CSE 433S: Introduction to Computer Security

Authenticated Encryption
Asymmetric Crypto

Washington University in St. Louis

Knowledge Check

- What is MAC, name one property of MAC
- What is Hash function, name the most important function of hash
- What should there be two keys in MAC design
- What was the construction that allows hash function to handle very long messages
- If I have a message that I want to send to the bank, but I don’t care who can read it, what can I do?

Recap: the story so far

Confidentiality: semantic security against a CPA

- Encryption secure against **eavesdropping only**

Integrity:

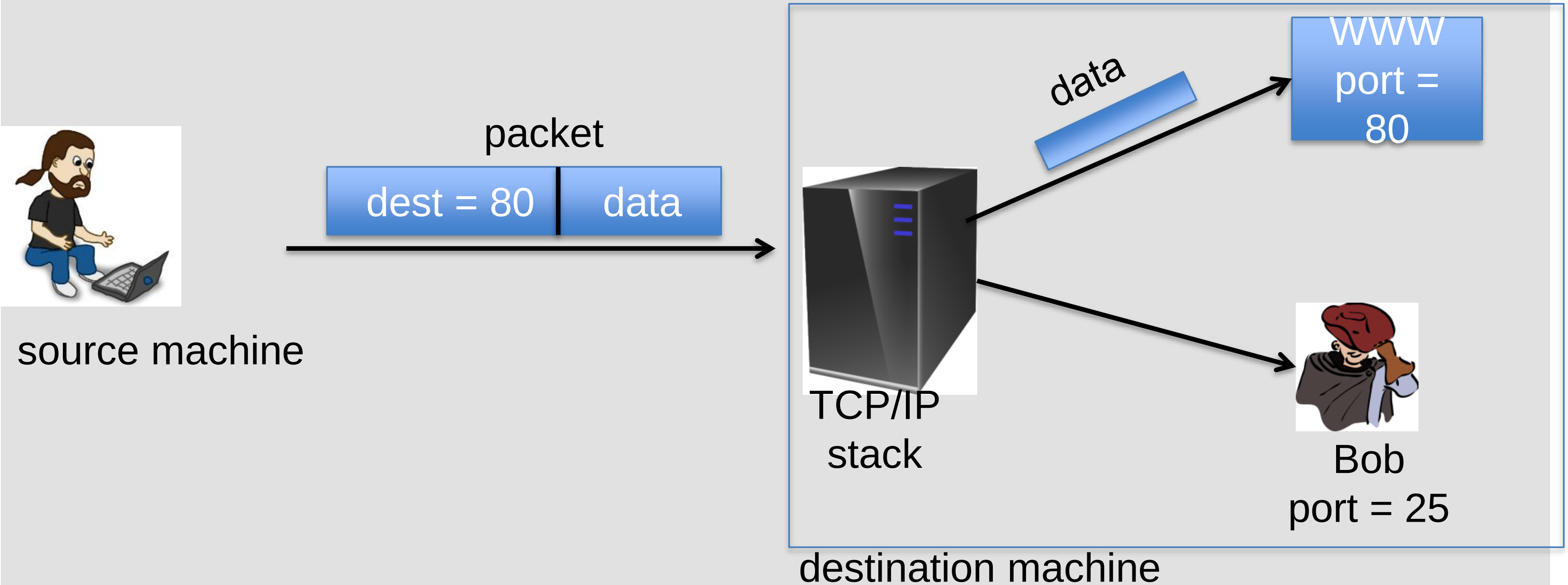
- Existential unforgeability under a CPA
- CBC-MAC, HMAC
- Hash functions

