Decentralization as a Privacy-Enhancing Technology

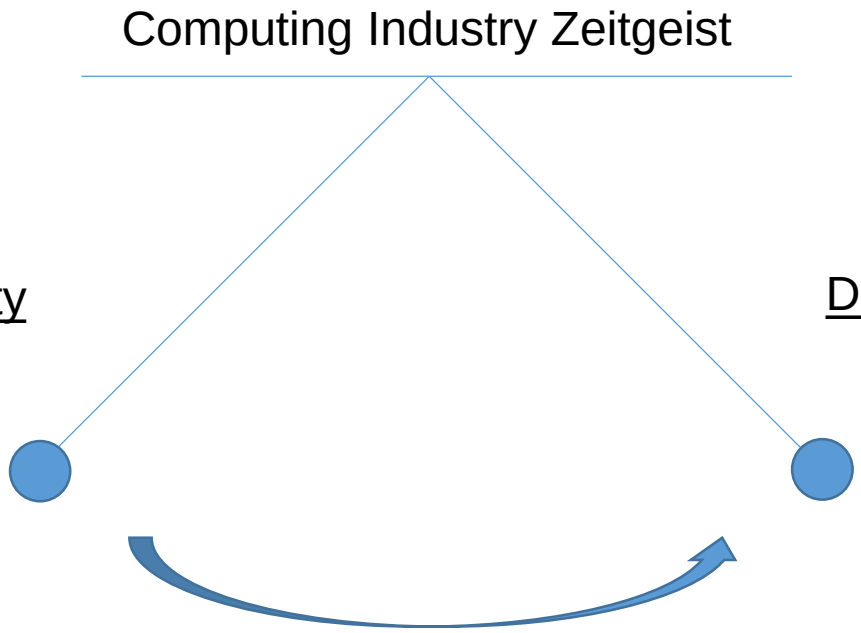
George Danezis, University College London

With Marios Isaakidis (UCL), Carmela Troncoso (IMDEA), and Harry Halpin (INRIA/LEAP)

THE ALAN TURING INSTITUTE

NEXTLEAP

The De/Centralization pendulum



Centralized authority

- Mainframes
- AOL / CDNs
- The Cloud
- ‘Services’

Decentralized authority

- The PC
- The Internet
- Peer-to-peer
- Bitcoin

What do we mean by centralized authority?

Decentralized: Infrastructure vs. authority

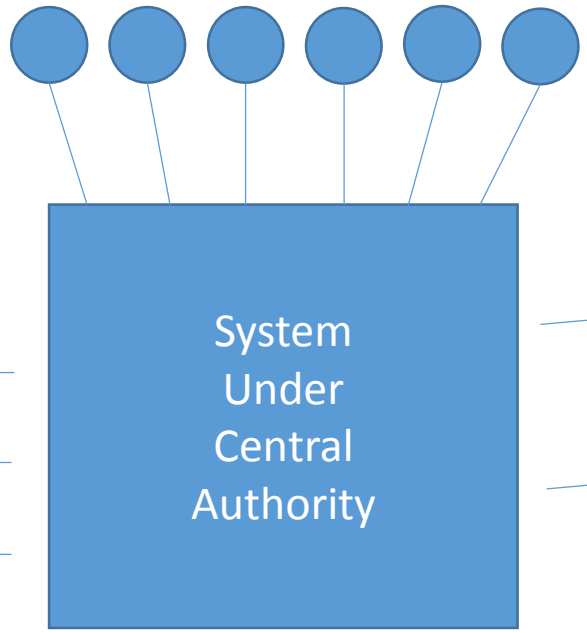
The Google Cloud

- Very large distributed system.
- Paired Datacentres.
- Chubby: uses paxos for distributed locks.
- BigTable: eventually consistent bulk storage.
- Map-reduce: indexing.
- Sharding to serve users.

Gnutella

- Many peers storing local files & flood fill search.
- Peers connect to other peers to ask for files.
- Peers download from others.
- Super-peers can optimize some routing.

A critical view of centralized authority



Problems

What if the authority goes rogue?

No Privacy?

No Integrity?

No availability?

Coercion

Example: webmail, web-server, search engine, online storage.

Social networks

Solutions?
Secure cloud?

Integrity fix:
zero-knowledge

Privacy fix:
Homomorphic Enc.

Availability and Coercion?

Very expensive
(3.2GHz vs. 1Hz-100Hz)

Two events with profound significance …

Napster (2000)

- Distributed peers could share music.
- Through a centralized indexing service.
- Legal challenge in 2000 (RIAA).
- Ordered to keep track of activities to enforce copyright.
- Closes service in 2001.

