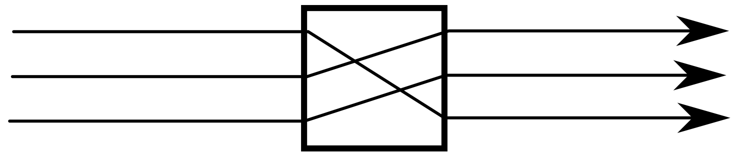
Anonymity

Christian Grothoff
christian@grothoff.org
http://grothoff.org/christian/

"You look at this and you say this is insane. It's insane. And if it is only Hollywood that has to deal with this, OK, that's fine. Let them be insane. The problem is their insane rules are now being applied to the whole world. This insanity of control is expanding as everything you do touches copyrights"
–Lawrence Lessig

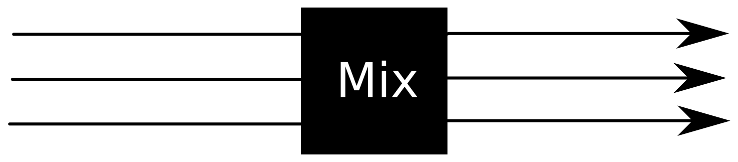
Review: Mixing

David Chaum's mix (1981) and cascades of mixes are the traditional basis for destroying linkability:



Review: Mixing

David Chaum's mix (1981) and cascades of mixes are the traditional basis for destroying linkability:



Agenda

- Definitions and Metrics
- Techniques, Research Proposals and Systems
  - Dining Cryptographers, Mixes
  - **Mixminion**
  - PipeNet, Busses
  - Mute, Ants, StealthNet

Mixminion

G. Danezis, R. Dingledine, D. Hopwood and N. Mathewson describe *Mixminion: Design of a Type III Anonymous Remailer*:

- based on mixmailers (only application is E-mail)
- possibility to reply
- directory servers to evaluate participating remailers (reputation system)
- exit policies

Mixminion: key idea

The key idea behind the replies is splitting the path into two legs:

- the first half is chosen by the responder to hide the responder identity
- the second half was communicated by the receiver to hide the receiver identity
- a crossover-node in the middle is used to switch the headers specifying the path

Mixminion: replay?

Replay attack were already an issue in previous mixnet implementations.

- Mixes are vulnerable to replay attacks
- Mixminion: servers keep hash of previously processed messages until the server key is rotated

⇒ Bounded amount of state in the server, no possibility for replay attack due to key rotation

Mixminion: Directory Servers

- Inform users about servers
- Probe servers for reliability
- Allow a partitioning attack unless the user always queries all directory servers for everything

Mixminion: Nymservers

- Nymservers keep list of use-once reply blocks for a user
- Vulnerable to DoS attacks (deplete reply blocks)
- Nymservers could also store mail (use one reply block for many messages).

Mixminion: obvious problems

- no benefits for running a mixmailer for the operator
- quite a bit of public key cryptography
- trustworthiness of directory servers questionable
- servers must keep significant (but bounded) amount of state
- limited to E-mail

Mixminion: more problems

- exit nodes are fair game for legal actions
- no accounting to defend against abuse
- statistical correlation of entities communicating over time possible (observe participation)
- vulnerable to DoS attacks

⇒ bridging between an anonymous network and a traditional protocol is difficult

Reputation

R. Dingledine and P. Syverson wrote about *Reliable MIX Cascade Networks through Reputation*:

- traditional approach uses external trusted witnesses that probe the mix
- this design allows a mix-cascade to monitor itself

Key idea

- nodes send test-messages to monitor their own cascade
- nodes announce the failure of their own cascade, damaging the reputation of all nodes in the cascade

Reputation: problems

- Reputation of the reporters
- does not detect failure instantly (loss)
- adversary could create fresh identities

PipeNet

Wei Dei suggested *PipeNet*:

- initiator knows receiver identity, but not vice-versa
- layered encryption, forwarding without delay
- constant traffic on each link to avoid observability

Is this useful?

Buses

A. Beimel and S. Dolev introduce *Buses for Anonymous Message Delivery*:

- Anonymity like in the public transportation system.
- A bus is a group of messages traveling on the network.
- Buses travel **fixed scheduled** routes.

