Anonymizers

Class 8
Overview

- Mix Networks
- Re-encryption based Mix Nets
- TOR
What is Anonymity?

- The state of not being identifiable within a set of subjects
  - You cannot be anonymous by yourself!
  - Hide your activities among others’ similar activities
  - Big difference between anonymity and confidentiality
- **Unlinkability** of action and identity
  - For example, sender and his email are no more related after observing communication than they were before
- **Unobservability** (hard to achieve)
  - Any item of interest (message, event, action) is indistinguishable from any other item of interest
Sending Message

Alice → Plaintext ← Bob
Sending Message: Problem

Alice

Plaintext

Bob

Capture Message

Link Alice to Bob!
Solution 1: Encryption

Embed message in "frog"

Link Alice to Bob!

Alice

Bob

Malory
Solution 2: Multiple Routers

Alice → Plaintext → Router 1 → Plaintext → Router 2 → Plaintext → Router 3 → Bob
Solution 2: Multiple Routers

A powerful Malory can still link A to B!
Solution 3: Mix Net

Each router waits until it has accumulated k messages.
Solution 3: Mix Net: Problem

Mallory can still see the messages!
Solution 4: Mix Net & Encryption

Encrypt messages

Alice

``the payer''

``the merchant''

Encrypt messages
Solution 4: Mix Net & Encryption: Problem

Mallory can still follow the messages!
Solution 5: Onion Routers and Source Routing

- Use source routing
  - Alice decides which routers will forward the "frogs"
  - The forwarded frog should differ from received frog
- The outer "frog" swallows all intermediate frogs

- Each intermediate router
  - Waits to receive enough messages
  - Removes the outmost encryption (with its private key)
  - Permutes the inner messages (still encrypted with other keys)
  - Forwards each inner message to next router
Solution 5 (cont’d)

Encrypt message with 3 keys

$K_1$, $K_2$, $K_3$

Router 1  Router 2  Router 3

decrypt message  decrypt message  decrypt message
Solution 5 (cont’d)

Encrypt message with 3 keys

$K_1$, $K_2$, $K_3$

$= E_{K_1}(R_2, E_{K_2}(R_3, E_{K_3}(Bob, E_{Bob}(M))))$
Solution 5 (cont’d)

Embed message in “frog”

$K_1, K_2, K_3$

$= E_{K_2}(R_3, E_{K_3}(Bob, E_{Bob}(M)))$
Solution 5 (cont’d)

Embed message in "frog"

\[ E_{K_3}(Bob, E_{Bob}(M)) \]

K₁, K₂, K₃

Router 1

Router 2

Router 3
Example: TOR

https://www.torproject.org
How TOR Works

Step 1: Alice’s Tor client obtains a list of Tor nodes from a directory server.

Courtesy of EFF
How TOR Works (cont’d)

Step 2: Alice’s Tor client picks a random path to destination server. **Green links** are encrypted, **red links** are in the clear.

Illuminated by EFF
How Tor Works (cont’d)

Step 3: If at a later time, the user visits another site, Alice’s tor client selects a second random path. Again, green links are encrypted, red links are in the clear.

Courtesy of EFF
TOR Cells

- Communication uses TCP
- All data is sent in fixed size cells
- Control Cells
  - Create and destroy a circuit
- Relay Cells
  - Carry end-to-end stream data
  - Control stream, data, open/close stream, extend circuits
TOR Circuits

- Describe the Onion Routers on the path
- Can be used by many TCP streams
- Built incrementally
Building a Circuit

1. Create $c_1$, $E(g^{x_1})$

2. Created $c_1$, $g^{y_1}$, $H(K_1)$

3. Relay $c_1$ (Extend, OR2, $E(g^{x_2})$)

4. Create $c_2$, $E(g^{x_2})$

5. Created $c_2$, $g^{y_2}$, $H(K_2)$

6. Relay $c_1$ (Extended, $g^{y_2}$, $H(K_2)$)
Additional Functionality

- Integrity checking
  - Only done at the edges of a stream
  - SHA-1 digest of data sent and received
  - First 4 bytes of digest are sent with each message for verification