This lecture: encryption secure against tampering

- ***Ensuring both confidentiality and integrity***

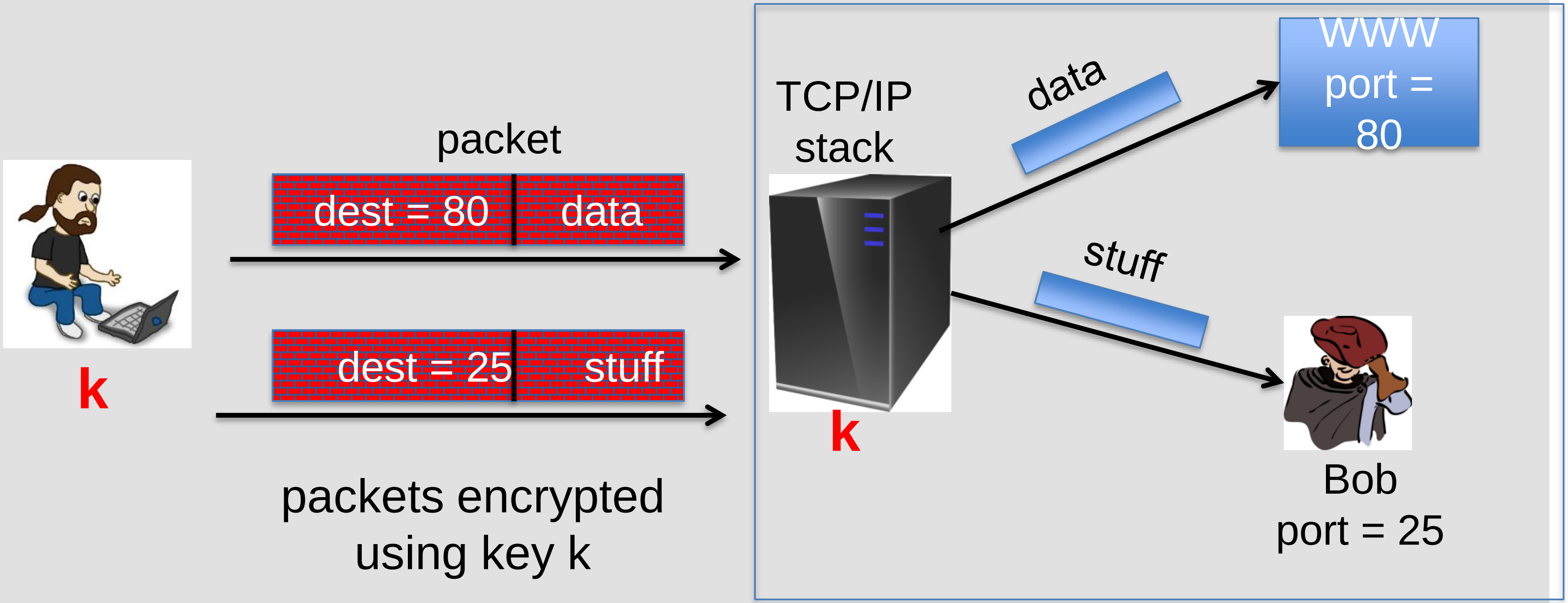
Sample tampering attacks

TCP/IP: (highly abstracted)



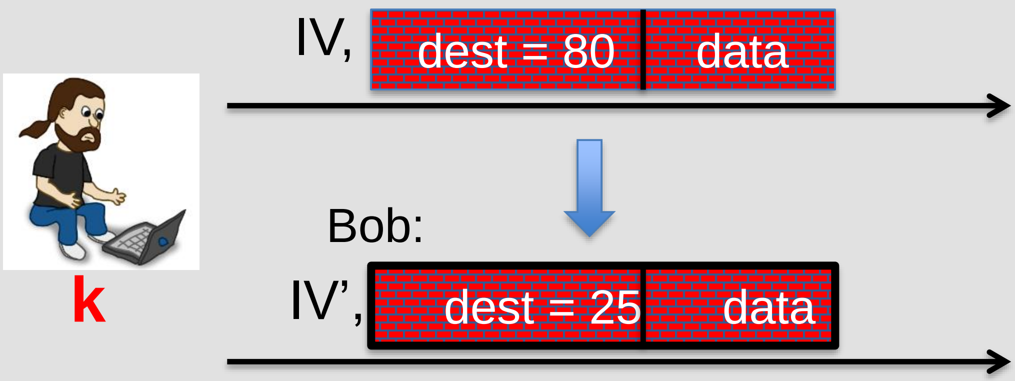
Sample tampering attacks

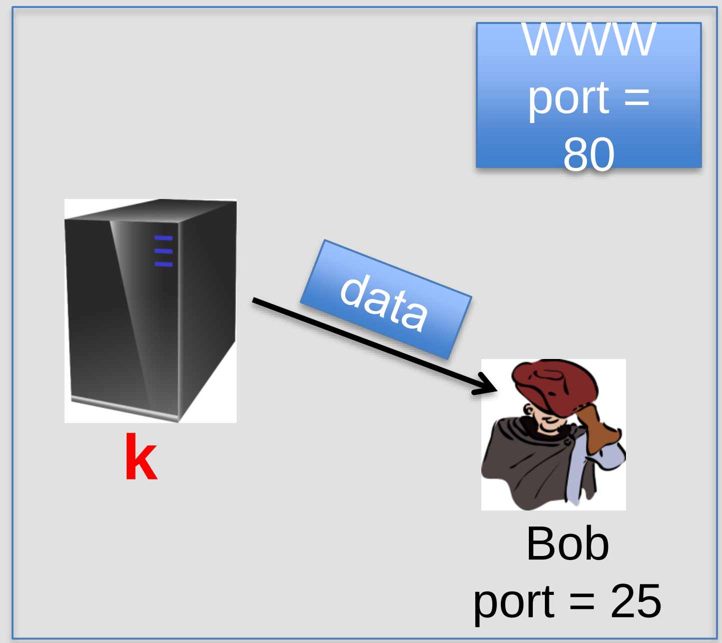
IPsec: (highly abstracted)



Reading someone else's data

Note: attacker obtains decryption of any ciphertext beginning with "dest=25"