Buses: claim to fame

- sender and receiver anonymity
- not based on statistical properties
- communication causes no visible change on the network

Buses: Communicaton Optimal Protocol

- One Bus
- with n^2 seats
- travels on a ring of n nodes.

A message M from p_i to p_j travels as $E_K(M)$ on seat $s_{i,j}$ where K is either a symmetric key known to p_i and p_j or the public key of p_j.

Buses: choices

Any implementation of this basic idea must define three essential properties of the system that are also critical for performance:

- size of the bus(es)
- latency (average number of stations until a passenger reaches his destination)
- number, frequency and routes of the buses

Buses: Reducing the number of seats

The following idea can reduce the number of seats:

- In order to send a message, a node picks a random seat and puts the message there.
- In order to hide that a message was sent, all other seats must be changed.
- Decrypt all seats with the private key of the local host, encrypt seat with message onion-style.

How many seats do we expect to need for m messages?

Buses: Problems with seat reduction

- Each node must perform lots of public key operations, even on empty seats.
- Easy to attack (overwrite all seats with garbage).
- Accidential overwriting makes communication unreliable and introduces the need to send acknowledgments (increasing traffic and latency)

Buses: Reducing latency

Use shortest-path routing:

- assume some graph over the nodes, with a bus traveling on each link in both directions in every time-slot.
- route seats through this graph on the shortest path to the receiver

Buses: Problems with latency reduction

- routing information must be propagated
- seats must have some form of routing header
- large amount of traffic and often empty seats

Buses: question

The bus schedule is known (or predictable).

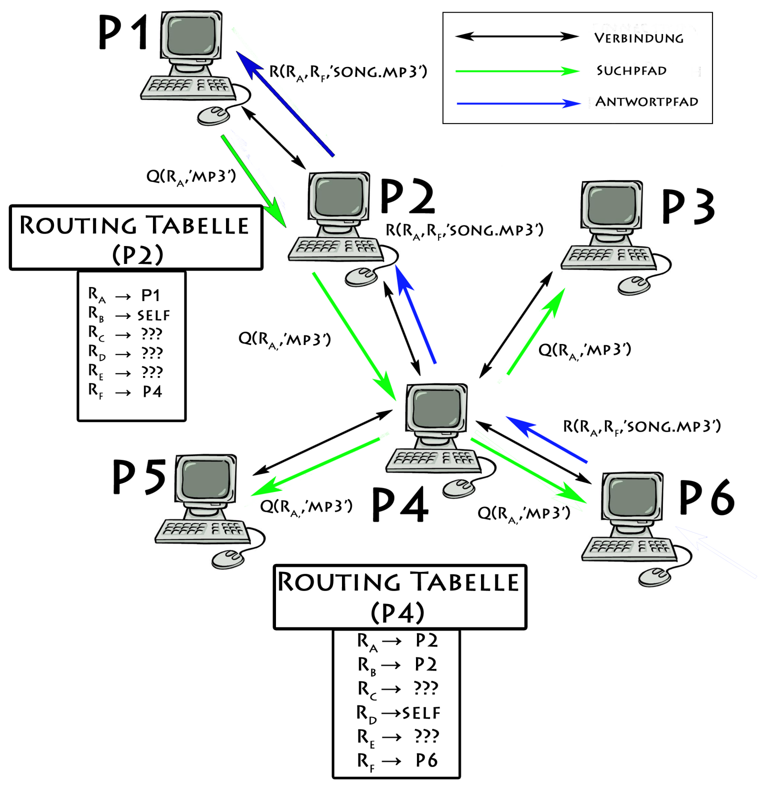
Supposed the adversary is also the recipient of a message.

What can an active adversary do?

Buses: other problems

- scalability questionable ($O(n)$ and worse)
- potentially lots of noise (empty seats)
- many variations with individual benefits and drawbacks
- How is this better than broadcast?

RShare/StealthNet



Routing Tabelle (P2)

R_A →	P1
R_B →	self
R_C →	???
R_D →	???
R_E →	???
R_F →	P4

Routing Tabelle (P4)

R_A →	P2
R_B →	P2
R_C →	???
R_D →	self
R_E →	???
R_F →	P6

Mute/Ants

Properties that a search-limiting mechanism should have:[1]

1. Single Integer Representation
2. Distributed Enforcement
3. Total Limit
4. Deterministic
5. Account for Branching
6. Account for Results

[1]according to Mute author Jason Rohrer

Utility Counters

UC starts at zero. Without hop counter:

$$UC_{new} = UC_{old} + \alpha * |localResults| + \beta * |forwardSet| + \gamma$$

Improved formula with hop counter:

$$UC_{new} = UC_{old} + \alpha * |localResults| * HC + \beta * |forwardSet|^{1+\frac{1}{HC}} + \gamma$$

What is the impact of using UCs on anonymity?

