Deployed Anonymity Networks: A brief history

Nick Mathewson
<nickm@freehaven.net>
The Free Haven Project

This presentation

- About me
- Introduction to anonymous communication
- High-latency networks
- Low-latency networks
- Lessons learned
- Conclusions and predictions

About me

- I wrote Mixminion and helped design it.
- I help design and write Tor as a full time job.
- I like researching problems that nobody knows how to solve yet.

Anonymity: what is it?

- *Informally*: One user does something, and nobody can tell who did it.
  - "Nobody" = ...?
- *More formally:* Within a certain **anonymity set** of possible actors, an attacker with given capability can't link actors to actions with better-than-chance probability.
  - "More users, more anonymity."
- *Mathematically:* (see the literature.)

Anonymity: what is it not?

- Steganography
  - "Nobody can tell I was participating."
- "Plausible deniability"
  - "You can't prove it was me!"
    - (Prove is such a strong word.)
- Cryptography
  - "You can't tell what I'm saying."
- Non-collection/non-retention
  - "I promise I'm not looking."
- Not writing your name on it
  - "Isn't the Internet anonymous already?"

Anonymity: who needs it?

- Private citizens (anonymity = "Privacy")
  - Avoid identification by communications partners
  - Avoid profiling by advertisers and others
  - Avoid retribution for stigmatized/oppressed opinions or interests.
- Businesses (anonymity = "Security")
  - Investigate competition
  - Hide strategic relationships
- Governments too ("Anti-traffic analysis")
  - Investigate savvy criminals
  - Hide location of employees

How do systems differ?

- Communication or publication?
  - (I'm mostly skipping over publication.)
- Low-latency or high-latency?
- Recipient or sender anonymity?
  - Also called initiator/responder.
- Censorship-resistance?
- Provide anonymous service,
  or anonymous access to other service?

Why focus on deployed systems?

- More problems need to get solved.
- Connecting to reality gives perspective...
  - On reality of problems
  - On cost and benefit of solutions
  - On relative strengths of attacks
  - On relative importance of features
- Too many unbuilt (unbuildable?) systems.
  - (one-shot, expensive, “magic”-powered, etc.)

Thread 1: high-latency communication

- Advantages
  - High latency variance prevents easy correlation
- Disadvantages
  - Too slow for interactive applications

Common high-latency mistakes

- “More features means more ways to be anonymous!”
- “Server discovery will take care of itself.”
- “We can afford to cut ourselves off from the Internet.”

Prehistory: Chaum's Mixes* (1981)

*not MIXes!

- Users encrypt messages and destinations with a mix server's public key, and send them to the mix.
- The mix receives a batch, re-orders it, and decrypts it.
- If the mix is honest, an observer can't link messages.
- Chain several mixes in case some are dishonest.

anon.penet.fi (~1991)

- J. Helsingus
- Single-hop remailing service with pseudonyms
- PGP encryption
- Single-hop means single-point of failure; fell to dubious legal attack.

Cypherpunk (Type I) remailers (~1993)

- Hal Finney et al
- Not (at first) influenced by Chaum (!)
- Built from existing mail servers and PGP
  - Text based; easy for Unix hackers to use.
- Vulnerable to many, many attacks
- Many extra features bolted on over the years
  - (But N optional features means 2^N possible sets of features. Users don't act the same!)
- Support for anonymous replies

Mixmaster (Type II) remailer (1995)

- Influenced by Chaum, Cypherpunk
- Uniform set of features
- Closed most attack vectors
  - Size correlation: all messages same size
  - Partitioning: one format, no PGP.
  - Replay: remember messages, and stop replays
  - Reply block flooding: no replies
- Invented novel techniques to resist blending
  - Timed dynamic pool algorithm

nym.alias.net (1996)

- Mazières and Kaashoek
- Email pseudonym service built using Type I
- Nymserver holds, for each pseudonym, a public key and a reply block.
- Mail to pseudonym is retransmitted to corresponding reply block.
- Vulnerable to flooding
- If part of reply block's path goes down, messages are lost

Babel (1996)

- Gülcü and Tsudik
- Included distinguishable anonymous replies
- Required non-anonymous parties to run special software.
- Experimental deployment, never widely distributed.

