Keeping Authorities "Honest or Bust" with Decentralized Witness Cosigning

Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky
– **Yale University**

Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoff, **Bryan Ford**
– **Swiss Federal Institute of Technology Lausanne (EPFL)**

IEEE Security & Privacy – May 24, 2016

We depend on many authorities

Conceptually simple but **security-critical** services

- Time Services (NTP)

- Digital Notaries

eNotary

e-timestamp

- Naming Authorites

DNSSEC SECURED by Afilias

SECURE64

- Certificate Authorities

VeriSign

thawte it's a trust thing

- Randomness Authorities (e.g., Lotteries)

- Software Update Services

GET IT ON Google play

But are authorities trustworthy?

WIRED Hack Obtains 9 Bogus Certificates for Prominent ...

HACK OBTAINS 9 BOGUS CERTIFICATES FOR PROMINENT WEBSITES; TRACED TO IRAN

But are authorities trustworthy?

CYBER CRIME SCAMS AND FRAUD

This Dude Hacked Lottery Computers To Win $14.3M Jackpot In U.S.

By Waqas on April 14, 2015 *Email* *@hackread*

But are authorities trustworthy?

threatpost CATEGORIES FEATURED PODCASTS VIDEOS

10/08/15 5:54 Advanced notice: Security updates for Adobe Acrobat and Reader are due on Patch Tuesday: https://t.co/QLqnpulr0A

Welcome > Blog Home > Cryptography > D-Link Accidentally Leaks Private Code-Signing Keys

by Michael Mimoso Follow @mike_mimoso September 18, 2015 , 10:21 am

But are authorities trustworthy?

New attacks on Network Time Protocol can defeat HTTPS and create chaos

Exploits can be used to snoop on encrypted traffic and cause debilitating outages.

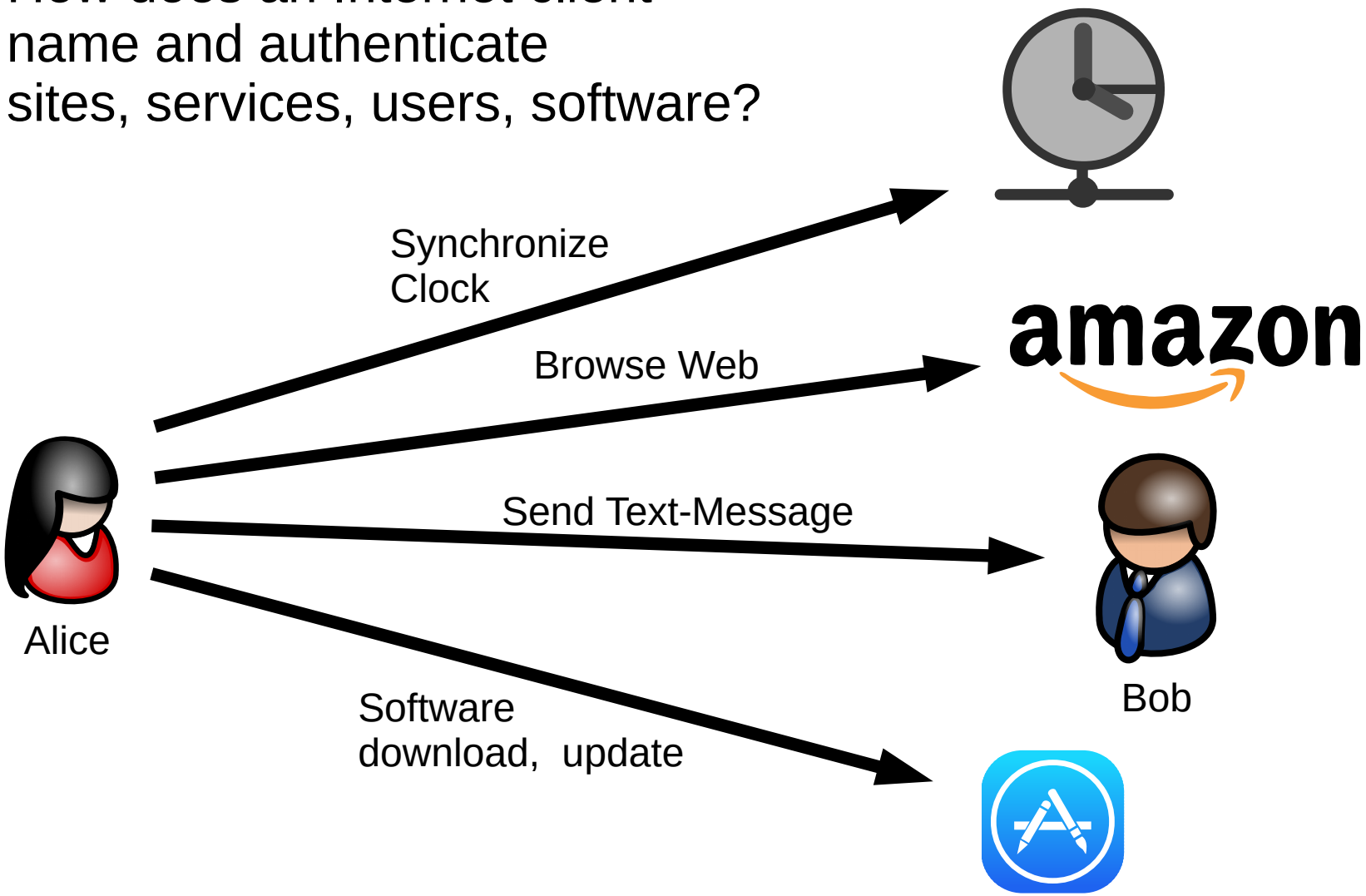
by **Dan Goodin** - Oct 22, 2015 12:07am CEST

Talk Outline

- **The trouble with trusting authorities**
- Grand challenge: decentralize the authorities!
- Baby step: decentralized witness cosigning
- CoSi: scalable collective Schnorr/Ed25519 signatures
- Experimental evaluation: scalability, signature size
- Comparison with prior transparency approaches
- Status, future work, and conclusions

Deep Dependence on Authorities

How does an Internet client
name and authenticate
sites, services, users, software?



Deep Dependence on Authorities

Respect my Authoritah!

?

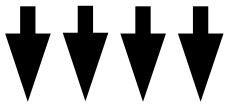
amazon

Alice

What is:
- The current time?
- Amazon's SSL public key?
- Bob's IM public key?
- Latest version of App?

Bob

Authorities Make & Sign Statements



"The time is 3PM."

amazon

"Amazon's public key is X."

Alice

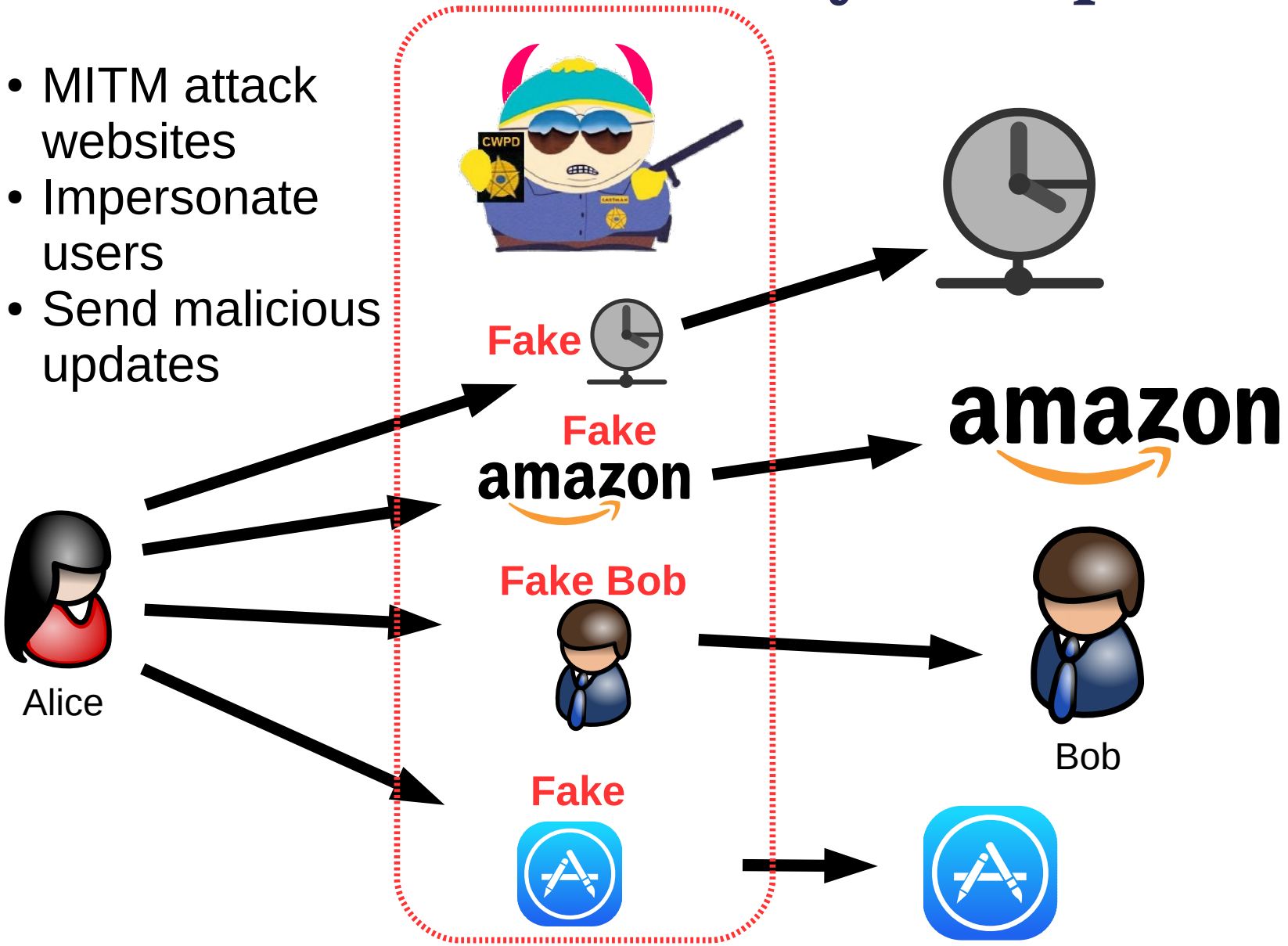
"Bob's public key is Y."

