

TLFeat

A Dedicated, Robust and Fast TLS
Feature Extraction Tool

User Guide

Version: 0.90
Date: 2023-07

Overview

TLSfeat is an easy-to-use and dedicated TLS feature extraction tool, which extracts comprehensive features from pcaps automatically, robustly and efficiently. TLSfeat aims to relieve the pain of TLS feature extraction, which can drastically accelerate analysis pipeline. TLSfeat is intended to be used by data scientists and security analysts to explore massive data at scale and in speed for building models and threat hunting, which can also be used in academic research.

TLSfeat has been well tested in Linux(Centos, Fedora, Ubuntu) and Mac OSX(x86, M1) using thousands of public pcaps up to 12G single pcaps.

TLSfeat software and its documents are free to use and distribute licensed by [CC BY-NC-ND 4.0](https://creativecommons.org/licenses/by-nc-nd/4.0/).

Design goals

- Easy to use and setup
- Comprehensive feature extraction
 - meta, statistical
 - SPLT, byte distribution
 - TLS(handshakes, fingerprints, certificates)
- Robust to process real-world traffic with broken and malicious messages
- Native support of parallel analysis for significant performance boost
- Cross-platform

Key concepts

Parallelization

Parallel analysis of massive pcaps, especially small pcaps, can gain substantial performance boost. TLSfeat supports native parallelization with multithreading.

TLSfeat analyzes a directory at one time, which contains one or many pcaps. A run of TLSfeat is a **job**; analysis of a single pcap is a **task**.

Feature file

Feature file is a JSON file, which stores structured features of a pcap analyzed by TLSfeat. Feature files can be readily consumed by data analysis language (such as Python) and other tools.

Analysis modes

TLSfeat provides two analysis modes, *small mode* and *large mode*, which processes different sizes of pcaps.

	Small mode	Large mode
Pcap max size	100MB	1GB
Max pcaps of a job	100	10
Max threads	6	2

Notes

Analyzing large pcaps may consume substantial memory, so the thread number is limited. Use large mode at discretion.

Download and install

Hardware requirement

- Operating systems: Linux(x86_64)
- Memory: 8 GB or higher
- CPU: i5 or higher
- Disk: 2 GB or higher (recommended, depending on feature files)

Operating systems	Architecture	Support
Centos/Fedora	x86_64	✓
Ubuntu	x86_64	✓
Windows	x86_64	(support soon)

1. Install

TLSfeat is distributed as a binary, and the only dependency is the shared library of libpcap.

Centos/Fedora

```
sudo dnf install libpcap-devel
```

Ubuntu

```
sudo apt-get install libpcap-dev
```

For Ubuntu, please copy following command to create a symlink. Change [libpcap.so.0.8](#) to actual file name, and using [libpcap.so.1](#) as symlink name.

```
sudo ln -s /usr/lib/x86_64-linux-gnu/libpcap.so /usr/lib/x86_64-linux-gnu/libpcap.so.1
```

2. Setup path

TLSfeat binary is ``bin/tlsfeat``. Run TLSfeat conveniently in two ways.

1) add to PATH

```
echo PATH=$PATH:<tlsfeat bin dir> >> ~/.bashrc  
source ~/.bashrc
```

2) create a symlink

```
sudo ln -s <tlsfeat bin path> /usr/local/bin/tlsfeat
```

Usage

TLSfeat is used in command line with several options, which accepts both relative and absolute path of pcap and output dir.

Usage

```
tlsfeat -d <dir> [-o <output>] [-t <thread>]
          [--small|--large] [--no-color][--version]
          [--license]
```

OPTIONS

```
-d, --dir      [required]pcap dir path, default small mode

-o, --output   [optional]output dir path, default '../featsout'
               if not specified

-t, --thread   [optional]number of threads

--small        enable analysis of small pcaps, pcap max 100.0
               MB, max 100 pcaps, max 6 threads

--large        enable analysis of large pcaps, pcap max 1.0 GB,
               max 10 pcaps, max 2 threads.

--no-color     print in no color, default multicolor
--version      show version
--license      show license
```

Examples

The release contains several small pcaps; more sample pcaps can be [downloaded](#). The following commands use samples downloaded as examples (suppose pcaps are located at **/opt/pcaps**)

analyze small pcaps

```
tlsfeat -d /opt/pcaps/small_pcaps -o /opt/featsout/tests
```

analyze medium pcaps with default output dir and 4 threads

```
tlsfeat -d /opt/pcaps/medium_pcaps -t 4
```

analyze large pcaps

```
tlsfeat -d /opt/pcaps/large_pcaps --large
```

Printing in multicolors

TLSfeat provides nice printing in terminal with multicolors as default, which works well in terminal's dark theme. However, it may not work properly in terminal's light theme.

To print in mono color, use `--no-color` option.

2. Checkout artifacts

Artifacts generated by TLSfeat are saved in output directory. Output and its subdirectories will be created if they did not exist upon analysis.

There are four sub directories in the *output*.

