

# Cryptography CS 555

## Lecture 16



Department of Computer Sciences  
Purdue University

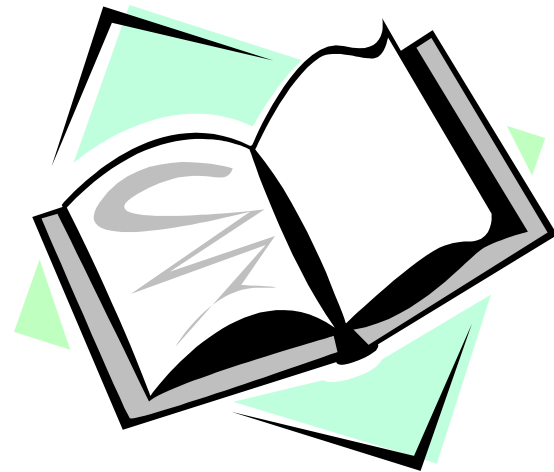
# Lecture Outline

- Authentication protocols: password based, challenge-response, zero-knowledge



# Recommended Reading

- HAC: Chapter 10 for authentication protocols



# Authentication

- **Entity authentication (identification)**: the process whereby one party is assured of the identity of a second party involved in a protocol and that the second has actually participated.
- **Data source authentication**: is represents an indication about the source of the data.



# Properties of Identification Protocols

- Computational efficiency
- Communication efficiency
- Security requirement of communication channels
- Trust in verifier
- Storage of secrets
- Involvement of a third party
- Nature of trust in the third party
- Nature of security: provable security, zero-knowledge security

# Authentication Using Fixed Passwords

- Prover authenticates to a verifier using a password.
- Require secure communication channels
- Total trust in verifier
- Passwords must be kept in encrypted password files or just digests of passwords are kept.
- Attacks:
  - Replay of fixed passwords
  - Online exhaustive password search
  - Offline password-guessing and dictionary attacks

# Unix crypt Algorithm

- Used to store Unix passwords
- Information stored in `/etc/passwd` is:
  - Iterated DES encryption of 0 (64 bits), using the password as key
  - 12 bit random salt taken from the system clock time at the password creation
- Unix uses salting to change the expansion function in DES
  - to make dictionary attacks more difficult.
  - also to prevent use of off-the-shelf DES chips

# One-time passwords

- Each password is used only once
  - Defend against passive adversaries who eavesdrop and later attempt to impersonate
- Variations
  - shared lists of one-time passwords
    - challenge-response table
  - sequentially updated one-time passwords
  - one-time password sequences based on a one-way function

# Lamport's One-Time Password

Stronger authentication than password-based

- One-time setup:
  - A selects a value  $w$ , a hash function  $H()$ , and an integer  $t$ , computes  $w_0 = H^t(w)$  and sends  $w_0$  to B
  - B stores  $w_0$
- Protocol: to identify to B for the  $i^{\text{th}}$  time,  $1 \leq i \leq t$ 
  - A sends to B:  $A, i, w_i = H^{t-i}(w)$
  - B checks  $i = i_A, H(w_i) = w_{i-1}$
  - if both holds,  $i_A = i_A + 1$

# Challenge-Response Protocols

- Goal: one entity authenticates to other entity proving the knowledge of a secret, 'challenge'
- Time-variant parameters used to prevent replay, interleaving attacks, provide uniqueness and timeliness : nonce (used only once)
- Three types:
  - Random numbers
  - Sequences
  - Timestamp

# Challenge-Response Protocols

- **Random numbers:**
  - pseudo-random numbers that are unpredictable to an adversary;
  - vulnerable to birthday attacks, use larger sample;
  - must maintain state;
  - do not prevent interleaving attacks (parallel sessions)
- **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.

# Challenge-response based on symmetric-key encryption

- Unilateral authentication, timestamp-based
  - A to B:  $E_K(t_A, B)$
- Unilateral authentication, random-number-based
  - B to A:  $r_B$
  - A to B:  $E_K(r_B, B)$
- Mutual authentication, using random numbers
  - B to A:  $r_B$
  - A to B:  $E_K(r_A, r_B, B)$
  - B to A:  $E_K(r_B, r_A)$

# Challenge-Response Protocols Using Digital Signatures

- unilateral authentication with timestamp  
A → B: cert<sub>A</sub>, t<sub>A</sub>, B, S<sub>A</sub>(t<sub>A</sub>, B)
- unilateral authentication with random numbers  
A ← B: r<sub>B</sub>  
A → B: cert<sub>A</sub>, r<sub>A</sub>, B, S<sub>A</sub>(r<sub>A</sub>, r<sub>B</sub>, B)
- mutual authentication with random numbers  
A ← B: r<sub>B</sub>  
A → B: cert<sub>A</sub>, r<sub>A</sub>, B, S<sub>A</sub>(r<sub>A</sub>, r<sub>B</sub>, B)  
A ← B: cert<sub>B</sub>, A, S<sub>B</sub>(r<sub>B</sub>, r<sub>A</sub>, A)

# Zero-Knowledge Protocols

- **Motivation:**
  - Password-based protocols: when Alice authenticates to a server, she gives her password, so the server can then impersonate her.
  - Challenge-response improves on this, but still reveals partial information.
- **Zero-knowledge protocols:** allows a prover to prove that it possesses a secret without revealing any information of use to the verifier.

# Fiat-Shamir Identification Protocol

## Setup:

- 1) A trusted server T makes public  $n = pq$ , keeping secrets  $p$  and  $q$  (selected as in RSA)
- 2) Every claimant A selects  $0 < s < n$ ,  $\gcd(s, n) = 1$ , computes  $v = s^2 \bmod n$  and registers  $v$  with the server T

## Protocol run $t$ rounds, proof accepted if all $t$ rounds succeed

1. A selects  $r$ ,  $0 < r < n$ , and sends to B  $x = r^2 \bmod n$
2. B randomly selects  $e = 0$  or  $e = 1$ , and sends  $e$  to A
3. If  $e = 0$ , A sends  $y = r$ , else sends  $y = rs \bmod n$
4. B rejects  $y = 0$ , otherwise accept if  $y^2 \equiv x v^e \bmod n$

# Schnorr Identification Protocol

Like DSA select public  $(p, q, \alpha, \beta)$ , and  $t$ ,  
 $t$  defines the security of the system

**Setup:** a claimant  $A$  has an identifier  $I_A$ , private  $a$ ,  $0 < a < q$ ,  
 $v = \beta^{-a} \bmod p$ , obtain certificate that provides  
authenticity of  $v$ ,  $\text{cert}_A$

**Protocol:**

$A$  selects  $0 < r < q$ , computes  $x = \beta^r \bmod p$ ,

$B$  returns challenge  $1 \leq e \leq 2^t \leq q$

$A$  sends  $y = (ae + r) \bmod q$

$B$  computes  $z = \beta^y v^e \bmod p$ , accept  $A$  if  $z = x$

$1 \leq e \leq 2^t \leq q$

# Next Lecture...

- Authentication & Key Agreement Protocols
- Network Authentication services: Kerberos

