15. Patch Deployment and Certificate Revocation
ENEE 757 | CMSC 818V

Prof. Tudor Dumitraș
Assistant Professor, ECE
University of Maryland, College Park

http://ter.ps/757
https://www.facebook.com/SDSAatUMD

Today’s Lecture

• Where we’ve been
  – Authentication and access control
  – Internet security
  – Malware distribution networks
  – SSL/TLS
  – PKI

• Where we’re going today
  – Attacks against PKI certificates and certificate revocation
  – Patch deployment

• Where we’re going next
  – Automatic exploit generation and obfuscation
Recall – CA Hierarchy

• Certification Authorities (CAs) are used for **bootstrapping trust**

• Browsers, operating systems, etc. have trusted root certificate authorities
  – Verisign, Thawte, Comodo, organizations from over 30 countries, ...
  – ~200 trusted root CAs overall

• A Root CA signs certificates for intermediate CAs, they sign certificates for lower-level CAs, etc.
  – Certificate “chain of trust”

• CA is responsible for verifying the identities of certificate requestors, domain ownership

• **Who is in your TCB?**

Reducing the TCB: Key Pinning

• Bind domain to list of keys
  – At least one whitelisted key must appear on the chain of trust
  – **Leaf pinning**: accept only certain leaf certificate(s) or public keys for a site (strictest security)
  – **CA pinning**: allow only a few CAs to sign certificates for a site (most common in practice)
    • Pinning to multiple CAs helps in case one CA is compromised

• How to distribute pins to the clients?
  – Preloaded **pinning policies**: distribute pins with software updates
    • Does not work well with non-automated software updates: when pins change but the user does not update, the application stops working
    • Particularly tricky with non-browser SW (e.g. mobile apps)
  – **HPKP** (RFC 7469): HTTP header declaring pins
    • Trust-on-first-use (TOFU) model
Reducing the TCB: Certificate Transparency

- Log servers maintain certificate transparency log
  - Anyone can submit certificates
  - CAs are supposed to submit first

- Log is tamper-proof, append-only
  - Can produce audit proofs verifying that:
    - A certificate was appended to the log
    - The new log version includes everything in the earlier version, in the same order,
    - All new entries come after the entries in the older version
  - Monitors can inspect logs for weird certificates
    - Example: Google can check who is issuing certificates for google.com

- Certificate transparency doesn’t strictly reduce the TCB
  - But it helps expose rogue CAs

Reducing the TCB: DANE

- DNS-based Authentication of Named Entities (DANE)
  - RFC 6698
  - Bind certificates to DNS names, using DNSSEC
    - Adds the DNSSEC Root CA to the TCB

- Not widely deployed
Alternative: Web of Trust

- Used in PGP (Pretty Good Privacy)
- There are no root certificate authorities
  - Instead, each person must select a set of keys they trust
  - If public-key certificate is signed by one of the trusted keys, the public key contained in it will be deemed valid
- Trust can be transitive
  - Can use certified keys for further certification

X.509 PKI Standard

- Internet standard (1988-2000)
- Specifies certificate format
  - X.509 certificates are used in IPsec and SSL/TLS
- Specifies certificate directory service
  - For retrieving other users’ CA-certified public keys
- Specifies a set of authentication protocols
  - For proving identity using public-key signatures
- Can use with any digital signature scheme and hash function
  - Must hash the certificate content before signing
  - Only the hash is signed, for efficiency
X.509 Certificate

Security Threat: Rogue Certificates

- Recall: What can you do if you could issue a certificate for a third-party domain?

