Lecture 9

Public Key Management
Knowledge must be Reliable

Wholeness Statement

In public key cryptography, the public key is available to everybody, so there is no concern about somebody getting the key. The concern is whose public key it is. Certificates issued by trusted certification authorities are an attempt to solve this problem. SCI solves the problem of reliable knowledge by the discovery of the 7th state of consciousness, Unity Consciousness, where everything is known in terms of the Self.

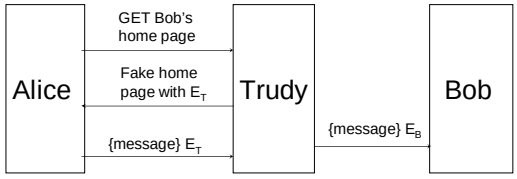
Overview

1. How do you know who a public key belongs to?
2. Use a certification authority (CA) that Alice and Bob both trust
3. A certificate associates an identity with a public key and is signed using the private key of the CA
4. PGP lets anyone be a CA
5. SSL is a protocol that uses the server's public key to establish a session key

Key Management

- Goal: bind identity to a key
  - Reason: we want be sure with whom we are communicating
- Cannot be done with classical cryptosystems because keys are shared
  - Required the protocols we talked about yesterday
- In public key cryptosystems, we bind identity to the public key
  - Public key is used to communicate with a specific entity or individual
- Incorrect binding means loss of secrecy
- We assume a principal is identified by a name

Man-in-the-middle Attack by Trudy



Trudy intercepts Alice's message and sends possibly a different message to Bob.

Why Certification Authorities (CAs)

- How do we get a specific person's public key?
- If Alice asks Bob for his over the network and Trudy intercepts the request, Trudy can send her public key to Alice; then Trudy can decipher anything Alice enciphers with it.
- This is demonstrated in the above diagram.
- Note that E_T is Trudy's public key and E_B is Bob's public key.
- Trudy could then send the message on to Bob by re-encrypting it with Bob's real public key or send a different message.
- The solution is to use a Certification Authority (CA)

Certificates

Definition 9-2. A *certificate* is a token that binds an identity to a cryptographic key.

- When Bob wants to communicate with Alice, he obtains Alice's certificate, C_{Alice} , from Cathy:
  $C_{Alice} = \{e_{Alice} \| \text{Alice} \| T\}d_{Cathy}$
- Bob decrypts Alice's certificate with Cathy's public key to obtain Alice's public key
- Now Bob can send Alice a message securely by encrypting it with Alice's public key
- For this to work, Bob must know Cathy's public key which represents a potential problem
- Two approaches: one approach eliminates the encryption of the certificate by Cathy, a second approach structures certificates into signature chains (see PGP below)

Certification Authority

Definition 9-3. A *certification authority* (CA) is an entity that issues certificates.

Digital Certificates

A CA is somebody that Alice and Bob trust. (CAs are the public key equivalent of the KDCs used by Kerberos in the last lecture).

Let e_{CA} be the public key of the CA and e_{Bob} be the public key of Bob. A certificate issued by CA for Bob would look something like this

Bob, Bob's address, e_{Bob}, issuer, serial#, expiration date ||
$\{Hash(Bob, Bob's address, e_{Bob}, issuer, serial\#, expiration date)\}d_{CA}$
where

1. The issuer is the name of the CA that signed this certificate
2. Whoever's public key is mentioned in the certificate is the subject, e.g., Bob.
3. serial# is the serial number of the certificate (in case it has to be revoked)
4. expiration date is the date that the certificate expires
5. Hash(...) is a crytographic checksum (hash) of the certificate information.
6. d_{CA} is the private key of the CA.

Properties of Certificates and CAs

1. A certificate cannot be used to send a session key as was done in Kerberos since the contents of the certificate are not encrypted. This makes sense because the public key is public.
2. A certificate binds an identity (e.g., Bob) to a cryptographic key (e.g., Bob's public key).
3. The contents of the certificate are hashed and then the hash is encrypted with the CA's private key. We trust the contents of the certificate only if we trust the CA and we have a trusted copy of the CA's public key.
4. A certificate is verified as follows:
   a. apply the same hashing algorithm that the CA used to the plaintext of the certificate; call the result X.
   b. decrypt the encrypted hash using the CA's public key to get Y
   c. If X == Y, then we feel confident that the certificate originated from the CA.

