Goals

- Understand goals of entity authentication
- Understand strength and limitations of entity authentication protocols including passwords
- Understand subtle problems when entity authentication protocols are deployed in practice
- Understand variants of key establishment protocols and subtle attacks

Definitions (ctd)

- Confidentiality
- Integrity
- Availability
- Data authentication
- Data encryption
- Anonymity
- Identification

- Authorisation
- Non-repudiation of origin, receipt
- Contract signing
- Notarisation and Timestamping
- E-voting, e-auction, ...

Identification

- the problem
- passwords
- challenge response with symmetric key and MAC (symmetric tokens)
- challenge response with public key (signatures, ZK)
- biometry

Entity authentication

Hello Bob, I am Alice

Why should I believe her?

entity authentication: one is corroborated of the identity of another party, and of the fact that this party is alive (active) during the protocol
Entity authentication is based on one or more of the following elements:

- **what someone knows**
  - password, PIN
- **what someone has**
  - magstripe card, smart card
- **what someone is** (biometrics)
  - fingerprint, retina, hand shape,...
- **how someone does something**
  - manual signature, typing pattern
- **where someone is**
  - dialback, location based services (GSM, Galileo)

---

**Entity authentication with passwords**

Hello Bob, I am Alice. My password P is Xur%9pLr

OK!

**BUT**

- Eve can guess the password
- Eve can listen to the channel and learn Alice’s password
- Bob needs to know Alice’s secret
- Bob needs to store Alice’s secret in a secure way

Possibility of replay: liveness is missing

---

**Improved identification with passwords**

Hello Bob, I am Alice. My password P is Xur%9pLr

OK!

Bob stores f(P) rather than Alice’s secret P

- it is difficult to deduce P from f(P)

---

**Password entropy: effective key length**

![Password entropy chart]

Problem: passwords from dictionaries

---

**Improved+ identification with passwords**

Hello Bob, I am Alice. My password P is Xur%9pLr

OK!

Bob stores f(P,S) || S rather than Alice’s secret P

- it is harder to attack the passwords of all users simultaneously

---

**Example: UNIX**

- Function f(P) = DES applied 25 times to the all zero plaintext with as key the password P (8 7-bit characters)
- Salt: 12-bit modification to DES
- etc/passwd public
- PC: 20–40 million passwords/second
- But time-memory tradeoff...
  - Precomputation per salt $2^{25} . 2^{25}$
  - Storage per salt: 2 Terabyte
  - Find one key in time $25.2^{38}$
Improving password security

- Apply the function $f$ “$x$” times to the password (iteratively)
  - if $x = 100$ million, testing a password guess takes a few seconds
  - need to increase $x$ with time (Moore’s law)
  - examples: PBKDF2 (Password-Based Key Derivation Function 2), scrypt, bcrypt
- Disadvantage: one cannot use the same hashed password file on a faster server and on an embedded device with an 8-bit microprocessor
  - need to use different values of $x$ depending on the computational power of the machine

Problem: human memory is limited

- Solution: store key $K$ on magstripe, USB key, hard disk
- Stops guessing attacks

But this does not solve the other problems related to passwords

And now you identify the card, not the user….

Improvement: Static Data Authentication

- Replace $K$ by a signature of a third party CA (Certification Authority) on Alice’s name: $\text{Sig}_{SKCA}(\text{Alice}) = \text{special certificate}$
- Advantage: can be verified using a public string $\text{PK}_{CA}$
- Advantage: can only be generated by CA
- Disadvantage: signature = 40..128 bytes
- Disadvantage: can still be copied/intercepted

“Certificate” for static data authentication

- Unique name owner
- Unique serial number
- Validity period
- Revocation information
- Name of issuing CA
- CA’s Digital signature on the data in the certificate

Entity authentication with symmetric token

Challenge response protocol

- Random number $r$
- $\text{MACK}(r)$

• Eavesdropping no longer effective
• Bob still needs secret key $K$

Detects whether Alice is alive!

