Summary

• Key Agreement and PKI-based Authentication
  – Security Model
  – Example
• Password-based Authentication
  – Security Model
  – Example
• Group Key Agreement
  – Security Model
  – Example
• Conclusion
Summary

- Key Agreement and PKI-based Authentication
  - Security Model
  - Example
- Password-based Authentication
  - Security Model
  - Example
- Group Key Agreement
  - Security Model
  - Example
- Conclusion

Authenticated Key Exchange

Two parties agree on a common secret key, in order to establish a secret channel (e.g. SSL)

- Implicit authentication
  only the intended partners can compute the session key
- Semantic security
  - the session key is indistinguishable from a random string
  - modeled via a Test-query
Further Properties

- Mutual authentication
  they are both sure to share the secret
  with the people they think they do

- Forward-secrecy
  even if a long-term secret data is corrupted,
  previous shared secrets are still
  semantically secure

Formal Model

A can ask
- reveal-queries
- test-query
- execute-queries
- send-queries
- corrupt-queries

0/1
Semantic Security

- A misuse of the secret data is modeled by the `reveal`-query, which is answered by this secret data.
- For the semantic security, the adversary asks one `test`-query which is answered, according to a bit $b$, by:
  - $b=0$: the actual secret data
  - $b=1$: a random string
  ⇒ the adversary has to guess this bit $b$

Security Definitions (AKE)

<table>
<thead>
<tr>
<th>PROTOCOL</th>
<th>Public data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flip a coin $b$</td>
<td>«Test» a key $sk$</td>
</tr>
<tr>
<td>$sk$ if $b=0$, random if $b=1$</td>
<td>Outputs $b'$ (guess for $b$)</td>
</tr>
</tbody>
</table>
Passive/Active Adversaries

- Passive adversary: history built using the execute-queries → transcripts
- Active adversary: entire control of the network with send-queries:
  - to send message to Alice or Bob
    (in place of Bob or Alice respectively)
  - to intercept, forward and/or modify messages

Forward Secrecy

Forward secrecy means that the adversary cannot distinguish a session key established before any corruption of the long-term private keys:

- the corrupt-query is answered by the long-term private key of the corrupted party
- then the test-query must be asked on a session key established before any corrupt-query
Summary

- Key Agreement and PKI-based Authentication
  - Security Model
  - Example
- Password-based Authentication
  - Security Model
  - Example
- Group Key Agreement
  - Security Model
  - Example
- Conclusion

Diffie-Hellman Key Exchange

The most classical key exchange scheme has been proposed by Diffie-Hellman:

\[ G = \langle g \rangle, \text{ cyclic group of prime order } q \]

- Alice chooses a random \( x \in \mathbb{Z}_q \), computes and sends \( X = g^x \)
- Bob chooses a random \( y \in \mathbb{Z}_q \), computes and sends \( Y = g^y \)
- They each can compute the session key

\[ K = Y^x = X^y \]
Properties

- If flows are authenticated, it is well-known to provide the semantic security of the session key under the **Decisional Diffie-Hellman Problem**
- If one derives the session key as $k = H(K)$, where $H$ is assumed to behave like a random oracle, semantic security is relative to the **Computational Diffie-Hellman Problem**

Authenticated Key Exchange

$\begin{align*}
\text{Alice} & \quad (S_a, P_a) \\
x \in \mathbb{Z}_q, \ X = g^x \\
\quad \quad \quad \quad \quad \quad \text{Bob, } X, \textbf{S}(S_a, X) \\
\quad \quad \quad \quad \quad \quad K = Y^x \\
\quad \quad \quad \quad \quad \quad k = H(\text{Alice, Bob, X, Y, K}) \\
\text{Bob} & \quad (S_b, P_b) \\
y \in \mathbb{Z}_q, \ Y = g^y \\
\quad \quad \quad \quad \quad \quad K = X^y \\
\end{align*}$

But there is no explicit authentication $\Rightarrow$ replay attacks
Replay Attack

The adversary intercepts "Bob, X, S(S_a, X)"

He can initiate a new session with it

Bob believes it comes from Alice

- Bob accepts the key, but does no share it with Alice
  ⇒ no mutual authentication
- The adversary does not know the key either
  ⇒ still semantic security