Mixminion (Type III) remailer (2002-)

- Goal: replace cypherpunk and mixmaster by reintegrating replies into secure remailer.
- Goal: close all remaining known holes in remailer network.
- Adds single-use reply blocks
  - Replies indistinguishable from forward messages
- Integrates and formalizes server directories
  - (Enabling key rotation, which make replays harder.)
- Drops SMTP transport
- K/N fragmentation

Thread 2: low-latency communication

- Advantages
  - Fast, so suitable for web traffic, SSH, IRC, IM, etc.
  - Easy to integrate with interactive apps
- Disadvantages
  - Fast, so attackers watching both ends can try to correlate timing and volume.
  - Easy to integrate with annoying interactive apps

Common low-latency mistakes (1)

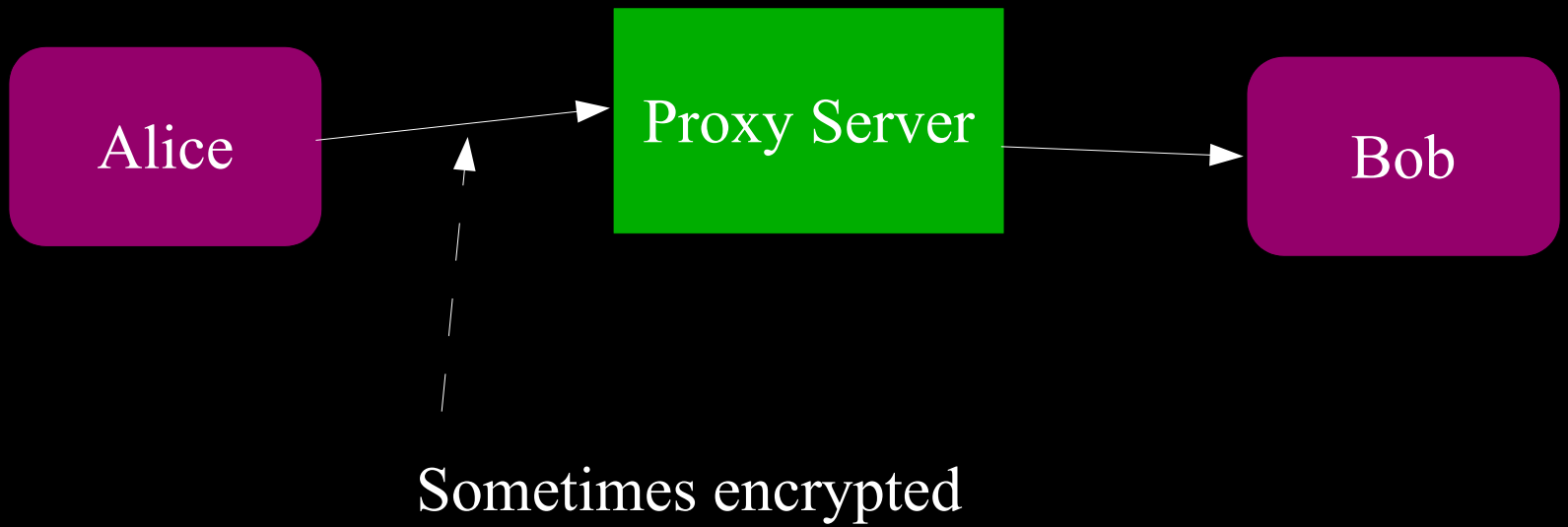
- Ignoring end-to-end attacks
  - By claiming perfection.
  - By defending against something harder.
- Voodoo padding
  - “Surely, this will confuse the attacker!”
    - “After all, it confuses me!”
    - “What do you mean, the attacker knows statistics?”
- Constant-volume padding
  - Expensive (only tried partially, once)
  - Beatable by an active attacker