… bittorrent (2001)

E-Gold (2008)

- Online currency backed in grams of gold (launched 1996)
- Central entity kept balances & gold. Instant trades.
- Constant uncertainty about status of “money transmitter”
- 2006-08 DoJ categorises as transmitter and prosecutes.
- Service closes down.

… bitcoin (2008-09)

The (naïve) promises of decentralized authority

Can a decentralized authority architecture be a game changer?

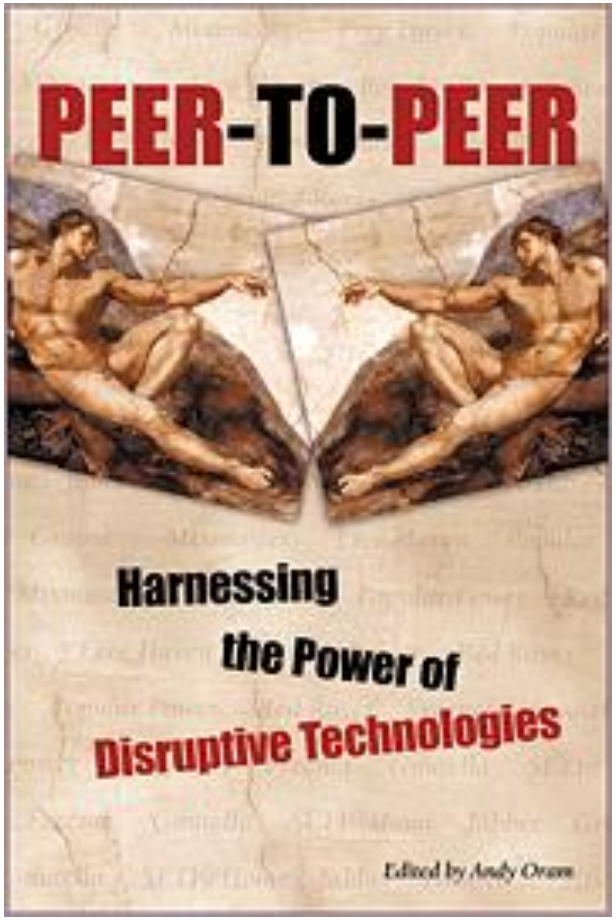
- Privacy: no single entity -> no mass surveillance?
  - Think: SNS / Prism.
- Integrity: no single entity -> no mass control? No government?
- Availability: no single entity -> no suppression.

How it all started?

Ross Anderson. "The eternity service." In *Proceedings of PRAGOCRYPT*, vol. 96, pp. 242-252. 1996.

“ I had been alarmed by the Scientologists' success at closing down the penet remailer in Finland; the modern era only started once the printing press enabled seditious thoughts to be spread too widely to ban. […] So I invented the Eternity Service as a means of putting electronic documents beyond the censor's grasp. ”

The hype last time …



- February 2001
  - Internet,
  - Napster,
  - Commons,
  - SETI@Home,
  - Jabber,
  - mixmaster,
  - gnutella,
  - freenet,
  - redrover,
  - publius,
  - free haven, …

What have we learned since?

The wave of decentralization following Snowden (2013)

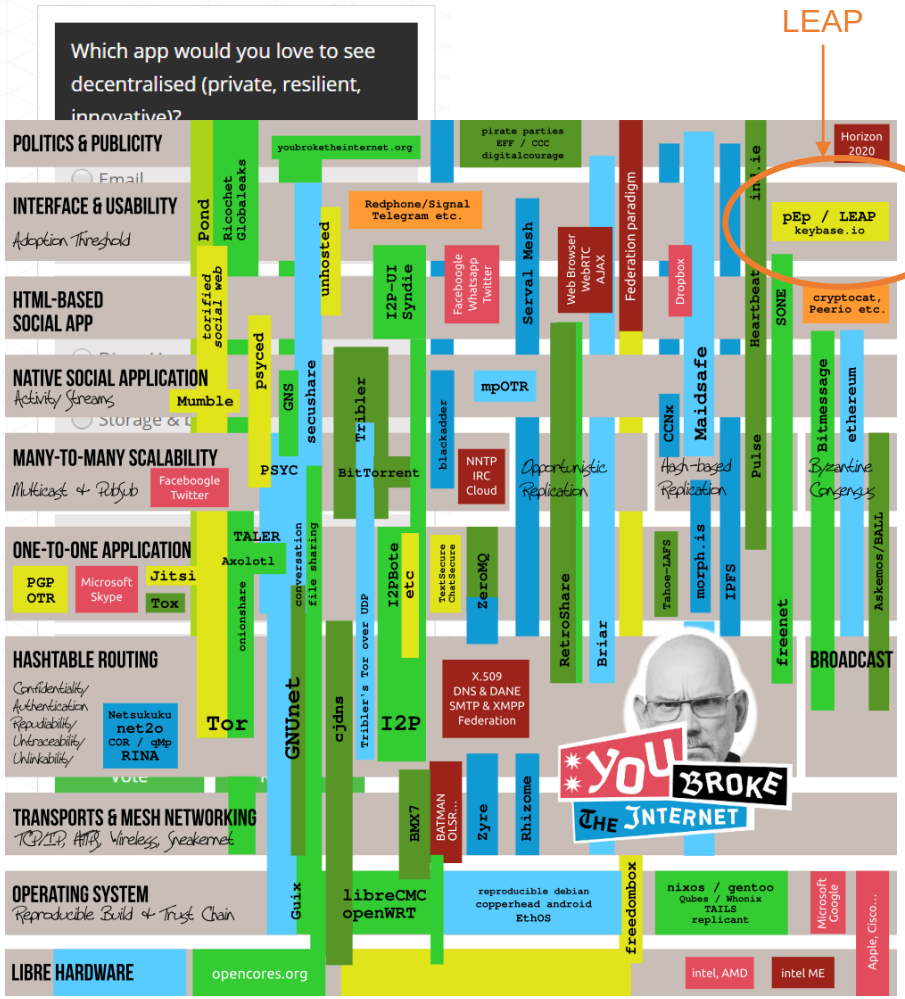
redecentralize.org Interviews Community About