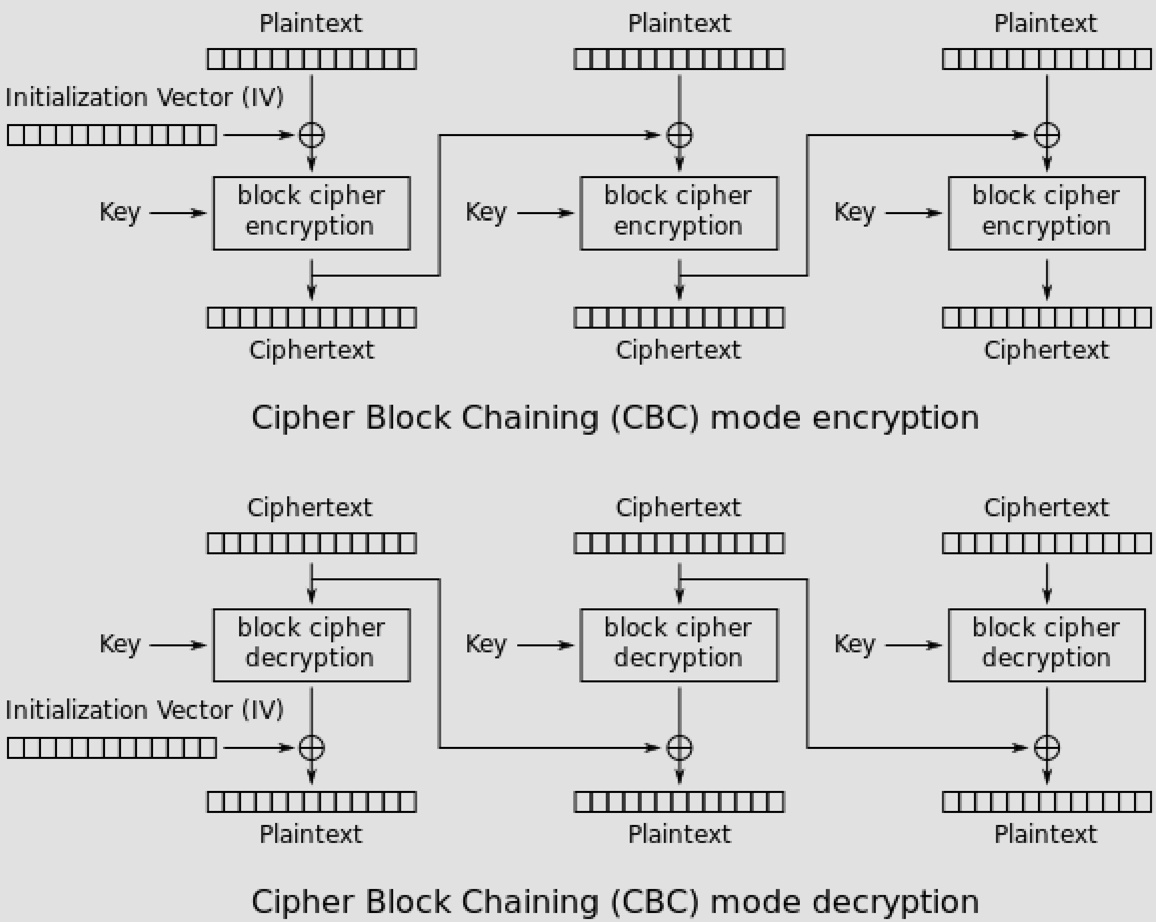


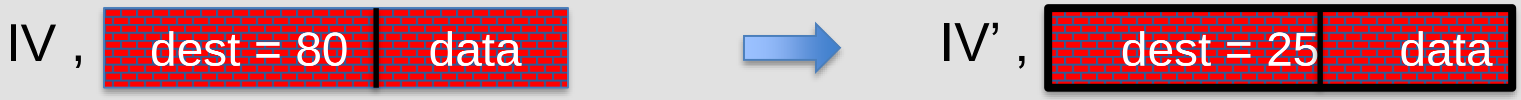


Easy to do for CBC with rand. IV

(only IV is changed)

CBC Mode


Plaintext
Plaintext
Plaintext
Initialization Vector (IV)
Key
block cipher encryption
Key
block cipher encryption
Key
block cipher encryption
Ciphertext
Ciphertext
Ciphertext
Cipher Block Chaining (CBC) mode encryption
Ciphertext
Ciphertext
Ciphertext
Key
block cipher decryption
Key
block cipher decryption
Key
block cipher decryption
Initialization Vector (IV)
Plaintext
Plaintext
Plaintext
Cipher Block Chaining (CBC) mode decryption



Encryption is done with CBC with a random IV.

What should IV' be? $m[0] = D(k, c[0]) \oplus IV$ = "dest=80…"

$IV' = IV \oplus (\ldots 25 \ldots)$

$IV' = IV \oplus (\ldots 80 \ldots)$

$IV' = IV \oplus (\ldots 80 \ldots) \oplus (\ldots 25 \ldots)$

It can't be done

The lesson

CPA security cannot guarantee secrecy under active attacks.

Only use one of two modes:

- If message needs integrity but no confidentiality:

  use a **MAC**

- If message needs both integrity and confidentiality:

  use ***authenticated encryption*** modes

Goals

An **authenticated encryption** system (E,D) is a cipher where

As usual: $E: K \times M \times N \longrightarrow C$

but $D: K \times C \times N \longrightarrow M \cup \{\perp\}$

ciphertext is rejected

Security: the system must provide

- **sem. security** under a CPA attack, and
- **ciphertext integrity:** attacker cannot create new ciphertexts that decrypt properly

