Handling SSL and TLS 1.3 at Scale While Maintaining Privacy

Brian Deitch, SE, Zscaler

Udayan Palekar, Director of Product Management, Zscaler
Safe harbor

Forward-looking statements

This presentation has been prepared by Zscaler, Inc. ("Zscaler") for informational purposes only and not for any other purpose. Nothing contained in this presentation is, or should be construed as, a recommendation, promise or representation by the presenter or Zscaler or any officer, director, employee, agent or advisor of Zscaler. This presentation does not purport to be all-inclusive or to contain all of the information you may desire.

This presentation contains forward-looking statements. All statements other than statements of historical fact, including statements regarding our planned products and upgrades, business strategy and plans and objectives of management for future operations of Zscaler are forward-looking statements. These statements involve known and a significant number of unknown risks, uncertainties, assumptions and other factors that could cause results to differ materially from statements made in this message, including any performance or achievements expressed or implied by the forward-looking statements. Moreover, we operate in a very competitive and rapidly changing environment, and new risks may emerge from time to time. It is not possible for us to predict all risks, nor can we assess the impact of all factors on our business or the extent to which any factor, or combination of factors, may cause actual results or outcomes to differ materially from those contained in any forward-looking statements we may make. Additional risks and uncertainties that could affect our financial and operating results are included in our most recent filings with the Securities and Exchange Commission. You can locate these reports though our website at http://ir.zscaler.com or on the SEC website at www.sec.gov.

In some cases, you can identify forward-looking statements by terms such as “anticipate,” “believe,” “continues,” “contemplate,” “could,” “estimate,” “expect,” “explore” “intend,” “likely,” “may,” “plan,” “potential,” “predict,” “project,” “should,” “target,” “will” or “would” or the negative of these terms or other similar words. Zscaler based these forward-looking statements largely on its current expectations and projections about future events that it believes may affect its business. Actual outcomes and results may differ materially from those contemplated by these forward-looking statements. All forward-looking statements in this message are based on information available to us as of the date hereof, and we do not assume any obligation to update the forward-looking statements provided to reflect events that occur or circumstances that exist after the date on which they were made.
How did we get to TLS?
TLS History

- SSLv1/v2 (1994)
- SSLv3 (1995)
- TLS 1.0 (1999)
- TLS 1.1 (2006)
- TLS 1.2 (2008)
- Insecure Renegotiation (2009)
- Beast Crime (2011)

- TLS 1.3 (2018)
- ROBOT (2017)
- DROWN (2016)
- Poodle Dire Freak LogJam (2015)
- Heartbleed (2014)
- RC4 Time Lucky 13 (2013)

- DOWNGRADE ATTACK (2019)
I want this website. Do you support TLS 1.2?

Sure do, let’s establish this connection.
TLS Exploit: POODLE
Padding Oracle On Downgraded Legacy Encryption

I want this website. Do you support TLS 1.2?
No

How about TLS 1.1?
No

How about TLS 1.0?
No

How about SSL 3.0?
Yes

MITM

Server
TLS 1.2
Still supports SSL v3!
TLS Exploit: POODLE
Padding Oracle On Downgraded Legacy Encryption

SEND

Ciphertext

Key

Initialization Vector (IV)

Plaintext

Cipher Block Chaining (CBC) mode decryption

RECEIVE

Ciphertext

Key

Plaintext

Ciphertext

Key

Plaintext

Ciphertext

Key

Plaintext

Server

TLS 1.2

Inside SSLv3

MITM
TLS Exploit: POODLE
Padding Oracle On Downgraded Legacy Encryption

Send

<table>
<thead>
<tr>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>MITM</td>
</tr>
</tbody>
</table>

Receive

<table>
<thead>
<tr>
<th>Server TLS 1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inside SSLv3</td>
</tr>
</tbody>
</table>

**BLOCK CIPHER**
GZV JFRXP YILDM
ULC OZABQ LEVIA
GRC HWLTG 0000

**BLOCK CIPHER**
GZV JFRXP YILDM
ULC OZABQ LEVIA
GRC HWLTG A0000
TLS Exploit: POODLE
Padding Oracle On Downgraded Legacy Encryption

SEND

MITM

DECRYPTED HTTP REQUEST
log=briandeitch&pwd=password-submit=
Log+In&redirect_to=https%3A%2F%2Fbriandeitch.com

Server
TLS 1.2

Inside SSLv3

RECEIVE

Users
TLS Threat Trends
HTTPS (TLS) adoption via Google
HTTPS (TLS) adoption via Zscaler

