CSE 484 and CSE M 584 (Winter 2009)

“Crypto” Bigger Picture Memory, Randomness, Anonymity, and Information Leakage

Tadayoshi Kohno

Thanks to Dan Boneh, Dieter Gollmann, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Randomness issues

- Many applications (especially security ones) require randomness
- "Obvious" uses:
  - Generate secret cryptographic keys
  - Generate random initialization vectors for encryption
- Other "non-obvious" uses:
  - Generate passwords for new users
  - Shuffle the order of votes (in an electronic voting machine)
  - Shuffle cards (for an online gambling site)

C's rand() Function

- C has a built-in random function: rand()

```
unsigned long int next = 1;
/* rand:  return pseudo-random integer on 0..32767 */
int rand(void) {
   next = next * 1103515245 + 12345;
   return (unsigned int)(next/65536) % 32768;
}
/* srand:  set seed for rand() */
void srand(unsigned int seed) {
   next = seed;
}
```

- Problem: don't use rand() for security-critical applications!
  - Given a few sample outputs, you can predict subsequent ones

Dr.Dobb's Portal
The World of Software Development

ABOUT US | CONTACT | ADVERTISE | SUBSCRIBE | SOURCE CODE | CURRENT PRINT ISSUE
NEWSLETTERS | RESOURCES | BLOGS | PODCASTS | CAREERS

Windows/.NET

July 22, 2001

Randomness and the Netscape Browser

How secure is the World Wide Web?

Ian Goldberg and David Wagner

- Email
- Prin
- Discuss
- Rep

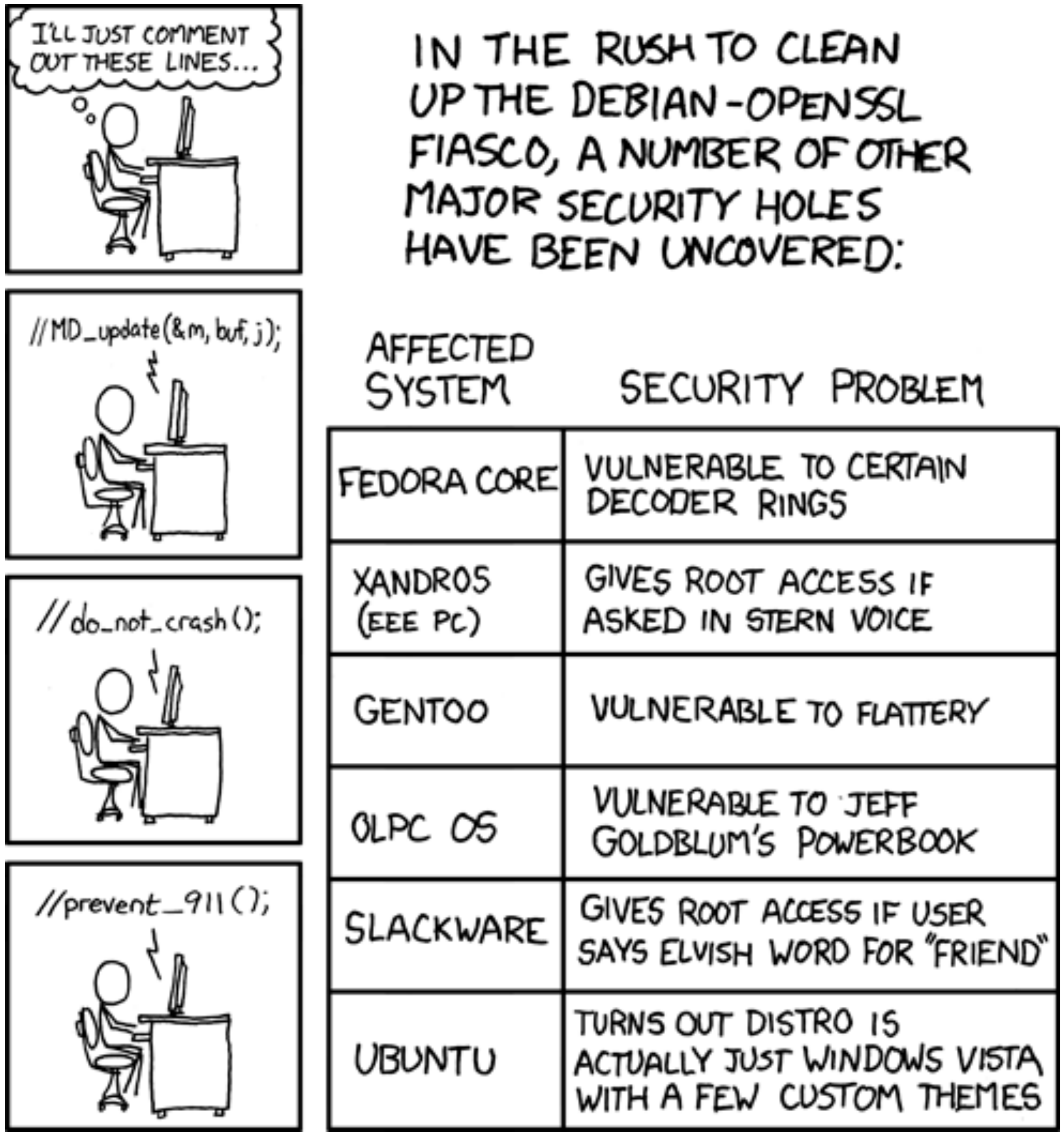
add to:

- Del.icio.us
- Slash
- Digg
- Y!
- Google
- MyWe
- Spurl
- Blin
- Furl

No one was more surprised than Netscape Communications when a pair of computer-science students broke the Netscape encryption scheme. Ian and David describe how they attacked the popular Web browser and what they found out.