Authenticated encryption

Def: cipher (E,D) provides **<u>authenticated encryption</u> (AE)** if it is

(1) semantically secure under CPA, and

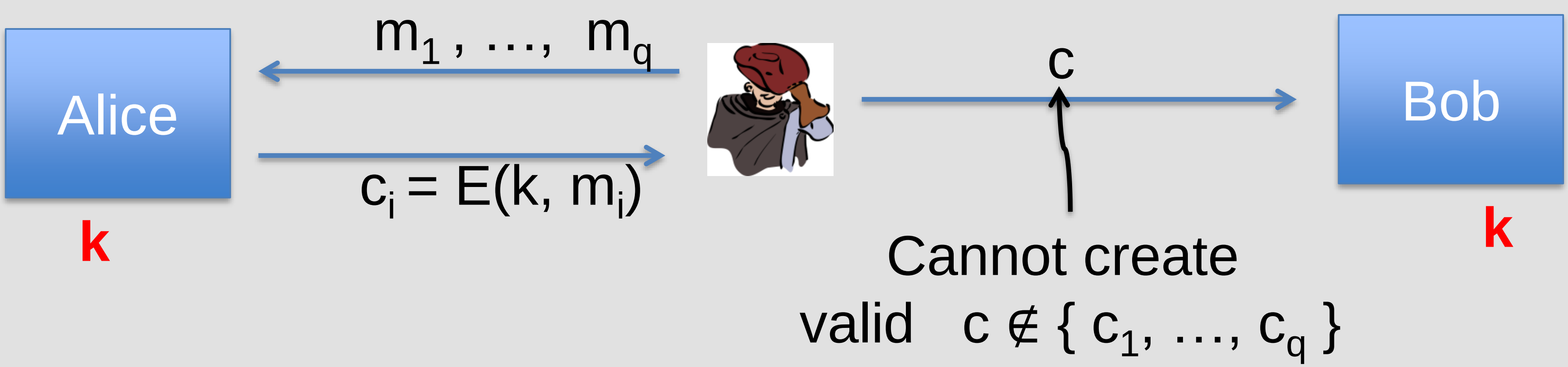
(2) has ciphertext integrity

Bad example: CBC with rand. IV does not provide AE

- $D(k,\cdot)$ never outputs $\perp$, hence adv. easily wins CI game

Implication 1: authenticity

Attacker cannot fool Bob into thinking a message was sent from Alice



$\Rightarrow$ if $D(k,c) \neq \perp$ Bob knows message is from someone who knows k

(but message could be a replay)

Implication 2

Authenticated encryption $\Rightarrow$

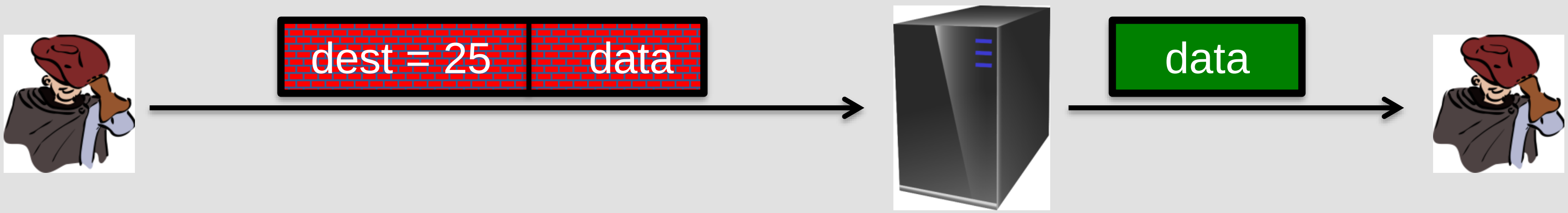
Security against

chosen ciphertext attacks (CCA)

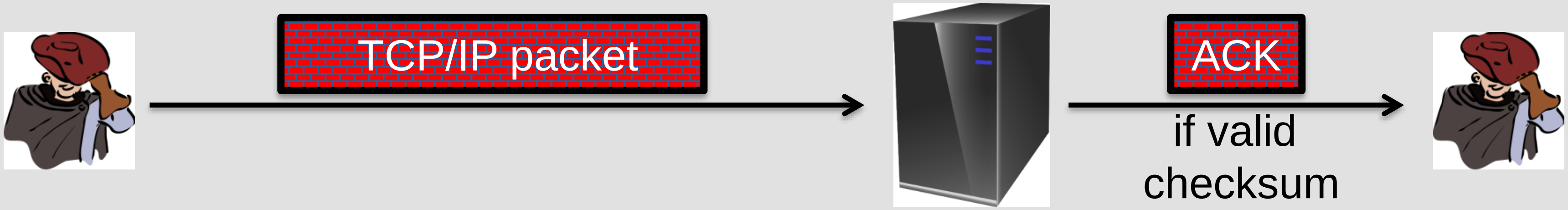
Example chosen ciphertext attacks

Adversary has ciphertext c that it wants to decrypt

- Often, adv. can fool server into decrypting **certain** ciphertexts (not c)



- Often, adversary can learn partial information about plaintext



Chosen ciphertext security

Adversary's power: both CPA and CCA

- Can obtain the encryption of arbitrary messages of his choice
- Can decrypt any ciphertext of his choice, other than challenge

(conservative modeling of real life)

Adversary's goal: Break sematic security

Authenticated enc. ⇒ CCA security

Thm: Let (E,D) be a cipher that provides AE.
Then (E,D) is CCA secure !

In particular, for any q-query eff. A there exist eff. B_1, B_2 s.t.

$$\mathrm{Adv}_{\mathrm{CCA}}[A,E] \le 2q \cdot \mathrm{Adv}_{\mathrm{CI}}[B_1,E] + \mathrm{Adv}_{\mathrm{CPA}}[B_2,E]$$

So what?

