6.857 Computer and Network Security November 12, 2002

Lecture Notes 19 : Key Establishment Methods

Lecturer: R. Rivest *Scribe: Hamler/Hu/Lee/Sanchala*

1 Introduction

Today we discuss how to establish a shared secret key between two parties that establish secure communication over an untrusted network. The method used to establish a shared secret key must ensure that the key is "fresh", that is the key is not a replay of a previous key, and "ephemeral", that it will go away after the session. This part of the process has many names, such as key establishment, key transport, key agreement, etc.... In addition, the method also includes authentication such that a party knows that the shared key is indeed shared with the desired counter-party.

2 Outline

- Sessions
- Challenge Response
- Kerberos
- RSA Transport
- Needham-Schroeder
- Diffie-Hellman
- Station-To-Station (STS)
- Identity Based Encryption (IBE)

Note: Rivest found an article on CNN news on-line regarding a new law in California where break-ins will have to be disclosed when they disclose confidential information. The following is an except from the article describing the law:

> In April, 2002, hackers broke into the payroll database for the state of California. For more than a month, cybercriminals rooted around in the personal information of 265,000 Golden State employees, ranging from Governor Gray Davis to maintenance workers and clerks.
>
> Worse, the California Controller's Office, which ran the database, failed to notify state employees for more than two weeks after the breach was discovered. Although officials with the Controller's office insisted the break-in probably hadn't resulted in any significant harm, the incident enraged Golden State pols and employees, whose Social

> Security numbers, bank account information, and home addresses were fair game for the hackers.
>
> This lapse sparked what may mark a dramatic shift in legal policy toward cybersecurity. Over strenuous objections from the business lobby, on Sept. 26 California enacted a sweeping measure that mandates public disclosure of computer-security breaches in which confidential information may have been compromised. The law covers not just state agencies but private enterprises doing business in California. Come July 1, 2003, those who fail to disclose that a breach has occurred could be liable for civil damages or face class actions. [1]

A lot of break-ins are not reported and are swept under the floor. It will be interesting to see the effect of this law on attitudes toward security as the number of break-ins reported increase.

3 Key Establishment Methods

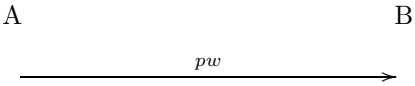
The different key establishment methods assume that the parties involved already have certain capabilities or access to information. The variation of assumptions that each protocol makes for the parties includes:

- Both parties may already have a shared secret key, for long term use, or a short secret such as a password.
- PK of other party, PK of Certificate Authority (many log-in protocols assume this)
- Trusted Third Party (TTP) (Kerberos)
- Asymmetry of computation power (Smart Card vs. bank terminal)
- Mathematical assumptions (IBE)

We begin our inspection of key establishment methods from the ground up.

3.1 Session Key Establishment

The barebones approach to authenticating a user is simple password authentication, originating from the ways when A initiates a session by logging in to B, does her work, and logs out. B just has to ensure that $h(pw) \in DB_B$; no shared secret is used to encrypt communication.

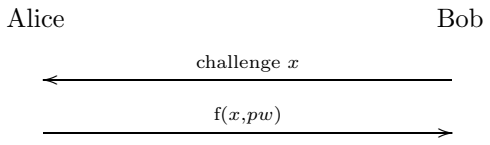


As expected, this method offers no security, as the password is sent in the clear. It is vulnerable to eavesdropping, hijacking, and replay attacks.

[1] Available via: `http://www.itsecure.com.au/news/story.htm?StoryID=272`

3.2 Challenge Response

We improve on the simple approach slightly by having the server send the user a challenge (nonce) x and *Alice* responds with a hash of x and her password.



