Crowds
Anonymous Web Transactions

Why anonymity?

- The web contains a wealth of information on topics that you might want to explore privately
- Support groups
  - victims of crime
  - private health concerns
- Job search
  - don't want to inform current employer of search
- Concerned with private retrieval, not publishing

Privacy on the web … NOT

- Browsers advertise
  - IP address, domain name, organization, referring page
  - platform: O/S, browser
  - which information is requested
- Information available to
  - end servers
  - local system administrators
  - other third parties (e.g., doubleclick.com)
- Cookies (not so sweet)

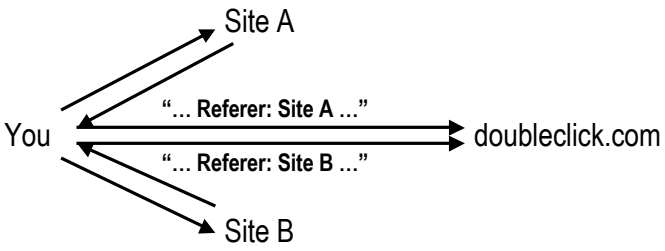
Example

- A typical HTTP request

GET http://www.amazon.com/ HTTP/1.0
User-Agent: Mozilla/3.01 (X11; I; SunOS 4.1.4 sun4m)
Host: www.amazon.com
Referer: http://www.alcoholics-anonymous.org/
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Cookie: session-id-time=868867200; session-id=6828-2461327-649945; group_discount_cookie=F

Example: doubleclick.com

- Numerous sites link to ads at doubleclick.com
- Due to Referer: field, doubleclick may capture your whole click-stream!



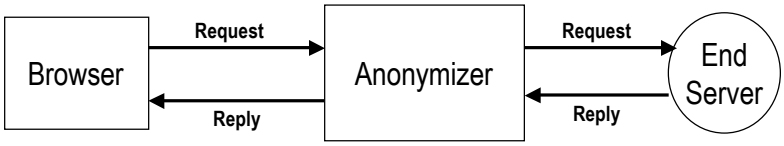
Online privacy in the press

Facets of anonymity

- Adversaries
  - Eavesdroppers
    - local (system administrators)
    - global (backbone administrator)
  - Active attackers (local, global)
  - End servers, other users
- Properties
  - Sender anonymity
  - Receiver anonymity
  - Unlinkability of sender and receiver

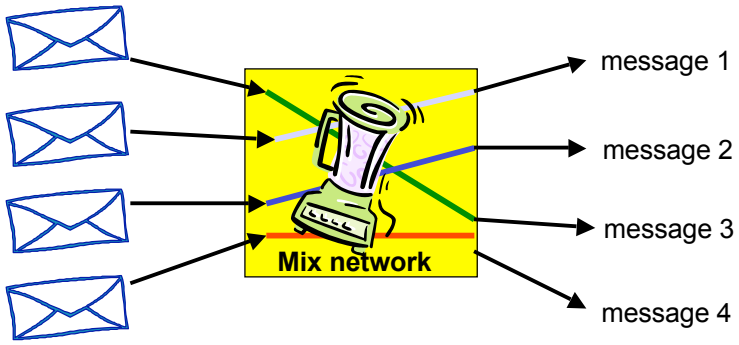
The Anonymizer

- Acts as a proxy for users
- Hides information from end servers



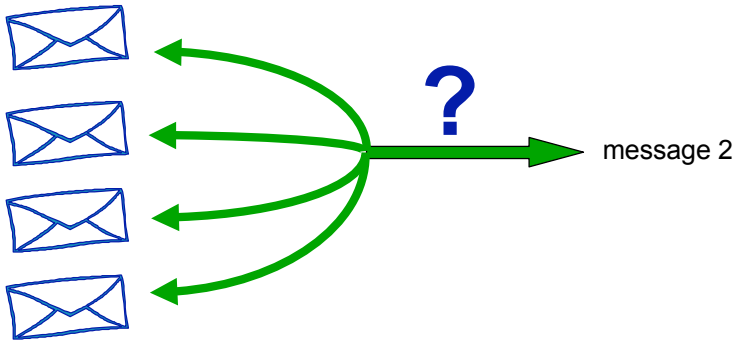
- Sees all web traffic
- Adds ads to pages

What does a mix network do?