Authenticated encryption:

- ensures confidentiality against an active adversary that can decrypt some ciphertexts

Limitations:

- does not prevent replay attacks
- does not account for side channels (timing)

Combining MAC and ENC (CCA)

Encryption key k_E. MAC key = k_I

Option 1: (SSL)

msg m ⇒ msg m | tag ($S(k_I, m)$) ⇒ $E(k_E, m \| tag)$

Option 2: (IPsec)

always correct

msg m ⇒ $E(k_E, m)$ ⇒ ciphertext | tag ($S(k_I, c)$)

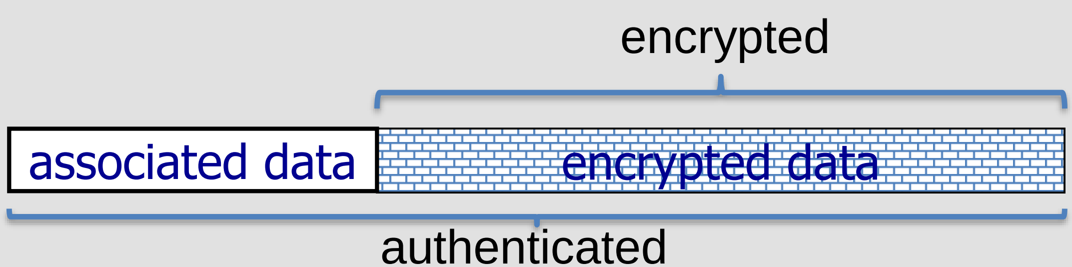
Option 3: (SSH)

msg m ⇒ $E(k_E, m)$ ⇒ ciphertext | tag ($S(k_I, m)$)

Standards (at a high level)

- **GCM**: CTR mode encryption then CW-MAC
  (accelerated via Intel's PCLMULQDQ instruction)
- **CCM**: CBC-MAC then CTR mode encryption
  (802.11i)
- **EAX**: CTR mode encryption then CMAC

All support AEAD: (auth. enc. with associated data).
All are nonce-based.



An example API (OpenSSL)

int **AES_GCM_Init**(AES_GCM_CTX *ain,
    unsigned char ***nonce**,   unsigned long noncelen,
    unsigned char ***key**,   unsigned int klen)

int **AES_GCM_EncryptUpdate**(AES_GCM_CTX *a,
    unsigned char ***aad**,   unsigned long aadlen,
    unsigned char ***data**,   unsigned long datalen,
    unsigned char ***out**,   unsigned long *outlen)

Further reading

- The Order of Encryption and Authentication for Protecting Communications, H. Krawczyk, Crypto 2001.
- Authenticated-Encryption with Associated-Data, P. Rogaway, Proc. of CCS 2002.
- Password Interception in a SSL/TLS Channel, B. Canvel, A. Hiltgen, S. Vaudenay, M. Vuagnoux, Crypto 2003. [padding oracle]
- Plaintext Recovery Attacks Against SSH, M. Albrecht, K. Paterson and G. Watson, IEEE S&P 2009 [ssh attack]
- Problem areas for the IP security protocols, S. Bellovin, Usenix Security 1996.

Summary

Authenticated encryption:

CPA security + ciphertext integrity

- Confidentiality in presence of **active** adversary
- Prevents chosen-ciphertext attacks

Limitation: cannot help bad implementations …

Authenticated encryption modes:

- Standards: GCM, CCM, EAX, [OCB]
- General construction: encrypt-then-MAC

CSE 433S: Introduction to Computer Security

Protocol Designs

Washington University in St. Louis

Slides contain content from Professor Dan Boneh at Stanford University

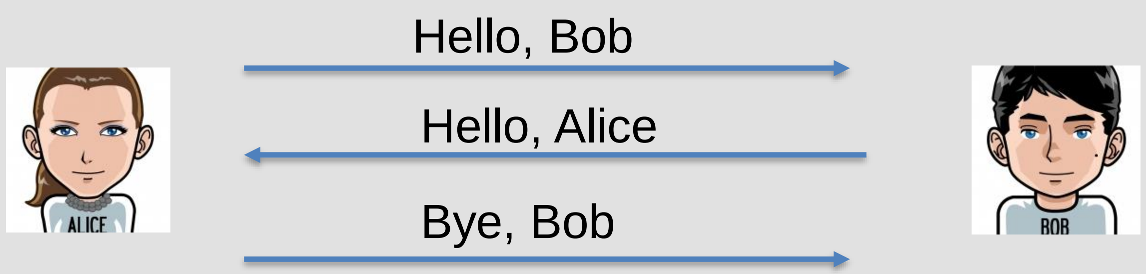
How do you prove your identity to someone over the network ?

Why Security Protocols

- **Alice and Bob want to communicate securely over the Internet, they need to**:
  - (Mutually) authenticate
  - Establish and exchange keys
  - Agree to cryptographic operations and algorithms
- **Building blocks**:
  - Public-key (asymmetric) and secret-key (symmetric) algorithms, hash functions

Network Security Protocol

- A protocol is a set of rules for exchanging messages between 2 or more entities
- A protocol has a number of rounds (>1) and a number of messages (>1)



Basic Elements

- A **message** is a unit of information send from one entity to another as part of a protocol
- A **round** is a basic unit of protocol time:
  - Wake up because of:
    - Alarm clock
    - Initial start or
    - Receive message(s) from other(s)
  - Compute something
  - Send message(s) to others
  - Repeat steps 2-3, if needed
  - Wait for message(s) or sleep until alarm clock

- When acting honestly, entities (participants) achieve the stated goal of the protocol, e.g.:
  - A successfully authenticates to B
  - A and B exchange a fresh session key
- Adversary can defeat this goal
  - e.g., by successfully impersonating A in an authentication protocol with B

The Entities (2-party setting)

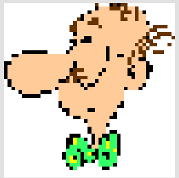
- Alice and Bob – want to mutually authenticate and/or share a key
- • Eve, the adversary – passive or active
- • In more complex protocols, TTP – 3rd party trusted by both Alice and Bob

Challenge-response Authentication

- Goal: Bob wants Alice to “prove” her identity to him

Protocol ap1.0: Alice says “I am Alice”

“I am Alice”

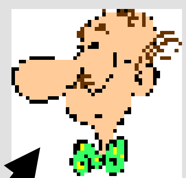


Failure scenario??