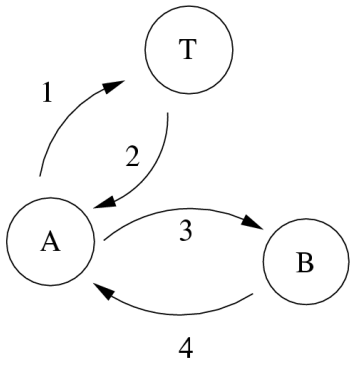
Note: f can denote a function such as SHA-1

By sending a challenge, *Bob* can ensure that *Alice* is initiating the session, and avoid a replay attack. This method, however, is still vulnerable to eavesdropping for the duration of the session, as well as hijacking. The above is presented for historical purposes, as it offers no forward secure line, which is the problem we are trying to solve.

3.3 Kerberos

Kerberos, which is used at MIT, assumes use of a Trusted Third Party T which shares secret keys with all parties.

We assume that A and T share a shared symmetric key K_A, and B and T share a shared secret K_B, such that A and T, and B and T can communicate securely. A and B have no prior relationship, but A can use T to help establish a shared secret between A and B.



1. $A \rightarrow T : A, B, N_A$ (where N_A is a nonce of non-repeating value)
2. $T \rightarrow A : E_{K_B}(K, A, L), E_{K_A}(K, N_A, L, B)$ (where L is the lifetime of the key)
3. $A \rightarrow B : E_{K_B}(K, A, L), E_K(time)$
4. $B \rightarrow A : E_K(time + 1)$

A sends a message to T requesting a key for communication with B. T responds with two messages: a message encrypted with K_B for A to send to B notifying B of the communication and a "ticket" to use for the communication, and a message encrypted with the key of A to notify A of the shared secret. B receives key K and understands that it is a key for communication from A, as endorsed by T. A also wants to make sure that B is there, so she sends $E_K(time)$ as a challenge, the first time E_K is used for secure communication. A and B can use this key for now on.

Note that Step 3 could be $T \rightarrow B$ as well, though it is usually implemented as $A \rightarrow B$.

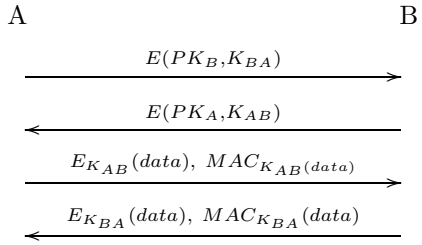
This method assumes that the Trusted Third Party has a good random number generator for generating secrets, and is not corrupt, since T knows the secret keys being used. Also, note that the $A \rightarrow T$ notation makes many assumptions that are not explicitly stated, such has taking care to ensure non-malleability, etc... that we acknowledge by intuition, but may not always be acknowledged during implementation.

Ideally, we would also want to protect against reflection attacks and have two keys rather than one, since a message sent by A could be reflected back to A, and the protocol does not have enough explicitness to distinguish between her own message or B's.

3.4 RSA Transport

In the case of there not being Trusted Third Parties, we use Public Keys for key transport (though there is still a TTP to act as a certificate authority).

A creates a key K_{BA} for B to use to communicate with A, and sends B the key encrypted with B's public key. Likewise, B responds with a key K_{AB} encrypted with A's public key.

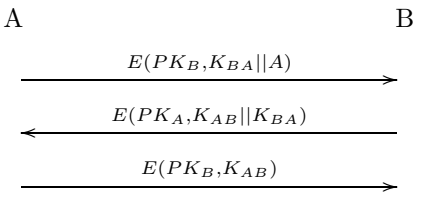


Note that this protocol should be made more explicit by adding a nonce to avoid replay, and include in the key establishing process who the sender and intended recipient are. Also, many times people prefer not to use the same key for encryption as MAC key since the algorithms may interact with each other; therefore K'_{AB}, a subsidiary of K_{AB}, may be used.

This method is not fool-proof; see Diffie-Hellman.