Problems in Practice

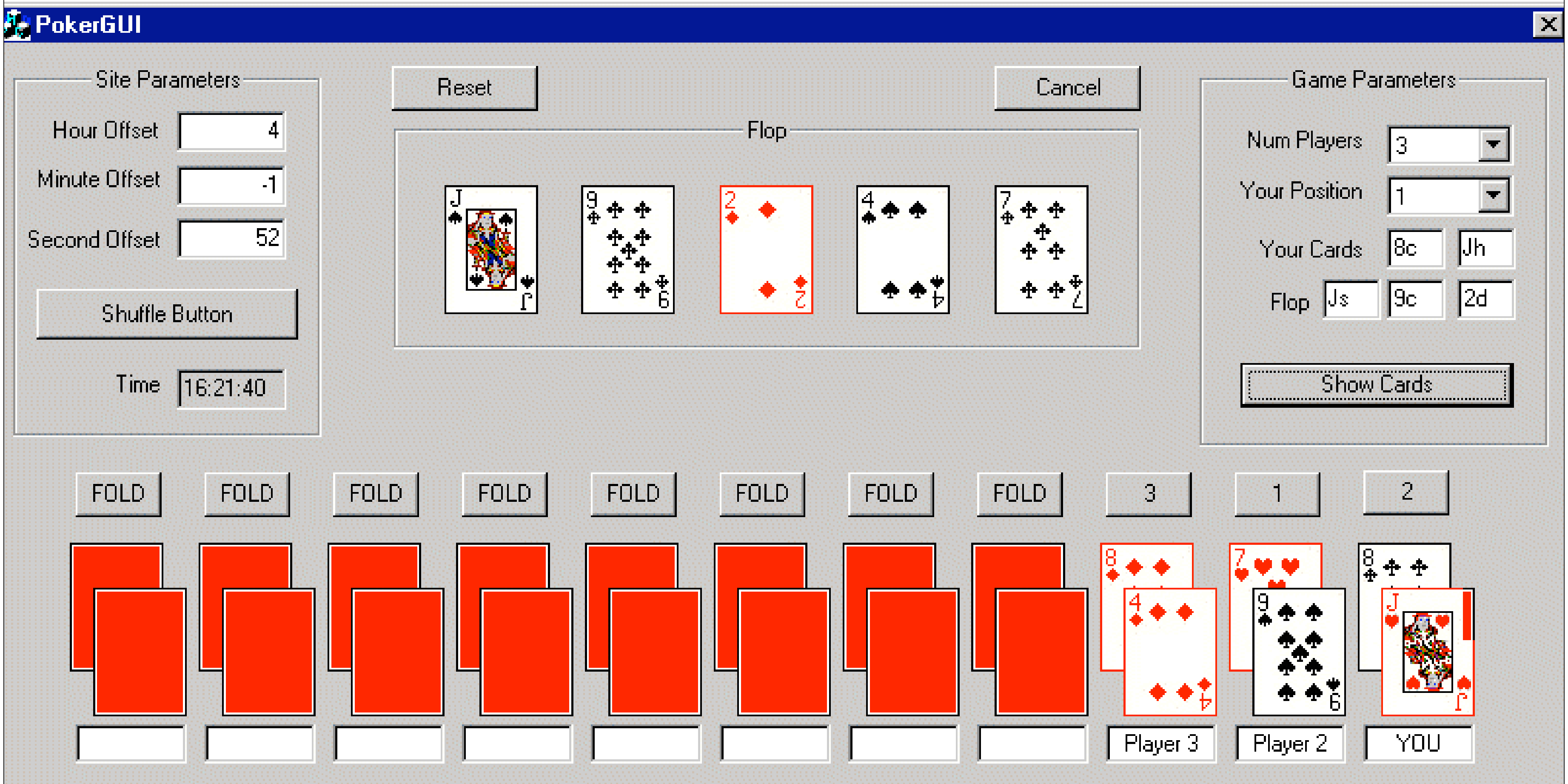
- One institution used (something like) rand() to generate passwords for new users
  - Given your password, you could predict the passwords of other users
- Kerberos (1988 - 1996)
  - Random number generator improperly seeded
  - Possible to trivially break into machines that rely upon Kerberos for authentication
- Online gambling websites
  - Random numbers to shuffle cards
  - Real money at stake
  - But what if poor choice of random numbers?



I'LL JUST COMMENT OUT THESE LINES...
//MD_update(&m, buf, j);
//do_not_crash();
//prevent_911();
IN THE RUSH TO CLEAN UP THE DEBIAN-OPENSSL FIASCO, A NUMBER OF OTHER MAJOR SECURITY HOLES HAVE BEEN UNCOVERED:
AFFECTED SYSTEM
SECURITY PROBLEM
FEDORA CORE
VULNERABLE TO CERTAIN DECODER RINGS
XANDROS (EEE PC)
GIVES ROOT ACCESS IF ASKED IN STERN VOICE
GENTOO
VULNERABLE TO FLATTERY
OLPC OS
VULNERABLE TO JEFF GOLDBLUM'S POWERBOOK
SLACKWARE
GIVES ROOT ACCESS IF USER SAYS ELVISH WORD FOR "FRIEND"
UBUNTU
TURNS OUT DISTRO IS ACTUALLY JUST WINDOWS VISTA WITH A FEW CUSTOM THEMES

xkcd

Images from http://www.cigital.com/news/index.php?pg=art&artid=20



Images from http://www.cigital.com/news/index.php?pg=art&artid=20

Images from http://www.cigital.com/news/index.php?pg=art&artid=20

Big news... CNN, etc..

