

Active TLS Stack Fingerprinting: Characterizing TLS Server Deployments at Scale

Markus Sosnowski*, Johannes Zirngibl*, Patrick Sattler*,Georg Carle*, Claas Grohnfeld**, Michele Russo**, Daniele Sgandurra**

*Chair of Network Architectures and Services, Technical University of Munich, Germany *Al4Sec, Huawei Technologies Munich, Germany



Three facts about the TLS:

- 1. It is currently the de facto standard for encrypted communication on the Internet¹
- 2. It is old, and it has grown to a complex ecosystem due to its continuous development²
- 3. Thus, during the handshake the client and server capabilities must be exchanged.
- → This meta-data allows to fingerprint the TLS stack (config, implementations, and hardware)

¹C. Labovitz, "Internet Traffic 2009-2019," in Proc. Asia Pacific Regional Internet Conf. Operational Technologies, 2019.

² P. Kotzias, A. Razaghpanah, J. Amann u. a., Coming of Age: A Longitudinal Study of TLS Deployment,* in Proc. ACM Int. Measurement Conference (IMC), 2018.

Background

TLS Fingerprinting

- Collecting TLS characteristics (⇒ represented as fingerprint)
- Build a database mapping fingerprints with not directly related data, e.g.:

Fingerprint	Indicates
771_1301	IETF webserver
771_1302	Nginx docker image
770_cf	TrickBot CnC server

Background

Example TLS 1.3 handshake

Client		Server
Ī-	Client Hello (CH) [ver., session info., Cipher Suites, _, Extensions {versions, ALPNs,}]	
	Server Hello (SH) [ver., session info., Cipher Suite, _, Extensions {version, ALPN,}]	
-	Change Cipher Spec	
•	Encrypted Extensions, Certificate, Certificate Verify,, Finished, [Application Data]	
-	Changed Cipher Spec	
	, Finished, [Application Data]	
ţ	Application Data	→ļ
		_

Legend: fingerprintable server data, encrypted data

Motivation

Future Applications:

1. Enhance existing Intrusion Detection Systems Servers from network flows are fingerprinted on-demand and results compared with known malicious ones

2. Internet-wide measurements

Security researchers use fingerprinting to find previously unknown threats

3. Monitoring of own Servers

Deviations from a fingerprints baseline can indicate an unintended software change or a malware infection

Goals

Problem: Early fingerprints with default CHs were not distinctive enough

This was due to the question-answer design of TLS, e.g.:



- \rightarrow use unusual CHs that trigger distinguishable behaviors
- \rightarrow find multiple CHs to increase the learned data
- \rightarrow find a trade-off between learned data and scan costs (time and impact)

Methodology

Research Questions

- 1. How can we relate similar TLS server deployments?
- 2. How can we improve the effectiveness of our CHs?
- 3. What is the performance of actual fingerprinting use-cases?

Methodology

Constructing Fingerprints

Extract data such that similar deployments have the same fingerprint

• A handshake is represented textually, e.g.,



• The final TLS fingerprint is a combination of multiple handshakes, e.g.,

771_1301..., 771_1302..., 770_fa..., ...

Effective Scanning Configurations

Research Questions

- 1. How can we relate similar TLS server deployments?
- 2. How can we improve the effectiveness of our CHs?
- 3. What is the performance of actual fingerprinting use-cases?

Effective Scanning Configurations

Challenge: Without knowledge about the implementation of every TLS server, it is impossible to select the ideal CHs for fingerprinting.

 $\rightarrow~$ However, we can optimize the effectiveness of the CHs

We propose and empiric design of CHs:

- 1. Measure the effectiveness with a metric (e.g., distinct fingerprints)
- 2. Perform measurement with a large amount of randomly generated candidates
- 3. Select the CHs that maximize the metric

This way we generated the 10 general-purpose CHs used in the following analyses

Research Questions

- 1. How can we relate similar TLS server deployments?
- 2. How can we improve the effectiveness of our CHs?
- 3. What is the performance of actual fingerprinting use-cases?

Overview

Long-running study over 30 weeks with weekly measurements

Toplists

- Alexa Top 1 Million
- The Majestic Million

Blocklists

- abuse.ch Feodo Tracker
- abuse.ch SSL Blacklist

 \rightarrow In total, we collected 104 Million fingerprints

CDN Server Detection

CDNs enable us to evaluate the approach because

- they are a single actor deploying TLS servers on a large-scale (large amount of data samples)
- their servers can be verified (AS, HTTP headers, and x509 certificates) to get a ground-truth
- \rightarrow Evaluated a CDN detection based on past observations per week

Metrics:

- Precision $\left(\frac{TP}{TP+FP}\right)$: "correct classifications"
- Recall $\left(\frac{TP}{TP+FN}\right)$: "detected CDN servers"

CDN Server Detection



Note: The approach enabled us to detect off-net CDN servers in sometimes unexpected ASs

CnC Server Detection

Fingerprinting allows to detect potentially malicious servers

- We analyzed new additions to the lists based on past information
- · We considered how often a fingerprint is observed from CnC servers vs. from toplist servers
- This results in a score \in [0, 1] how certain we are that we found a CnC server
- If the score was above a tune-able threshold, the server is classified as CnC server

CnC Server Detection



 \rightarrow Fingerprints work great if combined with additional indicators

Conclusion

- Proposed a selection of TLS handshake features and their encoding as fingerprint to relate servers
- Provided a methodology for finding effective CHs and 10 general-purpose CHs that maximize information extraction from servers
- Demonstrated the potential of Active TLS Stack Fingerprinting based • on detecting CDN and CnC servers
- The approach resulted in more effective fingerprinting compared to related work JARM³
- Open-sourced our data and code •

https://active-tls-fingerprinting.github.io

