CS 330: Network Applications & Protocols

Network Security

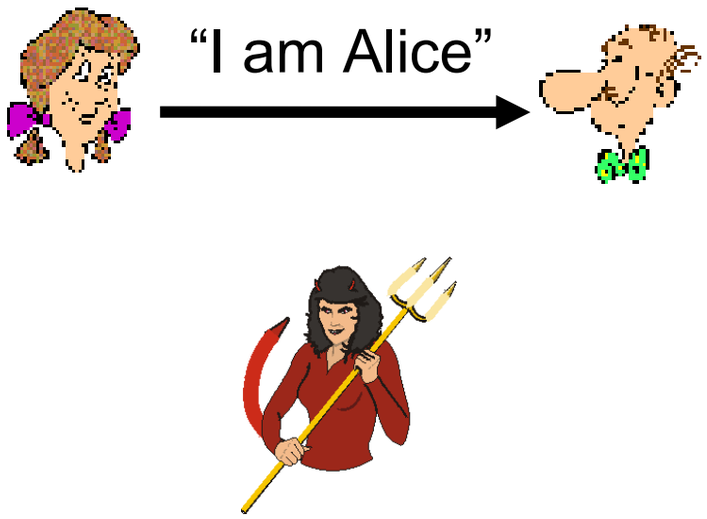
Galin Zhelezov
Department of Physical Sciences
York College of Pennsylvania

Overview of Network Security

- **What is Network Security?**
- **Principles of Cryptography**
- **Message Integrity, Authentication**
- **Operational Security: Firewalls and IDS**

Authentication

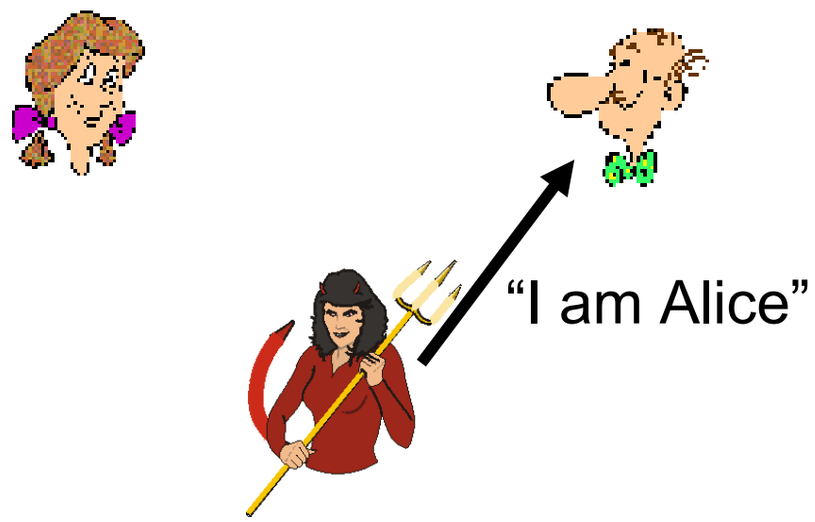
- **Goal: Bob wants Alice to "prove" her identity to him**
- **Authentication Protocol ap1.0: Alice says "I am Alice"**



What is the failure scenario?

Authentication

- **Goal: Bob wants Alice to "prove" her identity to him**
- **Authentication Protocol ap1.0: Alice says "I am Alice"**



In a network, Bob can not "see" Alice, so Trudy simply declares herself to be Alice

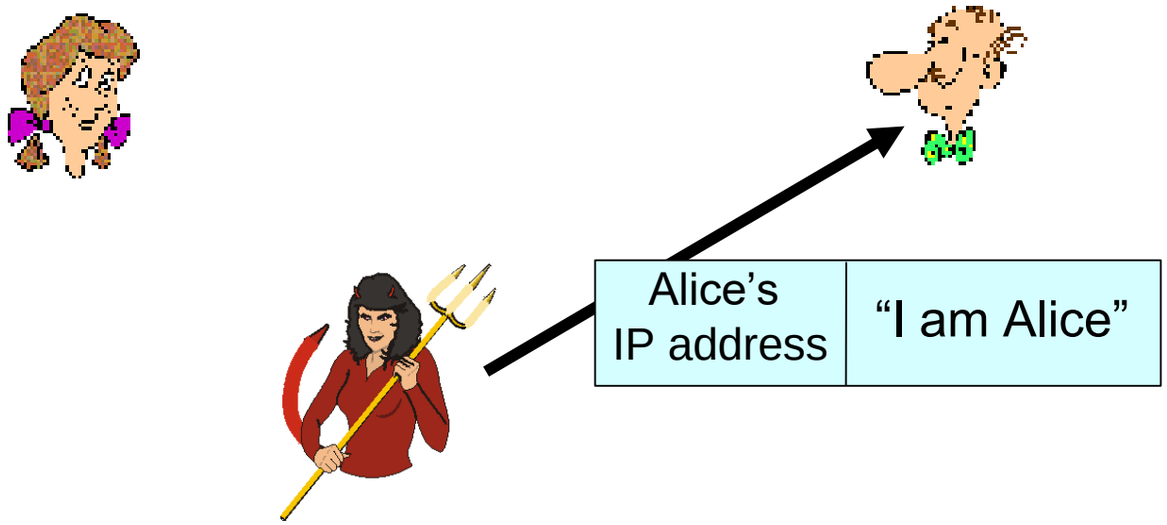
Authentication: Another Try

- **Authentication Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address**
- **Is an IP address enough to authenticate a sender?**

What is the failure scenario?

