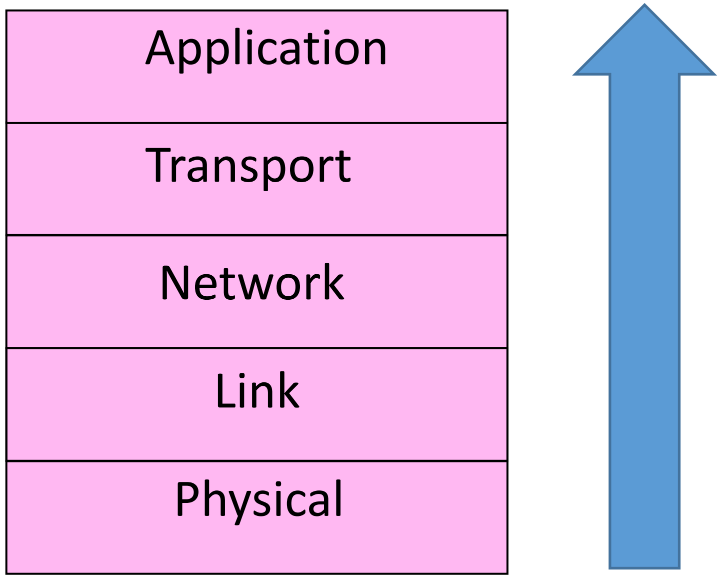
Network Security

Where we are in the Course

- Security crosses all layers



Security Threats

- "Security" is like "performance"
  - Means many things to many people
  - Must define the properties we want
- Key part of network security is clearly stating the <u>threat model</u>
  - The dangers and attacker's abilities
  - Can't assess risk otherwise

Security Threats (2)

- Some example threats
  - It’s not all about encrypting messages

Attacker	**Ability**	**Threat**
Eavesdropper	Intercept messages	Read contents of message
Observer	Inspect packet destinations	Collect conversations
Intruder	Compromised host	Tamper with contents of message
Impersonator	Remote social engineering	Trick party into giving information
Extortionist	Remote / botnet	Disrupt network services

Risk Management

- Security is hard as a negative goal
  - Try to ensure security properties and don’t let anything bad happen!
- End-to-end principle in action (can’t trust network!)
- Only as secure as the weakest link
  - Could be design flaw or bug in code
  - But often the weak link is elsewhere …

Risk Management (2)

- 802.11 security … early on, WEP:
  - Cryptography was flawed; can run cracking software to read WiFi traffic
- Today, WPA2/802.11i security:
  - Computationally infeasible to break!
- So that means 802.11 is secure against eavesdropping?

Risk Management (3)

- Many possible threats
  - We just made the first one harder!
  - 802.11 is more secure against eavesdropping in that the risk of successful attack is lower. But it is not “secure”.

Threat Model	**Old WiFi (WEP)**	**New WiFi (WPA2)**
Break encryption from outside	Very easy	Very difficult
Guess WiFi password	Often possible	Often possible
Get password from computer	May be possible	May be possible
Physically break into home	Difficult	Difficult

Cryptography

Cryptology

- Rich history, especially spies / military
  - From the Greek "hidden writing"
- Cryptography
  - Focus is encrypting information
- Cryptanalysis
  - Focus is how to break codes
- Modern emphasis is on codes that are "computationally infeasible" to break
  - Takes too long compute solution

Uses of Cryptography

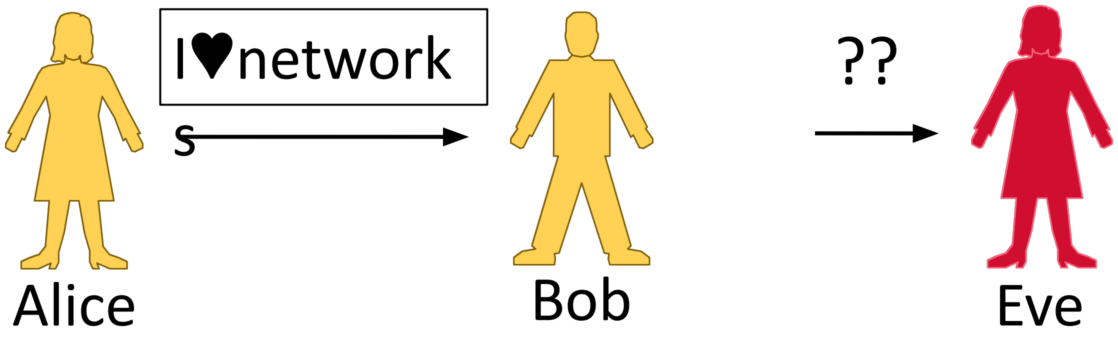
- Encrypting information is useful for more than deterring eavesdroppers
  - Prove message came from real sender
  - Prove remote party is who they say
  - Prove message hasn't been altered
- Designing secure cryptographic scheme tricky!
  - Use approved design (library) in approved way

Internet Reality

- Most of the protocols were developed before the Internet grew popular
  - It was a smaller, more trusted world
  - So protocols lacked security …
- We have strong security needs today
  - Clients talk with unverified servers
  - Servers talk with anonymous clients
  - Security has been retrofitted
  - This is far from ideal!

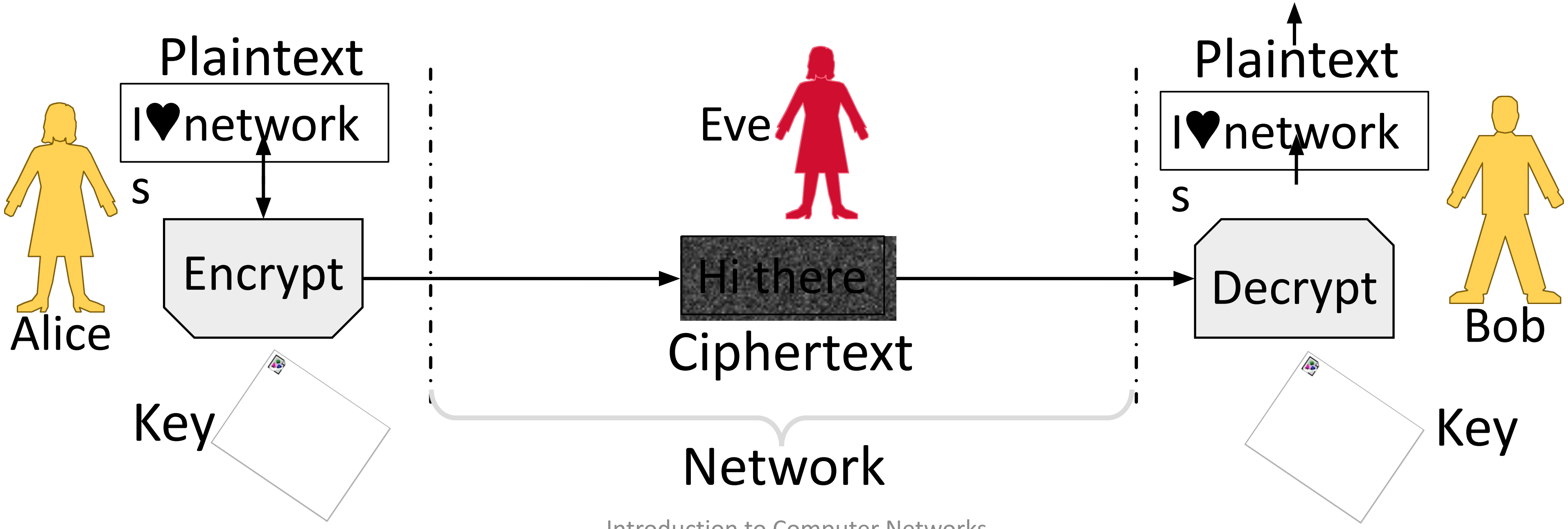
Goal and Threat Model

- Goal is to send a private message from Alice to Bob
  - This is called confidentiality
- Threat is Eve will read the message
  - Eve is a passive adversary (observes)