Bob

"The hash of latest iOS is Z."

Problem #1: Authority Compromise

- MITM attack websites
- Impersonate users
- Send malicious updates



Problem #2: Weak Links

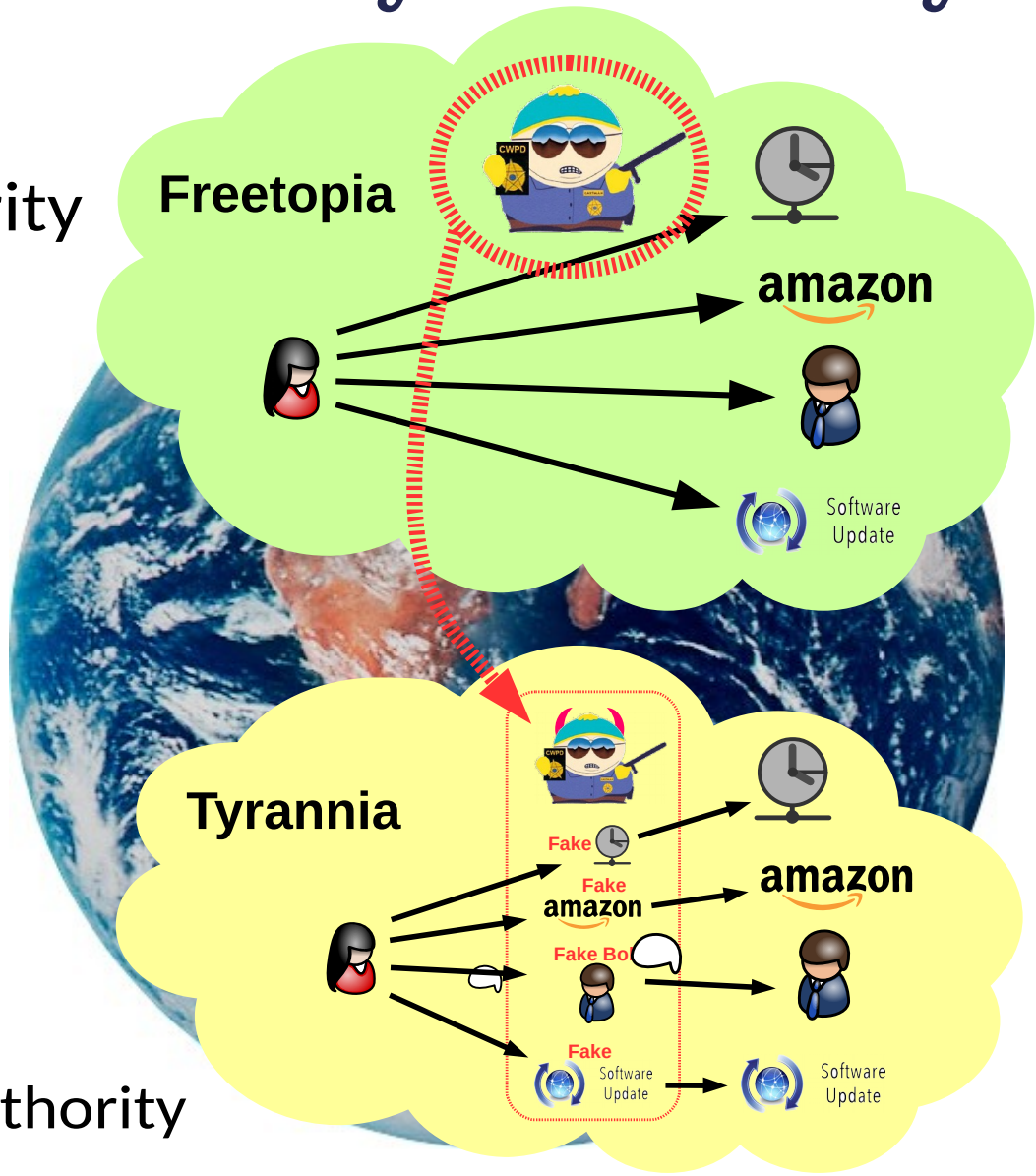
- Clients often depend on **many** authorities:
  e.g., hundreds of CAs trusted by web browsers
- Any CA can issue cert
  for any domain name

Attacker often needs to **compromise only one**

- **Weakest-link** security
- @#$% happens
  DigiNotar,
  Comodo,
  CNNIC/MCS

Problem #3: Secret Key Portability

- Attacker need not compromise authority "in-place"
- Instead steal the authority's **secret key**
  - Use it to create an "evil twin" on attacker's turf
  - Avoid detection by confining use to specific targets
  - Block targets from reporting to real authority



Problem #4: Everybody Wants In

Hackers, organized crime, governments…

The Register®
Biting the hand that feeds IT

Security

Is Kazakhstan about to man-in-the-middle diddle all of its internet traffic with dodgy root certs?

Come on, guys. Don't go giving the Russians any ideas

Problem #4: Everybody Wants In

Hackers, organized crime, governments…

Talk Outline

- The trouble with trusting authorities
- **Grand challenge: decentralize the authorities!**
- Baby step: decentralized witness cosigning
- CoSi: scalable collective Schnorr/Ed25519 signatures
- Experimental evaluation: scalability, signature size
- Comparison with prior transparency approaches
- Status, future work, and conclusions

What To Do?

We have to assume that no individual…

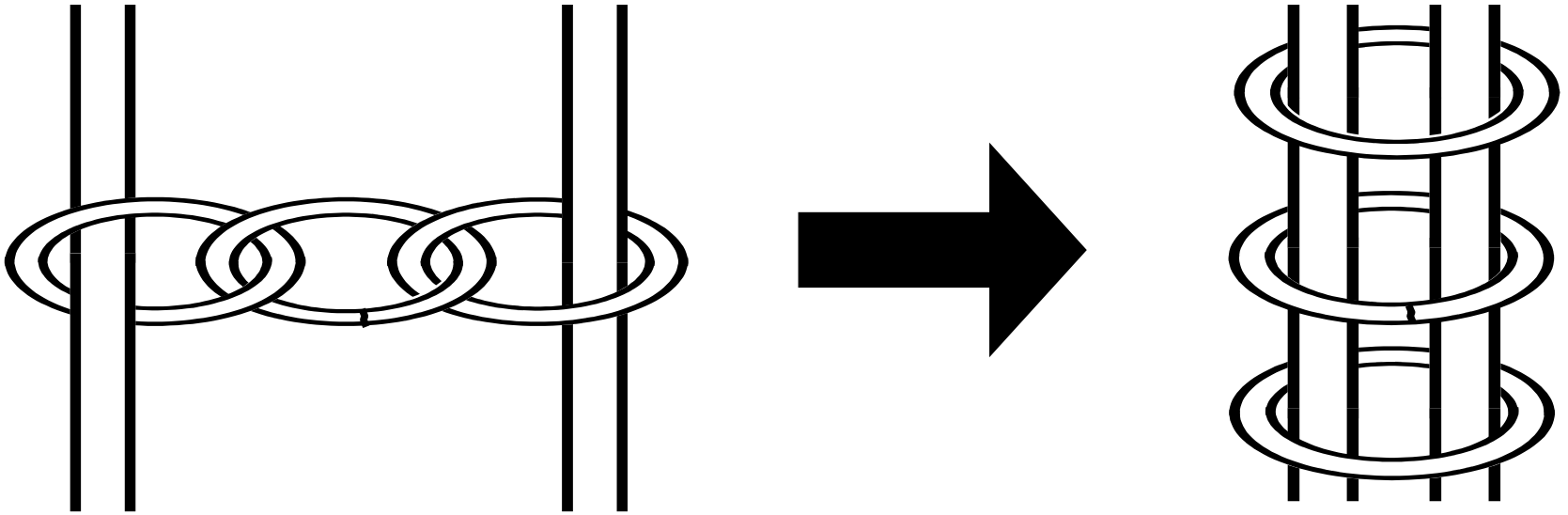
- Hardware platform
- Software system
- System/network administrator
- Authoritative organization

