CryptoWorks21

Fundamentals of Network Security

3. Network Security Protocols

CryptoWorks21 • July 25 & 27, 2017

Dr Douglas Stebila
McMaster University

https://www.douglas.stebila.ca/teaching/cryptoworks21

Fundamentals of Network Security

1. Basics of Information Security
   - Security architecture and infrastructure; security goals (confidentiality, integrity, availability, and authenticity); threats/vulnerabilities/attacks; risk management
2. Cryptographic Building Blocks
   - Symmetric crypto: ciphers (stream, block), hash functions, message authentication codes, pseudorandom functions
   - Public key crypto: public key encryption, digital signatures, key agreement
3. Network Security Protocols & Standards
   - In detail: public key infrastructure, TLS
   - Overview: Networking, SSH, IPsec, Kerberos, WEP
4. Network Scanning and Defence
   - Traffic sniffing and network reconnaissance (mmap)
   - Network protection: firewalls and intrusion detection
5. Access Control & Authentication; Web Application Security
   - Access control: discretionary/mandatory/role-based; phases
   - Authentication: something you know/have/are/somewhere you are
   - Web security: cookies, SQL injection
   - Supplemental material: Passwords

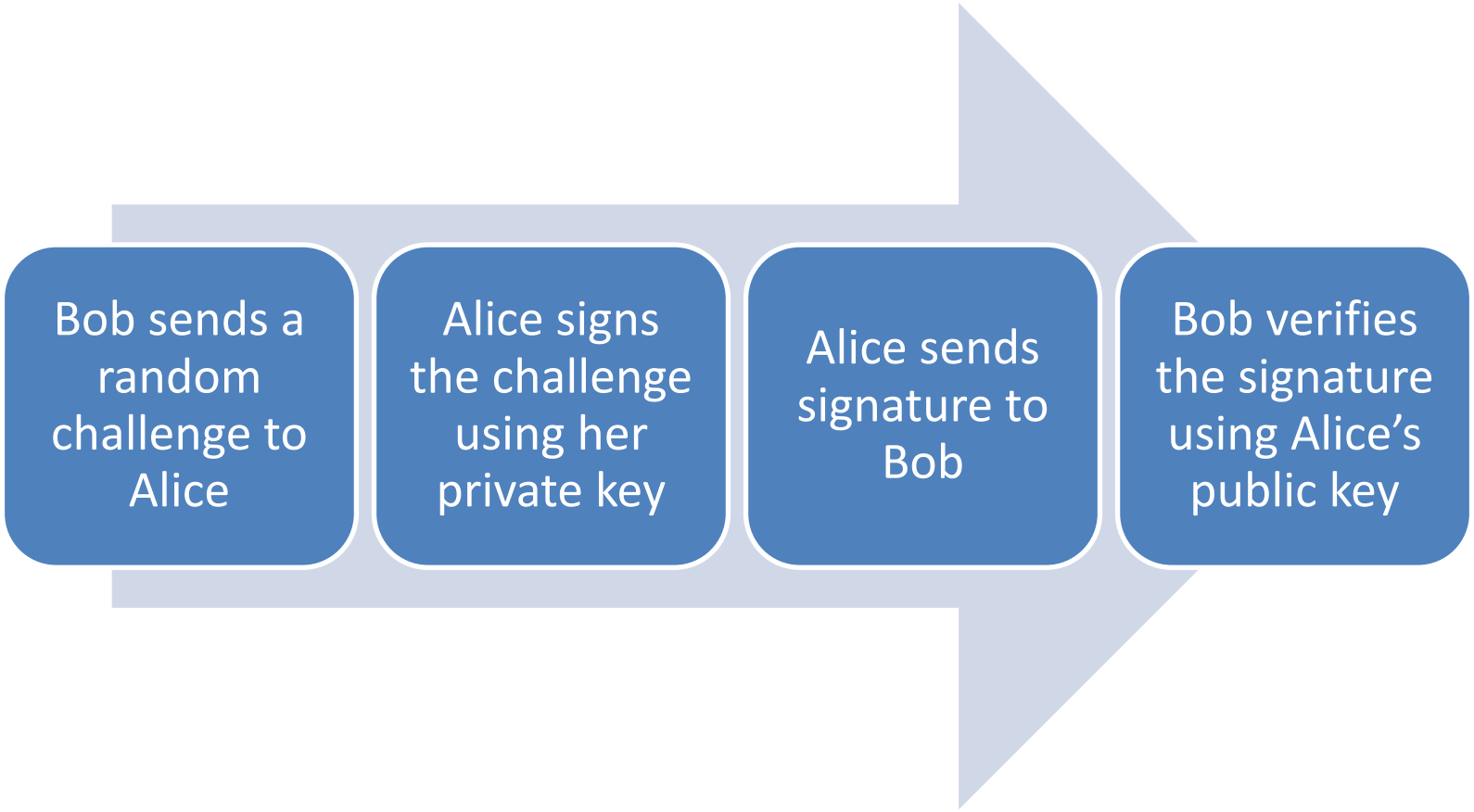
Network Security Protocols

- Public Key Infrastructure
- Networking
- Transport Layer Security (TLS)
- Other protocols
  - Secure Shell (SSH)
  - IPsec
  - Kerberos
  - Wired Equivalent Protocol (WEP)

Problem: How does Bob get Alice's public key to begin with?

PUBLIC KEY INFRASTRUCTURES (PKI)

Using digital signatures for entity authentication

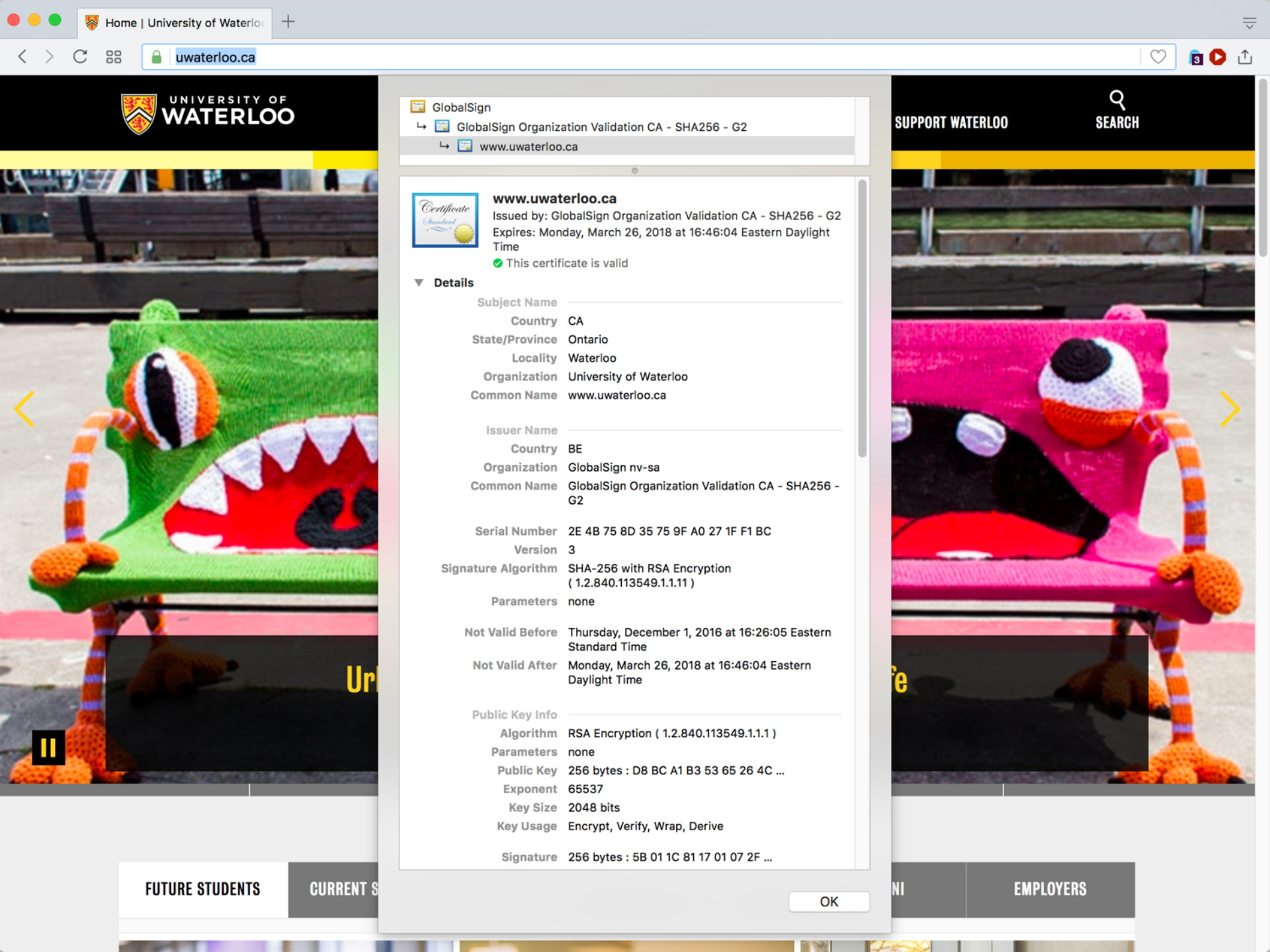


Certificates and certificate authorities

- A **certificate** is an assertion by a trusted third party that a particular public key belongs to a particular entity.

- The **certificate authority** generates a certificate by
  1. Obtaining the user's public key by some trust mechanism.
  2. Verifying that the user really is who she says she is.
  3. Signing (using the certificate authority's public key) the user's public key and name.

- This allows two parties who have never met to establish trust between them:
  - Exchange certificates.
  - Do authentication using digital signatures.
  - If they each trust the certificate authority that signed the other party's certificate, they can now be certain who the other party is.


Home | University of Waterloo
uwaterloo.ca
UNIVERSITY OF WATERLOO
SUPPORT WATERLOO
SEARCH
GlobalSign
GlobalSign Organization Validation CA - SHA256 - G2
www.uwaterloo.ca
www.uwaterloo.ca
Issued by: GlobalSign Organization Validation CA - SHA256 - G2
Expires: Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time
This certificate is valid
Details
Subject Name
Country CA
State/Province Ontario
Locality Waterloo
Organization University of Waterloo
Common Name www.uwaterloo.ca
Issuer Name
Country BE
Organization GlobalSign nv-sa
Common Name GlobalSign Organization Validation CA - SHA256 - G2
Serial Number 2E 4B 75 8D 35 75 9F A0 27 1F F1 BC
Version 3
Signature Algorithm SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters none
Not Valid Before Thursday, December 1, 2016 at 16:26:05 Eastern Standard Time
Not Valid After Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time
Public Key Info
Algorithm RSA Encryption (1.2.840.113549.1.1.1)
Parameters none
Public Key 256 bytes : D8 BC A1 B3 53 65 26 4C ...
Exponent 65537
Key Size 2048 bits
Key Usage Encrypt, Verify, Wrap, Derive
Signature 256 bytes : 5B 01 1C 81 17 01 07 2F ...
OK
FUTURE STUDENTS
EMPLOYERS

Domain name

Certificate authority

Validity period

X.509 certificates

A standardized format for certificates.

Uses a strange (old) format called ASN.1 and a strange binary encoding.

Public key

Revocation information

CA's signature on everything

www.uwaterloo.ca
Issued by: GlobalSign Organization Validation CA - SHA256 - G2
Expires: Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time
This certificate is valid

Details

Subject Name	
Country	CA
State/Province	Ontario
Locality	Waterloo
Organization	University of Waterloo
Common Name	www.uwaterloo.ca
Issuer Name	
Country	BE
Organization	GlobalSign nv-sa
Common Name	GlobalSign Organization Validation CA - SHA256 - G2
Serial Number	2E 4B 75 8D 35 75 9F A0 27 1F F1 BC
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	none
Not Valid Before	Thursday, December 1, 2016 at 16:26:05 Eastern Standard Time
Not Valid After	Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	none
Public Key	256 bytes : D8 BC A1 B3 53 65 26 4C 39 D5 92 56 84 66 37 CC 1F 57 FD 5B 87 A7 38 36 D5 05 83 3E 6C 96 02 12 3E 6A C3 CF F3 BC 3C 6D BF FE BB BB 08 02 8C 97 AF D9 86 2A 6B F6 EE D7 0C DA E8 2F DA B1 14 E8 B5 EA 04 FF 12 3D BA ED 42 FA CE A7 93 AC 15 29 66 63 2E 39 7F F2 69 D7 82 01 CB B8 92 81 75 B3 F9 4A 87 32 05 67 E0 42 78 55 1F 03 17 A8 8F 6E 85 56 1B 1C AF 8E 35 A5 14 91 E9 25 61 AC 05 6F 9A FC 58 F8 7F 64 BE C7 D4 6A EB 2A BC 47 D6 30 35 51 1D BD 57 09 49 19 9E BC 43 09 F1 58 0C 88 E5 D1 9C CB 00 AA A8 66 E8 4B C9 CE AA 63 63 5A A9 AF 3D 63 90 E8 7A 2F 95 1B CC EC 2E 48 16 4A 0E B8 1F 69 45 82 3C F1 09 53 2C B6 69 8C 70 4C 99 89 6F 4E CA 0C 8D F5 1E 3A 5F 07 46 7D 63 ED 3D 38 B7 0E 88 ED 4F FD 00 C2 76 35 F7 99 5B 39 CE 26 CC C4 19 CA 47 DA 6D 80 61 7E 01 8E 96 DD
Exponent	65537
Key Size	2048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 5B 01 1C 81 17 01 07 2F ...

