your GNS zone
$ gnunet-gns # command-line resolution
$ gnunet-gns-proxy # launch SOCKS proxy
$ firefox # configure browser to use proxy
```

References

Alexandra Dirksen.
A blockchain picture book.
https://media.ccc.de/v/35c3-9573-a˙blockchain˙picture˙book), 12 2018.

Matthias Wachs, Martin Schanzenbach, and Christian Grothoff.
A censorship-resistant, privacy-enhancing and fully decentralized name system.
In *13th International Conference on Cryptology and Network Security (CANS 2014)*, pages 127–142, 2014.