Computer Security and Cryptography

CS381

来学嘉
计算机科学与工程系 电院3-423室
34205440 1356 4100825 laix@sjtu.edu.cn

2016-05

Organization

- Week 1 to week 16 (2016-02-24 to 2016-06-08)
- 东上院502
- Monday 3-4节; week 9-16
- Wednesday 3-4节; week 1-16
- lecture 10 + exercise 40 + random tests 40 + other 10
- Ask questions **in** class – counted as points
- Turn ON your mobile phone (after lecture)
- Slides and papers:
  - http://202.120.38.185/CS381
    - **computer-security**
  - http://202.120.38.185/references
- TA: '薛伟佳' xue_wei_jia@163.com, '黄格仕' <huang.ge.shi@foxmail.com>
- Send homework to: laix@sjtu.edu.cn and to TAs

Rule: do not disturb others!

2

Contents

- Introduction -- What is security?
- Cryptography
  - Classical ciphers
  - Today's ciphers
  - Public-key cryptography
  - Hash functions/MAC
  - Authentication protocols
- Applications
  - Digital certificates
  - Secure email
  - Internet security, e-banking

Network security
- SSL
- IPSEC
- Firewall
- VPN

Computer security
- Access control
- Malware
- DDos
- Intrusion

Examples
- Bitcoin
- Hardware
- Wireless

3

Why SSL?

- User wants buy a article from online shop, and pays with his credit card.
- Necessary:
  - Confidentiality of data (card number)
  - Authenticity of shop (no fraud)
- Wish:
  - Authenticity of user (card provides implicit authentication)
  - Non-repudiation of transaction (form sign)

4

SSL / TLS protocol

- SSL - Secure Sockets Layer
- TLS - Transport Layer Security
- A protocol for Transport layer security service, operates between TCP and applications
- The intension was to ensure e-commerce (encrypt credit card number)
- Netscape designed and built SSLv2 in 1994, TLS published in January 1999 as RFC 2246
- Independent of application layer
- support for negotiated encryption techniques.
  - easy to add new techniques.
- can switch encryption algorithms in the middle of a session.
- Many secure internet applications are built on the SSL

5

7 layers in OSI

OSI model
7. Application layer
NNTP ·SIP ·SSI ·DNS ·FTP ·Gopher ·HTTP ·NFS ·NTP ·SMPP ·SMTP ·SNMP ·Telnet ·DHCP ·Netconf ·RTP ·SPDY ·(more)
6. Presentation layer
MIME ·XDR ·**TLS** ·**SSL**
5. Session layer
Named pipe ·NetBIOS ·SAP ·PPTP ·SOCKS
4. Transport layer
TCP ·UDP ·SCTP ·DCCP ·SPX
3. Network layer
IP (IPv4, IPv6) ·ICMP ·**IPsec** ·IGMP ·IPX ·AppleTalk
2. Data link layer
ATM ·SDLC ·HDLC ·ARP ·CSLIP ·SLIP ·GFP ·PLIP ·IEEE 802.2 ·LLC ·L2TP ·IEEE 802.3 ·Frame Relay ·ITU-T G.hn DLL ·PPP ·X.25 ·Network switch
1. Physical layer
EIA/TIA-232 ·EIA/TIA-449 ·ITU-T V-Series ·I.430 ·I.431 ·POTS ·PDH ·SONET/SDH ·PON ·OTN ·DSL ·IEEE 802.3 ·IEEE 802.11 ·IEEE 802.15 ·IEEE 802.16 ·IEEE 1394 ·ITU-T G.hn PHY ·USB ·Bluetooth ·Hubs

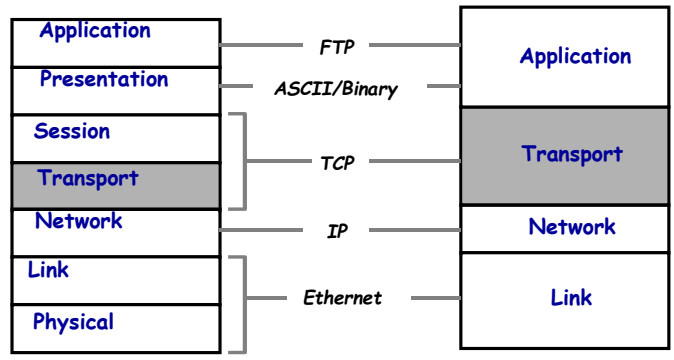
6

OSI vs. TCP/IP Stack

Layering: FTP Example

Provide functionalities for upper layer

use the functionalities of lower layer



Open Systems Interconnection (OSI) model ISO 7498 , ITU-T X.200

Security - OSI Layer

Application layer	ssh, S/MIME, PGP, https
Transport layer (TCP)	SSL, TLS, WTLS
Network layer (IP)	IPsec
Data Link layer	CHAP, PPTP, L2TP, WEP (WLAN), A5 (GSM), Bluetooth
Physical layer	Scrambling, Hopping, Quantum Communications