Extension	Key Usage (2.5.29.15)
Critical	YES
Usage	Digital Signature, Key Encipherment
Extension	Basic Constraints (2.5.29.19)
Critical	NO
Certificate Authority	NO
Extension	Extended Key Usage (2.5.29.37)
Critical	NO
Purpose #1	Server Authentication (1.3.6.1.5.5.7.3.1)
Purpose #2	Client Authentication (1.3.6.1.5.5.7.3.2)
Extension	Subject Key Identifier (2.5.29.14)
Critical	NO
Key ID	DB B9 21 BC CD 3E AF 70 C9 E9 3D 9B FF 42 B0 C8 88 8F 78 C3
Extension	Authority Key Identifier (2.5.29.35)
Critical	NO
Key ID	96 DE 61 F1 BD 1C 16 29 53 1C C0 CC 7D 3B 83 00 40 E6 1A 7C
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	www.uwaterloo.ca
DNS Name	uwaterloo.ca
Extension	Certificate Policies (2.5.29.32)
Critical	NO
Policy ID #1	(1.3.6.1.4.1.4146.1.20)
Qualifier ID #1	Certification Practice Statement (1.3.6.1.5.5.7.2.1)
CPS URI	https://www.globalsign.com/repository/
Policy ID #2	(2.23.140.1.2.2)
Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://crl.globalsign.com/gs/gsorganizationvalsha2g2.crl
Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO
Method #1	CA Issuers (1.3.6.1.5.5.7.48.2)
URI	http://secure.globalsign.com/cacert/gsorganizationvalsha2g2r1.crt
Method #2	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://ocsp2.globalsign.com/gsorganizationvalsha2g2
Fingerprints	
SHA-256	C7 DC B4 CD 45 9E D5 1A AA 03 86 73 31 4B F8 A9 53 A6 9C F1 B9 C4 35 A3 AD C6 4F 87 97 93 AD D6
SHA-1	53 9E 06 03 64 F9 24 F6 ED 9B A1 0E F9 46 81 1A E0 88 F8 A5

Certificate revocation

- Once a certificate's been issued, what happens if the user's private key has been compromised?
- We would like to be able to **revoke** the certificate, or indicate that it should no longer be trusted.
- When revoking certificates that were used for message authentication, what does that mean for documents signed using that certificate? May need a trusted timestamp server as well.

Certificate Revocation Lists (CRLs)

- Each CA can publish a file containing a list of certificates that have been revoked.
- Have to download whole list.
- CRL address often included in certificate.

Online Certificate Status Protocol (OCSP)

- An online service run by a CA for checking in real-time if a certificate has been revoked.
- Don't have to download whole list.
- Not widely implemented.

Public key infrastructure

A **public key infrastructure (PKI)** is

- a set of systems (hardware, software, policies, procedures)
- for managing (creating, distributing, storing, revoking)
- digital certificates.

Includes:

- one or more certificate authorities
- users
- relying parties
- possibly a timestamp server
- possibly a directory server storing certificates (e.g., LDAP server, Active Directory server)

Certificate types

Domain validation

- Identity confirmed by validating control over DNS record
- Let's Encrypt $0
- Comodo $77
- Thawte $149

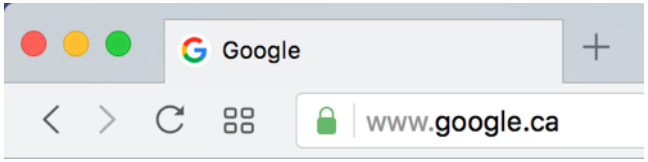
Organization validation

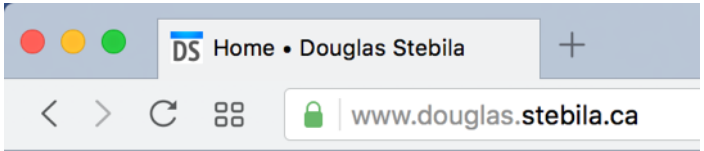
- Identity confirmed by some checks of legal status of organization
- Symantec $995
- Thawte $199

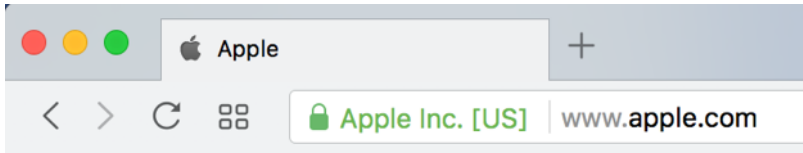
Extended validation

- More rigourous check of organization's existence
- Symantec $995
- Thawte $299

Eye-tracking studies show that users do not notice these additional security indicators

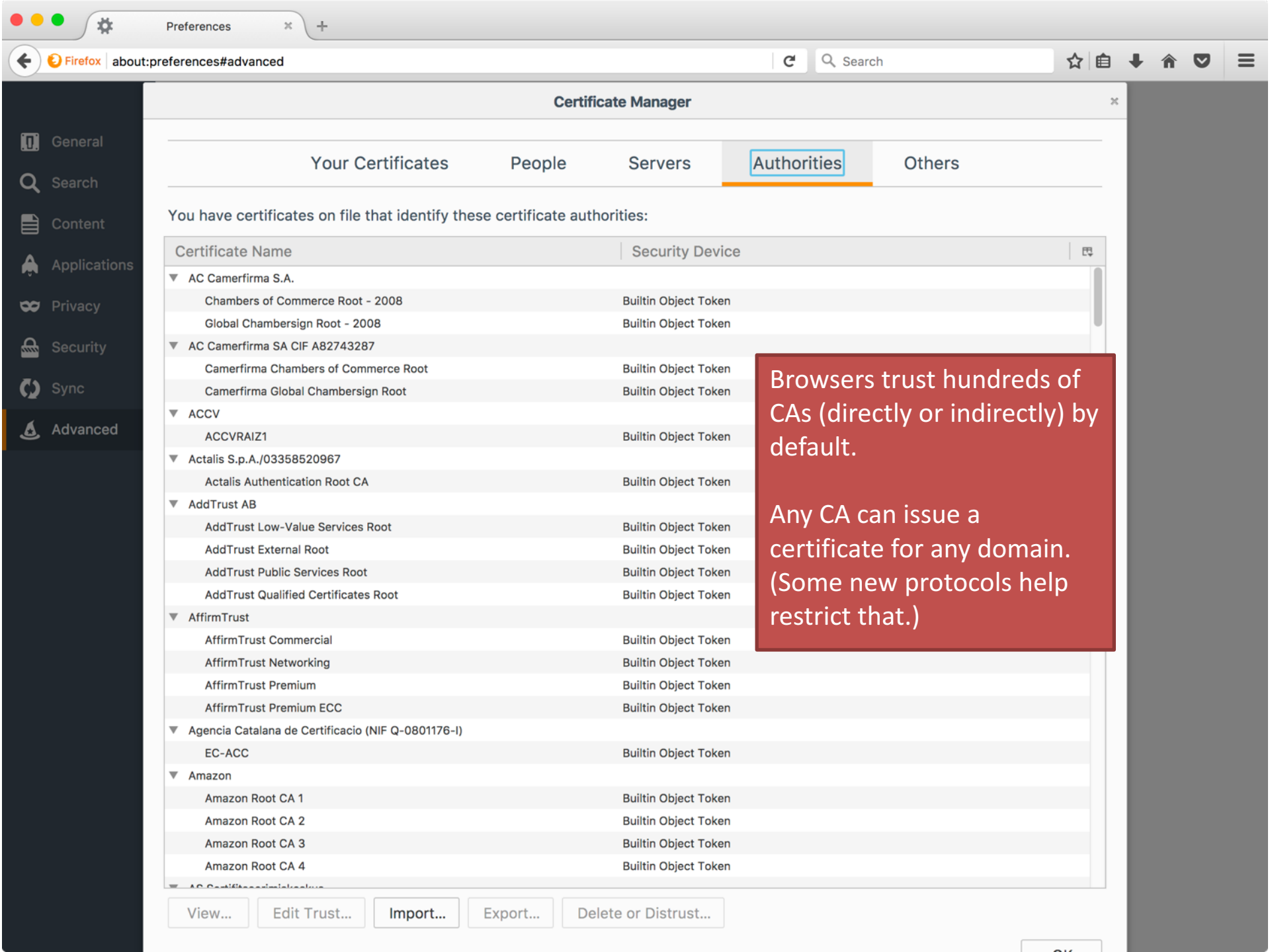






Certificate Transparency

- A certificate logging mechanism to allow anyone to check which certificates a CA has issued
- Auditors monitor CAs to watch for malicious behaviour
- Domain name owners monitor the logs to check for certificates issued for their domains


Preferences
Firefox about:preferences#advanced
Search
General
Search
Content
Applications
Privacy
Security
Sync
Advanced
Certificate Manager
Your Certificates
People
Servers
Authorities
Others
You have certificates on file that identify these certificate authorities:
Certificate Name
Security Device
AC Camerfirma S.A.
Chambers of Commerce Root - 2008
Builtin Object Token
Global Chambersign Root - 2008
Builtin Object Token
AC Camerfirma SA CIF A82743287
Camerfirma Chambers of Commerce Root
Builtin Object Token
Camerfirma Global Chambersign Root
Builtin Object Token
ACCV
ACCVRAIZ1
Builtin Object Token
Actalis S.p.A./03358520967
Actalis Authentication Root CA
Builtin Object Token
AddTrust AB
AddTrust Low-Value Services Root
Builtin Object Token
AddTrust External Root
Builtin Object Token
AddTrust Public Services Root
Builtin Object Token
AddTrust Qualified Certificates Root
Builtin Object Token
AffirmTrust
AffirmTrust Commercial
Builtin Object Token
AffirmTrust Networking
Builtin Object Token
AffirmTrust Premium
Builtin Object Token
AffirmTrust Premium ECC
Builtin Object Token
Agencia Catalana de Certificacio (NIF Q-0801176-I)
EC-ACC
Builtin Object Token
Amazon
Amazon Root CA 1
Builtin Object Token
Amazon Root CA 2
Builtin Object Token
Amazon Root CA 3
Builtin Object Token
Amazon Root CA 4
Builtin Object Token
View...
Edit Trust...
Import...
Export...
Delete or Distrust...
Browsers trust hundreds of CAs (directly or indirectly) by default.
Any CA can issue a certificate for any domain. (Some new protocols help restrict that.)

Secure email

- X.509 certificates can also be used to send secure email:
  - digitally signed
  - encrypted
- **S/MIME** (Secure/Multipurpose Internet Mail Extensions):
  - Supported in most desktop mail programs.
  - Relies on a public key infrastructure.
- **PGP** (Pretty Good Privacy):
  - Available as an add-on to most desktop mail programs.
  - Uses public keys, but doesn't require CAs: users manually distribute their keys in a "web of trust"
- Not widely used:
  - Users must know how set up public keys and obtain S/MIME X.509 certificate or distribute PGP public keys.
  - Little to no support in webmail.

Applications of PKIs

- Web site authentication (TLS)
- Email authentication (S/MIME, PGP)
- Domain names (DNSSEC)
- Digital identities
  - e.g., national identity cards (Belgium, Spain, Germany)
- Business-to-business e-commerce
  - e.g., digitally signing transactions, XML signatures

NETWORKING

IETF Internet Protocol suite

	Layer	Examples
Most defined by the Internet Engineering Task Force (IETF)	**Application**	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
	Transport	connection-oriented (TCP) connectionless (UDP)
	Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
	Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

There's also the 7-layer OSI model.

Link (a.k.a. network access) layer

The **link** or **network access layer** is the physical layer and is associated with computer hardware.

Computer networks can use a large number of connections and transmission media

- Telephone wires
- Ethernet (twisted pair) cables
- Optic Fibre cables
- Satellite communications
- Mobile phone networks
- Wireless networks
- Bluetooth

At this layer physical addresses identify network nodes

- Ethernet MAC address

Internet (a.k.a. network) layer

The **Internet layer** runs a low level protocol called the Internet Protocol (IP) (plus a few extra helpers, e.g. ICMP).

- IPv4 (1981), IPv6 (1996)

Host addressing and identification:

- Each host has a unique IP address:
  - IPv4, 32 bit, e.g., 131.181.118.220
  - IPv6, 128 bit, e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334

Packet routing:

- Organizations are assigned a range of IP addresses that they manage and assign to their computers.

Transport Layer

The **transport layer** establishes basic data channels for applications. It uses **ports** to distinguish between different applications on the same host.

TCP (Transmission Control Protocol)

- **connection-oriented** protocol
  - back-and-forth, ongoing connections
- **reliability**
  - large messages split into packets
  - in-order delivery of packets, recombined to large message
  - error checking
  - retransmission of lost packets
  - congestion control

UDP (User Datagram Protocol)

- **connectionless** protocol
  - send a packet, that's it
- **unreliable**
  - simple error checking
  - no retransmission of lost packets
  - used for streaming
    - audio, video, VOIP

Application layer

Application layer protocols are used by applications to provide user services over a network.

Each application protocol has unique message formats that are sent and received to achieve their tasks.

- HTTP (web)
- FTP (file transfer)
- SSH, Telnet (login)
- SMTP, POP3, IMAP (email)
- XMPP (chat)
- BitTorrent (I'm sure you know what this is used for)

Each application protocol requires the lower network layers (TCP, IP, Network Access) to communicate on the network.