…is invulnerable to compromise (or coercion)

Decentralize the Authorities!

Split trust across independent parties

- So system resists compromise by individuals
- From **weakest-link** to **strongest-link** security
- Decentralized resistance to failure, coercion



Example: Tor Directory Authority

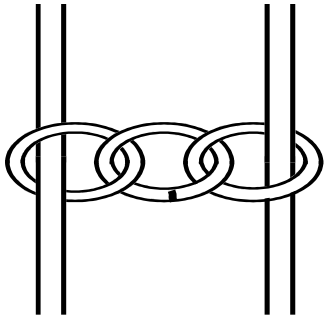
Split across ~10 servers – **but is this enough?**

- Could attacker hack or coerce ~5 operators?

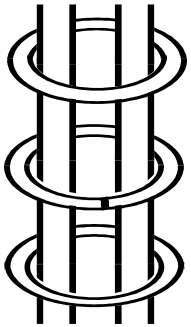
(image credit: Jordan Wright)

Trust-splitting needs to scale

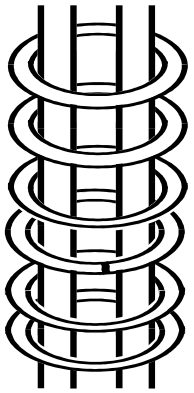
Weakest-link:
T = 1



Strongest-link:
T = 2-10



Collective authorities:
T = 100s,1000s



Trust-splitting needs to scale

Must incorporate **all diversity that makes sense**

- Not just ~10 parties “picked by someone”

Could we decentralize…

- **TLS certificate validation and signing**
  across the hundreds of certificate authorities?
- **DNSSEC root zone maintenance and signing**
  across the 1000+ worldwide TLD operators?
- **A national cryptocurrency**
  across the thousands of US national banks?

Make overall security **grow** as scale increase?

Talk Outline

- The trouble with trusting authorities
- Grand challenge: decentralize the authorities!
- **Baby step: decentralized witness cosigning**
- CoSi: scalable collective Schnorr/Ed25519 signatures
- Experimental evaluation: scalability, signature size
- Comparison with prior transparency approaches
- Status, future work, and conclusions

Not Gonna Happen Overnight…

A First Step: **Transparency**

More readily achievable near-term

- Who watches the watchers?
  Public **witnesses**!

Respect my Authoritah!

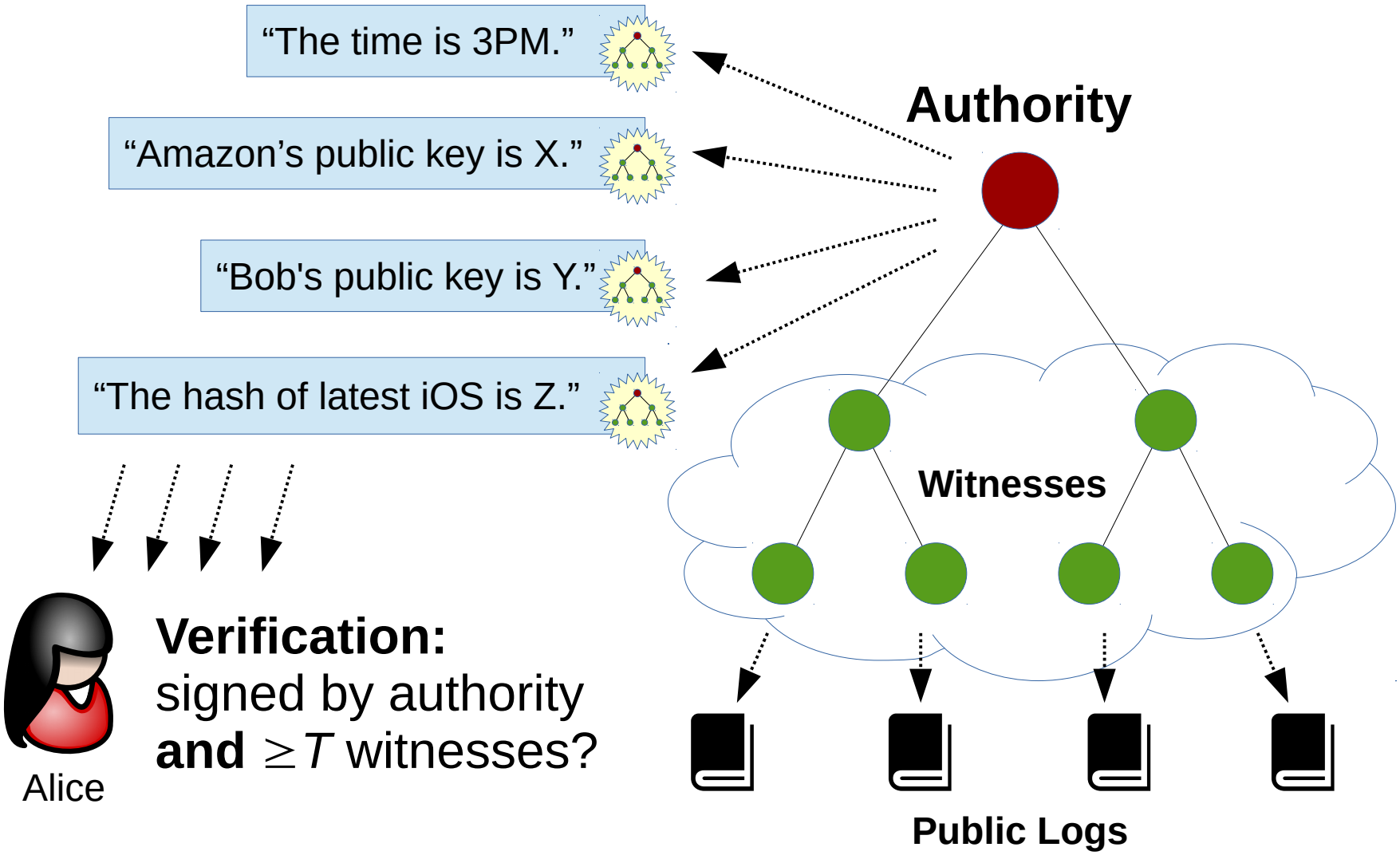
Ensure that **any** authoritative statement:

- Is exposed to **public scrutiny**
- Conforms to **checkable standards**

before clients will accept statement

Key: **practical to "retrofit" existing authorities**

Decentralized Witness Cosigning



Is the Signed Statement "Good"?

In general, **witnesses don't (and can't) know for sure**

- Does public key X really belong to Bob?
- Does software image Y have a secret backdoor?

