

Security II: Security Strikes Back

15-441/641 Fall 2019

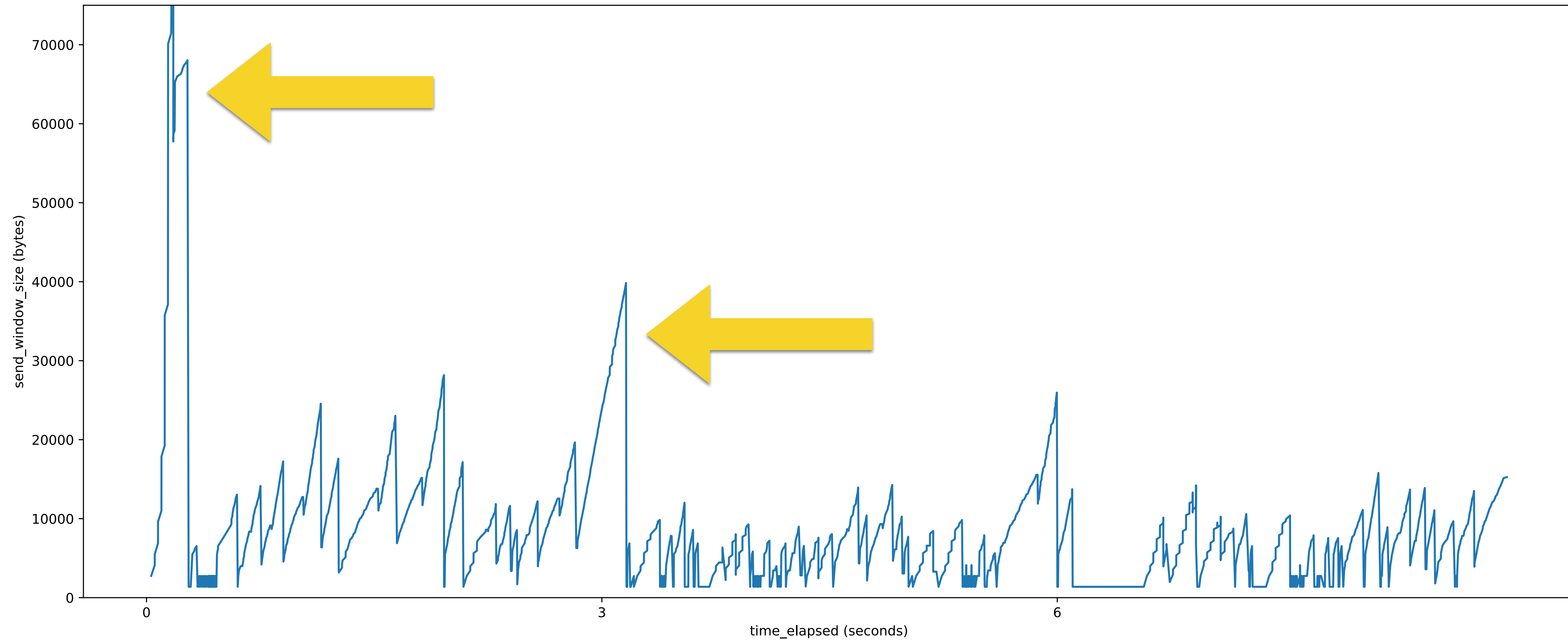
Profs Peter Steenkiste & **Justine Sherry**

**Carnegie
Mellon
University**

What should my graph look like?



Real graph from last year



Getting full credit on your graph

- Needs to show phases of TCP!
 - You might need to emulate a slower link
 - or send a longer file



Cryptography Overview

Symmetric Asymmetric

Confidentiality

One-Time Pad
Stream Ciphers
Block Ciphers

Encrypt w/ Public Key

Integrity

Message Authentication
Code
(e.g., HMAC, CBC-MAC)

Digital Signature

Authentication

MAC + Nonce

Digital Signature + Nonce



Symmetric vs. Asymmetric

Symmetric

- Shared secret
- 80 bit key for high security (in 2010)
- ~1,000,000 ops/s on 1GHz proc
- 10x speedup in HW

Asymmetric

- Public/private key pairs
- 2048 bit key for high security (in 2010)
- ~100 signs/s & ~1,000 verifies/s (RSA, 1GHz)
- Limited speedup in HW



Refresh from Tuesday

- What is confidentiality? What is integrity? What is authentication?
- Why does authentication require a nonce?
- How many keys are used when two parties communicate using symmetric cryptography?
- How many keys are used when two parties communicate using asymmetric cryptography?



How do we get keys?



Wait... how do we get the keys in the first place?

How do I get these keys in the first place??

Remember:

- Symmetric key primitives assumed Alice and Bob had already shared a key.
- Asymmetric key primitives assumed Alice knew Bob's public key.

This may work with friends, but when was the last time you saw Amazon.com walking down the street?



“Key Signing Party”



Key Setup

- We'll briefly look at 2 mechanisms:
 - Diffie Hellman Key Exchange
 - Certificate Authorities



Diffie-Hellman key exchange

- An early (1976) way to create a shared secret.
- Everyone knows a prime, p , and a generator, g .
- Alice and Bob want to share a secret, but only have internet to communicate over.





An activity: agree on a secret word
while the whole classroom can hear
you.

