CESNET Idle OS Traffic: A Dataset of Idle Traffic of Various Operating Systems

Václav Bartoš[1], Michaela Novotná[1,2], and Matej Hulák[1,3]

[1]CESNET, a.l.e., Prague, Czech Republic
[2]Brno University of Technology, Brno, Czech Republic
[3]Czech Technical University in Prague, Prague, Czech Republic
Corresponding author email: bartos@cesnet.cz

***Abstract*—Machine learning methods in networking research critically depend on the availability of high-quality datasets. While several network traffic datasets exist, they typically focus on backbone traffic, intrusion scenarios, or application behavior, containing a mix of many types of traffic. In this paper, we introduce the *CESNET Idle OS Traffic* dataset. It contains traffic traces of various operating system in the "idle" state, i.e. without any user interaction, captured in a controlled lab environment. The primary intended use of the dataset is to support research of operating system fingerprinting, but it can also be used to enhance dataset for anomaly and intrusion detection, in forensic analysis, or education. We also release the toolset used to create the dataset, which enable researchers to replicate the measurement process or extend the dataset with additional operating systems.**

I. Introduction

Machine learning (ML) methods are increasingly applied in networking research for tasks such as traffic classification, anomaly detection, and system fingerprinting. The effectiveness of these methods depends heavily on the quality and availability of datasets.

A number of network traffic datasets already exist, covering areas such as backbone traffic analysis, intrusion detection, or application-specific behavior. However, one area that has received little attention is the baseline traffic produced by idle operating systems. Even without user activity, modern operating systems generate background communication with update servers, time synchronization services, certificate authorities, or other infrastructure. This "idle" traffic is characteristic of each system and provides valuable information for research in operating system fingerprinting.

In this paper, we present the *CESNET Idle OS Traffic* dataset [1], a publicly available collection of idle traffic traces from multiple operating system families, types, and versions, executed in a controlled virtualized environment. The dataset contains raw packet captures (PCAP), flow records, and structured extractions of DNS, HTTP and TLS traffic, complemented by metadata about the virtual machines and capture sessions.

This research was partially funded by the SOCCER project, Grant Agreement No. 101128073 supported by the European Cybersecurity Competence Centre.

To support reproducibility, the dataset is accompanied by an open-source toolset that automates the process of deploying virtual machines, capturing idle traffic, and extracting structured data. This toolset enables researchers to repeat the capture process, extend the dataset with new operating systems, or validate existing results.

The primary intended use of the dataset is to support the development and evaluation of OS fingerprinting techniques, where idle traffic can serve as a reliable source of distinguishing features. However, we anticipate broader applicability, including in anomaly detection research, intrusion detection training, forensic analysis, and as a didactic resource for networking and cybersecurity education.

The remainder of the paper describes the dataset structure, the capture methodology, and potential research applications.

II. Related Work

The field of operating system (OS) fingerprinting has evolved significantly over the past decades. Traditional techniques focused on passive TCP/IP stack fingerprinting, which leverage characteristics such as initial TTL, window size, and TCP flags to distinguish OS types – methods implemented in tools like p0f [2] or siphon [3].

More recent approaches embrace machine learning over network flow data, moving beyond manually crafted rules [4]–[8]. For instance, a variety of classifiers—including K-Nearest Neighbors, Decision Trees, and neural networks—have been shown to outperform conventional methods, achieving higher accuracy in OS recognition [9]. A recent survey on OS fingerprinting approaches by Laštovička et al. [6] highlights the shift toward machine learning methods that rely on traffic behavior and can adapt to changes in network conditions.

On the data side, some publicly available datasets support OS fingerprinting research [6], [10], [11]. These datasets contain large volumes of network flow data captured in university environments, annotated with OS labels. They typically include a mix of features from TCP/IP, HTTP, and TLS traffic, and are commonly used to train and evaluate OS fingerprinting methods.

While some datasets offer broad coverage and large volume, they often lack precise annotations, which limits their usefulness for supervised learning. Others are smaller and well-

978-3-903176-75-1 ©2025 IFIP

labeled but lack diversity in OS versions or traffic types. This trade-off makes it difficult to build models that generalize well.

Idle traffic, in particular, should remain consistent across environments, yet it is underrepresented in both types of datasets. It still contains useful TCP/IP features such as TTL, window size, and TCP options, which reflect the OS network stack and can support passive fingerprinting. Despite its potential, idle traffic is mostly overlooked. Existing datasets focus on active or mixed-user behavior, where user interaction dominates traffic patterns. Idle traffic instead captures the default background behavior of the operating system.

To the best of our knowledge, no public dataset specifically targets idle traffic, which is continuously present regardless of user activity. This work aims to address that gap.

