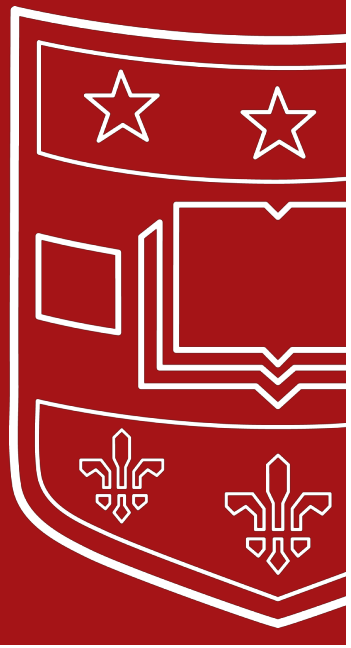
CSE 433S: Introduction to Computer Security

Network Defense Lecture

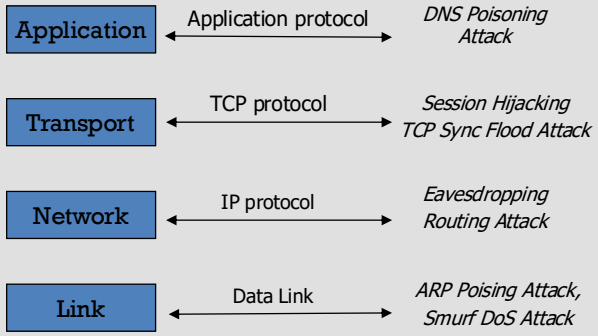


Washington University in St. Louis

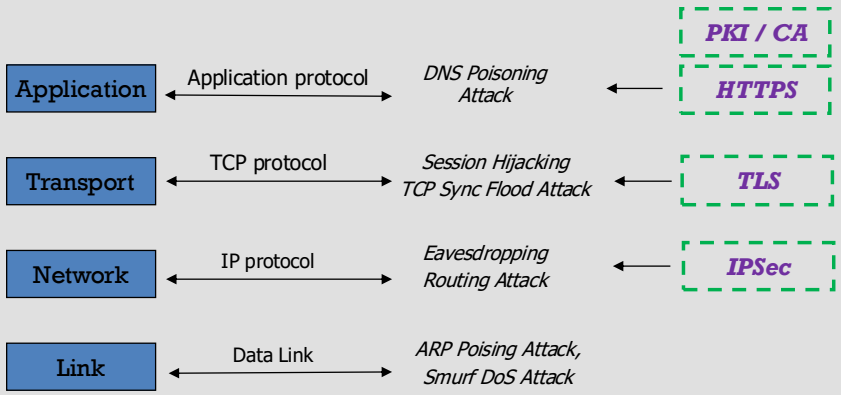
Review Questions

- How many symmetric keys does it take to support secure communications among 4 peers? What are the possible approaches to mitigate this issue?
- What is Diffie-Hellman key exchange, what attack is it vulnerable to, how do you launch that attack?
- What is public key crypto and how is different than symmetric key crypto? What key do Alice use if Alice wants to deliver a secret to Bob.
- In practice, can we use RSA to directly encrypt secret key for communication?
- What is digital signature? What key does Alice use to sign a file that she wants to authenticate and why?

Previously on Introduction to Computer Security – Network Attacks



This module - Network defense



Network Defense

- Public key infrastructure (PKI)
  - How do we know we are talking to the right entity on the web
- Certificate authority (CA)
  - Mechanisms to establish the web of trust
- IPSec
  - Enables confidentiality and integrity protection for communication between two network nodes at IP layer
- TLS (Transport Layer Security)
  - Enables security communication link over TCP

PKI/CA

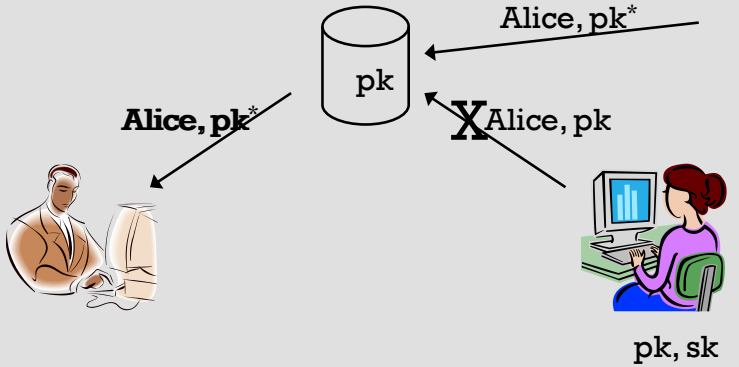
"Web of trust"

- Obtain public keys from friends in person
  - "Key-signing parties" from the PGP time
- Obtain "certificates" on my public key from my friends
- If A knows pk_B, and B issued a certificate for C, then C can send that certificate to A
  - What trust assumptions are being made here?

Public-key repository

- Store certificates in a central repository
  - E.g., MIT PGP keyserver
- To find Alice's public key
  - Get all public keys for "Alice," along with certificates on those keys
  - Look for a certificate signed by someone you trust whose public key you already have

Challenges on Public Key Repository

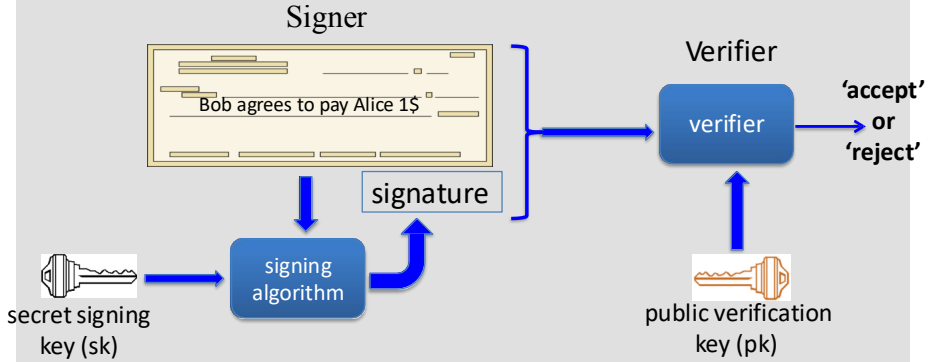


Public Key infrastructure

- Problem: How to determine the correct public key of a given entity
  - Binding between IDENTITY and PUBLIC KEY
- Possible attacks
  - Name spoofing: Eve associates Alice's name with Eve's public key
  - Key spoofing: Eve associates Alice's key with Eve's name
  - DoS: Eve associates Alice's name with a nonsensical (bogus) key

Digital signatures

Solution: make signature depend on document



Use signatures for secure key distribution!

- Assume a trusted party with a public key known to everyone
  - CA = certificate authority
  - Public key pk_{CA}
  - Private key sk_{CA}

Use signatures for secure key distribution!

