Anonymous Communication

Techniques to prevent traffic analysis, specifically discovery of source-destination patterns

Historically important
- Traffic patterns were very useful in WWII, for both learning about the enemy and throwing them off (with decoy traffic)

Important today because private content is increasingly being carried by public and private networks
- Internet telephony
- Browsing, shopping, content delivery via Internet
- Pay-per-view movies over cable

Basic Concepts

What do you want to hide?
- Sender anonymity: attacker cannot determine sender
- Receiver anonymity: attacker cannot determine receiver
- Unlinkability: attacker can determine senders and receivers, but cannot determine <sender, receiver> associations

From whom do you want to hide it?
- Local eavesdropper (e.g., your employer)
- Global eavesdropper (e.g., a government or coalition of governments)
- Your communication partner (e.g., a web server)
Uses of an Email Pseudonym Server
[Mazieres & Kaashoek 1998]

- nym.alias.net is an email pseudonym server (one of many)
  - allows anyone to create an email alias without revealing her identity

- Survey of users revealed numerous uses
  - To make political statements, to hide their correspondents, and to encrypt email in countries with oppressive governments
  - Where alternatives might lead to embarrassment, harassment, prosecution, or loss of job
  - Alcoholism, depression, being a sexual minority, whistle-blowing
  - Virus development, software piracy, and other illegal uses
  - For protection from the unforeseen ramifications of a USENET news posting
  - So that statements are judged on their own merit

Simple Proxies

- The most common technology for achieving sender anonymity from communication partner
- Much like network address translation
  - proxy replaces client’s address with its own

- Has been implemented for several protocols
  - HTTP, email, ...

The Anonymizer

- Tailored to HTTP (web) traffic
- Hides user’s address from web server (sender anonymity)

  10.42.6.9:1024 192.123.2.5:2028
  web client anonymizer web server

- Challenge: rewriting links in web pages
  - For example, a web page containing
    <A HREF=http://www.eff.org>
  - is rewritten to
  - Must be done reliably, or anonymity is lost
Weaknesses of The Anonymizer

- Administrator of The Anonymizer knows all
  - Common to all single-proxy solutions
  - Even for TLS-protected connections
- Translating URLs in web page scripts is difficult, if not impossible
  - Failure to translate can expose identity
  - Safest to disable JavaScript
- Cookies must be handled with care
- If browser’s Java security policy permits applet to connect only back to web server, then applet will probably fail

Janus

- Also known as Lucent Personalized Web Assistant (LPWA)
  - No longer operational
- Similar to Anonymizer, but generates pseudonyms, email addresses, and passwords for sites that require accounts
- How it works:
  - Initially the user provides her email address and a password to Janus
  - When at a web site, user types control codes for her account, password, and email address, respectively
  - Janus replaces these codes with pseudonymous ones

An Example Use of Janus

- I give to Janus:
  - Email: “reiter@cmu.edu”
  - Password: “tomato”
- When I visit “www.nytimes.com/subscribe”, I enter
  - Subscriber ID: “U” ← control code for account name
  - Password: “P” ← control code for password
  - Email address: “@” ← control code for email address
An Example Use of Janus (cont.)

- Janus finds “\*”, “\^”, and “\@” and replaces them:
  - “\*” → f(reiter@cmu.edu, “tomato”, “nytimes.com”)
  - “\^” → g(reiter@cmu.edu, “tomato”, “nytimes.com”)
  - “\@” → h(reiter@cmu.edu, “tomato”, “nytimes.com”)

  where f, g, and h are one-way on first two inputs, and the output of h is of the form “xxxx@janus.com”.

- Because f, g, and h are deterministic, future accesses will yield the same account information.

- “xxxx@janus.com” is forwarded to “reiter@cmu.edu”.

Unlinkability Via Mixes

- A “mix” is a special “router”.
- Attempts to hide correspondence between incoming and outgoing messages.

![Diagram of Unlinkability Via Mixes]

- Batches
- Changes order
- Changes encoding

Changing the Encoding

- Change of encoding should be something that only mix server can perform.
  - Usually implemented with a public key cryptosystem.

![Diagram of Changing the Encoding]
Chaining Mixes

- Mixes can be “chained” together
  - If any mix server is trustworthy, then unlinkability can be achieved
  - Defends against compromise of mix servers

Return Addresses

Chaining Return Addresses

- $c_i$ and $d_i$ are encrypted under $K_{Mi}$
  - $d_i$ is decrypted on outbound direction (request)
  - $c_i$ is decrypted on return direction (response)
  - $M_i$ uses $K_i$ to encrypt on return direction
  - $B = M_{i+1}$
Chaining Return Addresses (cont.)