5. How do we get the public key of the CA? We can't go to the CA's web page to get it because Trudy could intercept the request like she did with Alice above. A solution that is often used is that a browser ships with the public keys of certain trusted CA's. You can see this list in Internet Explorer by pulling down the **Tools** menu, choose **Internet Options**, click the **Content** tab, and then the **Certificates** button. Then click the "**Trusted Root Certification Authorities**" tab. You can add and remove certificates from the list.
6. CA's are in the business to make money. It cost Ralph $250 to get a certificate from *http://www.geotrust.com/*; the certificate has to be renewed every year. Ralph sent them his domain name, public key, and credit card number; they sent a certificate after first verifying that he is who he says he is. For $250 they don't do much verification. They called and made a voiceprint of Ralph's voice.

7. A server can send a chain of certificates to a client. A chain is useful if Alice doesn't know the CA that signed the certificate containing Bob's public key. For example, Bob could send Alice the following chain of certificates. (For simplicity, the hash is not included)
   $\{Bob, e_{Bob}, CA1, ... \}d_{CA1}$
   $\{CA1, e_{CA1}, CA2, ... \}d_{CA2}$
   $\{CA2, e_{CA2}, CA3, ... \}d_{CA3}$
   - Suppose Alice doesn't know anything about CA1 who signed Bob's certificate above, but does trust CA3 because Alice has CA3's public key, e_{CA3}.
   - Alice can search the certificate chain looking for a certificate issued by CA3. She finds $\{CA2, e_{CA2}, CA3, ... \}d_{CA3}$, i.e., CA3 has signed CA2's certificate.
   - She uses e_{CA3} (which she obtained somehow, perhaps it came with her browser) to verify $\{CA2, e_{CA2}, CA3, ... \}d_{CA3}$
   - Now she can use e_{CA2} to verify $\{CA1, e_{CA1}, CA2, ... \}d_{CA2}$
   - Then she uses e_{CA1} to verify $\{Bob, e_{Bob}, CA1, ... \}d_{CA1}$
   - At this point she has verified e_{Bob} which she can now use to send an encrypted message to Bob.

Example

- If you have an account with amazon.com you can see a certificate chain.
- In Internet Explorer, proceed as follows. Log onto your account, click the lock icon at the top of the browser window, click "View certificates", then click the "Certification Path" tab.
- You can follow a similar procedure in the Firefox browser.

Alice and Bob revisited

- Returning to the problem we started out with, namely how can Alice get Bob's public key.
- If Bob's web server supports certificates, it will return Bob's certificate which contains

  Bob, Bob's address, e_{Bob}, issuer, serial#, expiration date || {Hash(Bob, Bob's address, e_{Bob}, issuer, serial#, expiration date)}d_{CA}
- Now Alice checks that the name in the certificate is "Bob" and that she trusts the CA who issued the certificate.
- To make sure that the certificate was not tampered with she uses the CA's trusted public key to verify the digital signature.
- She then retrieves Bob's public key from the certificate and uses it with confidence to communicate securely with Bob.

Main Point

1. A certificate binds an identity to a public key. In ignorance ones identity is bound to changing things such as a job or an award. In enlightenment, one experiences that the self is nothing other than the non-changing, eternal field of pure consciousness.

Example Use of a Certificate

Using Internet Explorer (or Firefox), try this experiment:

1. Open http://mumde.net/CS466 with the browser. Note that the usual URL begins with www, e.g. http://www.mumde.net/CS466.
2. mumde.net will get translated into the same IP address as www.mumde.net, namely 64.209.134.189. Use http://www.hcidata.info/host2ip.htm to verify this.
3. In spite of the fact that both domain names map to the same IP address, if you enter mumde.net instead of www.mumde.net, Internet Explorer will display the following error message.

Certificate Error Messages

If you try this in Internet Explorer, you will get the error message:

There is a problem with this website's security certificate.