Mute Sender Anonymity

Use a hybrid approach for flodding:

- Initiator picks random 20-byte SHA1 hash value
- Each hop re-hashes the current value
- If last bytes is ≤ 51, switch to utility counters

Does this solve the problem?

Mute Responder Anonymity

Use a third approach for the end:

- Forward with branching until UC hits the limit
- Then switch to chain mode
- Each node on startup once determines an operational mode n with probability $p(n)$, and in chain mode forwards to the same n neighbours, where:

$$p(n) = \begin{cases} \frac{3}{4} & n = 0 \\ 2^{-n+2} & n > 0 \end{cases} \tag{1}$$

Does this solve the problem?

Off System

- Claims to be “anonymous”, but has no mechanisms
- Splits files in 128k blocks, XORs them for encryption (possibly also with a key)
- URLs specify which blocks (and keys) need to be XOR’ed for decryption
- Same encrypted block becomes part of many files

“Owner-Free refers both to the fact that nobody owns the system as a whole and nobody can own any of the data blocks stored in the system.” –Off Introduction

Our Legal System: What Colour are your bits?[2]

- "In Paranoia, everything has a colour-coded security level (...) and everybody has a clearance on the same scale. You are not allowed to touch, or have any dealings with, anything that exceeds your clearance. If you're a Red Troubleshooter, you're not allowed to walk through an Orange door. Formally, you're not really supposed to even know about the existence of anything above your clearance. Anyone who breaks the rules is a Commie Mutant Traitor, subject to the death penalty."

[2]http://ansuz.sooke.bc.ca/lawpoli/colour/2004061001.php

What Colour are your bits?

- “And very much of intellectual property law comes down to rules regarding intangible attributes of bits — Who created the bits? Where did they come from? Where are they going? Are they copies of other bits? Those questions are perhaps answerable by ”metadata”, but metadata suggests to me additional bits attached to the bits in question, and I’d like to emphasize that I’m talking here about something that is not properly captured by bits at all and actually cannot be, ever. Let’s call it ”Colour”, because it turns out to behave a lot like the colour-coded security clearances of the Paranoia universe.”

What Colour are your bits?

- “Random numbers have a Colour different from that of non-random numbers. (...) all cryptographers understand that it’s not the numbers that matter when you’re talking about randomness. What matters is where the numbers came from — that is, exactly, their Colour.”

Off System & Colour

- Off: Our bits are the XOR of two (coloured) bits, so they have lost their restrictions (no copyright)

Off System & Colour

- Off: Our bits are the XOR of two coloured bits, so they have no colour (no copyright)
- Law: Your bits are derived from two coloured bits, so you need **both** clearances

Crowds

M. Reiter and A. Rubin introduced *Crowds: Anonymity for Web Transactions*:

- primary application is web-surfing
- each member of the crowd chooses the next hop at random
- communication in the crowd is link-encrypted

Crowds: features

- fewer public key operations than in a mix-net
- bi-directional communication (replies)
- efficiency and high scalability
- simple protocol

Crowds: non-goals

- no anonymity against local eavesdropper
- no responder anonymity
- no effort to defend against denial-of-service attacks (especially not against routers tampering with the indirected data)

Crowds: design

- node joins crowd (by signing up with central $blender$ server), crowd forms one path with a single key for the entire path
- multiple, chained proxies, each proxy either exits or extends with probability $p_f > \frac{1}{2}$
- reply is routed back on the same path

Crowds: local eavesdropper

- there is no noise in the system

$\Rightarrow$ eavesdropper can see that a connection is initiated

- request is routed along static path with one session key

$\Rightarrow$ eavesdropper needs one node on the path for full exposure

Crowds: collaborating jondos

Suppose c out of n jondos are collaborating and p_f is the indirection probability. **Theorem 5.2:** If

$$n \geq \frac{p_f}{p_f - \frac{1}{2}} \cdot (c + 1)$$

the probability that the a collaborating jondo is the first node that the initiator connects to is lower than 50%.

