Public Key Infrastructure (PKI)

- Asymmetric cryptography VS symmetric cryptography
- Digital Signature
- Certificate
- Certification Authority (CA)
- Secure Sockets Layer (SSL) protocol

Asymmetric cryptography vs symmetric cryptography

Symmetric cryptography

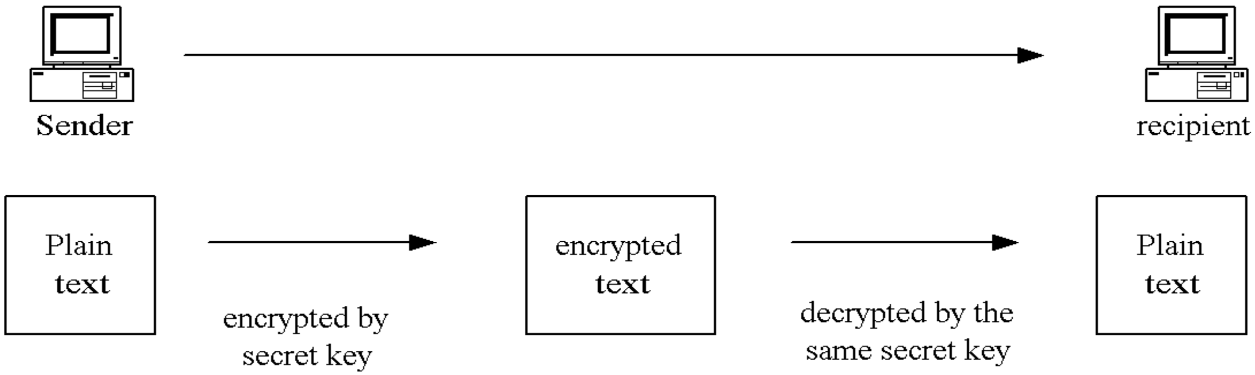
- traditional form of cryptography
- a single key is used for both encryption and decryption
- the sender and receiver share a key

Asymmetric cryptography (public key cryptography)

- uses two mathematically related keys
- a message encrypted by one key can only be decrypted by the other key
- receive secure messages by simply publishing one key (the public key) and keeping the other secret (the private key)

Asymmetric cryptography vs symmetric cryptography

Symmetric cryptography (secret-key)



Sender

recipient

Plain text

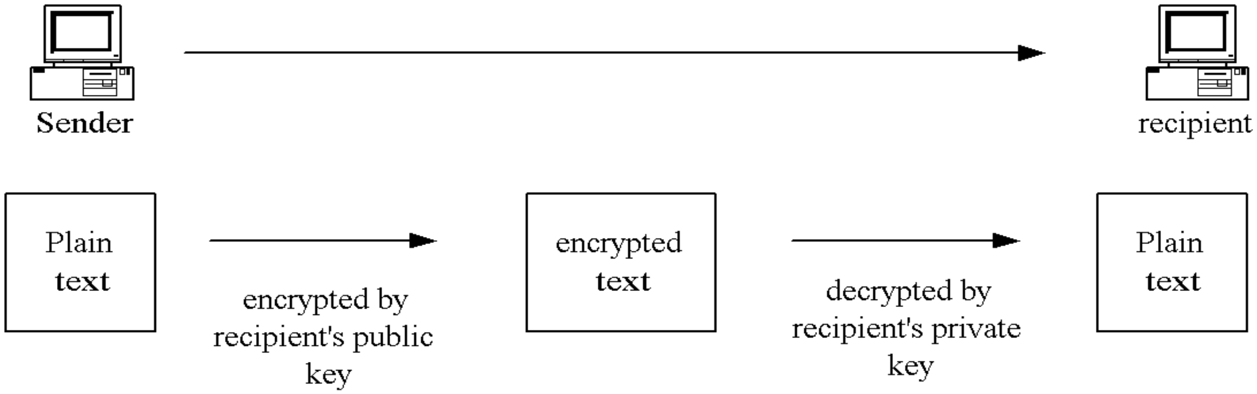
encrypted by secret key

encrypted text

decrypted by the same secret key

Plain text

Asymmetric cryptography (public-key)



Sender

recipient

Plain text

encrypted by recipient's public key

encrypted text

decrypted by recipient's private key

Plain text

RSA (Rivest-Shamir-Adelman) Implementation

1. Choose two large primes: p and q; n=pq
2. Choose e < n and relatively prime to (p-1)(q-1)
3. Find d such that (ed-1) is divisible by (p-1)(q-1)
4. The public key is the pair (n, e); the private key is (n, d)

It is currently difficult to obtain the private key d from the public key (n, e). However if one could factor n into p and q, then one could obtain the private key d. Thus the security of RSA is based on the assumption that factoring is difficult.