The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

Certificate Error Messages

If you try this in Firefox, you will get the error message:

Security error: Domain name mismatch

You have attempted to establish a connection with "mumde.net". However, the security certificate presented belongs to "www.mumde.net". It is possible, though unlikely, that someone may be trying to intercept your communication with this web site.

If you suspect that the certificate shown does not belong to "mumde.net", please cancel the connection and notify the site administrator.

Why does this happen?

- The course web site uses the https protocol to communicate securely (see the discussion of SSL below)
- SSL uses the public key of the server in its protocol.
- This is indicated by the lock icon that is displayed at the top of the browser window in both Internet Explorer and Firefox.
- Clicking on the lock will display details of the certificate.
- If you do this, you will notice that the name on the certificate is www.mumde.net, not mumde.net
- Since the name of the domain you requested (mumde.net) is not the same as the name on the certificate (www.mumde.net), the browser displays the warning
- The browser just compares the names, it doesn't try to convert the names to IP addresses!!

- The certificate used in this experiment identifies a web server, not a person.
- You will not find Ralph's name anyplace in the certificate.
- One could also purchase a certificate that identified the owner of the web site and send that to a server, but the WWW usually does not require the client to have a certificate.

Advantages of CAs over KDCs

- The CA doesn't have to be online. It can be in a locked room, create a certificate and put it on a floppy disk. A user has to communicate with a KDC online (as described in lecture 8) to get a session key.
- Since a CA is not online, it can be simpler (economy of mechanism)
- There is no single point of failure for a CA. But if the KDC (Cathy) goes down, Alice and Bob cannot create a session key.
- Certificates are not security sensitive. All an attacker can do is delete certificates, he can't create bogus certificates because he doesn't have the private key of the CA.
- Since a compromised CA doesn't have a private key, it can't decipher conversations but a compromised KDC can (it has the keys that it shares with the users that trust it). That is, you have to trust a KDC more than a CA. All you give the CA is your public key.

Disadvantages of CAs

- How can a CA revoke a certificate?
  - The established method is to use a Certificate Revocation List (CRL).
  - But the client has to have a current copy of the CRL.
  - An X.509 CRL includes a list of serial numbers of unexpired revoked certificates and an issue time of the CRL.
- A certificate is valid if
  a. it has a valid CA signature
  b. has not expired
  c. has not been revoked
- Browsers check the signature, but do not always check for expired or revoked certificates

Disadvantages of CAs

Ways that Trudy can compromise the verification of a certificate

a. Steal the private key of the CA. She can then use it to sign any certificate. If a CA's private key gets stolen, every certificate that was signed by it should be revoked. This is a major disaster for the CA.
b. Trudy could work for the CA and issue a certificate without thoroughly checking out the identity of the owner of the certificate. Many, many security breaches are inside jobs.
c. Trudy could steal somebody's identity and get a certificate using that identity.
d. Trudy could take Bob's certificate and modify it in a way that the cryptographic checksum is the same as the checksum of the original certificate. The verification of the certificate will not detect this. A good cryptographic checksum algorithm will make this extremely difficult to do!
e. Trudy could add the public key of a bogus CA to the list of Trusted Root Certification Authorities in a browser and send a certificate signed by the bogus CA to the browser. This is particularly a problem in a computer lab like we have at M.U.M.

PGP

Public Key Infrastructure (PKI)

- A public key infrastructure consists of the components necessary to securely distribute public keys.

Should consist of

1. a repository for retrieving certificates
2. a method for revoking certificates
3. a method to evaluate a chain of certificates until one signed by a trusted CA (also called a trust anchor) is found.

Alternatives to PKI

- Exchange the public keys in person or assume that Trudy isn't listening.
- Assume that each entity knows its private key (non-trivial if using a public terminal, needs to be downloaded from somewhere)
- See

  http://www.schneier.com/paper-pki.pdf for another look at the risks of using the PKI.

How PGP does certificates

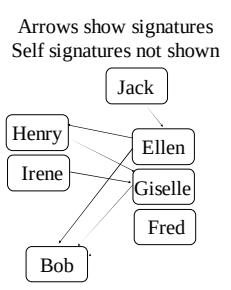
- The certificates we discussed so far are based on a hierarchical model where there are relatively few CAs who issue certificates.
- PGP uses a different model where each individual can act as a CA.
- CAs are arranged arbitrarily instead of in a tree structure
- PGP does not rely on commercial CAs who sell certificates.
- Relies on each individual's knowledge of the certifiers
- Here is an example.