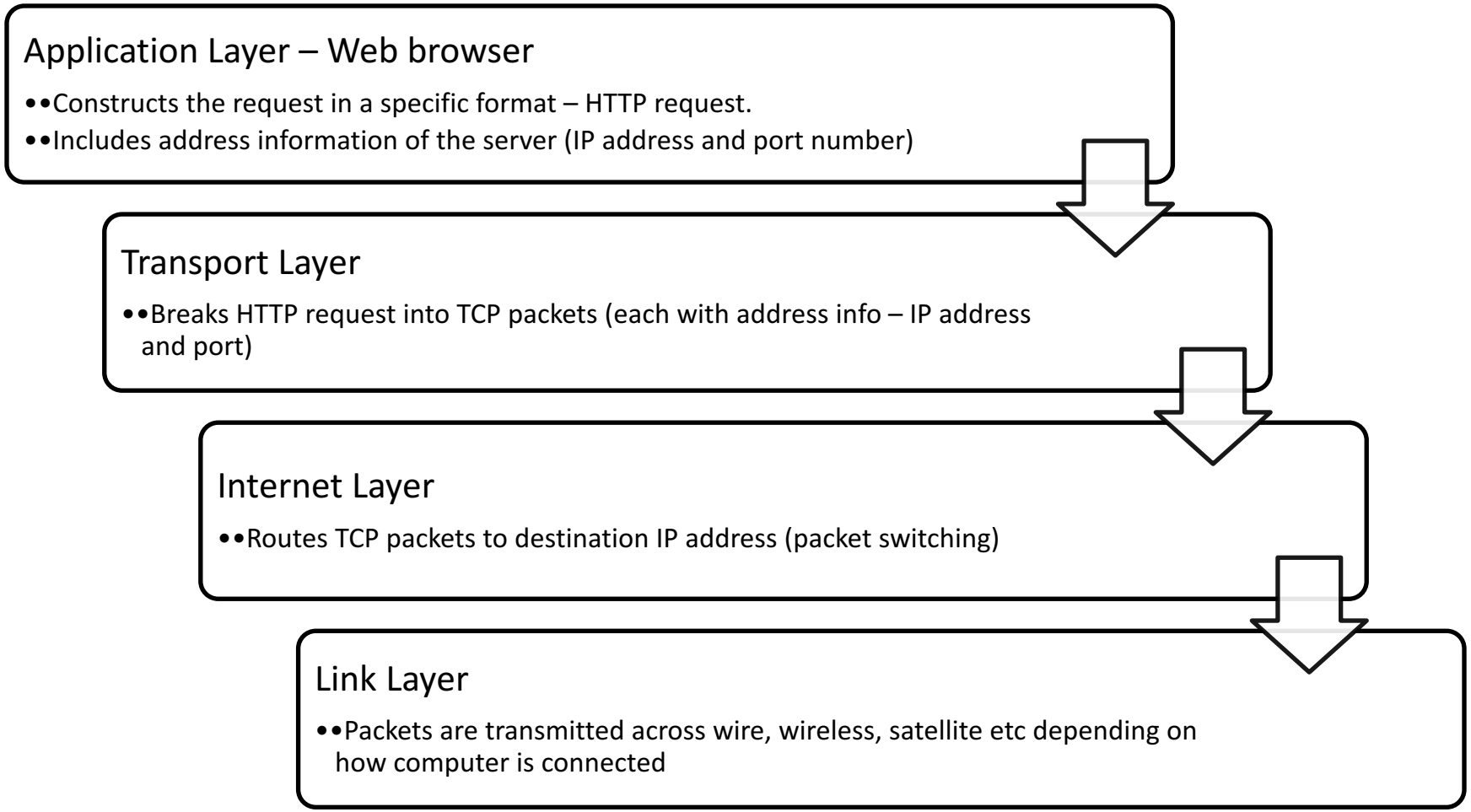
Many use an intermediate protocol called SSL/TLS for encryption and authentication.

Client-server on the Internet

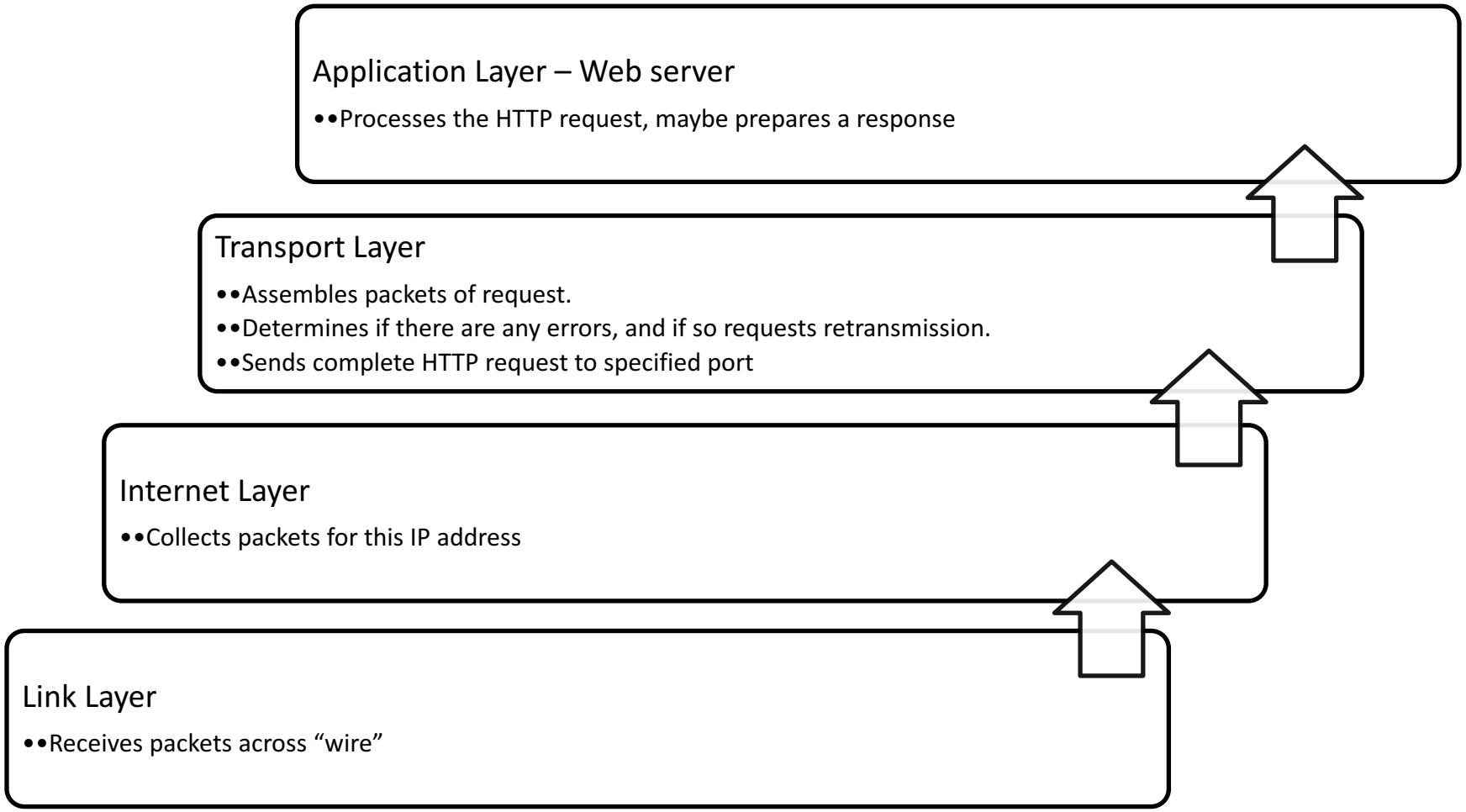
- Each **application server** listens for messages on a particular port number. Common ports:
  - web servers: port 80 (HTTP), 443 (HTTPS)
  - login: port 22 (SSH), 23 (Telnet)
  - file transfer: port 20/21 (FTP), 22 (SFTP/SCP)
  - email servers: port 25 (SMTP), 220/993 (IMAP), 110 (POP)

- Clients identify the machine they want to connect to using an IP address.
- Clients identify the program they want to use using a port number.

Example: requesting a webpage



Example: receiving a webpage request



Network security protocols

Network-related security protocols in common use include:

- Secure Shell (SSH):
  Used for remote login, file transfer, and limited VPN service.
- Transport Layer Security (TLS):
  Used extensively on the web and is often referred to in privacy policies as a means of providing confidential web connections.
- IP Security (IPsec):
  Provides security services at the IP level and is used to provide Virtual Private Network (VPN) services.
- WiFi security (WEP, WPA):
  Provides security services at the link layer for wireless communication

IETF Internet Protocol suite

Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

TRANSPORT LAYER SECURITY (TLS)

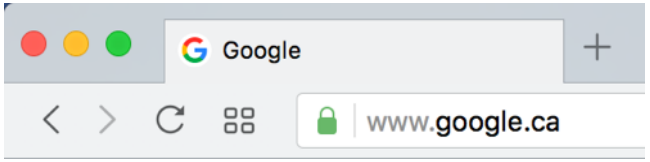
A.K.A. SECURE SOCKETS LAYER (SSL)

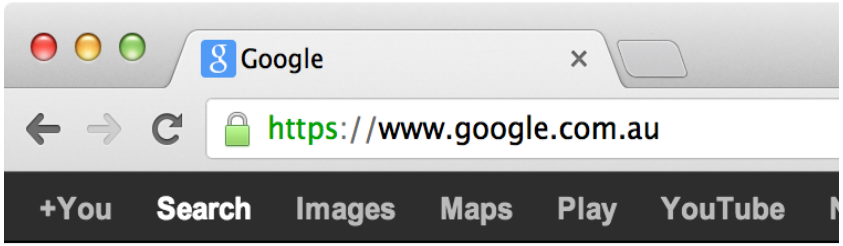
Terminology

- SSL: Secure Sockets Layer
- Proposed by Netscape
  - SSLv2: 1995
  - SSLv3: 1996

- TLS: Transport Layer Security
- IETF Standardization of SSL
  - TLSv1.0 = SSLv3: 1999
  - TLSv1.1: 2006
  - TLSv1.2: 2008
  - TLSv1.3: 2017?

- HTTPS: HTTP (Hypertext Transport Protocol) over SSL





Security goals of TLS

- Provides **authentication** based on public key certificates
  - server-to-client (always)
  - client-to-server (optional)
- Provides **confidentiality** and integrity of message transmission

- But only protects confidentiality if authentication is correct.

IETF Internet Protocol suite

TLS adds encryption to many application level protocols

Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN • ADSL • GSM/3G

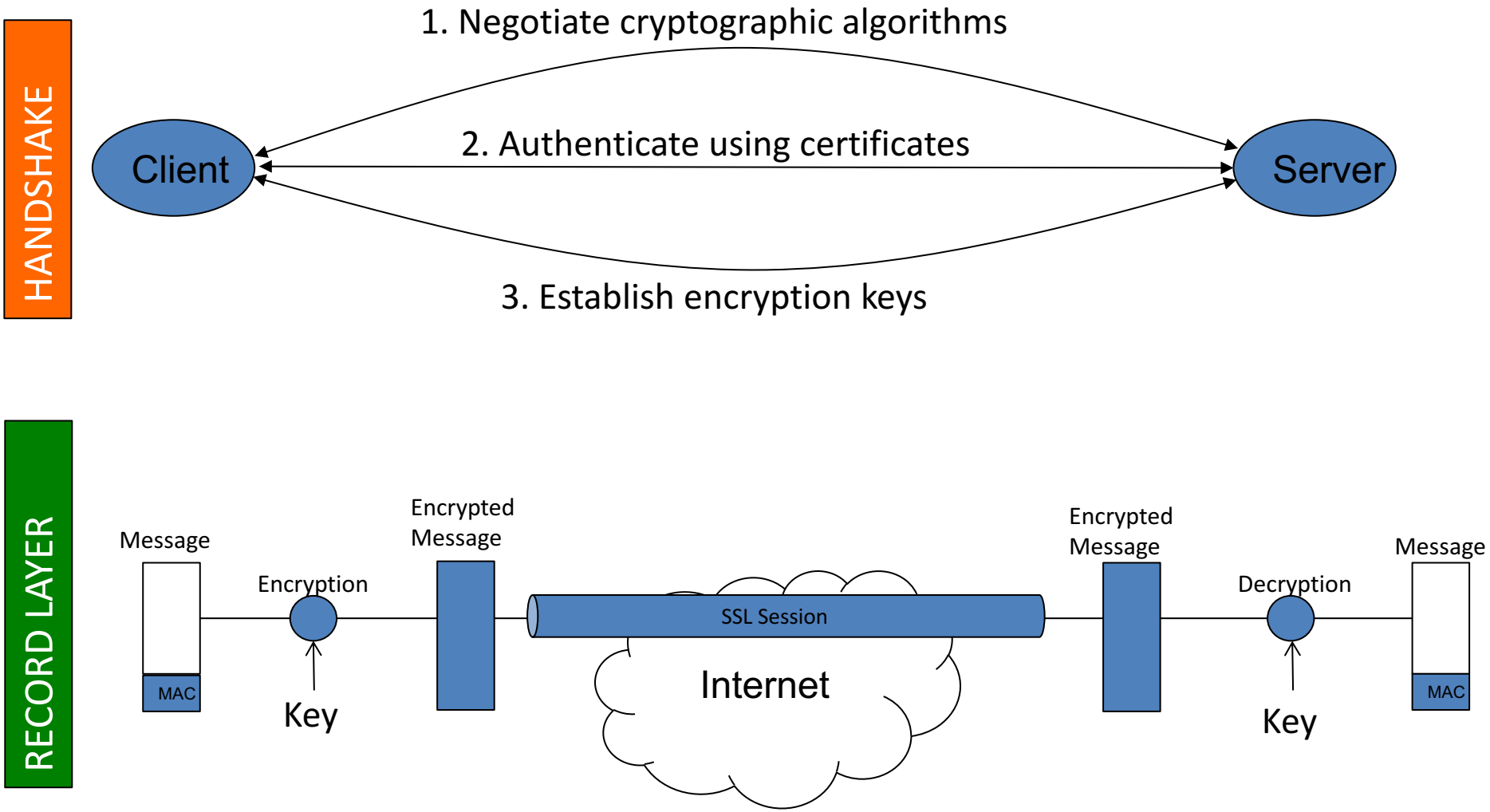
TLS

TLS and HTTP

- TLS can be used to provide protection for HTTP communications:
  - Port 443 is reserved for HTTP over TLS
- HTTPS is the name of the URL scheme used with this port.

