

Introduction to Computer Networks

Message Authentication

(§8.2-8.3, §8.4.2-8.4.3)



David Wetherall (djw@uw.edu)

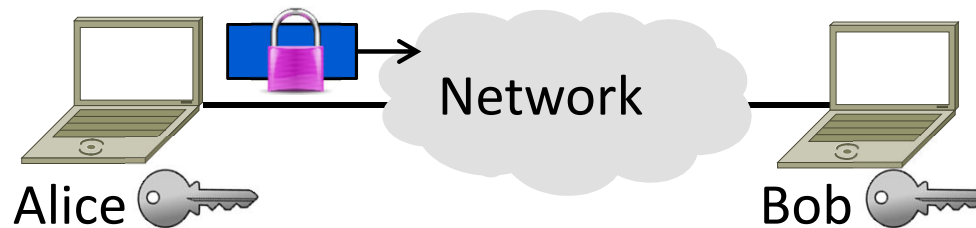
Professor of Computer Science & Engineering

UNIVERSITY *of* WASHINGTON

Topic

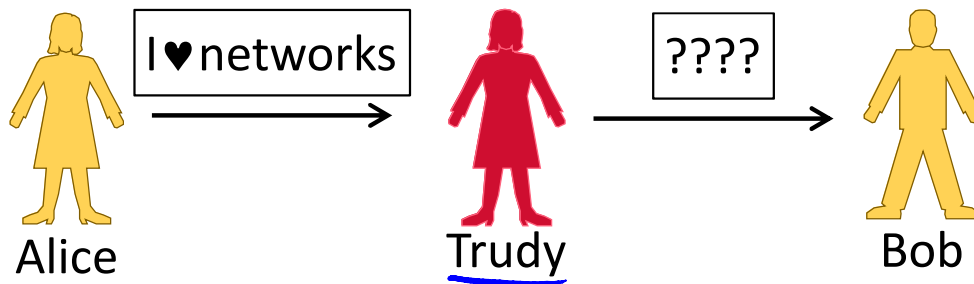


- Encrypting information to provide authenticity (=correct sender) and integrity (=unaltered)
 - Confidentiality isn't enough



Goal and Threat Model

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



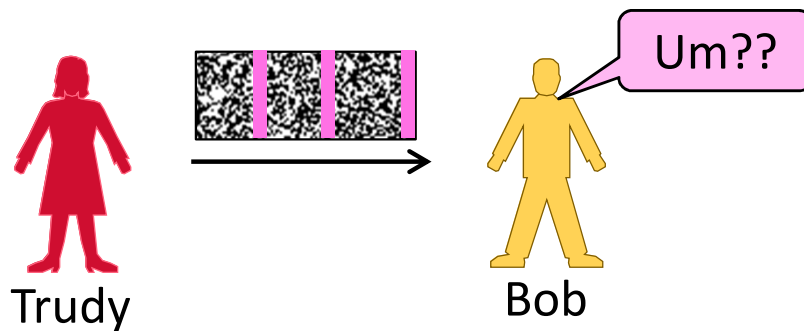
Wait a Minute!

- We're already encrypting messages to provide confidentiality
- Why isn't this enough?