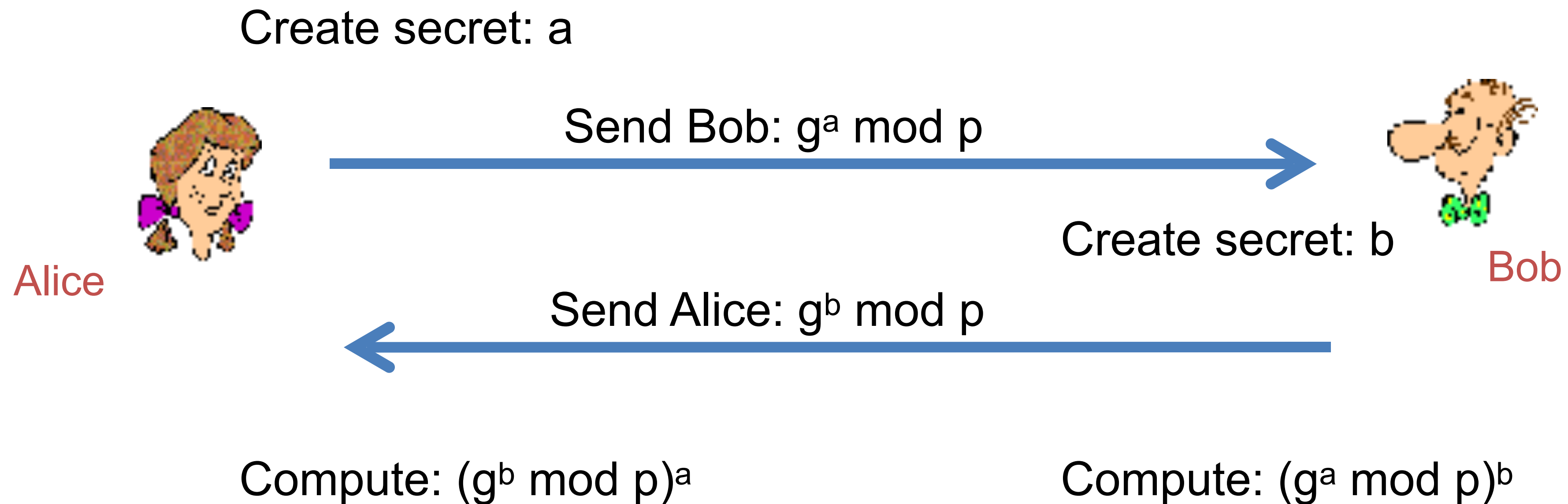


Why is this hard?



DH key exchange

Everyone: large prime p and generator g



Voila: They both know g^{ab} which is secret!



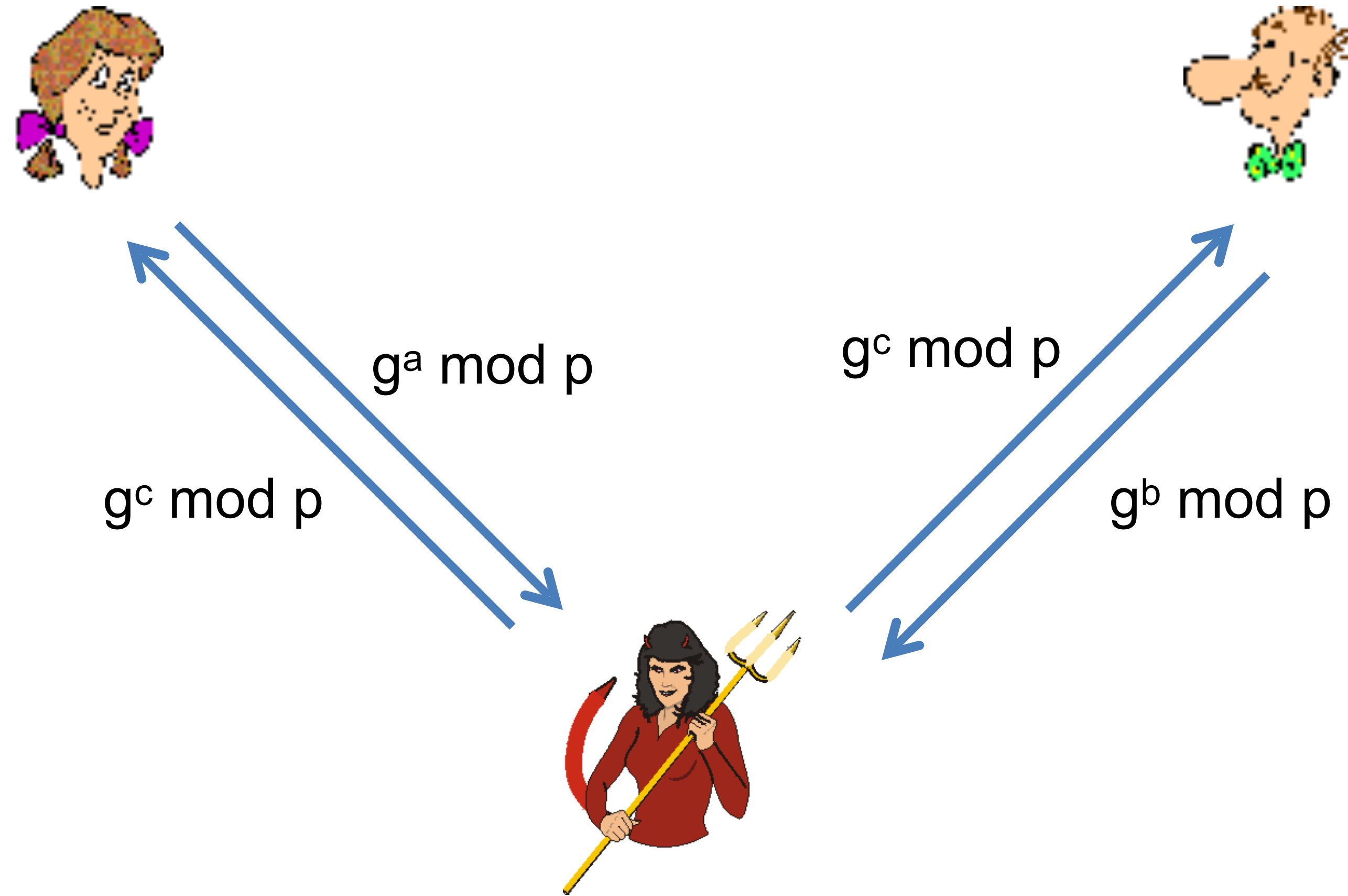
Math says: No attacker can compute $g^{ab} \bmod p$ just by listening to their communication!
(It's computationally intractable)



