CIS 4360
Secure Computer Systems
PKI and Kerberos

Professor Qiang Zeng
Spring 2017

TEMPLE UNIVERSITY
Previous Class

• Important Applications of Crypto
  – User Authentication
    • verify the identity based on “something you know”
    • Sending the password over a secure channel
    • Challenge-response Hash(c || PA) over an insecure channel
  – Data Integrity
    • Verify whether the data has been tampered with
    • MAC or Digital Signature
  – Confidentiality
• Diffie-Hellman Key Agreement
  – For Forward Secrecy
DH Key Agreement

Alice

\[ A = g^a \mod p \]

\[ K = B^a \mod p \]

Bob

\[ B = g^b \mod p \]

\[ K = A^b \mod p = (g^a \mod p)^b \mod p = g^{ab} \mod p = (g^b \mod p)^a \mod p = B^a \mod p \]
Key Sizes for the Diffie-Hellman Cipher

- $A = g^a \mod p$
- $p$ should be $\geq 2048$ bits, similar in RSA's $n$
- $a$ and $b$ should be $\geq 224$ bits to achieve 112 bits key strength
- $g$ does not matter much, e.g., $g$ can be 2
How to achieve authentication and data integrity of communication over an insecure channel?

Two ways:
(1) Alice sends the MAC along with the message, and the MAC is created using the key shared between Alice and Bob
(2) Alice signs the message and sends the signature along with the message
Why is Diffie-Hellman Key Agreement subject to the man-in-the-middle Attack?
Outline

• PKI
  – Digital Certificate
  – Certificate Authority
  – Verifying Certificates and Chain of Trust
  – Revoking Certificates

• Kerberos
SSL Handshake: a simple version that does not provide forward secrecy

Alice and Bob are nonce numbers; \( R_{Alice} \) and \( R_{Bob} \). Alice selects a key (formally called the pre-master secret) and encrypts it using Bob’s public key; \( K \) is the master secret, which is used to derive the shared secrets/session keys for encryption and MAC for the session.
SSL Handshake with forward secrecy

1. Generate ephemeral DH key pair
2. Create signature over DH public parameters using RSA private key
3. Generate Pre-Master secret using Server's ephemeral DH private param and Client's ephemeral DH public param
4. Generate session keys from Pre-Master secret and **discard ephemeral DH key pair**
5. Generate session keys from Pre-Master secret and **discard ephemeral DH key pair**
Whose Public Key?

- A Public Key is just a stream of bits
- How do you know whose public key it is?
- When one shows you her public key and claims she is gmail.com, how to verify whether it is true or a lie?
- When logging on pnc.com, how do you know whom you are providing the password to?
- What if your private key is leaked and you want to change the corresponding public key?
Digital Certificate

• A Digital Certificate (also called Public Key Certificate) is a file proving that a public key belongs to some entity

• Public-Key Infrastructure (PKI) is the infrastructure that manages (issues, uses, revokes) digital certificates
  – E-commerce
  – Online banking
  – Confidential emails
  – …
What is inside a certificate?

• Public key
• Subject: the entity that owns the public key
  – E.g., gmail.com, pnc.com, citi.com
• Issuer: the entity that issued the certificate
• Signature: the issuer signs the certificate and generates the signature
• Other info.: effective time, etc.
X.509 Digital Certificates

- Version Number
- Serial Number
- Signature Algorithm ID
- Issuer Name
- Validity period
  - Not Before
  - Not After
- Subject name
- Subject Public Key Info
  - Public Key Algorithm
  - Subject Public Key
- Extensions (optional)
  - ...
- Certificate Signature Algorithm
- Certificate Signature
qiang@Qiangs-MacBook-Air:~$ openssl s_client -connect google.com:443 2>/dev/null | openssl x509 -in /dev/stdin -noout -text

Certificate:

Data:
  Version: 3 (0x2)
  Serial Number:
    56:75:bc:e5:ef:d8:ad:69
  Signature Algorithm: sha256WithRSAEncryption
  Issuer: C=US, O=Google Inc, CN=Google Internet Authority G2
  Validity
    Not After : Mar  9 13:35:00 2017 GMT
  Subject: C=US, ST=California, L=Mountain View, O=Google Inc, CN=*.google.com
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public Key: (2048 bit)
      Modulus (2048 bit):
      Exponent: 65537 (0x10001)

... Signature Algorithm: sha256WithRSAEncryption
Who issues certificates?

- Certificate Authority (CA)
- A Root CA issues certificates to intermediate CAs, which further issue certificates to others
  - It is much more flexible than relying on a Root CA to issue certificate
  - E.g., geotrust.com is a Root CA, which issues a certificate to Google, which further issues certificates to its domains such as google.com, mail.google.com
Who does a CA consult?

• A CA usually sets up a functional department of RA
• A Registration Authority (RA) is an authority that accepts, processes and verifies user requests for a digital certificate by checking the requester’s identity
• Once the requester’s identity is verified, the RA forwards the request to the certification authority to sign and issue a digital certificate
Tree of Certificates (per Root CA)
Chain of Trust

• A Digital Certificate is verified using a **Chain of Trust**, i.e., from the bottom up along the tree of certificates until reaching the root
  – The certificate of a Root CA is called the **Root Certificate**
  – The **Trust Anchor** is the Root CA

• Root Certificates are pre-installed in your browsers. Your browser company *assumes* that you trust the corresponding Root CAs
  – Top Root CAs: Comodo, VeriSign (Symante), GoDaddy
  – You do not really have much choice; essentially, you trust Chrome, Safari, IE
How to verify a digital certificate? -- Example

Certificates are verified from the bottom up

- (1) Verify the certificate for www.pnc.com
- (2) Verify the certificate for “Symantec Class 3 EV SSL CA – G3”
- (3) The browser notices that the certificate in step (2) was signed by a root CA and thus accepts it
What if you have leaked the private key?

- A certificate may be revoked
  - Private key is stolen
  - Certificate owner is found illegal
  - The organization is deleted
- A **Certificate Revocation List (CRL)** is a list of the serial numbers of revoked certificates
  - Created and signed by a CA
- It means that whenever you verify a certificate, you need to consult the current CRLs
  - Complicated PKI; one of its main disadvantages
A revoked certificate can be removed from the CRL once the current time reaches the expiration date. Without expiration dates, the CRLs would grow indefinitely.

Another reason is that CAs love money.
Kerberos

- A client sends a request for a ticket to the Key Distribution Center (KDC)
- The KDC creates a *ticket-granting ticket* (TGT) for the client, encrypts it using the client's password as the key, and sends the encrypted TGT back to the client
- The TGT, which expires at a specified time, permits the client to obtain additional tickets for accessing services
Kerberos supports Single Sign-On (SSO)

*TGT: Ticket-Granting Ticket*
Main Limitation of Kerberos

• Single point of failure: it requires that the KDC be online. If the KDC fails, the whole authentication system fails.
Kerberos vs PKI

• Both PKI and Kerberos can be used for authentication; But PKI is mainly used to authenticate a service, while Kerberos is to authenticate both services and users

• PKI mainly builds on asymmetric cryptography, while Kerberos mainly builds on symmetric cryptography

• PKI is used over the Internet, while Kerberos is typically used within a single organization
Writing Assignments

• Is PKI subject to Single Point of Failure?
• What is the big pain of PKI?