Common low-latency mistakes (2)

- Single point of failure
  - Let one server know what you're doing
  - Let one server describe the rest of the net

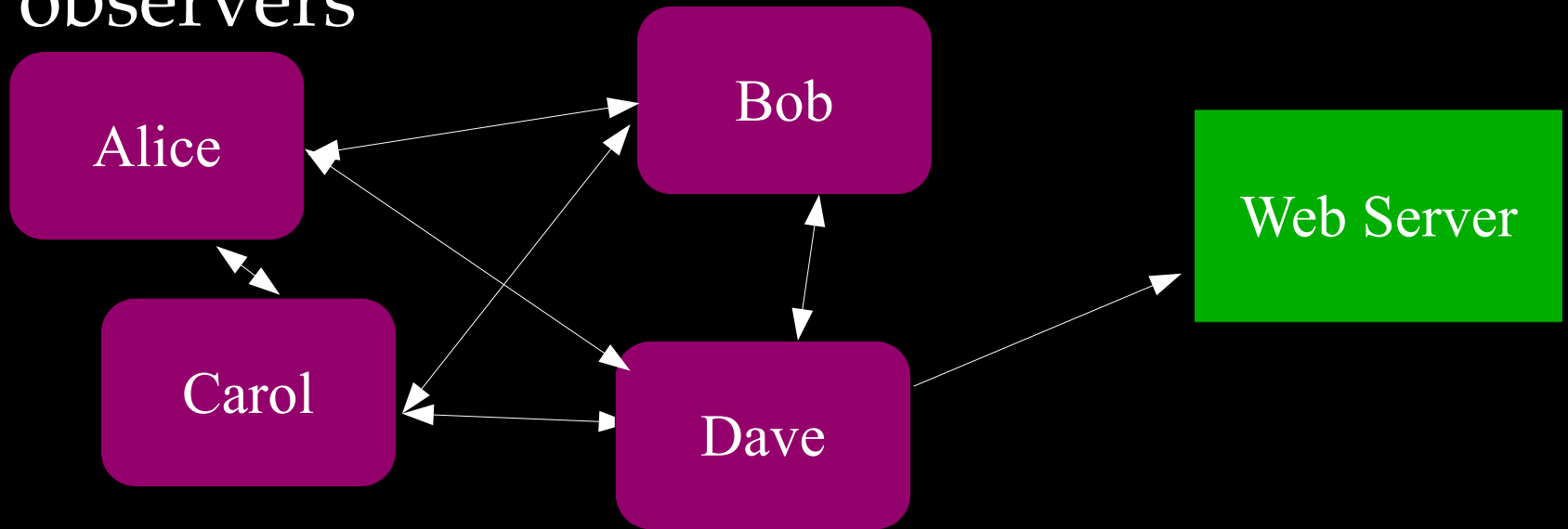
Scrubbing proxies

- First proxy: unknown
- Most famous: Anonymizer.com
- Simple, easy to build.
- Single point of failure, easily correlated.



Crowds (~1996)