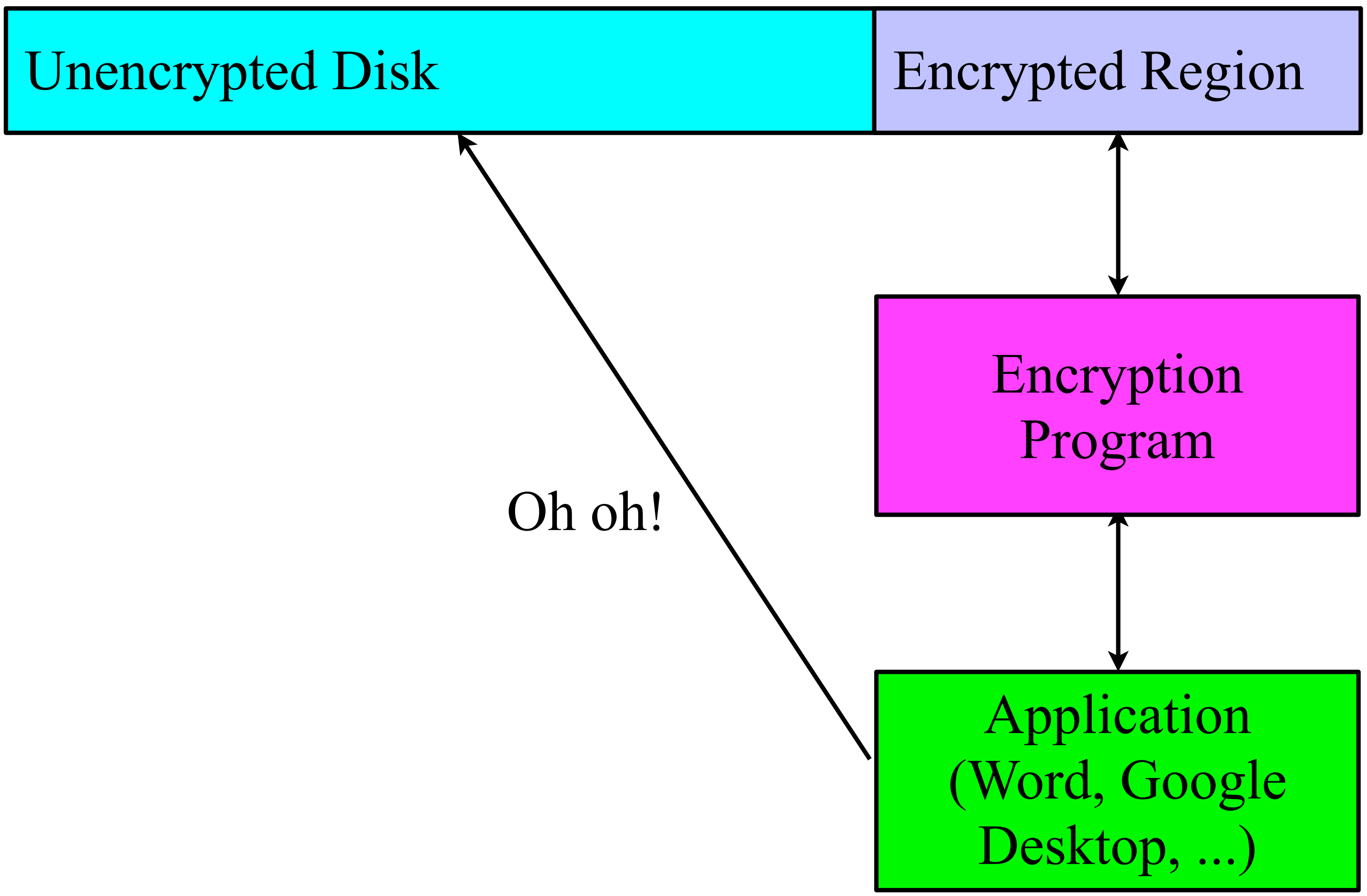
Obtaining Pseudorandom Numbers

- For security applications, want "cryptographically secure pseudorandom numbers"
- Libraries include:
  - OpenSSL
  - CryptoAPI (Microsoft)
- Linux:
  - /dev/random
  - /dev/urandom
- Internally:
  - Pool from multiple sources (interrupt timers, keyboard, ...)
  - Physical sources (radioactive decay, ...)

Secure Deletion

- (See other slide deck, or paper here: http://citp.princeton.edu/memory/.)

Disk Encryption and Other Applications



See Czeskis, St. Hilaire, Koscher, Gribble, Kohno, Schneier 2008

Anonymity

Privacy on Public Networks

- Internet is designed as a public network
  - Machines on your LAN may see your traffic, network routers see all traffic that passes through them
- Routing information is public
  - IP packet headers identify source and destination
  - Even a passive observer can easily figure out who is talking to whom
- Encryption does not hide identities
  - Encryption hides payload, but not routing information
  - Even IP-level encryption (tunnel-mode IPSec/ESP) reveals IP addresses of IPSec gateways

Applications of Anonymity (I)

- Privacy
  - Hide online transactions, Web browsing, etc. from intrusive governments, marketers and archivists
- Untraceable electronic mail
  - Corporate whistle-blowers
  - Political dissidents
  - Socially sensitive communications (online AA meeting)
  - Confidential business negotiations
- Law enforcement and intelligence
  - Sting operations and honeypots
  - Secret communications on a public network

Applications of Anonymity (II)

- Digital cash
  - Electronic currency with properties of paper money (online purchases unlinkable to buyer's identity)
- Anonymous electronic voting
- Censorship-resistant publishing

What is Anonymity?

- Anonymity is the state of being not identifiable within a set of subjects
  - You cannot be anonymous by yourself!
    - Big difference between anonymity and confidentiality
  - Hide your activities among others' similar activities
- 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)

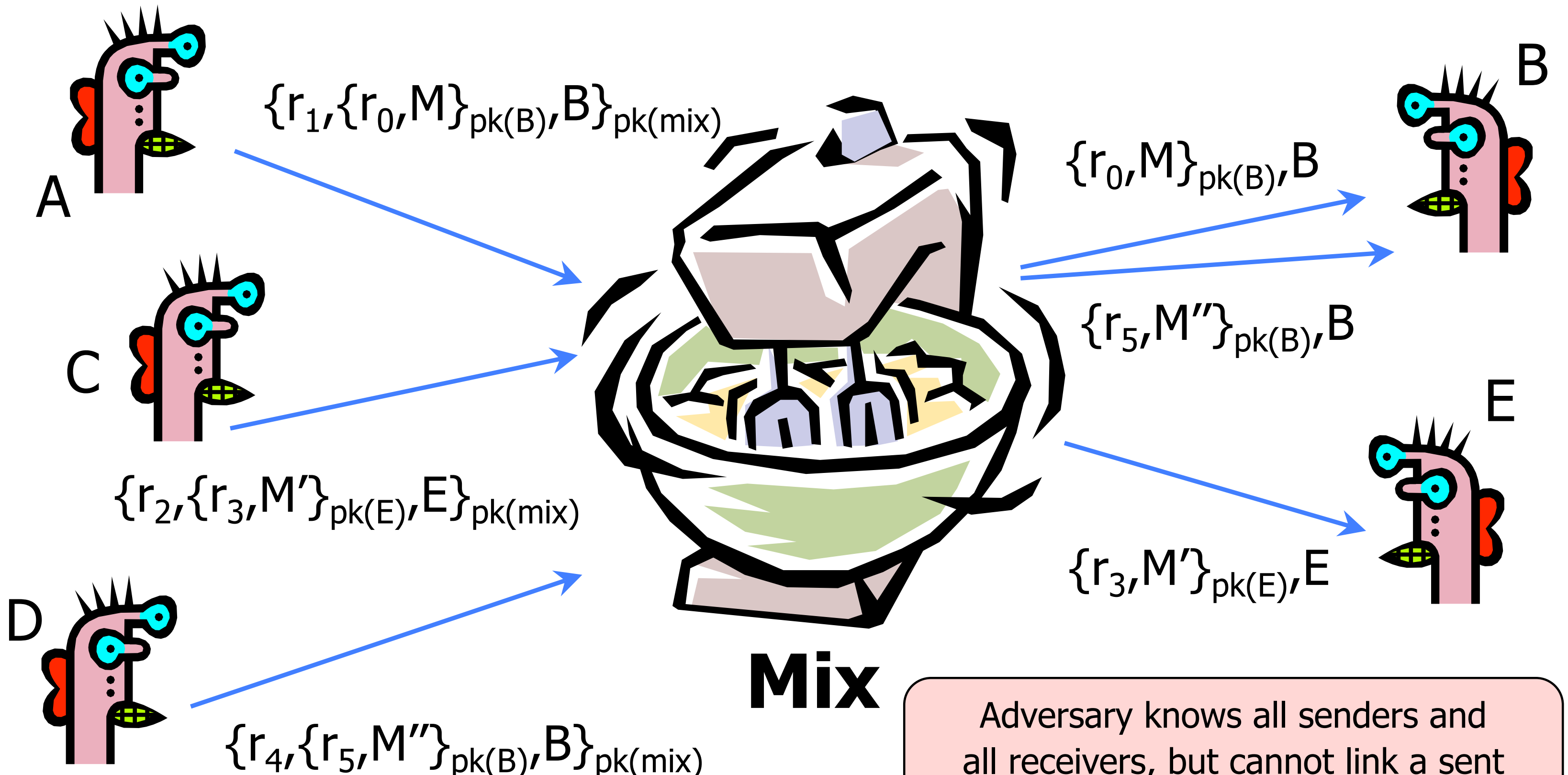
Chaum's Mix

- Early proposal for anonymous email
  - David Chaum. "Untraceable electronic mail, return addresses, and digital pseudonyms". Communications of the ACM, February 1981.