Crowds: An attack[3]

The adversary may be able to deduce the initiator over time

- if an adversary controls one or more members of the crowd and
- if the protocol has multiple interactions between initiator and responder that can be correlated and that take different paths, since the initiator has a higher probability to be the sender of a query than all other nodes

[3]See also: M. Wright, M. Adler, B. Levine and C. Shields: *An Analysis of the Degradation of Anonymous Protocols*

Crowds: solution

- try to use only one static path
- paths must change when new jondos join
- *solution:* new jondos must join in groups, controlled by the central registration server

Crowds: scalability

Since the amount of traffic a node receives depends only on p_f, not on the size n of the crowd, scalability is great.

The requirement that all paths must be re-formed whenever nodes join is *much* worse, especially since the anonymity of the system depends on large crowds.

Crowds: choosing p_f

- The network load on an individual jondo does not change at all if that jondo changes the parameter p_f.
- Since the last jondo on a path must decrypt, it is optimal for CPU load to choose $p_f = 0$.
- If a jondo chooses $p_f = 1$, this is optimal for the rest of the crowd (jondo becomes a proxy!).
- If the jondo then additionally follows the Crowd requirement to indirect its own requests, they are trivially detectable and the jondo itself is exposed.

Crowds

Do you see any problems?

Crowds: Problems

- no search, no responder anonymity
- lower indirection probability benefits only other nodes
- exit nodes are fair game for legal actions
- no accounting to defend against abuse
- no possibility to trade efficiency for anonymity
- large routing tables needed at nodes, not easy to bound

Tarzan

M. Freedman, E. Sit, J. Cates and R. Morris wrote *Introducing Tarzan, a Peer-to-Peer Anonymizing Network Layer*:

- generic Crowds implementation (tunnel)
- main difference from Crowds: onion routing style encryption
- goal: middleware (integrate into kernel!)

Hordes

C. Shields and B. Levine introduced a variant of Crowds: *A Protocol for Anonymous Communication Over the Internet*.

- use the common Crowds protocol on the forward-path
- use multicast (groups) for the response

⇒ faster response (no indirections)

⇒ less state in the jondos (really?)

Hordes

Do you see any problems?

Hordes: Problems

- multicast is non-trivial — Hordes discounts that multicast groups also must be managed
- multicast partitions the network, allowing partitioning attacks that can be combined with the probabilistic attack on Crowds

Salsa & AP3

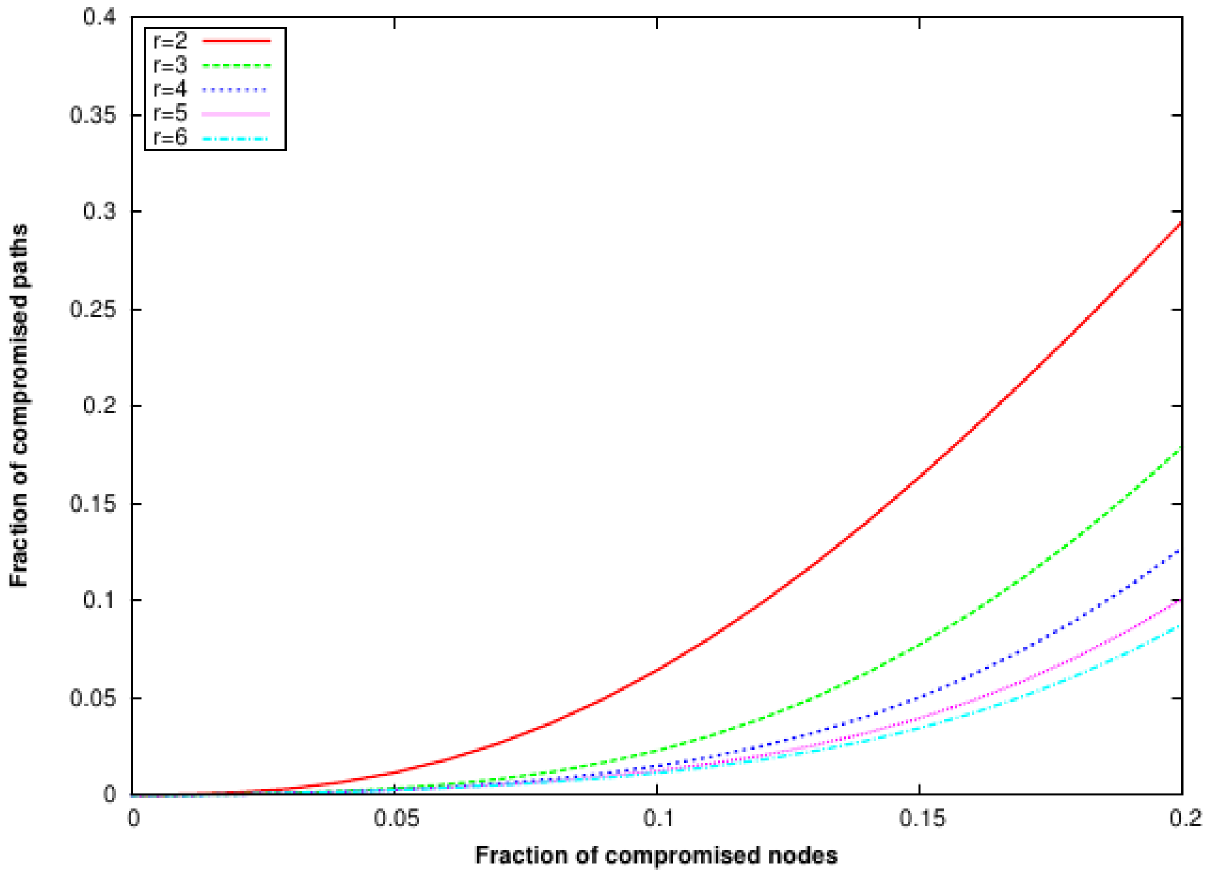
- Goal: eliminate trusted blender server
- Idea: Use DHT (AP3: Pastry, Salsa: custom DHT) to find peers
- Sybil defense with trusted authority (AP3) or IP-based hash (Salsa)