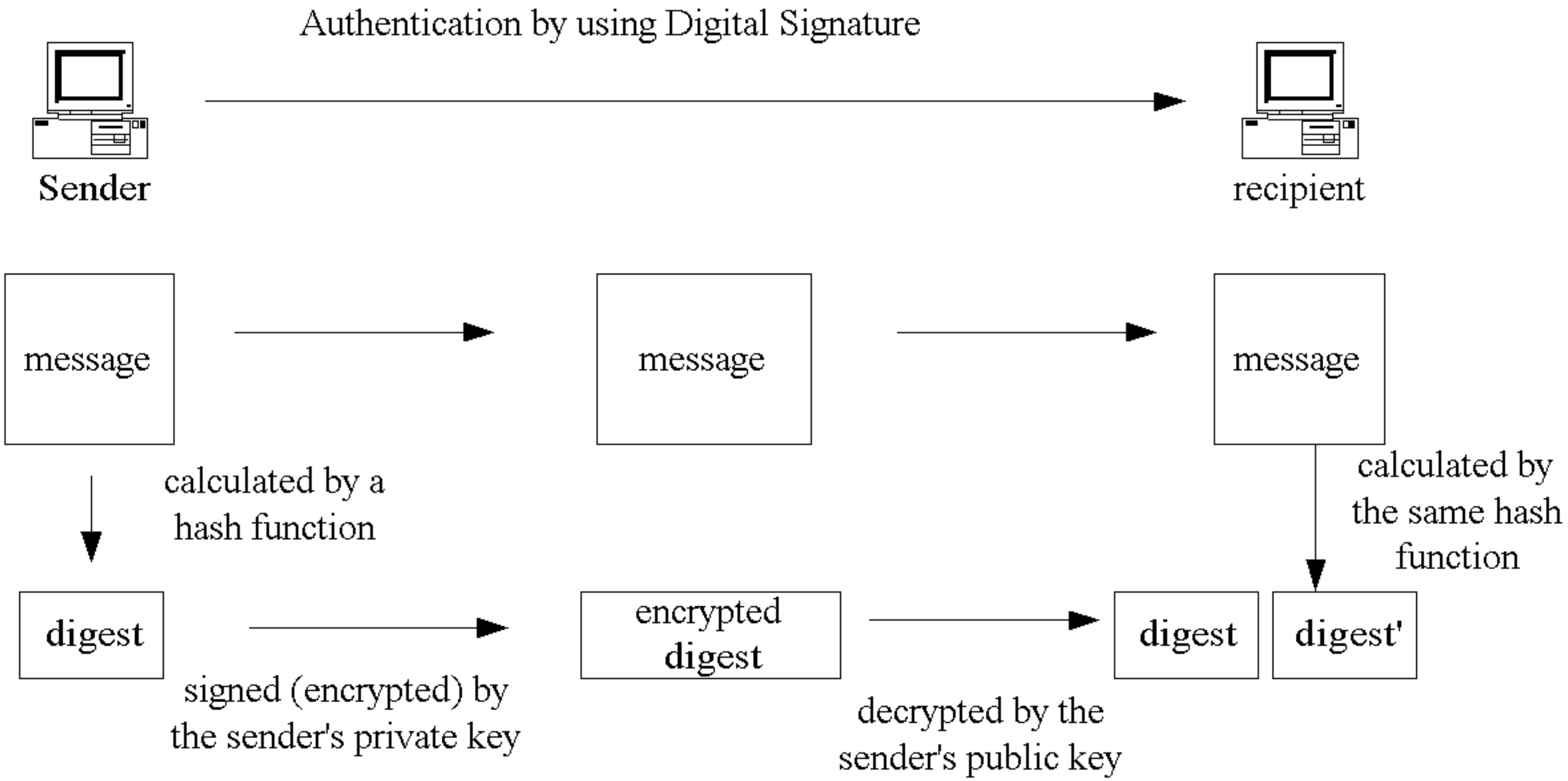
RSA Encryption

1. Suppose A wants to send a message m to B. A creates the ciphertext c by exponentiating: $c = m^e \bmod n$, where e and n are B's public key. A sends c to B.
2. To decrypt, B also exponentiates: $m = c^d \bmod n$; the relationship between e and d ensures that B correctly recovers m. Since only B knows d, only B can decrypt this message.
3. That is A is using B's public key (n, e) to encrypt her message and B use his private key (n, d) to decrypt the ciphertext

Digital Signature

- The digital signature of a document is a piece of information based on both the document and the signer's private key
- It is created by encrypting a digest of the message, and other information (such as a sequence number) with the sender's private key
- Anyone may decrypt the signature using the public key and then compare the digest of the message
- If the digests are matched, the signature is only good for that message. It also ensures the integrity of the message since no one can change the digest and still sign it

Digital Signature



Authentication by using Digital Signature

Sender

recipient

message

message

message

calculated by a hash function

calculated by the same hash function

digest

encrypted digest

digest

digest'

signed (encrypted) by the sender's private key

decrypted by the sender's public key

If the digest and digest' are identical, the digital signature is good.

RSA Digital Signature