- Alice asks the CA to sign the *binding* (Alice, pk)
  - $cert_{CA \to Alice} = Sign_{sk_{CA}}(Alice, pk)$
- (CA must verify Alice's identity out of band)

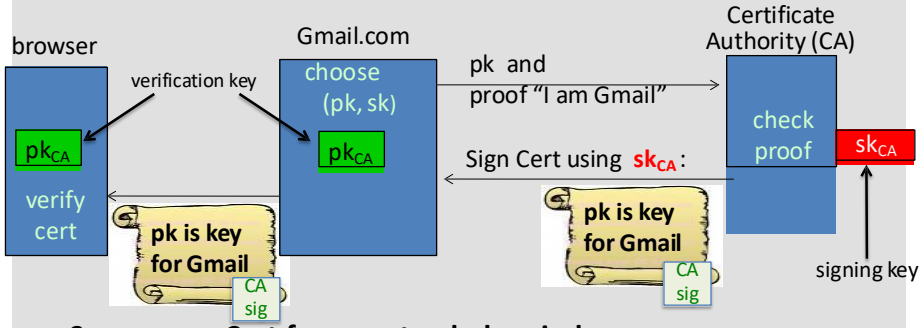
Use signatures for secure key distribution!

- Bob obtains Alice, pk, and the certificate $cert_{CA \to Alice}$ …
  - … verifies that $Vrfy_{pK_{CA}}((Alice, pk), cert_{CA \to Alice}) = 1$
- Bob is then assured that pk is Alice's public key
  - As long as the CA is trustworthy…
    - Honest, and properly verifies Alice's identity
  - …*and the CA's private key has not been compromised*

Important application: Certificates

Problem: browser needs server's public-key to setup a session key

Solution: server asks trusted 3rd party (CA) to sign its public-key pk



Server uses Cert for an extended period (e.g. one year)

Dan Boneh

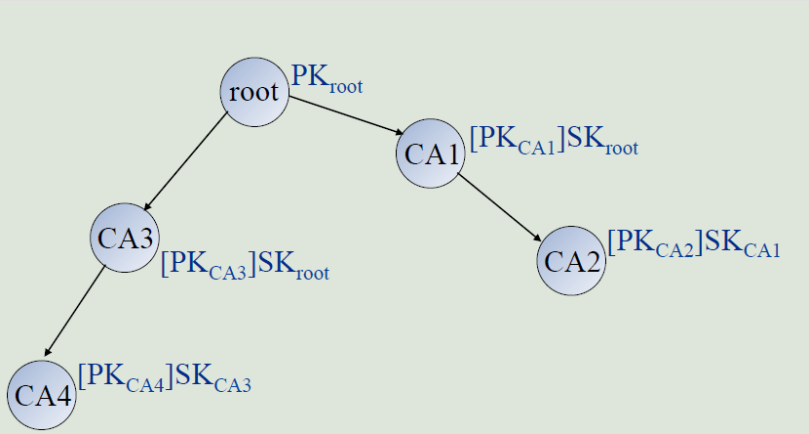
Chicken-and-egg problem?

- How does Bob get pk_{CA} in the first place?
- Several possibilities…

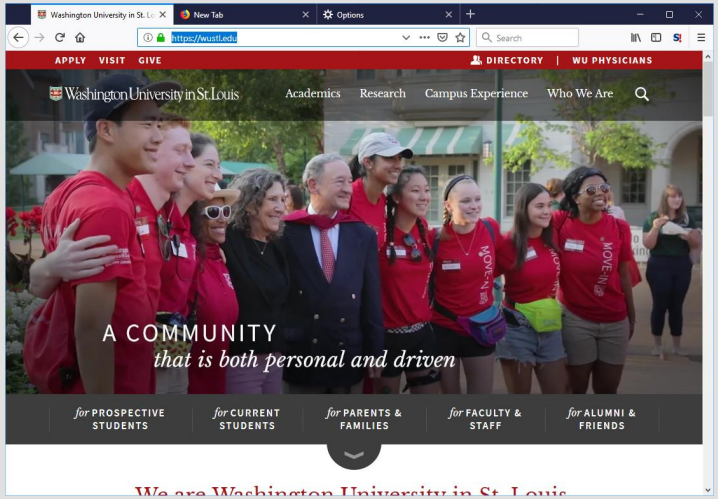
"Roots of trust"

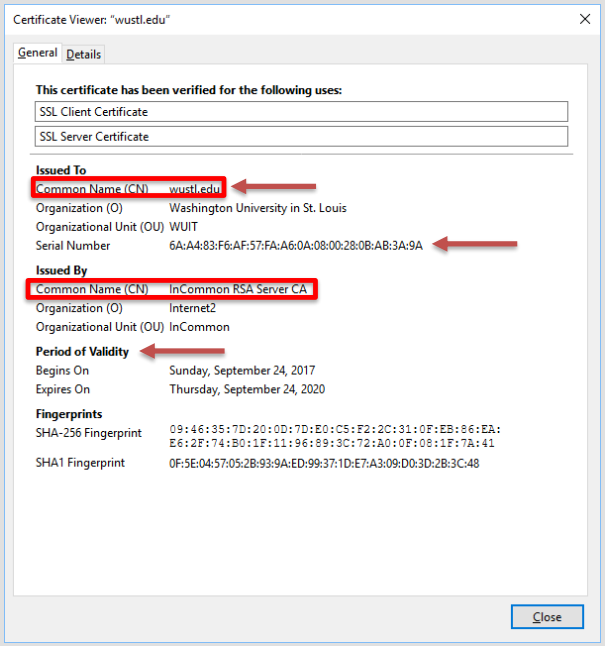
- Bob only needs to securely obtain a small number of CA's public keys
  - Need to ensure secure distribution only for these few, initial public keys
- E.g., distribute as part of an operating system, or web browser
  - Firefox:
    Tools->Options->Privacy & Security->View certificates->Authorities

Certificate Hierarchy

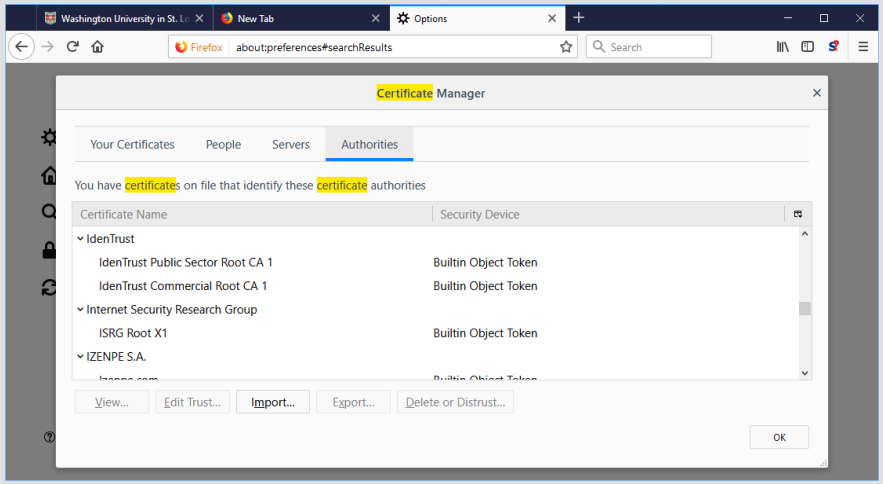


Is WUSTL secure?