For $i=1, \ldots, n$,

- $e_i = e_{i-1}$
- $d_i = d_{i-1}$
- $M_i = M_{i-1}$
- $A = M_n$
- $B = M_0$

Attacks on Mixes

- **Replay**: Send the same message through twice
  - Time $t$
  - Time $t + d$

  - Possible defense
    - Sender includes timestamp within each “layer”
    - Mix drops each message with expired timestamp
    - Mix keeps copy of each message until its timestamp expires, and refuses to process the same message again

Attacks on Mixes (cont.)

- **Bridging**: Attacker submits all but one mixed message
Attacks on Mixes (cont.)

- Length can disclose correspondence between inputs and outputs

- Possible defense
  - Break/pad messages into fixed-length blocks, and make sure that mix transformation is length preserving
  - Maintain constant amount of communication between each mix server

Sender Anonymity: DC-Nets

Basic idea: for one sender to anonymously send one bit $b$
- each pair of potential senders $s_i, s_j$ share a secret key bit $k_{i,j}$
- actual sender $s_i$ broadcasts
  $b \oplus k_{i,1} \oplus ... \oplus k_{i,i-1} \oplus k_{i,i+1} \oplus ... \oplus k_{i,n}$
- each other $s_j$ broadcasts
  $k_{i,j} \oplus ... \oplus k_{i,j} \oplus k_{j,j} \oplus ... \oplus k_{j,n}$
- XOR of all broadcast messages is $b$

DC-Nets (cont.)

$m_1: 10101$
$k_{1,2}: 00011$
$k_{1,3}: 10111$

$m_2: 00000$
$k_{2,1}: 00011$
$k_{2,3}: 10111$

$m_3: 00000$
$k_{3,1}: 10111$
$k_{3,2}: 01011$
Security of DC-Nets

- Consider the following graph
  - each node represents a participant
  - each edge represents a key shared by its endpoints
  - each participant has a message to send (0 or 1)
  - assume that the graph is connected

- Definition: An anonymity set seen by a set \( K \) of keys is a set of connected vertices in the graph formed by removing the edges corresponding to \( K \).

- Examples:
  - anonymity set seen by \( K = \emptyset \) is \( \{ \text{all vertices} \} \)?
  - anonymity sets seen by \( K = \{ \text{all edges} \} \) for each \( s \in V \)
  - in a complete graph, the anonymity set seen by all keys incident on a set \( S \) of vertices? \( V \setminus S \)
  - in a biconnected graph, the anonymity set seen by all keys incident on one vertex \( s \)? \( V \setminus \{ s \} \)

- Theorem: Any attacker knowing keys \( K \) can learn only the parity of the messages of an anonymity set seen by \( K \).

Ring Networks

- Communication systems based on cycles, called rings, are a common structure for LANs (e.g., token ring)
  - a bit travels around ring from sender to destination
DC-Nets Over a Ring (Round 1)

DC-Nets Over a Ring (Round 2)

Properties of Ring Implementation

- A threefold (at least) decrease in bandwidth compared to one in which messages travel half-way around ring on average
- May incur collisions due to concurrent senders
  - avoid token reservation; may reveal sender
  - collisions detected after full trip around the ring
  - after detection, sender can wait a random time to retransmit
Sender Anonymity: Crowds

- A proxy-based approach developed for web browsing
  - each user joins a "crowd" and runs a local proxy
  - each user request is routed along a random path to destination server
  - each proxy on the path cannot tell if its predecessor is the source or if its predecessor is just passing the request on behalf of another

Main adversaries addressed
- web server
- other crowd members

Crowds Proxy Algorithm

(client, request) -> receive_request():

if [client = browser]
  sanitize(request); /* strip cookies & identifying headers */
if (my_path_id = 1)
  /* if my_path_id is not initialized */
  next[my_path_id] <- Crowd; /* select next proxy at random */
else
  path_id <- remove_path_id(request); /* remove incoming path id */
if (translate[path_id] = 1)
  /* incoming path_id is new */
  coin <- coin_flip(pf); /* tails with probability p */
  if (coin = 'heads')
    translate[path_id] <- 'submit';
  else
    translate[path_id] <- new_path_id(); /* outgoing path id */
  next[translate[path_id]] <- Crowd; /* select next proxy */
if (translate[path_id] = 'submit')
  submit_request(); /* send request to destination server */
else
  forward_request[translate[path_id]]; /* send request to next proxy */

subroutine forward_request(out_path_id)
  send out_path_id||request to next[out_path_id];
  reply <- await_reply(∞); /* wait for reply or recognizable proxy failure */
  if (reply = 'proxy failed') /* proxy failed */
    Crowd <- Crowd \ {next[out_path_id]};
  else
    forward_request(out_path_id);
    /* received reply from jondo */
    send reply to client;

subroutine submit_request()
  send request to destination(request);
  reply <- await_reply(timeout); /* wait for reply, timeout, or server failure */
  send reply to client;
Anonymity Properties

- Anonymity versus web server
  - Proxy at source of request always forwards request to some proxy in the crowd
  - Web server thus receives the request from a Crowd member chosen uniformly at random

- Anonymity versus colluding crowd members
  - Colluding members will suspect who they receive request from
  - Define
    - $H_k$, $k \geq 1$, to be event that first collaborator on path occupies the $k$-th position on the path (where the initiator is in 0-th position)
    - $I$ to be the event that first collaborator is immediately preceded by path initiator
  - Then, we want to compute $P(I | H_{1+})$.