III. Dataset Description

The CESNET Idle OS Traffic dataset consists of controlled captures of network traffic generated by idle virtual machines (VMs). Its primary goal is to provide a reproducible collection of baseline traffic traces that reflect the background communication patterns of different operating systems.

A. Capture Environment

The dataset was created in a virtualized environment using VirtualBox with NAT networking. Virtual machine images were sourced from platforms such as Vagrant and osboxes.org, ensuring coverage of multiple operating system families, types, and versions. Each VM was started and left idle for over 24 hours, during which all network traffic was recorded. The capture process also includes the startup phase of the operating system, when some background services initiate communication.

In total, the dataset contains captures from 39 operating systems, including various Linux distributions (e.g., Ubuntu, Debian), Windows versions, Android, or FreeBSD. The data was captured in September 2025[1].

B. Data Structure

The dataset is distributed as a compressed archive with the following folder hierarchy:

```
cesnet-idle-os-traffic.zip/
+ <os_family>__<os_type>__<os_version>/
|  + <timestamp>__<source>__<identifier>/
|  |  + flow.csv
|  |  + info.json
|  |  + traffic.pcap
|  + dns.csv
|  + http.csv
|  + tls.csv
+ merged_dns.csv
+ merged_http.csv
+ merged_tls.csv
```

where:

- **os_family**: The operating system family (e.g., `linux`, `windows`, `macos`).
- **os_type**: The specific operating system (e.g., `ubuntu`, `debian`, `windows_10`).
- **os_version**: Version of the operating system (e.g., `20.04`, `23H2`).
- **timestamp**: Date and time when the capture started.
- **source**: Origin of the VM image (e.g., `vagrant`, `osboxes.org`).
- **identifier**: Unique identifier based on the source (e.g., vagrant box name or hash).

C. Captured Data Types

a) PCAP Data (`traffic.pcap`): Raw packet captures form the foundation of the dataset. These files contain all network packets sent and received by the virtual machine during the capture period.

b) Flow Data (`flow.csv`): Flow records are derived from the PCAP files using the `ipfixprobe`[2] exporter with multiple plugins enabled. The file contains bidirectional flows enriched with application-layer information, stored in the CSV format.

c) HTTP Traffic (`http.csv`, `merged_http.csv`): HTTP request data is extracted from the raw traffic. Each record includes fields such as OS identifiers, user-agent string, hostname, and URI. The `http.csv` files correspond to individual virtual machines, while `merged_http.csv` aggregates data across the dataset.

d) TLS Traffic (`tls.csv`, `merged_tls.csv`): TLS session metadata includes protocol version, ALPN, JA3 fingerprint, and SNI values. Individual `tls.csv` files represent one virtual machine, whereas `merged_tls.csv` provides a consolidated view.

e) DNS Traffic (`dns.csv`, `merged_dns.csv`): Domain names queried by the host via DNS are extracted from the data. Similarly to the previous items, the `dns.csv` file contains domains from one virtual machine, `merged_dns.csv` combines all of them into a single file.

f) Metadata (`info.json`): Each capture folder contains a JSON file describing the VM and the capture session. Key attributes include VM name, OS family/type/version, image source, IPv4 and MAC address of the machine in, and capture start and end times.

Since no real user data are involved, the dataset does not raise any privacy concerns and does not need to be anonymized.

D. Dataset Size and Scope

The dataset comprises **39** PCAP files with corresponding flow and metadata files. In total, there are 42,906 connections (flow records) captured, with big differences in numbers of flows per operating system – from just a few generated by Kali Linux to almost 6,586 by Linux Ubuntu. The compressed archive size is approximately 4.1 GB.

[1]There is also an older version of the dataset from March 2025, where VMs run for 1 hour only. Here we only report data from the later, longer run.

[2]https://github.com/CESNET/ipfixprobe

TABLE I: Flow and Byte Statistics per VM in the first 24 hours.

VM	Connections	Incoming kB	Outgoing kB	kB total
Windows 11	2075	1406655	10208	1416863
Android 9.0	2923	377772	8571	386343
Linux Ubuntu 23.10	136	270474	1299	271772
Linux Debian 12	659	242634	1162	243796
Linux RHEL 9.3	1363	63627	492	64119
Linux Manjaro 22	1737	54410	660	55070
Linux Fedora 40	1452	16665	269	16934
FreeBSD 14.0	16	1190	23	1213
Linux Mint 21.2	1082	423	400	823
Linux openSUSE 15.4	446	106	116	221
OpenBSD 7.4	990	94	82	176
Linux CentOS 7	503	37	41	78