Encryption/Decryption Model

- Alice encrypts private message (<u>plaintext</u>) using key
- Eve sees <u>ciphertext</u> but not plaintext
- Bob decrypts using key to get the private message



Encryption/Decryption (2)

- Encryption is a reversible mapping
  - Ciphertext is encrypted plaintext
- Assume attacker knows algorithm
  - Security does not rely on its secrecy
- Algorithm is parameterized by keys
  - Security does rely on key secrecy
  - Must be distributed (Achilles' heel)

Encryption/Decryption (3)

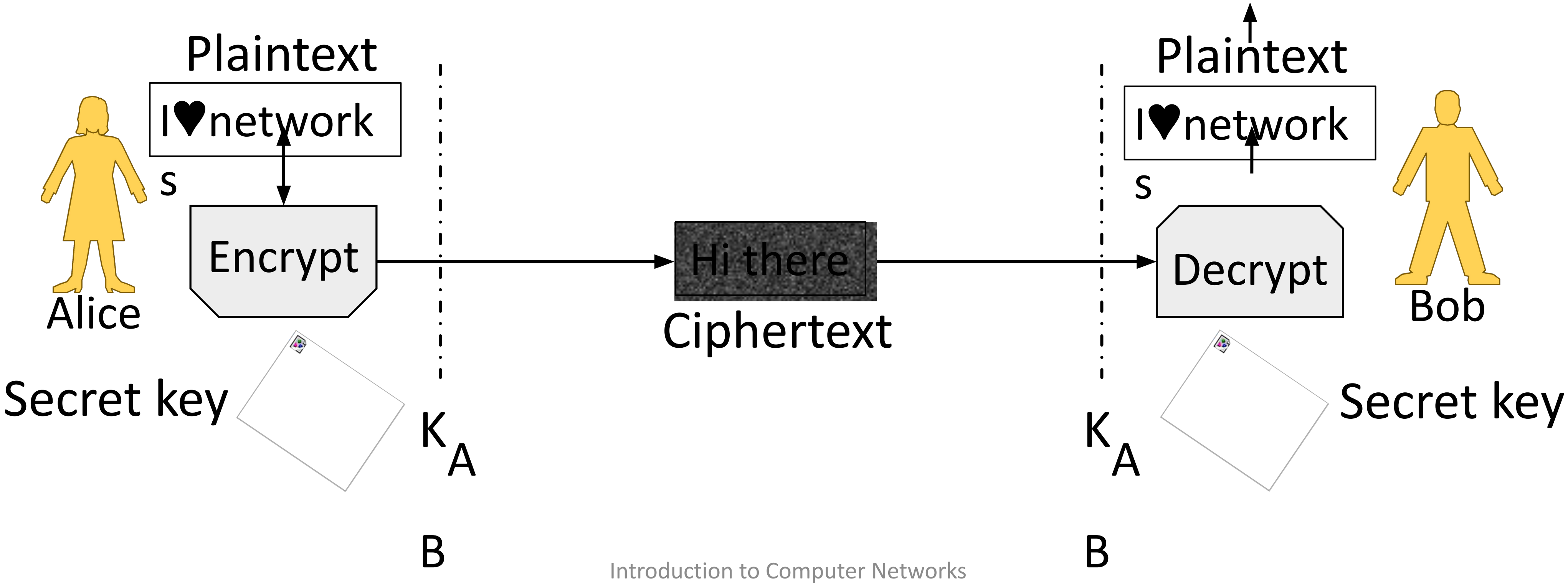
Two main kinds of encryption:

1. Symmetric key encryption », e.g., AES
   - Alice and Bob share secret key
   - Encryption is a bit mangling box

2. Public key encryption », e.g., RSA
   - Alice and Bob each have a key in two parts: a public part (widely known), and a private part (only owner knows)
   - Encryption is based on mathematics (e.g., RSA is based on difficulty of factoring)

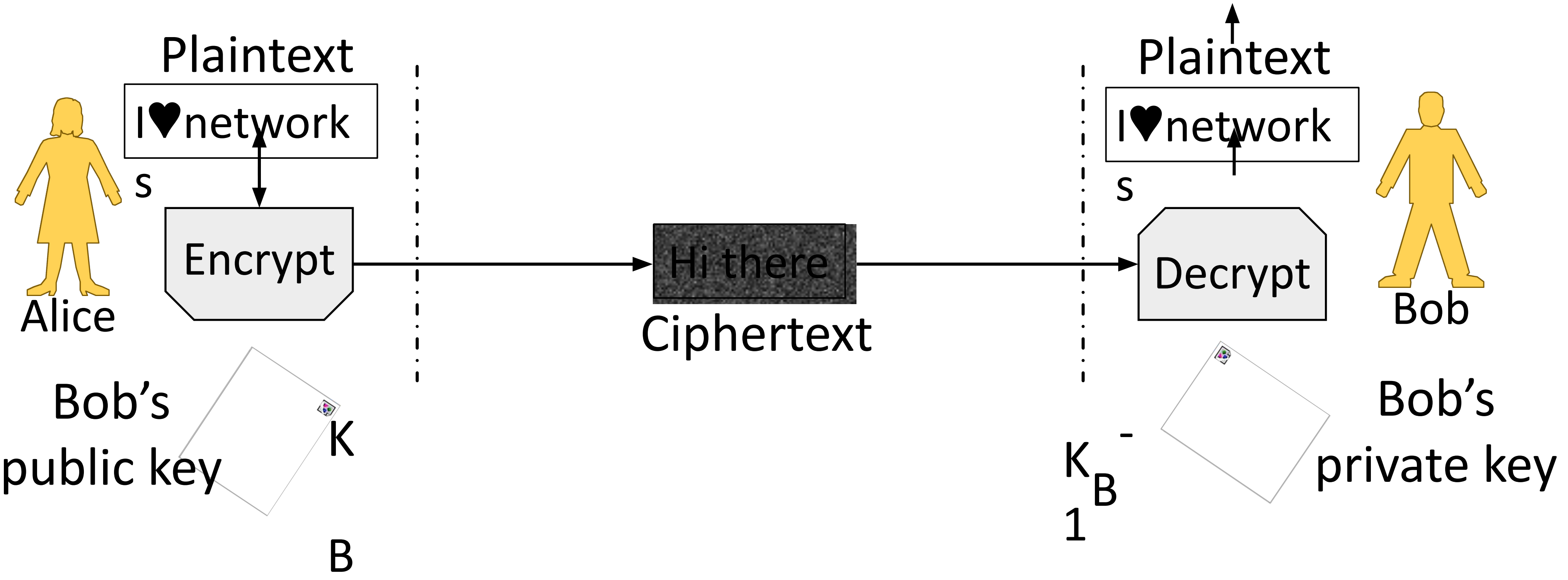
Symmetric (Secret Key) Encryption

- Alice and Bob have the same secret key, K_{AB}
  - Anyone with the secret key can encrypt/decrypt