Authentication

- Goal: Bob wants Alice to “prove” her identity to him

Protocol ap1.0: Alice says “I am Alice”



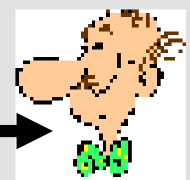
“I am Alice”

in a network,
Bob can not “see”
Alice, so Trudy simply
declares
herself to be Alice

Authentication: another try

<u>Protocol ap2.0:</u> Alice says "I am Alice" in an IP packet containing her source IP address

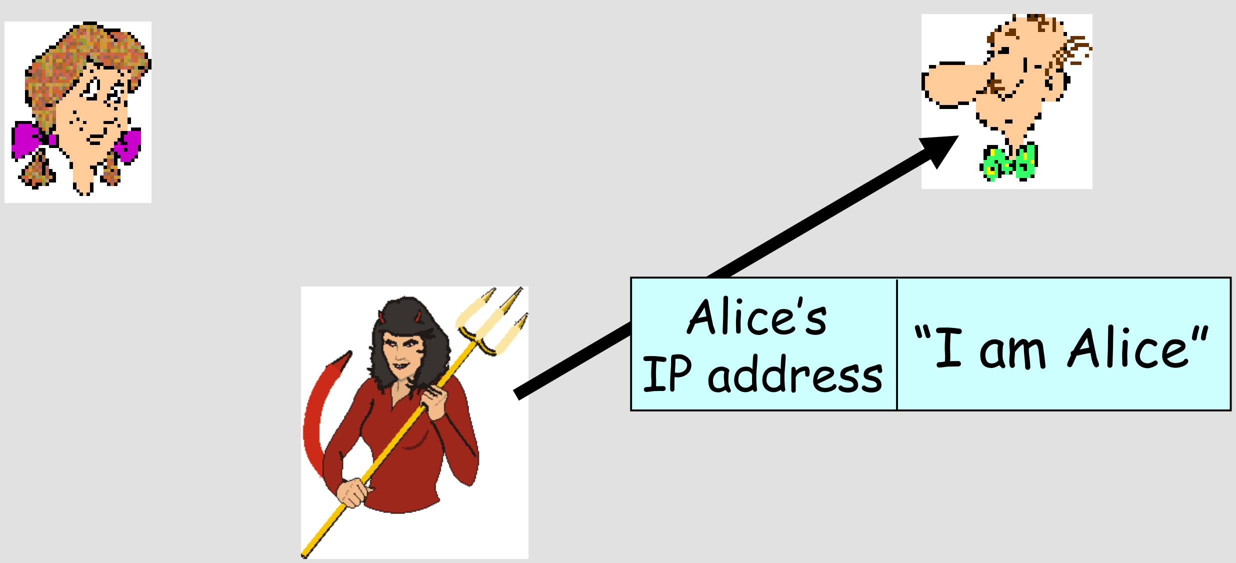
Alice's IP address	"I am Alice"



Failure scenario??

Authentication: another try

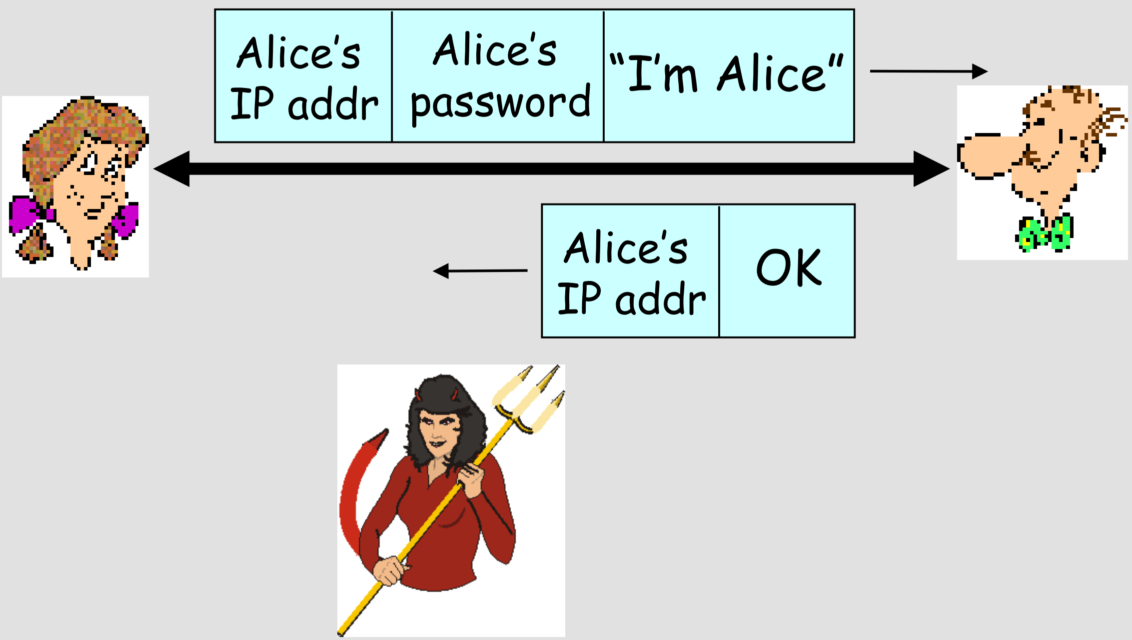
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