TRAFFIC BREAKDOWN

- HTTPS: 78%
- HTTP: 18%
- TUNNEL: 4%
TLS seen by Zscaler

**TLS SESSIONS**

- **TLS 1.2**: >98%
- **TLS 1.0**: 1.47%
- **TLS 1.1**: 0.5%
- **SSL 3.0**: <0.01%
SSL/TLS-based threats

Top phished brands over HTTPS

- 58% Microsoft
- 12% Facebook
- 10% Amazon
- 10% Apple
- 4% Adobe
- 4% Dropbox
- 2% DocuSign
TLS-based malware threats

Top 5 C2 activity over HTTPS

- 51% Trickbot
- 15% Emotet/Heado
- 12% Qadars
- 10% Zbot variants
- 2% Dridex
TLS certificate types

- 89% Domain Validated (DV)
- 11% Organization Validated (OV)
- 0% Extended Validation (EV)
TLS CAs blocked

Security blocks by SSL/TLS certificate CA distribution

<table>
<thead>
<tr>
<th>Certificate CA</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let’s Encrypt</td>
<td>50%</td>
</tr>
<tr>
<td>COMODO</td>
<td>19%</td>
</tr>
<tr>
<td>DigiCert</td>
<td>7%</td>
</tr>
<tr>
<td>cPanel</td>
<td>7%</td>
</tr>
<tr>
<td>RapidSSL</td>
<td>5%</td>
</tr>
<tr>
<td>Add Trust</td>
<td>5%</td>
</tr>
<tr>
<td>GoDaddy</td>
<td>4%</td>
</tr>
<tr>
<td>Google</td>
<td>3%</td>
</tr>
</tbody>
</table>
TLS CAs blocked

Security blocks SSL/TLS certificate CA distribution

- 50% Let’s Encrypt
- 19% COMODO
- 7% DigiCert
- 7% cPanel
- 5% RapidSSL
- 5% Add Trust
- 4% GoDaddy
- 3% Google
What’s New in TLS 1.3
IS TLS 1.3 REALLY FASTER?

TLS 1.2 vs. TLS 1.3

**CLIENT**
- Client Hello
- Supported cipher suites
- Key Share
- Finished
- HTTP GET

**SERVER**
- Server Hello
- Chosen cipher suite
- Key Share
- Certificate & Signature
- Finished
- HTTP Respond

**CLIENT**
- Client Hello
- Supported cipher suites
- Key Share
- HTTP GET

**SERVER**
- Server Hello
- Chosen cipher suite
- Key Share
- Certificate & Signature
- Finished
- HTTP Respond

**TLS 1.3 HAS 1 LESS ROUND TRIP**
What about Resumption?

**TLS 1.2** -vs- **TLS 1.3**

**CLIENT**
- Client Hello
- Session ID/Ticket
- Finished
- HTTP GET

**SERVER**
- Server Hello
- Finished
- HTTP Respond

**CLIENT**
- Client Hello
- Session Ticket (PSK)
- Key share
- HTTP GET

**SERVER**
- Server Hello
- Key share
- Finished
- HTTP Respond

**TLS 1.3 0-RTT**
TLS 1.3 0-RTT Caveats

- Replayable
- Not Forward Secret
Capture this 0-RTT HTTP POST:

POST /sendmoney.php?Account=12345678&Amount=10

To this:

POST /sendmoney.php?Account=987654321&Amount=100
<table>
<thead>
<tr>
<th>REMOVED</th>
<th>SIMPLIFIED</th>
<th>ADDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Static RSA Handshake</td>
<td>• Fixed DHE Groups</td>
<td>• Full handshake signature</td>
</tr>
<tr>
<td>• DROWN</td>
<td>• Logjam</td>
<td>• FREAK, Logjam</td>
</tr>
<tr>
<td>• CBC MtE modes</td>
<td>• RSASSA-PSS</td>
<td>• Downgrade protection</td>
</tr>
<tr>
<td>• Vaudenay, Lucky 13, POODLE</td>
<td>• BB’06</td>
<td>• Improve</td>
</tr>
<tr>
<td>• RC4</td>
<td>• AEAD Nonce</td>
<td>• TLS_FALLBACK_SCSV</td>
</tr>
<tr>
<td>• SHA1, MD5</td>
<td>• Nonce-Disrespecting Adversaries</td>
<td>• Ratcheted resumption with optional (EC)DHE</td>
</tr>
<tr>
<td>• Compression</td>
<td>• Session IDs and Tickets</td>
<td>• Curve 25519 and 448</td>
</tr>
<tr>
<td>• CRIME</td>
<td>• Tickets + PSK</td>
<td></td>
</tr>
<tr>
<td>• Renegotiation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
END OF DEITCH
Interactive Reports: SSL Traffic Overview
Cipher usage distribution
SSL/TLS version distribution
Invalid certificate breakdown
Extended SSL cipher logging to uninspected SSL traffic