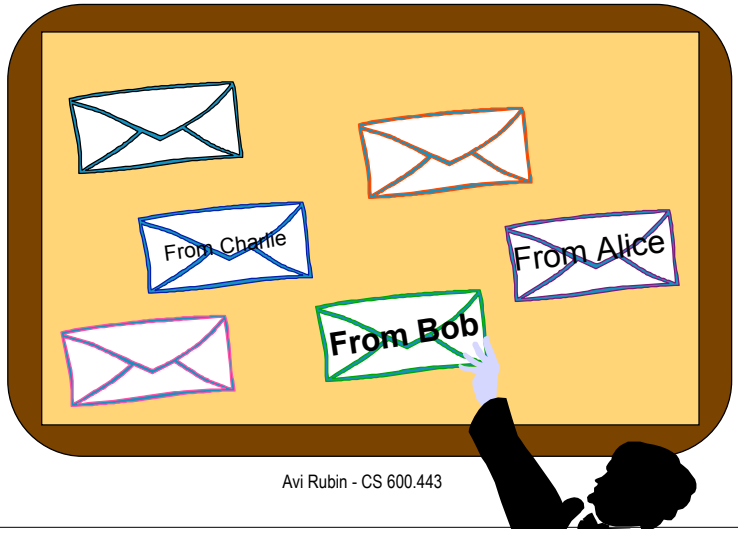
Randomly permutes and decrypts inputs

What does a mix network do?

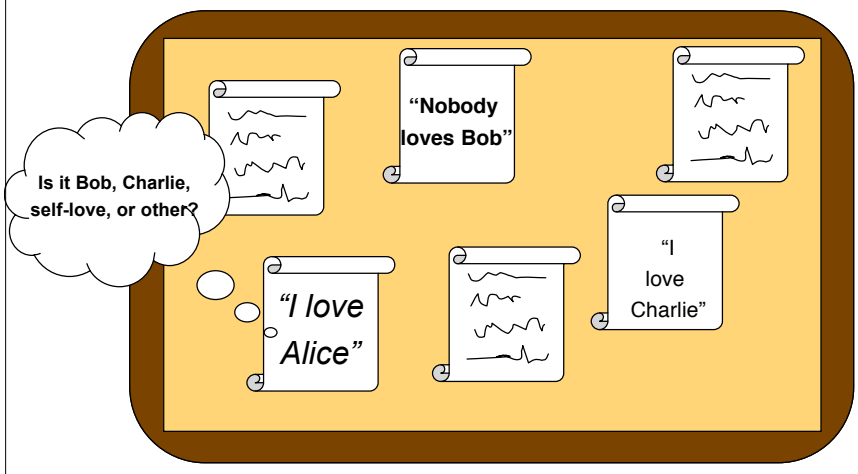


Key property: We can't tell which ciphertext corresponds to a given message

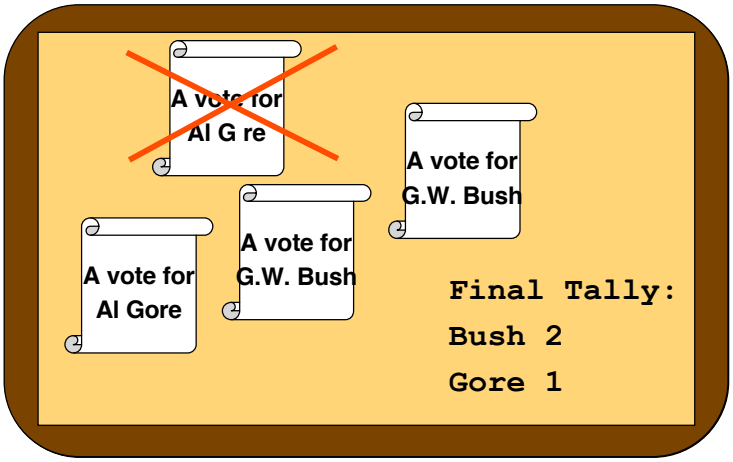
Example application: Anonymizing bulletin board or e-mail