Authentication: Another Try

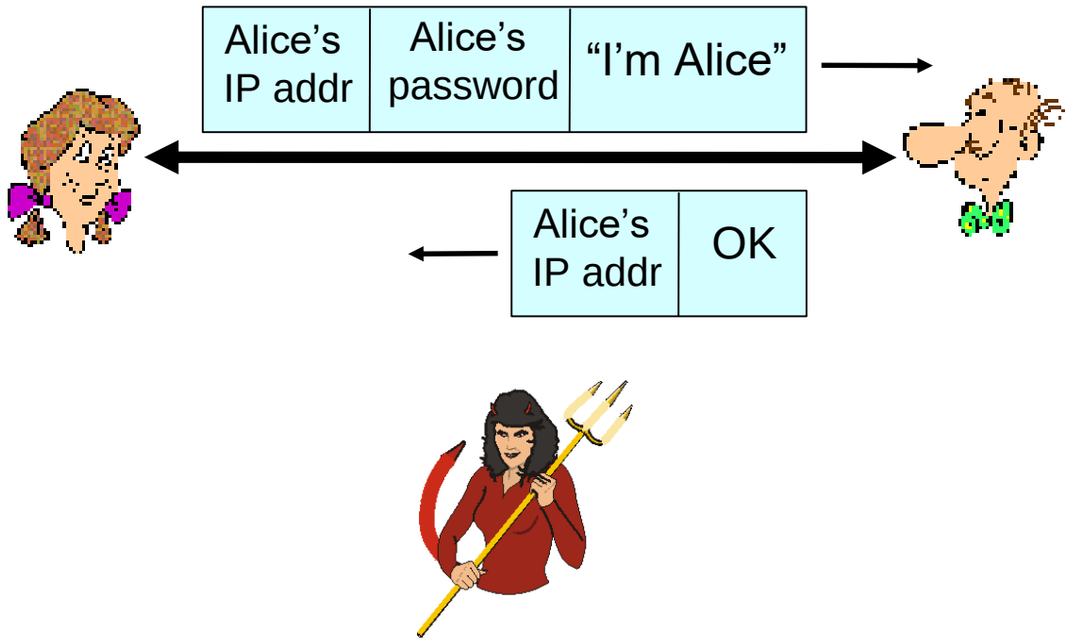
- **Authentication Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address**
- **Is an IP address enough to authenticate a sender? Of course NOT**



Trudy can create a packet "spoofing" Alice's address

Authentication: Yet Another Try

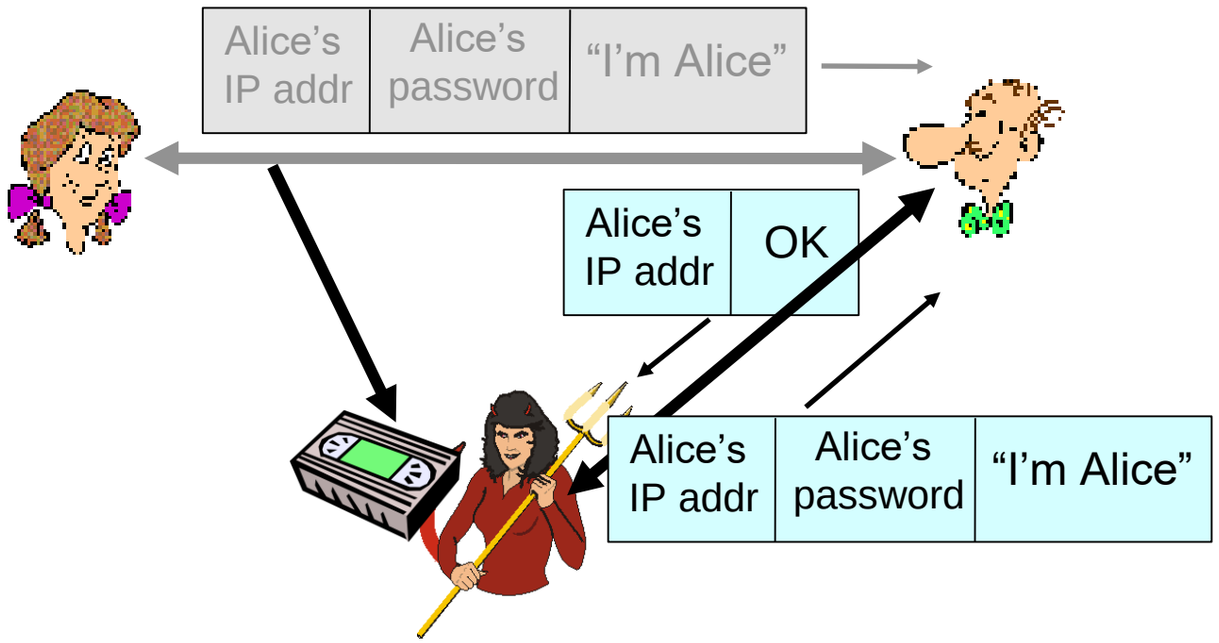
- **Authentication Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.**



What is the failure scenario?

