

Lecture 12

Identification & Authentication

1

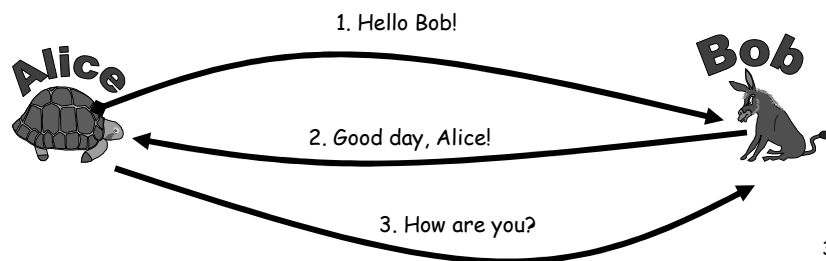
Where are we now?

- We know the following:
 - Conventional cryptography
 - Hash functions and MACs
 - Public key cryptography
 - Encryption
 - Signatures
 - Identification (Fiat-Shamir), Zero Knowledge
- And now what?
 - Protocols
 - Authentication/Identification
 - Key distribution

2

Secure Protocols

- 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)



Secure Protocols

- 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:
 1. *Wake up because of:*
 - a) *Alarm (clock)*
 - b) *Initial start or*
 - c) *Receive message(s) from other(s)*
 2. *Compute something*
 3. *Send message(s) to others*
 4. *Repeat steps 2-3, if needed*
 5. *Wait for message(s) or clock*

4

What's a *secure* protocol?

- When acting honestly, *entities* (participants) achieve the stated **goal** of the protocol (e.g., A successfully authenticates to B, or A and B exchange a fresh session key).
- Neither passive nor active adversary can defeat this objective (e.g., by successfully impersonating A in an authentication protocol with B).

5

The Entities

- **Alice** and **Bob** who wish to authenticate each other and/or to share a key
- **Eve**, the adversary
 - passive or active
- In more complex protocols, **TTP**, a 3rd party trusted by both Alice and Bob

6

Definitions

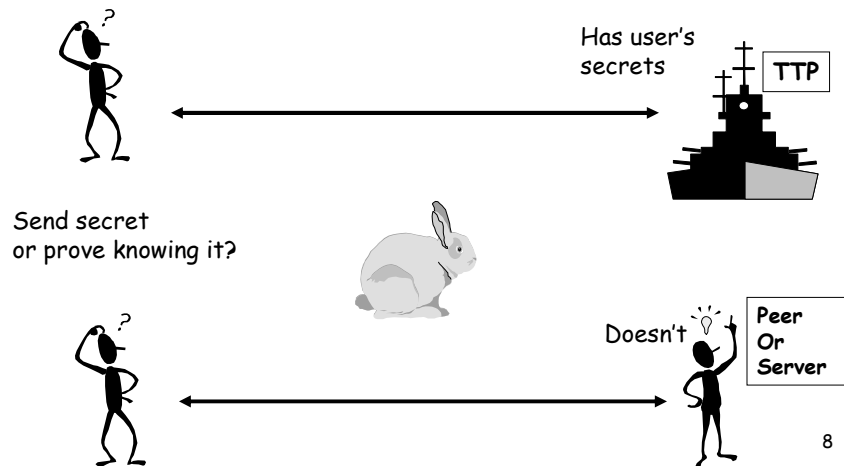
- **Entity authentication:**
 - corroboration that an entity is the one claimed.
- **Unilateral authentication:**
 - entity authentication: providing one entity with assurance of the other's identity but not vice versa.
- **Mutual authentication:**
 - entity authentication which provides both entities with assurance of each other's identity.

7

Purpose

Examples:

- Bank transactions, e.g., cash withdrawals
- Remote login
- File access
- P2P transaction



8

Basis for Authentication

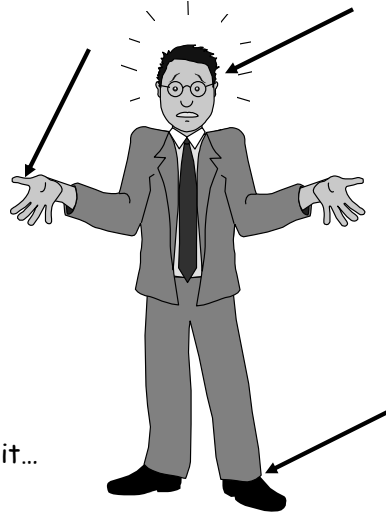
- Something you **know** (a PIN, or password).
- Something you **have**:
 - A secure token, e.g., generating a one-time password.
 - key embedded in a `secure area' on host machine, in browser software, etc.
 - a smartcard (which may contain keys and can perform cryptographic operations on behalf of a user).
- Something you **are** (a biometric).