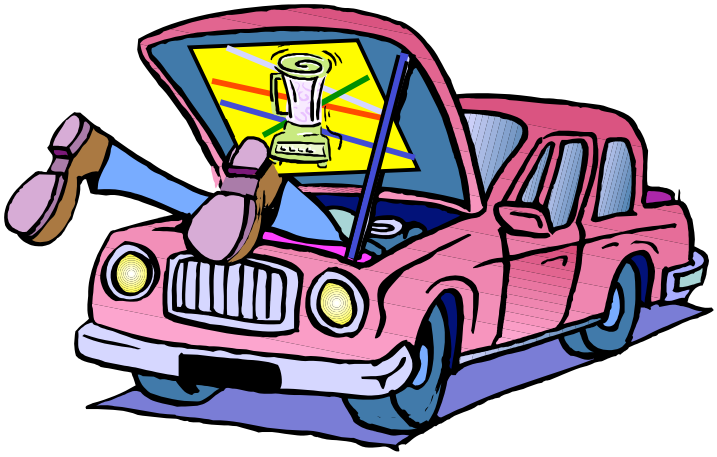
Example application: Anonymizing bulletin board or e-mail



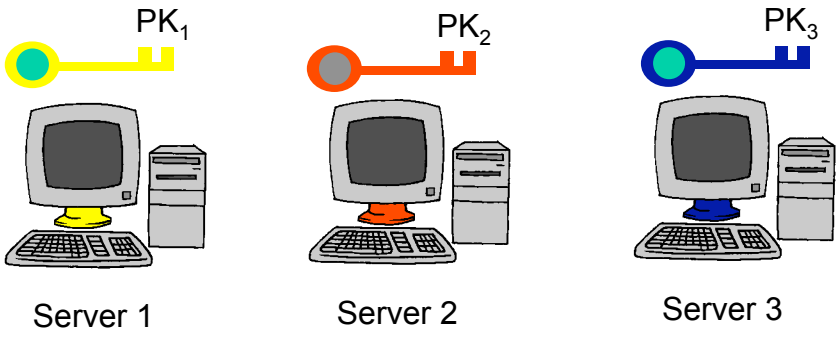
Another application: Voting



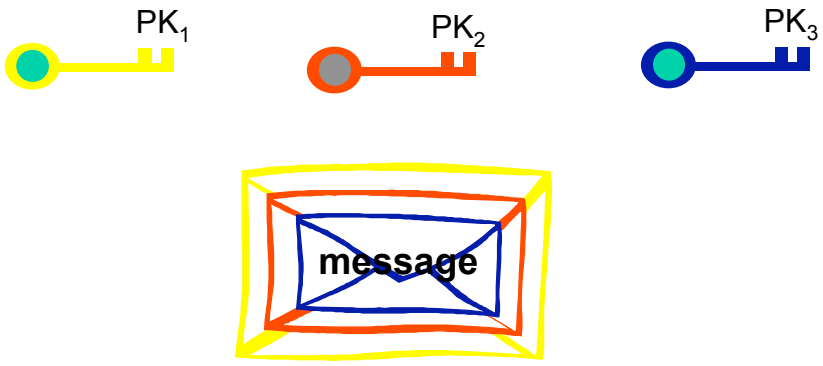
A look under the hood



Basic Mix (Chaum '81)