Security mindset: are we good
to go?



DH key exchange & Man-In-The-Middle



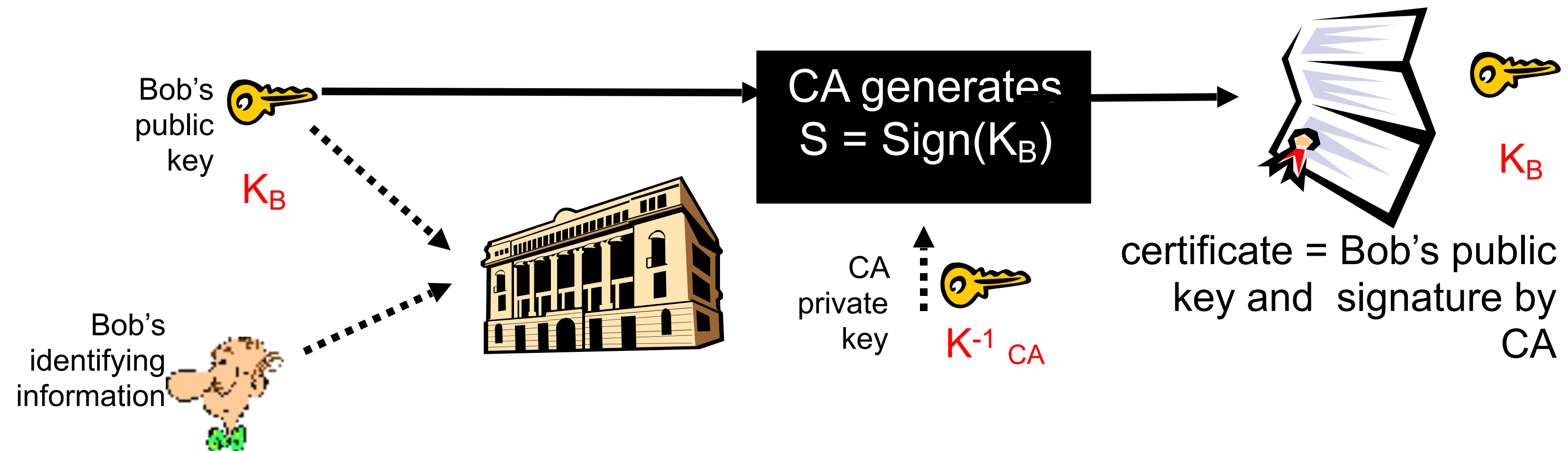
Threat Model

- Always important to be clear about what you think your attacker is capable of!
- If you think your attacker is capable of *modifying* traffic, can't use DH!
- But if attacker is just an eavesdropper — you're good to go!