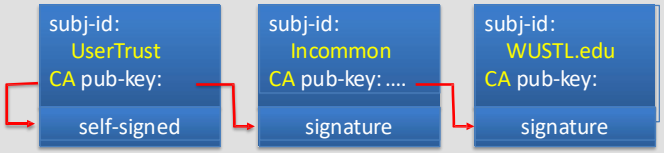


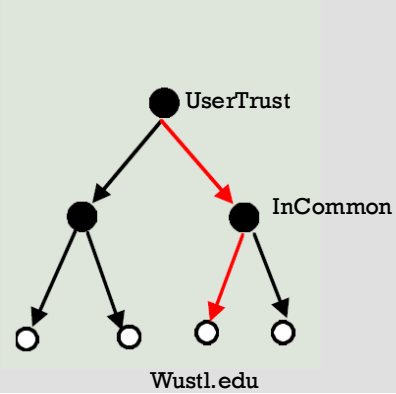


What else are we trusting?



Certificate Path

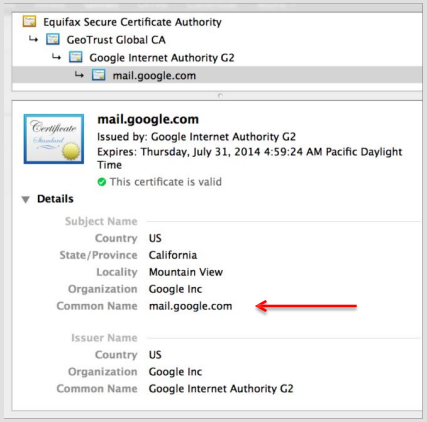




- Verifier has to know the public key of root CA
- Other public keys can be discovered
- **All** CA on the path has be trusted by the verifier

Certificates: example

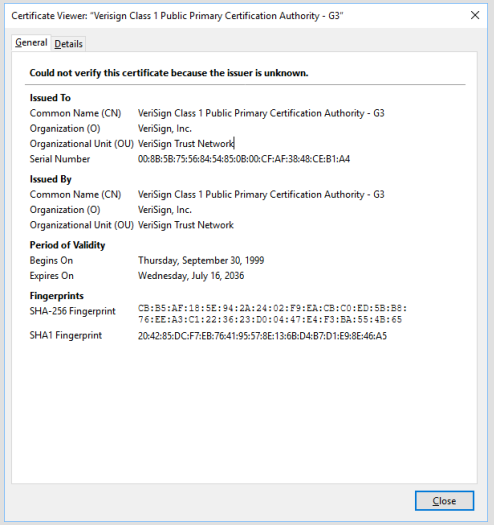
Important fields:



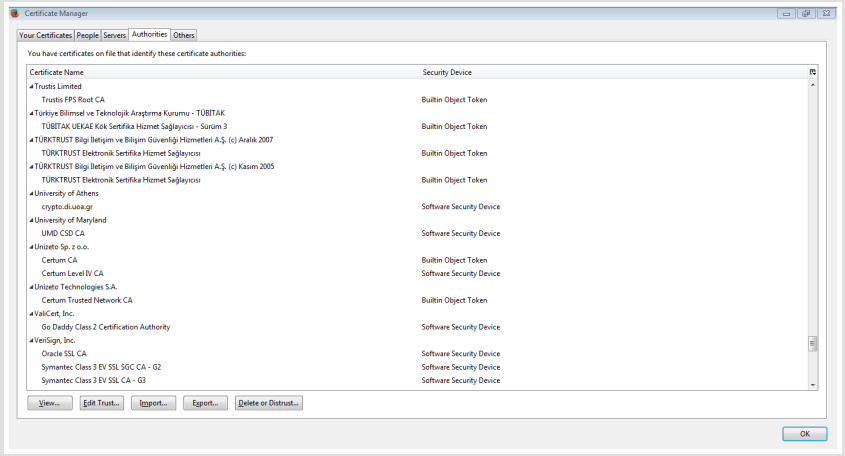
PKI in practice… Many problems

- Does not work quite as well as in theory…
  - Proliferation of root CAs
  - Revocation
  - Common Name Issue

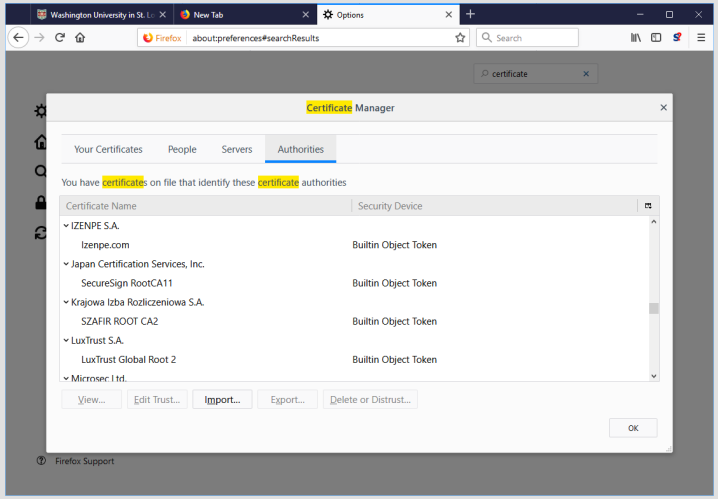
Something you trust



Do you trust them all ?



A lot of people can sign, and this is just top level



The numbers

Browsers accept certificates from a large number of CAs:

- Top level CAs ≈ 60
- Intermediate CAs ≈ 1200

Revocation

Why do we need revocation?

- It could be
  - CA is compromised
  - Bob forget his private key
  - Bob's private key is stolen
  - Bob disclose his private key
- Certificate revocation needs to occur when:
  - certificate holder key compromise/loss
  - CA key compromise
  - end of contract (e.g. certificates for employees)

General Requirement of Revocation

- Timeliness
  - Before using a certificate, must check most recent revocation status
- Efficiency
  - Computation
  - Bandwidth and storage
  - Availability
- Security

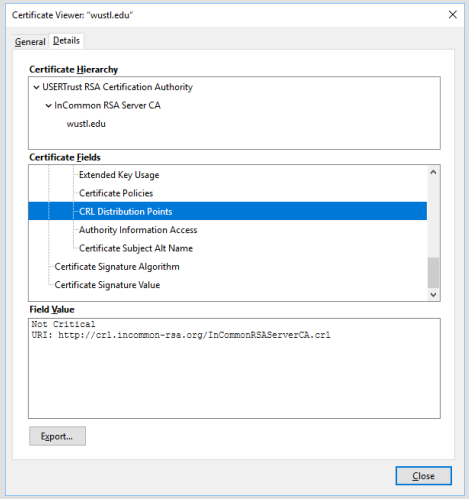
Revocation Type

- Implicit
  - Each certificate is periodically issued
  - Alice has a fresh certificate -> Alice not revoked
  - No need to distribute/publish revocation info
- Explicit
  - Only revoked certificates are periodically announced
  - Alice's certificate not listed among the revoked -> Alice not revoked
  - Need to distribute/publish revocation info

