Workshop 5, Week 6
Information Security and Privacy, INFO30006
August 2017

Vanessa Teague, vjteague@unimelb.edu.au

August 28, 2017

This workshop considers exchanging keys for end-to-end encrypted communications on the Internet. We'll examine the security properties that these protocols have, and try to understand what some modifications would do to change them.

Before you begin, briefly review the definitions of *forward secrecy* and *future secrecy* from lectures.

Suppose Bob wants to send a secret message to Alice without anyone else reading it. George wants to learn the contents of the message. We assume that George controls everything about the communications infrastructure, and can read, delay, block, add or modify messages between Alice and Bob. However, George does not control Alice or Bob's devices and he cannot read an encrypted message for which he does not have the key.

1 Protocol 1: Super-simple RSA-based key exchange

The simplest protocol goes like this:

1. Alice advertises her RSA public key, $RSApubKey_A$
2. Bob encrypts the message m using RSA and Alice's public key, then sends the ciphertext $c = RSA(m, RSApubKey_A)$ to Alice.

Question: Assuming that the RSA algorithm is secure, give one way that George could read the message.

1.1 Protocol 2: Using RSA to exchange a key

A faster protocol is this:

1. Alice advertises her RSA public key, $RSApubKey_A$
2. Bob generates a secret key k for a secret-key encryption algorithm like AES.
3. Bob encrypts the message m using AES and k, then sends the ciphertext $c_m = AES(m, k)$ to Alice.
4. Bob encrypts the key k using RSA and Alice's public key, then sends the ciphertext $c_k = RSA(k, RSApubKey_A)$ to Alice.

Questions:

1. Does the attack you found on Protocol 1 still work?
2. Suppose that Bob has an independent way of checking which RSA public key truly belongs to Alice. Can you think of a way for George to read the message?
3. Does the protocol provide *forward secrecy* or *future secrecy* against an attacker who learns Alice's RSA public key?
4. Does the protocol provide *forward secrecy* or *future secrecy* against an attacker who learns only the short-term secret key k for one message?

2 Protocol 3: Diffie-Hellman key exchange

Now consider the Diffie-Hellman key exchange protocol we discussed in class.

1. Alice and Bob agree on public values g and p (a large prime). George learns these numbers too.
2. Alice generates a secret a and sends $A = g^a \bmod p$ to Bob.
3. Bob generates a secret b and sends $B = g^b \bmod p$ to Alice.
4. Alice and Bob both compute $k = g^{ab} \bmod p$.

5. Bob sends Alice a message encrypted with their shared key k. For example, $c = AES(m, k)$

Question: Explain how George can read the message. (Hint: this is hard but we did it in class.)

3 Protocol 4: Signed Diffie-Hellman key exchange

Now consider a signed version. Suppose that Bob has an independent way of checking which public key *for signatures* truly belongs to Alice.

1. Alice and Bob agree on public values g and p (a large prime). George learns these numbers too.

2. Alice generates a secret a and sends $A = g^a \bmod p$ to Bob. **She signs A with her private signing key.**

3. Bob generates a secret b and sends $B = g^b \bmod p$ to Alice.

4. **when Bob receives A from Alice, he checks her signature to make sure A is truly from her.**

5. Alice and Bob both compute $k = g^{ab} \bmod p$.

6. Bob sends Alice a message encrypted with their shared key k. For example, $c = AES(m, k)$

Questions:

1. Assume George doesn't know Alice's private signing key. Does the attack you found on Protocol 3 still work?

2. Suppose that Alice and Bob generate new a and b values (and hence compute a new secret k) every time they communicate. Does the protocol provide *forward secrecy* or *future secrecy* against an attacker who learns only the short-term secret key k for one message?

3. Suppose George learns Alice's private signing key.

    - *Forward secrecy:* Can George decrypt past messages?
    - *Future secrecy:* Can George now intercept future messages?