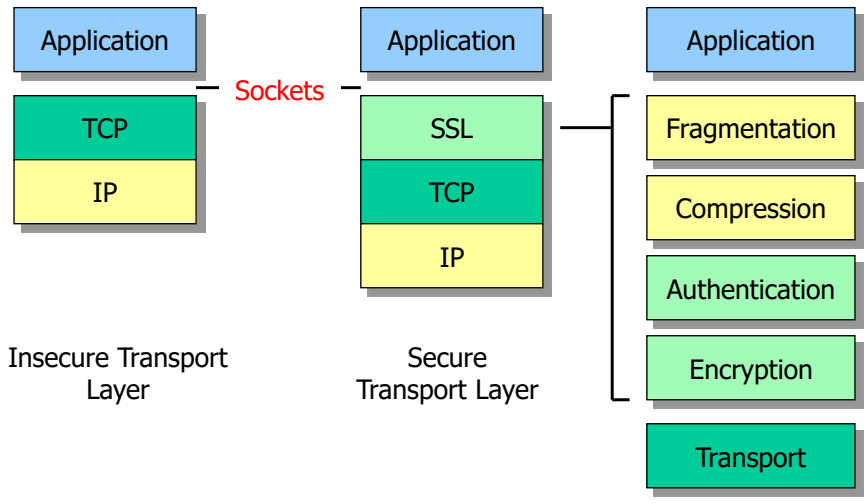
8

Protocols in TLS /SSL

- The primary goal of the SSL/TLS Protocol is to provide privacy and data integrity between two communicating applications (RFC 2246)
- Authentication (optional) by using public-key certificates.
- two main layers:
  - the TLS Handshake Protocol – key setup
  - the TLS Record Protocol -communication

9

SSL/TLS Protocol Layers



10

SSL Architecture

- **SSL session**
  - an association between client & server
  - created by the Handshake Protocol
  - define a set of cryptographic parameters
    - essentially, the master secret
  - shared by multiple SSL connections
- **SSL connection**
  - a transient, peer-to-peer communications link, typically a TCP connection
  - associated with a SSL session

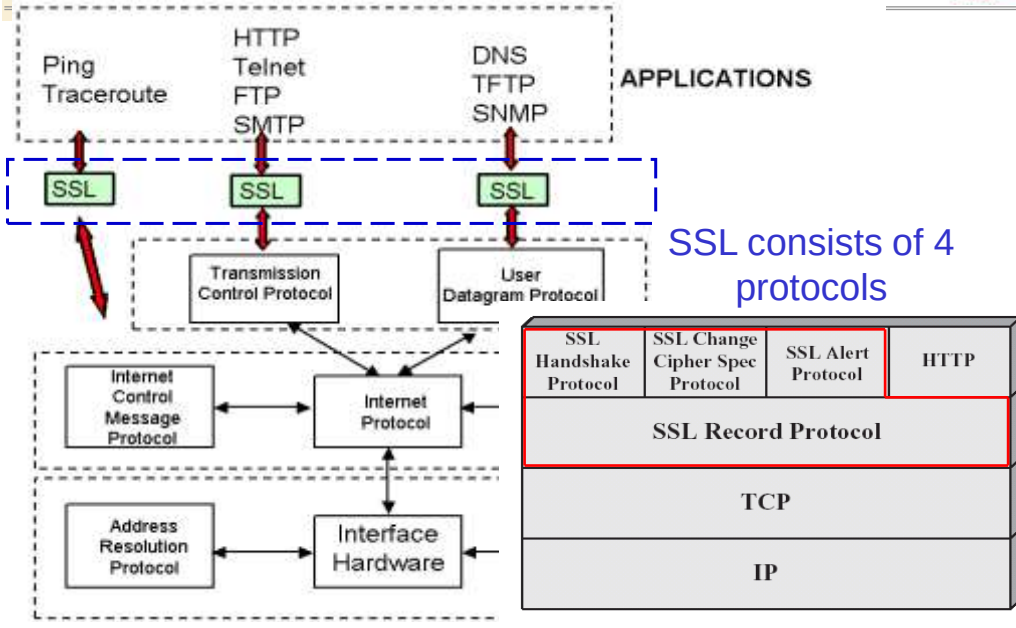
11

TLS: client protocols

- 2 layers:
  - Handshake protocol: establish security
  - Record protocol: use security
- 3 control protocols in TLS
  - Handshake: crypto setup
  - Alert: errors and shutdown
  - Change Cipher
- Higher level Applications over SSL/TLS:
  - HTTP, POP, IMAP, SMTP, VPN,…
  - application messages have lower priority