But witnesses can still ensure **all signatures are public**

- If authority coerced or its keys used to produce bad statement, attacker can't ensure its secrecy
  - Backdoors possible but must "hide in plain sight"
- Creates "Ulysses Pact" deterrent against coercion
  - "the point…is to keep governments from even trying to put secret pressure on tech companies, because the system is set up so that the secret immediately gets out" - Cory Doctorow, 10-March-2016

Talk Outline

- The trouble with trusting authorities
- Grand challenge: decentralize the authorities!
- Baby step: decentralized witness cosigning
- **CoSi: scalable collective Schnorr/Ed25519 signatures**
- Experimental evaluation: scalability, signature size
- Comparison with prior transparency approaches
- Status, future work, and conclusions

Setup: Keypairs and CoSi Groups

Individual Keypairs:

Standard Schnorr (Ed25519)

- Private key: k
- Public key: $K = g^k$

CoSi group:

List of public keys

- $K_1, K_2, \ldots, K_N$

Assumptions:

- Verifier has full list
  - (nonessential)
- All keys self-signed
  - (important to avoid related-key attacks)

Schnorr Signature

- Generator g of prime order q group
- Public/private key pair: $(K=g^k, k)$

	Signer		Verifier
Commitment	$V=g^v$	→	V
Challenge	c	←	$c = H(M \mid V)$
Response	$r = (v - kc)$	→	r

Signature on M: (c, r)

Commitment recovery	$V' = g^r K^c = g^{v-kc} g^{kc} = g^v = V$
Challenge recovery	$c' = H(M \mid V')$
Decision	$c' = c$? ✓

Schnorr Multisignature

- Key pairs: $(K_1=g^{k_1}, k_1)$ and $(K_2=g^{k_2}, k_2)$

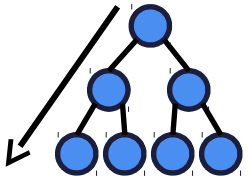
	Signer 1		Signer 2	Verifier		
Commitment	$V_1=g^{v_1}$		$V_2=g^{v_2}$ →	V_1	V_2	$V=V_1*V_2$
Challenge	c	c	←	$c = H(M\|V_1)$	$c = H(M\|V)$	
Response	$r_1 = (v_1 - k_1c)$		$r_2 = (v_2 - k_2c)$ →	r_1	r_2	$r=r_1+r_2$

Signature on M: (c, r_1) Same signature!

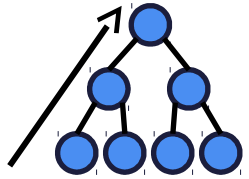
Commitment recovery	Same verification!	$V' = g^rK^c$	$K=K_1*K_2$
Challenge recovery	Done once!	$c' = H(M\|V')$	
Decision		$c' = c$?	

CoSi Protocol Signing Rounds

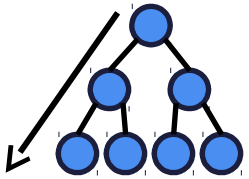
1. Announcement Phase



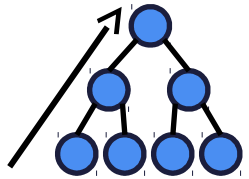
2. Commitment Phase



3. Challenge Phase



4. Response Phase

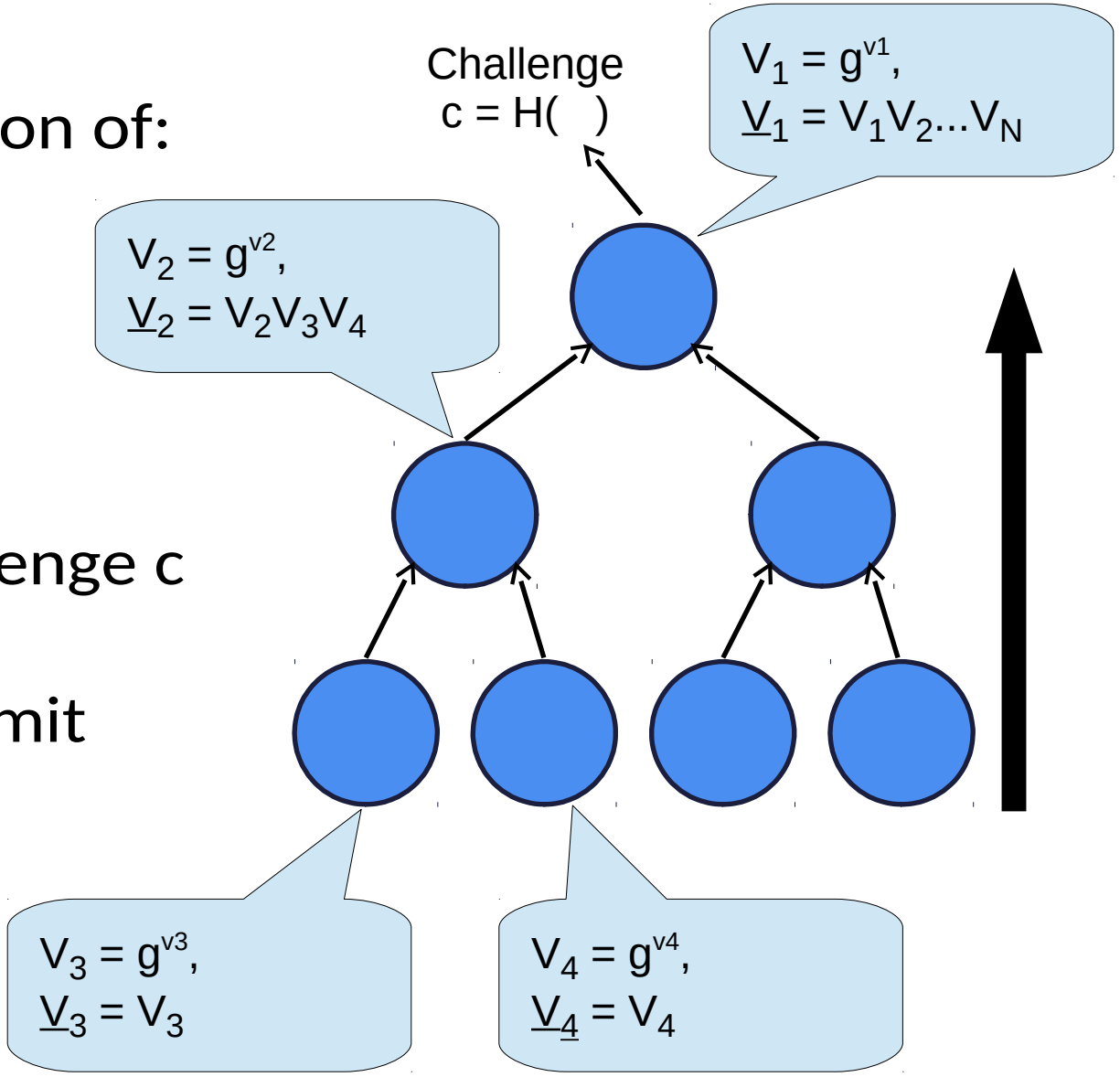


CoSi Commit Phase

Tree computation of:

- Commits V_i
- Aggregate commits $\underline{V}_i$

Collective challenge c is hash of aggregate commit



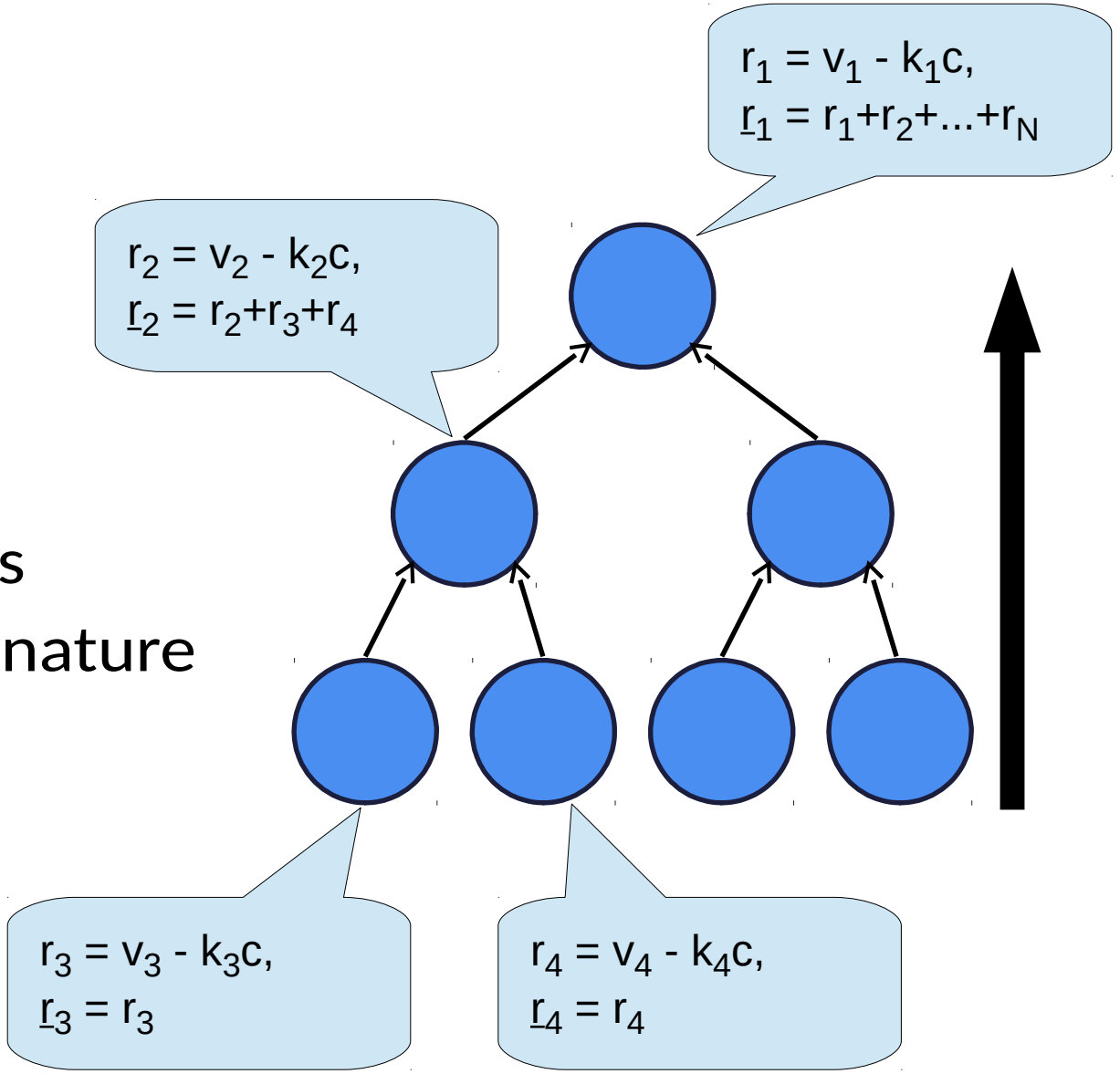
CoSi Response Phase

Compute

- Responses r_i
- Aggregate responses $\underline{r}_i$

Each $(c,\underline{r}_i)$ forms valid **partial** signature

$(c,\underline{r}_1)$ forms **complete** signature



Unavailable Witness Servers

Assume server failures are **rare** but **non-negligible**

- *Persistently bad* servers get administratively booted

Exceptions: If a server A is down, proceed anyway

- Modified collective key: $K' = K * K^{-1}_A$
- Modified commitment: $V' = V * V^{-1}_A$
- Modified response: $r' = r - r_A$

Verification: CoSi signature includes roll-call bit-vector

- Enables verifier to recompute modified public key K'
- Can use **any** criteria to decide if "too many" missing

Variations (see paper for details)

- Complex/contextual verification predicates
  - Witness subgroups, weights, expressions, ...
- Minimizing cothority certificate size
  - Via Merkle key-trees
- Tolerating network churn
  - Via binomial swap forests (Cappos, San Fermin)
- Tolerating cosigner churn
  - Avoiding restarts via commit trees
- Single-pass CoSi for asynchronous networks
  - Via BLS signatures, opportunistic signature combining

Talk Outline

- The trouble with trusting authorities
- Grand challenge: decentralize the authorities!
- Baby step: decentralized witness cosigning
- CoSi: scalable collective Schnorr/Ed25519 signatures
- **Experimental evaluation: scalability, signature size**
- Comparison with prior transparency approaches
- Status, future work, and conclusions

Experimental Evaluation

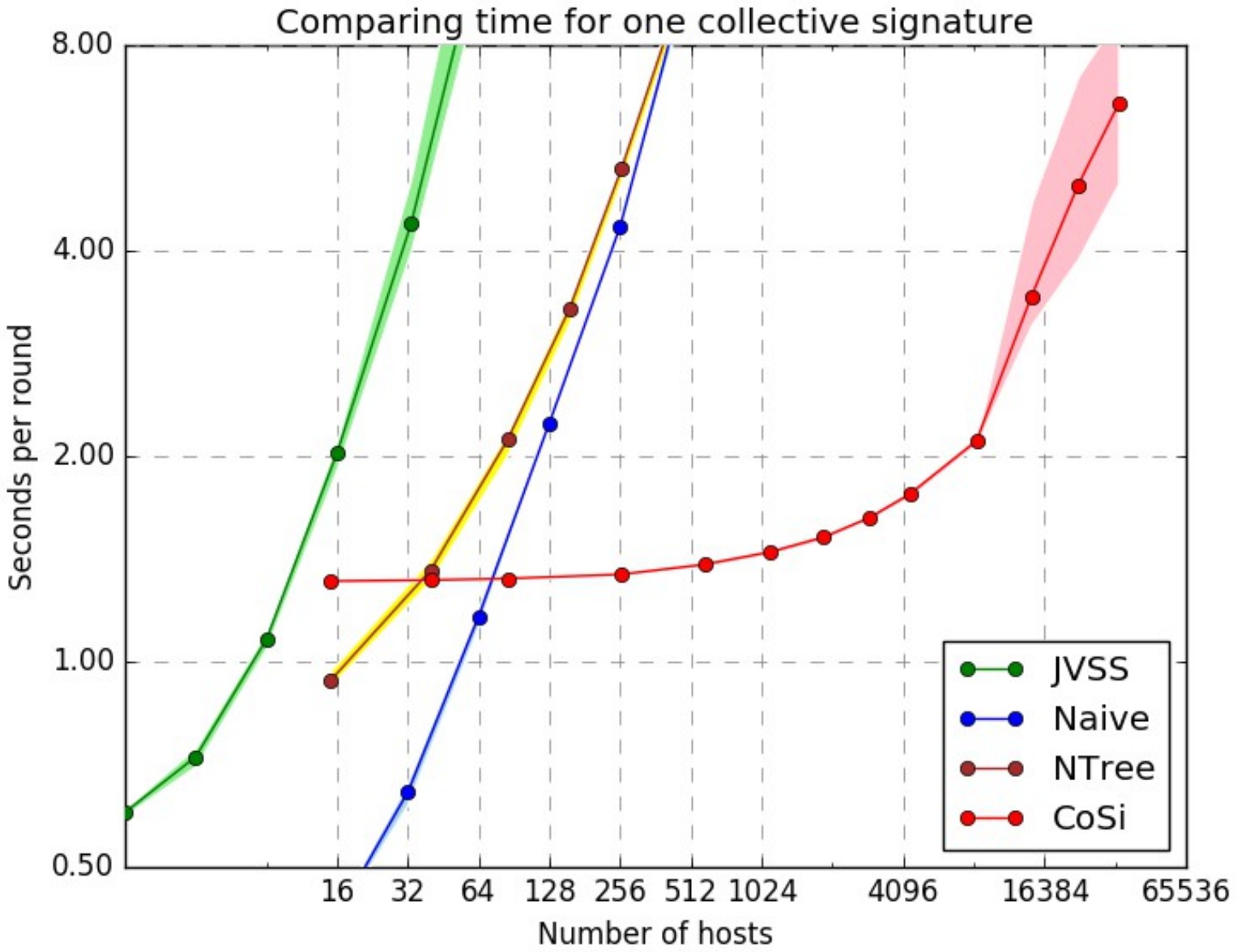
Experiments run on DeterLab network testbed

- Up to **32,768** virtual CoSi witnesses
- Multiplexed atop up to 64 physical machines
  - introduces oversubscription overhead, unfortunately
  - Conservative results, likely worse than "real" deployment
- Impose 200ms roundtrip latencies between all servers
  - to simulate **globally-distributed** witness group