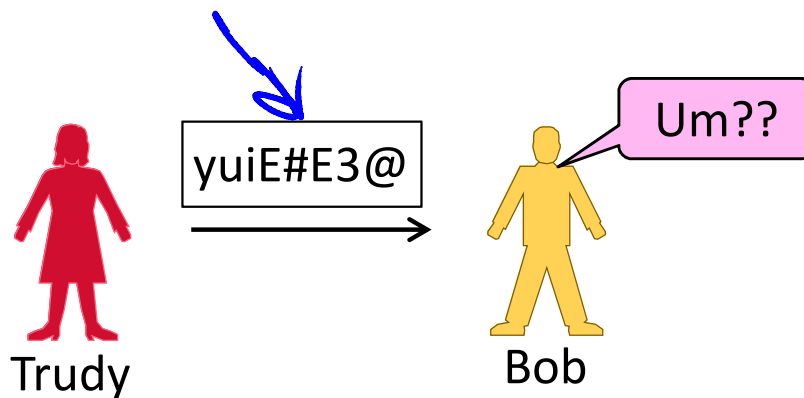
Encryption Issues

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



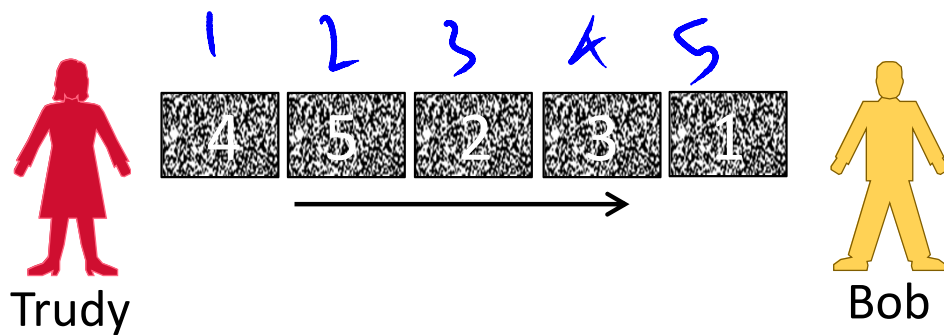
Encryption Issues (2)

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



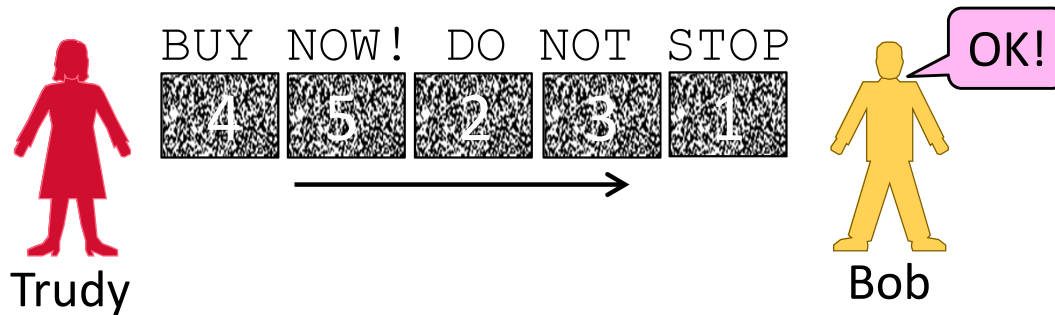
Encryption Issues (3)

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



Encryption Issues (4)

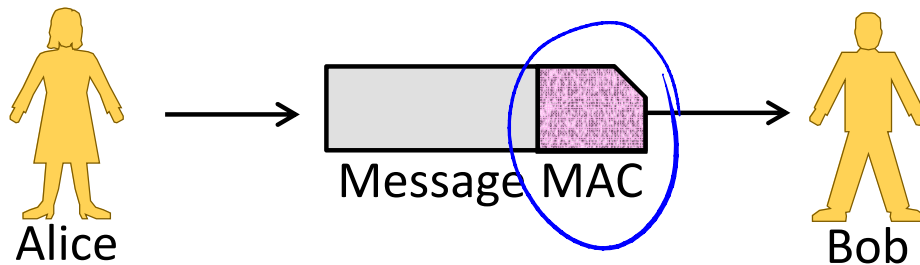
- What if Trudy reorders message?
 - Bob will receive altered message



- Should have been (Woops)
STOP DO NOT BUY NOW!