Computing $P(I | H_{1+})$

$$P(I | H_{1+}) = \frac{P(I \cap H_{1+})}{P(H_{1+})} = \frac{P(I)}{P(H_{1+})}$$ since $I \supset H_{1+}$

- We need to compute $P(I)$ and $P(H_{1+})$
- Let $n$ = # crowd members, $c$ = # collaborators

Computing $P(H_{1+})$

- To compute $P(H_{1+})$, let's first compute $P(H_j)$
  $$P(H_j) = \left( \frac{p_j(n-c)}{n} \right)^{j-1} \frac{c}{n}$$
- $P(H_{1+})$ follows from $P(H_j)$
  $$P(H_{1+}) = \sum_{j} P(H_j) = \sum_{j} \left( \frac{p_j(n-c)}{n} \right)^{j-1} \frac{c}{n} = \frac{c}{n - \sum_{j} p_j(n-c)}$$
Computing $P(I)$

$$P(I) = P(H_i)P(I \mid H_i) + P(H_{i-1})P(I \mid H_{i-1})$$

$$P(H_i) = \frac{c}{n}$$

$$P(I \mid H_i) = 1$$

$$P(H_{i-1}) = \frac{1}{n-c}$$

$$P(I \mid H_{i-1}) = \frac{1}{n-c}$$

$$P(I) = \frac{cp_f}{n^2 - np_f(n-c)}$$

Putting it all together …

If we want $P(I \mid H_{i-1}) \leq \frac{1}{2}$, then it suffices for

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

For example, $p_f = 3/4$ and $n \geq 3(c+1)$ implies $P(I \mid H_{i-1}) \leq \frac{1}{2}$

Computing $P(I \mid H_{i-1})$

Putting it all together …

$$P(I \mid H_{i-1}) = \frac{P(I)}{P(H_{i-1})} = \frac{n - p_f(n-c-1)}{n}$$

If we want $P(I \mid H_{i-1}) \leq \frac{1}{2}$, then it suffices for

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

For example, $p_f = 3/4$ and $n \geq 3(c+1)$ implies $P(I \mid H_{i-1}) \leq \frac{1}{2}$

Timing Attacks

Timing attacks arise from the structure of HTML

Some HTML commands elicit an immediate request from browser
Timing Attacks (cont.)

- One approach to prevent timing attacks
  - First and last proxy parse HTML to identify automatic requests
  - First (local) proxy blocks automatic requests from browser
  - Last proxy retrieves pages and sends them along

- Simple parsing may not suffice to get all references to other pages
- Disable active content
- Makes caching less effective

Anonymity Decays Due to Path Linking

- Anonymity decays (versus collaborating proxies) if multiple paths can be linked to the same initiator
  - Linking can occur based on timing, content, etc.
- Initiator precedes first collaborator with higher probability than any other proxy
- So, the true initiator will precede collaborators more often than any other on linked paths
- Over time, this exposes initiator (if paths can be linked)
- Can be delayed by reconfiguring paths very rarely
  - But path reconfigurations are required for a new member to join
- This threat applies to any sender-anonymous system

Receiver Anonymity Via Broadcast

- To hide receiver
  - deliver each message to all nodes (broadcast)
  - label each message so that the addressee and nobody else can recognize it is addressed to her (an implicit address)

- Implicit address
  - vs. explicit address: latter names a place in the network
  - is visible if it can be publicly tested for equality, invisible otherwise
  - is public if known to every user, private if distinct and secret to a particular user
Visible Implicit Addresses

Visible implicit addresses: pseudonyms
- users choose arbitrary pseudonyms for themselves
- pseudonyms are used to label messages
- can be used as private address, but ideally only once
  - multiple uses enables linking of messages to same user

Invisible (and public) implicit addresses can be realized with a public key cryptosystem
- message is addressed by adding redundancy and then encrypting it with addressee’s public key: $E_k(m, h(m))$
- each receiver decrypts all messages, uses redundancy to decide which messages are addressed to it
- can similarly be realized if sender shares a distinct secret key with each receiver