Entity authentication with symmetric token

With implicit challenge from clock

- $\text{MACK}(\text{time})$

• Eavesdropping no longer effective
• Bob still needs secret key $K$
• Resynchronization mechanism needed
Lamport’s one-time passwords

iterated one-way function

$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow \ldots \rightarrow x_t$

- Disadvantage: only works with one Bob

Entity authentication with public key token

Challenge response protocol

SK_A \rightarrow PK_A

random number \( r \)

Sig_{SK_A}(r)

- Eavesdropping no longer effective
- Bob no longer needs a secret – only PK_A

Entity authentication with ZK

Zero knowledge

SK_A \rightarrow PK_A

Commitment \( c \)

Challenge \( e \)

Response(SK_A, c, c)

- Mathematical proof that Bob only learns that he is talking to Alice (1 bit of information)
- Bob cannot use this information to convince a third party that he is/was talking to Alice

ZK definitions

- **complete**: if Alice knows the secret, she can carry out the protocol successfully
- **sound**: Eve (who wants to impersonate Alice) can only convince Bob with a very small probability that she is Alice;
- **zero knowledge**: even a dishonest Bob does not learn anything except for 1 bit (he is talking to Alice); he could have produced himself all the other information he obtains during the protocol.

Overview Identification Protocols

<table>
<thead>
<tr>
<th>Identification Protocols</th>
<th>Guess</th>
<th>Eavesdrop channel (liveliness)</th>
<th>Impersonation by Bob</th>
<th>Secret info for Bob</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>Magstripe (SK)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>Magstripe (PK)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>3</td>
</tr>
<tr>
<td>Dynamic password</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>Smart card (SK)</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>Smart Card (PK)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>5</td>
</tr>
</tbody>
</table>

Entity authentication with password

Challenge response protocol

P \rightarrow random number \( r \)

MAC_P(r)

- Eavesdropping no longer effective
- Bob still needs secret key \( P \)
- Exhaustive search for \( P \) is easy based on a single transcript
Entity authentication in practice

- Phishing – mutual authentication
- Forward credentials - biometry
- Interrupt after initial authentication – authenticated key establishment
- Mafia fraud – distance bounding
- Protocol errors – check that local device authentication is linked to entity authentication protocol (example: EMV)

Mutual authentication

- Phishing is impersonating of the verifier (e.g. the bank)
- Most applications need entity authentication in two directions
- !! This is not complete the same as 2 parallel unilateral protocols for entity authentication

2 stage authentication

- Local: user to device
- Device to rest of the world

Biometry

- Based on our unique features
- Identification or verification
  - Is this Alice?
  - Check against watchlist
  - Has this person ever registered in the system?

Biometric procedures

- Registration
- Template extraction
- Measurement
- Processing
- Template matching
- Link with applications

Robustness/performance

- Performance evaluation
  - False Acceptance Ratio or False Match Rate
  - False Rejection Ratio or False Non-Match Rate
- Application dependent
Robustness/performance (2)

Fingerprint
- Used for PC/laptop access
- Widely available
- Reliable and inexpensive
- Simple interface

Fingerprint (2)
- Small sensor
- Small template (100 bytes)
- Commercially available
  - Optical/thermical/capacitive
  - Liveness detection
- Problems for some ethnic groups and some professions
- Connotation with crime

Fingerprint (3): gummy fingers

Hand geometry
- Flexible performance tuning
- Mostly 3D geometry
- Example: 1996 Olympics

Voice recognition
- Speech processing technology well developed
- Can be used at a distance
- Can use microphone of our gsm
- But tools to spoof exist as well
- Typical applications: complement PIN for mobile or domotica
Iris Scan
- No contact and fast
- Conventional CCD camera
- 200 parameters
- Template: 512 bytes
- All ethnic groups
- Reveals health status

Retina scan
- Stable and unique pattern of blood vessels
- Invasive
- High security

Manual signature
- Measure distance, speed, accelerations, pressure
- Familiar
- Easy to use
- Template needs continuous update
- Technology not fully mature

Facial recognition
- User friendly
- No cooperation needed
- Reliability limited
- Robustness issues
  - Lighting conditions
  - Glasses/hair/beard/...