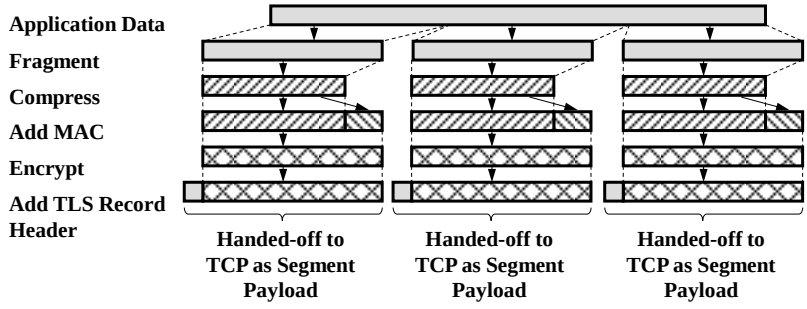
12

SSL subprotocols



SSL consists of 4 protocols

SSL record protocol



- TLS Record Protocol is used for encapsulation of higher level protocols
- Sender's record protocol takes messages to be transmitted, fragments them into blocks of 2^{14} bytes, compresses, applies an HMAC, encrypts, and sends
  - A record can contain multiple messages, the usual crypto components
- Receiver's record protocol receives, decrypts, verifies, decompresses, and reassembles

SSL /TLS Record Protocol

provided security services:

- **confidentiality**
  - using symmetric encryption with a shared secret key defined by Handshake Protocol
  - RC2-40, RC4-40, DES-40, DES, 3DES, IDEA, Fortezza, RC4-128, AES
  - message is compressed before encryption
  - The Record Protocol can be used without encryption.
- **message integrity**
  - using a MAC with shared secret key
  - similar to HMAC but with different padding (MD5, SHA)
  - The Record Protocol can operate without a MAC only when another protocol is using the Record Protocol as a transport for negotiating security parameters

15

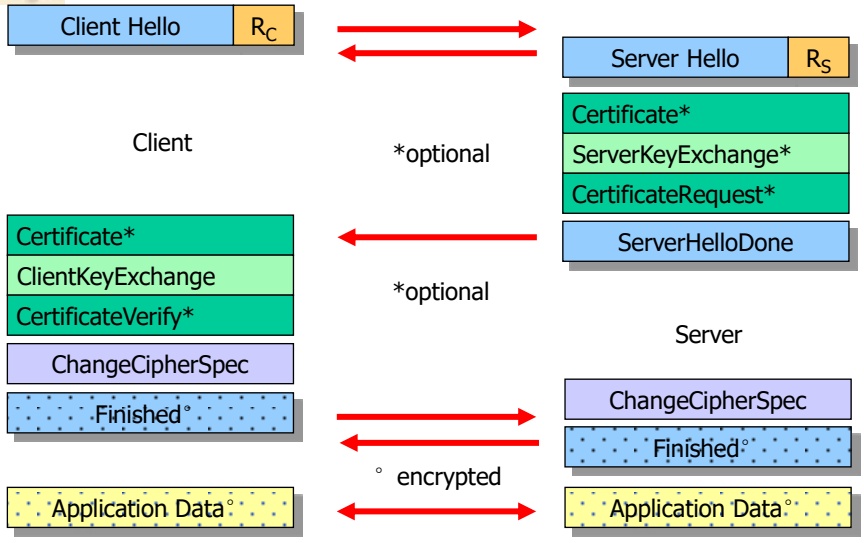
SSL/TLS Handshake Protocol

- TLS Handshake Protocol
  - Allow server and client to authenticate each other
    - optional, but generally require server to authenticate to client
  - negotiate an encryption algorithm
  - exchange keys before the application protocol starts.

- contains a series of messages in phases
  - Establish Security Capabilities
  - Server Authentication and Key Exchange
  - Client Authentication and Key Exchange
  - Finish

16

The SSL/TLS Handshake Protocol



17

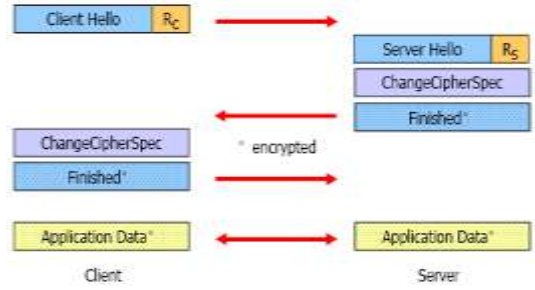
SSL Handshake messages

message	parameters
hello_request	Null
client_hello	Version,; Random numbers; session Id; cipher parameters; compression
server_hello	
certificate	X.509 v3 certificates
server_key_exchange	parameters; signature
certificate_request	type，CAs
server_done	Null
certificate_verify	signature
client_key_exchange	parameters; signature
finished	Hash value

TLS Handshake: resume

An abbreviated protocol -- reuses sessions

- E.g. HTTPS persistent connection



- redo cipher (skip certificates, key exchange)
- application layer must check the outcome
- Abort if the negotiated crypto is too weak
- Cipher suite changes / re-initializations
  - Whenever the application asks
  - Mandated every 2^{64} bytes

19

SSL/TLS Change Cipher Spec Protocol

- one of 3 SSL control protocols
- a single message
- causes pending state to become current
- updating the cipher suite in use
- TLS: cipher suites
  - One mandatory (strong)
    TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
  - Other RFCs add more, e.g. from #3268(strong)
    TLS_DHE_RSA_WITH_AES_128_CBC_SHA
  - Exportable version
    TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
    (512-bit public-key, 40-bit secret-key)