Authentication: Yet Another Try

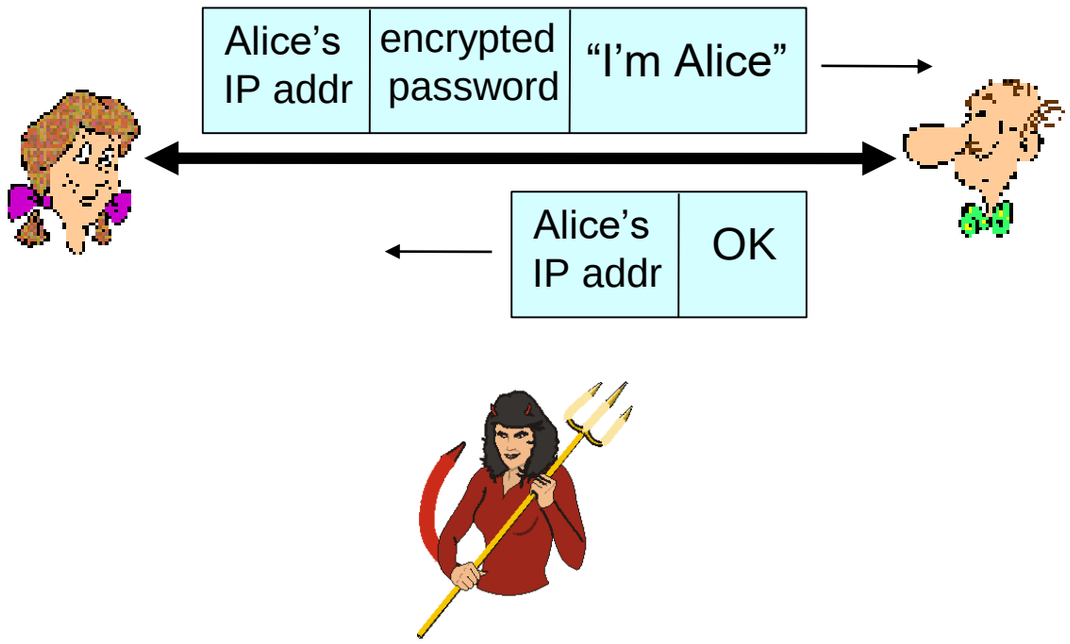
- **Authentication Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.**



Playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: Yet Another Try (again)

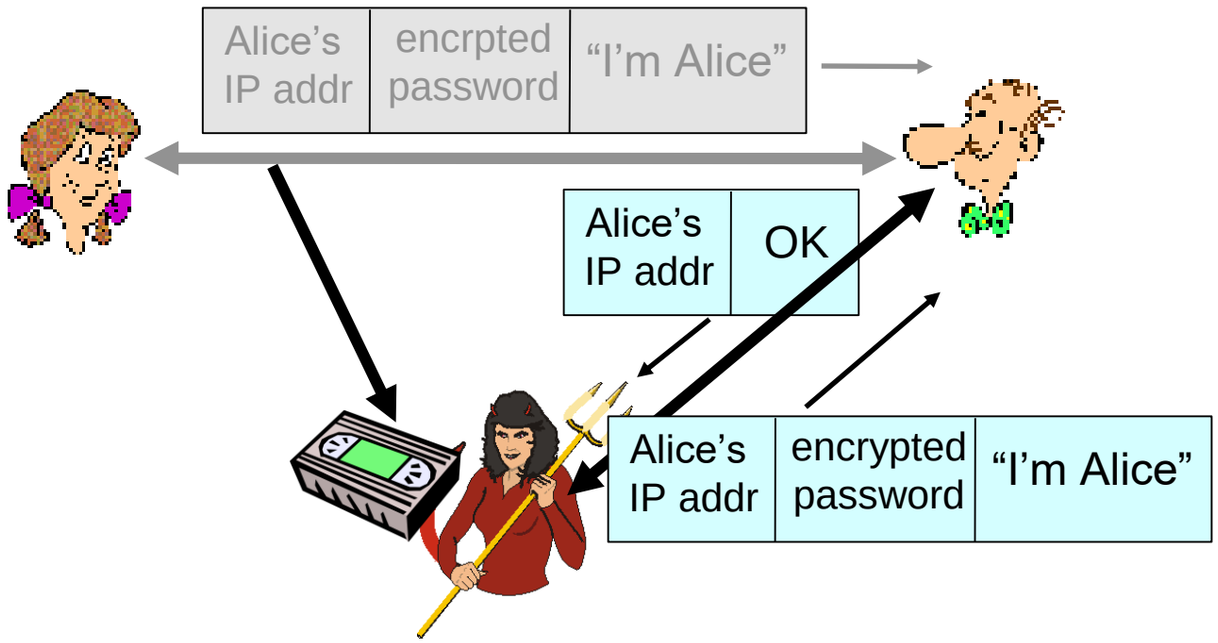
- **Authentication Protocol ap3.1: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.**



What is the failure scenario?

Authentication: Yet Another Try (again)

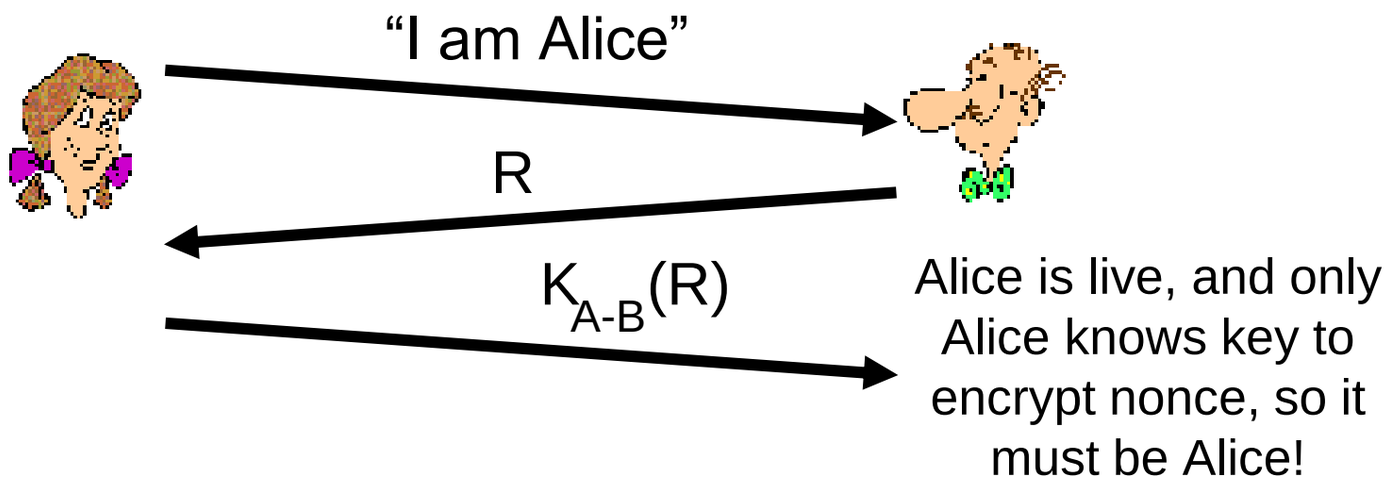
- **Authentication Protocol ap3.1: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.**



