Basic key exchange

Pawel Wocjan

Department of Electrical Engineering & Computer Science
University of Central Florida
wocjan@eecs.ucf.edu

October 28, 2013
Trusted third parties
key management

problem: storing mutual secret keys for $n$ users is difficult
each user has to store $n$ keys
A better solution

use an online trusted third party (TTP)

Alice

\[ k_A \]

Bob

\[ k_B \]

TTP

\[ k_C \]

Charlie

\[ k_D \]

Dave

each user has to store only one key
Toy protocol for generating secret keys

Alice \( k_A \)  

Alice wants key with Bob

Alice wants key with Bob

TTP

chooses random \( k_{AB} \)

\[ E(k_A, AB||k_{AB}) \]

\[ t = E(k_B, AB||k_{AB}) \]

\[ (E, D) \] is a CPA-secure cipher

\( t \) ticket

Bob \( k_B \)
Toy protocol for generating secret keys

- Alice wants a shared secret key with Bob
- Eavesdropping security only
- Eve sees

\[ E(k_A, AB\|k_{AB}) \text{ and } E(k_B, AB\|k_{AB}) \]

- \((E, D)\) is CPA-secure \(\Rightarrow\) Eve learns nothing about \(k_{AB}\)
- Observe that TTP is needed for every key exchange and TTP knows all session keys
- This is the basis of Kerberos system
Toy protocol is insecure against active attacks

- for instance, insecure against replay attacks
- attacker records session between Alice and merchant Bob, say, a book order
- attacker replays session to Bob ⇒ Bob thinks Alice wants to order another copy
Key question

- can we generate shared keys without an online trusted third party?
- yes, generation of shared secret keys is possible without TTP
- this marked the starting point for public key cryptography
  - Mekle (1974)
  - Diffie-Hellman (1976)
  - RSA (1977)
- more recently: ID-based encryption and functional encryption
  - [ID-based cryptography](http://en.wikipedia.org/wiki/ID-based_cryptography)
  - [Functional encryption](http://en.wikipedia.org/wiki/Functional_encryption)
Merkle Puzzle
Key exchange without an online TTP

- goal: Alice and Bob want shared secret key, unknown to Eve
- for now: security against eavesdropping, no tampering
- can this be achieved with generic symmetric crypto?
- yes, but in a very inefficient way
Merkle Puzzles (1974)

- main tool: puzzles, i.e., problems that can be solved with some effort
- example of a puzzle:

  let $E(k, m)$ be a symmetric cipher with $k \in \{0, 1\}^{128}$

  - puzzle($P$) = $E(P, \text{message})$ where $P = 0^{96} \| b_1 \ldots b_{32}$
  - task: find $P$ by trying all $2^{32}$ possibilities
Merkle Puzzles

Alice: prepare $2^{32}$ puzzles
choose random $P_i \in \{0, 1\}^{32}$ and $x_i, k_i \in \{0, 1\}^{128}$
set puzzle$_i = E(0^{96} || P_i, \text{Puzzle#} \#x_i || k_i)$
send puzzle$_1, \ldots, \text{puzzle}_{2^{32}}$ to Bob

Bob
choose random puzzle puzzle$_j$
solve it to obtain $(x_j, k_j)$
send $x_j$ to Alice

Alice
lookup puzzle with number $x_j$
use $k_j$ as shared secret key
Security analysis

- Alice’s work: $O(n)$ to prepare $n$ puzzles
- Bob’s work: $O(n)$ to solve one puzzle
- Eve’s work: $O(n^2)$ because she has to solve all puzzles
- quadratic gap is best possible in the black-box setting
Key exchange without an online TTP

- goal: Alice and Bob want to establish a secret key, unknown to the Eve
- for now, we seek security against eavesdropping only (no tempering)
- can this be done with an exponential gap?
Diffie-Hellman protocol

- let $G$ be a cyclic group of order $|G|$ that is generated by $g$

  \[ G = \langle g \rangle = \{ g^0, \ldots, g^{|G|-1} \} \]

- Alice chooses random $a \in \{1, \ldots, |G| - 1\}$ and sends

  \[ \text{Alice, } A = g^a \text{ to Bob} \]

- Bob chooses random $b \in \{1, \ldots, |G| - 1\}$ and sends

  \[ \text{Bob, } B = g^b \text{ to Alice} \]

- the common secret key between Alice and Bob

  \[ B^a = g^{b \cdot a} = g^{a \cdot b} = A^b \]
The Diffie-Hellman protocol over $\mathbb{F}_p^\times$

- let $p$ be a large prime
- let $\mathbb{F}_p = \{0, 1, \ldots, p - 1\}$ and $\mathbb{F}_p^\times = \{1, \ldots, p - 1\}$
- $(\mathbb{F}_p, +, \cdot)$ endowed the modular addition $+$ and modular multiplication mod $p$ is a finite field
- $(\mathbb{F}_p^\times, \cdot)$ endowed modular multiplication is a cyclic group of order $p - 1$
How hard is the DH function modulo $p$?

- Suppose prime $p$ is $n$ bits long.
- Best known algorithm (GNFS): run time $\exp(\tilde{O}(n^{1/3}))$.

<table>
<thead>
<tr>
<th>Cipher key size</th>
<th>Modulus size</th>
<th>Elliptic curve size</th>
</tr>
</thead>
<tbody>
<tr>
<td>80 bits</td>
<td>1024 bits</td>
<td>160 bits</td>
</tr>
<tr>
<td>128 bits</td>
<td>3072 bits</td>
<td>256 bits</td>
</tr>
<tr>
<td>256 bits</td>
<td>15360 bits</td>
<td>512 bits</td>
</tr>
</tbody>
</table>

- As a result: slow transition away from mod $p$ to elliptic curves.
- Google uses Elliptic Curve DH (ECDH) protocol for key exchange.
Insecure against man-in-the-middle attack
Public key encryption
Public key encryption

- def: a public-key encryption system is a triple of algorithms \((G, E, D)\)
  - \(G\): randomized algorithm outputs a key pair \((k_p, k_s)\)
  - \(E(k_p, m)\): randomized algorithm encrypts \(m \in \mathcal{M}\) to \(c \in \mathcal{C}\)
  - \(D(k_s, m)\): deterministic algorithm either decrypts \(c \in \mathcal{C}\) to \(m \in \mathcal{M}\) or \(\perp\)

- subject to the consistency condition:
  \[
  \forall k_p, k_s \text{ output by } G \text{ we must have } \\
  \forall m \in \mathcal{M} \quad D(k_s, E(p_k, m)) = m
  \]
Semantic security
Establishing a shared secret key

- Alice runs $G$ and obtains $(k_p, k_s)$ and sends to Bob
  - $\text{Alice}, k_p$

- Bob chooses random $x \in \{0, 1\}^{128}$ and sends to Alice
  - $\text{Bob}, c = E(p_k, x)$

- Alice decrypts $c$ and obtains $x$, which is the shared secret
Security

- adversary sees $k_p, E(p_k, x)$ and seeks to determine $x$
- semantic security ensures that adversary cannot recover $x$
- this can be used to derive a session key from $x$
- this protocol is vulnerable to the man-in-the-middle attack
Public key constructions

▶ constructions generally rely on hard problems from number theory and algebra