Mutual Authentication

Adding key confirmation rounds: mutual authentication

[Bellare-Pointcheval-Rogaway Eurocrypt ‘00]
Summary

• Key Agreement
  and PKI-based Authentication
  – Security Model
  – Example

• Password-based Authentication
  – Security Model
  – Example

• Group Key Agreement
  – Security Model
  – Example

• Conclusion

Password-based Authentication

The parties share a low-entropy secret
  – a password
  – exhaustive search is possible (say $2^{20}$)

• Basic attack: on-line exhaustive search
  – the adversary guesses a password
  – tries to play the protocol
  – failure $\Rightarrow$ erase the password from the list
  – restart…

after $2^{20}$ attempts, the adversary wins
Dictionary Attack

The on-line exhaustive search
- cannot be prevented
- can be made less serious (delay, limitations, ...)

We want it to be the best attack...

Off-line exhaustive search:
- passive/active attack
- failure ⇒ erase MANY passwords from the list
  this is called dictionary attack

Summary

- Key Agreement
  and PKI-based Authentication
  - Security Model
  - Example
- Password-based Authentication
  - Security Model
  - Example
- Group Key Agreement
  - Security Model
  - Example
- Conclusion
**Example: EKE**

The most famous scheme EKE: Encrypted Key Exchange

Must be done carefully

<table>
<thead>
<tr>
<th>Alice</th>
<th>Password $\pi$</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x \in \mathbb{Z}_q$, $X = g^x$</td>
<td>$X' = E_{\pi}(\text{Bob, } X)$</td>
<td>$X = D_{\pi}(X')$</td>
</tr>
<tr>
<td>$Y = D_{\pi}(Y')$</td>
<td>$Y' = E_{\pi}(\text{Alice, } Y)$</td>
<td>$y \in \mathbb{Z}_q$, $Y = g^y$</td>
</tr>
<tr>
<td>$K = Y^x$</td>
<td></td>
<td>$K = X^y$</td>
</tr>
<tr>
<td>$k = H(\text{Alice, Bob, } X, Y, K)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Any redundancy is serious:
- From $X'$, for any password $\pi$
  - decrypt $X'$
  - check whether it begins with “Bob”
**EKE - AuthA**

Alice

\[ x \in \mathbb{Z}_q, \quad X = g^x \]

\[ Y = \mathcal{D}_n(Y') \]

\[ K = Y^x \]

\[ k = H(\text{Alice}, \text{Bob}, X, Y, K) \]

**Bob**

\[ X = \mathcal{D}_n(X') \]

\[ y \in \mathbb{Z}_q^*, \quad Y = g^y \]

\[ K = X^y \]

**AuthA**

Bellare-Rogaway 2000

Alice

\[ x \in \mathbb{Z}_q, \quad X = g^x \]

\[ Y = \mathcal{D}_n(Y') \]

\[ K = Y^x \]

\[ k_1 \text{ correct?} \]

\[ k = H(\text{Alice}, \text{Bob}, X, Y, K) \]

**Bob**

\[ X = \mathcal{D}_n(X') \]

\[ y \in \mathbb{Z}_q^*, \quad Y = g^y \]

\[ K = X^y \]

**EKE**

Bellovin-Merritt 1992

**Provably secure if E is an ideal cipher**

[Bresson-Chevassut-Pointcheval ACM CCS '03]

David Pointcheval

**Improvement**

\[ E \] = an ideal cipher

replaced by the One-Time Pad

\[ E_\pi(m) = G(\pi) \oplus m \]

[Bresson-Chevassut-Pointcheval LBNL-53099]

Alice

\[ x \in \mathbb{Z}_q, \quad X = g^x \]

\[ K = Y^x \]

\[ k_1 \text{ correct?} \]

\[ k = H(\text{Alice}, \text{Bob}, X, Y, K) \]

**Bob**

\[ X = X' / G(\pi) \]

\[ y \in \mathbb{Z}_q^*, \quad Y = g^y \]

\[ K = X^y \]

\[ k_1 = H_1(\text{Alice}, \text{Bob}, K) \]
Summary

• Key Agreement and PKI-based Authentication
  – Security Model
  – Example
• Password-based Authentication
  – Security Model
  – Example
• Group Key Agreement
  – Security Model
  – Example
• Conclusion