1. Suppose A wants to send a message to B in such a way that B is assured the message is both authentic, has not been tampered with, and from A. A creates a digital signature s by exponentiating: $s = m^{d} \bmod n$, where m is the message digest; d and n are A's private key. A sends the message and s to B.
2. To verify the signature, B exponentiates and checks that the message digest m is recovered: $m = s^{e}$ mod n, where e and n are A's public key.
3. Hence, the digital signature is created by encrypting the message digest with one RSA private key.

A public-key infrastructure (PKI) consists of protocols, services, and standards supporting applications of public-key cryptography.

PKI consists of:

- Certificates
- Certificate Authorities (CA)
- Certificate Revocation Lists (CRL)
- Repositories to store public-keys for people

Certificates

Certificates are digital documents attesting to the binding of a public key to an individual or other entity. They allow verification of the claim that a specific public key does in fact belong to a specific individual. It also includes identification information as to who the own of the certificate is, as well as a signature by a CA validating that the data hasn't been forged. Certificates help prevent someone from using a phony key to impersonate someone else. In their simplest form, certificates contain a public key and a name. As commonly used, a certificate also contains an expiration date, the name of the certifying authority that issued the certificate, a serial number, and perhaps other information.

Certificate Authorities (CA)

Certificates are issued by a Certificate Authority, who usually will sign the certificate as well as provide some revocation facilities.