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

Authentication: another try

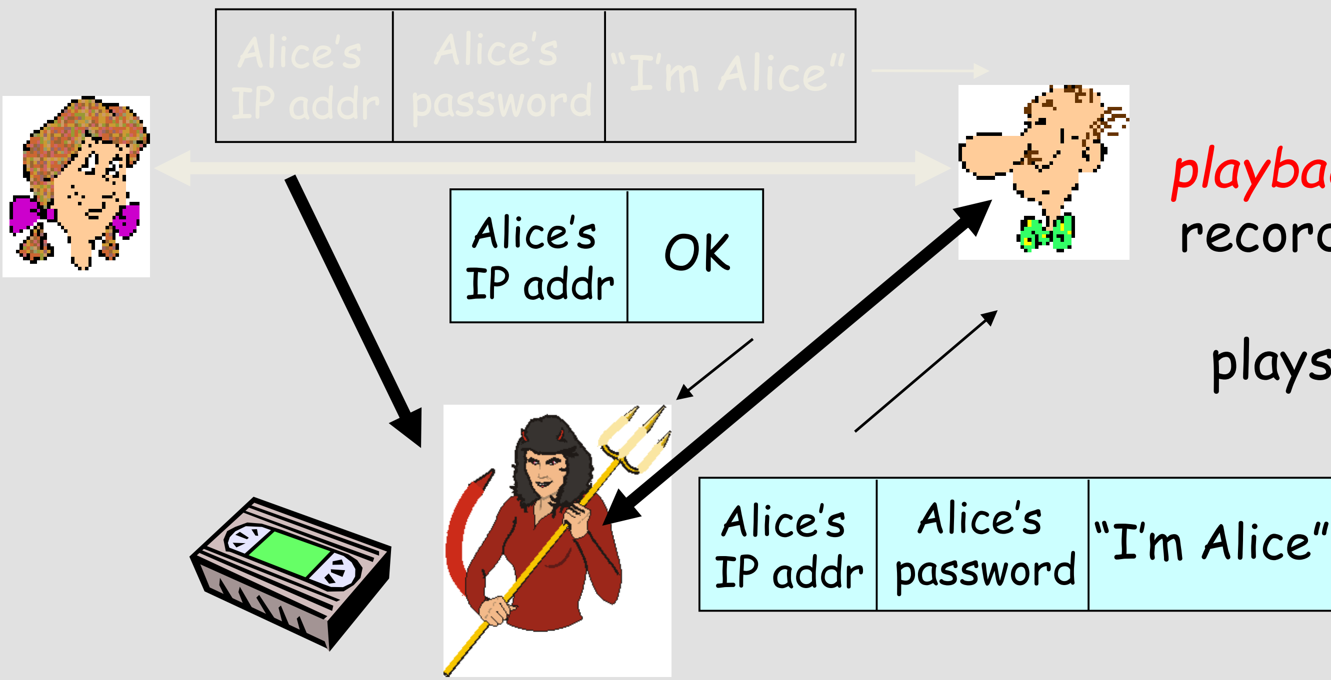
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario??

Authentication: another try

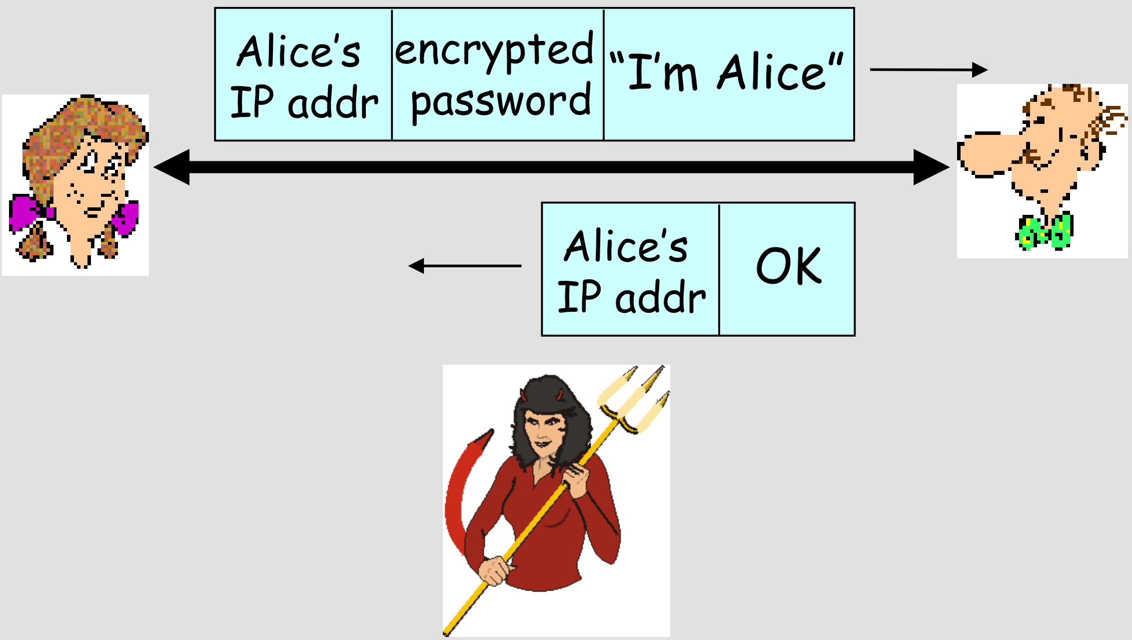
<u>Protocol ap3.0:</u> Alice says "I am Alice" and sends her secret password to "prove" it.



