Digital certificates

TLS, HTTPS, Revocation

https://s-media-cache-ak0.pinimg.com/originals/ed/af/55/edaf5554d92824ef3555d8b9f60c5.jpg
• For convenience, we will use $PK_A$ and $SK_A$ to denote public and secret keys for Alice.
Trusted third party, revisited (1)

- TTP is a bottleneck for every conversation
- TTP must be online to start a new conversation
- TTP can read every message
- TTP must be trusted to tell the truth!
- Does not solve bootstrapping problem
Trusted 3rd party, revisited (2)

Alice owns PK_A. Signed, PK_T

S(SK_A, E(PK_B, m)) + cert

PK_T

PK_A plus verification

Bob: Verify cert with PK_T, verify message with PK_A
With certificates

- TTP is a bottleneck for every conversation
- TTP must be online to start a new conversation
- TTP can read every message
- TTP must be trusted to tell the truth!
- Does not solve bootstrapping problem
Certificates in practice

- TTP = Certificate Authority
  - Verisign, Comodo, Thawt, etc.
- Alice = web server
- Bob = user who visits alice.com
  - Validate talking to the real alice.com
  - Set up encrypted session for HTTPS
- This is a hierarchical public key infrastructure (PKI)
This certificate has been verified for the following uses:

SSL Client Certificate

SSL Server Certificate

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issued To</td>
<td></td>
</tr>
<tr>
<td>Common Name (CN)</td>
<td>.cs.umd.edu</td>
</tr>
<tr>
<td>Organization (O)</td>
<td>University of Maryland, College Park</td>
</tr>
<tr>
<td>Organizational Unit (OU)</td>
<td>&lt;Not Part Of Certificate&gt;</td>
</tr>
<tr>
<td>Serial Number</td>
<td>0F:F6:E0:5D:8C:8F:F3:65:79:B0:7D:45:73:04</td>
</tr>
<tr>
<td>Issued By</td>
<td></td>
</tr>
<tr>
<td>Common Name (CN)</td>
<td>DigiCert SHA2 High Assurance Server CA</td>
</tr>
<tr>
<td>Organization (O)</td>
<td>DigiCert Inc</td>
</tr>
<tr>
<td>Organizational Unit (OU)</td>
<td><a href="http://www.digicert.com">www.digicert.com</a></td>
</tr>
<tr>
<td>Period of Validity</td>
<td></td>
</tr>
<tr>
<td>Begins On</td>
<td>8/11/14</td>
</tr>
<tr>
<td>Expires On</td>
<td>8/16/17</td>
</tr>
<tr>
<td>Fingerprints</td>
<td></td>
</tr>
</tbody>
</table>
Certificate types

Why are these different?

This is an EV (extended validation) certificate; browsers show the full name for these kinds of certs.
Transport layer security (TLS)

- Runs on top of TCP/IP
- Protocols for secure comms
  - Confidentiality with block and stream ciphers
  - Integrity with MACs
  - Authenticity with certificates
- Replacement for SSL (secure sockets layer)
  - Several problems including padding attacks
TLS protocol overview

**Browser** (initiates connection)
- **Client hello**
  - Version, crypto options, nonce
- **Server hello + server cert (PKs)**
  - Version, crypto options, nonce, signed PK certificate
- **Server key exchange (if using DH)**
- **Client key exchange**
  - PreMaster secret encrypted with server’s PK
- \[\sim\sim\sim\sim\sim\text{Switch to negotiated cipher}\sim\sim\sim\sim\sim\]

**Server** (authenticates itself)
- **Compute K based on nonces & PreMaster**
- **Data transmission**
- **Compute K based on nonces & PreMaster**
HTTPS

• HTTP “on top of” TLS

• Pros: **Avoid MITM**
  • Includes e.g. reducing video quality, inserting ads

• Cons
  • Takes more time
  • Network service/ISP can’t compress or cache it
  • Network service/ISP wants to insert ads

[https://www.eff.org/https-everywhere](https://www.eff.org/https-everywhere)
Revoking certificates