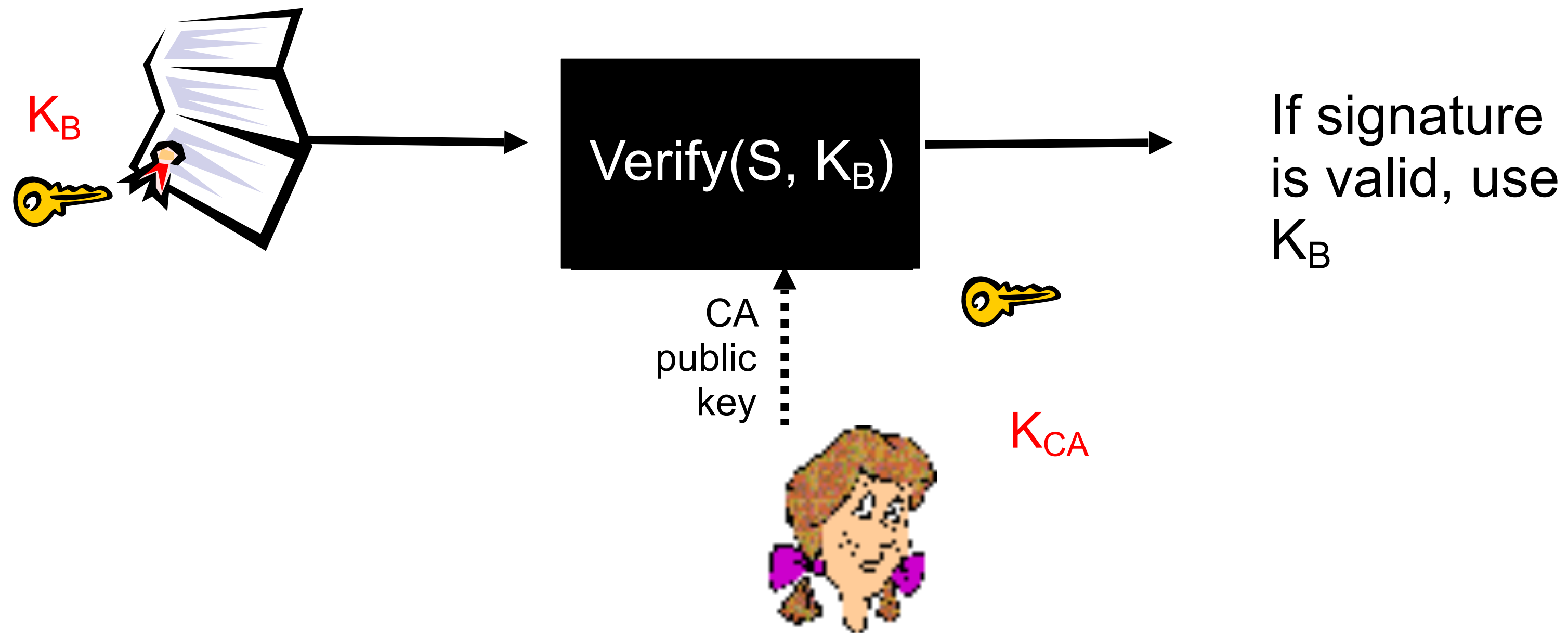
Certification Authorities

- **Certification authority (CA):** binds public key to particular entity, E.
- An entity E registers its public key with CA.
 - E provides “proof of identity” to CA.
 - CA creates certificate binding E to its public key.
 - Certificate contains E’s public key AND the CA’s signature of E’s public key.

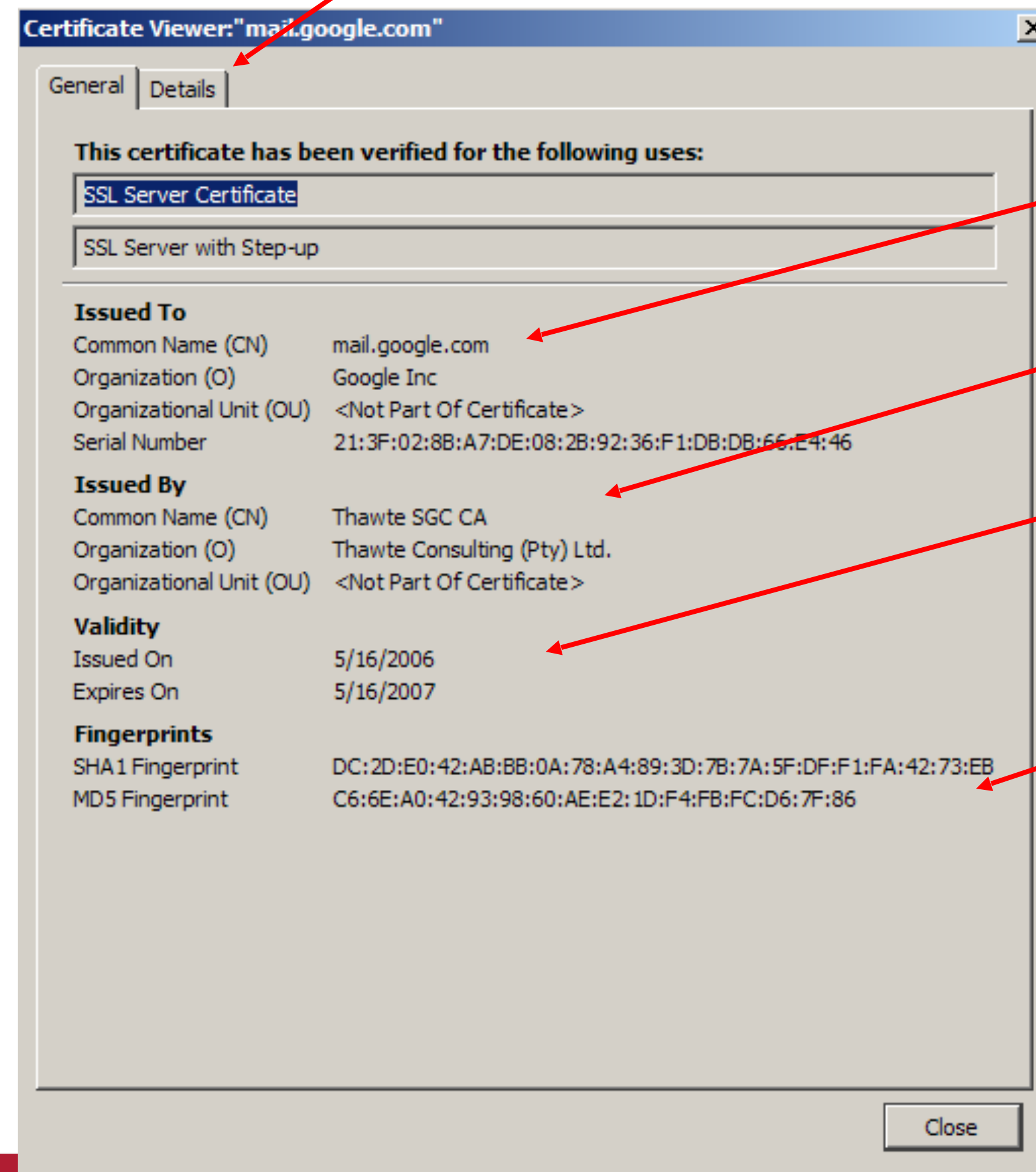


Certification Authorities

- When Alice wants Bob's public key:
 - Gets Bob's certificate (Bob or elsewhere).
 - Use CA's public key to verify the signature within Bob's certificate, then accepts public key



Certificate Contents



- Cert owner
- Cert issuer
- Valid dates
- Fingerprint of signature



Which Authority Should You Trust?