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

Authentication: yet another try

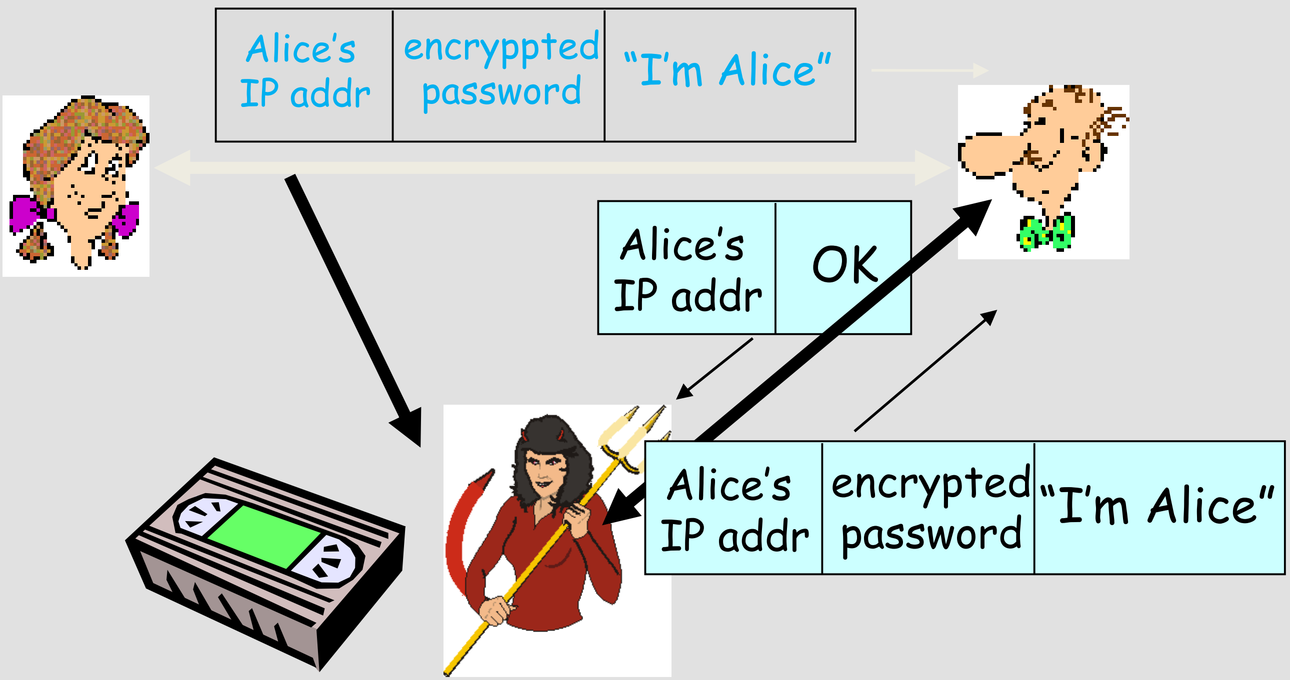
Protocol ap3.1: Alice says “I am Alice” and sends her *encrypted* secret password to “prove” it.



Failure scenario??

Authentication: another try

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



record and playback still works!

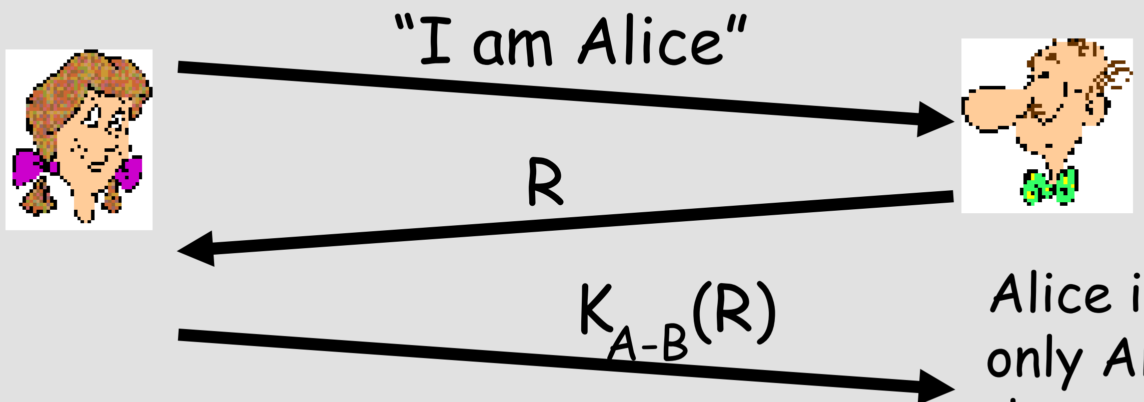
Authentication: yet another try

Challenge and response

Goal: avoid playback attack

Nonce: number (R) used only *once –in-a-lifetime*

ap4.0: to prove Alice "live", Bob sends Alice nonce, R. Alice must return R, encrypted with shared secret key



Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

Failures, drawbacks?

In principle

- *Random numbers*:
  - pseudo-random numbers that are unpredictable to an adversary;
  - need strong pseudo-random strings;
  - must maintain state;
- *Sequences*:
  - serial number or counters;
  - long-term state information must be maintained by both parties+ synchronization;
- *Timestamp*:
  - provides timeliness and detects forced delays;
  - requires synchronized clocks.