Revocation Method

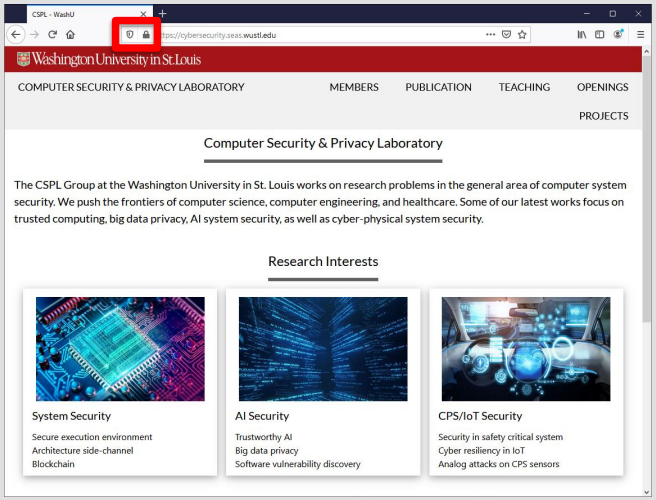
- CRL - Certificate Revocation List
  - CRL-DP, indirect CRL, dynamic CRL-DP,
  - delta-CRL, windowed CRL, etc.
  - CRT and other Authenticated Data Structures
- OCSP – On-line Certificate Status Protocol
- CRS - Certificate Revocation System

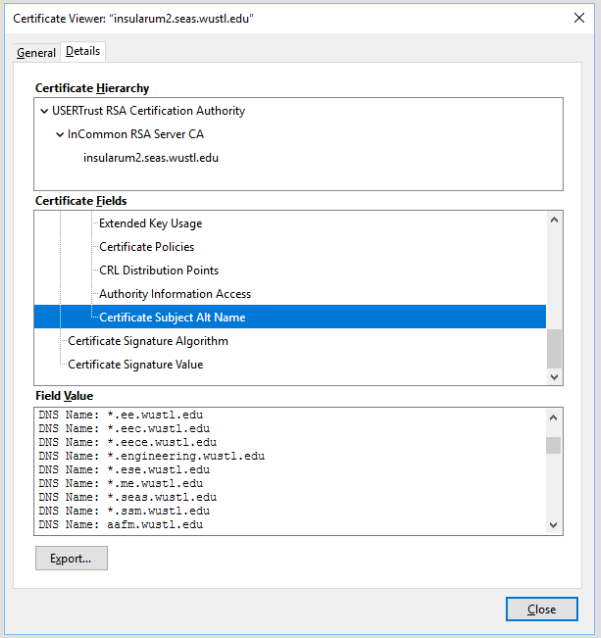
Revocation for WUSTL



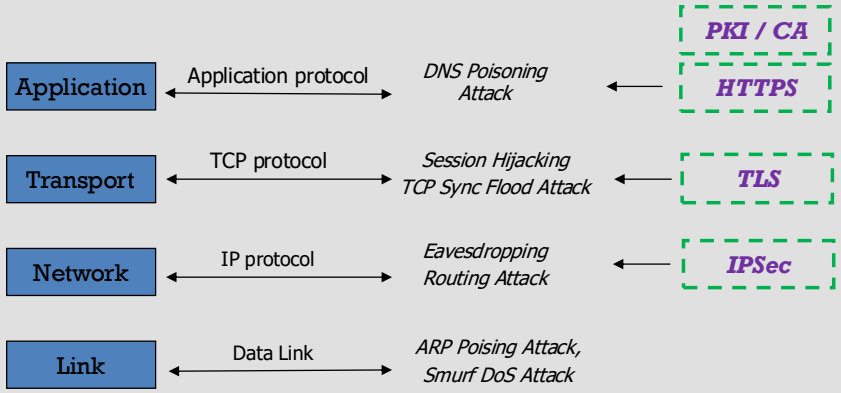
Alternative Names

My website is secure? How?



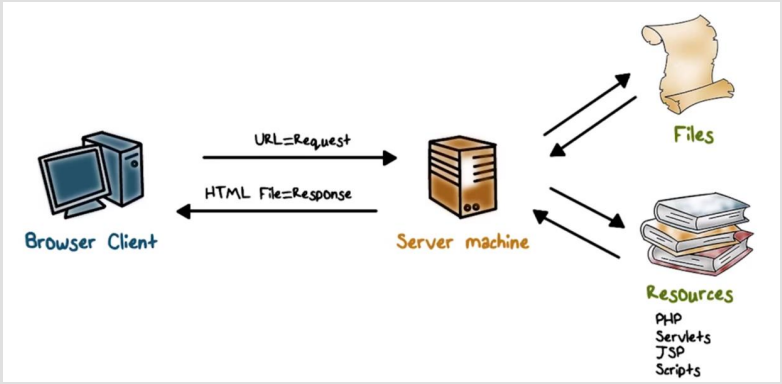


This module - Network defense



HTTPS/TLS

HTTP



Threat Model

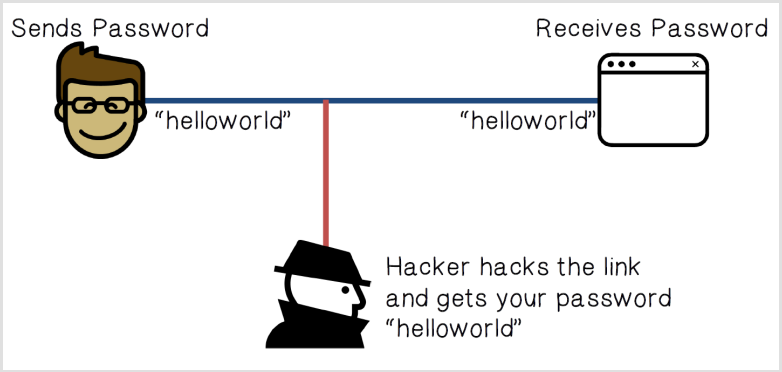
Network Attacker:

- Controls network infrastructure: Routers, DNS
- Eavesdrops, injects, blocks, and modifies packets

Examples:

- Wireless network at Internet Café
- Internet access at hotels (untrusted ISP)