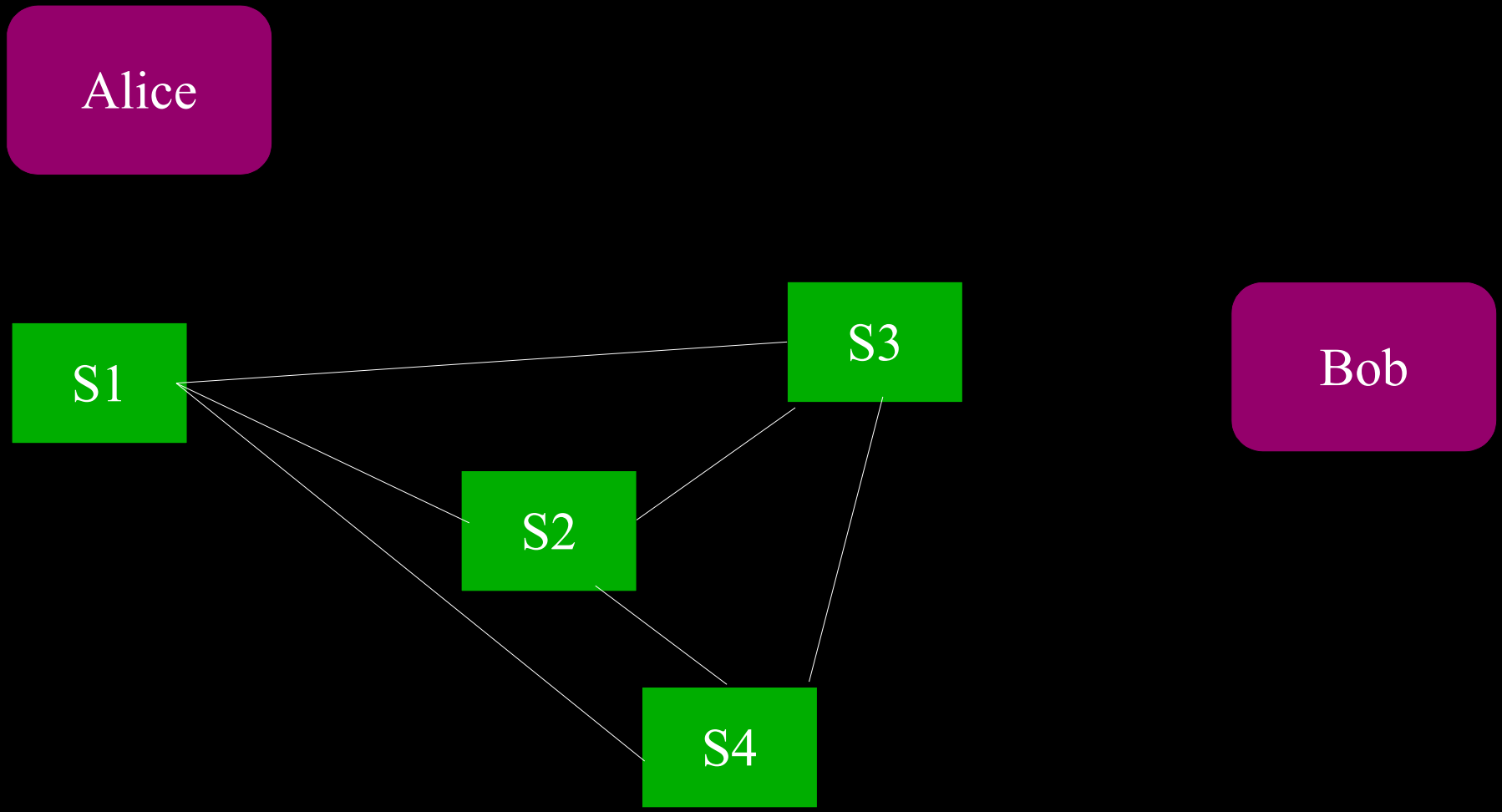
- AT&T Labs group
- HTTP only, code not distributed
- Users relay requests (no encryption!)
- Deniability, not untraceability
- Vulnerable to predecessor attacks, global observers



