Securing the SSL/TLS channel against man-in-the-middle attacks: Future technologies - HTTP Strict Transport Security and Pinning of Certs

Tobias Gondrom

OWASP London
Chair of IETF Web Security WG

tobias.gondrom@gondrom.org
Securing the SSL/TLS channel against man-in-the-middle attacks:

Future technologies - HTTP Strict Transport Security and Pinning of Certs

Tobias Gondrom

OWASP London
Chair of IETF Web Security WG

tobias.gondrom@gondrom.org
Tobias Gondrom

- 12 years information security experience (Global Head of Security, CISO, CTO)
- 10 years application development experience
- Information Security & Risk Management, Research and Advisory, Managing Director, Thames Stanley Ltd.
- Author of Standards on Digital Signatures and Secure Archiving
- Member of the IETF Security Directorate
- London OWASP chapter board member, OWASP Global Industry Committee [www.owasp.org](http://www.owasp.org)
Defending against MITMA

- Past Attacks/Breaches
- Insufficient Transport Layer Protection
- Possible Solutions
  - HSTS - Secure Channels: Strict Transport Security
  - Cert Pinning
- When
Defending against MITMA

- Past Attacks/Breaches
- Insufficient Transport Layer Protection
- Possible Solutions
  - HSTS - Secure Channels: Strict Transport Security
  - Cert Pinning
- When
CA breaches
March 15th 2011: Comodo breach

• Nine fake certificates for seven domains were issued: mail.google.com, login.live.com, www.google.com, login.yahoo.com (three certificates), login.skype.com, addons.mozilla.org, and global trustee

• Hacked several times afterwards
CA breaches
June (?) 2011: DigiNotar breach

• Discovered on June 19th

• July 10, 2011: wildcard cert issued for Google, subsequently used by unknown persons in Iran to conduct a man-in-the-middle attack against Google services

• August 28, 2011, certificate problems were observed on multiple Internet service providers in Iran

• Tor Project has published extensive updates on the scope of the attack, including a list of 531 fraudulent certificates issued by DigiNotar
CA breaches
June (?) 2011: DigiNotar breach

• All browser vendors remove trust of DigiNotar swiftly, e.g. August 30, 2011: Mozilla removed DigiNotar certificates from their list of trusted CAs (via patches etc.)

• September 20, 2011 – DigiNotar filed for bankruptcy

• Remark: Google Chrome users were protected from this attack because Chrome was able to detect the fraudulent certificate due to pinning.

• Statements have appeared that the DigiNotar attacker is the same person who attacked Comodo earlier

• The attacker claims to be an individual Iranian who has chosen to help the government monitor individuals' communications. Additionally, he claims to have compromised four additional as-yet-unspecified certificate authorities.
MITMA - TLS attack

Attacker replaced Server cert with own compromised cert and could read all communication (incl. passwords) in the clear.
The situation

• Browsers trust CA certificates for all domains equally (any trusted CA can sign for any identity, true or fake, e.g. google.com, paypal.com, ...)

• hundreds of CAs

• From 46 countries/jurisdictions

• If a single one is broken, all TLS/SSL domains are prone to attacks
From EFF: SSL Observatory

- 1,482 CA Certificates trustable by Windows or Firefox
- 1,167 distinct issuer strings
- 651 organizations, but ownerships & jurisdictions overlap
- (If a CA can sign for one domain, it can sign for any domain.)
Defending against MITMA

- Past Attacks/Breaches

- Insufficient Transport Layer Protection

- Possible Solutions
  - HSTS - Secure Channels: Strict Transport Security
  - Cert Pinning
  - When
OWASP Top 10 – Insufficient Transport Layer Protection

- A1: Injection
- A2: Cross-Site Scripting (XSS)
- A3: Broken Authentication and Session Management
- A4: Insecure Direct Object References
- A5: Cross Site Request Forgery (CSRF)
- A6: Security Misconfiguration
- A7: Failure to Restrict URL Access
- A8: Insecure Cryptographic Storage
- A9: Insufficient Transport Layer Protection
- A10: Unvalidated Redirects and Forwards
What’s the problem

- Some are not using / not mandating TLS/SSL
- Relies on trust relationships (trust on first use / trusted source)
- Weak channel protection
- Authentication & leakage of credentials

=> Today, Web Applications try to fix this on the Application level with little support of the underlying infrastructure
A9 – Insufficient Transport Layer Protection

Transmitting sensitive data insecurely

- Failure to identify all sensitive data
- Failure to identify all the places that this sensitive data is sent
  - On the web, to backend databases, to business partners, internal communications
  - Failure to properly protect this data in every location

Typical Impact

- Attackers access or modify confidential or private information
  - e.g., credit cards, health care records, financial data (yours or your customers)
- Attackers extract secrets to use in additional attacks
- Company embarrassment, customer dissatisfaction, and loss of trust
- Expense of cleaning up the incident
- Business gets sued and/or fined
Insufficient Transport Layer Protection

1. External attacker steals credentials and data off network

2. Internal attacker steals credentials and data from internal network

External Victim

Business Partners

Employees

External Attacker

Internal Attacker
Still not using SSL?