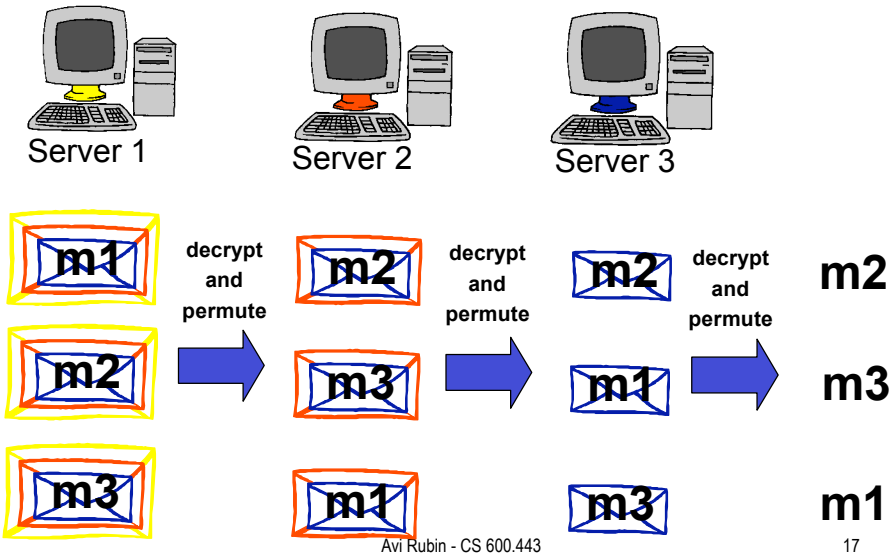


Encryption of Message



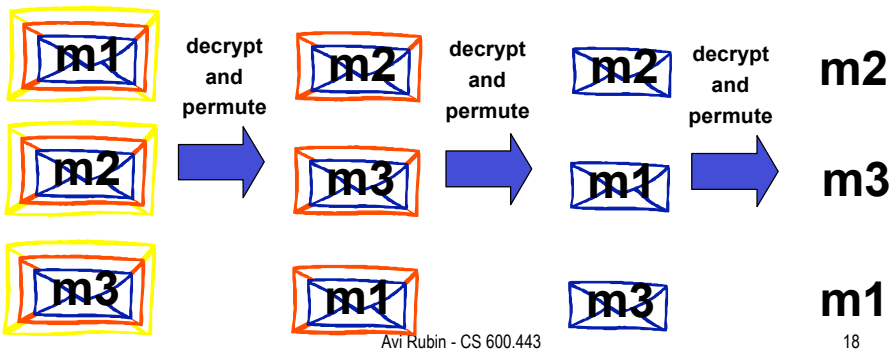
Ciphertext = $E_{PK1}[E_{PK2}[E_{PK3}[message]]]$