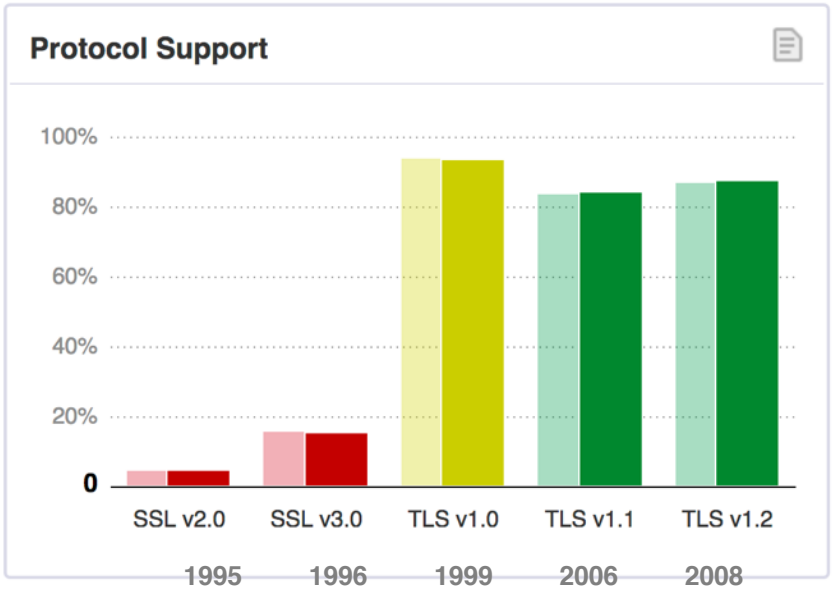
- http://www.develop.com implies the use of standard HTTP using port 80.
- https://www.develop.com implies the use of HTTP over TLS using port 443.

SSL/TLS Protocol



What is TLS?

- 5 protocol versions
- vast array of standards
- many implementations!
- 300+ combinations of cryptographic primitives
- different levels of security
- different modes of authentication
- additional functionality:
  - alerts & errors
  - session resumption
  - renegotiation
  - compression



https://www.trustworthyinternet.org/ssl-pulse/
July 2, 2017

The current approved version of TLS is version 1.2, which is specified in:

- RFC 5246: “The Transport Layer Security (TLS) Protocol Version 1.2”.

The current standard replaces these former versions, which are now considered obsolete:

- RFC 2246: “The TLS Protocol Version 1.0”.
- RFC 4346: “The Transport Layer Security (TLS) Protocol Version 1.1”.

as well as the never standardized SSL 3.0:

- RFC 6101: “The Secure Sockets Layer (SSL) Protocol Version 3.0”.

Other RFCs subsequently extended TLS.

Extensions to TLS 1.0 include:

- RFC 2595: “Using TLS with IMAP, POP3 and ACAP”. Specifies an extension to the IMAP, POP3 and ACAP services that allow the server and client to use transport-layer security to provide private, authenticated communication over the Internet.
- RFC 2712: “Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)”. The 40-bit cipher suites defined in this memo appear only for the purpose of documenting the fact that those cipher suite codes have already been assigned.
- RFC 2817: “Upgrading to TLS Within HTTP/1.1”, explains how to use the Upgrade mechanism in HTTP/1.1 to initiate Transport Layer Security (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).
- RFC 2818: “HTTP Over TLS”, distinguishes secured traffic from insecure traffic by the use of a different 'server port'.
- RFC 3207: “SMTP Service Extension for Secure SMTP over Transport Layer Security”. Specifies an extension to the SMTP service that allows an SMTP server and client to use transport-layer security to provide private, authenticated communication over the Internet.
- RFC 3268: “AES Ciphersuites for TLS”. Adds Advanced Encryption Standard (AES) cipher suites to the previously existing symmetric ciphers.
- RFC 3546: “Transport Layer Security (TLS) Extensions”, adds a mechanism for negotiating protocol extensions during session initialisation and defines some extensions. Made obsolete by RFC 4366.
- RFC 3749: “Transport Layer Security Protocol Compression Methods”, specifies the framework for compression methods and the DEFLATE compression method.
- RFC 3943: “Transport Layer Security (TLS) Protocol Compression Using Lempel-Ziv-Stac (LZS)”.
- RFC 4132: “Addition of Camellia Cipher Suites to Transport Layer Security (TLS)”.
- RFC 4162: “Addition of SEED Cipher Suites to Transport Layer Security (TLS)”.
- RFC 4217: “Securing FTP with TLS”.
- RFC 4279: “Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)”, adds three sets of new cipher suites for the TLS protocol to support authentication based on pre-shared keys.

Extensions to TLS 1.1 include:

- RFC 4347: “Datagram Transport Layer Security” specifies a TLS variant that works over datagram protocols (such as UDP).
- RFC 4366: “Transport Layer Security (TLS) Extensions” describes both a set of specific extensions and a generic extension mechanism.
- RFC 4492: “Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)”.
- RFC 4507: “Transport Layer Security (TLS) Session Resumption without Server-Side State”.
- RFC 4680: “TLS Handshake Message for Supplemental Data”.
- RFC 4681: “TLS User Mapping Extension”.
- RFC 4785: “Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)”.
- RFC 5054: “Using the Secure Remote Password (SRP) Protocol for TLS Authentication”. Defines the TLS-SRP ciphersuites.
- RFC 5081: “Using OpenPGP Keys for Transport Layer Security (TLS) Authentication”, obsoleted by RFC 6091.

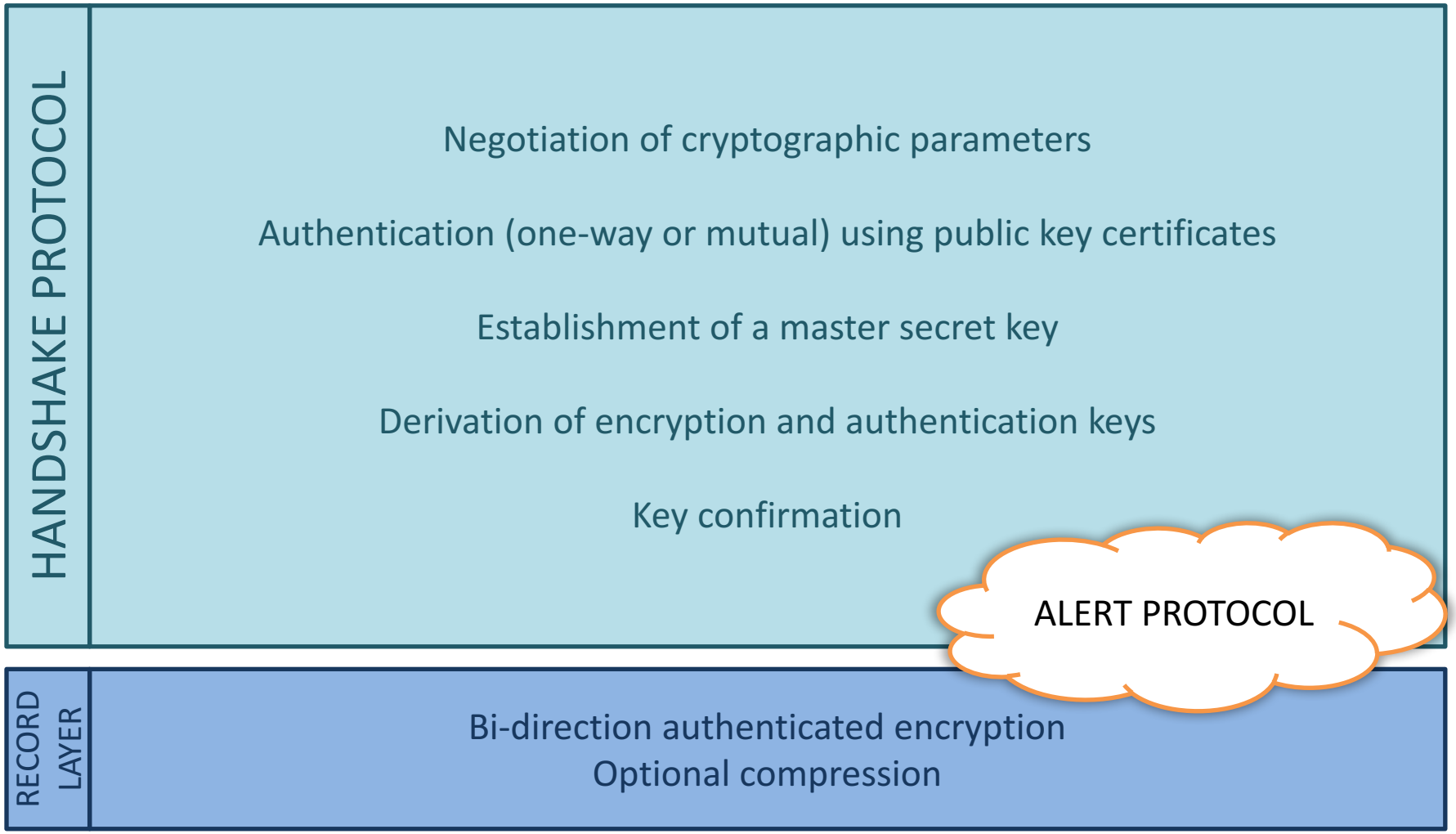
Extensions to TLS 1.2 include:

- RFC 5746: “Transport Layer Security (TLS) Renegotiation Indication Extension”.
- RFC 5878: “Transport Layer Security (TLS) Authorization Extensions”.
- RFC 6091: “Using OpenPGP Keys for Transport Layer Security (TLS) Authentication“.
- RFC 6176: “Prohibiting Secure Sockets Layer (SSL) Version 2.0”.
- RFC 6209: “Addition of the ARIA Cipher Suites to Transport Layer Security (TLS)”.

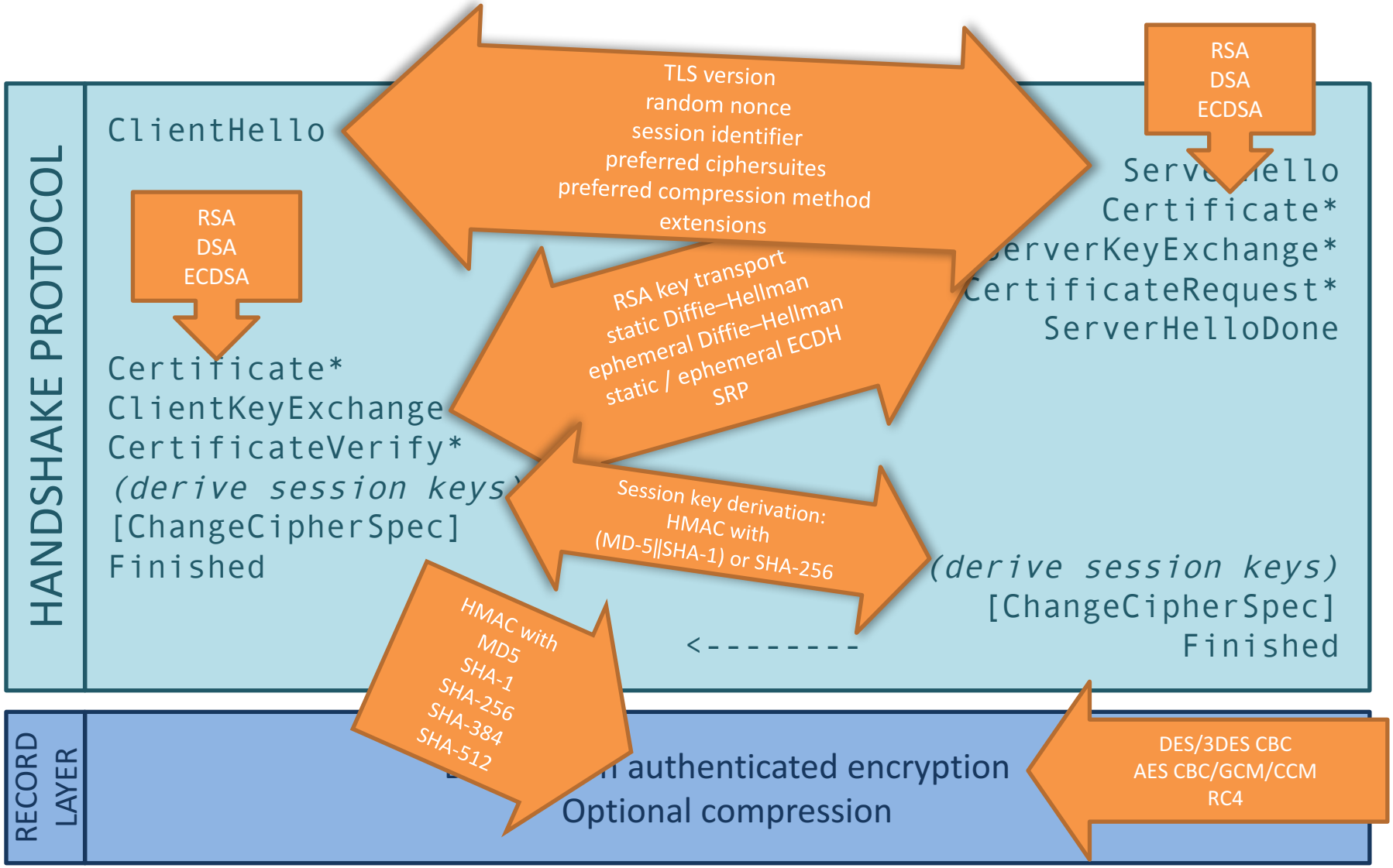
What is TLS?