3.5 Needham-Schroeder (1978)

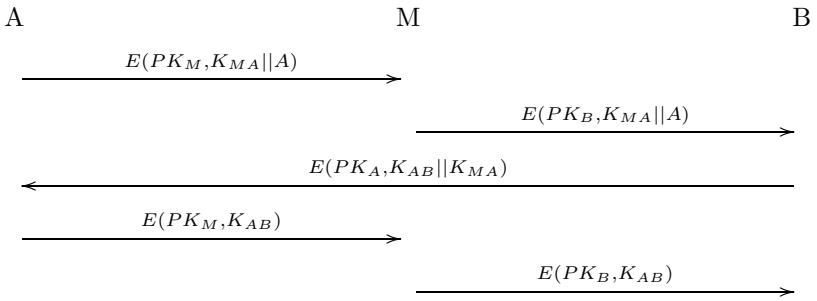
One of the very first protocols used for key establishment is the Needham-Schroeder Protocol, another flavor of key transport. The protocol assumes that each party knows each other's public key, and completes its task in three messages.



A sends to B a message encrypted with B's public key and a key K_{BA} to use for communication from B to A that is concatenated to explicitly state the sender. B responds with a message encrypted with A's public with a new key K_{AB} for A to use to communicate with B, and also returns K_{BA}, which functions as a nonce to avoid replay. A then replies to confirm that she has received K_{AB}. A and B have established keys to use for each direction of communication.

On initial inspection, this protocol was thought to be secure. It prevents replay, explicitly states the sender and recipient, and confirms receipt. However, in 1995 (a relatively long time for a flaw to be discovered), Gavin Lowe discovered a bug that allowed a man-in-the-middle attack that would violate the protocol. An interesting side-note: though Needham-Schroeder was thought to be secure for so long, Lowe discovered this flaw via an automated security analysis tool.

The flaw is as follows:



We have A, middle-man M, and B. A wants to talk to M, however M betrays A's trust and forwards the message to B. B is led to believe that he is talking to A. The protocol is violated in that A receives a key to communicate with B, though she thinks she is talking to M.

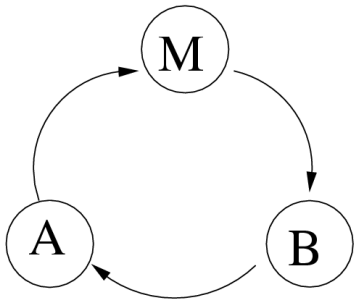


Figure 1: The confusion that the man-in-the-middle attack caused in Needham-Schroeder and who each party thinks he is talking.

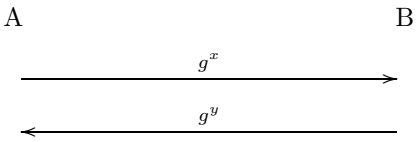
A scenario for abuse of this flaw could be if M and B were web merchants, and A logs on to M. M could impersonate A and buy from A's account. Ultimately, B can be fooled regarding who he thinks he is talking to.

A fix for this situation can be applied in the third message, where B can be more explicit and state that he is the sender. That is, B should send $E(PK_A, K_{AB}||K_{MA}||B)$. A will realize that M is not the sender, and should abort the protocol at this point.

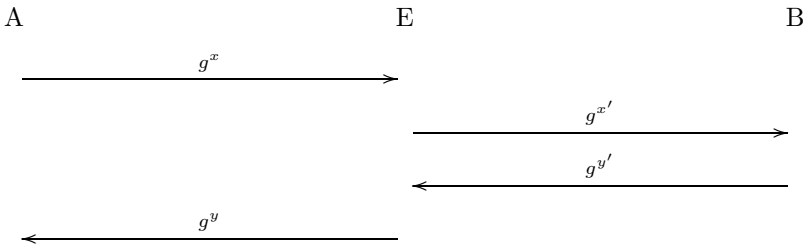
Also, we want to ensure forward security. What if B's private key is exposed? A third party could decode past logs of communication. Forward security would ensure that the compromise of long-term secrets do not compromise past communications. The solution to this flaw is to not encrypt with public keys so that the private key is not used for decryption. This is where Diffie-Hellman comes into play. Moral: Don't encrypt with public keys (though signing is okay).