- If the browser detects a problem with a certificate, it asks user what to do
 - Invalid, expired, self-signed, ...
- Users often blindly click “yes”
 - They don’t know about certificates or TLS; don’t understand implications of a bad certificates
- Certificates are hard to read and can be misleading
 - Most information makes no sense to user
 - Names can be confusing, e.g., minor variants



Which Authority Should You Trust?

- Today: many authc

DigiNotar

From Wikipedia, the free encyclopedia

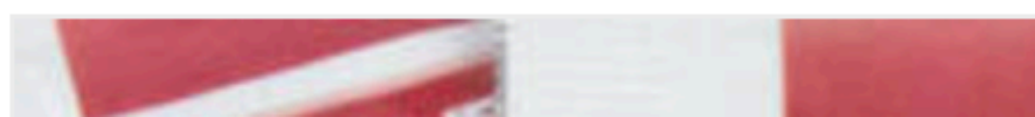
DigiNotar was a Dutch certificate authority on had become clear that a security breach had operaf

Security

Fuming Google tear one over rogue SSL

We've got just the thing for

By Iain Thomson in San Francisco 29 Oct




Security

Google publishes list of Certificate Authorities it doesn't trust

Thawte experiment aims to expose issuers of dodgy creds

By Richard Chirgwin 23 Mar 2016 at 04:02

25  SHARE ▼

Google's announced another expansion to the security information offered in its transparency projects: it's now going to track certificates you might *not* want to trust.

Certificate Authorities (CAs) that your browser (or smartphone) trusts have a suitable entry in "settings", but if a site presents a certificate from



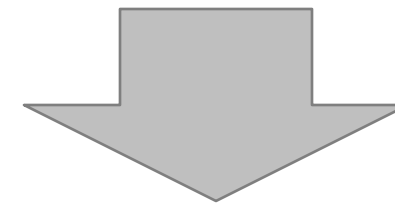
How do we apply symmetric and asymmetric crypto on the Internet?



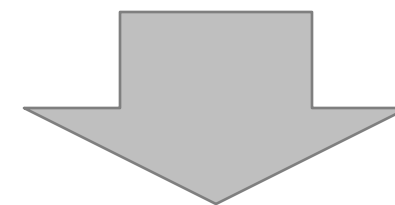
Let's put it all together!

Transport Layer Security (TLS)
aka Secure Socket Layer (SSL)

Uses **certificate authority** to provide public key



Uses **asymmetric crypto** to establish symmetric key



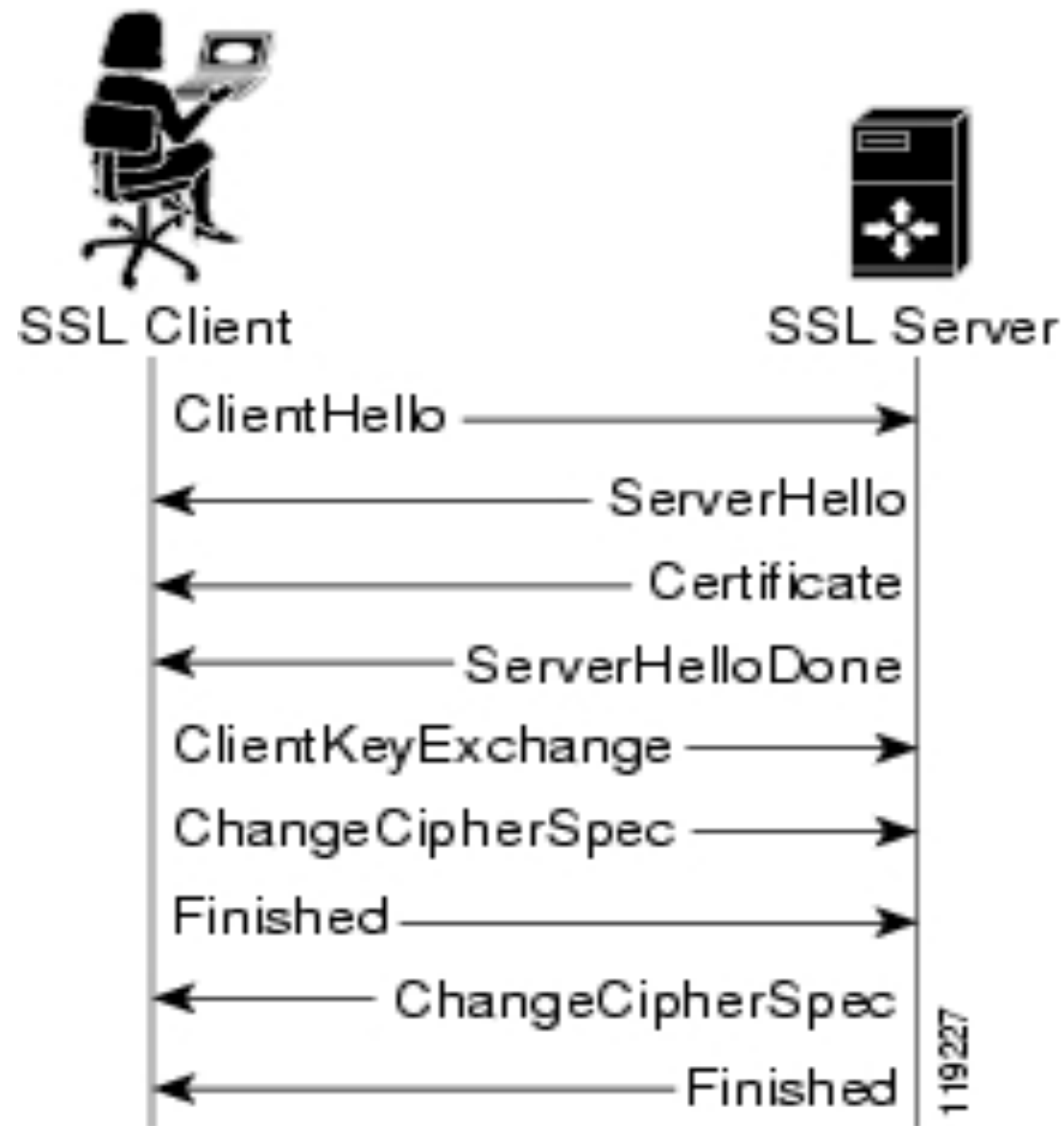
Uses **symmetric crypto** for data encryption