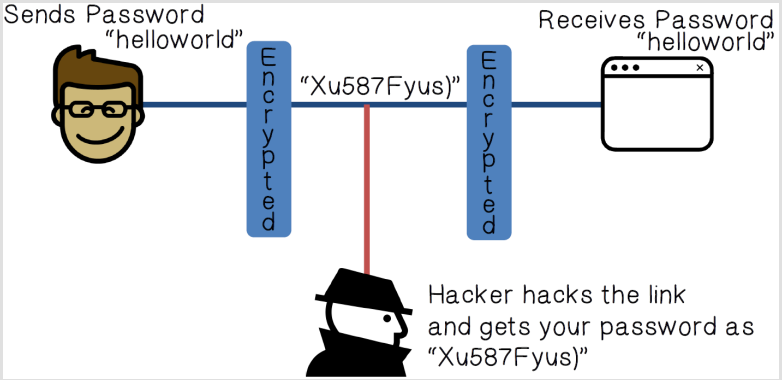
HTTP



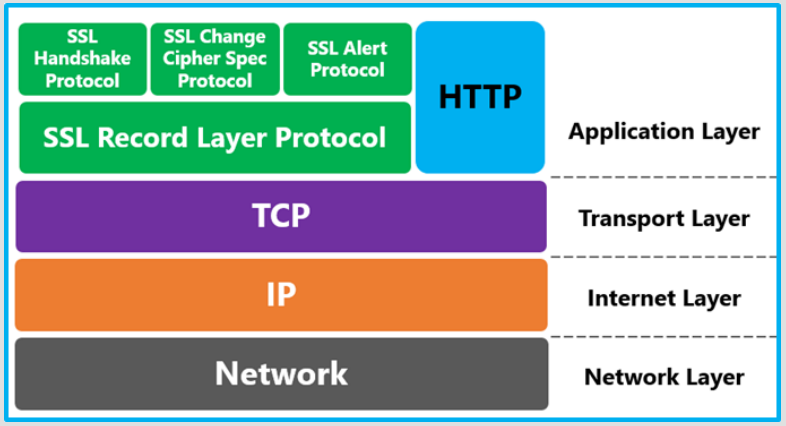
HTTPS

HTTPS: Hyper Text Protocol Secure= all communication between your browser and a website is encrypted.

HTTPS

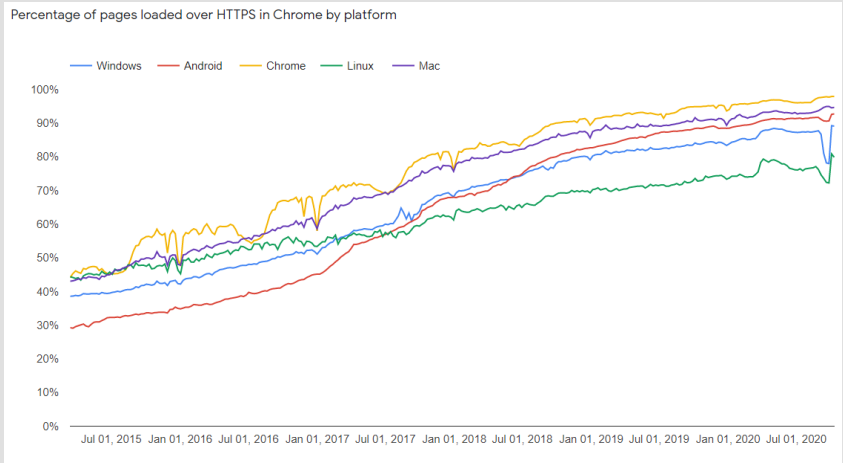


Where is HTTPS

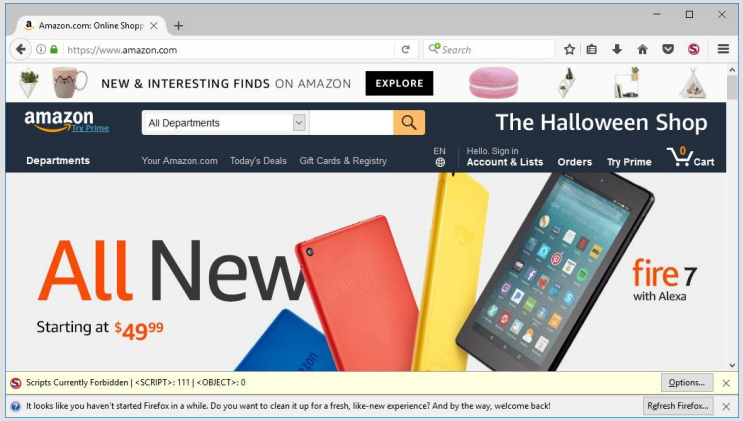


HTTPS

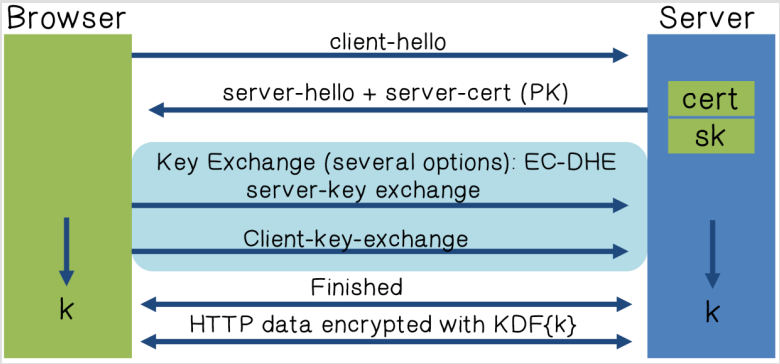
- Creates a secure channel over an insecure network
- Is an effective protection against man-in-the-middle attacks
- Can still provide security even when only one side of the communication is secure



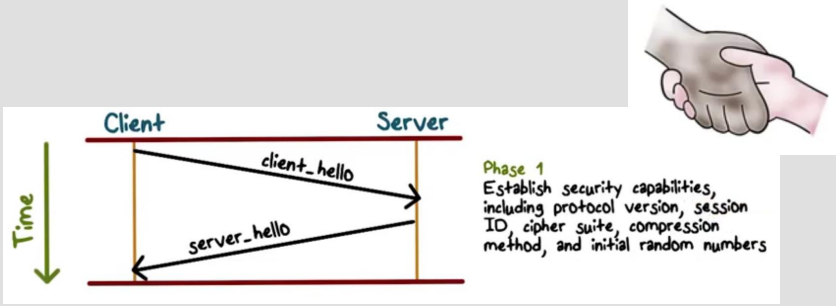
First few hundred millisecs when you go to amazon.com



TLS Summary



The Handshake Protocol

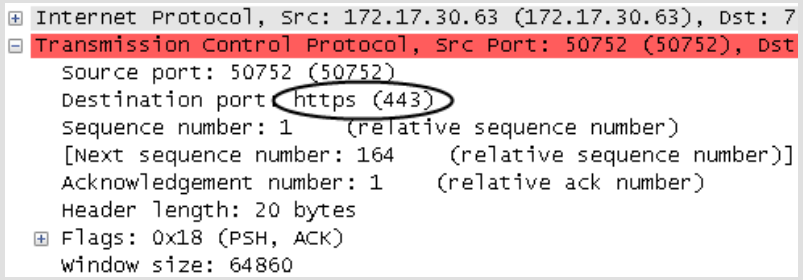


The Handshake Protocol

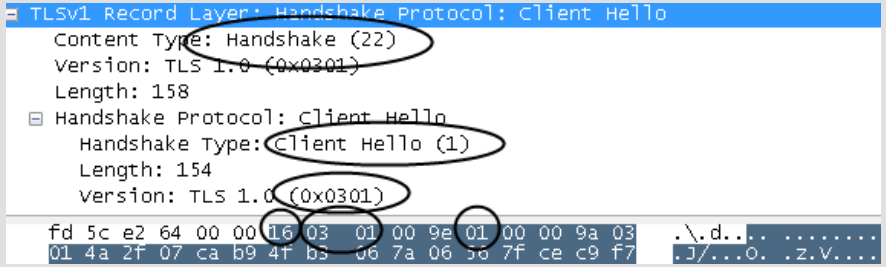
The Parameters:

- Version: the highest TLS version understood by the client
- Random: a 32-bit timestamp and 28 bytes generated by a secure random number generator
- Session ID: a variable-length session identifier
- CipherSuite: a list containing the combinations of cryptographic algorithms supported by the client
- Compression Method: a list of compression methods supported by the client

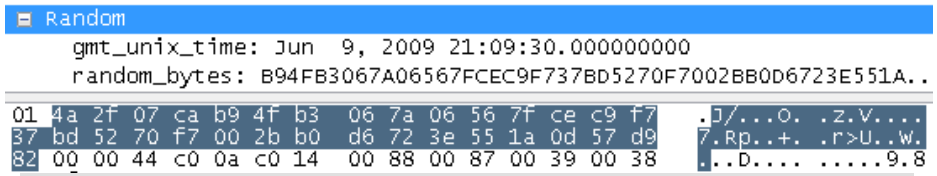
HTTPS -> Port 443



Client Hello – Version

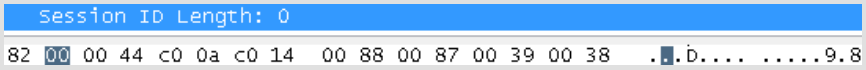


Client Hello - Random

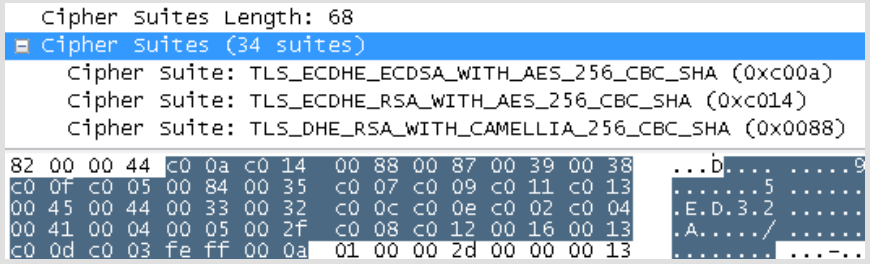


Why do we need this?

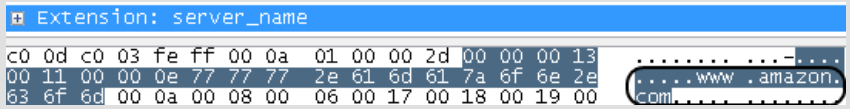
Client Hello – Session ID



Client Hello – Cipher Suite

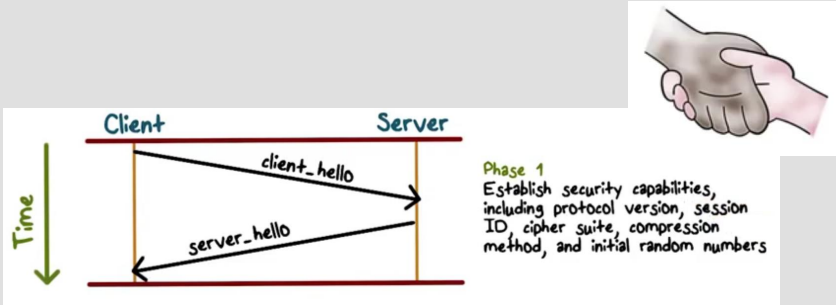


Client Hello – Server Name Extension

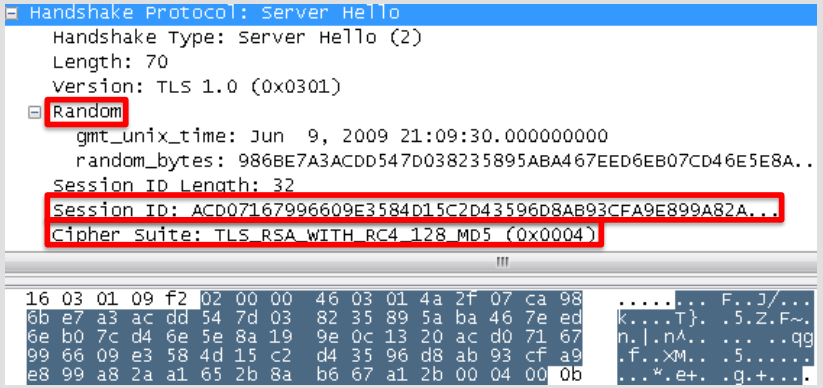


Why do we need this?

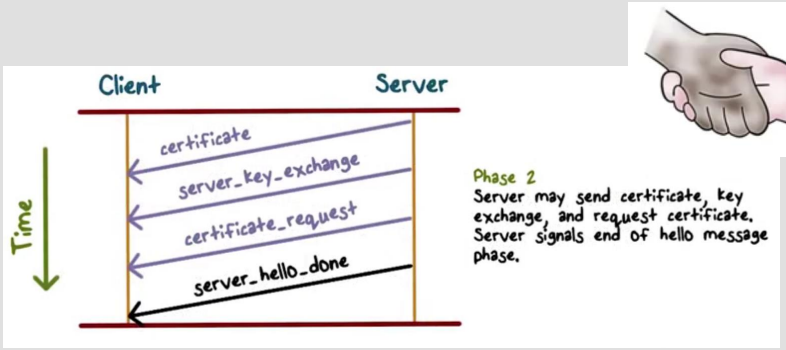
The Handshake Protocol



Server Hello



The Handshake Protocol

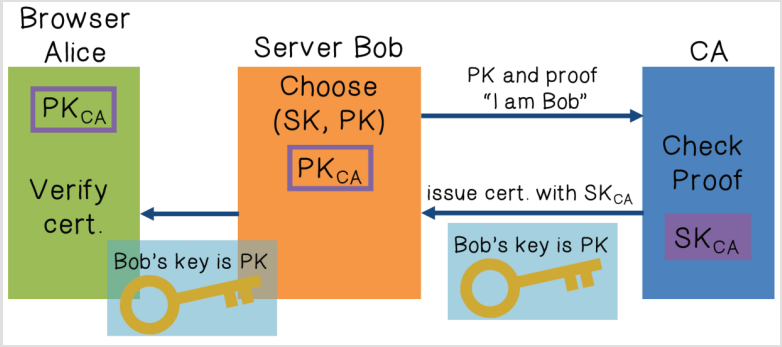


Certificate Message

```
Handshake Protocol: Certificate
    Handshake Type: Certificate (11)
    Length: 2464
    Certificates Length: 2461
  - Certificates (2461 bytes)
      Certificate Length: 1271
    - Certificate (id-at-commonName=www.amazon.com,id-at-organizationN
      - signedCertificate
            version: v3 (2)
```

```
e8 99 a8 2a a1 65 2b 8a  b6 67 a1 2b 00 04 00 0b   ...*.e+. .g.+....
00 09 a0 00 09 9d 00 04  f7 30 82 04 f3 30 82 03   ........ .0...0..
db a0 03 02 01 02 02 10  16 9d 04 1c 31 30 be 3d   ........ ....10.=
56 66 06 f2 67 9b a1 72  30 0d 06 09 2a 86 48 86   Vf..g..r 0...*.H.
f7 0d 01 01 05 05 00 30  81 b0 31 0b 30 09 06 03   .......0 ..1.0...
55 04 06 13 02 55 53 31  17 30 15 06 03 55 04 0a   U....US1 .0...U..
13 0e 56 65 72 69 53 69  67 6e 2c 20 49 6e 63 2e   ..VeriSi gn, Inc.
31 1f 30 1d 06 03 55 04  0b 13 16 56 65 72 69 53   1.0...U. ...VeriS
69 67 6e 20 54 72 75 73  74 20 4e 65 74 77 6f 72   ign Trus t Networ
```

What is a certificate ?