Record and playback still works!

Authentication: Still Trying

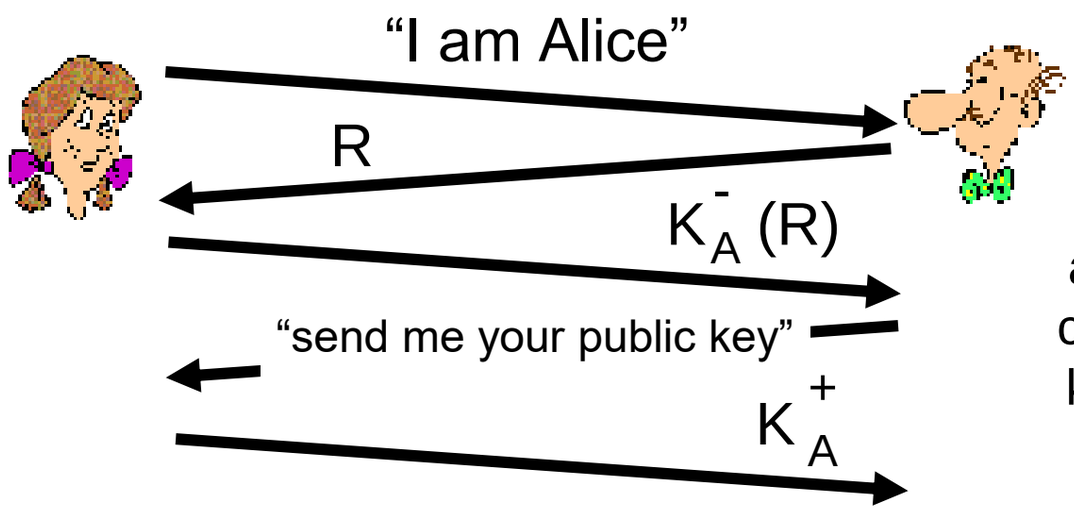
- **Goal: must avoid playback attacks**
- **Utilize a nonce - a number (R) used only once-in-a-lifetime**
- **Authentication Protocol ap4.0: to prove Alice is "live", Bob sends Alice a nonce, R. Alice must return R, encrypted with shared secret key**



Failures, drawbacks?

Authentication: Still Trying, Really

- **Authentication Protocol ap4.0 requires shared symmetric key**
  - Can we authenticate using public key techniques?
- **Authentication Protocol ap5.0: use nonce and public key cryptography**



Bob computes

$K_A^+(K_A^-(R)) = R$

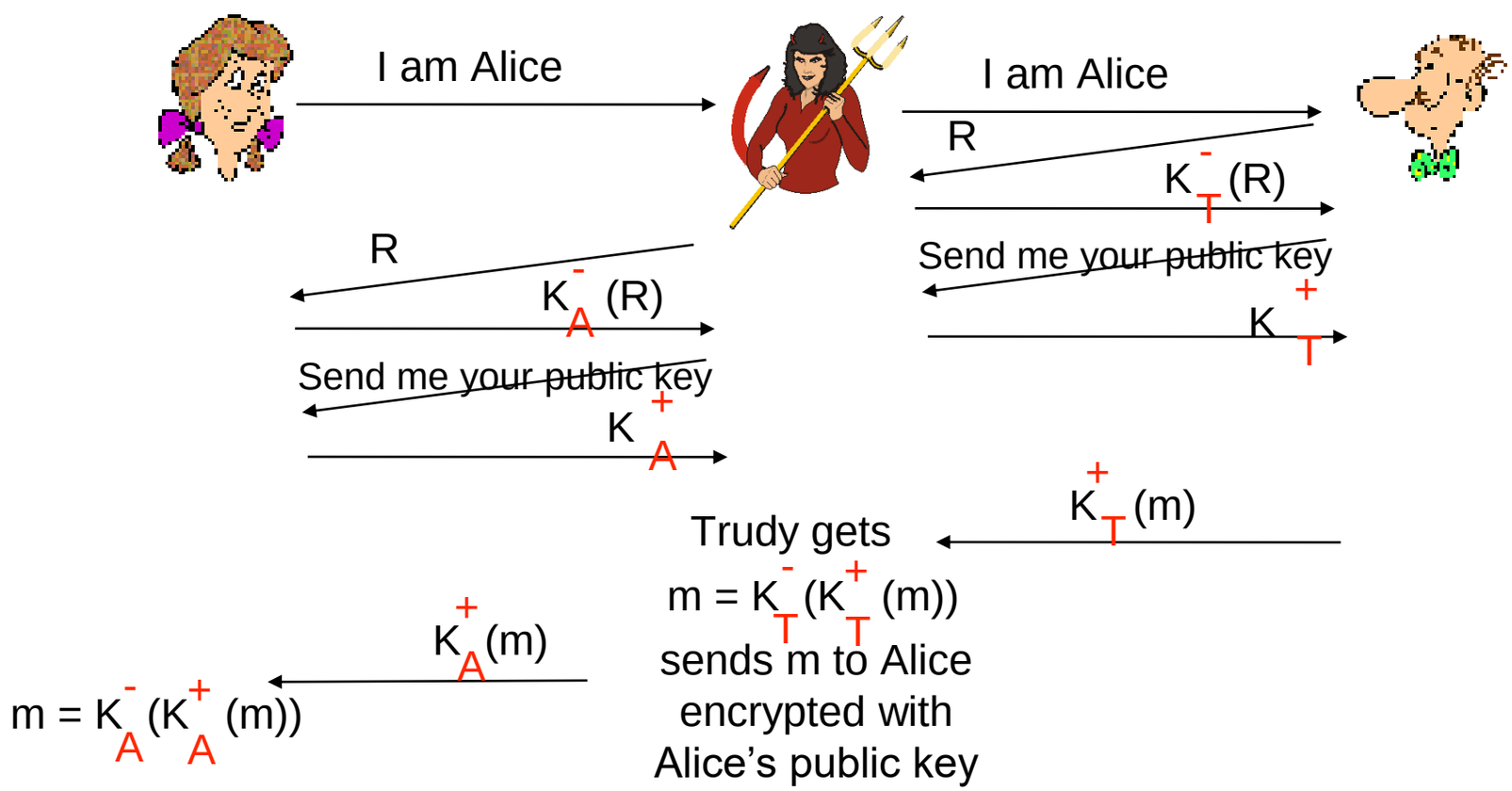
and knows only Alice could have the private key, that encrypted R such that