Basic Chaumian Mix

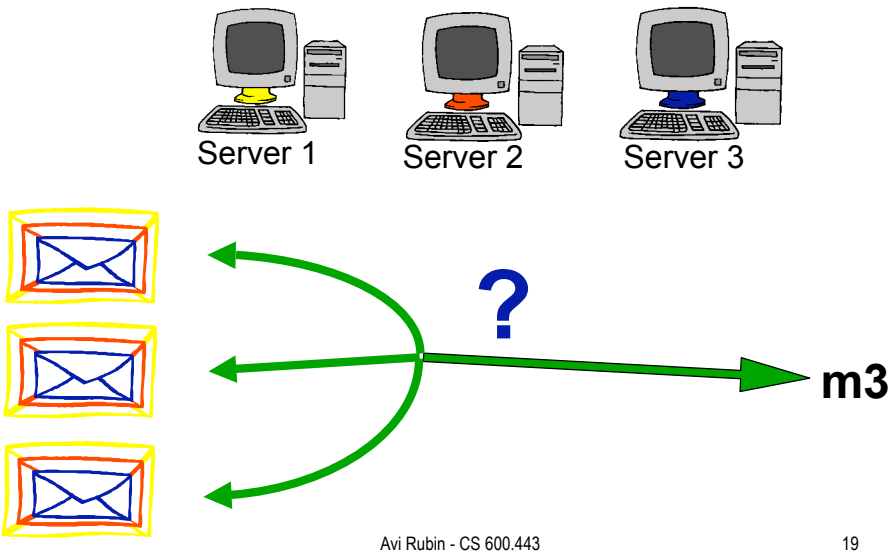


Basic Chaumian Mix

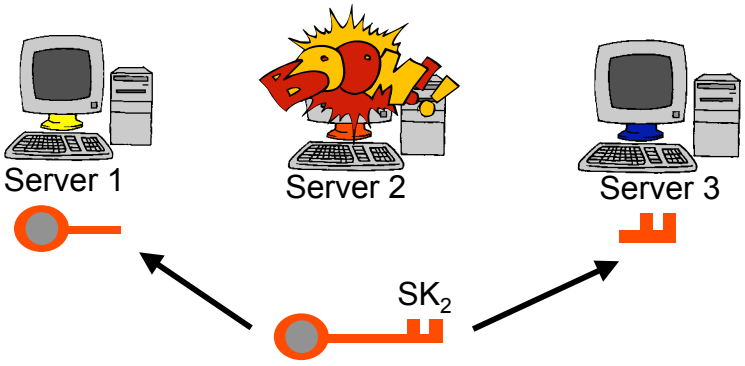
Observe: As long as one server is honest, privacy is preserved



Basic Chaumian Mix

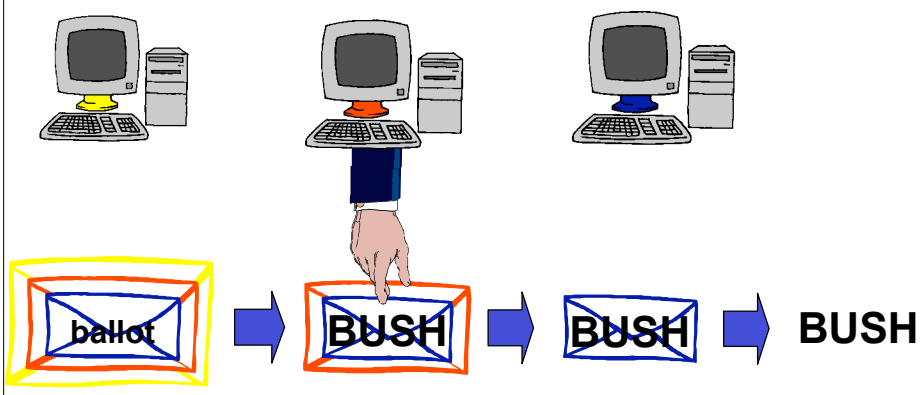


What if one server fails?



- Solution idea: Share key among others
- Privacy now requires a *majority* of honest servers
- Tolerance of failure is called *robustness*

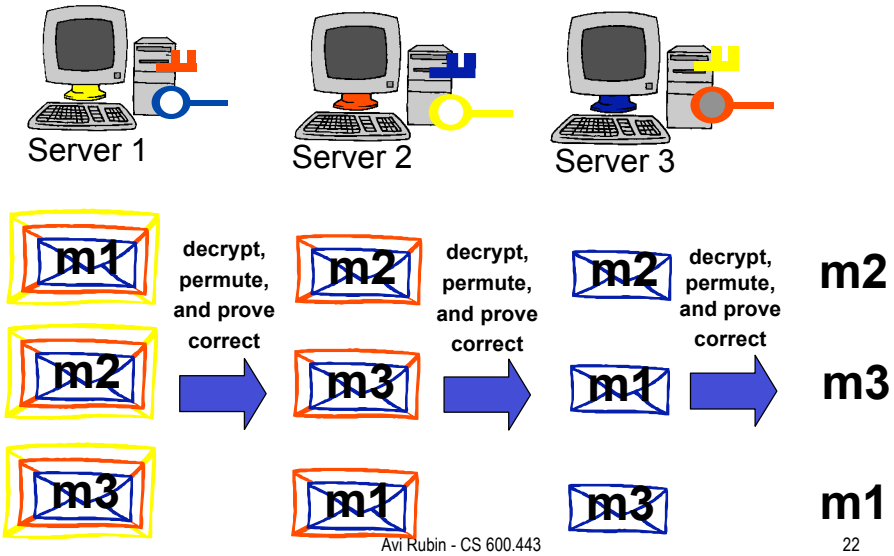
What if one server cheats?