This is called “hybrid encryption.”



Setup Channel with TLS “Handshake”



Handshake Steps:

- 1) Client and server negotiate exact cryptographic protocols
- 2) Client validates public key certificate with CA public key.
- 3) Client encrypts secret random value with server's key, and sends it as a challenge.
- 4) Server decrypts, proving it has the corresponding private key.
- 5) **This value is used to derive symmetric session keys for encryption & MACs.**



How TLS Handles Data

1) Data arrives as a stream from the application via the TLS Socket



2) The data is segmented by TLS into chunks



3) A session key is used to encrypt and MAC each chunk to form a TLS “record”, which includes a short header and data that is encrypted, as well as a MAC.

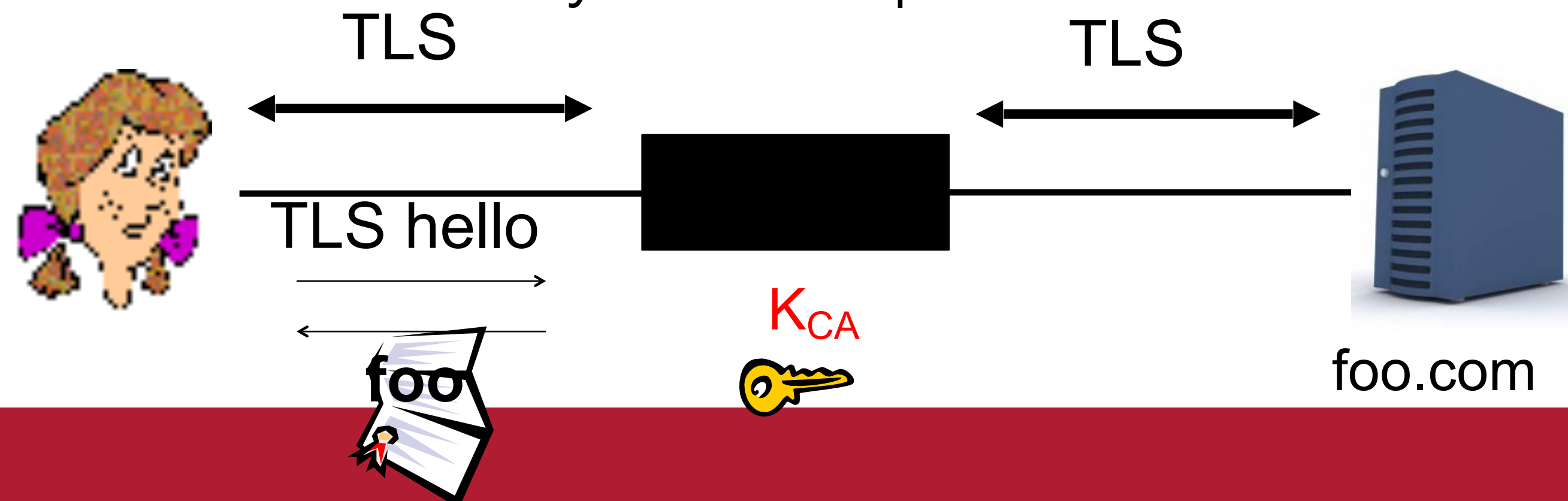


4) Records form a byte stream that is fed to a TCP socket for transmission.



Middleboxes + TLS :(

- Middleboxes are very widely used in the Internet
 - Companies have firewalls
 - Cellular operators use caches, compression, ...
- But TLS makes middleboxes ineffective
- “Solution”: install fake root certificate on device
 - Common for corporate networks
 - Sometimes also done by service providers



BONUS CONFIDENTIALITY TIME



Does TLS keep who you are talking to confidential?



TLS gives confidentiality, but not anonymity.

Anonymity is confidentiality for *who is talking*, not just *what they are saying*.



Do we even want anonymity?