Before spam, people thought anonymous email was a good idea ☺

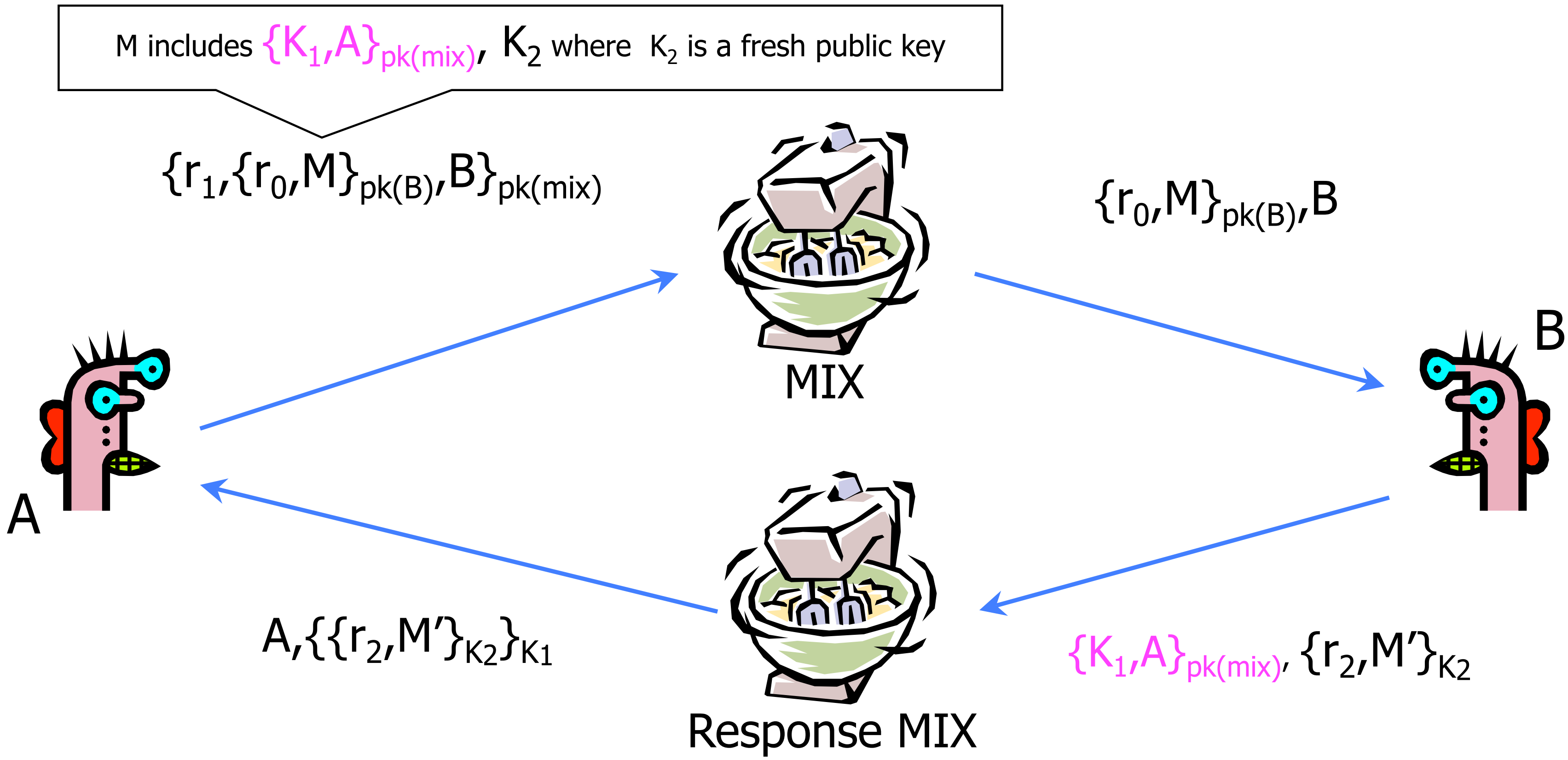
- Public key crypto + trusted re-mailer (Mix)
  - Untrusted communication medium
  - Public keys used as persistent pseudonyms
- Modern anonymity systems use Mix as the basic building block

Basic Mix Design

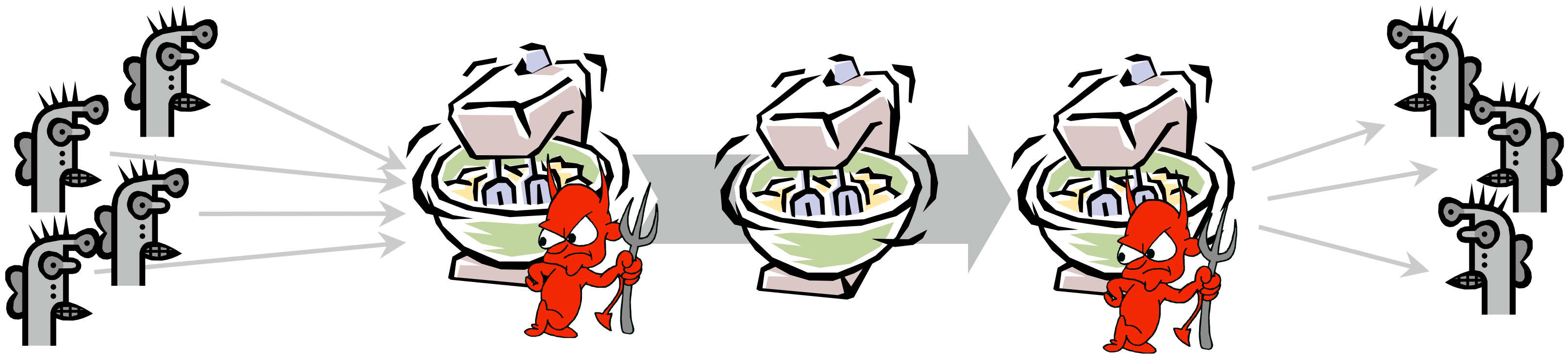


Adversary knows all senders and all receivers, but cannot link a sent message with a received message