20

CipherSuite RSA

- initial state of a TLS connection during the first handshake
  - TLS_NULL_WITH_NULL_NULL = { 0x00,0x00 }
- CipherSuite for server with RSA certificate
  - TLS_RSA_WITH_NULL_MD5 = { 0x00,0x01 };
  - TLS_RSA_WITH_NULL_SHA = { 0x00,0x02 };
  - TLS_RSA_EXPORT_WITH_RC4_40_MD5 = { 0x00,0x03 };
  - TLS_RSA_WITH_RC4_128_MD5 = { 0x00,0x04 };
  - TLS_RSA_WITH_RC4_128_SHA = { 0x00,0x05 };
  - TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5 = { 0x00,0x06 };
  - TLS_RSA_WITH_IDEA_CBC_SHA = { 0x00,0x07 };
  - TLS_RSA_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x08 };
  - TLS_RSA_WITH_DES_CBC_SHA = { 0x00,0x09 };
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0A };

21

CipherSuite DH

- TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x0B };
- TLS_DH_DSS_WITH_DES_CBC_SHA = { 0x00,0x0C };
- TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA = { 0x00,0x0D };
- TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x0E };
- TLS_DH_RSA_WITH_DES_CBC_SHA = { 0x00,0x0F };
- TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x10 };
- TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x11 };
- TLS_DHE_DSS_WITH_DES_CBC_SHA = { 0x00,0x12 };
- TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA = { 0x00,0x13 };
- TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x14 };
- TLS_DHE_RSA_WITH_DES_CBC_SHA = { 0x00,0x15 };
- TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA = { 0x00,0x16 };

22

CipherSuite anonymous DH

- used for completely anonymous Diffie-Hellman communications where neither party is authenticated.
- this mode is vulnerable to man-in-the-middle attacks

- TLS_DH_anon_EXPORT_WITH_RC4_40_MD5 = { 0x00,0x17 };
- TLS_DH_anon_WITH_RC4_128_MD5 = { 0x00,0x18 };
- TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA = { 0x00,0x19 };
- TLS_DH_anon_WITH_DES_CBC_SHA = { 0x00,0x1A };
- TLS_DH_anon_WITH_3DES_EDE_CBC_SHA = { 0x00,0x1B };

23

CipherSuite exportable version

- the size of the largest public key is 512-bit for both DH and RSA

- ciphers

Cipher	Type	Key Material	Expanded Key Material	Effective Key Bits	IV Size	Block Size
NULL	* Stream	0	0	0	0	N/A
IDEA_CBC	Block	16	16	128	8	8
RC2_CBC_40	* Block	5	16	40	8	8
RC4_40	* Stream	5	16	40	0	N/A
RC4_128	Stream	16	16	128	0	N/A
DES40_CBC	* Block	5	8	40	8	8
DES_CBC	Block	8	8	56	8	8
3DES_EDE_CBC	Block	24	24	168	8	8

24

SSL Alert Protocol

- conveys SSL-related alerts to peer entity
- severity
  - warning or fatal
- specific alert
  - unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
  - close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown
- compressed & encrypted like all SSL data

25

TLS: connection state

- 4 states
  - Current and pending states
  - Independently for read and write directions
- Each state contains 3 algorithms
  - MAC, compression, block cipher (plus parameters, e.g. IV)
  - Initialized with null-null-null
- Change_cipher_state sets current=pending
  - You have to initialize before you can use

- *Heartbeat extension --- Heartbleed*

26

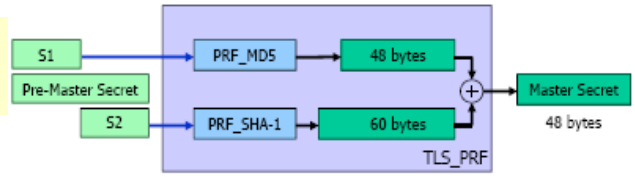
TLS: key exchange

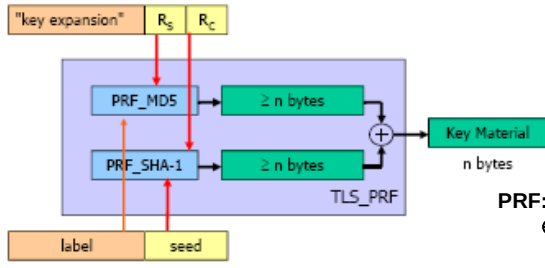
- *Pre-master-secret* is from one of
  - Diffie-Hellman key exchange
  - Client secret encrypted with server's certificate public key
  - Kerberos (see RFC 2712)
- Shared master secret computed as:
  PRF(pre-master-secret, client_random || server_random)
  - master secret --a 48-byte secret shared by the two peers
  - client random -- a 32-byte value provided by the client.
  - server random -- 32-byte
- Shared master secret is then used to generate keys: (MAC-key, cipher-key, IV,..)