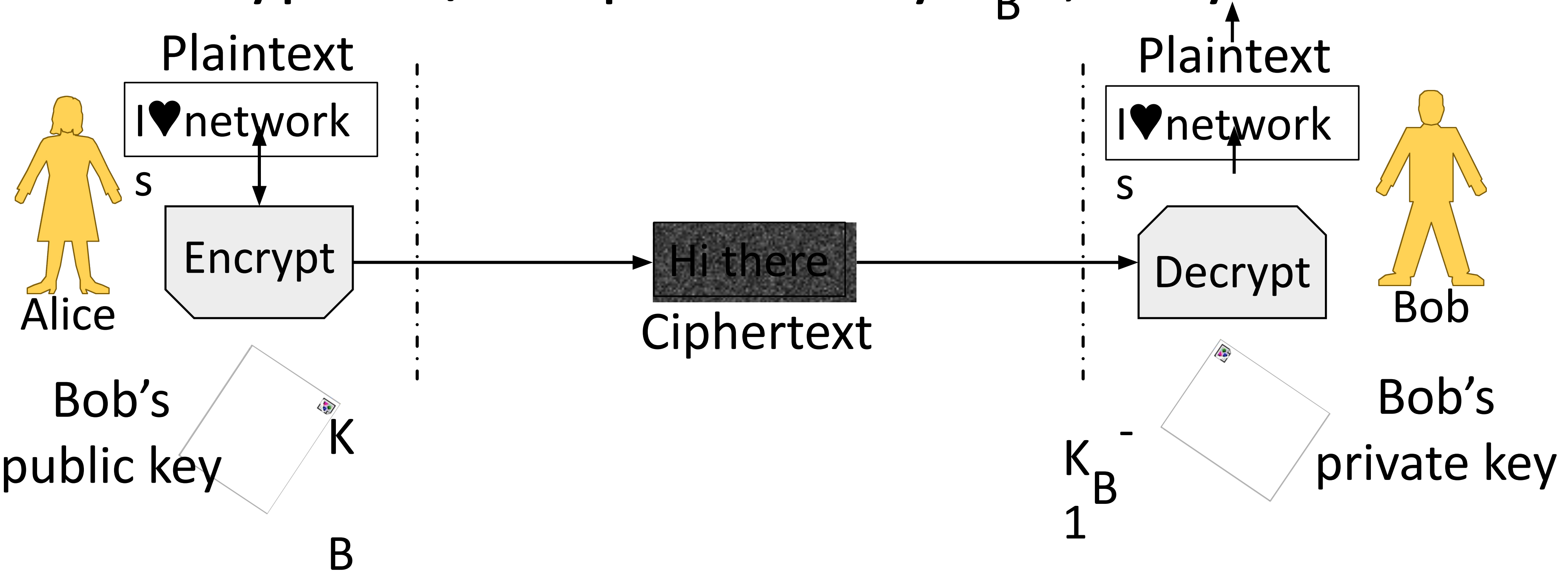
Public Key (Asymmetric) Encryption

- Alice and Bob have public/private key pairs (K_B / K_B^{-1})
  - Public keys are well-known, private keys are secret

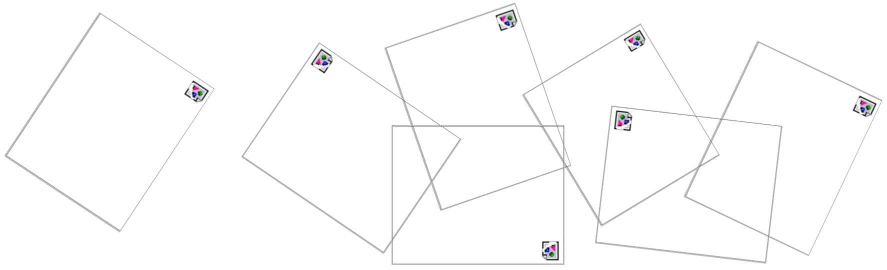


Public Key Encryption (2)

- Alice encrypts w/ Bob's pubkey K_B; anyone can send
- Bob decrypts w/ his private key K_B^{-1}; only he can



Key Distribution



- This is a big problem on a network!
  - Often want to talk to new parties
- Symmetric encryption problematic
  - Have to first set up shared secret
- Public key idea has own difficulties
  - Need trusted directory service
  - We'll look at certificates later

Symmetric vs. Public Key

- Have complementary properties
  - Want the best of both!

Property	Symmetric	Public Key
Key Distribution	Hard – share secret per pair of users	Easier – publish public key per user
Runtime Performanc e	Fast – good for high data rate	Slow – few, small, messages

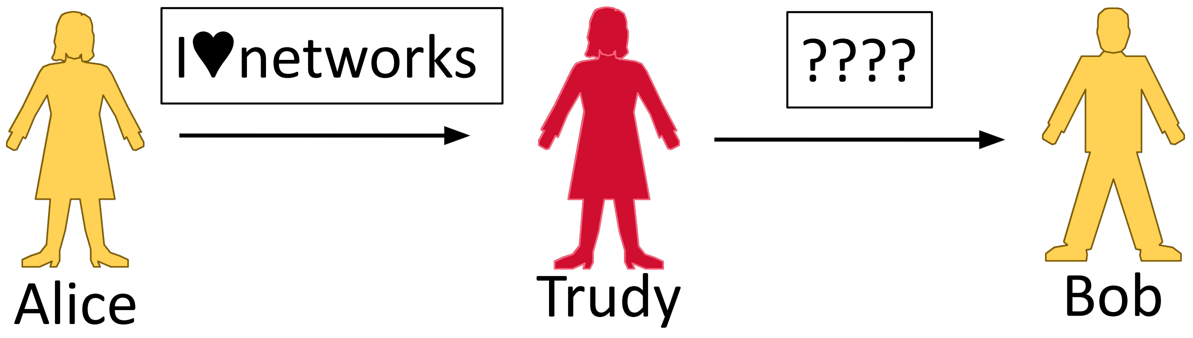
Winning Combination

- Alice uses public key encryption to send Bob a small private message
  - It's a key! (Say 256 bits.)
- Alice/Bob send messages with symmetric encryption
  - Using the key they now share
- The key is called a session key
  - Generated for short-term use

Message Authentication

Goal and Threat Model

- Goal is for Bob to verify the message is from Alice and unchanged
  - This is called integrity/authenticity
- Threat is Trudy will tamper with messages
  - Trudy is an active adversary (interferes)



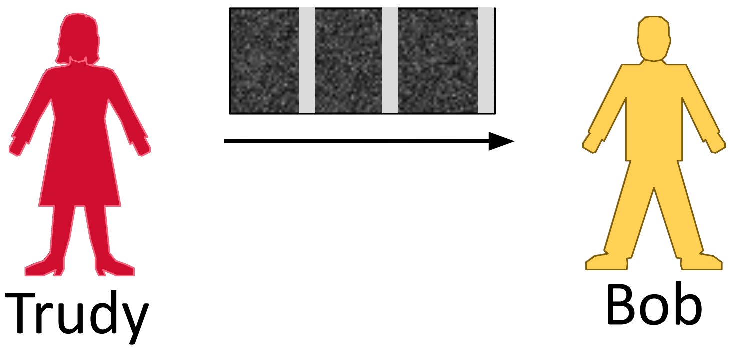
Wait a Minute!

- We’re already encrypting messages to provide confidentiality

- Why isn’t this enough?