Redecentralize · taking back the net

Quietly, some geeks are decentralizing the net. Again. Who are they? Why are they doing it? What new technologies are they using? How will this change the world?

We interview them to find out, and we're a community to help out.

Watch Subscribe Discuss



Dangerous waters:

(1) A narrative about a past golden age.
(2) A theory about an external factor that let to decay & moral corruption.
(3) A theory of social change based on the removal of said external factor.

"palingenetic myth" (Roger Griffin, 1991)

What have we learned in the past 15-20 years?

- How are systems decentralized?
- How decentralization supports privacy?
- What we gain from decentralizing?
- What may be lost with respect to privacy/security when decentralizing?
- What implicit centralized assumptions remain?

A focus on privacy, with reflections on blockchains.

Main lesson:

- Decentralization is a whole design space.
- No golden age. Maybe a golden future.

Decentralization: How?

The many faces of decentralization

Users & infrastructure tensions

- **Common**: multiple sources of authority!
- There is *no* infrastructure:
  - Difficult to imagine: telecommunications / WAN
  - *Samba* / LAN protocols.
  - Direct IM (historical skype voice channel).

LAN / Radio

- Users *are* the infrastructure:
  - They use each other as infrastructure.
  - Example: *Freenet, Gnutella* (-> supernodes)
    Distributed Hash Tables (*Kademia*)

Issue: churn, reliability

- Infrastructure is *distinct* from users:
  - Examples:
    Bitcoin separation between “miners” and other users.
    Tor separation between relays and users.

Centralizing Tendencies

How do nodes related to each other?

- Distributed:
  - Well defined entities relating to each other.
  - Well established distributed system with Byzantine failures.
  - Examples: MPC, Distributed Storage, Tor relays.
- Federated:
  - Multiple sources of authority representing users.
  - Example: Email / SMTP / Jabber
- Peer-to-peer:
  - Open world, no central “admission control”
  - Examples: Bitcoin Miners, Torrent swarms
- Social-based:
  - Relations of trust between nodes.
  - Theoretical systems: XXX
- Auditing / Accountability relation:
  - Doer / Verifier distinction
  - Examples: Electronic voting systems, certificate transparency, bitcoin miners

Closed World
Admission?

Imbalance
of power

Sybil attacks

Where do these come from?
Social engineering.

Can verify only so far:
completeness, availability.

Structure of the network

Technical, Distributed Systems, interaction:

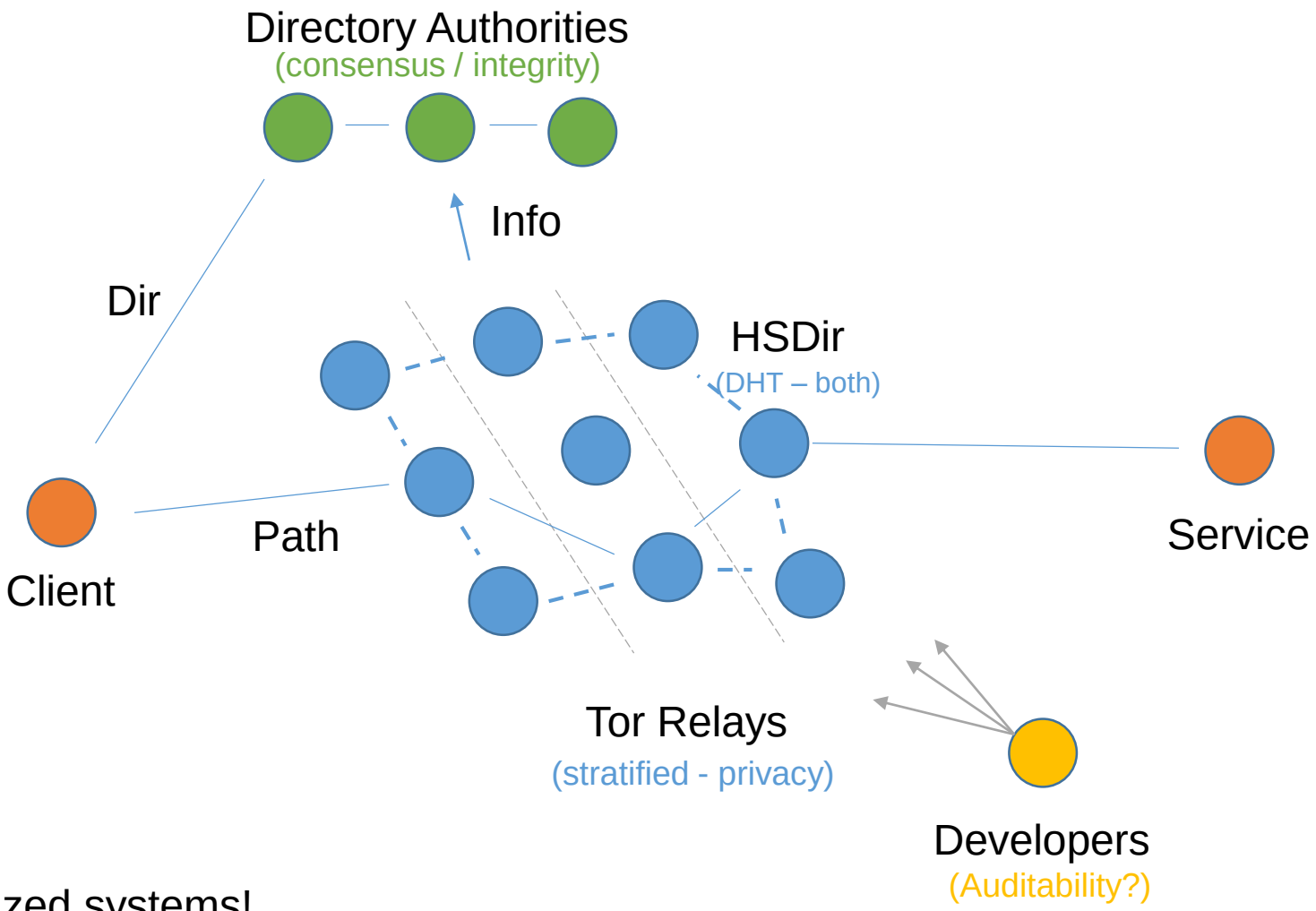
- Mesh:
  - All talk to all – $O(n^2)$ channels, run out of sockets.
- Gossip:
  - Sparse connectivity, opportunistic gossip
  - No efficient routing – broadcast only
  - Example: Bitcoin mining, Gnutella, CT
- Structured:
  - Nodes assume positions to facilitate efficient routing. Coordination?
  - Example: Infrastructure less Torrents, Tor HSDir.
- Restricted (Stratified, cascaded):
  - Specialization. Eg. Tor routers (Exit, Middle, Guard)

- Scale Free / Social:
  - Do not talk to strangers
  - Examples: Darknet mode Freenet; MCON – covertness
- Hierarchical:
  - Contradiction in terms? Maybe not.
  - Spanning tree protocols: AS, BGP, SCION architecture.
- Content centric:
  - Structure interactions around content.
  - Examples: CCN, …

Diversity: real systems combine the above for different parts of their infrastructure:

- Tor routers (stratified)
- Tor HSDir (structured)
- Tor Directories (Mesh)

Case studies: The tor anonymity system



4 Decentralized systems!

Reflections on Tor

- Complexity

- Relation between dev, authorities and relays:
  - Development:
    - centralized, but extremely verifiable.
    - Decentralized Deployment.
  - Directory authorities:
    - More-centralized / less open.
    - High-integrity with verifiability
  - Relays:
    - More-decentralized / more open: privacy
  - Clients / Services:
    - Autonomy to pick relays / HSDir.

Decentralization: what Privacy?

What privacy properties are supported.

Privacy of content

- At the heart of traditional cryptography.
  - Can we realize a functionality without TTPs?
- Threshold encryption / Decryption:
  - All systems based on threshold assumptions are about distributed architectures.
  - Eg. Distributed decryption of ballots in electronic election.
- Distributed storage:
  - Original Eternity Service, Free Haven, Tahoe-LAFS, IPFS
  - Encrypt blocks and store them (availability).
  - Joint decryption / retrieval.
- Private computations / SMPC
  - “Multi-party” assumes parties do not collude: distributed authority.
  - Often presented as peers: example 2PC.