27

TLS: key exchange

Compute Shared master secret from Pre-Master Secret
PRF(PMS, client_random || server_random)





PRF: iterated HMAC(secret, MD5, label || seed)
⊕ iterated HMAC(secret, SHA1, label || seed)

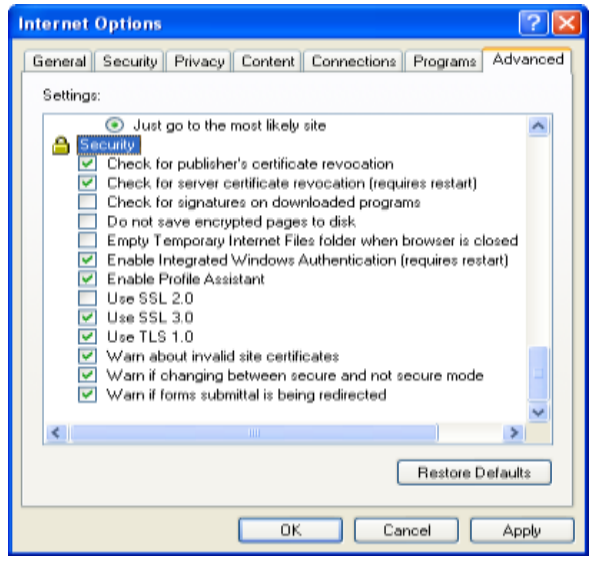
Generating Key Material
key_block = PRF(master_secret, "key exp.", server_random + client_random)
Such as keys and IVs for encryption, MAC

Generating True Random Numbers (RFC 1750)

- The security of cryptographic protocols relies heavily on the availability of random key material and nonces.
- On standard computer platforms it is not a trivial task to collect true random material in sufficient quantities:
  - Key Stroke Timing
  - Mouse Movements
  - Sampled Sound Card Input Noise
  - Air Turbulence in Disk Drives
  - RAID Disk Array Controllers
  - Network Packet Arrival Times
  - Computer Clocks
- Best Strategy: Combining various random sources with a strong mixing function (e.g. MD5 or SHA-1 hash) into an entropy pool (e.g. Unix /dev/random) protects against single device failures.

29

SSL/TLS Configuration Options
Internet Explorer



30

SSL history – v2

- Transport layer security service
  - Actually a protocol, not an API, though an API is usually nearby
- Popularity really came from Netscape's attempt to jumpstart ecommerce
- 1994: Netscape designed and built SSLv2
  - and told consumers that they needed SSL; credit card numbers were too sensitive to let go unencrypted
- Only Netscape Commerce Server supported SSL
  - It relied on X.509 certificates issued by RSADSI
- Microsoft had PCT (Private Communications Technology), backwards-compatible with SSLv2
  - Fixed various problems, added some new features

31

SSL history – v3

- Microsoft Secure Transport Layer Protocol (STLP)
  - Derived from SSLv3
  - Supported unreliable transport (UDP), client authentication via shared secrets
- 1996 IETF Transport Layer Security working group
  - to reconcile SSL and PCT/STLP (and others?) into an IETF protocol
  - SSLv3 "won" and is the basis for TLS
  - IESG (steering group) instructed working group to add DSS, DH, 3DES
    - Big deal because of Netscape's preference for RSA
    - But more so because 3DES was not exportable
      - 1995 "Danvers doctrine" said to make decisions based on good engineering rather than national policies
      - See RFC 3365, "Strong Security Requirements for Internet Engineering Task Force Standard Protocols "

32

SSL history - TLS

- RSADSI provided X.509 certificates for server authentication
- RSADSI spun off Verisign
- Others seemed poised to compete but didn't really
- Strongest competitor Thawte was eventually bought by Verisign
- Not much competition remains
- Public Key Infrastructure (PKIX) working group for IETF standardization of X.509 certs
  - TLS depended on this group's work
- TLS published in January 1999 as RFC 2246

33

U.S. Crypto Export control

- NSA made the decisions
- Authentication was generally approved
- Before Sept '98: encryption required export license
  - up to 40-bit DES, 512-bit key exchange
  - RC2 and RC4 particularly favored
  - larger sizes available to banks
- After Sept '98:
  - review still required
  - up to 56-bit DES and 1024-bit key exchange
- Traces of all this are visible throughout SSL
- After Jan 2000:
  - open source can just be posted on Internet
  - commercial/retail software still formally required to undergo review, but generally approval follows

34

TLS v1.0 and SSL v3

- IETF standard RFC 2246 similar to SSLv3
- with minor differences
  - in record format version number
  - uses HMAC for MAC
  - a pseudo-random function expands secrets
  - has additional alert codes
  - some changes in supported ciphers
  - changes in certificate negotiations
  - changes in use of padding

35

versions

表1 密码组对已知的可行攻击的安全性[68]