Anonymous Return Addresses



Mix Cascade

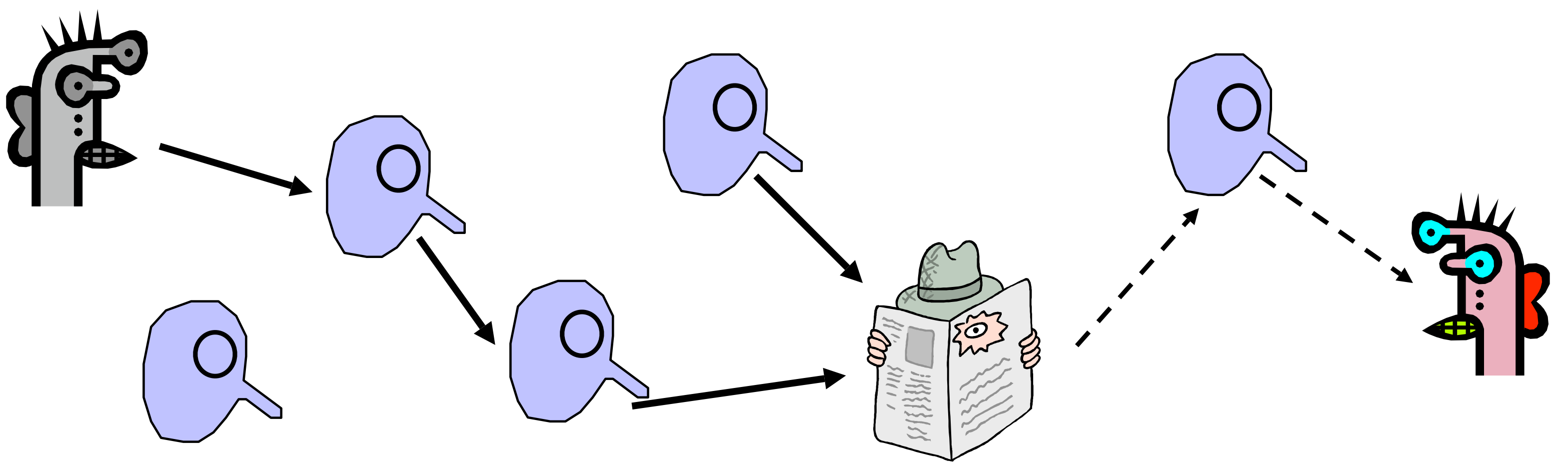


- Messages are sent through a sequence of mixes
  - Can also form an arbitrary network of mixes ("mixnet")
- Some of the mixes may be controlled by attacker, but even a single good mix guarantees anonymity
- Pad and buffer traffic to foil correlation attacks

Disadvantages of Basic Mixnets

- Public-key encryption and decryption at each mix are computationally expensive
- Basic mixnets have high latency
  - Ok for email, not Ok for anonymous Web browsing
- Challenge: low-latency anonymity network
  - Use public-key cryptography to establish a “circuit” with pairwise symmetric keys between hops on the circuit
  - Then use symmetric decryption and re-encryption to move data messages along the established circuits
  - Each node behaves like a mix; anonymity is preserved even if some nodes are compromised

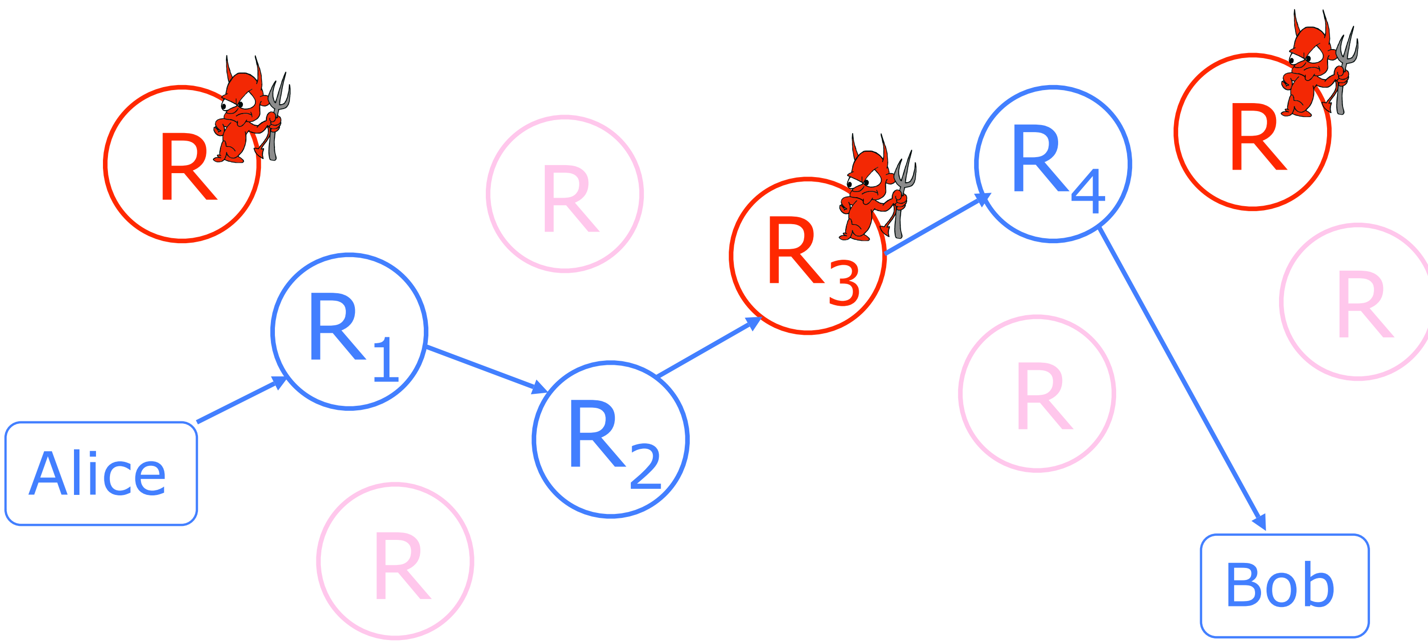
Another Idea: Randomized Routing



- Hide message source by routing it randomly
  - Popular technique: Crowds, Freenet, Onion routing
- Routers don't know for sure if the apparent source of a message is the true sender or another router