Attacks on Salsa & AP3[4]

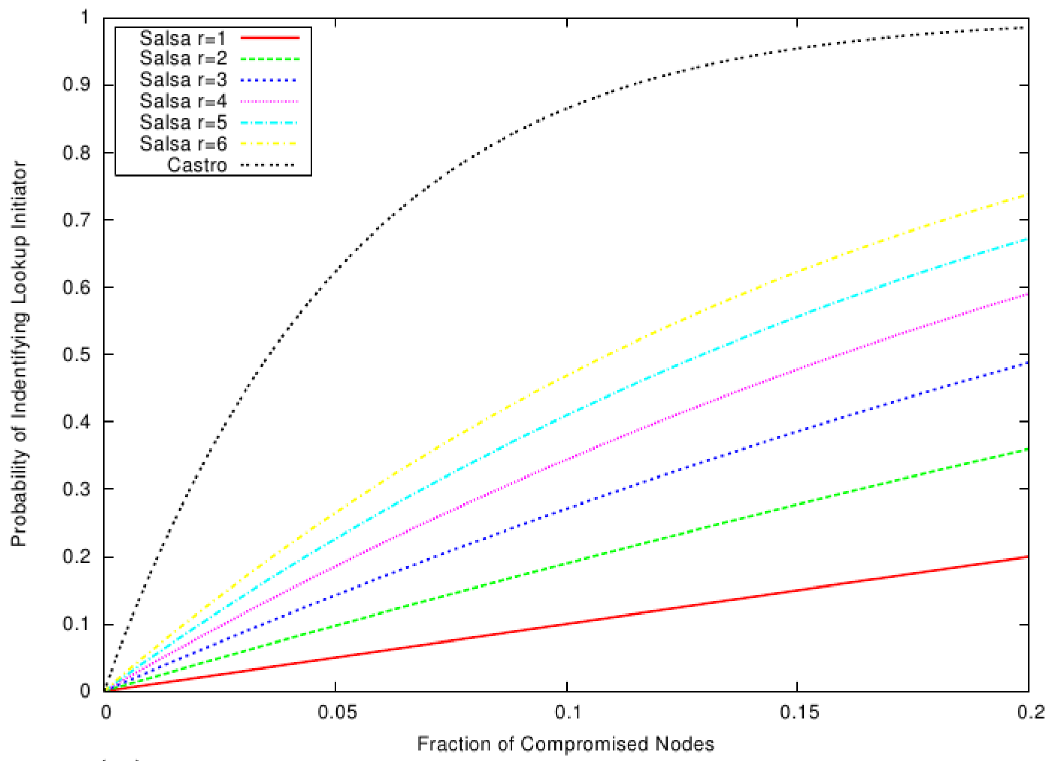
- Passive attack: detect lookup, then correlate with path construction later
- Active attack: return malicious peers during lookup