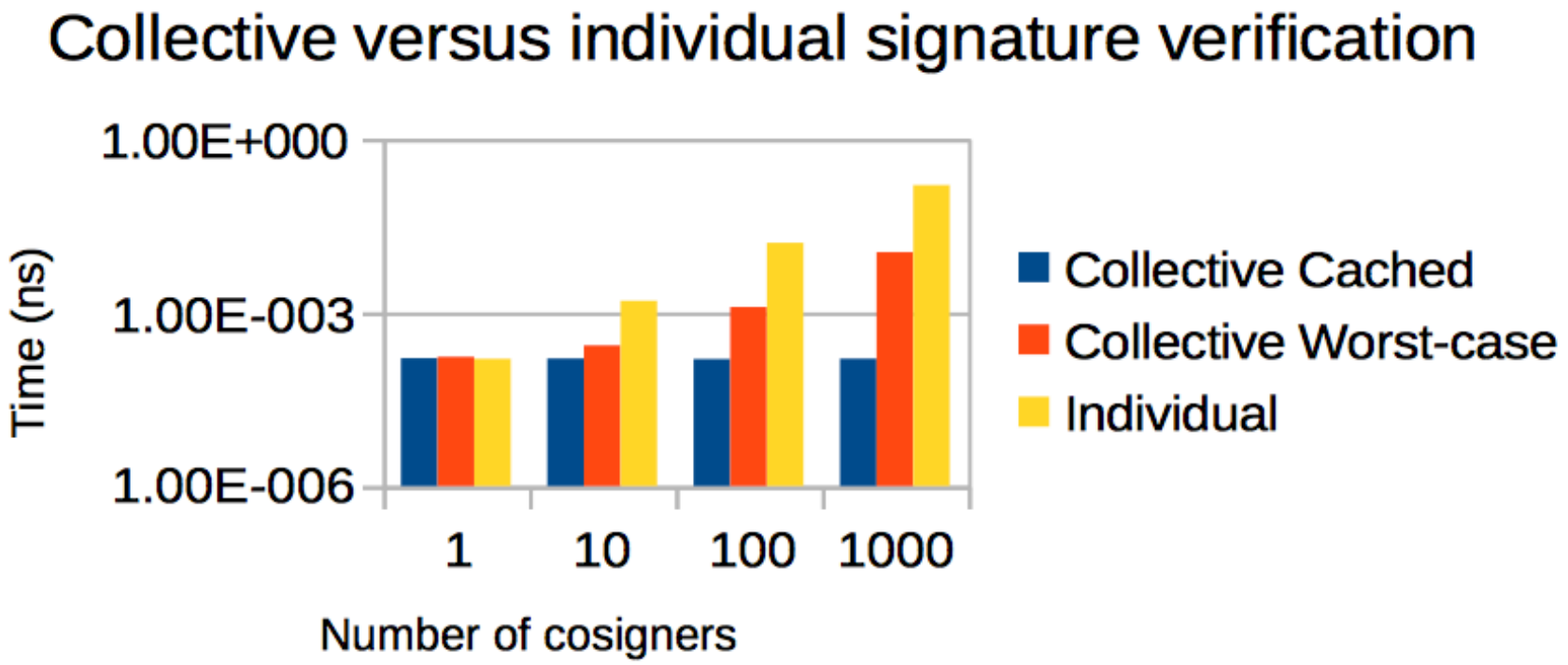
Future: deploy, evaluate at scale on "real Internet"

- Evaluate impact of high node, network churn
- See paper for approaches to handling if/when needed

Results: Collective Signing Time

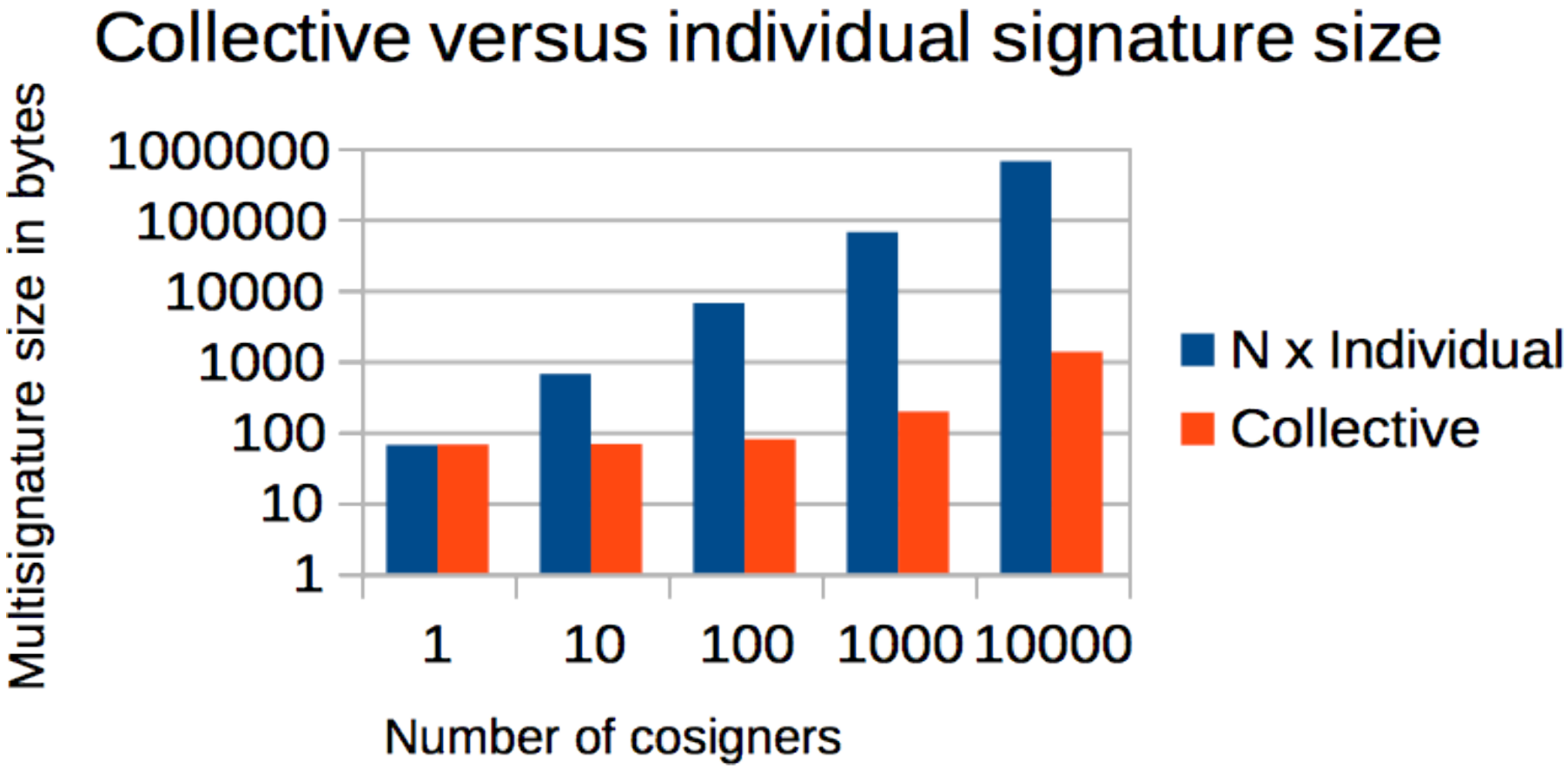


Results: Verification Cost



Results: Collective Signature Size

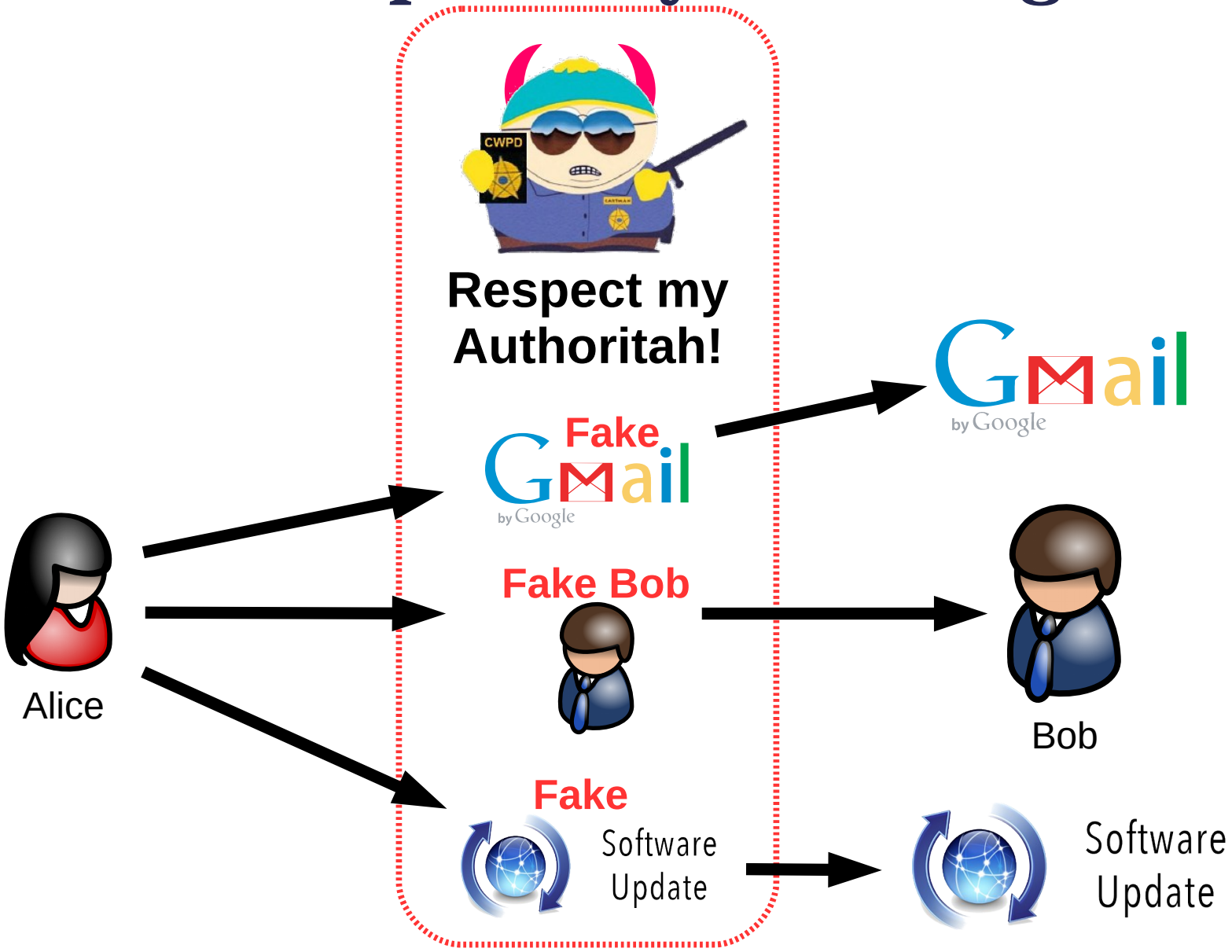
Ed25519: up to 512x smaller than N signatures