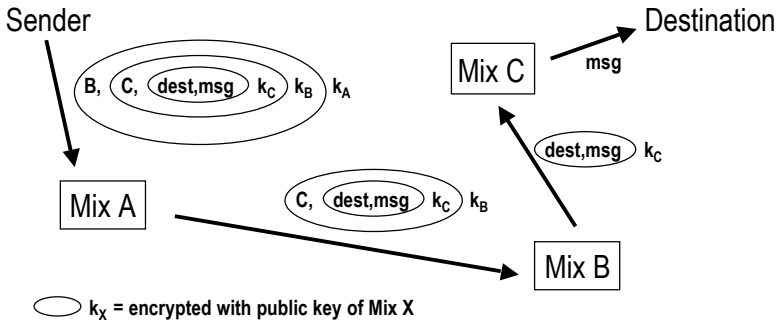
Solution idea:

- Have each server prove that it permuted and decrypted correctly
- Proof may be digitally signed and carried along with ciphertexts

Robust Mix



Mixes [Chaum81]



Sender routes message randomly through network of "Mixes", using layered public-key encryption.

Limitations of mixes

- Require public key cryptography (costly)
- All send fixed-length messages, at regular intervals, including "dummies" when necessary (more costly)
- Vulnerable to timing attacks by first and last mix, if used for synchronous communication
- No sender anonymity from first mix or sys. admin.

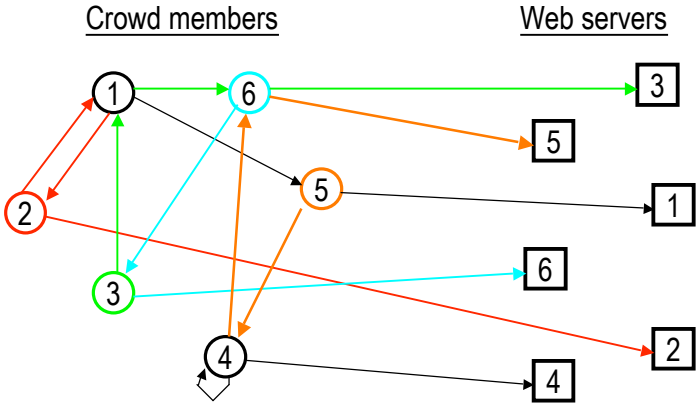
Crowds

- A new technique for anonymity on the web
- A system that implements it
  - code now available in the US and Canada
- Can be analyzed in the face of various adversaries
  - Degrees of anonymity

How Crowds works

- Each user joins a ***crowd*** of other users
- Each user represented by a ***jondo*** on her machine
- Each jondo either submits request to end server or forwards it to a randomly chosen jondo
- A jondo cannot tell if a request is initiated by the previous jondo or one before it
- Request and reply follow same path

Example



Other jondo jobs

- Strip out
  - cookies (option to include)
  - identifying headers
    - From:
    - Referer:
    - User-Agent:
    - Pragma:
- Add the word *Crowd* to title of page
- Provide messages to the user