$K_A^+(K_A^-(R)) = R$

How's this?

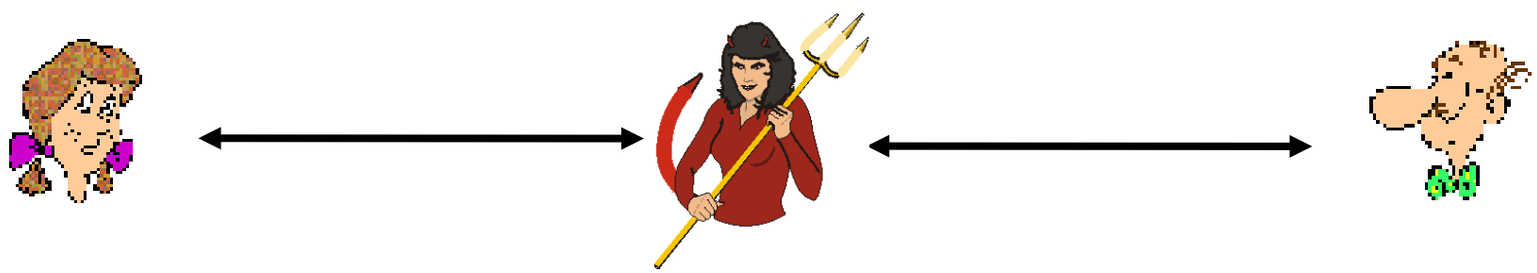
Authentication Protocol ap5.0: Security Hole

- **Man-in-the-middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)**



Authentication Protocol ap5.0: Security Hole

- **Man-in-the-middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)**
  - Difficult to detect:
    - Bob receives everything that Alice sends, and vice versa. (e.g., so Bob, Alice can meet one week later and recall conversation!)
    - Problem is that Trudy receives all messages as well!



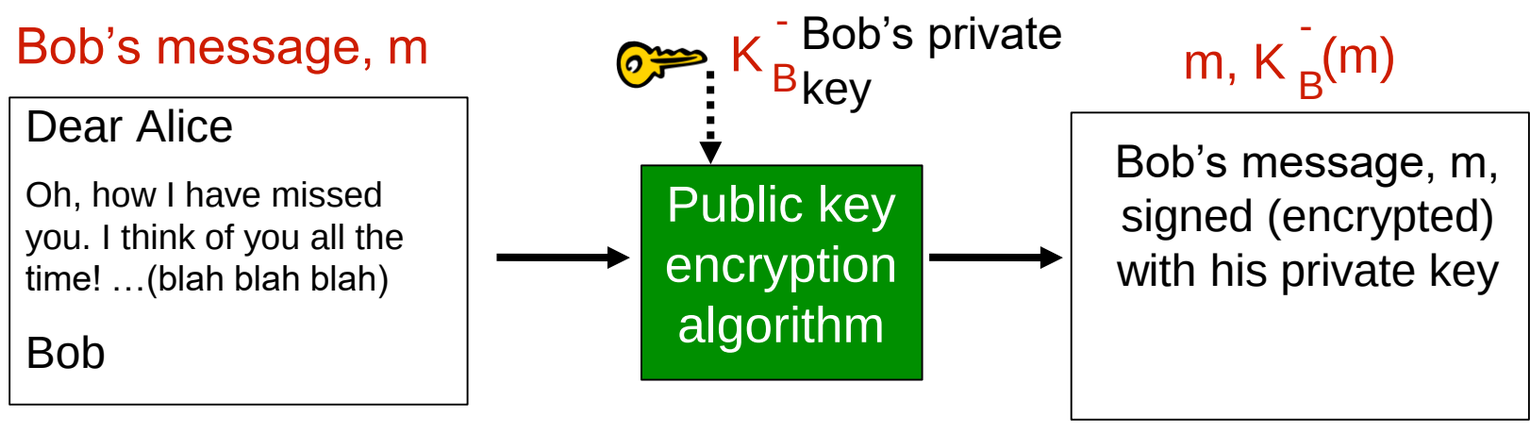
Nobody likes you Trudy

Digital Signatures

- **Cryptographic technique analogous to hand-written signatures**
  - Sender (Bob) digitally signs document, establishing he is document owner/creator
  - Verifiable and non-forgeable
    - Recipient (Alice) can prove to someone that Bob, and no one else (including Alice), must have signed document