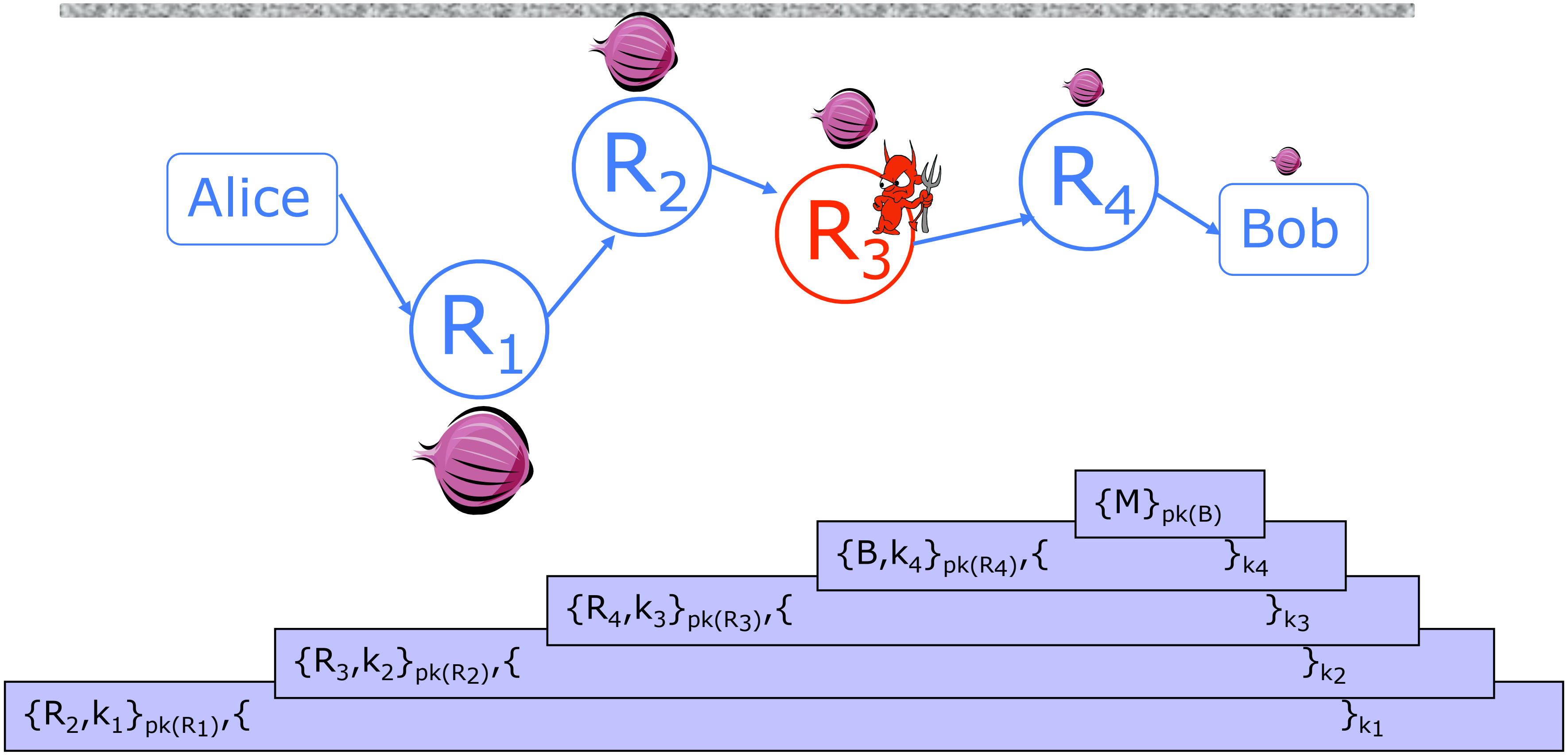
Onion Routing

[Reed, Syverson, Goldschlag '97]



- Sender chooses a random sequence of routers
  - Some routers are honest, some controlled by attacker
  - Sender controls the length of the path

Route Establishment



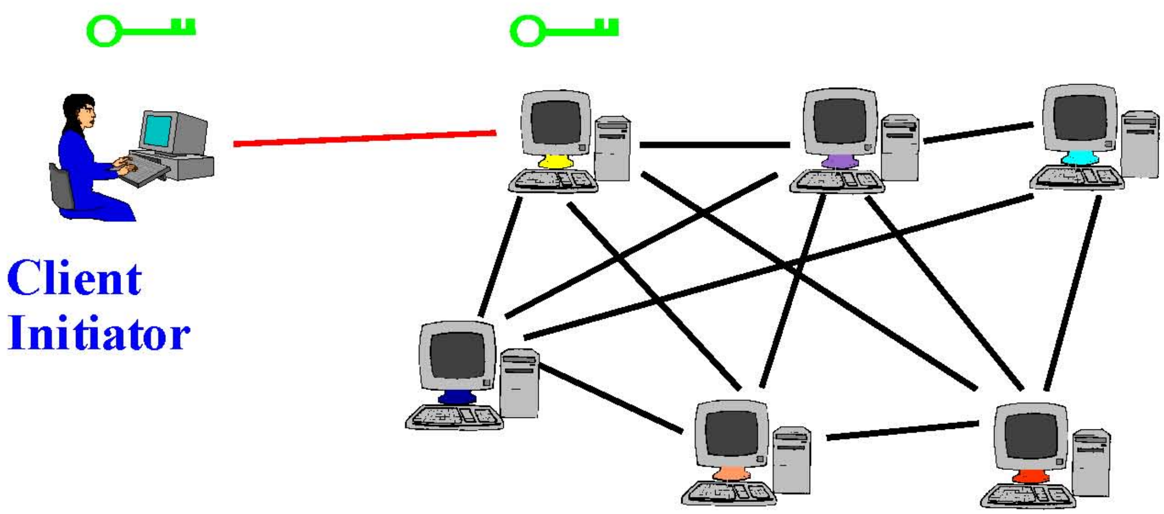
- Routing info for each link encrypted with router's public key
- Each router learns only the identity of the next router

Tor

- Second-generation onion routing network
  - http://tor.eff.org
  - Developed by Roger Dingledine, Nick Mathewson and Paul Syverson
  - Specifically designed for low-latency anonymous Internet communications
- Running since October 2003
- "Easy-to-use" client proxy
  - Freely available, can use it for anonymous browsing