[4]See: Information Leaks in Structured Peer-to-Peer Anonymous Communication Systems by P. Mittal and N. Borisov

Defenses against Active Attack: Redundant lookups

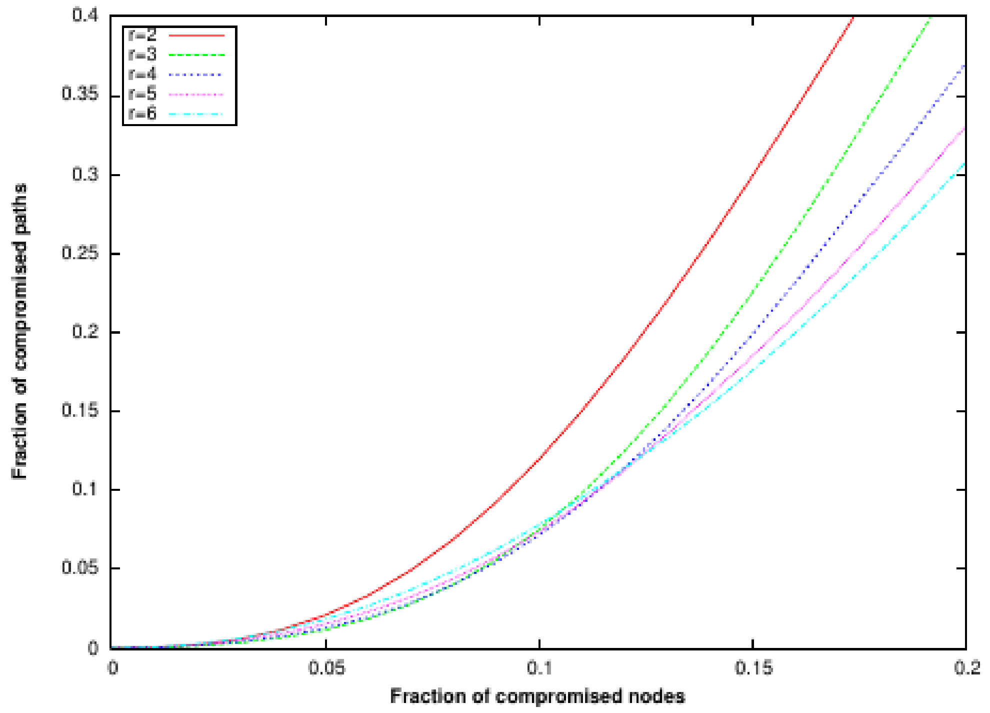


Defenses against Passive Attack: Minimize lookup footprint



(a) Information leak from secure lookups

Defending against the Combined Attack



$r = 3$ is optional for $f < 0.1$,then $r = 6$ becomes optimal.

P5: P2P Personal Privacy Protocol

R. Sherwood and B. Bahattacharjee describe P5, *a Protocol for Scalable Anonymous Communication* over the Internet.

- P5 uses mixes to achieve sender anonymity
- *broadcast* in P5 means application level broadcast

P5: claim to fame

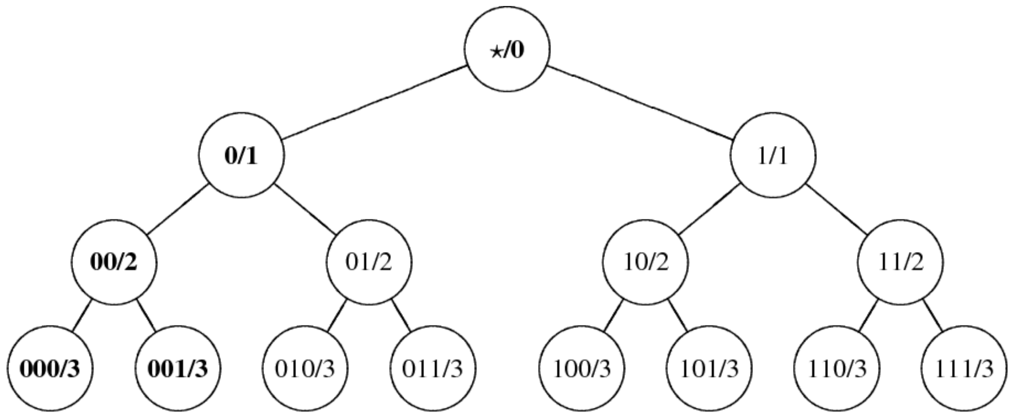
- sender-, receiver-, sender-receiver anonymity
- individual participants can trade-off anonymity for efficiency
- scalability