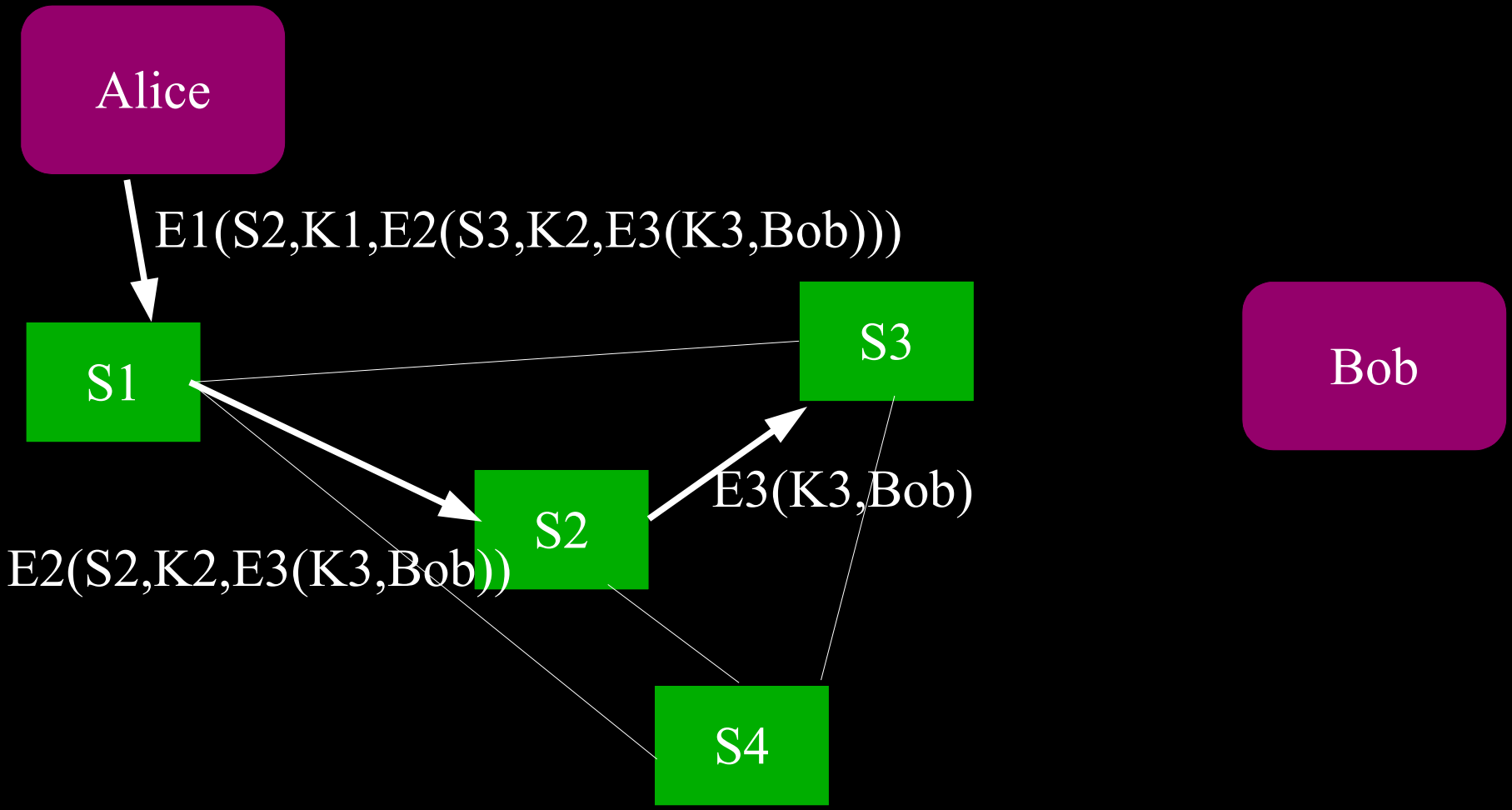
Onion routing v1 (~1996)

- Developed by researchers as US NRL
- Add multiple hops, with encryption at each.
- User picks separate cipher key for each hop.
  - Use expensive PK to establish symmetric keys.
  - Use cheap symmetric crypto for
- Each hop knows only previous and next.
  - No single hop can expose users.
- Separate proxies for each user application on entry and exit.