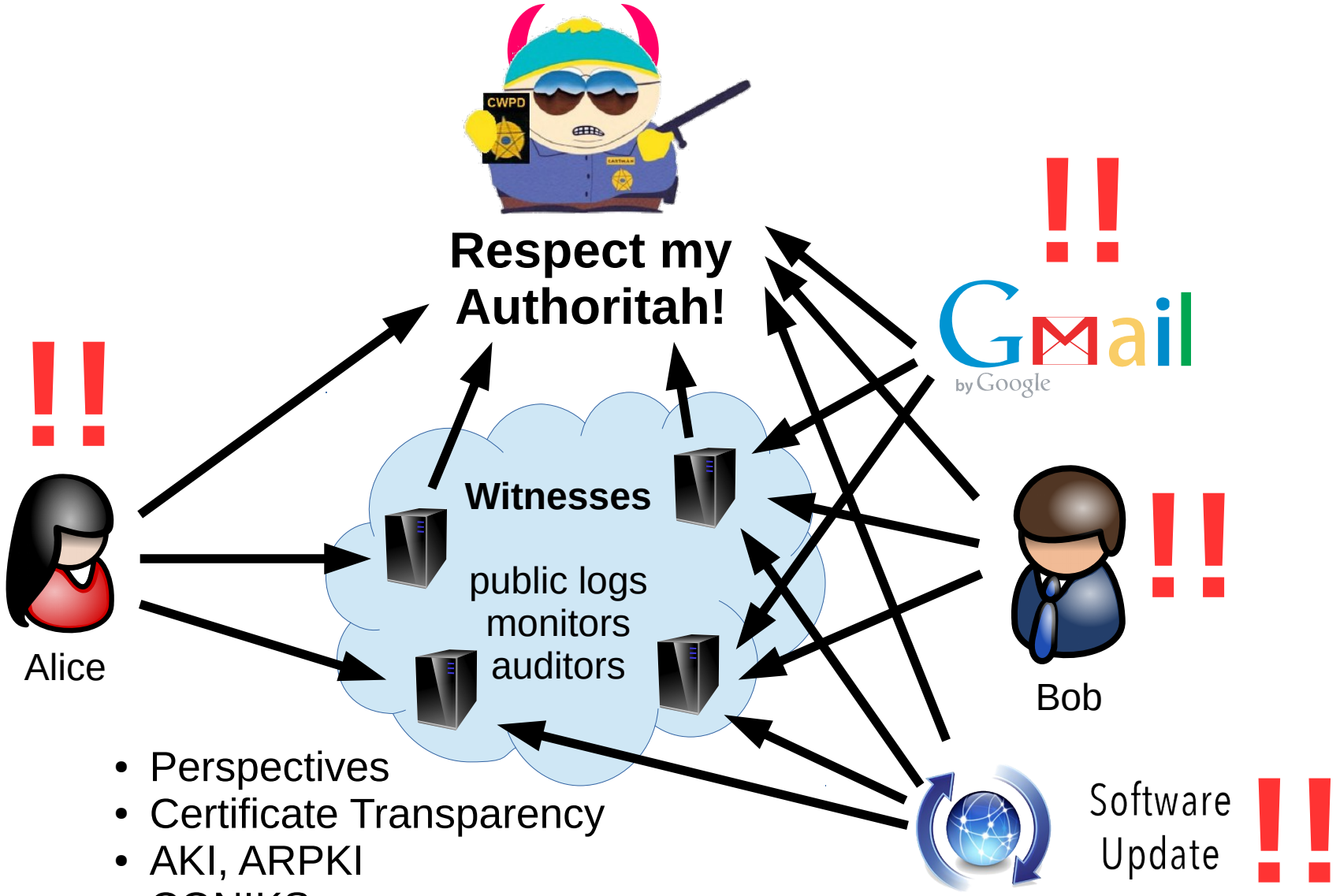
Talk Outline

- The trouble with trusting authorities
- Grand challenge: decentralize the authorities!
- Baby step: decentralized witness cosigning
- CoSi: scalable collective Schnorr/Ed25519 signatures
- Experimental evaluation: scalability, signature size
- **Comparison with prior transparency approaches**
- Status, future work, and conclusions