Degrees of anonymity

More

Absolute privacy: adversary cannot observe communication

Beyond suspicion: no user is more suspicious than any other

Probable innocence: each user is more likely innocent than not

Possible innocence: nontrivial probability that user is innocent

Exposed (default on web): adversary learns responsible user

Provably exposed: adversary can prove your actions to others

Less

What Crowds achieves

Attacker	*Sender Anonymity*	*Receiver Anonymity*
Local eavesdropper	exposed	p(beyond suspicion)$\to 1$ as n$\to\infty$
c collaborating jondos, where $n \geq \frac{p_f}{p_{f-1/2}}(c+1)$	probable innocence + p(absolute privacy)$\to 1$ as n$\to\infty$	p(absolute privacy)$\to 1$ as n$\to\infty$
End server	beyond suspicion	N/A

Probable innocence

- H_k = first collaborator is at position k
- I = first collaborator is immediately preceded by initiator
- Path initiator has *probable innocence* if $P(I \mid H_{1+}) <= 1/2$

$$P(H_i) = \left(\frac{p_f\,(n-c)}{n} \right)^{i-1} \left(\frac{c}{n} \right)$$

- (prob. forwarding * prob. of noncollaborator)^ i-1 * prob of collab.

Probable innocence (cont.)

$$P(H_{1+}) = \frac{c}{n} \sum_{k=0}^{\infty} \left(\frac{p_f\,(n-c)}{n} \right)^k \qquad P(H_1) = \frac{c}{n}$$

$$P(H_{2+}) = \frac{c}{n} \sum_{k=1}^{\infty} \left(\frac{p_f\,(n-c)}{n} \right)^k \qquad P(I \mid H_{2+}) = \frac{1}{n-c}$$

$I \mid H_{2+}$ means collabs. get from initiator on hop ≥ 2

$$P(I) = P(H_1)\,P(I \mid H_1) + P(H_{2+})\,P(I \mid H_{2+})$$

Number of collaborators

$$P(I \mid H_{1+}) = \frac{P(I \wedge H_1)}{P(H_{1+})} = \frac{P(I)}{P(H_{1+})}$$

If $n \geq \frac{p_f}{p_f - 1/2}(c+1)$, then $P(I \mid H_{1+}) <= 1/2$

E.g. take $p_f = 3/4$, then $n \geq 3(c+1)$ ensures probable innocence

So, we can tolerate having almost 1/3 of the jondos in the crowd collaborating for this value of p_f

Encryption

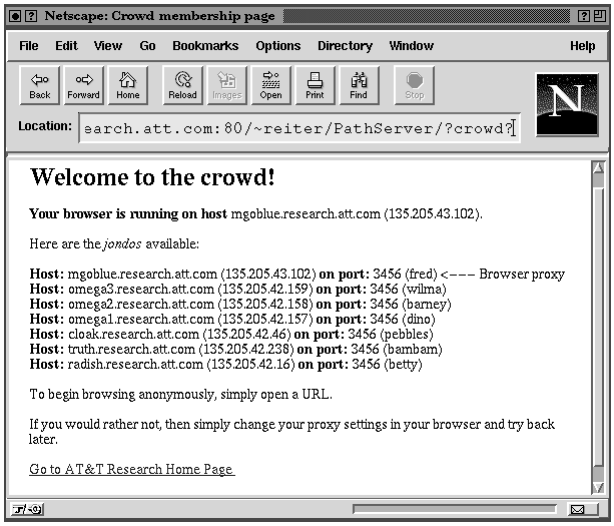
- Used to defend again local eavesdropper
- *Path key* used to encrypt request and replies
- Path key is re-encrypted with pairwise keys on each hop
- Fast stream cipher used to encrypt replies
- Requester/submitter build stream while waiting for reply

Static paths

- Dynamic paths enable collaborators to locate the initiator
  - link different paths based on content, timing
  - choose jondo from which paths are most often received
- Paths are rerouted only when
  - a jondo fails (path is rerouted as to not risk anonymity)
  - a new jondo joins
    - joiner idle until *join commit* occurs
    - at *join commit* all static paths are rerouted once
    - all user are notified about path rerouting