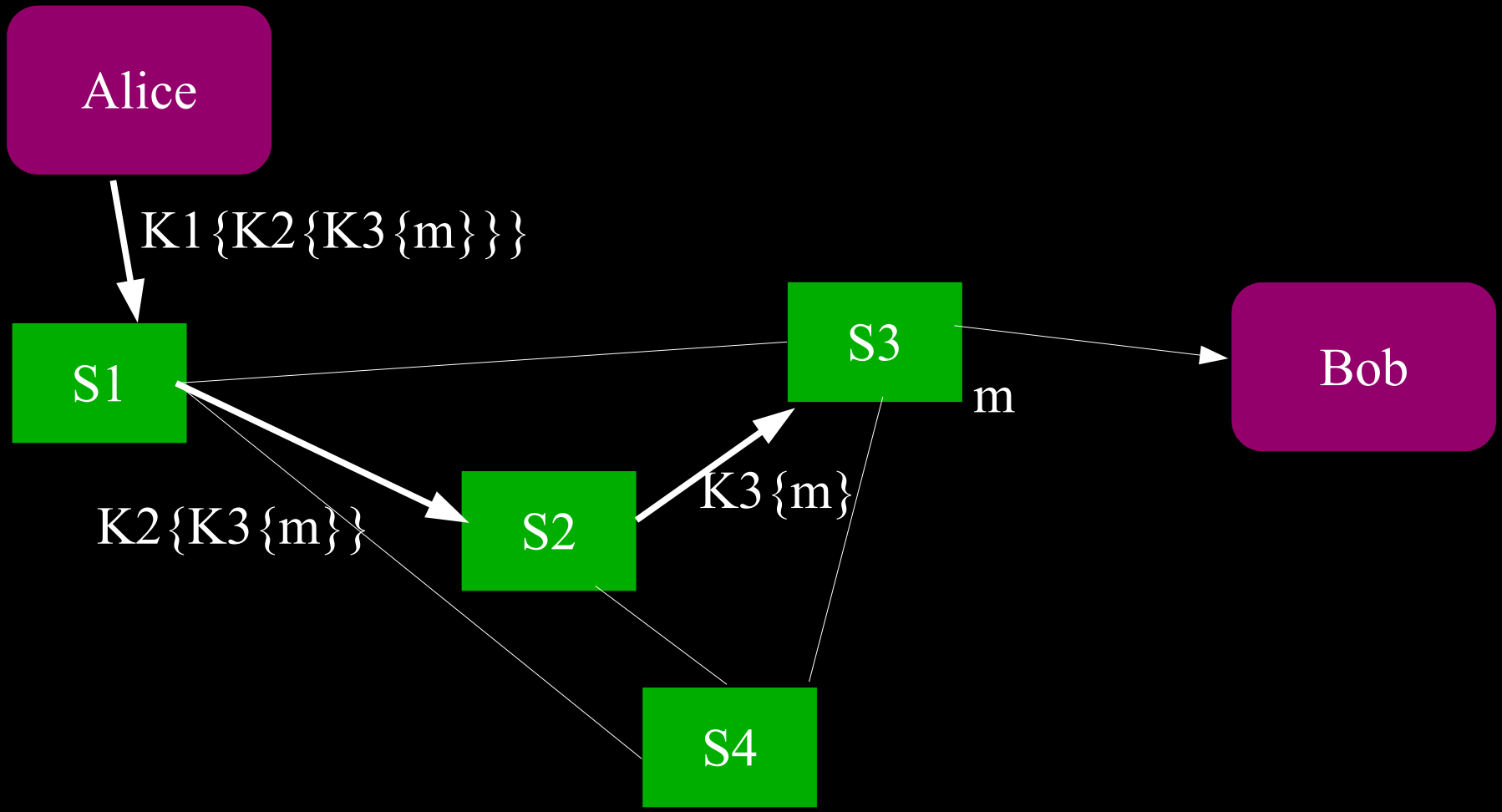
OR v1 details: net of encrypted links



OR v1 details: build a circuit



OR v1 details: use a circuit



OR v1: what happened?