http://en.wikipedia.org/wiki/Transport_Layer_Security

Structure of TLS



Structure of TLS



TLS_NULL_WITH_NULL_NULL TLS_RSA_WITH_NULL_MD5 TLS_RSA_WITH_NULL_SHA TLS_RSA_EXPORT_WITH_RC4_40_MD5 TLS_RSA_WITH_RC4_128_MD5 TLS_RSA_WITH_RC4_128_SHA TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
TLS_RSA_WITH_IDEA_CBC_SHA TLS_RSA_EXPORT_WITH_DES40_CBC_SHA TLS_RSA_WITH_DES_CBC_SHA TLS_RSA_WITH_3DES_EDE_CBC_SHA TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA TLS_DH_DSS_WITH_DES_CBC_SHA
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA TLS_DH_RSA_WITH_DES_CBC_SHA TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA
TLS_DHE_DSS_WITH_DES_CBC_SHA TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA TLS_DHE_RSA_WITH_DES_CBC_SHA TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DH_anon_EXPORT_WITH_RC4_40_MD5 TLS_DH_anon_WITH_RC4_128_MD5 TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA TLS_DH_anon_WITH_DES_CBC_SHA TLS_DH_anon_WITH_3DES_EDE_CBC_SHA
TLS_KRB5_WITH_DES_CBC_SHA TLS_KRB5_WITH_3DES_EDE_CBC_SHA TLS_KRB5_WITH_RC4_128_SHA TLS_KRB5_WITH_IDEA_CBC_SHA TLS_KRB5_WITH_DES_CBC_MD5 TLS_KRB5_WITH_3DES_EDE_CBC_MD5
TLS_KRB5_WITH_RC4_128_MD5 TLS_KRB5_WITH_IDEA_CBC_MD5 TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA TLS_KRB5_EXPORT_WITH_RC2_CBC_40_SHA TLS_KRB5_EXPORT_WITH_RC4_40_SHA
TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5 TLS_KRB5_EXPORT_WITH_RC2_CBC_40_MD5 TLS_KRB5_EXPORT_WITH_RC4_40_MD5 TLS_PSK_WITH_NULL_SHA TLS_DHE_PSK_WITH_NULL_SHA TLS_RSA_PSK_WITH_NULL_SHA
TLS_RSA_WITH_AES_128_CBC_SHA TLS_DH_DSS_WITH_AES_128_CBC_SHA TLS_DH_RSA_WITH_AES_128_CBC_SHA TLS_DHE_DSS_WITH_AES_128_CBC_SHA TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DH_anon_WITH_AES_128_CBC_SHA TLS_RSA_WITH_AES_256_CBC_SHA TLS_DH_DSS_WITH_AES_256_CBC_SHA TLS_DH_RSA_WITH_AES_256_CBC_SHA TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA TLS_DH_anon_WITH_AES_256_CBC_SHA TLS_RSA_WITH_NULL_SHA256 TLS_RSA_WITH_AES_128_CBC_SHA256 TLS_RSA_WITH_AES_256_CBC_SHA256 TLS_DH_DSS_WITH_AES_128_CBC_SHA256
TLS_DH_RSA_WITH_AES_128_CBC_SHA256 TLS_DHE_DSS_WITH_AES_128_CBC_SHA256 TLS_RSA_WITH_CAMELLIA_128_CBC_SHA TLS_DH_DSS_WITH_CAMELLIA_128_CBC_SHA TLS_DH_RSA_WITH_CAMELLIA_128_CBC_SHA
TLS_DHE_DSS_WITH_CAMELLIA_128_CBC_SHA TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA TLS_DH_anon_WITH_CAMELLIA_128_CBC_SHA TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 TLS_DH_DSS_WITH_AES_256_CBC_SHA256
TLS_DH_RSA_WITH_AES_256_CBC_SHA256 TLS_DHE_DSS_WITH_AES_256_CBC_SHA256 TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 TLS_DH_anon_WITH_AES_128_CBC_SHA256 TLS_DH_anon_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_CAMELLIA_256_CBC_SHA TLS_DH_DSS_WITH_CAMELLIA_256_CBC_SHA TLS_DH_RSA_WITH_CAMELLIA_256_CBC_SHA TLS_DHE_DSS_WITH_CAMELLIA_256_CBC_SHA TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA
TLS_DH_anon_WITH_CAMELLIA_256_CBC_SHA TLS_PSK_WITH_RC4_128_SHA TLS_PSK_WITH_3DES_EDE_CBC_SHA TLS_PSK_WITH_AES_128_CBC_SHA TLS_PSK_WITH_AES_256_CBC_SHA TLS_DHE_PSK_WITH_RC4_128_SHA
TLS_DHE_PSK_WITH_3DES_EDE_CBC_SHA TLS_DHE_PSK_WITH_AES_128_CBC_SHA TLS_DHE_PSK_WITH_AES_256_CBC_SHA TLS_RSA_PSK_WITH_RC4_128_SHA TLS_RSA_PSK_WITH_3DES_EDE_CBC_SHA
TLS_RSA_PSK_WITH_AES_128_CBC_SHA TLS_RSA_PSK_WITH_AES_256_CBC_SHA TLS_RSA_WITH_SEED_CBC_SHA TLS_DH_DSS_WITH_SEED_CBC_SHA TLS_DH_RSA_WITH_SEED_CBC_SHA TLS_DHE_DSS_WITH_SEED_CBC_SHA
TLS_DHE_RSA_WITH_SEED_CBC_SHA TLS_DH_anon_WITH_SEED_CBC_SHA TLS_RSA_WITH_AES_128_GCM_SHA256 TLS_RSA_WITH_AES_256_GCM_SHA384 TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 TLS_DH_RSA_WITH_AES_128_GCM_SHA256 TLS_DH_RSA_WITH_AES_256_GCM_SHA384 TLS_DHE_DSS_WITH_AES_128_GCM_SHA256 TLS_DHE_DSS_WITH_AES_256_GCM_SHA384
TLS_DH_DSS_WITH_AES_128_GCM_SHA256 TLS_DH_DSS_WITH_AES_256_GCM_SHA384 TLS_DH_anon_WITH_AES_128_GCM_SHA256 TLS_DH_anon_WITH_AES_256_GCM_SHA384 TLS_PSK_WITH_AES_128_GCM_SHA256
TLS_PSK_WITH_AES_256_GCM_SHA384 TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 TLS_RSA_PSK_WITH_AES_128_GCM_SHA256 TLS_RSA_PSK_WITH_AES_256_GCM_SHA384
TLS_PSK_WITH_AES_128_CBC_SHA256 TLS_PSK_WITH_AES_256_CBC_SHA384 TLS_PSK_WITH_NULL_SHA256 TLS_PSK_WITH_NULL_SHA384 TLS_DHE_PSK_WITH_AES_128_CBC_SHA256 TLS_DHE_PSK_WITH_AES_256_CBC_SHA384
TLS_DHE_PSK_WITH_NULL_SHA256 TLS_DHE_PSK_WITH_NULL_SHA384 TLS_RSA_PSK_WITH_AES_128_CBC_SHA256 TLS_RSA_PSK_WITH_AES_256_CBC_SHA384 TLS_RSA_PSK_WITH_NULL_SHA256 TLS_RSA_PSK_WITH_NULL_SHA384
TLS_RSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_DH_DSS_WITH_CAMELLIA_128_CBC_SHA256 TLS_DH_RSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_DHE_DSS_WITH_CAMELLIA_128_CBC_SHA256
TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_DH_anon_WITH_CAMELLIA_128_CBC_SHA256 TLS_RSA_WITH_CAMELLIA_256_CBC_SHA256 TLS_DH_DSS_WITH_CAMELLIA_256_CBC_SHA256
TLS_DH_RSA_WITH_CAMELLIA_256_CBC_SHA256 TLS_DHE_DSS_WITH_CAMELLIA_256_CBC_SHA256 TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA256 TLS_DH_anon_WITH_CAMELLIA_256_CBC_SHA256
TLS_EMPTY_RENEGOTIATION_INFO_SCSV TLS_ECDH_ECDSA_WITH_NULL_SHA TLS_ECDH_ECDSA_WITH_RC4_128_SHA TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA TLS_ECDHE_ECDSA_WITH_NULL_SHA TLS_ECDHE_ECDSA_WITH_RC4_128_SHA TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA TLS_ECDH_RSA_WITH_NULL_SHA TLS_ECDH_RSA_WITH_RC4_128_SHA TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA TLS_ECDHE_RSA_WITH_NULL_SHA TLS_ECDHE_RSA_WITH_RC4_128_SHA TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA TLS_ECDH_anon_WITH_NULL_SHA TLS_ECDH_anon_WITH_RC4_128_SHA TLS_ECDH_anon_WITH_3DES_EDE_CBC_SHA TLS_ECDH_anon_WITH_AES_128_CBC_SHA
TLS_ECDH_anon_WITH_AES_256_CBC_SHA TLS_SRP_SHA_WITH_3DES_EDE_CBC_SHA TLS_SRP_SHA_RSA_WITH_3DES_EDE_CBC_SHA TLS_SRP_SHA_DSS_WITH_3DES_EDE_CBC_SHA TLS_SRP_SHA_WITH_AES_128_CBC_SHA
TLS_SRP_SHA_RSA_WITH_AES_128_CBC_SHA TLS_SRP_SHA_DSS_WITH_AES_128_CBC_SHA TLS_SRP_SHA_WITH_AES_256_CBC_SHA TLS_SRP_SHA_RSA_WITH_AES_256_CBC_SHA TLS_SRP_SHA_DSS_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256 TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256 TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256 TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256 TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384 TLS_ECDHE_PSK_WITH_RC4_128_SHA
TLS_ECDHE_PSK_WITH_3DES_EDE_CBC_SHA TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384
TLS_ECDHE_PSK_WITH_NULL_SHA TLS_ECDHE_PSK_WITH_NULL_SHA256 TLS_ECDHE_PSK_WITH_NULL_SHA384 TLS_RSA_WITH_ARIA_128_CBC_SHA256 TLS_RSA_WITH_ARIA_256_CBC_SHA384
TLS_DH_DSS_WITH_ARIA_128_CBC_SHA256 TLS_DH_DSS_WITH_ARIA_256_CBC_SHA384 TLS_DH_RSA_WITH_ARIA_128_CBC_SHA256 TLS_DH_RSA_WITH_ARIA_256_CBC_SHA384 TLS_DHE_DSS_WITH_ARIA_128_CBC_SHA256
TLS_DHE_DSS_WITH_ARIA_256_CBC_SHA384 TLS_DHE_RSA_WITH_ARIA_128_CBC_SHA256 TLS_DHE_RSA_WITH_ARIA_256_CBC_SHA384 TLS_DH_anon_WITH_ARIA_128_CBC_SHA256 TLS_DH_anon_WITH_ARIA_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_ARIA_128_CBC_SHA256 TLS_ECDHE_ECDSA_WITH_ARIA_256_CBC_SHA384 TLS_ECDH_ECDSA_WITH_ARIA_128_CBC_SHA256 TLS_ECDH_ECDSA_WITH_ARIA_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_ARIA_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_ARIA_256_CBC_SHA384 TLS_ECDH_RSA_WITH_ARIA_128_CBC_SHA256 TLS_ECDH_RSA_WITH_ARIA_256_CBC_SHA384 TLS_RSA_WITH_ARIA_128_GCM_SHA256
TLS_RSA_WITH_ARIA_256_GCM_SHA384 TLS_DHE_RSA_WITH_ARIA_128_GCM_SHA256 TLS_DHE_RSA_WITH_ARIA_256_GCM_SHA384 TLS_DH_RSA_WITH_ARIA_128_GCM_SHA256 TLS_DH_RSA_WITH_ARIA_256_GCM_SHA384
TLS_DHE_DSS_WITH_ARIA_128_GCM_SHA256 TLS_DHE_DSS_WITH_ARIA_256_GCM_SHA384 TLS_DH_DSS_WITH_ARIA_128_GCM_SHA256 TLS_DH_DSS_WITH_ARIA_256_GCM_SHA384 TLS_DH_anon_WITH_ARIA_128_GCM_SHA256
TLS_DH_anon_WITH_ARIA_256_GCM_SHA384 TLS_ECDHE_ECDSA_WITH_ARIA_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_ARIA_256_GCM_SHA384 TLS_ECDH_ECDSA_WITH_ARIA_128_GCM_SHA256
TLS_ECDH_ECDSA_WITH_ARIA_256_GCM_SHA384 TLS_ECDHE_RSA_WITH_ARIA_128_GCM_SHA256 TLS_ECDHE_RSA_WITH_ARIA_256_GCM_SHA384 TLS_ECDH_RSA_WITH_ARIA_128_GCM_SHA256
TLS_ECDH_RSA_WITH_ARIA_256_GCM_SHA384 TLS_PSK_WITH_ARIA_128_CBC_SHA256 TLS_PSK_WITH_ARIA_256_CBC_SHA384 TLS_DHE_PSK_WITH_ARIA_128_CBC_SHA256 TLS_DHE_PSK_WITH_ARIA_256_CBC_SHA384
TLS_RSA_PSK_WITH_ARIA_128_CBC_SHA256 TLS_RSA_PSK_WITH_ARIA_256_CBC_SHA384 TLS_PSK_WITH_ARIA_128_GCM_SHA256 TLS_PSK_WITH_ARIA_256_GCM_SHA384 TLS_DHE_PSK_WITH_ARIA_128_GCM_SHA256
TLS_DHE_PSK_WITH_ARIA_256_GCM_SHA384 TLS_RSA_PSK_WITH_ARIA_128_GCM_SHA256 TLS_RSA_PSK_WITH_ARIA_256_GCM_SHA384 TLS_ECDHE_PSK_WITH_ARIA_128_CBC_SHA256 TLS_ECDHE_PSK_WITH_ARIA_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_ECDHE_ECDSA_WITH_CAMELLIA_256_CBC_SHA384 TLS_ECDH_ECDSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_ECDH_ECDSA_WITH_CAMELLIA_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_ECDHE_RSA_WITH_CAMELLIA_256_CBC_SHA384 TLS_ECDH_RSA_WITH_CAMELLIA_128_CBC_SHA256 TLS_ECDH_RSA_WITH_CAMELLIA_256_CBC_SHA384
TLS_RSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_RSA_WITH_CAMELLIA_256_GCM_SHA384 TLS_DHE_RSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_DHE_RSA_WITH_CAMELLIA_256_GCM_SHA384
TLS_DH_RSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_DH_RSA_WITH_CAMELLIA_256_GCM_SHA384 TLS_DHE_DSS_WITH_CAMELLIA_128_GCM_SHA256 TLS_DHE_DSS_WITH_CAMELLIA_256_GCM_SHA384
TLS_DH_DSS_WITH_CAMELLIA_128_GCM_SHA256 TLS_DH_DSS_WITH_CAMELLIA_256_GCM_SHA384 TLS_DH_anon_WITH_CAMELLIA_128_GCM_SHA256 TLS_DH_anon_WITH_CAMELLIA_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_ECDHE_ECDSA_WITH_CAMELLIA_256_GCM_SHA384 TLS_ECDH_ECDSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_ECDH_ECDSA_WITH_CAMELLIA_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_ECDHE_RSA_WITH_CAMELLIA_256_GCM_SHA384 TLS_ECDH_RSA_WITH_CAMELLIA_128_GCM_SHA256 TLS_ECDH_RSA_WITH_CAMELLIA_256_GCM_SHA384
TLS_PSK_WITH_CAMELLIA_128_GCM_SHA256 TLS_PSK_WITH_CAMELLIA_256_GCM_SHA384 TLS_DHE_PSK_WITH_CAMELLIA_128_GCM_SHA256 TLS_DHE_PSK_WITH_CAMELLIA_256_GCM_SHA384
TLS_RSA_PSK_WITH_CAMELLIA_128_GCM_SHA256 TLS_RSA_PSK_WITH_CAMELLIA_256_GCM_SHA384 TLS_PSK_WITH_CAMELLIA_128_CBC_SHA256 TLS_PSK_WITH_CAMELLIA_256_CBC_SHA384
TLS_DHE_PSK_WITH_CAMELLIA_128_CBC_SHA256 TLS_DHE_PSK_WITH_CAMELLIA_256_CBC_SHA384 TLS_RSA_PSK_WITH_CAMELLIA_128_CBC_SHA256 TLS_RSA_PSK_WITH_CAMELLIA_256_CBC_SHA384
TLS_ECDHE_PSK_WITH_CAMELLIA_128_CBC_SHA256 TLS_ECDHE_PSK_WITH_CAMELLIA_256_CBC_SHA384 TLS_RSA_WITH_AES_128_CCM TLS_RSA_WITH_AES_256_CCM TLS_DHE_RSA_WITH_AES_128_CCM
TLS_DHE_RSA_WITH_AES_256_CCM TLS_RSA_WITH_AES_128_CCM_8 TLS_RSA_WITH_AES_256_CCM_8 TLS_DHE_RSA_WITH_AES_128_CCM_8 TLS_DHE_RSA_WITH_AES_256_CCM_8 TLS_PSK_WITH_AES_128_CCM
TLS_PSK_WITH_AES_256_CCM TLS_DHE_PSK_WITH_AES_128_CCM TLS_DHE_PSK_WITH_AES_256_CCM TLS_PSK_WITH_AES_128_CCM_8 TLS_PSK_WITH_AES_256_CCM_8 TLS_PSK_DHE_WITH_AES_128_CCM_8
TLS_PSK_DHE_WITH_AES_256_CCM_8