The Transparency Challenge

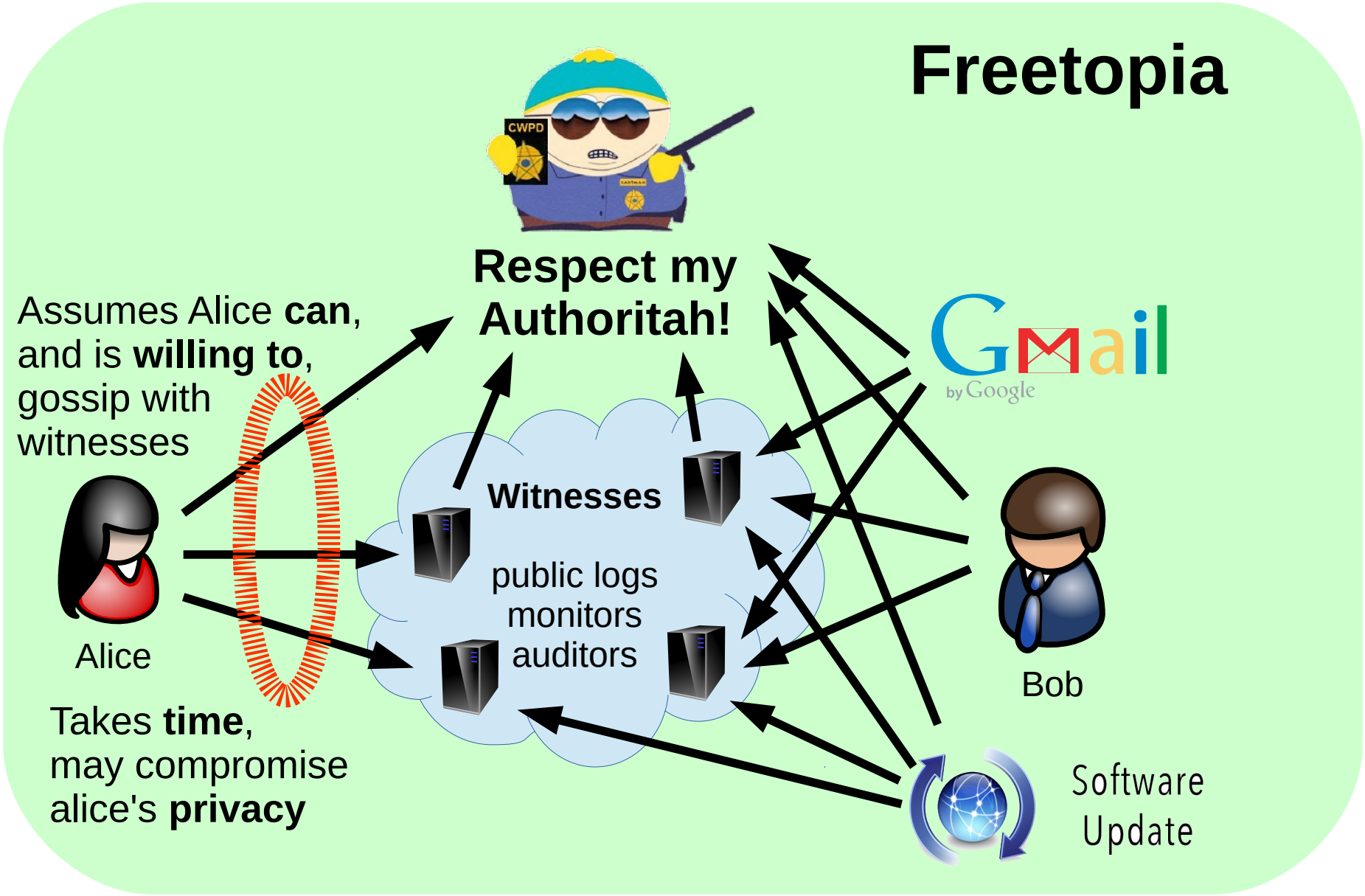


Existing Transparency Solutions



- Perspectives
- Certificate Transparency
- AKI, ARPKI
- CONIKS

An Important Assumption

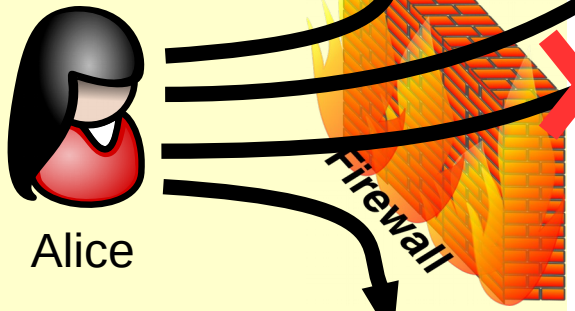


A Different Scenario

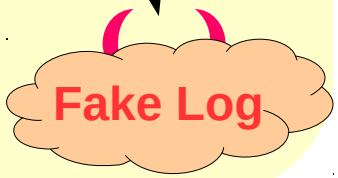
Tyrannia

Gen. Rex

Fake CA



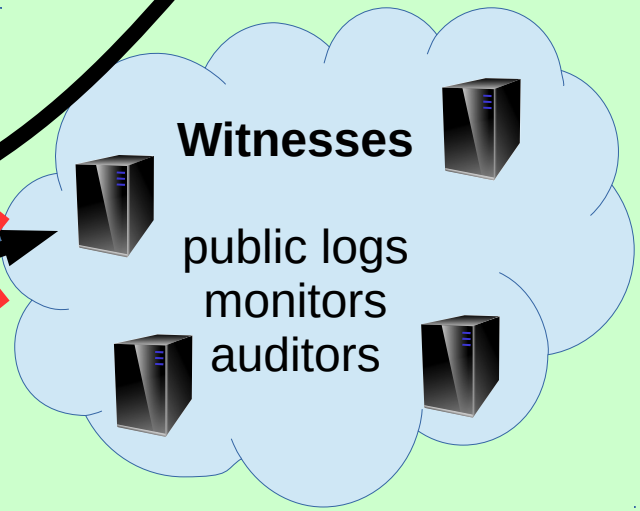
Alice



Fake Log

Freetopia

Respect my Authoritah!



Gmail by Google

Bob

Software Update

Gossip versus Collective Signing

Gossip can't protect Alice if she...

- **Can't** (because she's in Tyrannia)
- **Doesn't want to** (for privacy), or
- **Doesn't have time to**

cross-check each authoritative statements.

Collective signing **proactively** protects her from secret attacks even via her access network.

- Attacker can't secretly produce valid signature

An “Extreme” Scenario

What if an attacker **controls the target device,** wants to secretly coerce the device’s vendor to sign a back-doored operating system image?

- A phone **sealed in a forensics lab** can’t gossip!
  - Certificate Transparency can’t reveal backdoor
- Only protection is to bind the transparency **proactively** into the device-verified signature

Talk Outline

- The trouble with trusting authorities
- Grand challenge: decentralize the authorities!
- Baby step: decentralized witness cosigning
- CoSi: scalable collective Schnorr/Ed25519 signatures
- Experimental evaluation: scalability, signature size
- Comparison with prior transparency approaches
- **Status, future work, and conclusions**

Prototype available; give it a try!

Go to https://github.com/dedis/cosi

- Binaries: see releases
- Source: go get -u github.com/dedis/cosi

```
cosi sign -g group.toml -o sig msg_file
cosi verify -g group.toml -s sig msg_file
```

Run your own witness server: cosi server

Verifier libraries for C, Go – see README

Status, Incremental Deployment

Still experimental! But…

- DEDIS lab committed to supporting, assisting with integration/deployment efforts
- Don't want to trust collective signatures yet? Add in extension field alongside individual sig
- Don't want to trust protocol, server liveness? Fork/exec '`cosi sign`', set timer, kill if needed
- Don't want to trust `cosi` software? Sandbox it! Needs almost nothing to run.

Send feedback privately or discuss publicly on https://groups.google.com/forum/#!forum/cothority

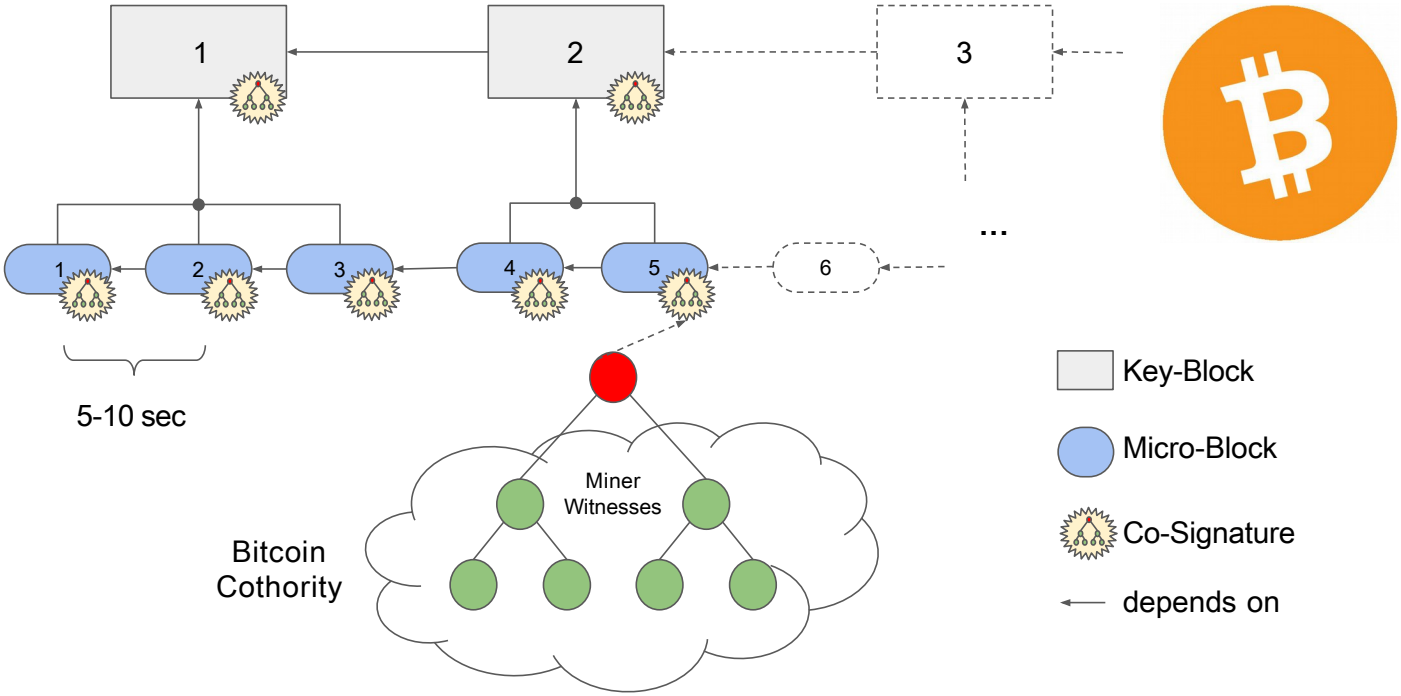
Other uses of collective signing

(credit: Tony Arcieri)

Other uses of collective signing

"Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing"

- To appear at USENIX Security 2016
- Draft: http://arxiv.org/abs/1602.06997



Conclusion

Grand challenge: **decentralize all the authorities!**

Practical baby step: **decentralized witness cosigning**

- Ensures that for **any** signed statements that exists, **many parties** have witnessed, publicly logged it
  - Protects even clients that can't gossip
- Can incrementally add to **existing** authorities
- CoSi protocol **scales** to large witness groups

Available: **https://github.com/dedis/cosi**

Public question/answer, discussion forum:
https://groups.google.com/forum/#!forum/cothority