- Small-scale demo, all servers at NRL
- Tech transfer model stoped wider deployment of internal patented code.
- (But see Tor below.)

Freedom (~1999-2001)

- Developed as commercial product by ZKS in Canada.
- Much like OR, but:
  - More effort at efficient routing.
  - Pay-per-service model.
  - Tried padding, briefly.
    - (It was ineffective and uneconomical)
  - Paid ISPs to run servers.
- Shut down in late 2001, probably due to cost problems.
  - Trust model was hard to market.

JAP/WebMIX network (~2000-)

- JAP = Java Anon Proxy
- Uses cascade topology instead of free-route.
  - Users can choose from several cascades.
  - Collecting traffic ensures that many streams share each pipe.
  - But if correlation still works, attack is easier.
- PR problems related to illegal court order.
- Still active, still running, open source.

Tor* (2001-)

*Tor, not TOR.

- Started as OR v2 (or v3?). Key differences:
- Open source (since 2003)
- Build circuits step-by-step, not all at once
  - Forward secure
  - not patented
- Multiplex many streams (TCP requests) over each circuit. (Less PK!)
- Sponsored by NRL, then EFF.

More Tor

- Drops application-specific proxies
  - TCP-over-SOCKS only
- Volunteer-operated: supports servers of different bandwidth and exit support.
- Adds recipient anonymity with hidden services.
- GUI-independent (contest for GUI ongoing)
- Only low-latency network to date with a demonstrably useful spec.

Others

- Invisible IRC/I2P
  - Focus is on closed network, with exit support as extra.
  - Voodoo padding. (Unimplemented?)
    - They claim this will fix end-to-end correlation
  - Other unanalyzed and under-defined (but interesting!) claims.
    - I wish they'd publish.
- Questionable proxy aggregators
  - too many to list

From academia: what was useful?

- New attacks
  - Blending, disclosure, tagging, interference, correlation, predecessor, partitioning.
  - Statistical attacks and defenses.
  - Whole-network analysis.
- Proof that some defenses don't help
  - Like many kinds of padding, extra hops in low-latency networks .

What wasn't?

- Provable shuffles (no point so far)
- One-shot schemes (no applications yet)
  - But maybe for voting
- Closed-userbase networks (same)
- Bandwidth-heavy systems (on today's net)
  - Heavy padding, DC-nets, and so on
- Micro-network analysis
  - But maybe it will scale

Lessons from commerce

- People will buy junk
  - You can compete successfully in the proxy farm or proxy-aggregation market on the basis of pretty marketing and a nice GUI.
  - Even anonymizer isn't very strong.
- Selling good anonymity is hard
  - "Trust us: you don't need to trust us!"
  - Bandwidth, costs will be higher.
  - But corporations, government agencies, and many citizens will, in fact, pay.
- Cryptographers and programmers are expensive.

Lessons from development (1)

- Systems without a specification or design...
  - ...get less academic attention.
  - ...never get compatible implementations.
  - ...are really hard to write slides about.
  - ...often fragment once original developers move on.
- Some mistakes are common if you don't read the literature.
  - Patitioning, voodoo padding, weird notions of deniability, compromise by bad first node.

Lessons from development (2)

- But academia can overlook important issues.
  - Server discovery and knowledge partitioning
  - Anti-blending strategies
  - Reply/forward distinguishability
  - Possibility of volunteer servers

Lessons from deployment

- Usability is a security parameter.
  - Users make mistakes.
  - Users think they're anonymous when they aren't.
- Abuse is (only) a moderate problem
  - Server operators must tolerate complaints, and sometimes threats, but seldom more for abuse.
  - The worst legal attacks against operators have been themselves abusive. (JAP, xs4all.)
  - Serious crime is rare; the world has not ended.

Questions?

- Mixminion: http://mixminion.net/
- Tor: http://tor.eff.org/
- Anonymity bibliography: http://freehaven.net/anonbib/