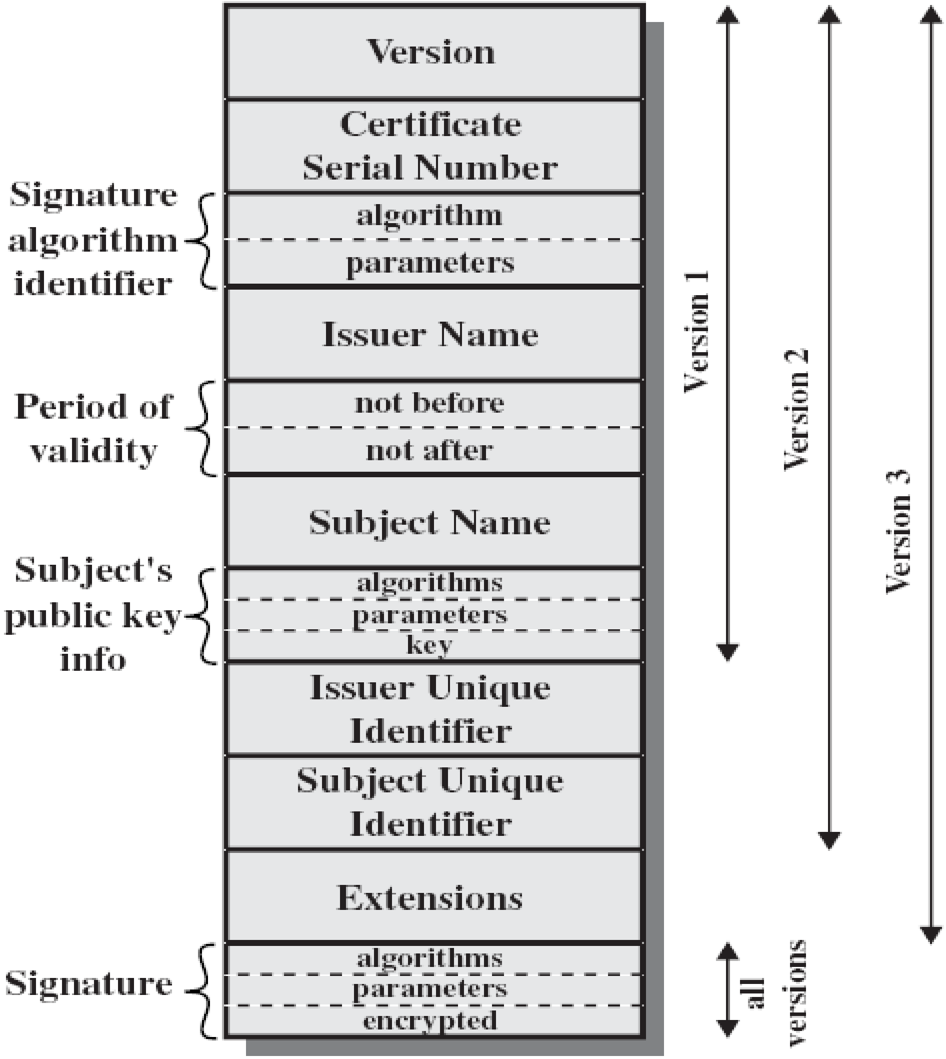
Certificate Revocation Lists (CRLs)

If the private-key is compromised (i.e. inadvertently made public), then the certificate containing that key needs to be "revoked". That essentially means the CA who assigned the certificate posts the certificate on its website. This allows people to publicly check this fact.

Repositories (e.g. LDAP directories)

So that public-keys for people can be found.