Chaum's Mix

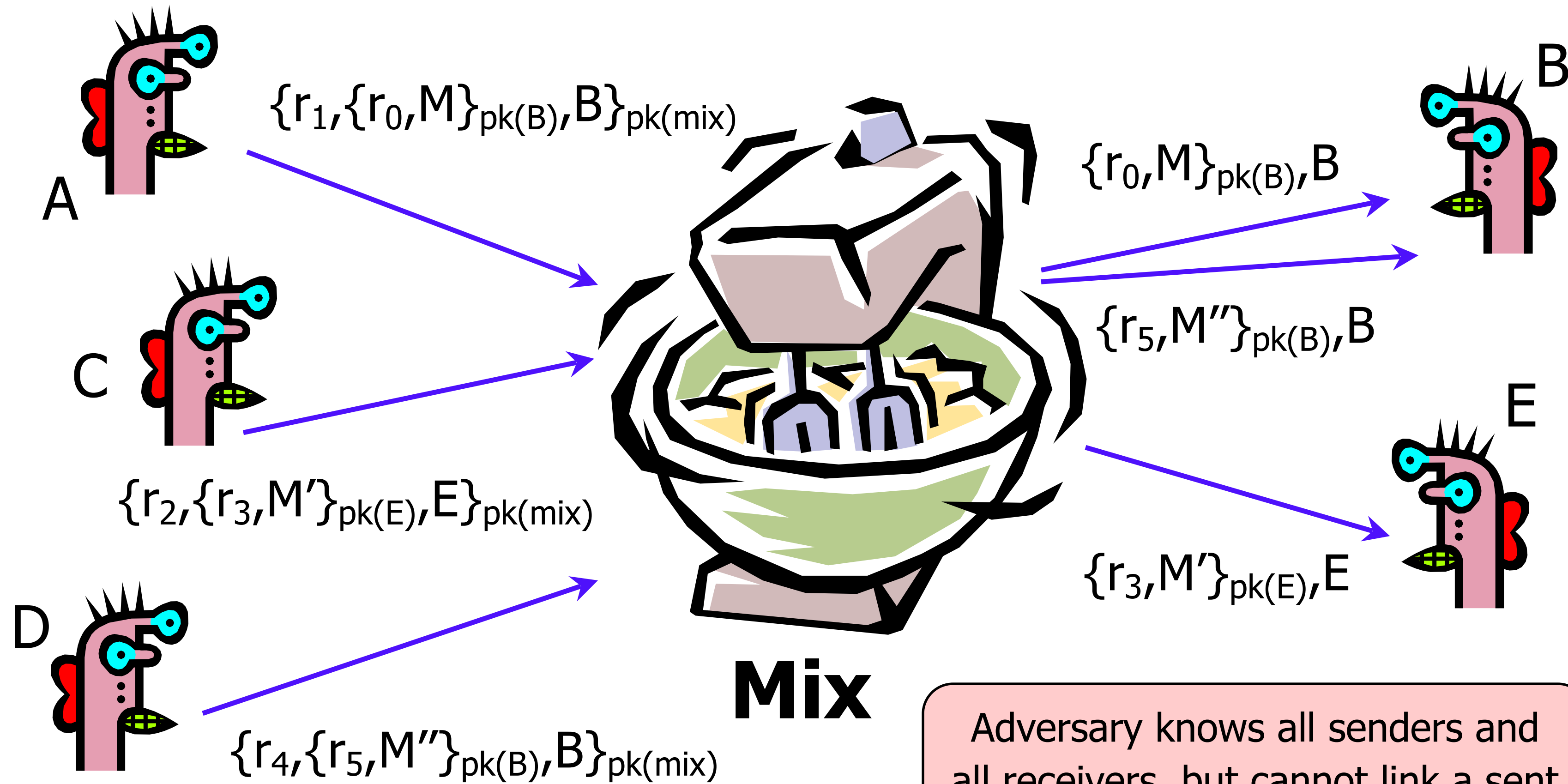
- Early proposal for anonymous email
 - David Chaum. “Untraceable electronic mail, return addresses, and digital pseudonyms”. Communications of the ACM, February 1981.

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



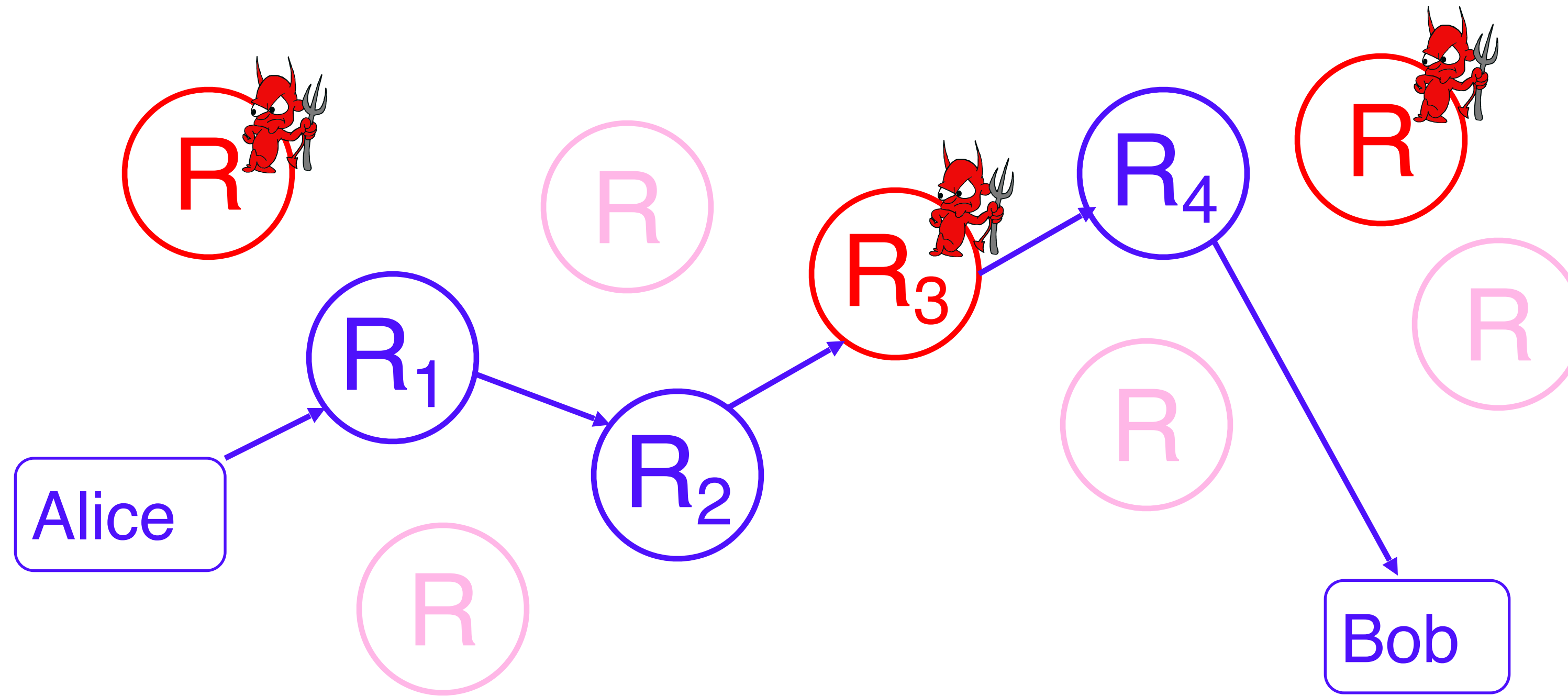
How can a basic mix help privacy?
How can a basic mix go wrong?