Alice and Bob re-revisited (PGP)

- Alice and Bob want to communicate securely via PGP
- Alice obtains Bob's certificate (containing his public key) which is signed by Ellen, Fred, Giselle, and Bob who are all acting as CAs.
- Alice knows none of the signers so she gets Giselle's certificate which was signed by Henry, Irene, and Giselle.
- Alice knows Henry slightly so she obtains his certificate which is signed by Ellen and Henry and uses it to verify Giselle's certificate. But she notes that Henry's signature is at the casual trust level (She may have gotten it from a web page rather than personally from Henry) so she decides to look elsewhere.
- She obtains Ellen's certificate which is signed by Jack and Ellen. Jack is her husband and she uses his certificate (containing his public key) to verify Ellen's certificate.
- Finally she uses Ellen's certificate to verify Bob's and she now feels confident that she has Bob's correct public key.

Validating Bob's Certificate

- Alice needs to validate Bob's certificate
  - Does not know Fred, Giselle, or Ellen
- Alice gets Giselle's certificate
  - Knows Henry slightly, but his signature is at "casual" level of trust
- Alice gets Ellen's certificate
  - Knows Jack, so uses his cert to validate Ellen's, then hers to validate Bob's



Main Point

2. In the PKI trust model used by PGP, everybody can create a certificate. You trust the certificates created by people you know and trust. The individual plays an important role in this model. SCI teaches that the individual is the unit of world peace.

SSL

Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

- SSL/TLS allow two parties to authenticate and establish a session key that is used during the rest of the session.
- SSL/TLS runs in user space (unlike TCP/IP which is part of the OS)
- SSL v3 was developed by Netscape
- TLS was developed by the IETF (Internet Engineering Task Force)
- SSL and TLS are similar but not compatible.
- SSL is the dominant player

SSL Records

Four types

1. user data
2. handshake messages
3. alerts (error messages or notification that connection is being closed)
4. change cipher spec

Alice and Bob re-re-revisited (SSL)

Alice-->Bob: Alice here || supported ciphers || R_{Alice}
- supported ciphers is a list of ciphers that Alice knows how to use;
- R_{Alice} is a random number

Bob-->Alice: certificate || selected cipher || R_{Bob}
- certificate contains Bob's public key, Alice must verify the certificate;
- selected cipher is one of the ciphers that Alice sent in the first message;
- R_{Bob} is a random number

Alice-->Bob: $\{S\}e_{Bob}$ || {keyed hash of handshake messages}
- S is the pre-master secret generated by Alice. It is used to compute the master secret K as follows:
- $K = f(S, R_{Alice}, R_{Bob})$
- The keyed hash of the handshake messages allows Bob to verify that Alice got the messages he sent.

Bob-->Alice: {keyed hash of handshake message}
- Lets Alice know that Bob got the handshake messages that she sent

Alice-->Bob: data protected with keys derived from K
- Use the master secret K to compute the keys used to encipher the data

Bob-->Alice: data protected with keys derived from K

The SSL Protocol

- In the above Alice has authenticated Bob but Bob has not authenticated Alice.
- SSL/TLS allows this to happen. Otherwise Alice would have to send her certificate to Bob and Bob would issue a challenge using Alice's public key.
- The browser doesn't do this since a customer would have to have a certificate to use an SSL site.
- What usually happens is the customer sends username and password protected by the keys derived from K.

Main Point

3. The secure socket layer provides secure communication using public key and conventional cryptography. It runs in user space on top of the transport layer (TCP/IP). Every discipline discovers that life is structured in layers.

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

1. The little gold lock displayed in Internet Explorer means that the browser is communicating securely with the web server.

2. SSL uses a certificate to get the public key of the server.

3. Transcendental Consciousness is a field of infinite correlation where every part of creation is interconnected and communicates through a frictionless flow of information; the laws of nature are everywhere.

4. Wholeness moving within itself: in Unity Consciousness one experiences and lives infinite correlation because everything is a manifestation of one's own Self.