X.509 Certificate



Version

Certificate Serial Number

Signature algorithm identifier: algorithm, parameters

Issuer Name

Period of validity: not before, not after

Subject Name

Subject's public key info: algorithms, parameters, key

Issuer Unique Identifier

Subject Unique Identifier

Extensions

Signature: algorithms, parameters, encrypted

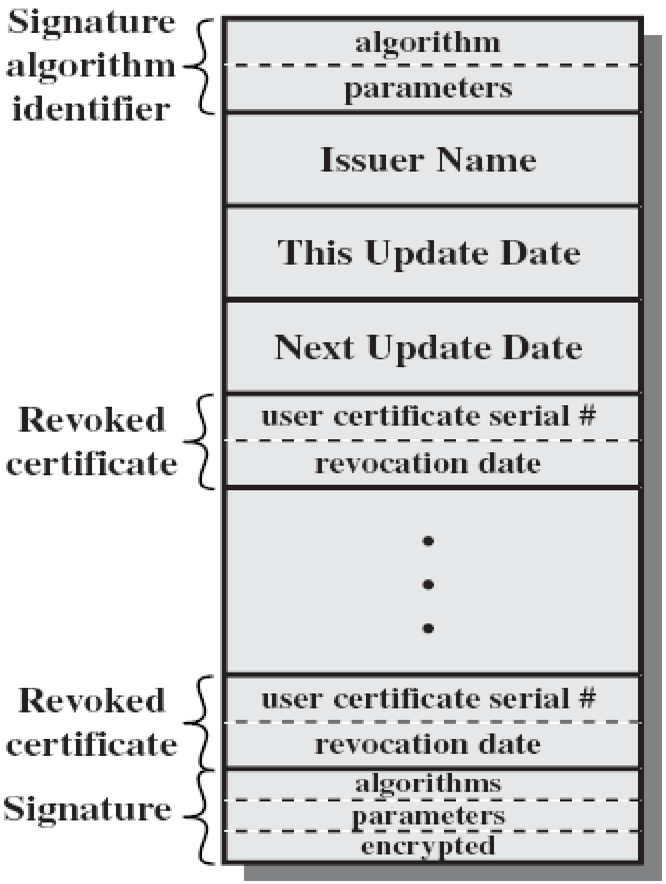
Version 1

Version 2

Version 3

all versions

(a) X.509 Certificate



Signature algorithm identifier: algorithm, parameters

Issuer Name

This Update Date

Next Update Date

Revoked certificate: user certificate serial #, revocation date

•
•
•

Revoked certificate: user certificate serial #, revocation date

Signature: algorithms, parameters, encrypted

(b) Certificate Revocation List

Sample X.509 certificates

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=HK, ST=Hong Kong, L=Hong Kong, O=CUHK, OU=SHLAM at NTEC, CN=SHLAM CA/emailAddress=shlam@ie.cuhk.edu.hk
        Validity
            Not Before: Feb 28 06:24:14 2006 GMT
            Not After : Feb 28 06:24:14 2007 GMT
        Subject: C=HK, ST=Hong Kong, L=Hong Kong, O=CUHK, OU=NTEC, CN=www.shlam.hkntec.net/emailAddress=shlam@ie.cuhk.edu.hk
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                Modulus (2048 bit):
                    00:c0:c3:0b:5f:ed:1b:80:95:91:22:0e:77:95:9c:
                    3f:58:44:fb:7f:ed:b2:0f:40:f3:1e:32:aa:9b:f1:
                    …………
                    6f:58:c7:fb:c6:2c:5f:63:a1:32:f1:43:25:a7:ad:
                    a8:11:4c:18:cf:14:2c:22:92:23:cf:06:a6:cc:ea:
                    6f:fb
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Authority Key Identifier:
                keyid:9A:52:5C:73:6B:02:08:A9:B3:3E:EF:55:24:C1:29:39:44:F9:85:B1