Anonymity & meta-data privacy

- Who is talking to whom?
  - Intrinsic: need a group of other users – decent. Authority.
  - Eg. mix network, Tor, crowds, Tarzan, election mix nets.
- Hide user action:
  - Information theoretic Private Information retrieval (PIR): assume a threshold of honest servers.
- Censorship circumvention:
  - Use a decentralized system for escaping censorship.
  - The original reason! Eg. Eternity,
- Covertness:
  - Traffic obfuscation against shaping (bittorrent)
- Unlinkability of operations
  - Example: *z.cash* – remove link between payer and payee in cryptocurrencies.
- Address book / social network privacy
  - Examples: DP5 – a private presence systems.
  - Xbook: private social networking.
- Plausible deniability:
  - Tangler: no block can be ascribed to a specific file.

Remove central “trust”

Can we use transparency & decentralize checking to turn trusted third parties into untrusted ones? Two approaches:

- Substitute TTPs with decentralized protocol:
  Eg. Distributed anon. credentials – the central bank is substituted by a joint oblivious functionality.
- Allow TTP but force transparency:
  Logging in certificate transparency. Include all observed certs (from central certification authorities) into the logs, and check for conflicts.

The problem of software development:

- *Is the actual software not inevitably a centralized point of failure?*
- Apply the transparency approach: Eg. Tor – all development is done in public repositories; deterministic builds ensure all can verify the genuine binary; authority to upgrade is in hand of operators.
- Same for bitcoin – choice to deploy is up to miners.

What decentralization buys you?

Architectural advantages of decentralization

Reduce costs, spare resources & deployment

- Spare capacity & spare infrastructure:
  - Early peer-to-peer: spare CPU (SETI@home) & strorage (Freenet)
  - Current resources that are difficult to centralize: Network location diversity: eg. Bridges for bypassing censorship.
  - When security is associated with diversity decentralization is an obligatory option (legal diversity, network diversity).

- Leveraging existing trust networks:
  - Through decentralization designers can use local “trust” assumptions.
  - Example: Drac anonymity network design – each user connects with friends to relay anonymously information.
  - Decentralized Social Networks rely on this heavily.

Flexible “trust” models

- Distributed Trust:
  - All threshold protocols require decentralized architectures.
  - Distributed key generation, public randomness, decryption, signing.
  - Ensures that a subset going rogue does not compromise the security properties of the system.
  - **Distributed Trusted Computing Base – no single entity can compromise it.**

> 3b. Discuss the validity of the following statements, and justify your answers:
>
> “There is always an entity that can compromise all security properties offered by a computer security system.”
>
> [5 marks]

- No natural single authority:
  - *What is there is genuinely no single authority that can run the system?*
  - Key examples: access control in “distributed” systems.
  - Eg. TAOS and SDSI access control logic rely on attributes from different authorities to decide access to resources.

Resisting formidable adversaries

- Separate deployment from operations:
  - Since operators are separate from developers, pressure on developers should be ineffective to violate the properties of the system.
  - In case of suppression open source ensures forks will survive.
  - Examples: Tor and Bitcoin.

- Censorship resistance:
  - Pressure on a small number of entities cannot entirely eradicate the use of the system.