3.6 Diffie-Hellman

Our next step on our tour is to examine Diffie-Hellman, which we have seen before.



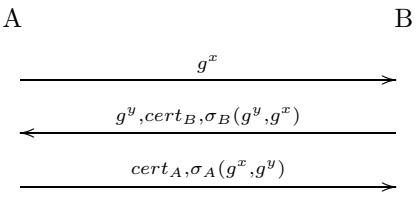
The key that A and B have in common is thus $K = g^{xy}$. However, there is no authentication in this process, and is susceptible to a woman-in-the-middle (Eve) attack:



Subsequently, E and A share $K = g^{xy}$, and E and B share $K' = g^{x'y'}$. E can intercept the key-establishment protocol and can impersonate both parties to the other. Since there is no authentication, a party may end up setting up a secret key with a complete stranger. We next examine Station-to-Station to handle this issue.

3.7 Station-to-Station (STS)

Station-to-Station is an approach to fixing Diffie-Hellman, which is comprised of a three-message exchange. It uses public key signatures to ensure identity.



The protocol starts like Diffie-Hellman with the first message; we assume that g and p are known to everyone. B then replies with his public key certificate along with g^y, and signed g^y and g^x. Likewise, A then replies with her public key certificate and signature on g^x and g^y.

However, the above protocol is broken: the first time B knows he is talking to A is at the third message. Middle-man M could send $cert_m, \sigma_m(g^x, g^y)$ to break the protocol. We fix this flaw by changing the second and third messages such that B sends using E_K. Since M does not know K, he cannot know E_K. Therefore B replies to A's first message with g^y, $cert_B$, $E_K(\sigma_B(g^y, g^x))$, and A replies with $cert_A$, $E_k(\sigma_A(g^x, g^y))$.

3.8 Identity Based Encryption (IBE)

Assume we want to send encrypted e-mail, but we only know the recipient's e-mail address. An interesting new idea that somewhat resembles Kerberos as it uses a trusted third party, Identity Based Encryption (by Boney and Franklin of Stanford) uses a party's e-mail address as its public key. For the corresponding secret key, IBE assumes the existence of a trusted global third party, CA which helps determine the secret key which corresponds.

With IBE, there are users, a "global sys-admin" that helps with computation, and global parameters G_1 and G_2 that are both public. There is also a $g \in G_1$ that is a public generator. In addition, the "global sys-admin" (or global CA) controls a Master Secret key $S = SK_{CA}$ which is not published, and corresponding public key $PK_{CA} = g^S = h$.

Given user A with name "A", her public key is $h_A = PK_A = hash("A")$. Therefore anyone who knows A's e-mail address can determine her public key. Her corresponding secret key is computed by the global CA: $SK_A = h_A^S$. Note: the global CA is highly trusted and knows everyone's SK.

Before we proceed, we need to introduce a "magic bilinear function" $f : G_1 \times G_1 \to G_2$ with properties such that

$$f(x^a, y^b) = f(x, y)^{ab} = f(x^{ab}, y) = f(x, y^{ab})$$

and

$$f(x^a, y) = f(x, y^a) = f(x, y)^a$$

which is a Weil pairing function on elliptic curves.

To send a message m encrypted via IBE, send g^r where r is random, and

$$m \oplus hash(f(h_a, h)^r)$$

To decrypt the message, we examine $f(h_A, h)^r$ and try to compute $f(SK_A, g^r)$ since

$$f(h_A, h)^r = f(h_A, g^S)^r = f(h_A^S, g)^r = f(SK_A, g^r)$$

and check if $f(h_A, h)^r$ is indeed equivalent to $f(SK_A, g^r)$.

Identity Based Encryption describes a public key infrastructure such that parties can securely communicate with each other only knowing the name of the recipient, and provides another way of key establishment.