Mixminion: A next-generation anonymous remailer

George Danezis
Roger Dingledine
Nick Mathewson

Outline

- Background
- Related systems
- A few improvements over past work
- Secure single-use reply block mechanism

Anonymous, message-based communication

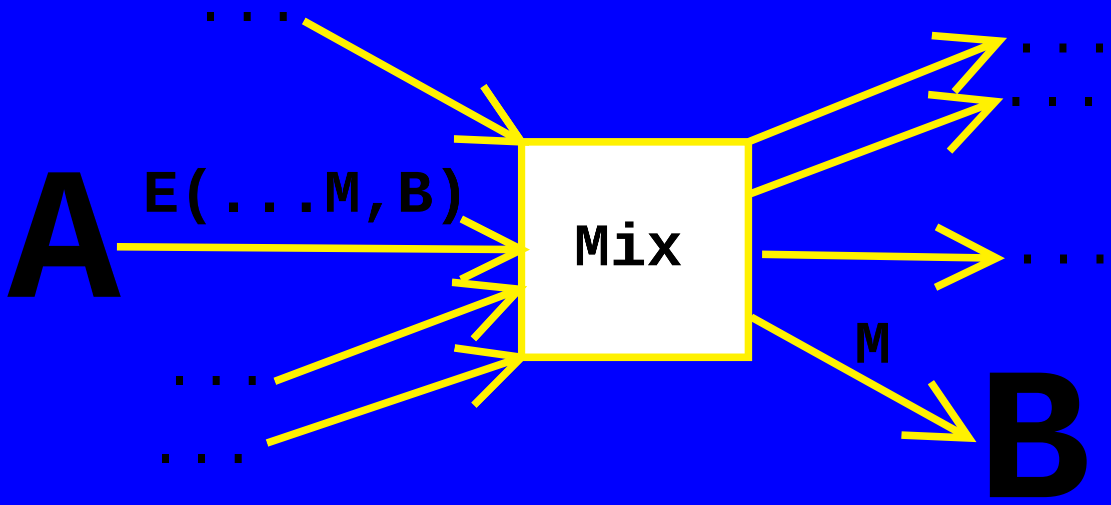
- Forward messages, only Alice remains anonymous
- Direct replies, only Bob remains anonymous
- Anonymized reply messages where Alice *and* Bob remain anonymous

Threat Model (we hope)

- Global passive adversary: can observe all links
- Controls some of the nodes/links
- Can send, modify, delay, etc some messages

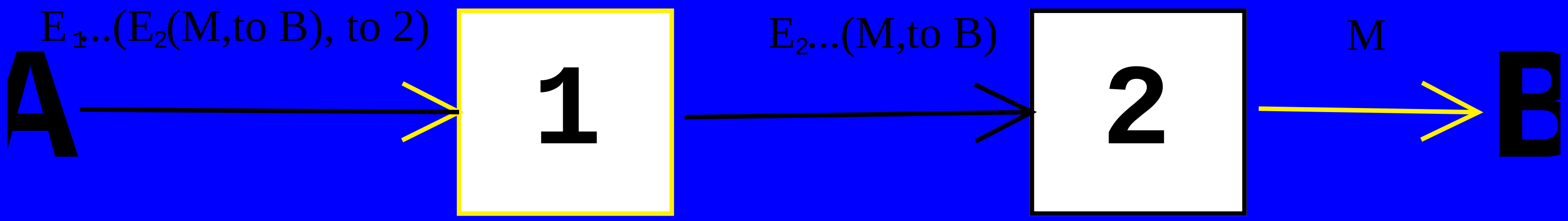
We are not real-time, fast, packet-based, or steganographic.