Digital Signatures

- **Simple digital signature for message m**
  - Bob signs m by encrypting with his private key K_B^-, creating "signed" message, $K_B^-(m)$



Digital Signatures

- **Suppose Alice receives message m, with signature: m, $K_B^-(m)$**
- **Alice can verify m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$**
- **If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key**

- **Alice can verify that:**
  - Bob signed m
  - No one else signed m
  - Bob signed m and not m' (i.e. m was not altered)
- **Non-repudiation:**
  - Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m

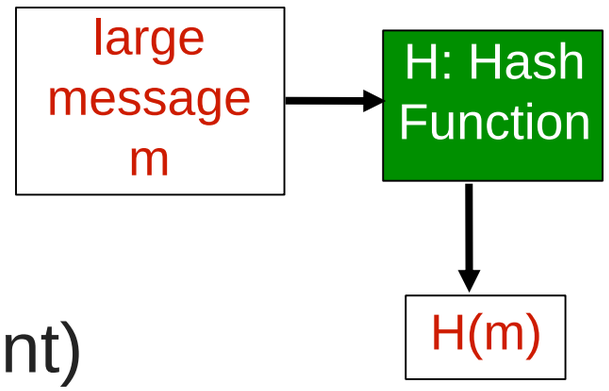
Digital Signatures

- **Digitally signing messages using encryption is computationally expensive**
- **Why not just encrypt a portion of the message to act as a digital signature?**
  - Still need to ensure that content of message hasn't changed
  - Use encrypted message digests as signature

Message Digests

- **Computationally expensive to public-key-encrypt long messages**
- **Goal: fixed-length, easy-to-compute digital "fingerprint"**
  - Apply hash function H to m, get fixed size message digest, H(m)

- **Hash function properties:**
  - Many-to-1
  - Produces fixed-size message digest (fingerprint)
  - Given message digest x, computationally infeasible to find m such that x = H(m)



Internet checksum: poor crypto hash function

Internet checksum has some properties of hash function:

- **produces fixed length digest (16-bit sum) of message**
- **is many-to-one**

But given message with given hash value, it is easy to find another message with same hash value:

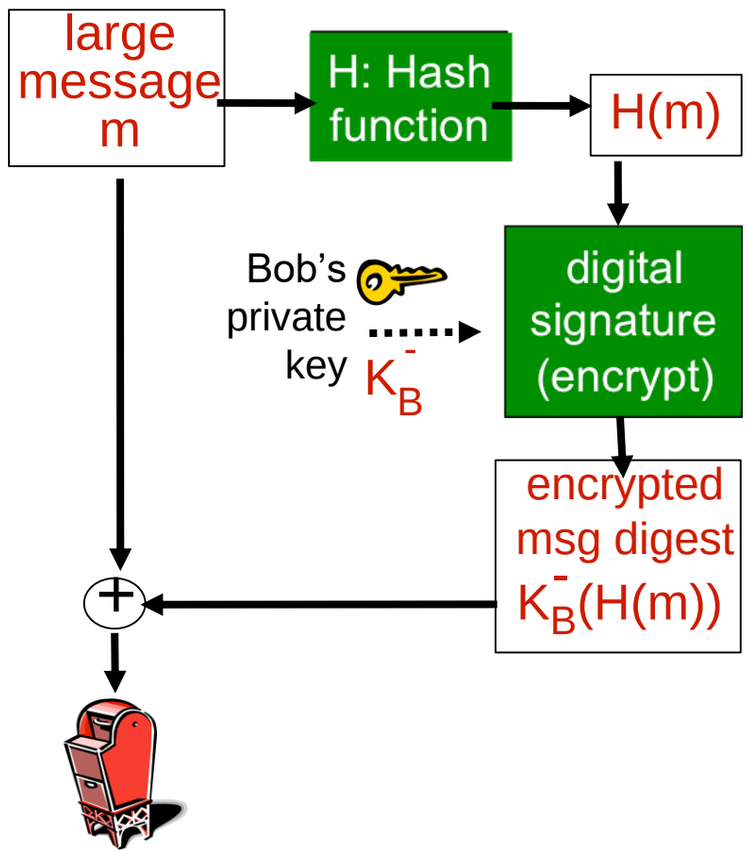
message	ASCII format
I O U 1	49 4F 55 31
0 0 . 9	30 30 2E 39
9 B O B	39 42 D2 42
	B2 C1 D2 AC

message	ASCII format
I O U 9	49 4F 55 39
0 0 . 1	30 30 2E 31
9 B O B	39 42 D2 42
	B2 C1 D2 AC