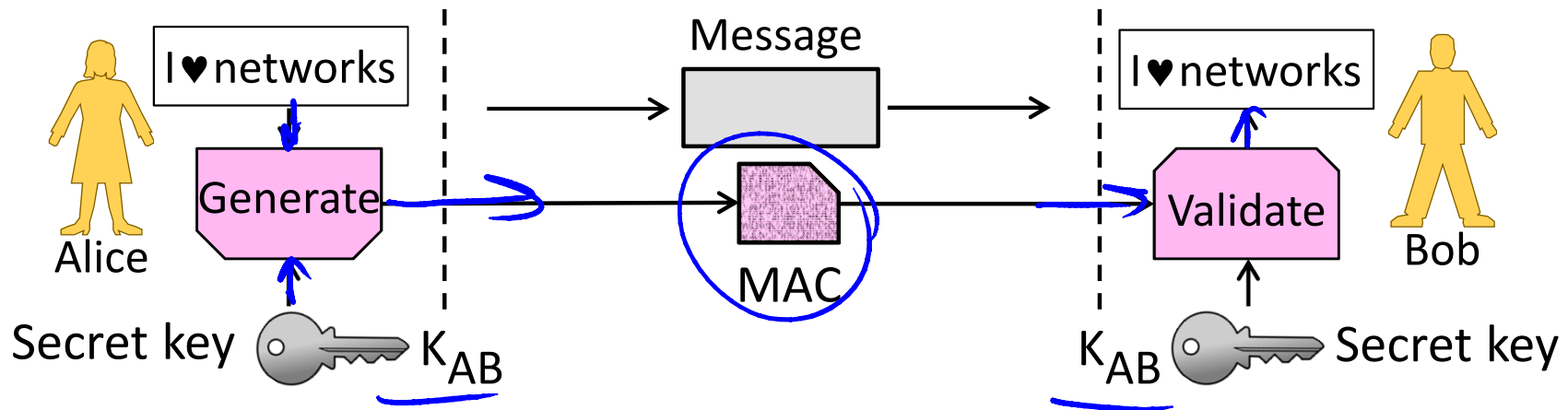
MAC (Message Authentication Code)

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



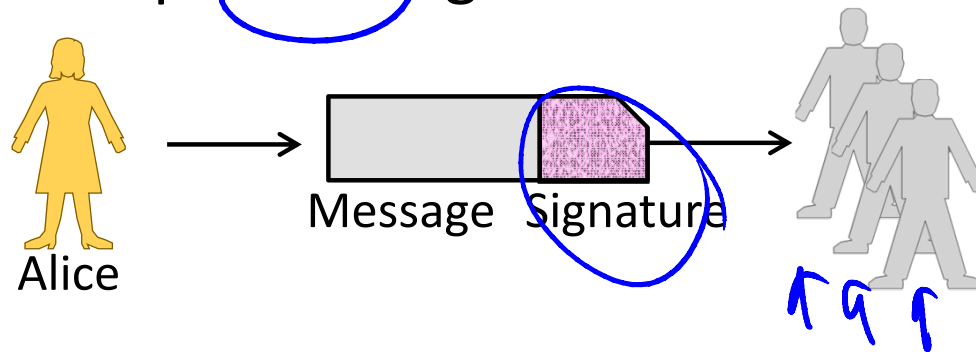
MAC (2)

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



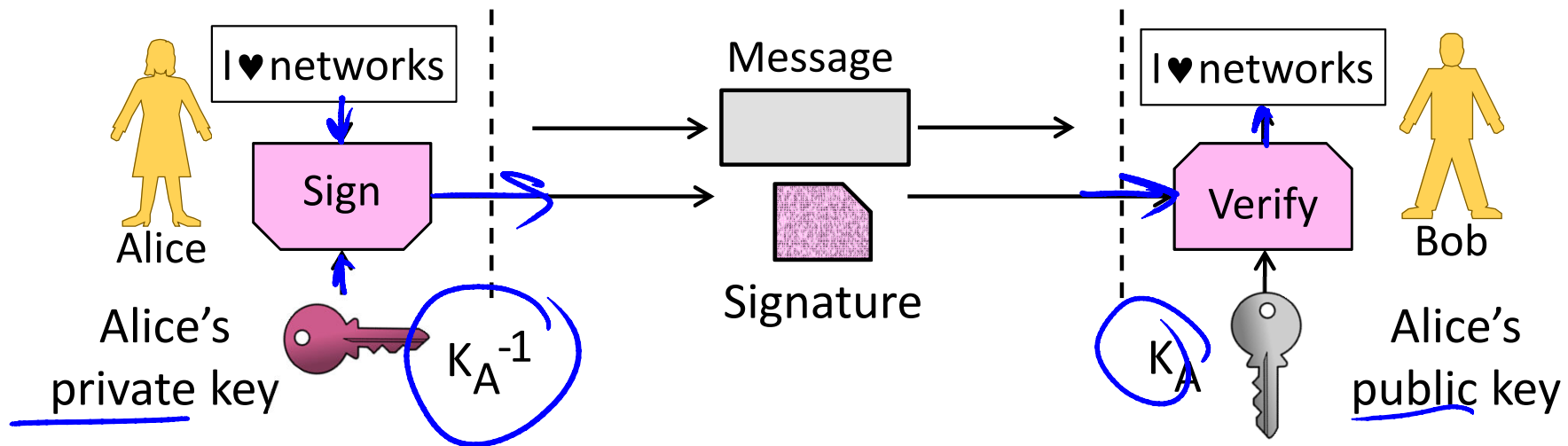
Digital Signature

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




Digital Signature (2)