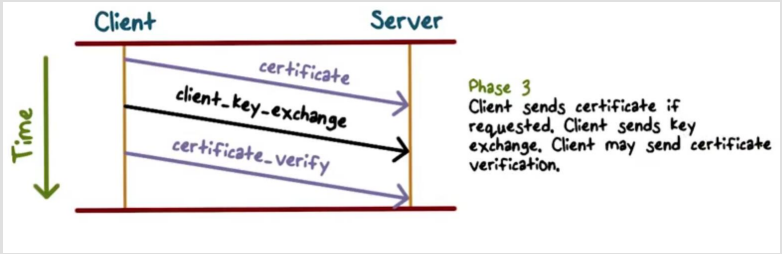


Server Hello Done

```
Handshake Protocol: Server Hello Done
    Handshake Type: Server Hello Done (14)
    Length: 0
```

```
03 90 0c 0e 00 00 00                               .......
```

The Handshake Protocol

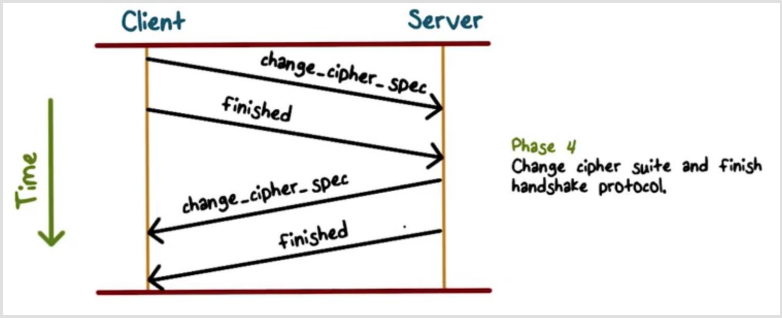


Key Exchange

```
Handshake Protocol: Client Key Exchange
    Handshake Type: Client Key Exchange (16)
    Length: 130

fd 5c e2 77 00 00 16 03  01 00 86 10 00 00 82 00   .\.w.... ........
80 8c 08 63 cc 01 6c d0  36 ef a2 28 89 e1 61 1a   ...c..l. 6..(..a.
8a 03 30 43 66 eb b0 4d  a1 cb 91 85 48 e0 24 f4   ..0Cf..M ....H.$.
88 a1 55 cd 85 03 0b 5b  d7 e6 7a 9d a2 01 90 89   ..U....[ ..z.....
ce ba 57 11 00 05 1c af  ce ef f4 2b 3f e6 30 f2   ..W..... ...+?.0.
6e 7d 55 e6 d6 c8 09 76  85 1d a2 51 7e 28 bf 83   n}U....v ...Q~(..
1e be 0b ed 60 1c 9b 00  c5 2e a6 22 38 ed a3 fb   ....`... ..."8...
7e dd 68 9a c2 ad dc 23  41 ba 30 f0 cd 95 60 48   ~.h....# A.0...`H
85 ed 44 8e 82 83 46 c3  b7 12 06 ae 48 11 4e b2   ..D...F. ....H.N.
da 14 03 01 00 01 01 16  03 01 00 20 6a 20 7a 41   ........ ... j zA
```

The Handshake Protocol

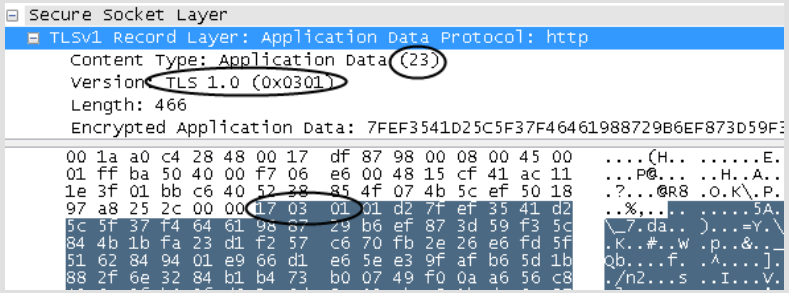


Client Change Cipher Spec

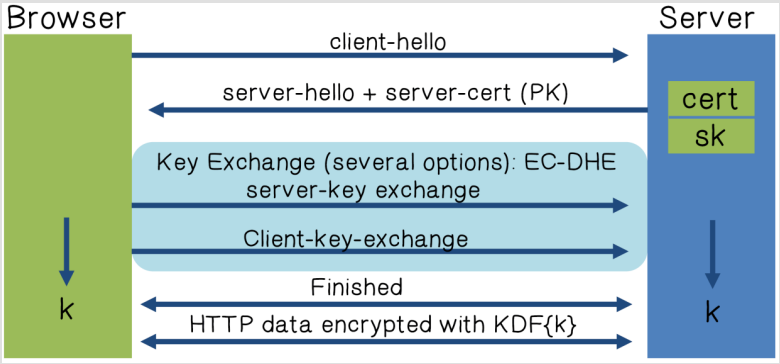
```
TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.0 (0x0301)
    Length: 1
    Change Cipher Spec Message

c0  da 14 03 01 00 01 01 16  03 01 00 20 6a 20 7a 41   ........ ... j zA
```