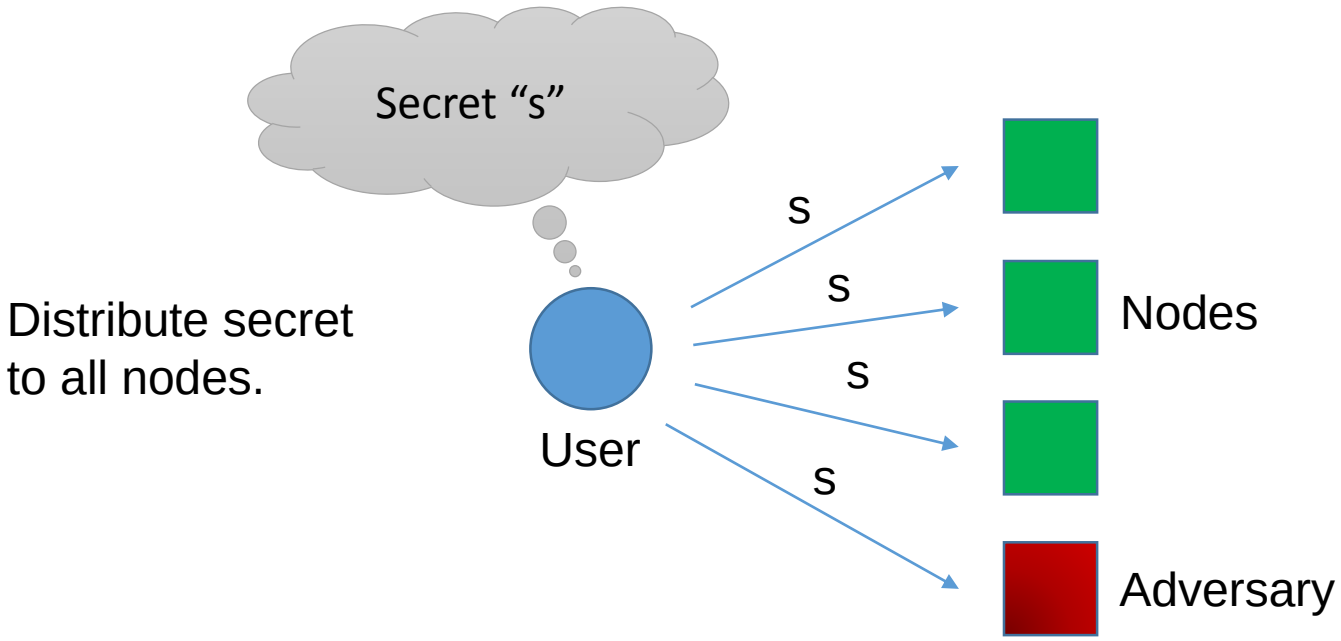
- Covertness:
  - Wide distribution of infrastructure (only some architectures) ensure no single points of suppression exist. Peer architectures (Membership concealing networks) hide participation.

- Survivability:
  - Peer-to-peer Botnet architectures: difficult to take down / and even detect the bot master. Is that a decentralized architecture?

What you lose when decentralizing?

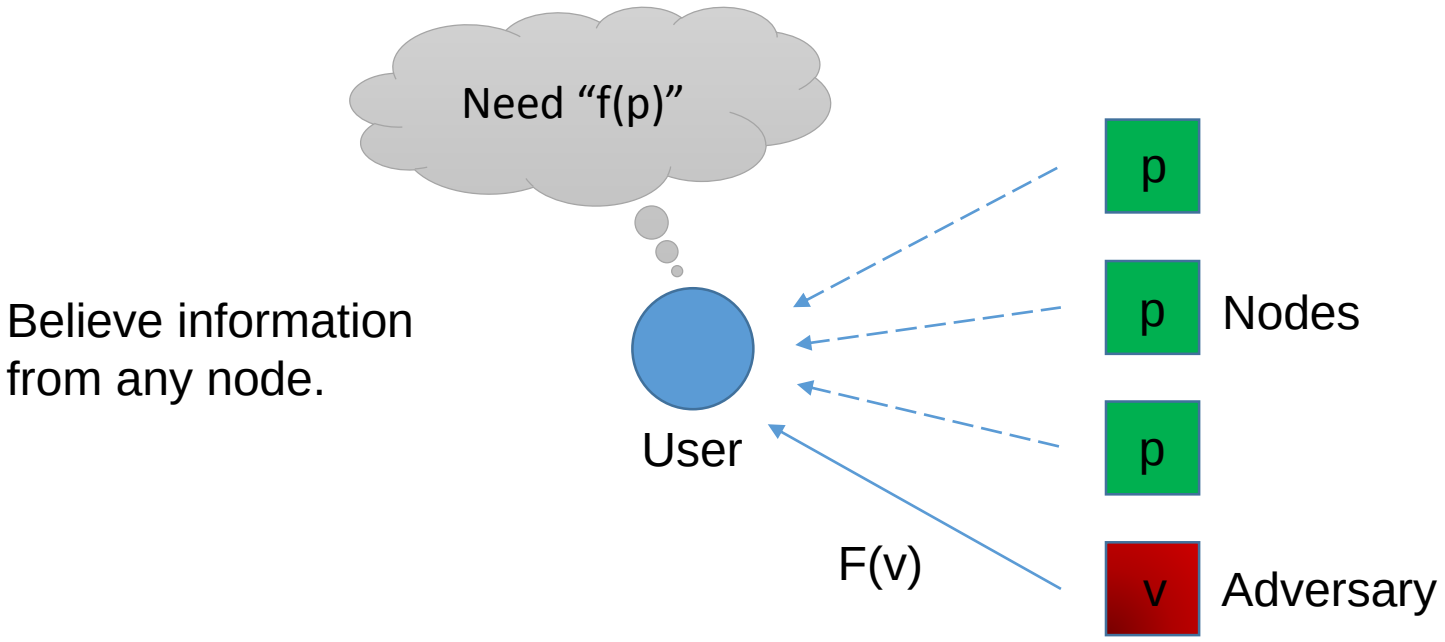
"Hell is the others"