密码组	协议版本				
	SSL 2.0	SSL 3.0	TLS 1.0	TLS 1.1	TLS 1.2
3DES CBC	不安全	可能安全	可能安全	可能安全	可能安全
AES CBC	不支持	不支持	可能安全	安全	安全
AES GCM	不支持	不支持	不支持	不支持	安全
AES CCM	不支持	不支持	不支持	不支持	安全
Camellia CBC	不支持	不支持	可能安全	安全	安全
Camellia GCM	不支持	不支持	不支持	不支持	安全
SEED CBC	不支持	不支持	可能安全	安全	安全
IDEA CBC	不安全	可能安全	可能安全	安全	不支持
DES CBC	不安全	不安全	不安全	不安全	不支持
RC2 CBC	不安全	不安全	不安全	不安全	不支持
RC4	不安全	不安全	不安全	不安全	不安全
ChaCha20 Poly1305	不支持	不支持	不支持	不支持	安全

2013年12月3日的调查数据总结了主流网站的TLS配置在目前主要攻击下的安全性。

36

TLS Key Exchange

- Anonymous Diffie-Hellman (DH)
  - unauthenticated, and not recommended.
- Static DH
  - Server's contribution is fixed in cert, rare
- Ephemeral DH
  - server authenticates contribution by signing it, common
- Fortezza
  - PCMCIA smart card
  - Skipjack (declassified June 1998)
  - Key escrow
  - Law Enforcement Access Field (LEAF)
    - Key encrypted in IV
  - Extremely rare
- Server Gated Crypto (SGC)
  - Historical
  - For approved financial transactions
  - Special server certs allow clients to use strong crypto where they would normally refuse

37

Client Authentication

- SSL/TLS, IE, Netscape all support client certs
  - But rarely used outside of corporate settings
  - Client is implicitly authenticated through credit-card number
- HTTP Basic authentication within SSL session
  - Cleartext password stored on server, but hidden on wire

38

Heartbleed

- **Heartbleed** is a security bug in the open-source OpenSSL cryptography library
- Heartbleed results from improper input validation (due to a missing bounds check) in the implementation of the TLS heartbeat extension
- It is classified as a buffer over-read,[5] a situation where software allows more data to be read than should be allowed
- A fix was released on April 7, 2014, when Heartbleed was disclosed. At that time, some 17% t (around half a million) of the Internet's secure web servers certified by trusted authorities were believed to be vulnerable to the attack,
- allowing theft of the servers' private keys and users' session cookies and passwords

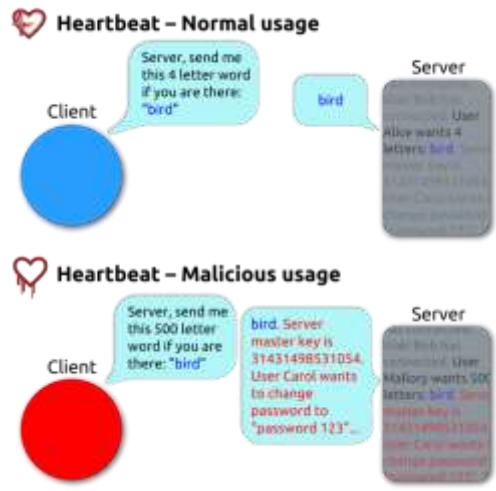
--wikipedia

39

Heartbeat--Heartbleed

Heartbleed
results from improper input validation (due to a missing bounds check) in the implementation of the TLS heartbeat extension

--wikipedia



40

SSL and Application Protocols

Service Name	Port	Secured Service
• https	443/tcp	http protocol over TLS/SSL
• smtps	465/tcp	smtp protocol over TLS/SSL
• nntps	563/tcp	nntp protocol over TLS/SSL
• sshell	614/tcp	SSLshell
• ldaps	636/tcp	ldap protocol over TLS/SSL
• ftps-data	989/tcp	ftp protocol, data, over TLS/SSL
• ftps	990/tcp	ftp, control, over TLS/SSL
• telnets	992/tcp	telnet protocol over TLS/SSL
• imaps	993/tcp	imap4 protocol over TLS/SSL
• ircs	994/tcp	irc protocol over TLS/SSL
• pop3s	995/tcp	pop3 protocol over TLS/SSL