            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication, Microsoft Server Gated Crypto, Netscape Server Gated Crypto
            X509v3 Basic Constraints: critical
                CA:FALSE
    Signature Algorithm: md5WithRSAEncryption
        35:c6:52:df:8a:a6:f4:b7:52:da:b6:c3:73:c9:38:50:93:89:
        4c:a9:5b:a8:15:5e:47:87:40:ed:9e:4c:75:f5:03:9c:86:de:
       ………
        94:57:a8:a6:9d:ef:da:2d:45:0d:81:85:06:e2:95:d7:84:8a:
        e5:12:d4:67:20:b4:ee:ca:50:89:b0:d4:07:54:c7:22:8b:2d:
        29:a5
```

Rogue CA certificate with MD5 Hash

- A weakness in the MD5 cryptographic hash function that allows the construction of different messages with the same MD5 hash. This is known as an MD5 "collision.
- As a proof of concept, a rogue Certification Authority (CA) certificate has been created in 2008
- Now most CAs are using SHA1 to sign their certificates.

SSL

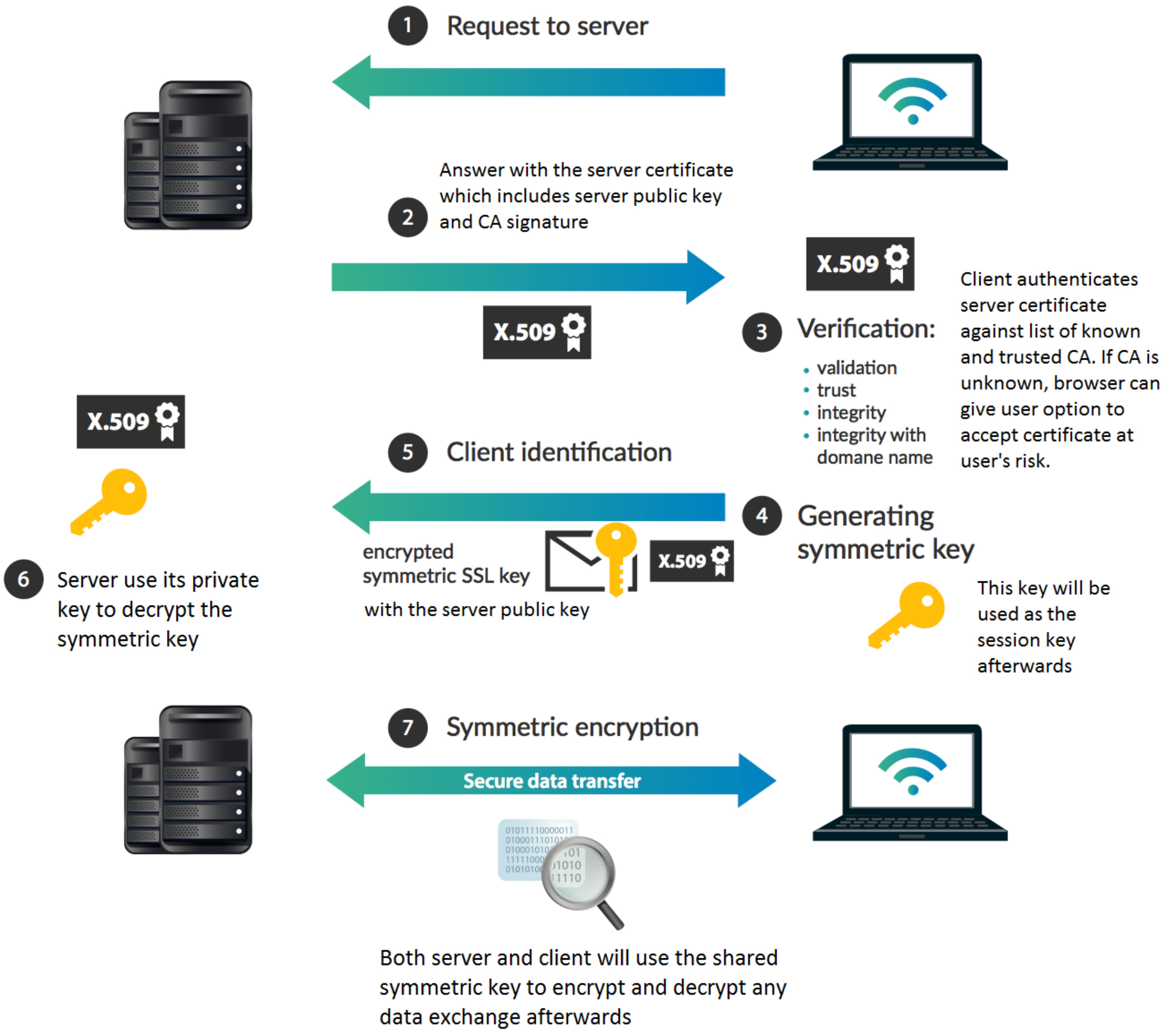
The Secure Sockets Layer (SSL) protocol, originally developed by Netscape, has become the universal standard on the Web for authenticating Web sites to Web browser users, and for encrypting communications between browser users and Web servers. Because SSL is built into all major browsers and Web servers, simply installing a digital certificate, or Server ID, enables SSL capabilities.

How SSL works

1. A customer contacts your site and accesses a secured URL: a page secured by a Server ID (indicated by a URL that begins with "https:" instead of just "http:" or by a message from the browser).
2. Your server responds, automatically sending the customer your site's digital certificate, which authenticates your site.
3. Your customer's Web browser generates a unique "session key" to encrypt all communications with the site.