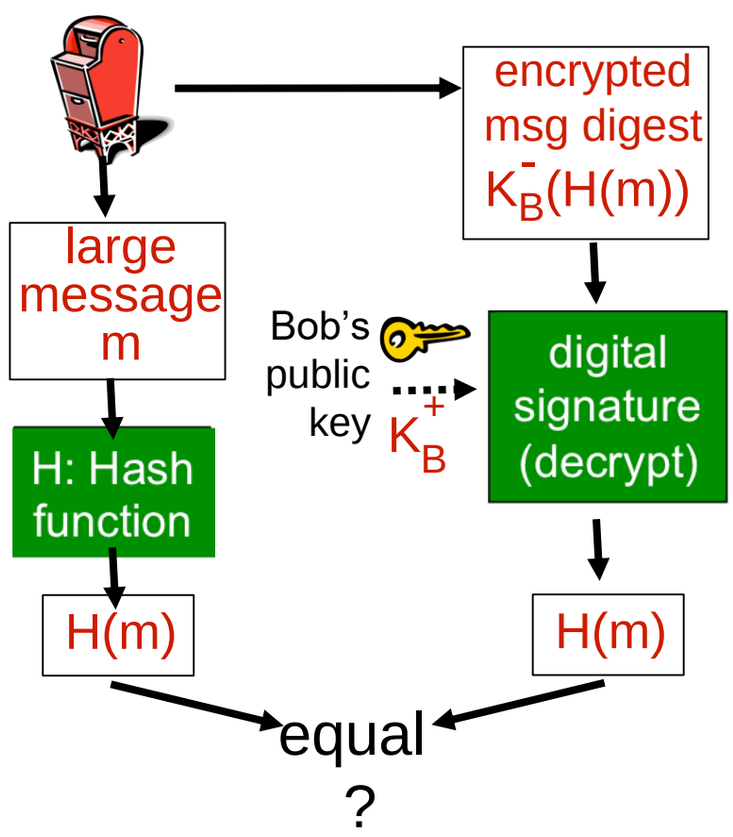
different messages but identical checksums!

Digital Signature = Signed Message Digest

Bob sends digitally signed message



Alice verifies signature, integrity of digitally signed message

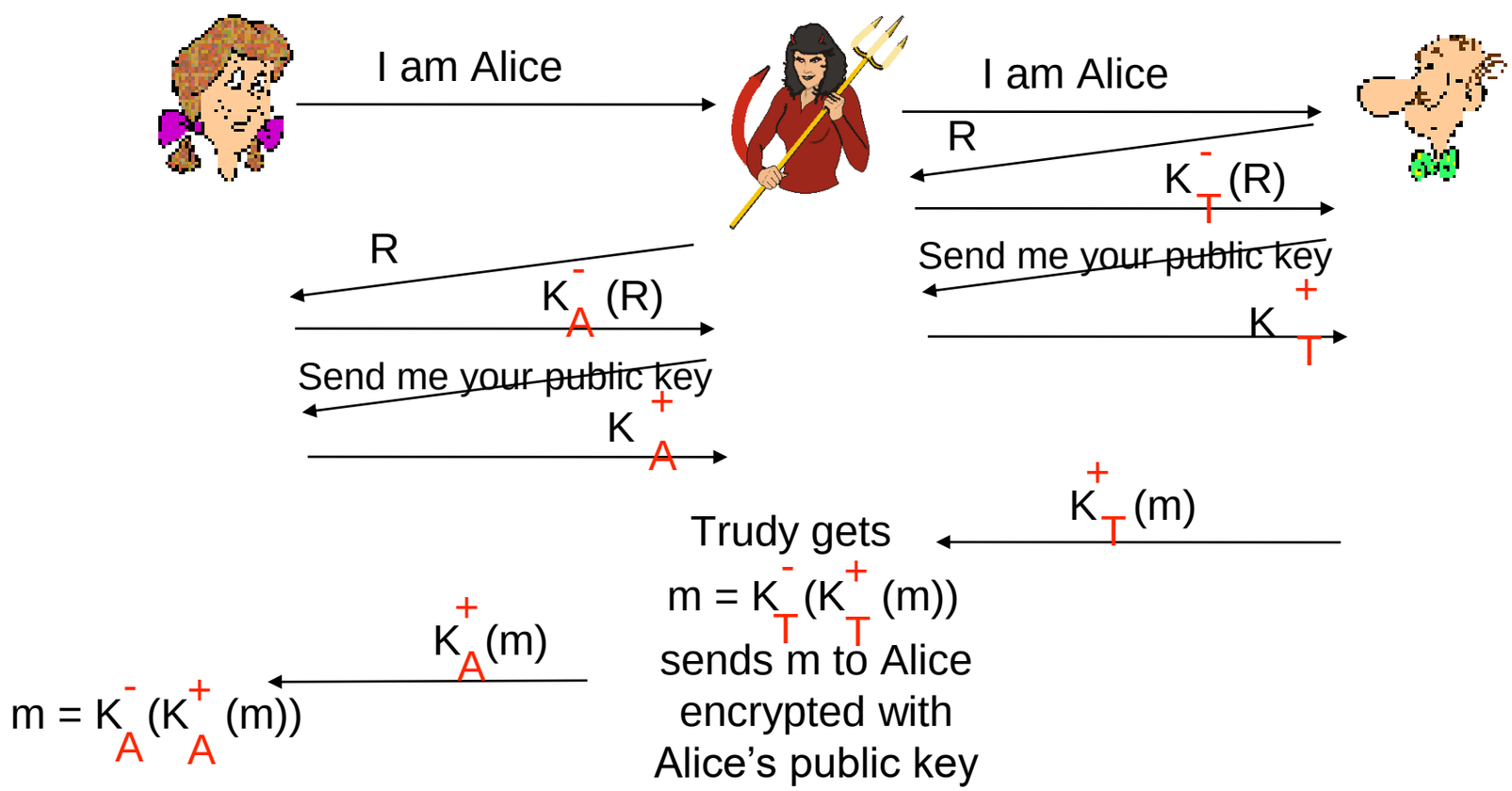


Hash Function Algorithms

- **MD5 hash function widely used**
  - Computes 128-bit message digest in 4-step process.
  - Arbitrary 128-bit string x, appears difficult to construct msg m whose MD5 hash is equal to x
- **SHA-1 is also used**
  - US standard (used by government bodies)
  - 160-bit message digest