Embedded images

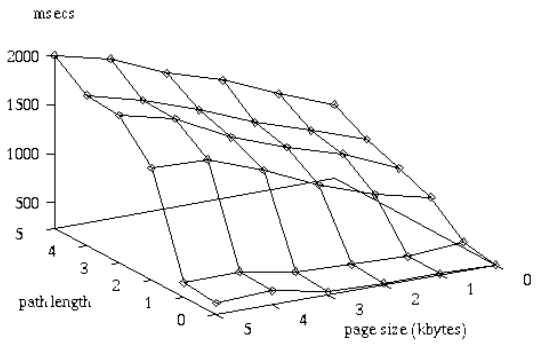
- Automatically retrieved by browser
- Can lead to a timing attack
  - first jondo realizes it is first due to timing of image requests
- To prevent
  - submitting jondo sends images along with page
  - requesting jondo does not request images of crowd, but reads them in reply



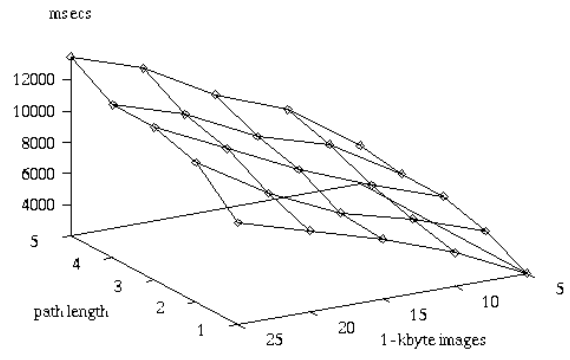
Performance tests

- Setup: 4 jondos
  - moderately loaded 150 Mhz Sparc 20s, running SunOS 4.1.4
- Web server
  - 133 Mhz SGI running Irix 5.3 and Apache web server
- All machines at AT&T on the same LAN
- Tested
  - different path lengths
  - different page sizes
  - different numbers of embedded images

Impact of page size



Impact of embedded images



Scale

- Expected number of appearances of a jondo on all paths is

$$O\left(\frac{1}{(1 - p_f)^2} \left(1 + \frac{1}{n}\right) \right)$$

- So crowds scales well as crowd size grows
- Expected path length is $\frac{1}{1 - p_f}$

Deployment issues

- Firewalls
  - jondos *outside* cannot connect to jondos *inside*
  - insiders can still participate, with weaker anonymity
- Different speed connections
  - separate crowds for slow modem users
- SSL
  - cannot parse HTML, to avoid timing attacks
- Must disable Java and Javascript

Crowd membership - mechanism

- Security relies on ability to control crowd membership
  - must maintain ratio of *n* to *c*
- We use a *blender* and user accounts
  - authenticates crowd joining requests
  - generates pairwise keys
  - notifies jondos of new members
- Blender is NOT a central point for passive attack/failure
- Blender only assists in crowd membership, has nothing to do with browsing afterwards

Crowd membership - policy

- Two types of crowds
  - 20-30 members, tight membership control
  - huge membership, controlled by sheer size and some heuristics
    - one jondo/machine
    - limited machines/domain

Future directions

- Use Diffie-Hellman for pairwise keys
  - blender does not get to know keys
  - more resistance to passive attacks
- Replicated blender
  - higher availability
  - resistance to active attacks (e.g., Rampart)
- Address the mobile code problem
  - have made some progress on this

Risks of Crowds

- Others' requests may come from your machine
  - people unaware of Crowds may think they are yours
- Web vendors that rely on IP address to avoid fraud
  - gives fraudsters an anonymous web presence
- Target advertisers may be upset
- Any argument against anonymity...

Vision

- Wide-scale adoption of crowds on the Internet
- No more correlation between
  - a web request
  - the machine that made the request
- Less targeted advertising
- A step towards greater user privacy