9

Concrete Scenarios

- ❖ PIN-, PW-, Biometric-based schemes
 - ❖ Kerberos (covered later)
 - ❖ SecureID tokens
 - ❖ Iris/retina scanners
 - ❖ Thumbprint & Handprint
 - ❖ Handwriting acceleration & pressure
- ❖ Public Key Identification Schemes:
 - ❖ Fiat-Shamir, Guillou-Quisqater, etc.
- ❖ Authentication protocols
 - ❖ conventional- and public key-based (covered later)

Human Failings



- ❖ Humans are notoriously unreliable
- ❖ Human memory is very volatile storage

What a human can remember:

- ❖ PIN (no more than 6-8 digits)
- ❖ Password (a word or a short phrase)
- ❖ Can a human do single-digit sums? Forget it...

11

Biometrics

- Accuracy:
 - False acceptance rate.
 - False rejection rate.
 - Can adversary select imposters?
 - Identical twins, family members, etc.
- Retinal scanner, fingerprint reader, handprint reader, voiceprint, keystroke timing, signature (shape or pressure), etc.

12

Fingerprints

- Vulnerability:
 - Dummy fingers and dead fingers
- Suitability and stability:
 - Not for people with high probability of damaged fingerprints (e.g., exema)
 - Not for kids who are still growing

13

Voice Recognition

- Single phrase:
 - Can use tape recorder to fake
- Stability:
 - Background noise
 - Colds, vocal cord damage/strain, laughing gas 😊
 - Use with public phones

14

Keystroke Timing

- Each person has a distinct typing timing and style
 - Hand/finger movements
- Suitability:
 - Best done for "local" authentication
 - Avoid network traffic delay

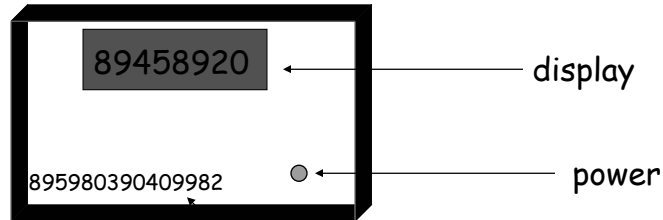
15

(non-digital) Signatures

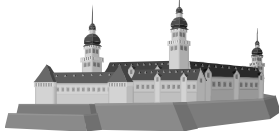
- Machines can't match human experts in recognizing shapes of signatures
- Add information of timing (dynamics) or pressure
 - Signing on an electronic tablet

16

SecureID



TTP/Server
knows all secrets!



Serial #
Id-based key
(inside)

17

Weak Authentication

- Passwords and PINs only provide **weak** authentication
 - User demonstrates knowledge of password matching his user ID, server compares to entry in password file.
 - Static, guessable, sniffable if unprotected.
 - Not suitable for use over open networks.
- For extra security
 - Don't send pw, but rather hash(pw,challenge)
 - Use one-way hashed and encrypted password file and slow down encryption process to make dictionary attack harder.
 - Access control on password file.
 - Lock out after number of failed attempts.
 - One-time passwords

18

Strong Authentication

- In **strong** authentication, one entity `proves' its identity to another by demonstrating knowledge of a secret known to be associated with that entity, *without revealing that secret itself during the protocol*.
- Also called `challenge-response' authentication.
- Uses cryptographic mechanisms to protect messages in protocol:
 - Encryption.
 - Integrity mechanism (MAC, one-way function or cryptographic check function).
 - Digital signatures.

19

Encryption-based Unilateral Authentication

- Assume Alice (client) and Bob (server) share a secret key K , Bob authenticates Alice.
- Alice sends an initiating message.
- Bob sends Alice a *challenge* message R .
- Alice *responds* with $\{R || B\}_K$, message R concatenated with B , encrypted under shared key K .
- Bob checks that he gets R back on decrypting Alice's reply.

20

The Protocol

1. $A \rightarrow B$: "Hi Bob, it's, me, Alice"
2. $B \rightarrow A$: R (challenge)
3. $A \rightarrow B$: $\{R || B\}_K$ (response)

Why not simply send $\{R\}_K$ in the last message?

21

Security of the Protocol

- Eve `sees' R and $\{R || B\}_K$. Because of perfect encryption, learns nothing about K .
- Bob gets his challenge R back again, as a response that only Alice can prepare. This allows him to be sure of origin and integrity of message.
- Eve can impersonate Bob easily: so Bob not authenticated to Alice \rightarrow unilateral (one-way) authentication.

22

Security Against Eve?

- Eve unable to come up with a correct response $\{R \parallel B\}_K$ to Bob's challenge as she doesn't know K .
- Challenge R must be unpredictable: otherwise Eve can masquerade as Alice in a subsequent protocol run, *replaying* Alice's response:

23

Replay Attack

1. $A \rightarrow E$: "Hi Bob, it's me, Alice"
2. $E \rightarrow A$: R
3. $A \rightarrow E$: $\{R \parallel B\}_K$
4. $E \rightarrow B$: "Hi Bob, it's me, Alice"
5. $B \rightarrow E$: R
6. $E \rightarrow B$: $\{R \parallel B\}_K$

24

Freshness and Liveness

- **Origin and integrity** authentication is not enough - also need means of checking *freshness* of messages and *liveness* (timeliness) of principals to counter replays.
- **Freshness**: assurance that message has not been used previously and originated within an acceptably recent timeframe.
- **Liveness**: assurance that message sent by a principal within an acceptably recent timeframe.
- Two main methods for providing freshness:
 - Nonce (Number used ONCE).
 - Time-stamps (clock-based or 'logical' time-stamps).