<table>
<thead>
<tr>
<th>Feature</th>
<th>Uniqueness</th>
<th>Permanent</th>
<th>Performance</th>
<th>Acceptability</th>
<th>Spoofing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facial</td>
<td>Low</td>
<td>Average</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Fingerprint</td>
<td>High</td>
<td>High</td>
<td>High??</td>
<td>Average</td>
<td>High??</td>
</tr>
<tr>
<td>Hand geometry</td>
<td>Average</td>
<td>Average</td>
<td>Average</td>
<td>Average</td>
<td>Average</td>
</tr>
<tr>
<td>Iris</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Retina</td>
<td>High</td>
<td>Average</td>
<td>High</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Signature</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Voice</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>

Biometry: pros and cons
- Real person
- User friendly
- Cannot be forwarded
- Little effort for user
- Privacy (medical)
- Intrusive?
- Liveliness?
- Cannot be replaced
- Risk for physical attacks
- Hygiene
- Does not work everyone, e.g., people with disabilities
- Reliability
- Secure implementation: derive key in a secure way from the biometric
- No cryptographic key
Keeping authenticity alive

• Establish who someone is
• Establish that this person is active/liveliness
• But what if the connection is broken after the initial phase?

\[ \text{secure setup} \]
\[ \text{attacker takes over} \]
\[ \text{random number } r \]
\[ \text{Sig}_{SKA}(r) \]
\[ \text{Rest of communication} \]

Solution

• Authenticated key agreement
• Run a mutual entity authentication protocol
• Establish a key
• Encrypt and authenticate all information exchanged using this key

The mafia fraud
– or the grandmaster chess problem

Location-based authentication

• Distance bounding: try to prove that you are physically close to the verifier
• Other uses of “location”
  – Dial-back: can be defeated using fake dial tone
  – IP addresses and MAC addresses can be spoofed
  – Mobile/wireless communications: operator knows access point, but how to convince others?
  – Trusted GPS: Galileo?

Authentication with device

• E.g. smart card, secure login token
• Needs 2 stages
  – Local: user to device
  – Device to rest of the world
• Are these 2 stages connected properly?

Warning about EMV

http://www.cl.cam.ac.uk/research/security/banking/nopin/oakland10chipbroken.pdf

Guidelines

NIST Special Publication 800-63 Version 1.0.2 (2006): Electronic Authentication Guideline: identifies four levels of assurance

http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf

See http://csrc.nist.gov/publications/PubsSPs.html for about 120 Special Publications (800 Series) from NIST on computer security and cryptography

Key establishment

- The problem
- How to establish secret keys using secret keys?
- How to establish secret keys using public keys?
  - Diffie-Hellman and STS
- How to distribute public keys? (PKI)

Key establishment: the problem

- Cryptology makes it easier to secure information, by replacing the security of information by the security of keys
- The main problem is how to establish these keys
  - 95% of the difficulty
  - integrate with application
  - if possible transparent to end users

GSM (1)

Challenge response protocol

random number r

MAC_K(r)

K → A8

k

 derivation of session key k for this call

encrypt all data with k

GSM (2)

- SIM card with long term secret key K (128 bits)
- secret algorithms
  - A3: MAC algorithm
  - A8: key derivation algorithm
  - A5.1/A5.2: encryption algorithm
- anonymity: IMSI (International Mobile Subscriber Identity) replaced by TIMSI (temporary IMSI)
  - the next TIMSI is sent (encrypted) during the call set-up

Point-to point symmetric key distribution

Before: Alice and Bob share long term secret K_{AB}

generate session key k

\[ E_{K_{AB}}(k || time || Bob) \]

\[ E_k(time || Alice || hello) \]

decrypt

extract k

- After: Alice and Bob share a short term key k
  - which they can use to protect a specific interaction
  - which can be thrown away at the end of the session
- Alice and Bob have also authenticated each other
Symmetric key distribution with 3rd party

Before (KDC=Key Distribution Center)
- Alice shares a long term secret with KDC: $K_A$
- Bob shares long term secret with KDC: $K_B$

```
KDC
generate
session key k
```

Alice needs a key for Bob

```
E K_A(k) || E K_B(k)
```

!! never use this protocol in practice
- it is just a toy example

```
E K_A(k)
E K_B(k)
```