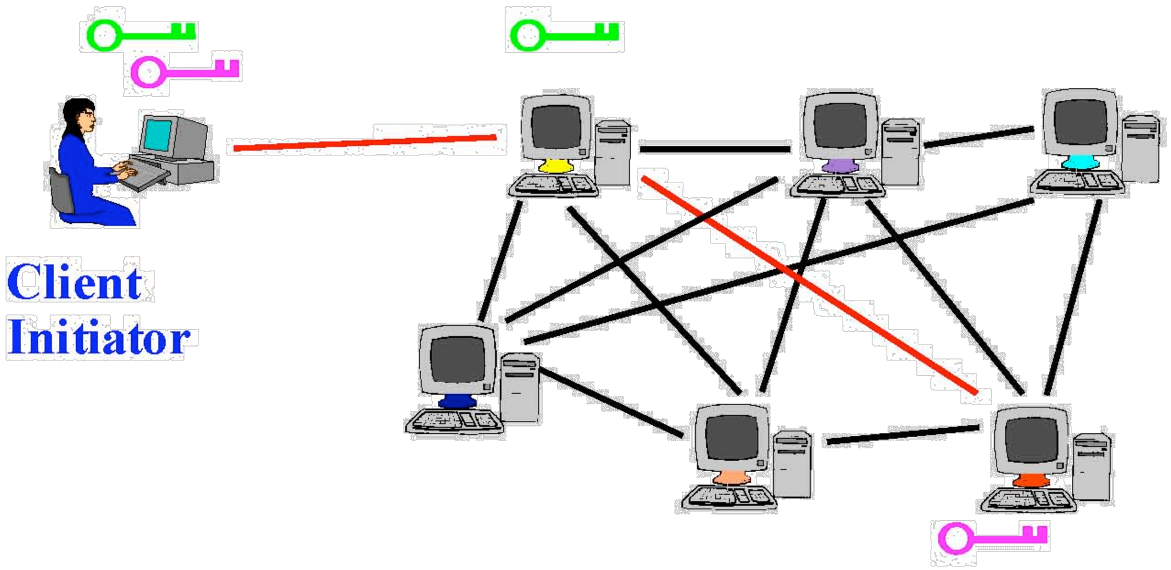
Tor Circuit Setup (1)

- Client proxy establish a symmetric session key and circuit with Onion Router #1



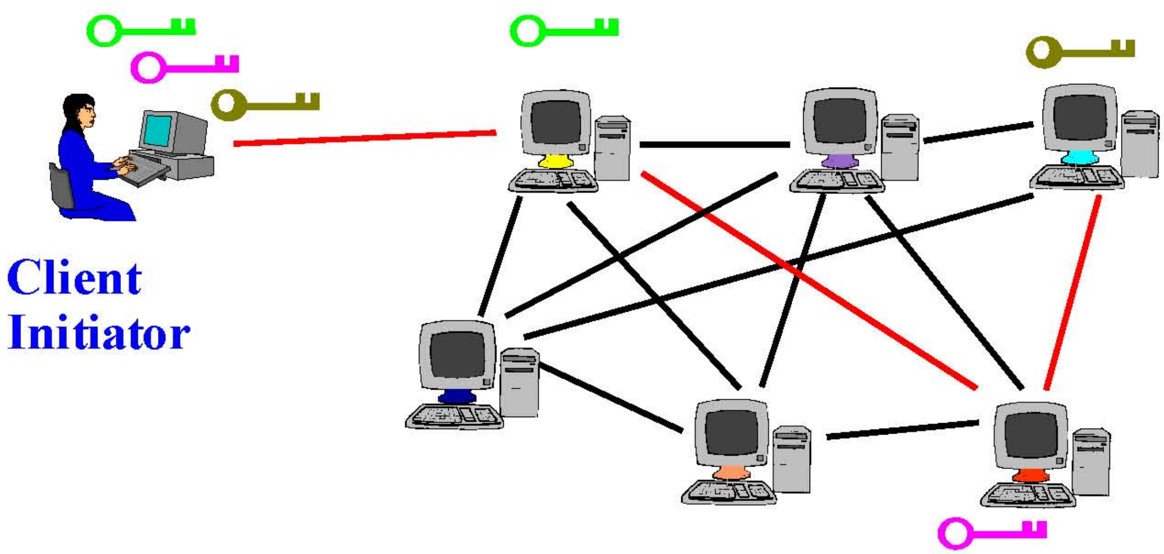
Tor Circuit Setup (2)

- Client proxy extends the circuit by establishing a symmetric session key with Onion Router #2
  - Tunnel through Onion Router #1 (don't need



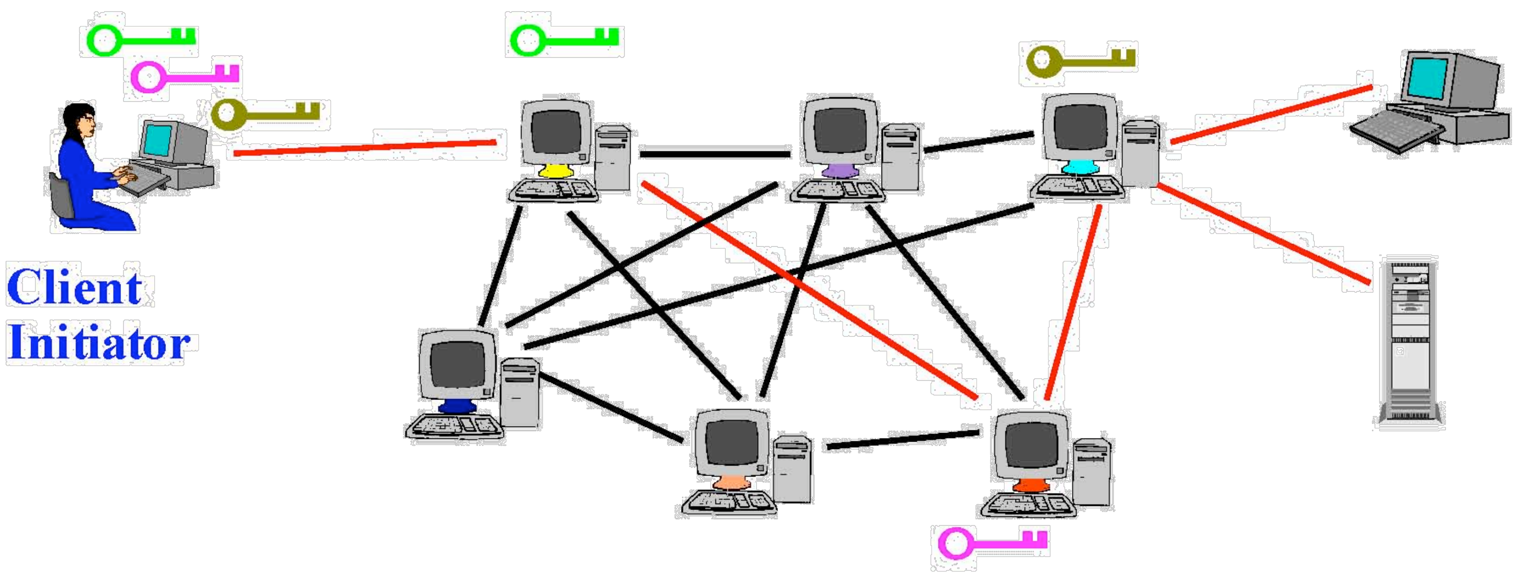
Tor Circuit Setup (3)

- Client proxy extends the circuit by establishing a symmetric session key with Onion Router #3
  - Tunnel through Onion Routers #1 and #2



Using a Tor Circuit

- Client applications connect and communicate over the established Tor circuit
  - Datagrams are decrypted and re-encrypted at each link