Modern anonymity networks: Tor & Onion Routing



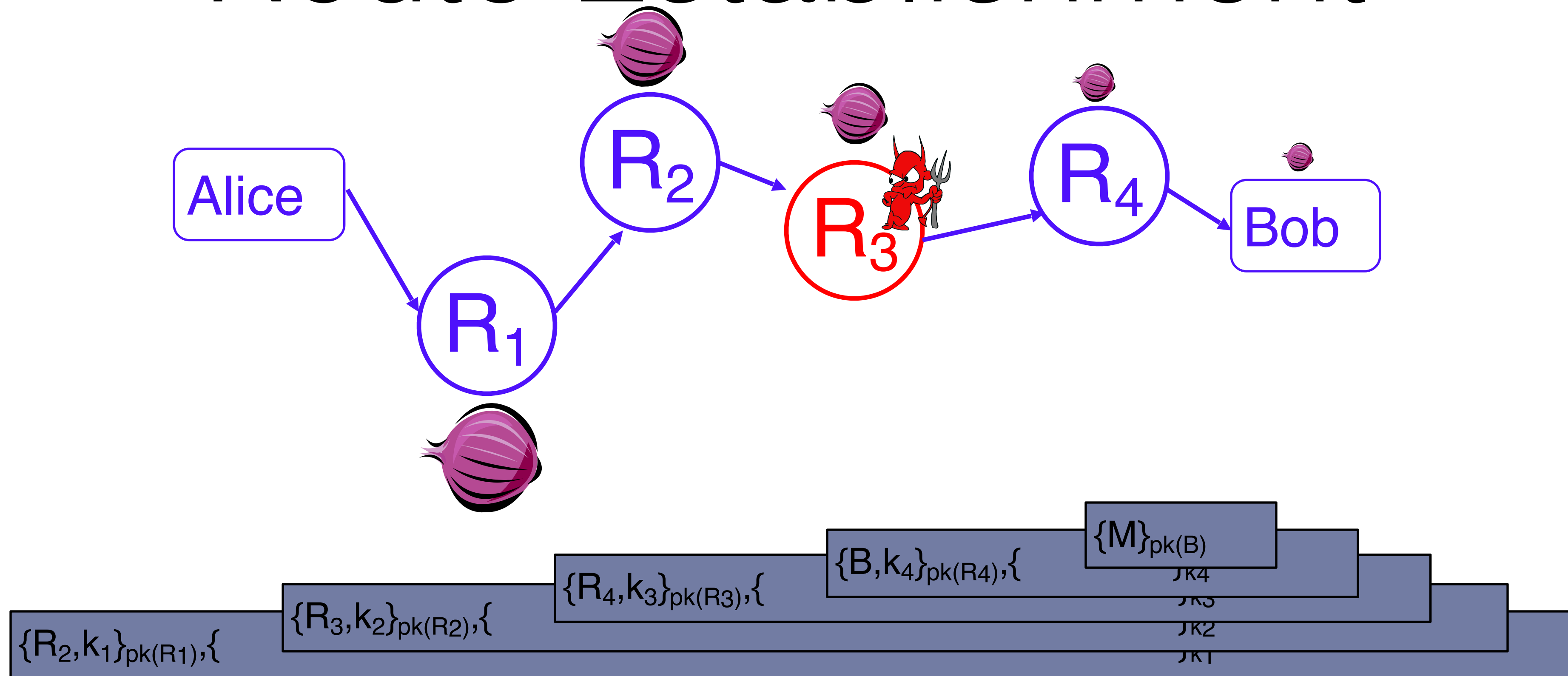
Onion Routing



- ▶ 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



Anonymity Activity



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
- 100 nodes on four continents, thousands of users
- “Easy-to-use” client proxy
 - Freely available, can use it for anonymous browsing



Have any of y'all used Tor
before?



Summary

- Internet design and growth => security challenges
- Symmetric (pre-shared key, fast) and asymmetric (key pairs, slow) primitives provide:
 - Confidentiality
 - Integrity
 - Authentication
- “Hybrid Encryption” leverages strengths of both.
- Great complexity exists in securely acquiring keys.
- Anonymity remains a great challenge in networking.