Recall: ap5.0 Security Hole

- **Man-in-the-middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)**

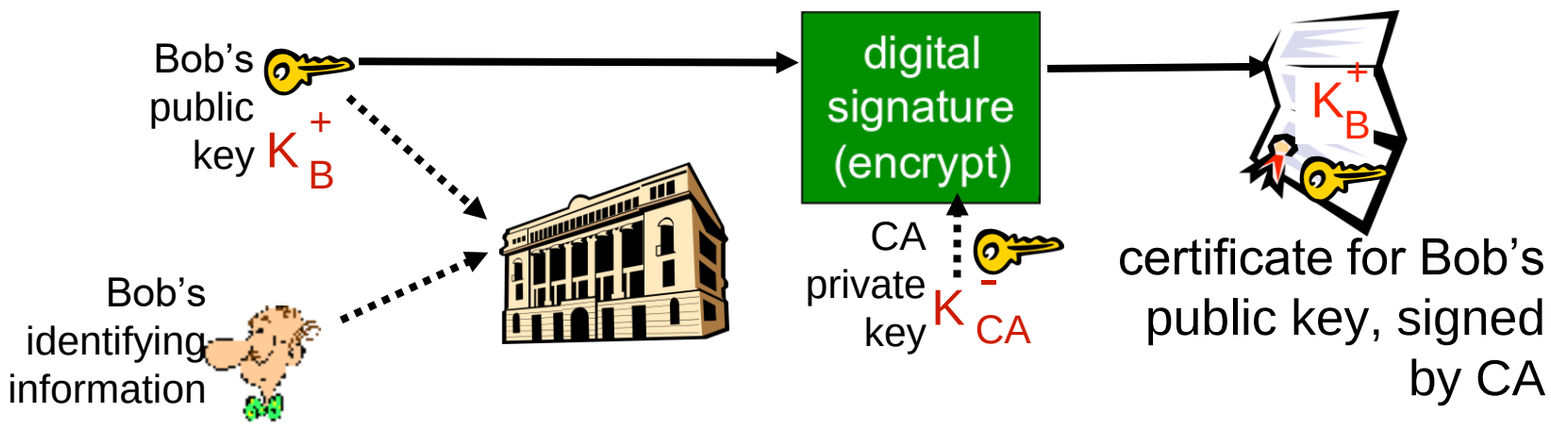


Public-key certification

- **motivation: Trudy plays pizza prank on Bob**
  - Trudy creates e-mail order:
    Dear Pizza Store, Please deliver to me four pepperoni pizzas. Thank you, Bob
  - Trudy signs order with her private key
  - Trudy sends order to Pizza Store
  - Trudy sends to Pizza Store her public key, but says it's Bob's public key
  - Pizza Store verifies signature; then delivers four pepperoni pizzas to Bob
  - Bob doesn't even like pepperoni

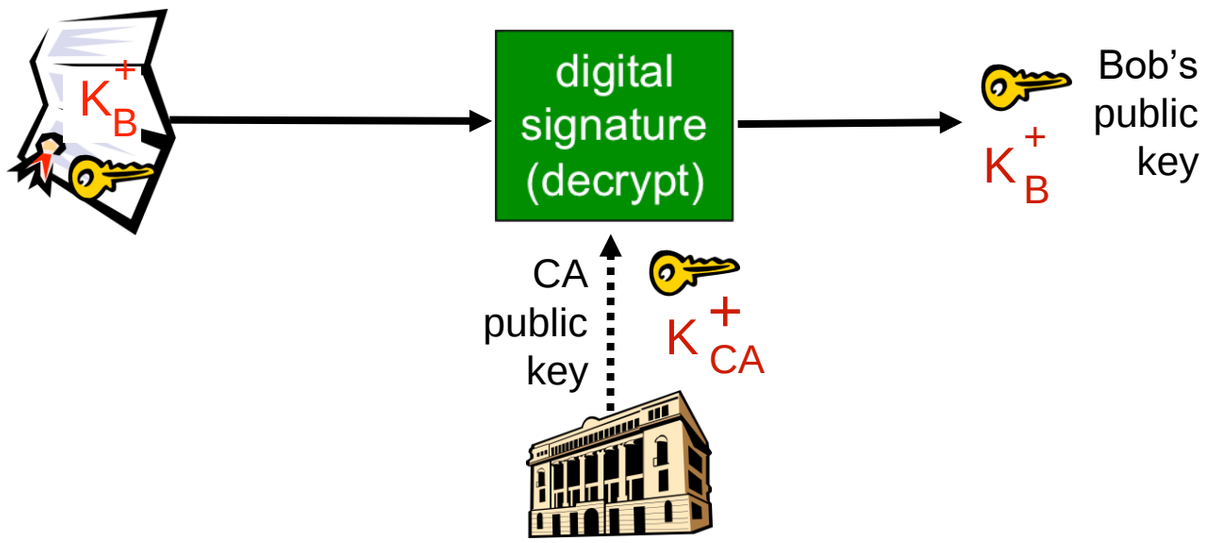
Certification Authorities

- **Certification authority (CA): binds public key to particular entity, E**
- **E (person, router) registers its public key with CA**
  - E provides “proof of identity” to CA
  - CA creates certificate binding E to its public key
  - Certificate containing E’s public key is digitally signed by CA – CA says “this is E’s public key”



Certification Authorities

- **When Alice wants Bob's public key**
  - Gets Bob's certificate (from Bob or elsewhere)
  - Apply CA's public key to Bob's certificate, get Bob's public key



Overview of Network Security

- **What is Network Security?**
- **Principles of Cryptography**
- **Message Integrity, Authentication**
- **Operational Security: Firewalls and IDS**