41

HTTPS

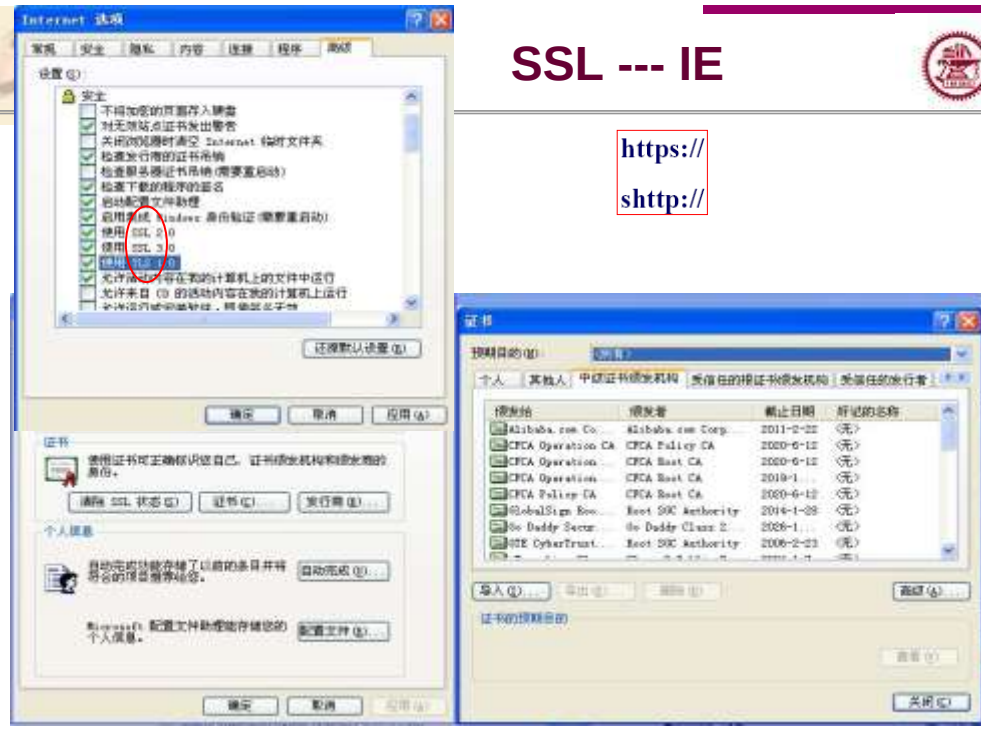
- RFC 2818 - HTTP Over TLS (HTTPS, port 443)
- URI Format https://www.example.com/~smith/home.html
- RFC 2817 - Upgrading to TLS Within HTTP/1.1
  - use the upgrade mechanism in HTTP/1.1 to initiate TLS over an existing TCP connection.
  - This allows unsecured and secured HTTP traffic to share the same well known port (in this case, http: at 80 rather than https: at 443).
  - It also enables "virtual hosting", so a single HTTP + TLS server can disambiguate traffic intended for several hostnames at a single IP address.