324 TLS ciphersuites

Is TLS secure?

What should TLS do?

- Server-to-client authentication
- Client-to-server authentication (optional)
- Confidential communication with integrity protection

What doesn't TLS do?

- (Trusted creation of certificates)
- Password-based authentication
- Stop denial of service attacks
- Prevent web application vulnerabilities

Components of TLS

- RSA, DSA, ECDSA
- Diffie–Hellman, ECDH
- HMAC
- MD5, SHA1, SHA-2
- DES, 3DES, RC4, AES

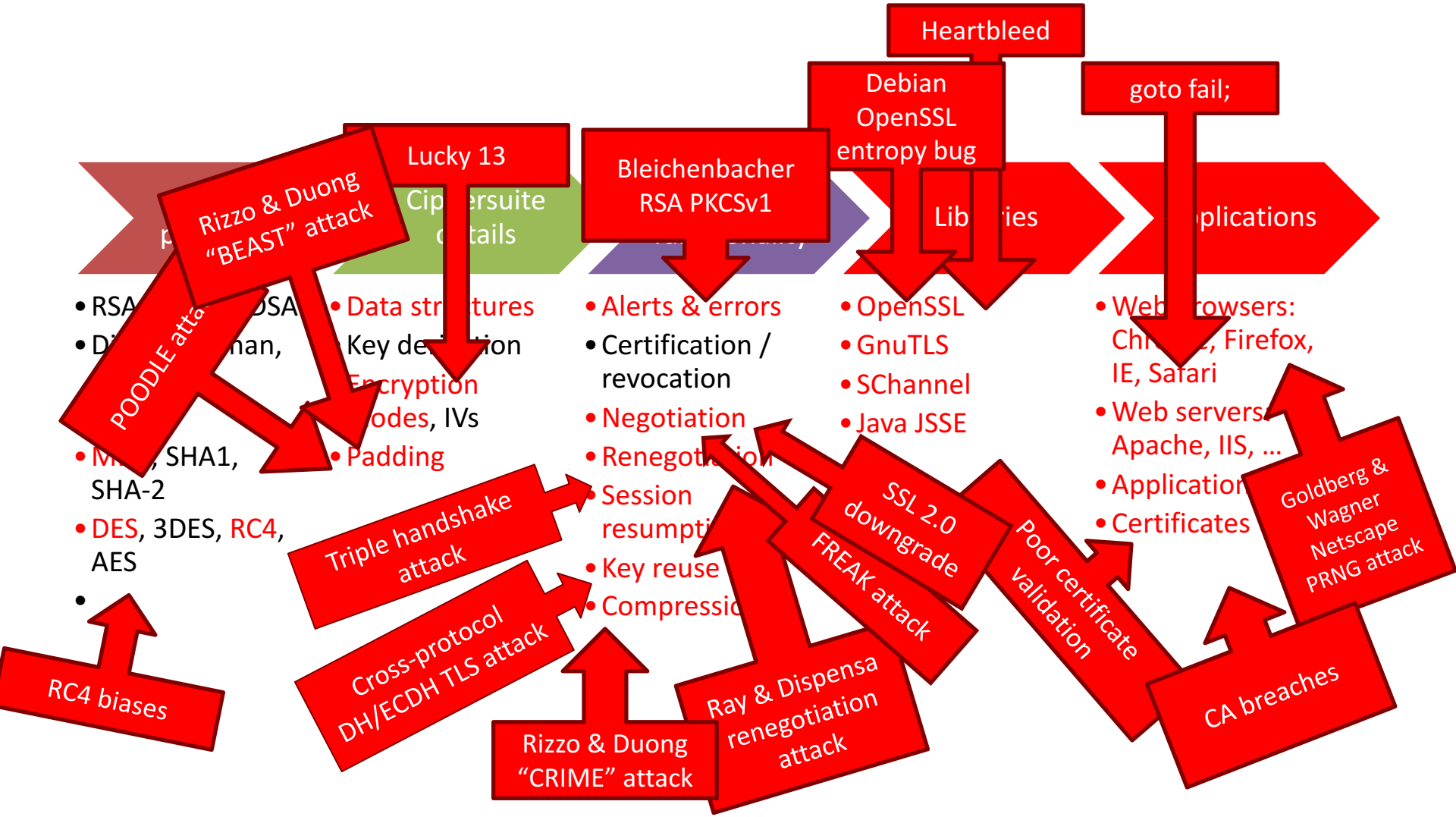
- Data structures
- Key derivation
- Encryption modes, IVs
- Padding

- Alerts & errors
- Certification / revocation
- Negotiation
- Renegotiation
- Session resumption
- Key reuse
- Compression

- OpenSSL
- GnuTLS
- SChannel
- Java JSSE

- Web browsers: Chrome, Firefox, IE, Safari
- Web servers: Apache, IIS, …
- Application SDKs
- Certificates

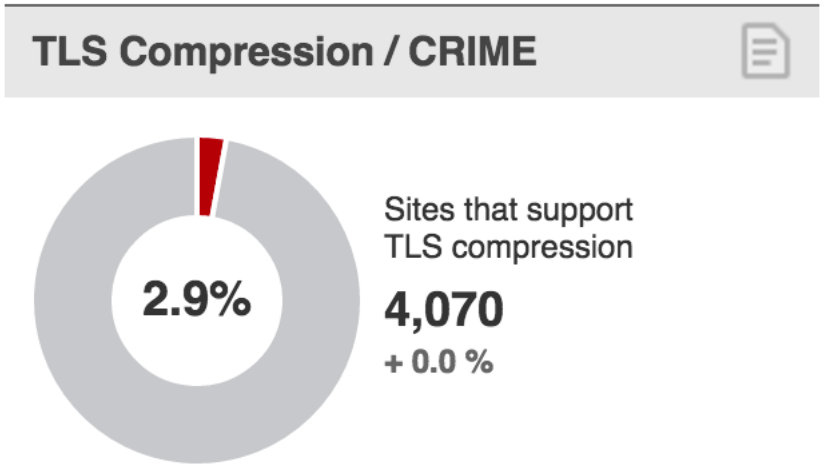
Real-world attacks on TLS



Esoteric features – compression

- TLS supports optional compression
- Message broken up into chunks, each chunk compressed then encrypted

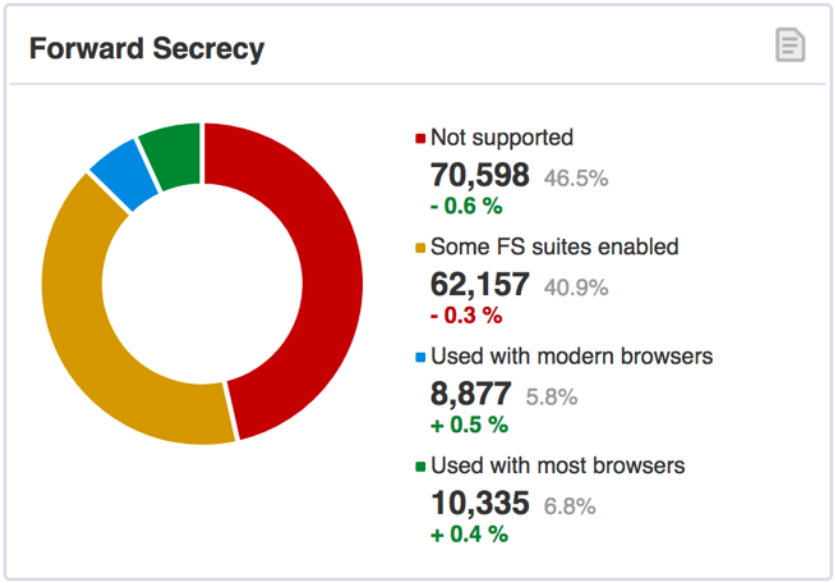
- Size of ciphertext
  => amount of compression
  => leaks plaintext info
- “CRIME” attack, Sept. 2012
- Fix: disable compression



https://www.trustworthyinternet.org/ssl-pulse/ June 1, 2016

(Perfect) Forward secrecy

- An adversary who later learns the server's long-term private key shouldn't be able to read previous transmissions

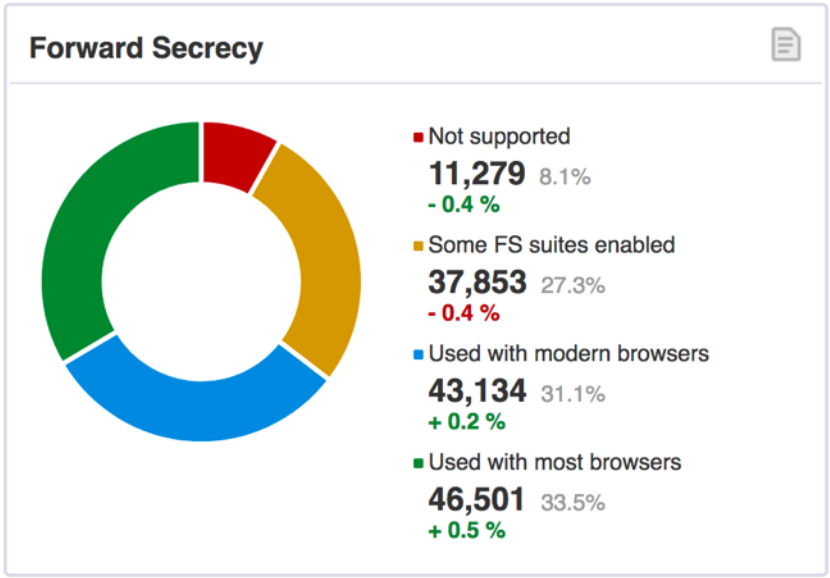


- RSA key transport: no PFS
- signed Diffie–Hellman: PFS

https://www.trustworthyinternet.org/ssl-pulse/ September 4, 2014

(Perfect) Forward secrecy

- An adversary who later learns the server's long-term private key shouldn't be able to read previous transmissions