- Rogue certificates
  - Issued by a Rogue CA
  - Issued by a CA that has been compromised
  - Compromised certificates (adversary knows the private key)
Back in 2008

[Sotirov et al. “Rogue Certificates”]

- Many CAs still used MD5
  - RapidSSL, FreeSSL, TrustCenter, RSA Data Security, Thawte, verisign.co.jp

- Sotirov et al. collected 30,000 website certificates

- 9,000 of them were signed using MD5 hash

- 97% of those were issued by RapidSSL

Colliding Certificates

[Sotirov et al. “Rogue Certificates”]

<table>
<thead>
<tr>
<th>set by the CA</th>
<th>serial number</th>
<th>chosen prefix (difference)</th>
<th>serial number</th>
<th>validity period</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>validity period</td>
<td></td>
<td>validity period</td>
<td></td>
</tr>
<tr>
<td></td>
<td>real cert domain name</td>
<td>Hash to the same MD5 value using birthday attack</td>
<td>rogue cert domain name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>real cert RSA key</td>
<td>collision bits (computed)</td>
<td>???</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X.509 extensions</td>
<td>identical bytes (copied from real cert)</td>
<td>X.509 extensions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>signature</td>
<td></td>
<td>signature</td>
<td></td>
</tr>
</tbody>
</table>

Valid for both certificates!
Generating Collisions
[Sotirov et al. “Rogue Certificates”]

1-2 days on a cluster of 200 PlayStation 3

Equivalent to 8000 desktop CPU cores or $20,000 on Amazon EC2

Generating Colliding Certificates
[Sotirov et al. “Rogue Certificates”]

• RapidSSL uses a fully automated system
  – $69 for a certificate, issued in 6 seconds
  – Sequential serial numbers

• Technique for generating colliding certificates
  – Get a certificate with serial number S
  – Predict time T when RapidSSL’s counter goes to S+1000
  – Generate the collision part of the certificate
  – Shortly before time T buy enough (non-colliding) certificates to increment the counter to S+999
  – Send colliding request at time T and get serial number S+1000 (why?)
Creating a Fake Intermediate CA

[Sotiropov et al. “Rogue Certificates”]

<table>
<thead>
<tr>
<th>serial number</th>
<th>chosen prefix (difference)</th>
</tr>
</thead>
<tbody>
<tr>
<td>validity period</td>
<td>rogue CA cert</td>
</tr>
<tr>
<td>real cert domain name</td>
<td>rogue CA RSA key</td>
</tr>
<tr>
<td>real cert RSA key</td>
<td>rogue CA X.509 extensions</td>
</tr>
<tr>
<td>X.509 extensions</td>
<td>Netscape Comment Extension (contents ignored by browsers)</td>
</tr>
<tr>
<td>signature</td>
<td>identical bytes (copied from real cert)</td>
</tr>
<tr>
<td></td>
<td>signature</td>
</tr>
</tbody>
</table>

We are now an intermediate CA. W00T!

Result: Ability to Conduct Man-in-the-Middle Attacks

[Sotiropov et al. “Rogue Certificates”]

• This is a “skeleton key” certificate: it can issue fully trusted certificates for any site

• To take advantage, need a network attack to redirect traffic
  – Insecure wireless, DNS poisoning, proxy auto-discovery, hacked routers, etc.
• Today MD5 collisions can be computed efficiently
  – https://marc-stevens.nl/p/hashclash/
**Stronger Attacks**

- Certificates are also involved in code signing
  - OS will execute certain programs only if they are signed by a trusted party
    - Examples: software updates, device drivers
  - Trust is derived from certificates shipped with the OS

- The Stuxnet worm (2010)
  - Used drivers signed by two Taiwanese semiconductor companies
  - Certificates had been compromised
    - The two companies were in close physical proximity
  - Result: Windows did not issue warnings about the driver and anti-virus products ignored it

**The Flame Virus**

- Cyber-espionage virus (2010-2012)

- Signed with a fake intermediate CA certificate that appears to be issued by Microsoft and thus accepted by any Windows Update service
  - Fake intermediate CA certificate was created using an MD5 chosen-prefix collision against an obscure Microsoft Terminal Server Licensing Service certificate that was enabled for code signing and still used MD5

- Different MD5 collision technique than Sotirov et al.‘s work
  - Attacker able to do sophisticated cryptanalysis
**Comodo and DigiNotar Hacks (2011)**