42

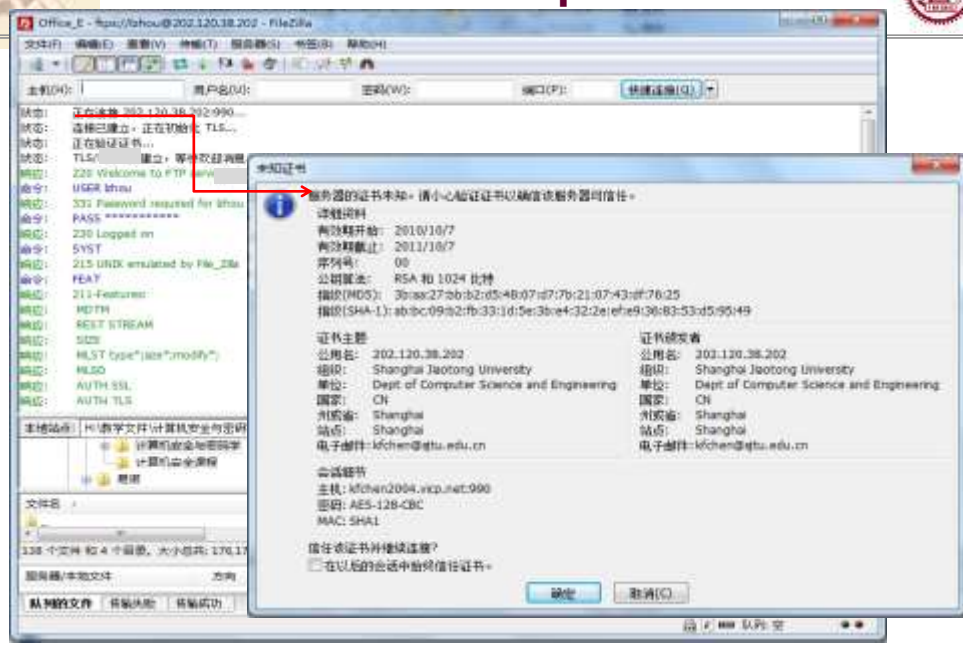
SSL --- IE

https://

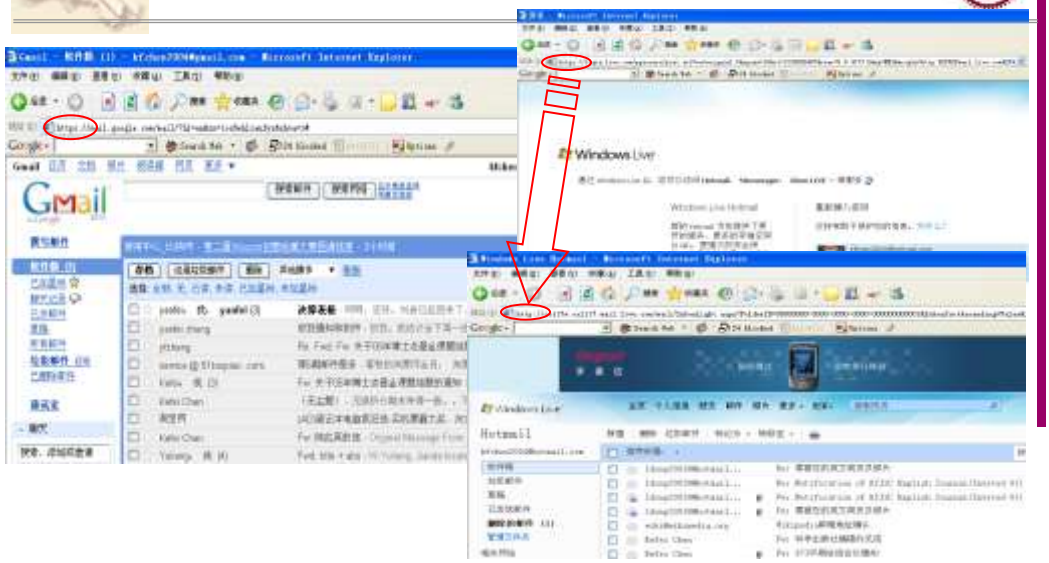
shttp://



SSL: secure ftp



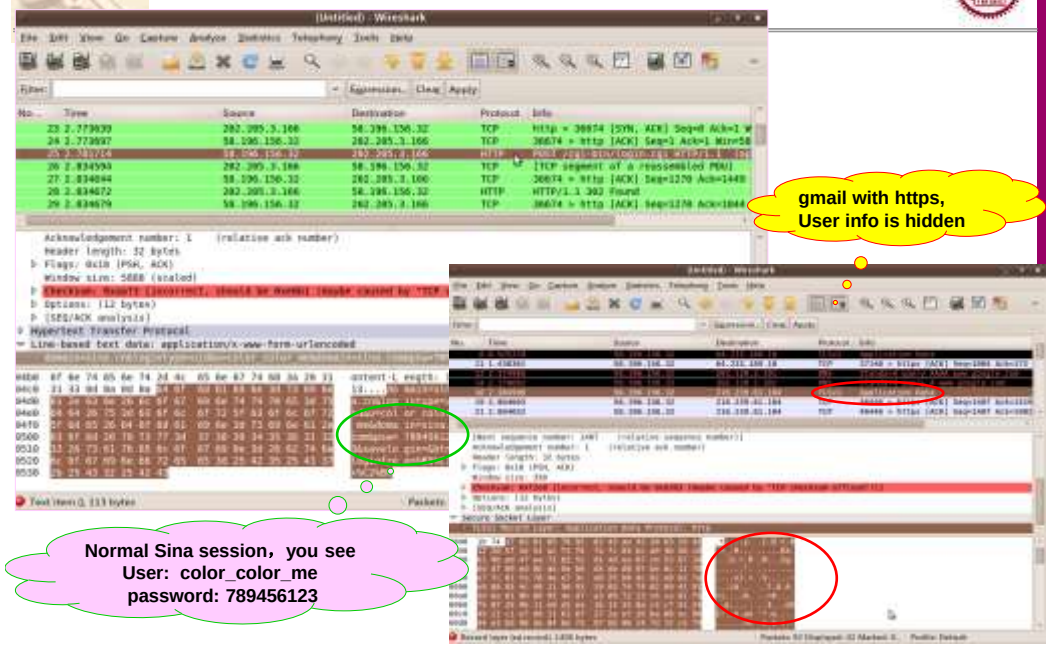
SSL: E-mail



HTTPS and Web Servers

- HTTP over SSL usually uses port 443
  - 443 means "use SSL“ for all versions
- Web pages typically include a lot of embedded content
  - potentially fetched over different TCP connections
  - session resumption is critical for performance
  - HTTP persistent connections are very helpful
- Proxies
  - A proxy is a man-in-the-middle
  - HTTP "CONNECT" method just relays data; proxy can't examine
  - Possible to reconfigure clients so that a real man-in-the-middle "attack" on https is possible
    - You set up your own CA
- Apache / OpenSSL / mod_ssl very common combination
  - Fairly complicated setup
- Plenty of commercial server support

The effect of SSL



Is HTTPS secure ?

- in general yes, but pay attention to:

1. When asked for password, look for the small lock icon,
2. verify that the URL is reasonable
3. Other browser issues

Exercise 14: Deadline: one-week

What would happen if 1 , 2 are ignored?

48

SSL VPN

- An SSL VPN can be used with a standard Web browser.
  - In contrast to the traditional IPsec VPN, an SSL VPN does not require the installation of specialized client software on the end user's computer.
  - It's used to give remote users with access to Web applications, client/server applications and internal network connections
- An SSL VPN consists of one or more VPN devices to which the user connects by using his Web browser. The traffic between the Web browser and the SSL VPN device is encrypted with the SSL/TLS