- RSA key transport: no PFS
- signed Diffie–Hellman: PFS

https://www.trustworthyinternet.org/ssl-pulse/ July 2, 2017

Certificate authority breaches and errors

- DigiNotar in Jul. 2011
  - security breach, malicious certificates for many domains issued
  - went out of business
- TURKTRUST in Aug. 2011
  - issued intermediate CA with wildcard signing capabilities
  - later used for man-in-the-middle proxy filtering/scanning
  - no evidence for use in attack
  - detected only in Jan 2013
- Digicert Malaysia in Nov. 2011
  - 22 certificates with weak private keys or missing revocation details issued
- KPN/Getronics in Nov. 2011
  - suspended CA business after detecting infection on its web server no evidence of certificate malfeasance

- Web browsers trust 650+ certificate authorities which can issue certificates for any domain on the Internet
- Extended validation certificates don't solve the problem

Cryptographic attacks

- Many weaknesses found in record layer symmetric encryption algorithms.
- Usually require large amount of data to succeed.
- Not urgent to fix, but cryptographic attacks only get better, never worse.
- Can be confusing as knowledge evolves:
  - BEAST attack 2011 => AES-CBC mode insecure, use RC4
  - Paterson et al. attack 2013 => RC4 insecure, use fixed AES-CBC

TLSv1.3: The Next Generation

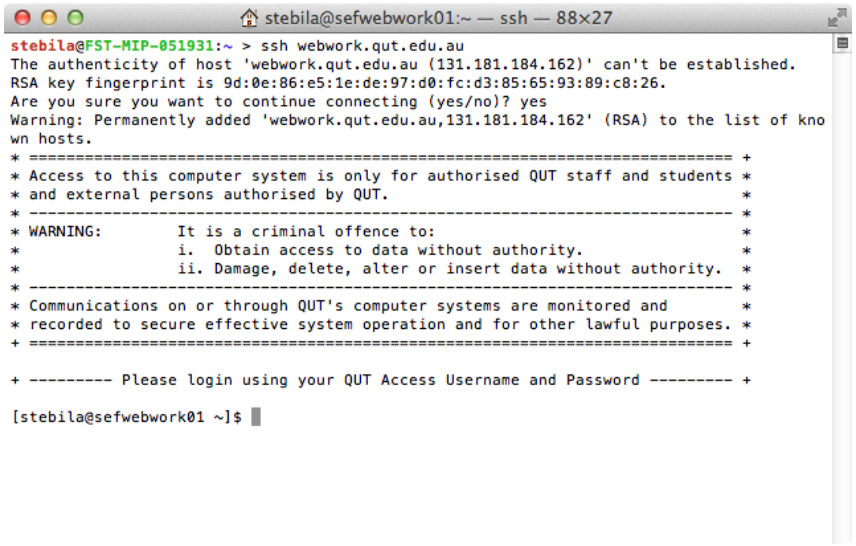
- Currently under development at the IETF

- Primary goals:
  - remove ciphersuites without forward secrecy
  - remove obsolete / deprecated algorithms
  - provide low-latency mode with fewer round trips
  - encrypt more of the handshake to improve privacy

SSH, IPsec, Kerberos

OTHER PROTOCOLS

SSH (Secure Shell) protocol



- SSH used for secure remote access (like telnet, but secure)
- Provides public key authentication of servers and clients and encrypted communication
- Specified in RFCs by the IETF

Use of SSH

- Primarily used as an application itself (remote login)
- Occasionally used as a "poor man's VPN"

Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

SSH security services

- Message Confidentiality.
  - Protects against unauthorised data disclosure.
  - Accomplished by the use of encryption mechanisms.
- Message Integrity.
  - SSH can determine if data has been changed (intentionally or unintentionally) during transit.
  - Integrity of data can be assured by using a message authentication code (MAC).
- Message Replay Protection.
  - The same data is not delivered multiple times.
- Peer Authentication.
  - Server to client authentication based on public keys
  - Client to server authentication based on passwords or public keys
  - Ensures that network traffic is being sent from the expected party.

Client authentication in SSH

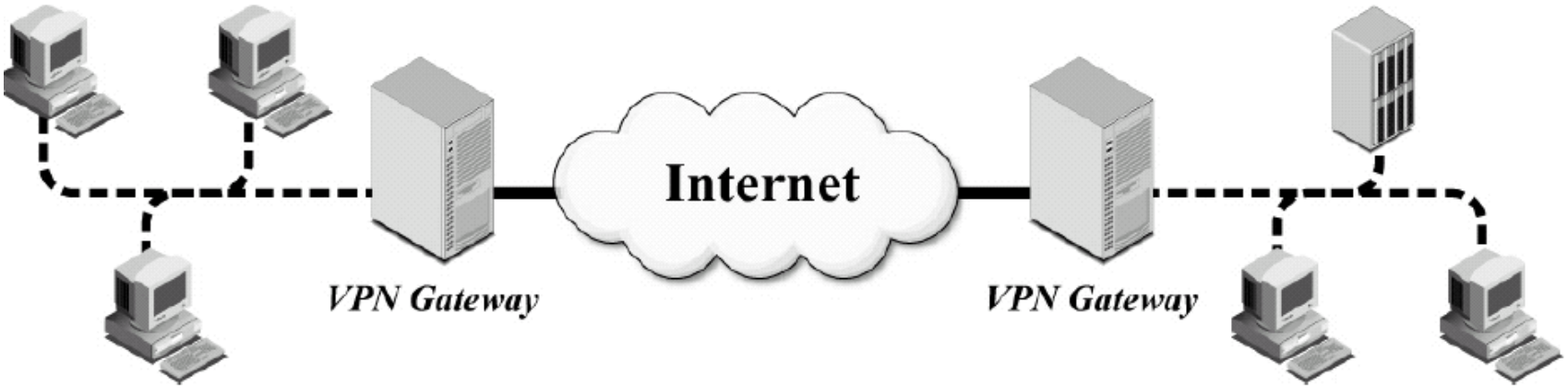
- **SSH** (Secure Shell) is often used for remote command-line access in Unix and Mac OS X.
- It supports public key authentication.
  - A security-conscious SSH installation would support **only** public key authentication and disable password-based authentication.
- Each account can have multiple associated public keys.
  - Multiple users can login to a single account without having to be told the password for that account. Easy to revoke one user's access to that account.
  - One user could have a different key from each local computer (laptop, desktop, ...); if one of local computer is lost/compromised, easy to revoke its access.
- Users can associate the same key with multiple accounts.
  - Yields a form of single sign-on.
  - Users can protect their private key using a password.

IPsec (Internet Protocol Security)

- Provides confidentiality and authentication for Internet communications
- Works at the IP layer of the protocol stack
  - TLS works at higher levels, so applications have to be designed to use TLS
  - IPsec can be used transparently with any application
- Often used for Virtual Private Networks (VPNs)

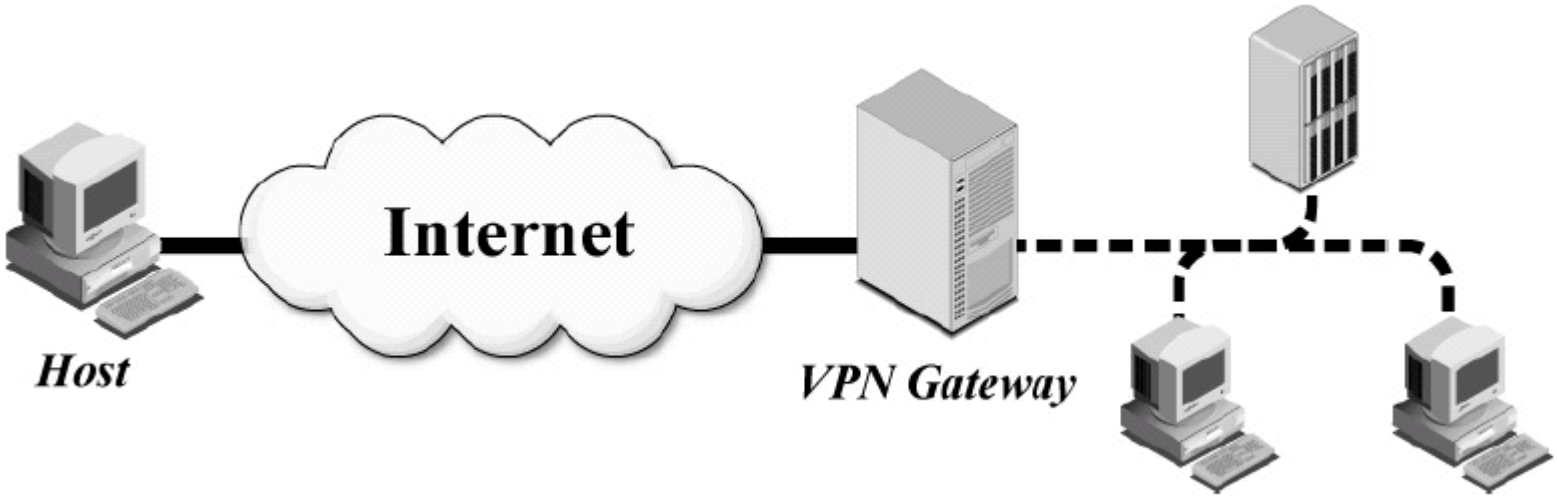
Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

IPsec: Common Architectures
Gateway-to-gateway



Source: NIST Special Publication 800-77

IPsec: Common Architectures
Host-to-gateway



Source: NIST Special Publication 800-77

Single sign-on

- **Single sign-on protocols** allow a user to use a credential from one identity provider with many relying parties.
- The user's authentication to the identity provider can be based on any form(s) of authentication: password, public key, biometric.
- The identity provider gives an assertion to the **relying party** stating who the user is.

Kerberos

- Protocol for cryptographic authentication of users and services on distributed systems
- Authentication based on knowledge of shared secrets
- Uses symmetric authentication
- Relies on a trusted third party to mediate authentication between parties

- Developed by MIT in the 1980s and 1990s
- Kerberos version 5 published in 1993
- Kerberos used as default authentication method in Windows 2000 and later as part of Active Directory services
- Most UNIX-based operating systems (Linux, Solaris, Mac OS X, FreeBSD) support Kerberos authetication of users and services
- Kerberos also supports public key (asymmetric) authentication

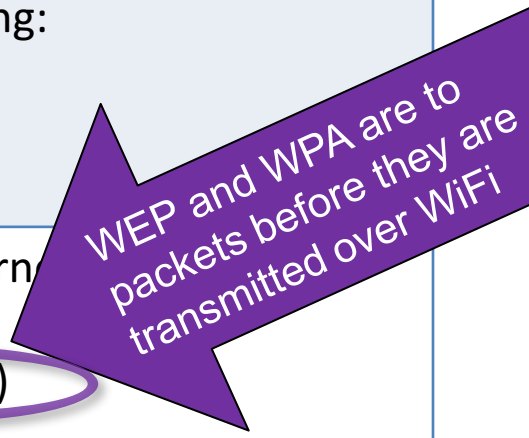
WIRED EQUIVALENT PRIVACY (WEP)

Wired Equivalent Privacy (WEP)