- Comodo and DigiNotar were Root CAs

- Iranian hacker compromised Comodo resellers and used their credentials to request certificates using Comodo’s APIs
  - Got Comodo to issue 9 rogue certificates for popular sites: mail.google.com, login.live.com, login.yahoo.com, login.skype.com, addons.mozilla.org, “global trustee”

- Later, same hacker compromised DigiNotar
  - Exploited vulnerability in public-facing server; all core certificate servers in a single Windows domain, controlled by a single admin password (Pr0d@dm1n)
  - Attack detected by Chrome because of key pinning
  - Rogue certificates issued for *.google.com, Skype, Facebook, www.cia.gov, and 527 other domains

---

**TrustWave**

- In Feb 2012, admitted issuance of an intermediate CA certificate to a corporate customer
  - Purpose: “re-sign” certificates for “data loss prevention”
  - Translation: forge certificates of third-party sites in order to spy on employees’ encrypted communications with the outside world

- Customer can now forge certificates for any site in world... and they will be accepted by any browser!
  - What if a “re-signed” certificate leaks out?

- Other examples (2015)
  - CNNIC
  - Symantec
  - Found using certificate transparency logs
Vulnerabilities in OpenSSL

- Debian OpenSSL entropy bug (2008)
  - Seed for the pseudo-random generator is derived only from process ID
    - Default maximum on Linux = 32768
  - Keypairs generated using this version are predictable

- Heartbleed (2014)
  - Incorrect handling of heartbeat messages, returning up to 64K of server memory
    - Memory may include the server’s private key (reported attack success rate ~0.02%)
    - Also visitor IPs, user emails and passwords, session cookies, etc.
  - The keys are not vulnerable by themselves
    - But they may have been compromised were served using a vulnerable OpenSSL version in the past

Defense Against Rogue Certificates: Revocation

- Many valid reasons to revoke a certificate
  - Private key corresponding to the certified public key has been compromised
  - User stopped paying his certification fee to the CA and the CA no longer wishes to certify him
  - CA’s certificate has been compromised!

- Expiration is a form of revocation, too
  - Many deployed systems don’t bother with revocation
  - Re-issuance of certificates is a big revenue source for certificate authorities
Certificate Revocation Mechanisms

- Online revocation service
  - When a certificate is presented, recipient goes to a special online service to verify whether it is still valid

- Certificate revocation list (CRL)
  - CA periodically issues a signed list of revoked certificates
  - Can issue a “delta CRL” containing only updates

Heartbleed Measurement


- Repeated scans using ZMap, starting 2 days after disclosure
  - Scanned Alexa top 1 million
  - Also 1% sample of IPv4 address space

- Checked for non-compliant behavior of vulnerable servers
  - Respond with random padding to requests with 0 length

- Estimate that 24–55% of Alexa sites were originally vulnerable
Heartbleed Patching


Up to 12% of sites still vulnerable at start of scan

Comparison with Debian Entropy Bug

Vulnerability Survival At Exploit Time
[Nappa et al., “Attack of the Clones”]

• Median percentage of hosts patched: **14%**
  – Considering both proof-of-concept and real-world exploits
  – Only one real-world exploit found more than 50% hosts patched

• These numbers must be interpreted as **upper bounds**

Threat Models Summary

• Software vulnerability
  – Threat: May allow the attacker to install malware on the host or to extract sensitive information (e.g. private keys, passwords)
  – Solution: Patch the vulnerable software
    • Process may be manual or automated

• Compromised certificate
  – Threat: Man-in-the-middle, masquerade as legitimate site
  – Solution: Revoke the certificate and issue a new one
    • Process is manual
Review of Lecture

• What did we learn?
  – Attacks using compromised certificates
  – Certificate revocations

• Sources
  – Vitaly Shmatikov and Zakir Durumeric

• What’s next?
  – Automatic exploit generation