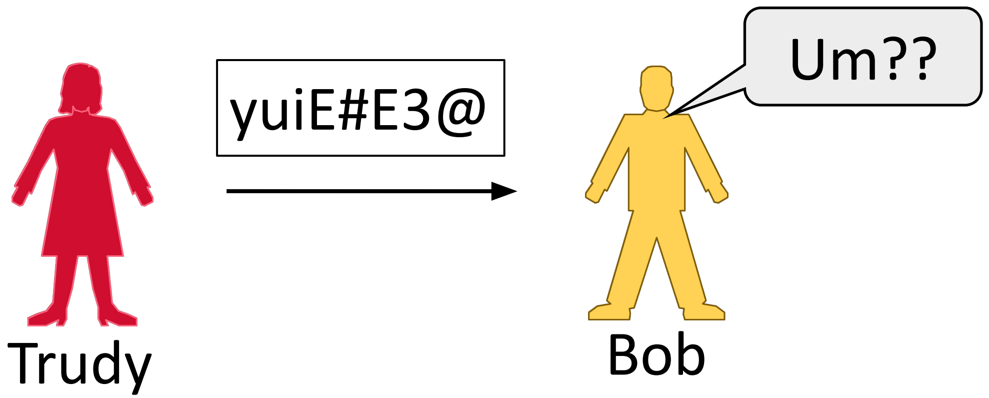
Encryption Issues

- What will happen if Trudy flips some of Alice's message bits?
  - Bob will decrypt it, and …



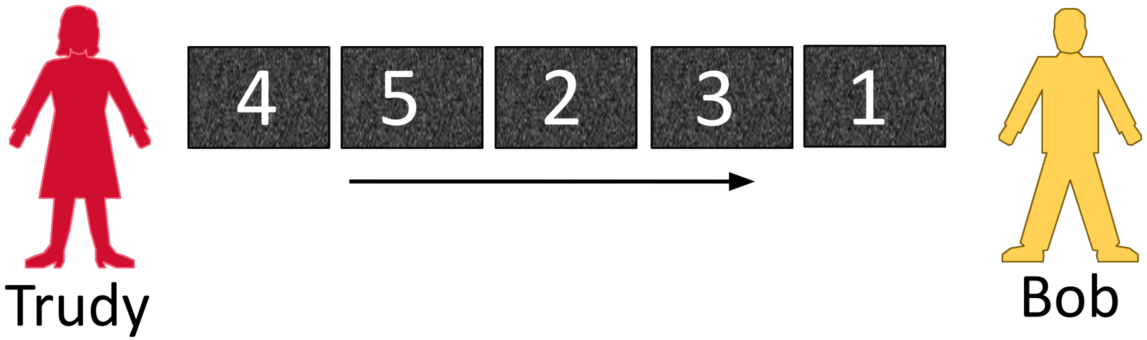
Encryption Issues (2)

- What will happen if Trudy flips some of Alice's message bits?
  - Bob will receive an altered message



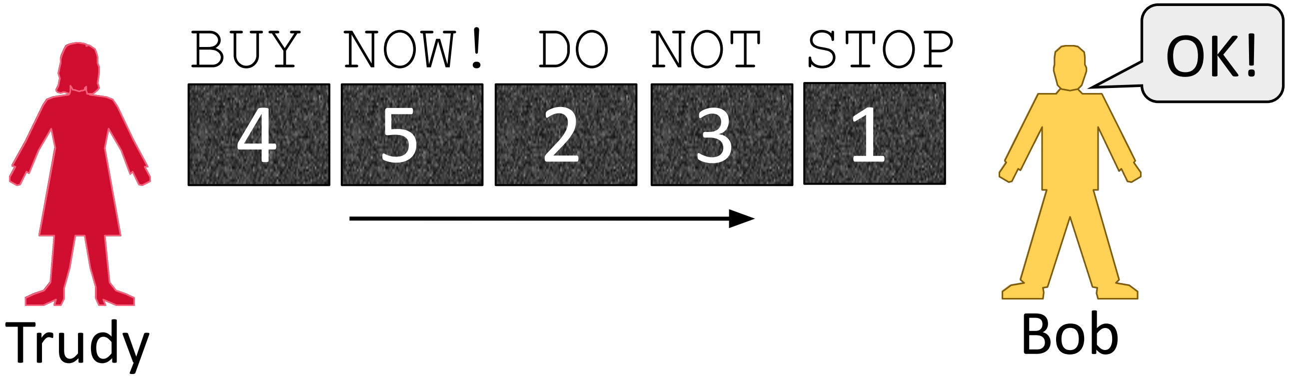
Encryption Issues (3)

- Typically encrypt blocks of data
- What if Trudy reorders message?
  - Bob will decrypt, and …



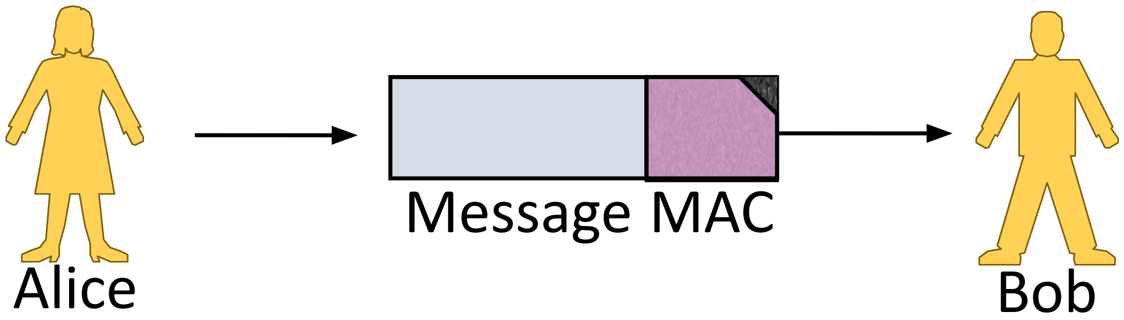
Encryption Issues (4)

- What if Trudy reorders message?
  - Bob will receive altered message



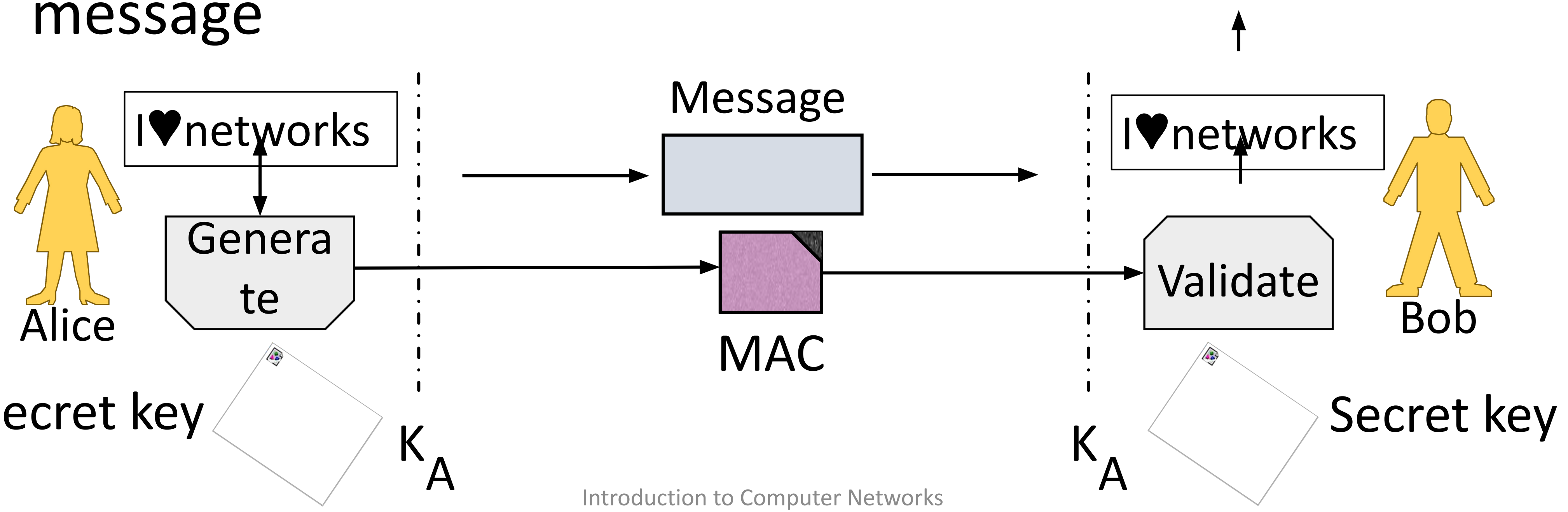
MAC (Message Authentication Code)

