Universal SSL

Nick Sullivan
@grittygrease
Real Real World Crypto: HTTPS
HTTPS Myths

• Only for banking
• Only for authentication
• Too hard
HTTPS is used for

- Visitor privacy
- Invasive intermediaries
- SEO?
First some good news...

realworldcrypto.com does *not* have any TLS vulnerabilities
The bad news

This webpage is not available

Details
Who else is not using HTTPS?
And at the low end...

- Personal sites
- Small businesses
- Shared hosting (Github pages, etc.)
Reasons at high end

- Sysadmin time/training
- Business process and risk
- Vendor cost (CDN, Hardware)
- Third party liability
- Mixed content warnings from ads
Reasons at low end

• Certificates cost money
• Hosting provider capabilities
• Setting up HTTPS is complicated
• Fixing vulnerabilities
Goal

Get more sites on HTTPS
How?

HTTPS as a service
CloudFlare Reverse Proxy

- Bandwidth saved by CloudFlare
- Bandwidth you pay for
Potential issues

- Certificate Management
- Scaling
- Performance
Problem

Certificate Management
Solution

Automated Certificate Issuance
How does a CA validate a site?

- Domain validation (DV)
- Organization validation (OV)
- Extended validation (EV)
How does a CA validate a site?

- Domain validation (DV)
  - WHOIS email
  - DNS
  - HTTP
Whois email

$ whois realworldcrypto.com

The Registry database contains ONLY .COM, .NET, .EDU domains and Registrars.

Domain Name: realworldcrypto.com
Registry Domain ID: 1839854081.DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.register.com
Registrar URL: http://www.register.com
Updated Date: 2013-12-20T05:00:00Z
Creation Date: 2013-12-20T16:52:54Z
Registrar Registration Expiration Date: 2023-12-20T05:00:00Z
Registrar: Register.com, LLC.
Registrar IANA ID: 9
Admin Name: Dan Boneh

Admin Email: dabo@cs.stanford.edu
DNS Validation

• If you control DNS, you control the site
• Add a TXT record to DNS with token from CA

$ dig realworldcrypto.com TXT
realworldcrypto.com. 14399 IN TXT "google-site-verification=8-V5SmsK-pBf9PLCE49ACqFCX4qymWylbNVFaIDbtXc"
HTTP Validation

• If you control page content, you control the site
• Add a meta-tag to HTML

<meta name="validator" content="...">
CloudFlare CDN

CA

HTTP GET

CSR

Certificate

Proof
Problem

Certificate Management
Problem

Scaling
Customer Power Law

- High-end enterprises: 1,000s
- Businesses with budgets: 10,000s
- Cost sensitive sites: 100,000s
- Free customers: 1,000,000s

All numbers approximate for illustration.
Assumptions

• One IP address per site

• Web server can handle around 10,000 certificates

• Service owns 10,000 IPv4 addresses
High-end enterprises

- 1,000 sites
- 1,000 certificates
- Easy to handle
Third party liability?

- Keyless SSL
  - Keep private key on premises
  - Open signing oracle to proxy
  - Proxy splits handshake
Hello! Let's start a encrypted conversation using TLS 1.2.

I want to talk to bank.com
I know the following cipher suites:
- ECDHE and RSA with 128bit AES in GCM mode and SHA256
- RSA with 128bit AES in GCM mode and SHA256

Here's a randomly chosen number:
3d86a5..04

Hi there, I think we can chat.

Let's use the cipher:
RSA with 128bit AES in GCM mode and SHA256
Here's my random number:
ca35f0..13
Here's my certificate chain:
[bank.com's certificate]

This certificate checks out: it was issued to bank.com and digitally signed by a certificate authority I trust.
Here's a secret encrypted with the RSA public key I took from your certificate:
[encrypted pre-master secret]
We can both derive the same key using this secret and the random numbers we exchanged.

I have decrypted the secret and derived the key.
From now on let's use the key to encrypt what we say.

[It's so great to speak privately]
[Can you get me the current balance of my checking account?]

[Sure thing, you have $12.05 left in that account]

Hey, you're the one with the key for bank.com, can you decrypt this for me?
[encrypted pre-master secret]

Sure, here's the decrypted message:
[pre-master secret]
Keyless SSL

Example handshake performance

No proxy: 895ms
Proxy with keyless: 346ms
Proxy with key: 149ms
Businesses with budgets

- 10,000 sites
- 10,000 certificates
- Near capacity for stock web server
Cost sensitive sites

- 100,000 sites
- 100,000 certificates
- This begins to get tricky
Subject Alternative Names

- Associate values to a certificate (DNS Name, IP)
Solution to certificate problem

• Put multiple sites on same SAN
• ~40 or so SANs before performance is affected

• Sites can’t spoof each other: managed key
Cost sensitive sites

- 100,000 sites
- 10,000 multi-SAN certificates
- Acceptable web server
Free customers

• 1,000,000 sites
• 100,000 multi-SAN certificates?
• Even with SANs, this doesn’t scale
Lazy Loading

- Load certificates into memory when needed
- No need to reload web server
- 100,000 certificates are possible
How many IP addresses?

- Let’s assume one IP per server per site
CloudFlare’s Global Network
IP addresses needed

• 1 certificate per IP per PoP
• 100,000 certificates
• ~3 million IPs for 30 pops
• CloudFlare only has ~1 million IP addresses
• Only ~16 million in a Class A network
Unicast vs. Anycast Networks

- **Unicast:** each machine gets its own IP
- **Anycast:** each machine gets the same IP
  - Network handles routing via BGP
Source addresses for one IP
As seen from Singapore
As seen from Santiago
Using Anycast

- 1 certificate per IP, no matter how many servers
- 100,000 certificates
- 100,000 IPs

- Still not ideal
Solution

Server Name Indication
(SNI)
What is it?