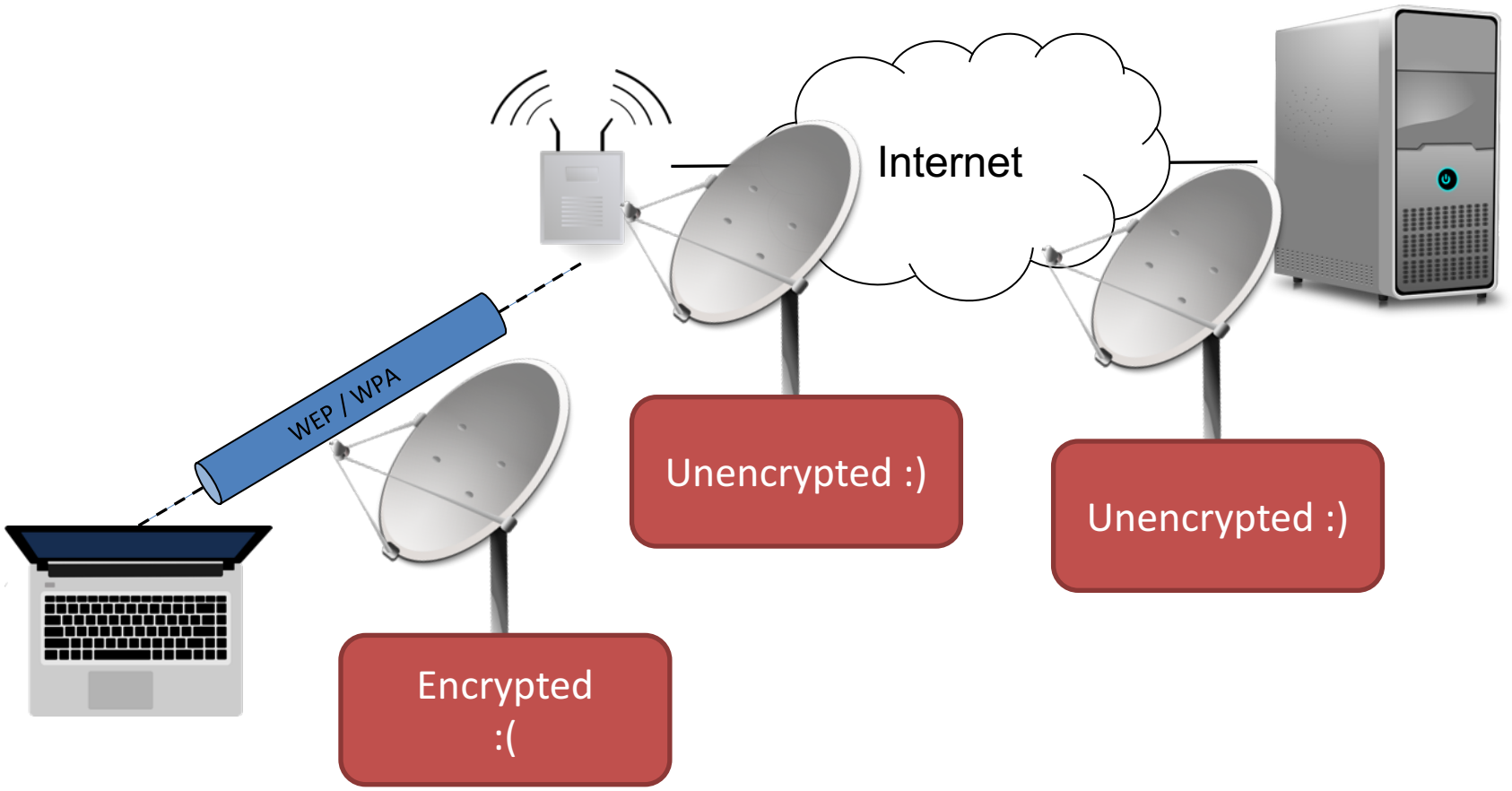
- **Goal:** provide login authentication, message authentication/integrity, and message confidentiality for 802.11 wireless networks
  - standardized in 1999
  - uses RC4 for encryption
  - uses CRC-32 checksum for integrity
  - uses pre-established shared keys

IETF Internet Protocol suite

Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethern physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

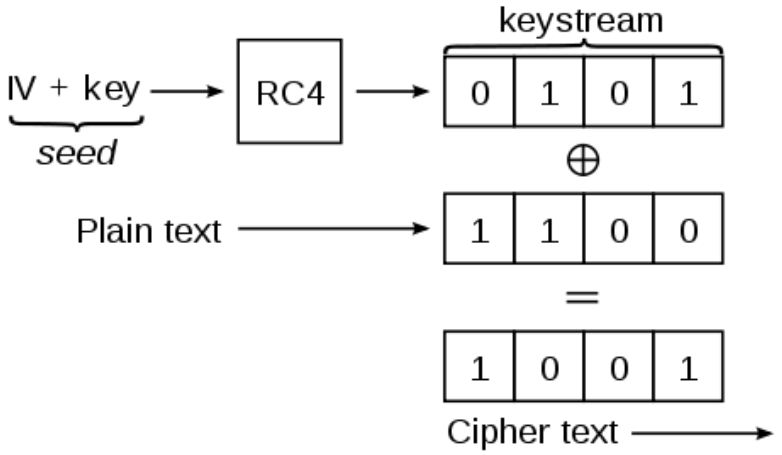


WiFi Security



WEP Encryption

- 64-bit WEP uses a 24-bit IV and a 40-bit key
- 128-bit WEP uses a 24-bit IV and a 104-bit key



- Append CRC-32(m) to message before encrypting

WEP Login Authentication

To authenticate to a 802.11 network using WEP shared key authentication:

- Access point sends 128-bit challenge in plaintext
- Client picks an IV, uses IV,K to encrypt challenge (with RC4)
- Server decrypts challenge and compares

Is this secure?

- Attacker sees challenge plaintext, ciphertext
- Ciphertext = XOR of keystream and message,checksum
- Attacker can derive keystream from ciphertext and plaintext challenge
- Attacker can use keystream to encrypt another challenge and gain access

Wireless Security

WEP

- **Login authentication**: completely insecure; attacker can impersonate after seeing a single packet
- **Message authentication/integrity**: completely insecure; attacker can undetectably modify any packet with 100% success rate
- **Message confidentiality:** completely insecure; attacker can recover secret key with high probability in just a minute using readily available tools

- In 2007, US retailer TJ Maxx had 45 million customer credit cards stolen because their wireless network was secured using WEP
- WEP prohibited for use in credit card processing (PCI-DSS) after June 2010.

Wi-Fi Protected Access

- Wi-Fi Protected Access (WPA, WPA2) standardized in 2003
- WPA mostly still secure, as long as strong passwords are used
- Wi-Fi Protected Setup 8-digit PINs brute-forced in a few hours

IETF Internet Protocol suite

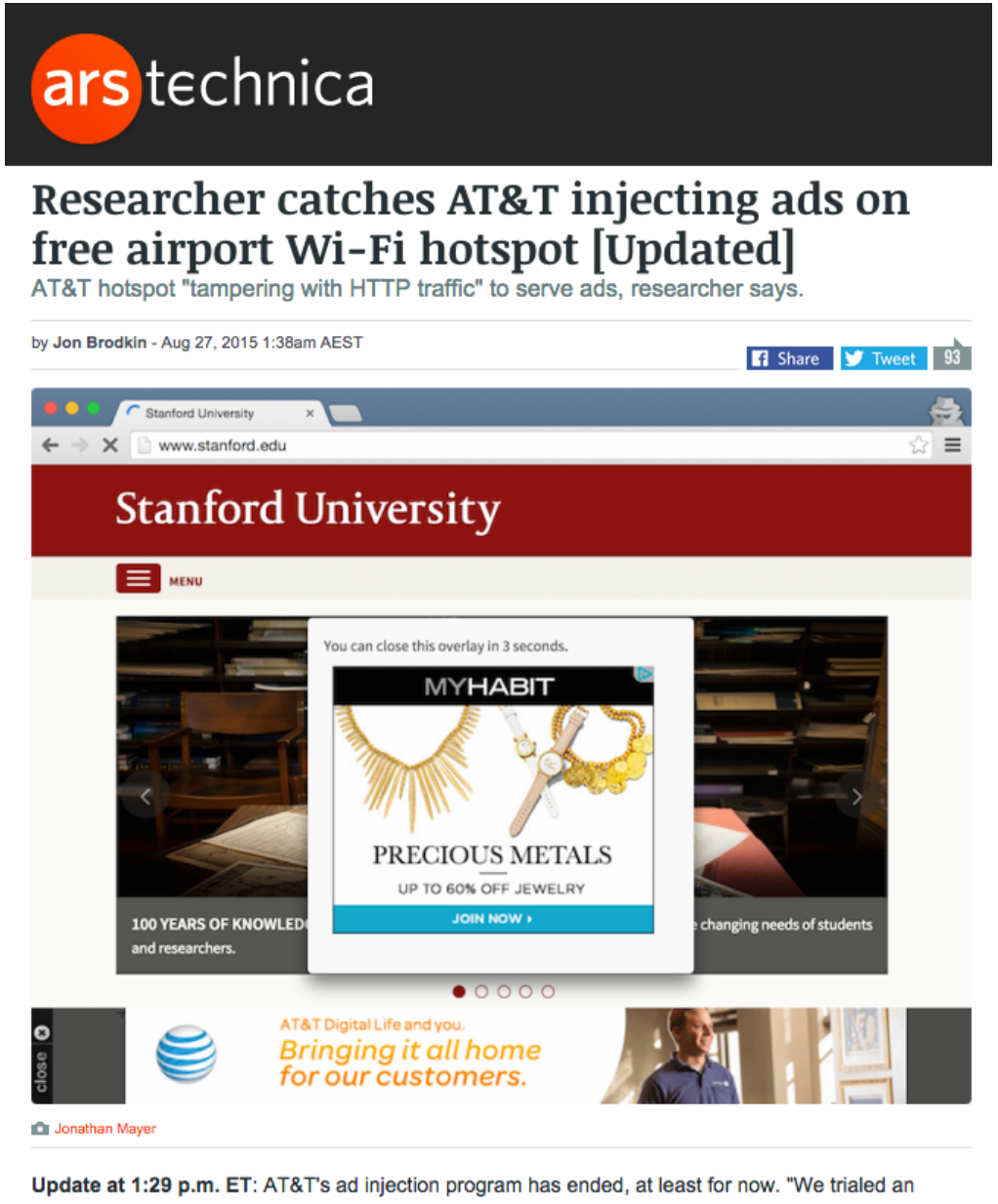
Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

Case study: Injecting ads in Wi-Fi hotspot

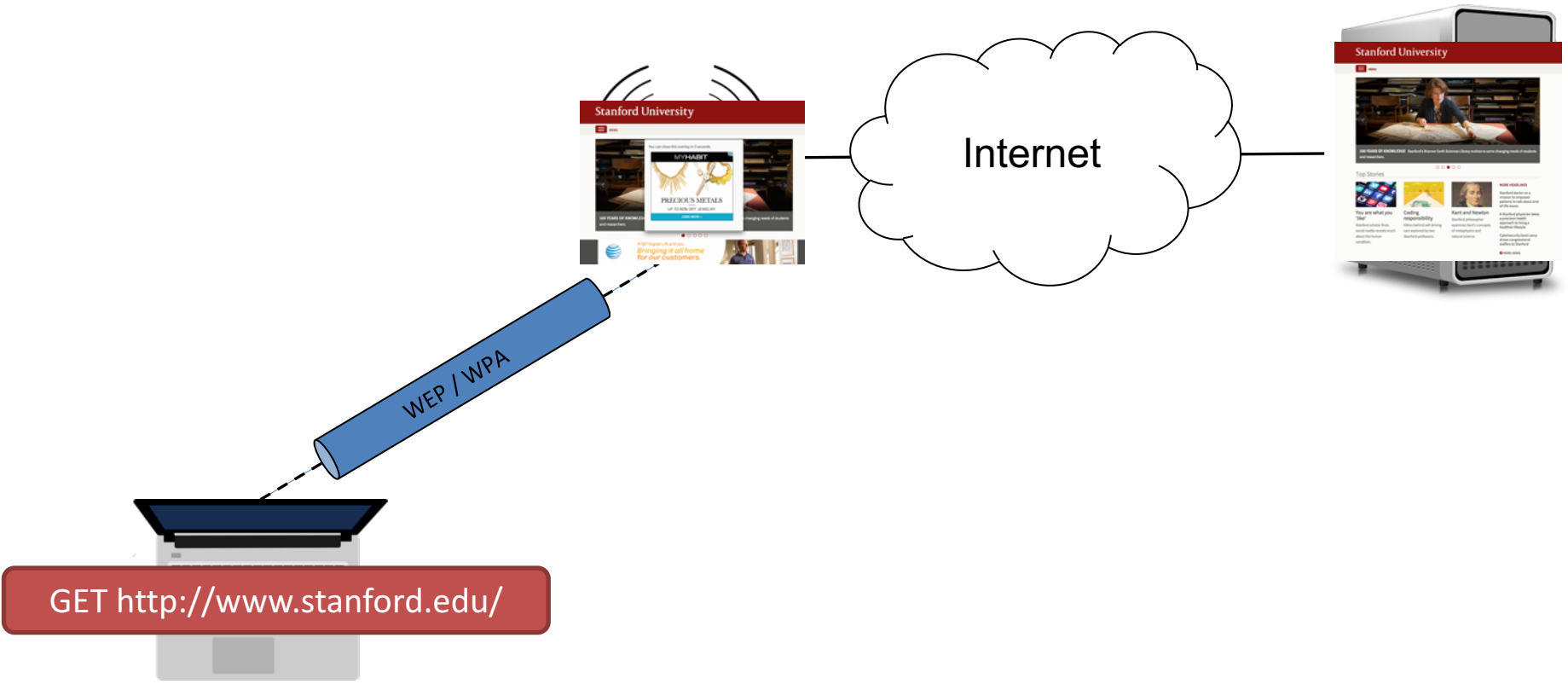
AT&T provides free Wi-Fi hotspots in airports.

In addition to making users view ads when they first connected to the hotspot, AT&T was also modifying HTTP responses from web servers to include their own ads on pages.

http://arstechnica.com/business/2015/08/atts-free-wi-fi-hotspot-injects-extra-ads-on-non-att-websites/



Case study:
Injecting ads in Wi-Fi hotspot



Case study: Injecting ads in Wi-Fi hotspot

Layer	Examples
Application	web (HTTP, HTTPS) email (SMTP, POP3, IMAP) login (SSH, Telnet)
Transport	connection-oriented (TCP) connectionless (UDP)
Internet	addressing and routing: • IPv4, IPv6 control (ICMP) security (IPsec)
Link	packet framing (Ethernet) physical connection • WLAN (WEP, WPA) • ADSL • GSM/3G

TLS

- Link-layer security would **not** protect against this attack
  - WEP/WPA
- Internet-layer, transport-layer, and application-layer would protect against this attack
  - IPsec: Use a VPN to a trusted gateway.
  - TLS: Encryption/integrity protection for web page connections.
  - SSH: Encryption/integrity protection for remote login.

Practicals

X.509 Certificates and Secure Email

- Using the XCA certificate authority software:
  - Set up a certificate authority
  - Generate a private key and certificate request for a user
  - Issue a certificate to the user
- Import the certificate into an email client and send a signed email

TLS

- Using your web browser, inspect the certificate used in a secure connection with a website
- Using the Wireshark packet sniffing software, inspect a TLS connection to see the protocol messages sent

Practicals available at https://www.douglas.stebila.ca/teaching/cryptoworks21