- MAC is a small token to validate the integrity/authenticity of a message
  - Conceptually ECCs again
  - Send the MAC along with message
  - Validate MAC, process the message
  - Example: HMAC scheme



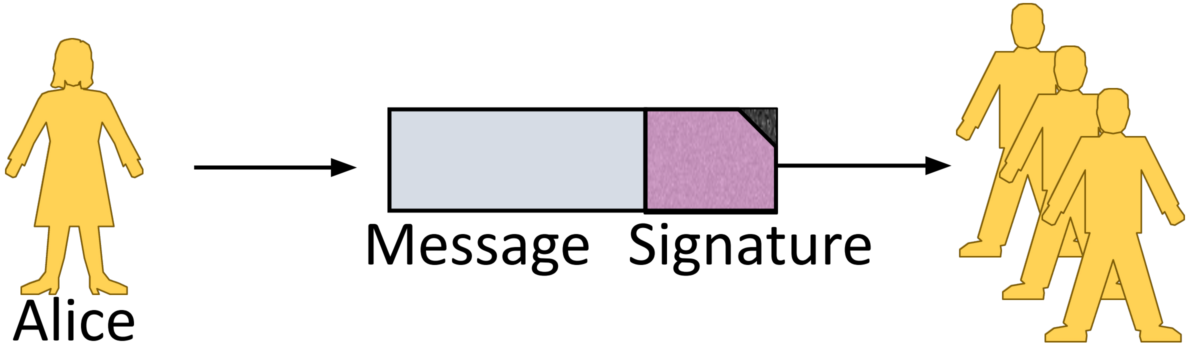
MAC (2)

- Sorta symmetric encryption operation – key shared
  - Lets Bob validate unaltered message came from Alice
  - Doesn't let Bob convince Charlie that Alice sent the message



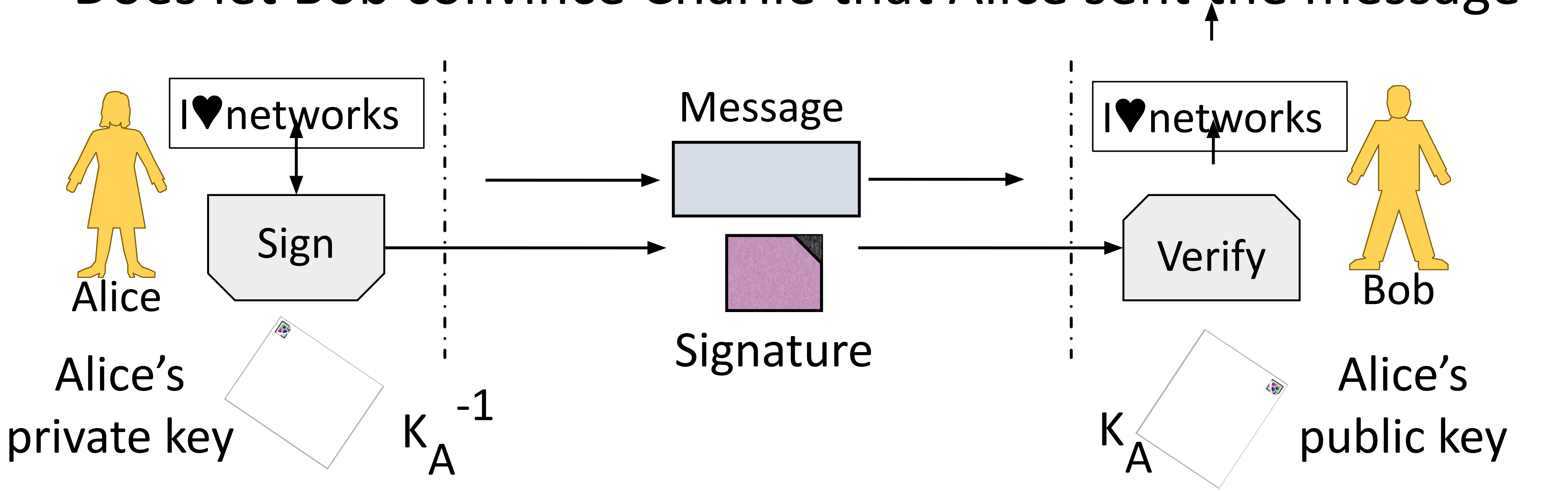
Digital Signature

- Signature validates the integrity/authenticity of message
  - Send it along with the message
  - Lets all parties validate
  - Example: RSA signatures



Digital Signature (2)

- Kind of public key operation – pub/priv key parts
  - Alice signs w/ private key, K_A^{-1}, Bob verifies w/ public key, K_A
  - Does let Bob convince Charlie that Alice sent the message

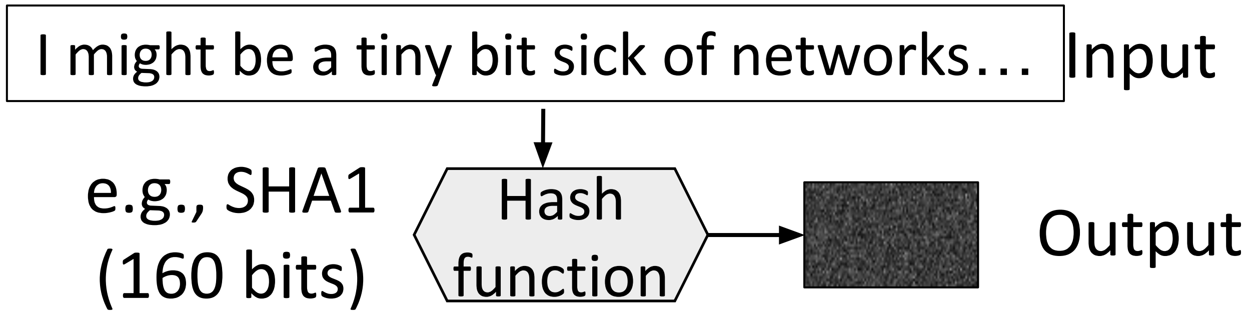


Speeding up Signatures

- Same tension as for confidentiality:
  - Public key has keying advantages
  - But it has slow performance!
- Use a technique to speed it up
  - <u>Message digest</u> stands for message
  - Sign the digest instead of full message