Symmetric key distribution with 3rd party(2)

- After: Alice and Bob share a short term key $k$
- Need to trust third party!
- Single point of failure in system

Kerberos/Single Sign On (SSO)

- Alice uses her password only once per day

```
AS
TGS
```

```
1
2
3
```

Kerberos/Single Sign On (2)

- Step 1: Alice gets a “day key” $K_A$ from AS (Authentication Server)
  - based on a Alice’s password (long term secret)
  - $K_A$ is stored on Alice’s machine and deleted in the evening
- Step 2: Alice uses $K_A$ to get application keys $k_i$ from TGS (Ticket Granting Server)
- Step 3: Alice can talk securely to applications (printer, file server) using application keys $k_i$

A public-key distribution protocol: Diffie-Hellman

- Before: Alice and Bob have never met and share no secrets; they know a public system parameter $\alpha$

```
generate x
compute $\alpha^x$
generate y
compute $\alpha^y$
compute $k=(\alpha^y)^x$
compute $k=(\alpha^x)^y$
```

- After: Alice and Bob share a short term key $k$
  - Eve cannot compute $k$: in several mathematical structures it is hard to derive $x$ from $\alpha^x$
  (this is known as the discrete logarithm problem)

Diffie-Hellman (continued)

```
generate x
compute $\alpha^x$
generate y
compute $\alpha^y$
compute $k=(\alpha^y)^x$
compute $k=(\alpha^x)^y$
```

- BUT: How does Alice know that she shares this secret key $k$ with Bob?
- Answer: Alice has no idea at all about who the other person is! The same holds for Bob.
Meet-in-the middle attack

- Eve shares a key $k_1$ with Alice and a key $k_2$ with Bob
- Requires active attack

$$k_1 = (\alpha^{y_1})^x = (\alpha^x)^{y_1}$$
$$k_2 = (\alpha^{y_2})^x = (\alpha^x)^{y_2}$$

Station to Station protocol (STS)

- The problem can be fixed by adding digital signatures
- This protocol plays a very important role on the Internet (under different names)

$$k = (\alpha^x)^y$$
$$\sqrt{\text{Sig}_B(\alpha^y || \alpha^x)}$$

IKE - Main Mode with Digital Signatures

$$H \text{ is equal to prf or the hash function tied to the signature algorithm}$$

$$\text{all inputs are concatenated}$$

Key transport using RSA

- How does Bob know that $k$ is a fresh key?
- How does Bob know that this key $k$ is coming from Alice?
- How does Alice know that Bob has received the key $k$ and that Bob is present (entity authentication)?

$$E_{PK_B}(k) \xrightarrow{SK_B} E_{PK_A}(k) \xrightarrow{SK_A}$$

Key transport using RSA (2)

- Freshness is solved with a timestamp $t_A$
**Key transport using RSA (3)**

\[ \text{generate } k \quad \text{Sig}_{SKA}(E_{PKB}(k \ || \ t_A)) \quad \text{decrypt using } SKB \text{ and verify using } PKA \]

- Alice authenticates by signing the message
- There are still attacks (signature stripping…)

**Key transport using RSA (4): X.509**

\[ \text{generate } k \quad \text{Sig}_{SKA}(B || t_A \ || E_{PKB}(A \ || \ k)) \quad \text{decrypt using } SKA \text{ and verify using } PK_A \]

Mutual: B can return a similar message including part of the first message
Problem (compared to D-H/STS): lack of forward secrecy
If the long term key \( SKB \) of Bob leaks, all past session keys can be recovered!

---

**A simple protocol**

\[ k \quad n_A \quad n_B \]

\[ E_A(n_A||n_B) \quad n_B \]

**Reflection attack**

- Eve does not know \( k \) and wants to impersonate Bob

\[ k \quad n_A \quad n_A' \]

\[ E_k(n_A||n_A') \quad E_k(n_A||n_A' = n_B) \]

---

**Conclusions**

- Properties of protocols are subtle
- Many standardized protocols exist
  - ISO/IEC, IETF
- Difficulty: which properties are needed for a specific application
- Rule #1 of protocol design: **Don’t**
  - not even by simplifying existing protocols

**Recommended reading**