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

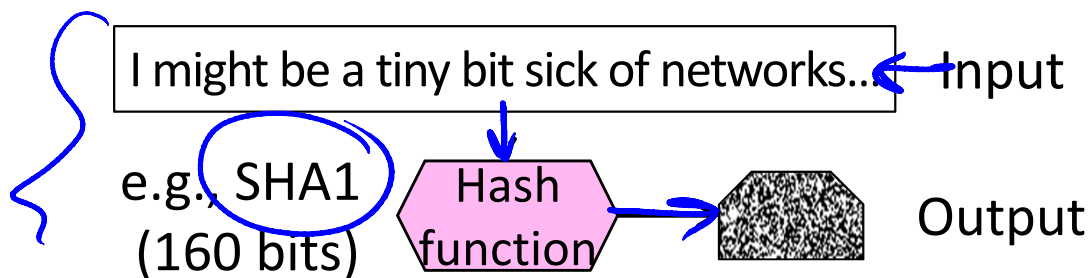


Speeding up Signatures

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

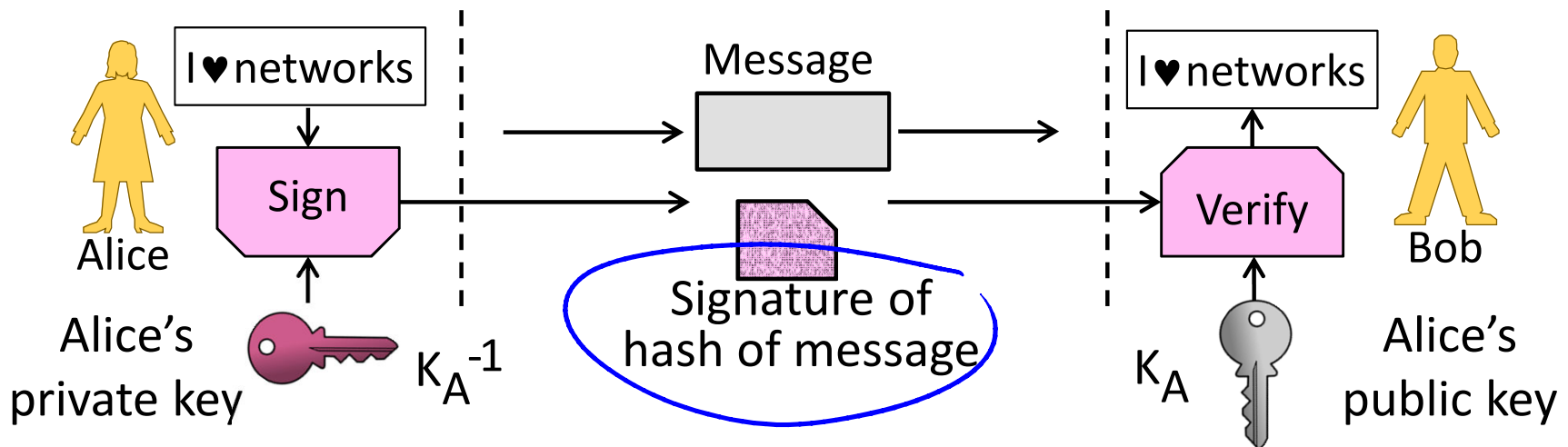
Message Digest or Cryptographic Hash

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

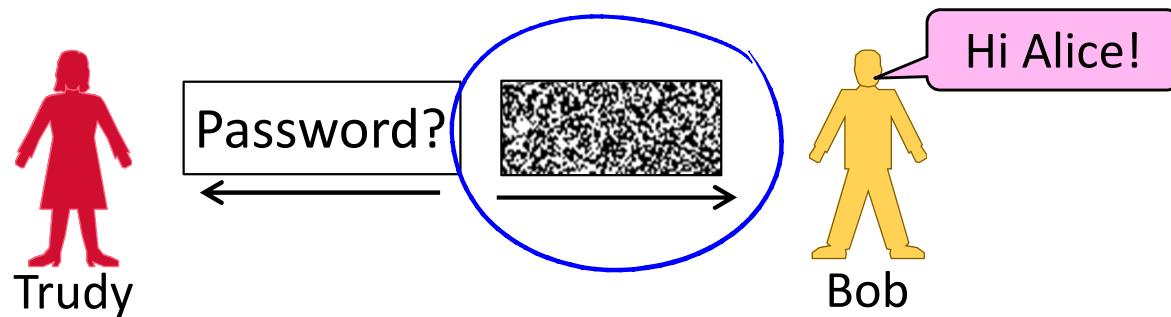


Preventing Replays

- We normally want more than confidentiality, integrity, and authenticity for secure messages!
 - Want to be sure message is fresh
- Don't want to mistake old message for a new one – a replay
 - Acting on it again may cause trouble

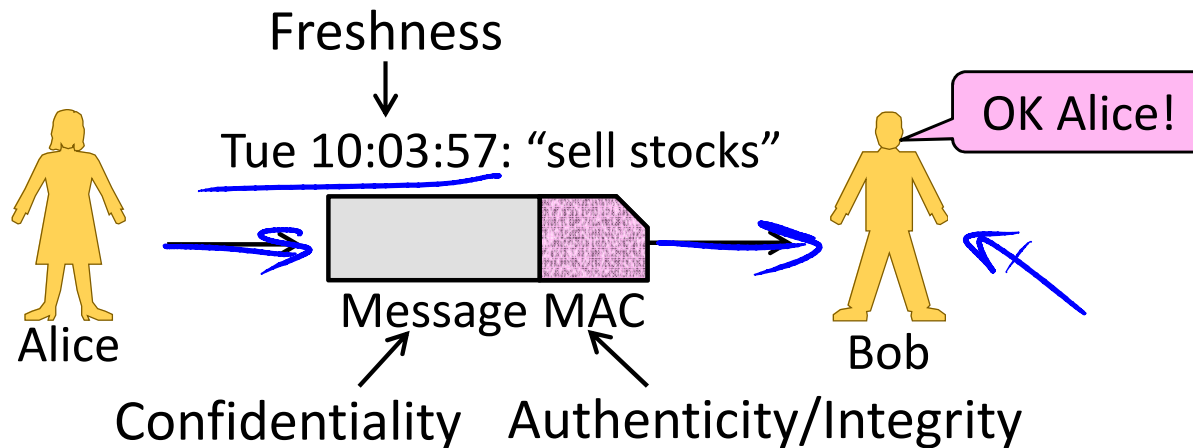