Finally, Encrypted Traffic



TLS Summary



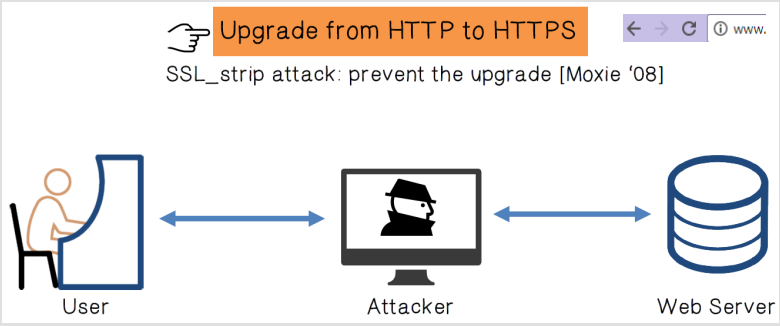
Problems with HTTPS

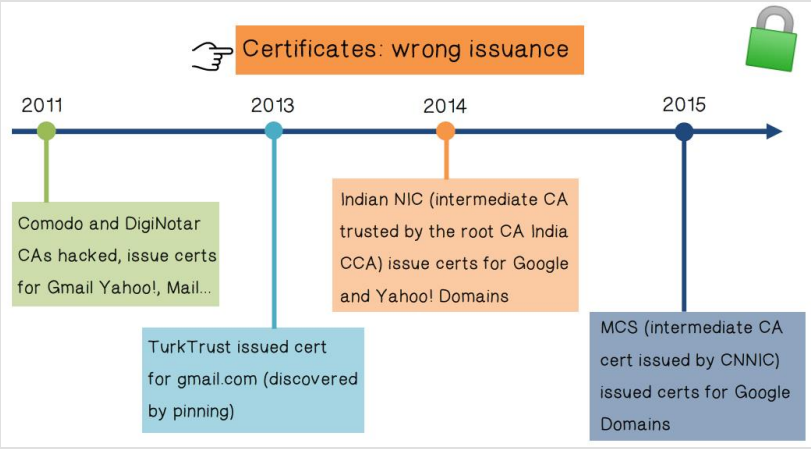
Upgrade from HTTP to HTTPS

Forged certs

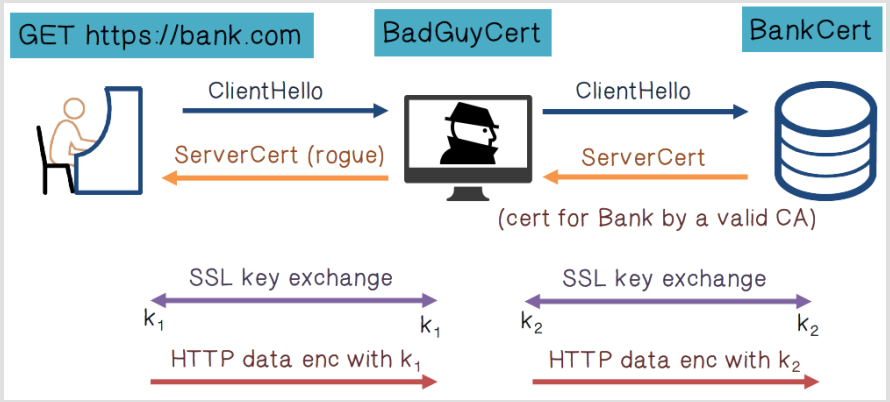
Mixed content: HTTP and HTTPS on the same page

HTTPS downgrading attack





MIMT using rouge cert



Problems with HTTPS

Upgrade from HTTP to HTTPS

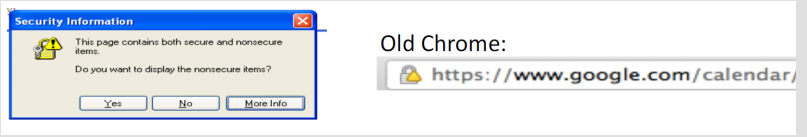
Forged certs

Mixed content: HTTP and HTTPS on the same page

Mixed Contents: HTTP and HTTPS

- Page loads over HTTPS, but contains content over HTTP (e.g. <script src="http://.../script.js>)

⇒ Active network attacker can hijack session by modifying script en-route to browser



Mostly ignored by users …

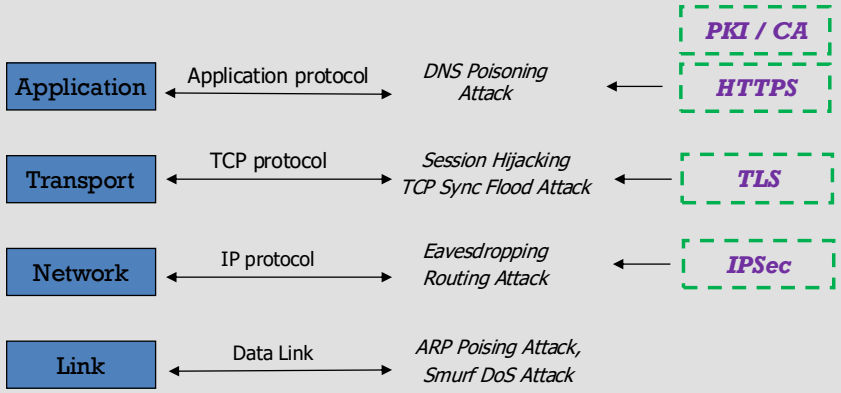
How do you attack TLS?

Padding Oracle Attack (BEAST)
Export Downgrade (FREAK)
Common Exponent (LogJam, CCS 15 Best Paper)
Side Channel Attack (Chen et al. SP 10)

Summary

- TLS is one of the most fundamental methods to secure the internet
  - TLS client hello / server hello
  - Key exchange
- HTTPS attacks
  - Downgrade attack
  - Rouge certificates
- TLS attack
  - Padding oracle (BEAST, Lucky 13)
  - Export downgrade (FREAK)
  - DH component reuse (Logjam)
  - Side channel information leakage

This module - Network defense



IPSec

IP is not Secure!

- IP protocol was designed in the late 70s to early 80s
- Part of DARPA Internet Project
- Very small network
  - All hosts are known!
  - So are the users!
  - Therefore, security was not an issue

Security Issues in IP

- Source spoofing
- Replay packets
- No data integrity or confidentiality

→
- DOS attacks
- Replay attacks
- MiTM attack
- Interleaving attacks
- Eavesdropping
- and more…

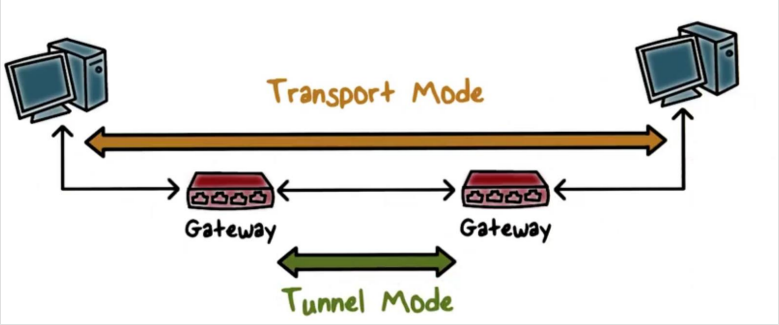
Fundamental Issue:

Networks are not fully secure

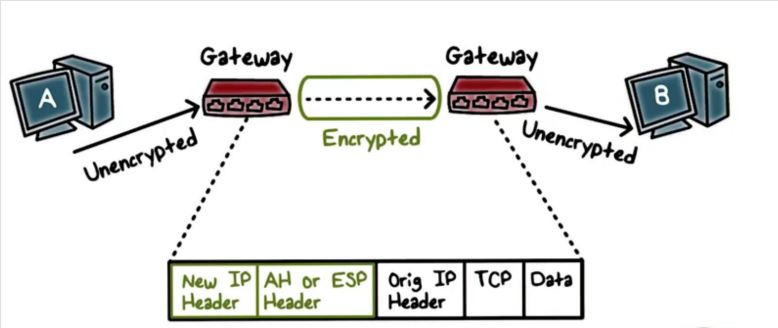
IPSec Architecture

- IPSec provides security in three situations:
  - Host-to-host
  - Host-to-gateway
  - Gateway-to-gateway
- IPSec operates in two modes:
  - *Transport mode* (for end-to-end, host-to-host)
  - *Tunnel mode* (for VPN)

IPSec Modes - Tansport



IPSec Modes - Tunnel Mode



Blockchain