Patterns of fragile decentralization: Privacy



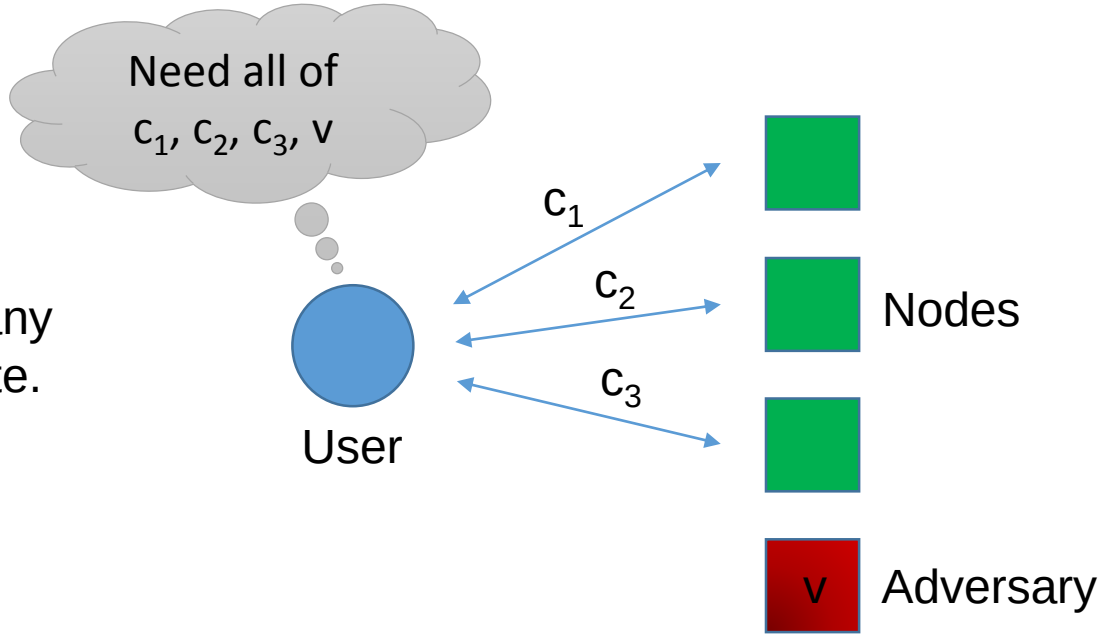
If any part of the decentralized system is corrupt you lose privacy.
Safe: split across all nodes (ok if any honest).

Patterns of fragile decentralization: Integrity



If any part of the decentralized system is corrupt you lose Integrity.
Safe: all nodes agree on the value sought.

Patterns of fragile decentralization: Availability



Require all / many others to operate.

If any part of the decentralized system is unavailable you lose service.
Safe: rely on small agile subset of nodes.

Morality: Achieving Privacy, Integrity and Availability cannot be done purely architecturally and will require some heavy crypto-magic™.

Increased attack surface

- Internal adversaries:
  - Other nodes may be controlled by the adversary.
  - Traditional security architecture, “crunchy on the outside – soft on the inside” is not applicable.
  - Extremely demanding security engineering problem!
  - Examples: routing security in distributed hash tables (DHTs).
- Content interception &Traffic analysis:
  - Actions mediated though others -> more opportunities for content interception and meta-data inference. Eg. Tor exit nodes.
- Attacks using inconsistent views
  - No single authority may mean no authoritative state!
  - A lot of work has to be done to ensure consistency.
  - Example attack: different views of relays in an anonymity system.
- Privacy loss.
  - Others are infrastructure – they see your secrets.
  - Example: miners in bitcoin see all transactions – and so is everyone else.
  - Participation in the system may be difficult to conceal.
- Denial of service
  - Others may decide to stop playing with you.

Cumbersome management

- Routing difficulties:
  - Pure overlays make routing uncertain.
  - This is also a problem for the Internet (BGP attacks)
  - Adversaries may poison names, paths and stop relaying.
- Performance loss
  - "The price of anarchy" – all act under partial information & local optimums.
- Difficult attack prevention
  - Centralized security measures cannot be deployed.
  - 2 key examples: (1) spam detection and prevention (2) anomaly detection.
  - Result: only properties that can be implemented using strong crypto survive.
- Challenging collaborative computation
  - Private & correct Joint computation harder than routing & storage.
  - Example: private statistics in the Tor network. Bitcoin: only pseudonymity.
- Network diversity:
  - Vastly different nodes in terms of power, bandwidth, availability, and willingness to help others.

Lack of accountability & reputation

- Information integrity.
  - Information may not be reliable, since other entities must be incentivised to be truthful. (Not just inconsistent but plain wrong)
  - Turns all problems in distributed systems into an economic mechanism design: elicit truthful participation.
  - Makes security engineering a superset of game theory and economics!