Preventing Replays (2)

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



Preventing Replays (3)

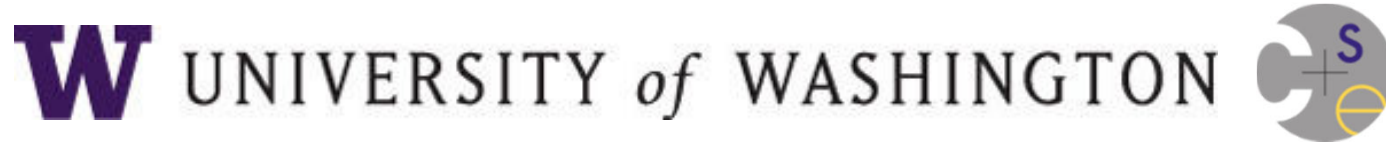
- To prevent replays, include proof of freshness in messages
 - Use a timestamp, or nonce



Takeaway

- Cryptographic designs can give us integrity, authenticity and freshness as well as confidentiality. Yay!
- 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

END



© 2013 D. Wetherall

Slide material from: TANENBAUM, ANDREW S.; WETHERALL, DAVID J., COMPUTER NETWORKS, 5th Edition, © 2011.
Electronically reproduced by permission of Pearson Education, Inc., Upper Saddle River, New Jersey