Model of Communication

• A set of $n$ players, modeled by oracles
• A multicast group consisting of a set of players

$pk_A, sk_A$  $pk_C, sk_C$

$pk_B, sk_B$  $pk_D, sk_D$

Multicast group with $sk$
Modeling the Adversary

- **reveal**: obtain an instance’s session key
- **corrupt**: obtain an instance’s secret key
- **execute**: obtain honest executions of the protocol
- **send**: send messages to instances

![Diagram](image)

Summary

- **Key Agreement** and PKI-based Authentication
  - Security Model
  - Example
- **Password-based Authentication**
  - Security Model
  - Example
- **Group Key Agreement**
  - Security Model
  - Example
- **Conclusion**
Group Key Exchange

• Generalization of the 2-party DH, the session key is $k = H(g^{x_1 x_2 \ldots x_n})$

• Ring-based algorithm
  – up-flow: the contributions of each instance are gathered
  – down-flow: the last instance broadcasts the result
  – end: instances compute the session key from the broadcast

The Algorithm

– Up-flow: $U_i$ raises received values to the power $x_i$
– Down-flow: $U_n$ broadcasts (except $g^{x_1 x_2 \ldots x_n}$)
Everything is authenticated (Signature/MAC)
Group CDH

• The CDH generalized to the multi-party case
  – given the values $g^{\prod x_i}$ for some choice of proper subset of $\{1, \ldots, n\}$
  – one has to compute the value $g^{x_1 \cdots x_n}$

• Example ($n=3$ and $I=\{1,2,3\}$)
  – given the set of the blue values $g, g^{x_1}, g^{x_2}, g^{x_1x_2}$
  – compute the red value $g^{x_1x_3}, g^{x_2x_3}, g^{x_1x_2x_3}$

• The GCDH $\iff$ DDH and CDH
  [Bresson-Chevassut-Pointcheval SAC '02]

Security Result

• Theorem (in the random oracle model)
  $$\text{Adv}^{\text{ake}}(T,n,q_s,q_e) \leq 2 q_s^n q_h \cdot \text{Succ}^{\text{gcdh}}(n,T) + 2n \cdot \text{Succ}^{\text{sign}}(q_s,T)$$
  [Bresson-Chevassut-Pointcheval-Quisquater ACM-CCS '01]

• Bad reduction:
  one has to guess the attacked session
  $\Rightarrow$ factor $q_s^n$

Random Self-Reducibility
  $$\text{Adv}^{\text{ake}}(T,n,q_s,q_e) \leq 2 q_h \cdot \text{Succ}^{\text{gcdh}}(n,T+n q_s) + 2n \cdot \text{Succ}^{\text{sign}}(q_s,T)$$
Dynamic Case

- When a party leaves or joins the group, the protocol has to be run again ⇒ costly when the network is not very reliable
- One can exploit the secret already shared ⇒ dynamic case

[Bresson-Chevassut-Pointcheval Asiacrypt '01]

Security Result

- Group of $n$ people
- Tested group of size $s$
- Number of dynamic modifications (setup, join, remove): $Q$
- Time: $T$

$$\text{Adv}^{\text{ake}}(A) \leq 2Q \cdot C_n^s \cdot q_h \cdot \text{Succ}^{\text{gcdh}}(s,T) + 2n \cdot \text{Succ}^{\text{sign}}(q_s,T)$$
Summary

• Key Agreement and PKI-based Authentication
  – Security Model
  – Example

• Password-based Authentication
  – Security Model
  – Example

• Group Key Agreement
  – Security Model
  – Example

• Conclusion

Conclusion

• Provably secure (Group) AKE
• But still not “practical security” for group AKE
• Various authentication modes
  Password-based
  – efficient
  – practical security
  – user-friendly
  ⇒ very promising
Our Accomplishment

Authentication of the parties:
- Public Key Infrastructures (signatures)
  [ACM CCS ‘01 - Asiacrypt ‘01]
- Secret keys - MAC
  [Eurocrypt ‘02]
- Passwords
  [Asiacrypt ‘02 - ACM CCS ‘03]