- When you detect compromise or change keys, you have to notify the CA

- CA then *revokes* the certificate
  - Revocation list
  - Online cert status protocol
  - Short expiry times
Revocation list

- CA publishes list of revoked certs
- User (in practice, browser) must periodically download the newest list
  - Check when validating a certificate
- Vulnerability window since last list update
  - Or until certificate expires
- Can be beaten via DOS (why?)
Online certificate status

- During validation, ask CA whether cert is revoked
- Gets rid of vulnerability window
  - But can’t accept any cert if CA is not online!
- And, the CA gets to know where you browse
Short expiration

- Make all certificates have very short expirations (e.g. 10 min or less)
  - For the most part, renew automatically
- Revocation == decline to renew
- Expensive, not implemented that I’m aware of
  - Also some browsers accept expired certs
Heartbleed + Revocation

• Certificate revocation is a manual process

• Easy to measure revocation; hard to measure when should have revoked

• Heartbleed = natural experiment (April 7, 2014)
  • OpenSSL bug allows reading memory
  • Potential compromise of keys at 100ks of hosts

• Correct procedure: patch, revoke, reissue

xkcd
Measurement study

- Certs from weekly scans of IP4 space
  - 19.5m total certs
  - 1.5m certs for Alexa top 1m domains
  - 600k+ leaf certs
  - Revocation lists from 99%+ of leaf certs
- Heuristic for determining sites that had been vulnerable to Heartbleed
- Heuristics for reissue
After heartbleed
Heartbleed reactions

- 100k *vulnerable* certs
  - 27% reissued by April 30
- 4% of heartbleed-related reissues use same key
- 40% of heartbleed reissues also revoked
  - Revocation usually after reissue
- Revocation is less common on the weekends
Trusting the Trusted Third Party

http://randomrock.com.br/randomrock/rock-n-movies-20-watchmen/
Where do CAs come from?

- CA public keys shipped with browsers, OS
  - iOS9 ships with >50 that start with A-C
    - see here for full list
CA compromise

- 2001: Verisign issued two code-signing certificates for Microsoft Corporation
  - To someone who *didn’t actually* work at MS
  - No functional revocation paradigm

- 2011: Signing keys compromised at Comodo and DigiNotar
  - Bad certs for Google, Yahoo!, Tor, others
  - Seem to have been used mostly in Iran

- Some CAs are less picky than others
Case study: Superfish (Feb 2015)

- Lenovo laptops shipped with “Superfish” adware
- Installs self-signed root cert into browsers
  - MITM on every HTTPS site to inject ads
- Worse: Same private key for every laptop
  - Password = “komodia” (company
- Lenovo “did not find any evidence to substantiate security concerns”

Fixing rogue CA problems

- Limit which CAs can issue for which domains
- Certificate pinning
  - Browser, apps fix certain CA or cert for a server
  - Shipped with product, or on first use
  - Not always appropriate, hard to maintain
Fixing rogue CA problems (2)

- Broad surveillance
  - People on many networks report certs to Notaries
  - Check that others saw the same cert you did
  - Privacy implications

- Public unforgeable audit log
  - Uses crypto, Merkle hash trees
    - Only accept certs published in log
  - Same idea: Non-equivocation

- Being implemented now

https://www.eff.org/observatory  https://www.eff.org/sovereign-keys
Web of trust
Web of trust

- Alternative PKI — not hierarchical
  - Pioneered by PGP
- Don’t rely on centralized authorities
- Everyone issues certificates for people they know
Trust chains in web of trust

Alice trusts Bob

Bob sends message to Donald

Donald vouches for Cookie

Cookie vouches for Bob
A matter of trust

• Context:
  • Alice trusts Bob to diligently check identity
  • But Bob is only signing identity, not necessarily belief that Cookie is equally vigilant

• Transitivity: Alice trusts Bob, and Bob trusts Cookie.
  • But does that mean Alice should trust Cookie?
  • Trust for honesty == trust for good judgment?
Web-of-trust in practice

- Automatically find many such paths
  - More, shorter paths = higher confidence?

- Difficult to use
  - Still have bootstrapping problems
  - When should I agree to sign what?
  - Historically, serious UX problems as well