4. The user's browser encrypts the session key itself with the site's public key so only the site can read the session key.
5. A secure session is now established. It all takes only seconds and requires no action by the user. Depending on the browser, the user may see a key icon becoming whole or a padlock closing, indicating that the session is secure.

Ref: SSL, TLS, HTTPS Explained Video

How SSL works



1 Request to server

2 Answer with the server certificate which includes server public key and CA signature

3 Verification:

- validation
- trust
- integrity
- integrity with domane name

Client authenticates server certificate against list of known and trusted CA. If CA is unknown, browser can give user option to accept certificate at user's risk.

4 Generating symmetric key

This key will be used as the session key afterwards

5 Client identification

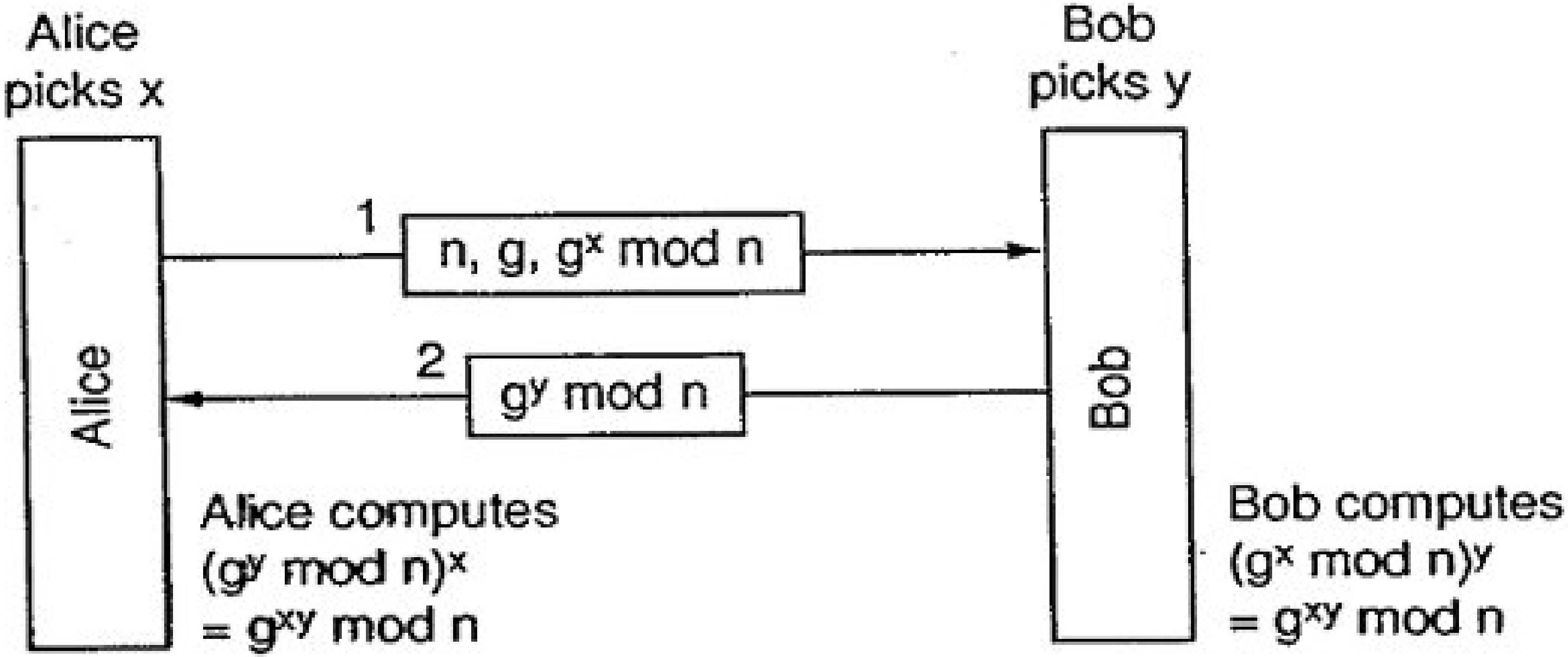
encrypted symmetric SSL key with the server public key

6 Server use its private key to decrypt the symmetric key

7 Symmetric encryption

Both server and client will use the shared symmetric key to encrypt and decrypt any data exchange afterwards

An example of Session Key exchange algorithm



An example of Session Key exchange algorithm (2)

1. A & B choose a pair of 2 prime numbers (n,g) from a list.
2. A picks a random number x and keep it secret; while B picks y and keep it secret
3. A send (n, g, g^x mod n) to B, B send back (g^y mod n)
4. A & B compute and get the same g^{xy} mod n
5. g^{xy} mod n is the shared key (one-time session key) for future communication

- Assume Alice pick n=47, g=3 and x=8; Bob pick y=10. The session key between A and B will be 4

Now you understand the basic principle of PKI, you may try to

- Set up a Secure Web server
- Build your CA Server
- Sign a server cert and user cert
- Set up Client Authentication and Access Control