- **feats**: feature files of each pcap
- **artifacts**: symbolic links of artifacts for each pcap
- **summary**: summary files of executed jobs
- **certs**: (empty)

Notes

1. As feature file name is identical to the pcap's name, feature files will be overwritten for repeated analysis for same output dir and same pcaps.
2. Watch the growing size of output dir, which could incrementally get very large especially for large pcaps; clean artifacts if necessary.
3. The performance of TLSfeat can be influenced by artifacts writing as well as disk, especially for large pcaps.

License

TLSfeat software and its documents are free to use and distribute licensed by CC BY-NC-ND 4.0.

TLSfeat uses several third party packages, see ThirdPartyNotices.txt.

Features

Meta features

Feature	Type	Description
pcap_id	str	pcap unique identifier, SHA256 of the first 8K bytes of the pcap
pcap_name	str	pcap name
stream_id	int	stream identifier, SHA256 of JSON(src_ip, src_port, dest_ip, dest_port, timestamp)
stream_index	int	TCP stream index in the pcap
src_ip	str	source IP(v4) address
src_port	int	source port
dest_ip	str	destination IP(v4) address
dest_port	int	destination port
timestamp	int	timestamp of the first packet of the stream in nanoseconds
duration	float	duration of the stream in seconds

Statistical features

Feature	Type	Description
Packet number		
pkt_total_num	int	bidirectional packet number
pkt_up_num	int	outbound packet number
pkt_down_num	int	inbound packet number
Packet length		
pkt_len_total_sum	int	sum of bidirectional packet length
pkt_len_total_max	int	maximum bidirectional packet length
pkt_len_total_min	int	minimum bidirectional packet length
pkt_len_total_mean	float	mean value of bidirectional packet length
pkt_len_total_std	float	standard variance of bidirectional packet length

Feature	Type	Description
pkt_len_up_sum	int	sum of outbound packet length
pkt_len_up_max	int	maximum outbound packet length
pkt_len_up_min	int	minimum outbound packet length
pkt_len_up_mean	float	mean value of outbound packet length
pkt_len_up_std	float	standard variance of outbound packet length
pkt_len_down_sum	int	sum of inbound packet length
pkt_len_down_max	int	maximum inbound packet length
pkt_len_down_min	int	minimum inbound packet length
pkt_len_down_mean	float	mean value of inbound packet length
pkt_len_down_std	float	standard variance of inbound packet length
Packet inter arrival time		
pkt_iat_total_sum	float	sum of bidirectional packet iat
pkt_iat_total_max	float	maximum bidirectional packet iat
pkt_iat_total_min	float	minimum bidirectional packet iat
pkt_iat_total_mean	float	mean value of bidirectional packet iat
pkt_iat_total_std	float	standard variance of bidirectional packet iat
pkt_iat_up_sum	float	sum of outbound packet iat
pkt_iat_up_max	float	maximum outbound packet iat
pkt_iat_up_min	float	minimum outbound packet iat
pkt_iat_up_mean	float	mean value of outbound packet iat
pkt_iat_up_std	float	standard variance of outbound packet iat
pkt_iat_down_sum	float	sum of inbound packet iat
pkt_iat_down_max	float	maximum inbound packet iat
pkt_iat_down_min	float	minimum inbound packet iat
pkt_iat_down_mean	float	mean value of inbound packet iat
pkt_iat_down_std	float	standard variance of inbound packet iat

SPLT features

Feature	Type	Description
mc_len	[float]	sequence of packet length in Markov matrix, array size 100
mc_time	[float]	sequence of packet iat in Markov matrix, array size 100

Byte distribution features

Feature	Type	Description
bd_dist	[float]	byte distribution of TCP payloads, array size 256
bd_std	float	standard variance of byte distribution
bd_entropy	float	entropy of byte distribution

TLS features

Feature	Type	Description
Client Hello		
tls_client_version	str	TLS version, e.g, TLSv1.1, TLSv1.2
tls_server_name	str	SNI, aka Server Name Indicator
tls_client_cipher_names	[str]	client cipher suites
tls_client_cipher_num	int	size of tls_client_cipher_names
Server Hello		
tls_server_version	str	TLS version
tls_server_cipher_name	str	server cipher suite
Fingerprints		
ja3	str	JA3 fingerprint
ja3s	str	JA3s fingerprint
ja3_fullstring	str	full string to construct JA3
ja3s_fullstring	str	full string to construct JA3s

Certificates		
cert_num	int	number of certificates in cert chain
cert_id	str	SHA256 of the cert
version	int	cert version
serial	str	serial number
not_before	str	cert start time
not_after	str	cert expire time
validity_days	int	cert validity days
issuer_common /country/state/city/org/ org_unit	str	issuer info
subject_common /country/state/city/org/ org_unit	str	subject info
pubkey_type	str	"RSA" or "ECC"(Elliptic Curve)
pubkey	str	public key in hex digits
pubkey_len	int	public key length in bits
subject_alt_names	[str]	subject alternative names