Tor Management Issues

- Many applications can share one circuit
  - Multiple TCP streams over one anonymous connection
- Tor router doesn't need root privileges
  - Encourages people to set up their own routers
  - More participants = better anonymity for everyone
- Directory servers
  - Maintain lists of active onion routers, their locations, current public keys, etc.
  - Control how new routers join the network
    - "Sybil attack": attacker creates a large number of routers
  - Directory servers' keys ship with Tor code

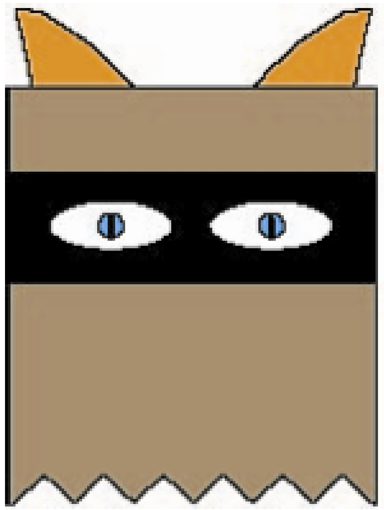
Attacks on Anonymity

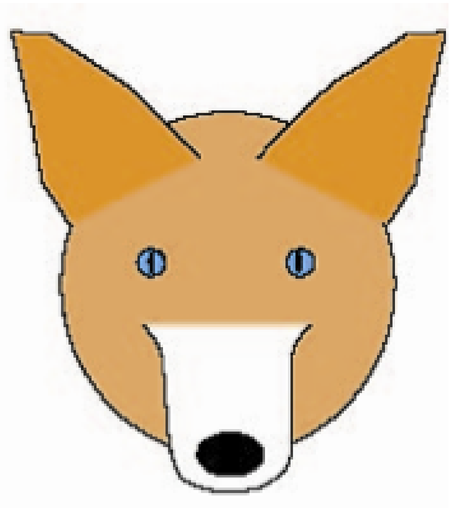
- Passive traffic analysis
  - Infer from network traffic who is talking to whom
  - To hide your traffic, must carry other people's traffic!
- Active traffic analysis
  - Inject packets or put a timing signature on packet flow
- Compromise of network nodes
  - Attacker may compromise some routers
  - It is not obvious which nodes have been compromised
    - Attacker may be passively logging traffic
  - Better not to trust any individual router
    - Assume that some <u>fraction</u> of routers is good, don't know which

Deployed Anonymity Systems

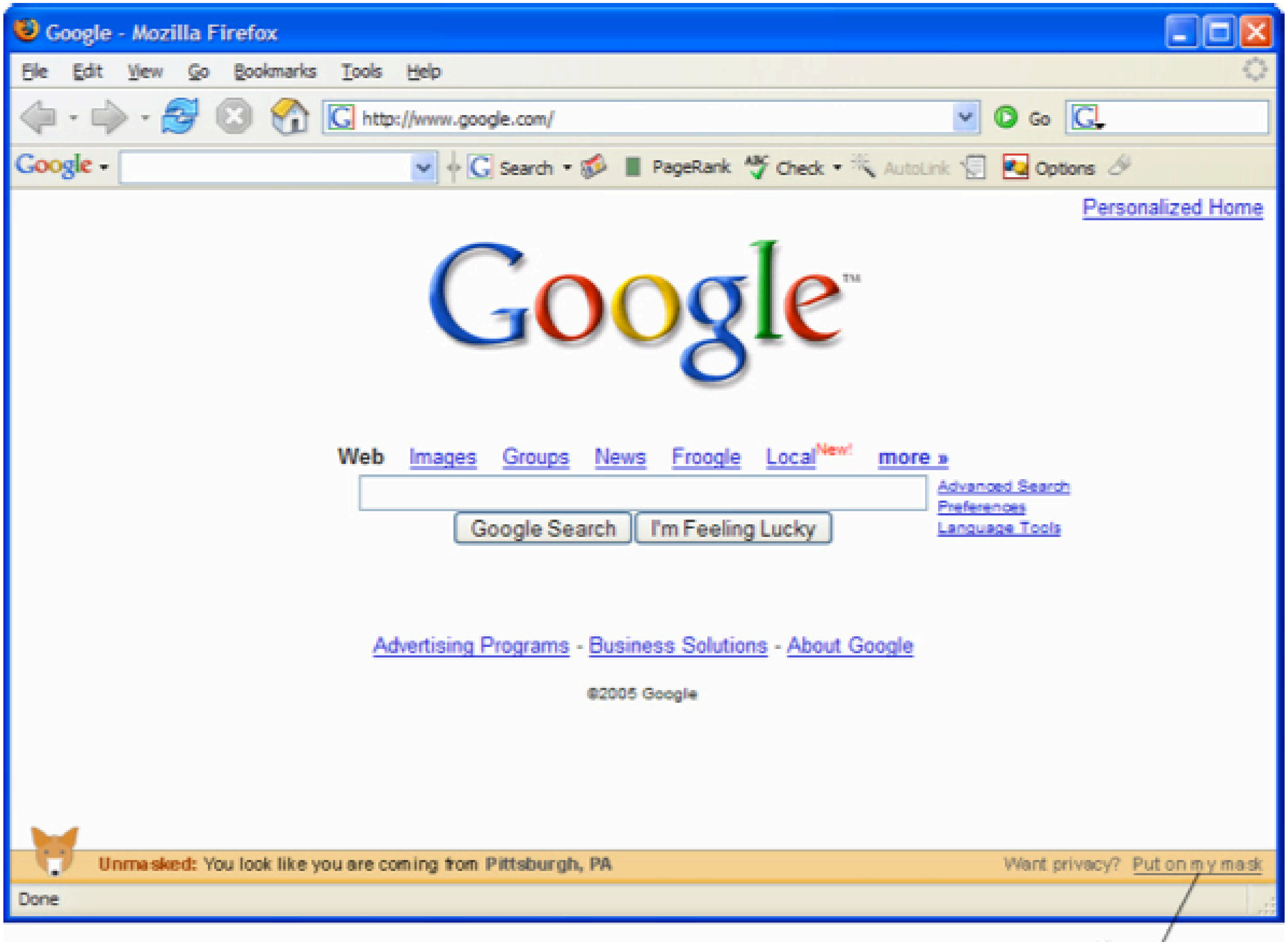
- Tor (http://tor.eff.org)
  - Overlay circuit-based anonymity network
  - Best for low-latency applications such as anonymous Web browsing
- Mixminion (http://www.mixminion.net)
  - Network of mixes
  - Best for high-latency applications such as anonymous email

FoxTor, Images from http://cups.cs.cmu.edu/foxtor/





FoxTor, Images from http://cups.cs.cmu.edu/foxtor/



FoxTor, Images from http://cups.cs.cmu.edu/foxtor/