Let's get rid of TLS 1.0!
Zscaler and TLS 1.3
PFS – Perfect Forward Secrecy with Elliptical Ciphers

ECDHE PFS Cipher support
Recently added ECDHE-ECDSA
Provides capability to decrypt SSL connections to servers which exclusively support ECDSA ciphers (common with CloudFlare hosted sites)

Supported ECDHE Cipher Suites
Zscaler supports the following ECDHE cipher suites for Perfect Forward Secrecy (PFS) depending on the TLS protocol:

<table>
<thead>
<tr>
<th>TLS Protocol</th>
<th>ECDHE Cipher Suite</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLS 1.0</td>
<td>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA</td>
</tr>
<tr>
<td>TLS 1.1</td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td>TLS 1.2</td>
<td>TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA384</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA384</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td></td>
<td>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA</td>
</tr>
</tbody>
</table>
Zscaler and TLS 1.3

Today and future

• Supporting TLS 1.3 today – downgrade to TLS 1.2
  • TLS 1.2 has no known vulnerability and is widely adopted due to PCI mandate.
    • If a client makes a connection to/via Zscaler with TLS 1.3 and mentions TLS 1.2 as supported version in its handshake, Zscaler would use TLS1.2 to negotiate the connection successfully.
    • While communicating with a server, Zscaler would always propose TLS 1.2, and if the server agrees, the connection is successful.

• Native Support for TLS 1.3
  • In progress – we’re already halfway there with PFS!
  • Today - If the server exclusively requires TLS 1.3, the connection would be considered as “Un-decryptable traffic” and the customer may choose to allow or block this based on policy configuration.
Our Goal – comprehensive, granular SSL inspection policy

Rule-based policies to define what SSL traffic should be intercepted

Granular policy controls:
- Content Categories
- Applications
- Users/Groups/Departments
- Locations
- Device types

Bypassed traffic can still be evaluated against other policies

Choice of additional decryption parameters and certificate
Emerging and experimental technologies

Going beyond TLS1.3 standard

• TLS 1.3 sends SNI in Client Hello.

💡 Let’s encrypt the SNI == ESNI
• Only works with TLS 1.3 – not previous TLS versions
• ESNI uses encryption derived from a public key uploaded to the server’s DNS record
• But…DNS is in cleartext (!)

• So…
💡 Let’s encrypt DNS!
DNS over HTTPS (DOH)
- Secures DNS communications over HTTPS
- Privacy concerns:
  - Expansion of control for the CDNs (e.g., Cloudflare DNS must be used for ESNI today)
  - e.g., Google public DNS - “Your client IP address is only logged temporarily (erased within a day or two), but information about ISPs and city/metro-level locations are kept longer for the purpose of making our service faster, better, and more secure.”

Could’t we simply intercept DNS over HTTPS? (uses port 443)

DOT – DNS over TLS
- Limited adoption of standard
- Secures connection over dedicated port 853
Summary – SSL inspection with Zscaler

Cloud scale, simple, future-proof

- **Cloud scale:**
  - Custom TCP and SSL stack to handle encrypted traffic on a global scale. Zscaler inspects traffic in real time, including encrypted traffic, so hackers and their malware can be easily identified.

- **Simple:**
  - The Zscaler security-as-a-service architecture operates seamlessly – no new planning requirements or costly hardware upgrades due to TLS 1.3 and new ciphers.

- **Future-proof:**
  - Zscaler has supported TLS 1.2 for several years and has also extended support for PFS with ECDHE and ECDSA ciphers.
  - With Zscaler, supporting TLS 1.3 becomes a seamless change vs. a major configuration overhaul.
Next steps
Learn more about SSL inspection with Zscaler

- Best practices for deploying SSL inspection

- Product overview
  https://www.zscaler.com/products/ssl-inspection

- Configuration
  https://support.zscaler.com/hc/en-us/articles/205060065-How-do-I-deploy-SSL-inspection-

- Custom SSL certificates
  https://support.zscaler.com/hc/en-us/articles/205060065-How-do-I-deploy-SSL-inspection-
Please share your feedback on this session!

Four easy steps:

2. Select Session Ratings
3. Choose this session from the list
4. Provide your feedback
Thank you!
In an open forum with Zscaler employees, partners, and customers

Your knowledge and learn from experts in cloud security

The conversation at community.zscaler.com