Now: Redirect to https before login.

How about you?
Facebook hjälp

Förnamn:
Efternamn:
Din e-postadress:
Välj lösenord:
Jag är: Ange kön:
Födelsedag: Dag: Månad: År:

Gå med

Skapa en sida för en kändis, ett band eller ett företag.
Common attack vectors

Attacks

- SSL downgrading
- Use of fake SSL certs
- SSL stripping
Moxie’s SSL Strip

Terminates SSL
Changes https to http

Normal https to the server
Acts as client
Moxie’s SSL Strip

Secure cookie?
Encoding, gzip?
Cached content?
Sessions?

Strip the secure attribute off all cookies.
Strip all encodings in the request.
Strip all if-modified-since in the request.
Redirect to same page, set-cookie expired
A9 – Avoiding Insufficient Transport Layer Protection

Use the mechanisms correctly

- Use TLS on **all** connections with sensitive data
- Use standard strong algorithms (disable old SSL algorithms)
- Manage keys/certificates properly
- Verify SSL certificates before using them
- Use proven mechanisms when sufficient
  - E.g., SSL vs. XML-Encryption

Defending against MITMA

- Past Attacks/Breaches
- Insufficient Transport Layer Protection

Possible Solutions
- HSTS - Secure Channels: Strict Transport Security
- Cert Pinning

- When
Who – Introducing the Players

• OWASP
• Top Ten
• Browser Security Day at OWASP Summit

• IETF
• Web Security WG
• Browser Vendors
• Secure Web-sites of critical information and payment systems (e.g. paypal, google, ebay, ...)
• Security Researchers and Plug-in developers for browsers
What’s been done / what’s coming

• Secure Channel:
  • HSTS Strict Transport Security
  • Cert Pinning
  • TLS cert pinning in DNSSEC

• Other methods:
  • Moxie’s Convergence (browser plug-in)
HSTS - Secure Channels: Strict Transport Security

• Server declares “I only talk TLS”

• Example:
  HTTP(S) Response Header:
  Strict-Transport-Security: max-age=15768000; includeSubDomains

• Header can be cached and also prevents leakage via subdomain-content through non-TLS links in content

• Weakness: “Trust on first use”

• Possible pre-loaded HSTS in browsers

• Already first deployments
Cert Pinning (1)

draft-ietf-websec-key-pinning-01

- Server identities tend to be long-lived, but clients have to re-establish the server's identity on every TLS session.

- How could Google/Chrome be resilient to DigiNotar attack?

- Google built-in in Chrome "preloaded" fingerprints for the known public keys in the certificate chains of Google properties. Thereby exposed the false *.google.com certificate DigiNotar signed.
Cert Pinning (2)

But....

.....preloading does not scale, so we need something dynamic:

=> Could use an HTTP header

i.e. transmit the SHA1 or SHA256 hash of the Subject Public Key Info structure of the X.509 certificate. (You could pin to end entity, intermediary, root. Select your degree of precision.)
Cert Pinning - Syntax

Header add Public-Key-Pins "max-age=10000; pin-sha1="0bT42aoSpAqWdY9WfRfL7i0HsVk="; pin-sha1="hvfN/qlp/zhXR3cuersh2Z7g=""
Cert Pinning - parameters

• List at least 2 certs: 1 live pin (a hash of an SPKI in the current cert chain) and at least one backup pin (a hash of an SPKI not in the current cert chain).

• Clients remember the most recently seen set of pins for max-age seconds after it was most recently seen.

• Clients drop TLS connections if not using the listed certs.
Cert Pinning – possible problems

Possible Problems:

• Bootstrap – “trust on first use”
  • Pre-loaded browser

• Servers might accidentally "brick" themselves (pin for a long time to an SPKI which is later lost, for example) – reason why backup cert is mandatory

• Attackers with ISP capabilities / man-in-the-middle access may try to “brick” domains for users even when outside of their reach (imagine: Iranian travelling abroad and no longer able to access Google, etc.)

• Recovery / cache flush mechanisms
Other Methods: Secure Channels: DNSSEC for TLS

- DNSSEC can be used to declare supported protocols for domains
- DNSSEC can be used to declare server certificate for domain

- Advantage: Advantage of trusted signed source
- Disadvantage: long time to deploy
Other Methods: Moxie’s Convergence – plug-in

Ask trusted party about cert

Notaries
Defending against MITMA

- Past Attacks/Breaches
- Insufficient Transport Layer Protection
- Possible Solutions
  - HSTS - Secure Channels: Strict Transport Security
  - Cert Pinning
- When
When - Timeframes

- HSTS Strict Transport Security – now
- Cert Pinning Q1 2013
- TLS in DNSSEC – 201?
Join the discussion

Ideas / feedback / participation welcome

IETF Websec:
http://tools.ietf.org/wg/websec/charters

Or drop me an email:
tobias.gondrom@gondrom.org
Questions?
Thank you