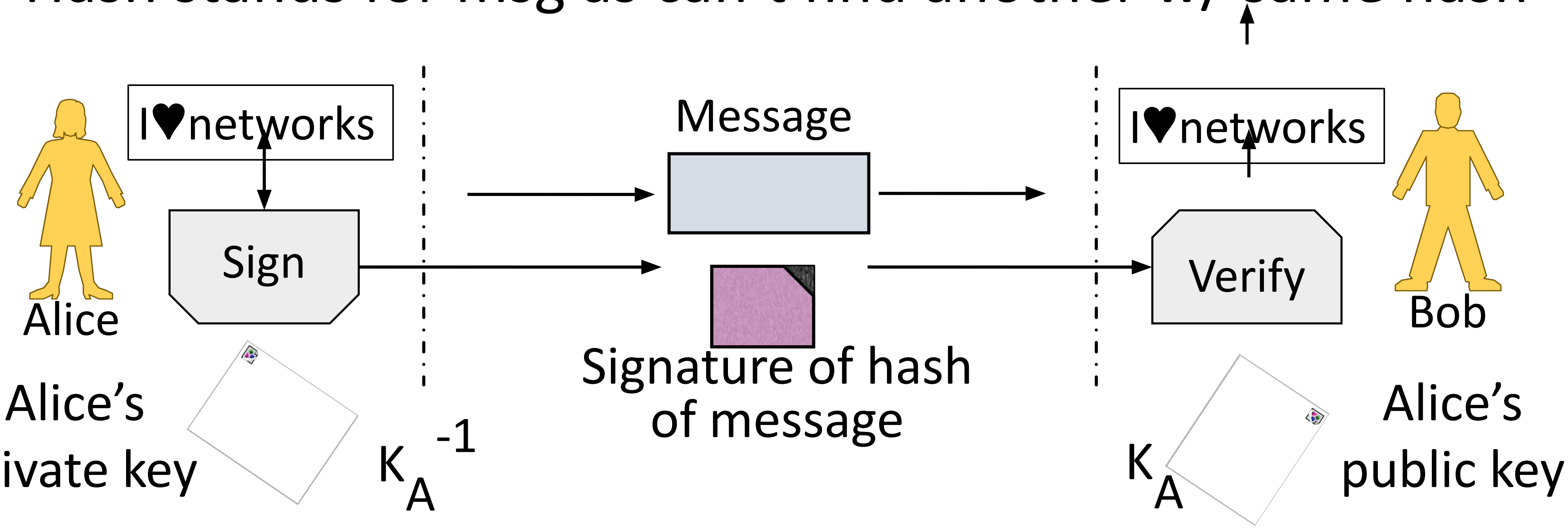
Message Digest or Cryptographic Hash

- Digest/Hash is a secure checksum
  - Deterministically mangles bits to pseudo-random output (like CRC)
  - Can’t find messages with same hash
  - Acts as a fixed-length descriptor of message – very useful!



Speeding up Signatures (2)

- Conceptually similar except sign the hash of message
  - Hash is fast to compute, so it speeds up overall operation
  - Hash stands for msg as can't find another w/ same hash

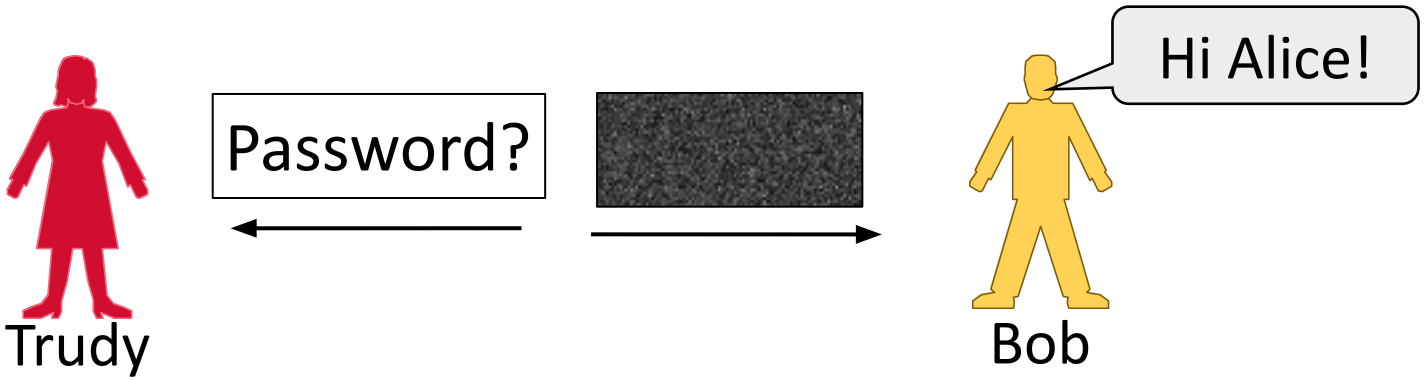


Preventing Replays

- We normally want more than confidentiality, integrity, and authenticity for secure messages!
  - Want to be sure message is fresh
- Need to distinguish message from <u>replays</u>
  - Repeat of older message
  - Acting on it again may cause trouble

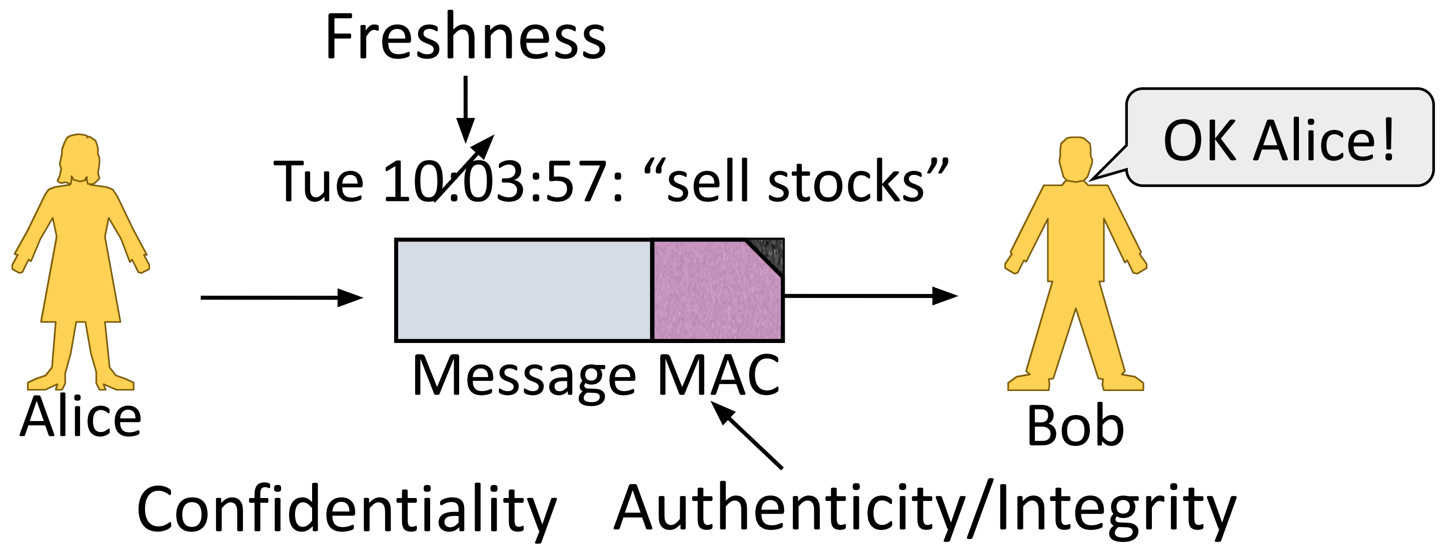
Preventing Replays (2)

- Replay attack:
  - Trudy records Alice's messages to Bob
  - Trudy later replays them (unread) to Bob
    - She pretends to be Alice



Preventing Replays (3)

- To prevent replays, include a proof of freshness in the messages
  - Use a timestamp, or <u>nonce</u>



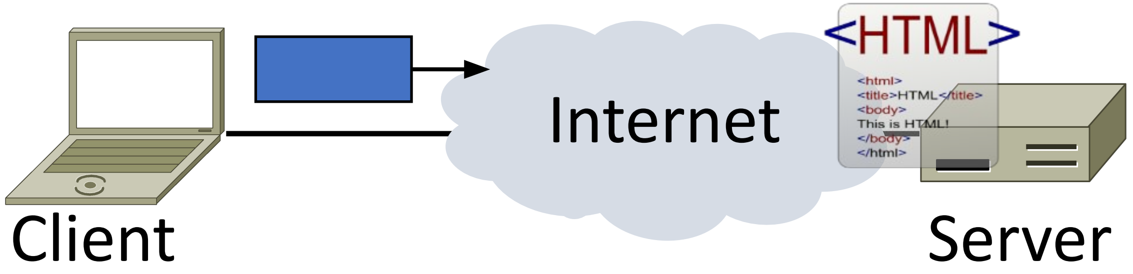
Takeaway

- Cryptographic designs can give us integrity, authenticity and freshness as well as confidentiality.
- Real protocol designs combine the properties in different ways
  - We'll see some examples
  - Note many pitfalls in how to combine, as well as in the primitives themselves