Basic building block: Mix



A mix batches, decrypts, and reorders messages

Multiple Hops



E_1...(E_2(M,to B), to 2)

A

1

E_2...(M,to B)

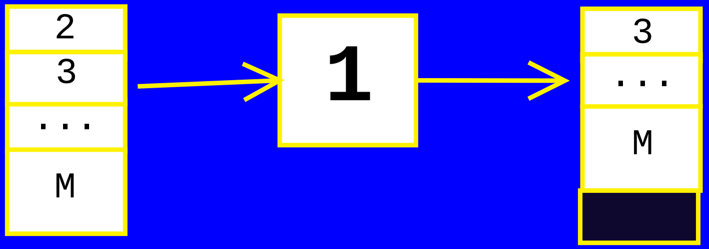
2

M

B

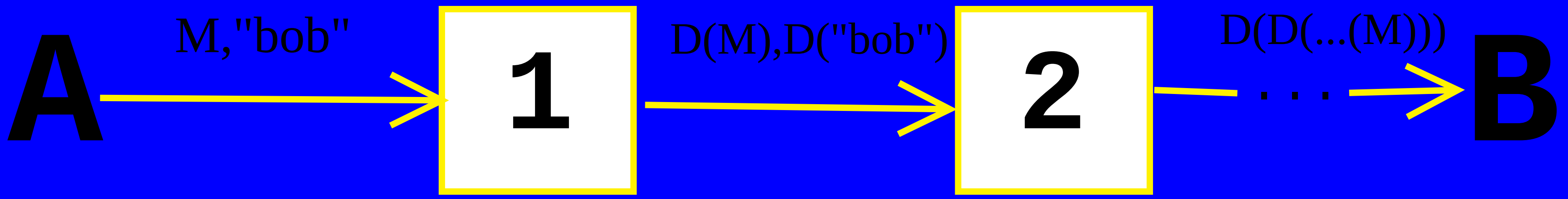
Assume not all hops will collude and reveal **A**

Fixed length messages by re-padding



- **Add random junk to the bottom to replace the info you strip off. Everything’s encrypted, so it looks ok.**

Reply block



- **"bob"** $= 1, E_1(2, ...E_n(B))$
- **In Mixminion, replies act like forward messages.**

Related systems

- *One-hop:* Anonymizer, hotmail, etc
- *Low-latency:* onion routing, Freedom
- *Remailers:* Cypherpunk, Mixmaster, Babel
- *Other:* flash mix, hybrid mix, provable shuffle, etc

Integrated directory servers

Act as reputation servers too

- Mixmaster's *ad hoc* scheme opens users up to partitioning attacks.
- Directory servers can be out of sync; evil DSs can give out rigged subsets to trace clients.
- DSs must successively sign directory bundles; a threshold of servers is assumed good.

Link encryption for forward anonymity

- Mixmaster uses SMTP for transport
- We use TLS over TCP
- Link encryption and short-term keys stop many attacks

Key rotation /
Replay prevention

- Mixmaster has no built-in key rotation
- ...and sketchy replay detection mechanism
- Solve them together: we keep hashes of all messages seen since the last key rotation.

Tagging attack on headers

- Mixmaster/Babel headers have a hash to integrity-check that hop. Doesn't check the rest of the header!
- We can flip some bits later in the header. If we own the hop that corresponds to the part we just broke, we can recognize the message.
- So we make the hash cover the entire header.

And payload too...
But you can’t know the payload when writing a reply block!

- Forward messages want hashes, and replies can’t have them.
- If replies are rare relative to forwards, replies are easy to track.

Messages have two headers and a payload

Build a path out of two legs, one for each header

- For forward messages, Alice makes both legs
- For direct replies, Alice can use the reply block directly
- For anonymized replies, Alice makes the first leg and uses Bob's reply block for the second.

Legs are connected by the Crossover Point

- One of the hops in the first header is marked as a *crossover point*

- At the crossover point, we decrypt the second header with a hash of the payload, and then swap the headers.

Forward messages are anonymous:

- If the second header or the payload are tagged in the first leg, then the second header is unrecoverable.
- If tagged in the second leg, we've already gotten anonymity from the first.

Replies are anonymous:

- The adversary can never recognize his tag.

Multiple-message tagging attacks

- If Alice sends multiple messages along the same path, Mallory can tag some, recognize the pattern at the crossover point, and follow the rest.

- Only works if Mallory owns the crossover point.

- Fix: Alice spreads over k crossover points (and hopes Mallory doesn't own most of them)

Nymservers and single-use reply blocks

- Work like pop/imap servers
- User anonymously sends a bunch of reply blocks to receive the mail that's waiting for him.

Future work

- Dummy traffic policy
- Exit abuse
- Directory servers
- Synchronous batching
- More analysis!

Play with our code

http://mixminion.net/

(Code, mailing list, design, spec)

Do you want to run a server?