25

Nonces

- Nonce: one-time random challenge.
- We depended on B to make sure R is a good nonce
- Main property 'one-time-ness', so could even use a counter (but must keep state)
- However, many protocols need nonces to be unpredictable to Eve. Generate at random from a large set (e.g., 2^{128}).

26

Time-stamps

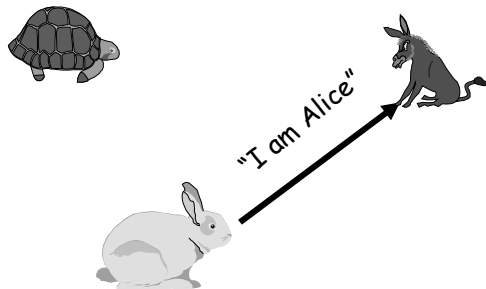
- Inclusion of date/time-stamp in message allows recipient to check it for freshness (as long as time-stamp protected by cryptographic means).
- $A \rightarrow B: \{T \parallel B\}_K$
 - results in fewer messages in protocol
- But requires synchronised clocks which is hard!

27

Authentication

Goal: Bob wants Alice to "prove" her identity to him

Protocol ap1.0: Alice says "I am Alice"



in a network,
Bob can not "see"
Alice, so Eve simply
declares
herself to be Alice

28

Authentication: another try

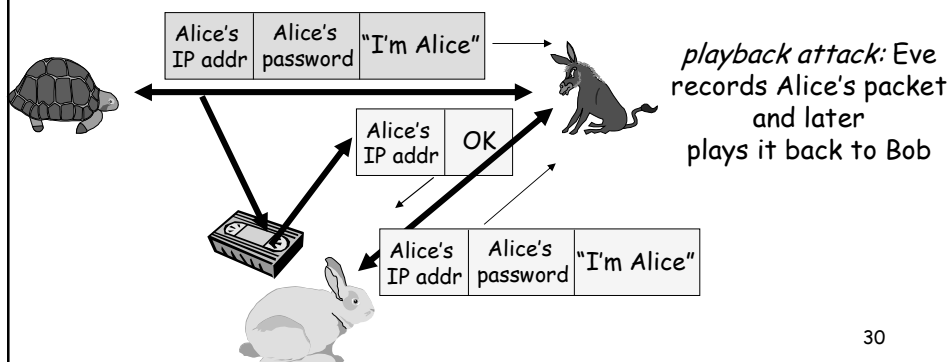
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



29

Authentication: another try

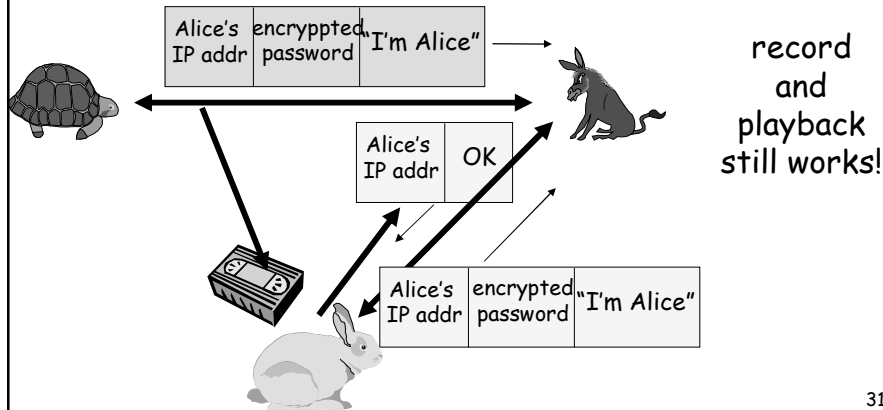
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



30

Authentication: another try

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



31

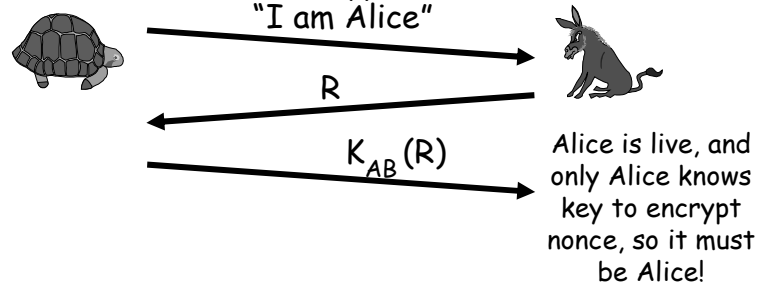
Authentication: yet another try

Goal: avoid playback attack

Nonce: number (R) used only *once*

ap4.0: to prove Alice "live", Bob sends Alice nonce, R.

Alice must return R, encrypted with shared secret key



32

Authentication: ap5.0

ap4.0 requires shared symmetric key

- can we authenticate using public key?

ap5.0: use nonces and public key cryptography