• TLS extension that adds the hostname to ClientHello
• Allows “virtual hosting”
• Multiple certificates behind one IP
Downside

- Not universally supported
## SNI Support

<table>
<thead>
<tr>
<th>OS Browser</th>
<th>Windows XP</th>
<th>Android</th>
<th>iOS/ MacOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS Browser</td>
<td>X</td>
<td>3.0+</td>
<td>iOS 4+ MacOS 10.5+</td>
</tr>
<tr>
<td>Chrome</td>
<td>3.0+</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Firefox</td>
<td>2.0+</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
THE TIMES
THEY ARE
A-CHANGIN'

BOB
DYLAN
Meanwhile...

- Windows XP end of life
- Microsoft and Google dropping support for SHA-1
- POODLE exploit causes SSL v3.0 to be dropped
SHAAAA-256 Support

<table>
<thead>
<tr>
<th>OS Browser</th>
<th>Windows XP</th>
<th>Android</th>
<th>iOS/MacOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>SP3</td>
<td>2.3+</td>
<td>iOS 3+ MacOS 10.5+</td>
</tr>
<tr>
<td>Firefox</td>
<td>1.5+</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
no SNI support, yes SHA-256

<table>
<thead>
<tr>
<th>OS Browser</th>
<th>Windows XP</th>
<th>Android</th>
<th>iOS/MacOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>XP SP3</td>
<td>2.3 only</td>
<td>iOS 3 only</td>
</tr>
<tr>
<td></td>
<td>3.0+ SP2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>3 – 25 SP3</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Firefox</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Use SNI

- 1,000,000 sites
- 100,000 multi-SAN certificates
- 10 certificates per IP
- 10,000 IPs

- Works on modern browsers
Problem

Scaling
Problem

Performance
Potential performance issues

• Server CPU usage
• Handshake latency
• Is the site slower with HTTPS?
CPU utilization - bulk crypt

- Modern Intel CPUs have instructions for AES
  - Advanced Encryption Standard Instruction Set (AES-NI)
  - Carry-less Multiplication (CLMUL)
- ChaCha20/Poly1305 for mobile — soon
- Encrypt and decrypt at line rate
CPU utilization - handshake

• Elliptic curve certificates
  • Assembly implementation of P256 in OpenSSL
  • 10x less computation than RSA on server side
Latency - handshake

• Session resumption
  • Session tickets, globally resumable
  • Session IDs, resumable within a PoP
SSL Handshake (Diffie-Hellman)

1. Visitor sends hello, client random, and cipher suites supported

2a. Server sends server random and public key certificate (also sent is a session ID for session resumption)

2b. The key signs for client random, server random, and public key certificate

3. Server sends the server DH parameter and a signature

4. Visitor sends the client DH parameter

Both the visitor and CloudFlare derive identical premaster secrets from the server DH parameter and client DH parameter.

Client random

Server random

Public key certificate

Server DH parameter

Signature from key server

Client DH parameter

Premaster secret

Session key

Private key

Client DH parameter

Premaster secret

Session key
Session resume with session ID

1. Visitor sends hello, client random, and cipher suites supported
2. Visitor sends session ID

Both CloudFlare and the visitor have an identical session key, and the encrypted session can resume.

CloudFlare takes the session ID and looks up the corresponding session key from the encrypted database.
Session resume with session ticket

1. Visitor sends hello, client random, and cipher suites supported

2. Visitor sends session ticket

Both CloudFlare and the visitor have an identical session key, and the encrypted session can resume

CloudFlare decrypts the session ticket using the ticket key
Latency - HTTP

• Use SPDY
HTTP vs HTTPS Test

Encrypted Websites Protect Our Privacy and are Significantly Faster

Compare load times of the unsecure HTTP and encrypted HTTPS versions of this page. Each test loads 360 unique, non-cached images (2.04 MB total). For fastest results, run each test 2-3 times in a private/incognito browsing session.

Done! Please try HTTPS.
HTTP vs HTTPS Test

Encrypted Websites Protect Our Privacy and are Significantly Faster

Compare load times of the unsecure HTTP and encrypted HTTPS versions of this page. Each test loads 360 unique, non-cached images (2.04 MB total). For fastest results, run each test 2-3 times in a private/incognito browsing session.

3.171 s
59% faster than HTTP
Problem

Performance
Problems

- Certificate Management ✓
- Scaling ✓
- Performance ✓
UNIVERSAL SSL
EVERYBODY GETS SSL!
Universal SSL

• No-hassle HTTPS
• ECDSA certificates
• SNI only
• **Free** and automatic

• Over a million new sites with HTTPS!
Universal SSL

• Modern browsers only
Some issues left to solve

- Back-end encryption
- Ad networks and mixed content warnings
CloudFlare flexible SSL — front-end over TLS, back-end unencrypted
Automatic Back-end Encryption

• Automatic issuance of certificates for origin
• CloudFlare Origin CA
• Let’s Encrypt ???
CloudFlare full SSL (strict) — front-end over TLS, back-end over TLS (validated)
Mixed content warnings

- Invite me back next year when we’ve fixed it
Universal SSL

Nick Sullivan
@grittygrease