Web Security

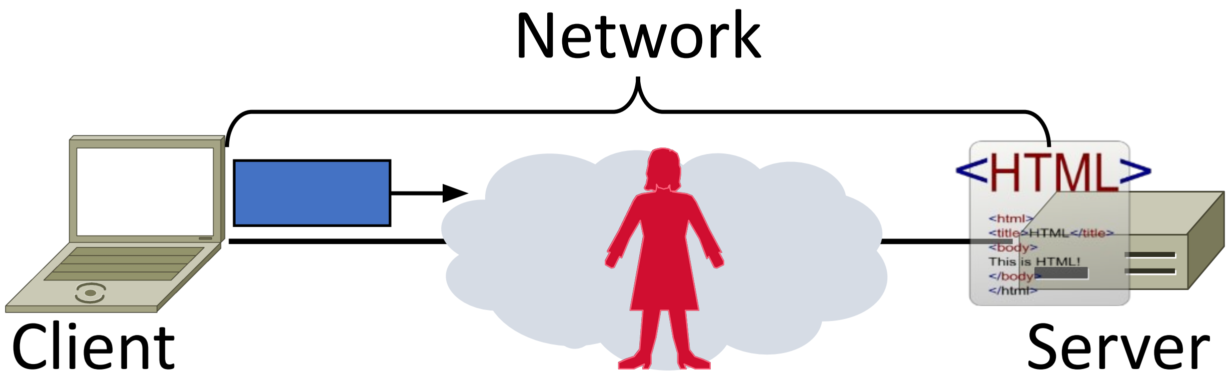
Goal and Threat Model

- Much can go wrong on the web!
  - Clients encounter malicious content
  - Web servers are target of break-ins
  - Fake content/servers trick users
  - Data sent over network is stolen …



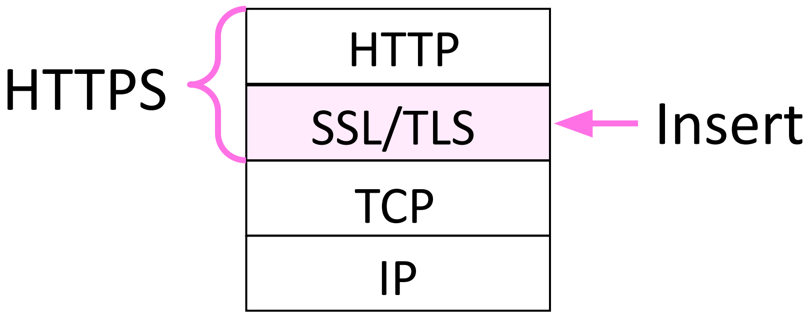
Goal and Threat Model (2)

- Goal of HTTPS is to secure HTTP
- We focus on network threats:
  1. Eavesdropping client/server traffic
  2. Tampering with client/server traffic
  3. Impersonating web servers



HTTPS Context

- HTTPS (HTTP Secure) is an add-on
  - Means HTTP over SSL/TLS
  - SSL (Secure Sockets Layer) precedes TLS (Transport Layer Security)

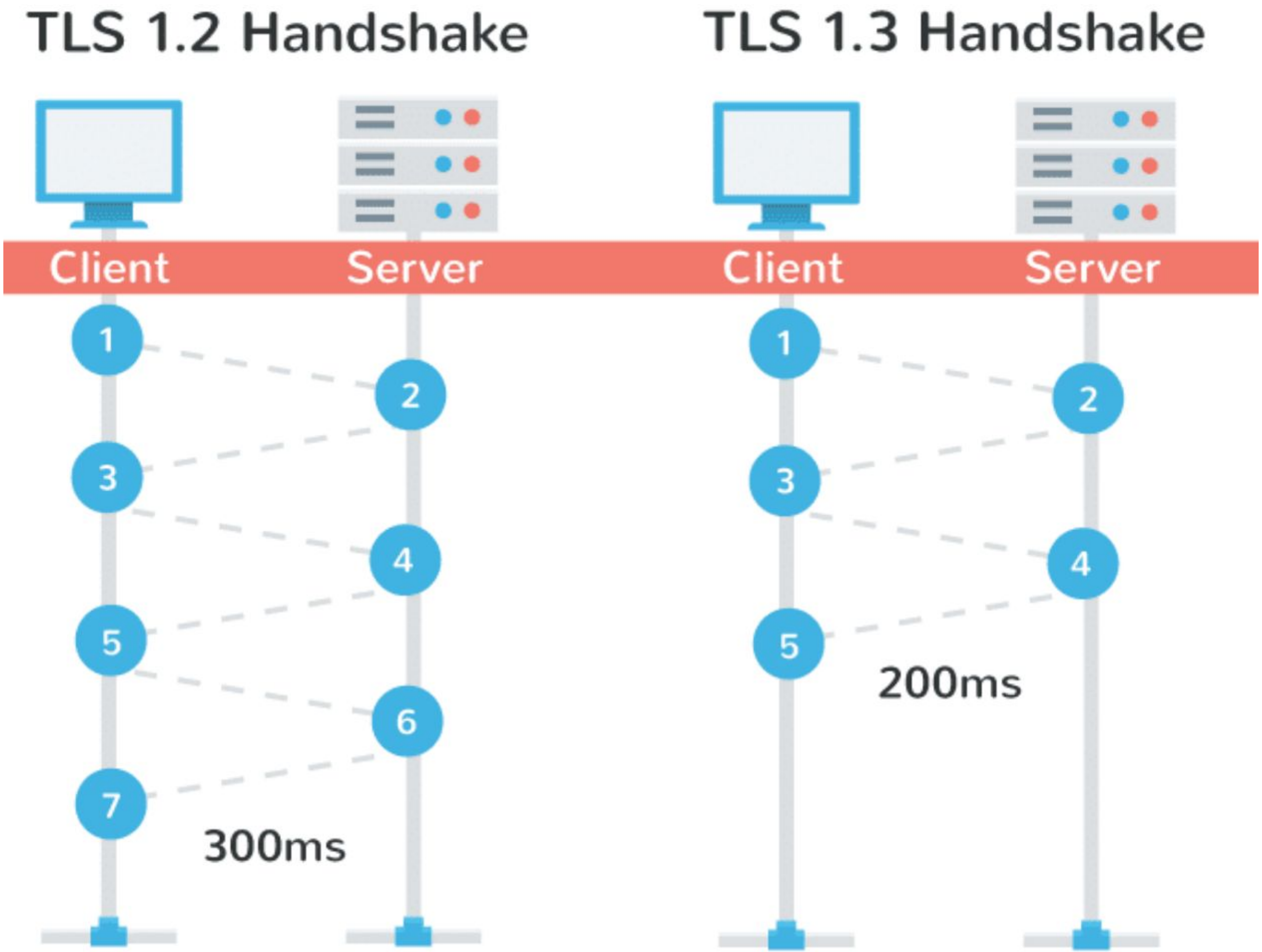


HTTPS Context (2)

- SSL came out of Netscape
  - SSL2 (flawed) made public in ‘95
  - SSL3 fixed flaws in ‘96
- TLS is the open standard
  - TLS 1.0 in ‘99, 1.1 in ‘06, 1.2 in ‘08, 1.3 in ‘18
- Motivated by secure web commerce
  - Slow adoption, now widespread use
  - Can be used by any app, not just HTTP

TLS 1.3

- Motivation 1: Strengthen security
  - Remove bad cyphers: SHA-1, RC4, DES, 3DES, AES-CBC, MD5, Arbitrary Diffie-Hellman groups, etc
  - Simplify configuration
- Motivation 2: Speed up protocol
  - 2 RTTs $\longrightarrow$ 1 RTT
  - 0 RTT (resumption) possible if site has been recently been visited


TLS 1.2 Handshake
TLS 1.3 Handshake
Client
Server
Client
Server
1
2
3
4
5
6
7
300ms
1
2
3
4
5
200ms

TLS 1.3 - OTHER

Usage | % of all users | ?

