Reliable MIX Cascade Networks through Reputation

Roger Dingledine, Reputation Technologies
Paul Syverson, Naval Research Lab

The Problem

The current remailer infrastructure is unreliab

This unreliability decreases anonymity

- many dropped/repeated messages
- fewer users

Ways of Improving Reliability

Build protocols with provable robustness guarantees

Provide economic incentives for reliability

Add reputation to \improve" reliability

Distinction between reliability and robustness

Related Work

MIXes (Chaum)

Robust MIX-nets (Flash Mix, Universally Veri able MIX)

Deployed Remailer Systems (cypherpunks, Mixmaster)

Remailer statistics (Levien's statistics, Jack B Nymble 2)

Threat Model | Adversary can:

Passively read *all tra c*

Compromise some fraction of the MIXes
(Insert, modify, delay, or drop messages)

Previous paper at Info Hiding 4

MIXes write per-hop receipts to prove good service; witnesses verify and tally failure claims.

But:

Global witnesses are trust and communication *bottlenecks*

Owning high reputation nodes means you own more paths?

What's a MIX cascade?

Fixed path through the MIX network

Longer cascades *)* lower chance all bad nodes *)*
more anonymity

Longer cascades *)* lower chance all good nodes *)*
less reliability

Cascades provide more defense against *intersection attack*.

Design Overview

Cascades rearrange periodically (e.g., daily)

A node fails its own cascade if it detects misbehavior

Nodes send test messages to monitor their cascades

Senders can demonstrate decryptions to show failure

Communal Randomness

Goal: collaborating nodes cannot predict the cascades

Centralized (but verifiable) for convenience

All nodes commit, then all reveal

But nodes can influence communal value by not revealing?

Heuristics for picking cascades

Increase cost of breaking anonymity
(pick randomly frn8n(9rlomly)-no

Increase cost of breakingy

Attack: Creepe589a/F15 (Death)]TJ/F219.833 Tf

Need to limit number of bad nodes in network

Proof of work, proof of bandwidth not strong enough

Advogato trust metric:
Number of bad nodes certi ed is based on number of confused nodes (good nodes that might certify bad nodes)

Certify by trustworthiness, not expected performance

So how do we choose cascades?

Pick a target safety factor S (eg 1 in 10^5 paths bad)

Choose first cascade randomly from large enough pool of high-reputation nodes

Replace chosen nodes to maintain pool size

When pool contains all remaining nodes, just build remaining cascades randomly

Detecting Misbehavior

Entry point: Incoming messages rejected?

Inside cascade: Messages replaced with dummy messages?

Exit point: Messages not delivered?

Detecting Misbehavior at Entry Point

Alice can send into any node. They all deliver to t0e head.

Thus nodes can insert indistinguishable test messages

Alice gets a receipt (if not, s0e tries elsewhere)

Head publishes batch snapshot (hashes of messages)

If message not in snapshot, receipt proves misbehavior

Detecting Misbehavior at Exit Point

Tail bounces tra c to all nodes. All nodes deliver.

If inserted test message doesn't arrive, somebody failed.

Optimize: if tail collects a delivery receipt, no broadcast.

Test messages

Nodes reuse recipient addresses in test messages

Reusing addresses helps protect against time-based intersection attack

Quality of Service, Resource Management

Nodes send failure messages and hourly heartbeats to Reputation Servers

Users compare advertised QoS and reputation from each

Future Directions

Better bandwidth use

More research on creepingdeath

Adapt to free-route MIX network

Figure out the details of the reputation servers

Reliability metric and mon with E ciencyn151