- Poor incentives & economics:
  - Lack of de-facto long term identities undermine repeated game equilibriums.
  - Example: The mojo nation storage protocol: hyperinflation, and collapse.

- Sybil attacks:
  - It is not trivial to tell whether others are “real”, or a mere multi-instanciation of a single adversary.
  - What is real anyway? The case of the flash mob.
  - Solutions: proof-of-work (Bitcoin), piggy-backing on centralized admission control (Tor – the IP network), or social authentication (advogato).
  - Deeper question: what makes a genuine constituent?

The Centralized bits in decentralized designs

Looking under the rug of dencentralized systems

Directories & state are (more) centralized

- Node / peer finding / indexing:
  - Classic example: Napster – files are on user machines but information routing, indexing and search done centrally. Fail!
  - Tor: Distributed directory infrastructure lists all relays & attributes. However centralized enough to allow blacklisting by firewalls.

- Path selection & reputation:
  - Global “reputation” scores …
  - Entities that configure /select / optimize paths.

- Question: is a lottery a decentralized state decision system?
  - Imagine that at any time we elect a dictator and their state becomes the state we all accept.
  - Of course subject to some checks: integrity.
  - However: completeness may be difficult to check.
  - … the bitcoin “consensus” backbone.

Other centralized bits & assumptions

- Authentication / authorization.
  - Let's use a single-sign on! Admission control for Sybil prevention. Nope.
- Abuse prevention
  - Lets create a global score for everyone! I know spam when I see it. Hm.
- Payment system
  - Decentralized systems are decentralized, for everything else there is mastercard.
  - Bitcoin to the rescue!
- Collective computations are centralized.
  - Let's face it: Multi-party computation is just too hard.
  - Remember: is picking one at random really decentralized?
- End-systems?
  - Pattern: What is we use the end user machine? User control.
  - Is that really decentralized? Only if endpoints can be effectively protected.
- Incentives are correct
  - Welcome to mechanism design, your second PhD.

Towards Rabid Decentralization

Decentralization will not happen by itself or naturally

Decentralization: No silver bullet

- Good will, slogans and demands are not enough.
  - Neither is return to a lost golden age.
- What do you need to build good secure decentralized systems?
  - Deep knowledge of distributed systems. They will by definition be distributed.
  - Deep knowledge of cryptography: necessary to achieve simultaneously privacy, integrity and availability.
  - Mechanism design, game theory and sociology – otherwise selfish or otherwise motivated actors will get you.
- How many people in the world exist that combine those?
  - How many of those work for Google?
  - Compare with the number that know how to build a simple centralized web service.
- The fundamental economic problem of building & maintaining such systems.

Vulnerability to one or many authorities

- Unsafe design pattern for one security property, is a good solution for the others.
- Examples:
  - Bitcoin: high-integrity – at the cost of a public ledger, ie. little privacy.
  - Tor routers: high-privacy at the cost of no available or correct collective statistics.
  - Zerocash: combines high-privacy & high-integrity "efficiently" – uses cryptographic assumptions (SNARKS) that will make you cry.

Open philosophical question:

Is being vulnerable to a "random" subset of decentralized authorities better than being vulnerable to one for either integrity or privacy?

Examples: decentralized social networks (*diaspora*).

Inefficient decentralization = no much decentralization

- A problematic dynamic: high-integrity requires a majority to honestly participate in decisions.
  - Example: bitcoin – all miners need to hear of all transactions / blocks, all need to verify new blocks.
- The bigger the decentralized network, the more work each peer needs to do.
  - Growing the network reduces its capacity to do work!
- Result: require enough separate authorities to ensure diversity, but as few as possible to ensure efficiency.
  - Conjecture: is that the reason mining pools are concentrate bitcoin mining?
  - What that that say about natural centralizing tendencies in decentralized systems / and markets?
  - Politics: Separation of powers (usually only 3!)

Decentralized institutions to support decentralized systems

- The promise of Bitcoin: algorithmic monetary policy, etc.
- More generic trend in decentralized systems:

"*They want to replace western civilization with a bunch of crappy Python scripts*" – Dr Halpin.

- More likely: Co-evolution of decentralized systems for privacy and accounting & social institutions embedding privacy and transparency.
  - What will these look like?
  - Ideas from 2001: Commons, Wikipedia, …
  - Governance in free software projects: Tor & Bitcoin …

In conclusion …

- How to make decentralized systems scale up: the more participants the more capacity and value?
- How to integrate strong integrity and privacy crypto protections despite the wide distribution and decentralization?
- How to co-design institutions, incentives, usability and governance in vast decentralized systems?

Join Sarah Meiklejohn and myself at University College London:

3 post-docs on systems, crypto and usability of distributed ledgers.