Global 75.59% + 1.71% = 77.3%

Version 1.3 (the latest one) of the Transport Layer Security (TLS) protocol. Removes weaker elliptic curves and hash functions.

Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			74						
	17	67	75		12.1				4
11	18	68	76	5 12.1	12.3	all	75	12.12	9.2
	76	69	77	5 13	13				
		70	78	5 TP					
			79						

TLS 1.3 - OTHER

Usage % of all users ?

Global 90.29% + 0.84% = 91.12%

Version 1.3 (the latest one) of the Transport Layer Security (TLS) protocol. Removes weaker elliptic curves and hash functions.

Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			87							
			88	[4] 13.1		13.7				
11	88	85	89	14	73	14.4	all	88	12.12	13.0
		86	90	TP						
		87	91							
			92							

SSL Operation

- Protocol provides:
  1. Verification of identity of server (and optionally client)
  2. Message exchange between the two with confidentiality, integrity, authenticity and freshness
- Consists of authentication phase (that sets up encryption) followed by data transfer phase

SSL/TLS Authentication

- Must allow clients to securely connect to servers not used before
  - Client must authenticate server
  - Server typically doesn't identify client

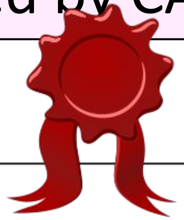
- Uses public key authentication
  - But how does client get server's key?
  - With certificates »

Certificates

- A certificate binds pubkey to identity, e.g., domain
  - Distributes public keys when signed by a party you trust
  - Commonly in a format called X.509

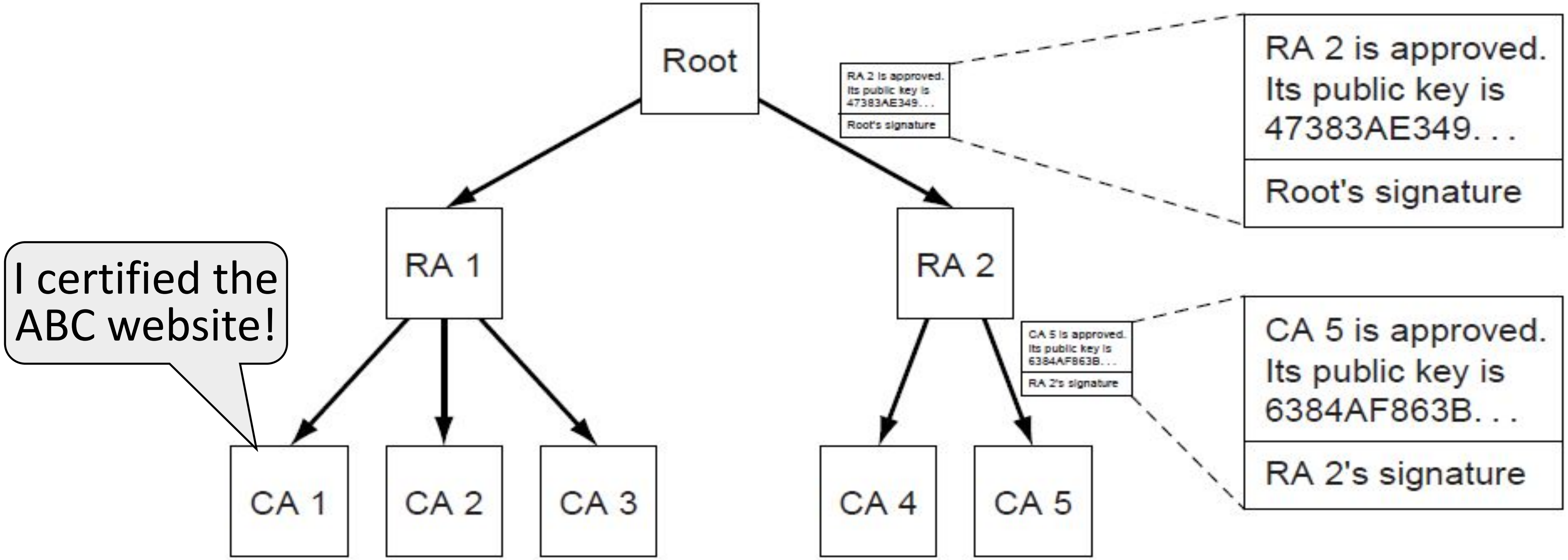
I hereby certify that the public key
    19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
belongs to
    Robert John Smith
    12345 University Avenue
    Berkeley, CA 94702
    Birthday: July 4, 1958
    Email: bob@superdupernet.com

Signed by CA



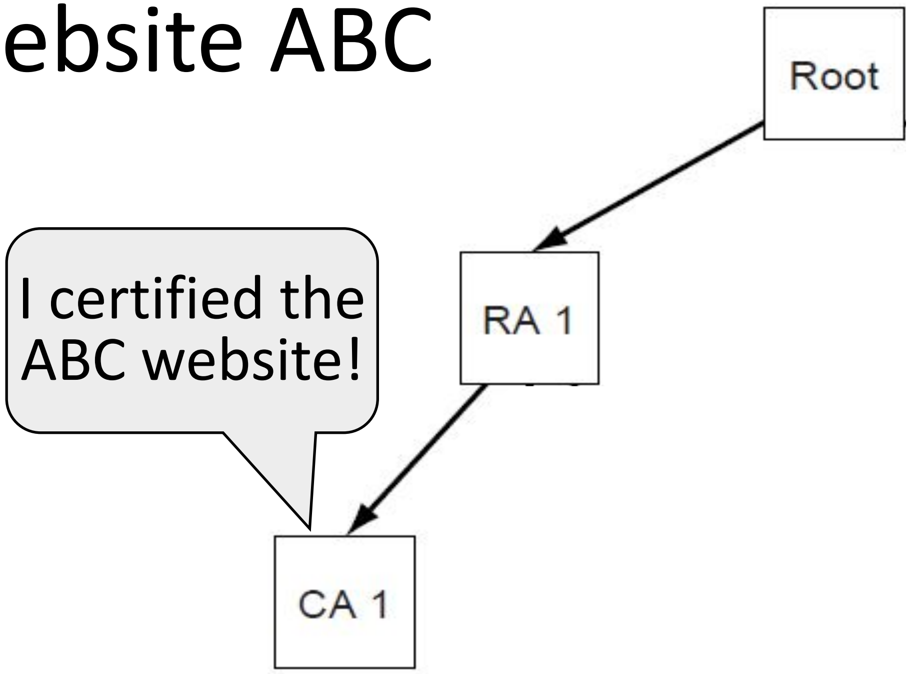
PKI (Public Key Infrastructure)

- Adds hierarchy to certificates to let parties issue
  - Issuing parties are called CAs (Certificate Authorities)



PKI (2)

- Need public key of PKI root and trust in servers on path to verify a public key of website ABC
  - Browser has Root's public key
  - {RA1's key is X} signed Root
  - {CA1's key is Y} signed RA1
  - {ABC's key Z} signed CA1



PKI (3)

- Browser/OS has public keys of the trusted roots of PKI
  - >100 root certificates!
  - Inspect your web browser

Certificate for wikipedia.org issued by DigiCert

PKI (4)

- Real-world complication:
  - Public keys may be compromised
  - Certificates must then be revoked
- PKI includes a CRL (Certificate Revocation List)
  - Browsers use to weed out bad keys