P5: modeled after global broadcast

- Initiator knows public key of the recipient
- Broadcast message encrypted with that key, all other nodes receive only garbage
- Initiator anonymity can be achieved by participating in a mix

P5: broadcast tree

Instead of a global broadcast, broadcast groups (b/m) are used. A message to $(00/2)$ goes to the nodes shown in boldface:



b is chosen by hashing the user's public key, m randomly by the user under consideration of a local security policy.

P5: routing

- The broadcast tree is typically *not* used for communication
- Peers create additional *routing keys*, use those for joining additional groups in the broadcast tree
- Peers advertise the set of channels they have joined in their groups

$\Rightarrow$ Routing keys generate "lateral" edges in the tree, enabling DHT-style greedy routing

P5: Trade-offs

- Communicating parties may choose to give additional information about their choice of m to each other (after initial communication via $(b/0)$

- Fixed amounts of noise are sent to mask activity, all packets have the same size

P5

Do you see any problems?

P5: Mob attack

- Peers can choose larger exposed value of m for efficiency
- Malicious mob can join same channel as A and reduce efficiency of the channel

$\Rightarrow$ A might then expose additional bits of b, reducing anonymity

P5: Topology

- “In our simulations, each user can determine the number of people in a group by consulting an *oracle* which maintains an up to date list of channel memberships”
- “In implementation, this information can be maintained in a secure distributed manner, either by the underlying application-layer multicast primitive, or at a well-known centralized *topology server*”

P5: problems

- sender anonymity via mixes and dummy traffic generates lots of noise
- noise is broadcasted and must be decrypted
- lots of public key cryptography
- relies on trusted topology servers
- nodes can increase anonymity **at the expense of others**

P5: min or problems

- How does the initiator obtain the public key of the recipient?
- Mob attack

DC-Net

- Based on Dining Cryptographers
- Instead of pairwise coin-flip, use secure PRNG with shared seed
- Transmission slots and reservations protect against collisions

$\Rightarrow$ Best anonymity guarantees (for given n)

$\Rightarrow$ Traffic is $O(n^2)$ (n broadcasts/round)

- Detection of jammers possible whp for $O(n)$ in computational cost

CliqueNet / Herbivore

E. Sirer, M. Polte and M. Robson designed *CliqueNet: A Self-Organizing Scalable, Peer-to-Peer Anonymous Communication Substrate*:

- based on DC-Nets
- more scalable, peer-to-peer and robust
- presumably supports TCP on top of CliqueNet
- Published 2004, to be released soon...

CliqueNet: goals

CliqueNet claims to achieve:

- **Strong Anonymity**, building on information-theoretic guarantees of DC-nets
- **High Scalability**, no significant performance loss if more nodes join
- **Robustness**, CliqueNet provides irrefutable, non-forgeable proofs to identify disruptive nodes

CliqueNet: The ideas

- automatically partition the network into smaller DC-nets of sizes between 3 and 5 participants
- some nodes, *ambassadors*, join multiple cliques for communication between cliques
- malicious hosts are detected and framed in a “distributed database”
- exponential back-off of round frequency if nothing is transmitted

Herbivore: details

- Pastry used for global organization
- Proof-of-work required for node to select virtual position
- Nodes maintain local (!?) *strike table* with misbehavior used to eliminate nodes from a clique
- Replication of documents used to defeat intersection attacks for file-sharing

CliqueNet: Problems

- small cliques provide little protection
- large cliques are rather expensive
- DC-Net of size 3 has less than 13% utilization
- adversaries that control a clique (more than 50%) can kick out good nodes
- anonymity: unrealistic assumption, adversary can join with 2-4 nodes
- scalability not addressed: inter-clique routing not considered!
- robustness: distributed database, mobs and malicious routers not addressed

Copyright