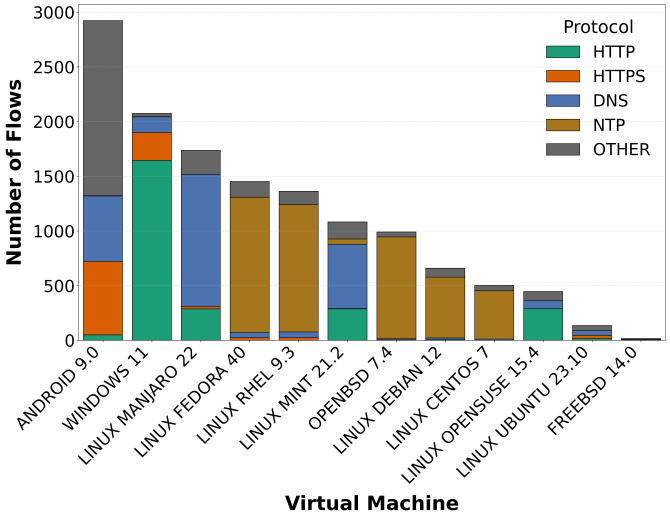

Fig. 1: Distribution of application-layer protocols across 12 selected VMs over a 24-hour period.

IV. Toolset & Replicability

The dataset was created using a dedicated toolset developed for automating creation and run of virtual machines and capture of their idle traffic. The toolset is publicly available on GitHub[3].

It consists of a set of scripts designed to automate the following tasks:

- **VM lifecycle management**: start, stop, and restart virtual machines according to a schedule.
- **Traffic capture**: record all packets at the host level using standard packet capture tools.
- **Data extraction**: process PCAP files into flow records (`flow.csv`) using `ipfixprobe`, and extract HTTP, TLS and DNS data into individual `.csv` files.
- **Metadata collection**: update or create `info.json` files containing VM configuration, operating system details, and capture session metadata.

This design ensures that the same procedure can be applied to different operating systems in a consistent manner.

[3]https://github.com/CESNET/idle-os-toolset

The availability of both the dataset and the capture toolset allows researchers to:

- Create a new version of the dataset from the same VM images. Note that the result might not be the same due to various reasons, like possible changes in behavior of the servers the VMs connect to.
- Extend the dataset by capturing additional operating system families, types, or versions.

The modularity of the toolset also enables integration into larger experimental workflows.

V. Data Insights

To provide better insight into the content of the dataset, this section presents key statistics and observations to illustrate the structure and content of the dataset. For clarity and brevity, we selected 12 operating systems (out of 39) and only report their data.

Table I lists the number of outgoing connections (IP flows) initiated by each system during the first 24 hours, along with the volume of data transferred in each direction. Figure 1 shows the distribution of application-layer protocols used. These results reveal substantial differences among operating systems in both the volume and nature of their idle traffic.

a) Communication Patterns and Destinations: A closer inspection of the traffic shows common types of background communication, such as connectivity checks (e.g., HTTP requests to `www.msftconnecttest.com`), time synchronization (e.g., NTP to `0.rhel.pool.ntp.org`), PKI updates (e.g., HTTP to `ocsp.digicert.com`), and package management system updates (e.g., HTTP to `archive.ubuntu.com`). Table II presents examples of domains contacted by selected systems based on captured DNS queries. Many of these domains are specific to particular OS families or distributions, making them useful for fingerprinting. In some cases, the exact set of contacted domains may also help identify a specific OS version.

b) User-Agent Strings: Another distinguishing feature is the User-Agent field in unencrypted HTTP traffic. Table III lists examples of User-Agent strings captured in the dataset. These strings often encode both the OS type and version, providing an additional feature for passive OS identification.

TABLE II: Domains contacted during initial boot by selected VMs.

Windows 11
- www.msftconnecttest.com
- go.microsoft.com
- ecs.office.com
- mobile.events.data.microsoft.com
- ctldl.windowsupdate.com
- ocsp.digicert.com
- login.live.com
- v10.events.data.microsoft.com
- activation-v2.sls.microsoft.com
- validation-v2.sls.microsoft.com
- ... (33 total)

Ubuntu 22.04 LTS
- api.snapcraft.io
- canonical-lgw01.cdn.snapcraftcontent.com
- canonical-bos01.cdn.snapcraftcontent.com
- dashboard.snapcraft.io
- archive.ubuntu.com
- security.ubuntu.com
- motd.ubuntu.com
- esm.ubuntu.com
- ntp.ubuntu.com

FreeBSD 14.0
- vuxml.freebsd.org
- www.ietf.org

Android 9.0
- www.google.com
- connectivitycheck.gstatic.com
- time.android.com
- android.clients.google.com
- play-fe.googleapis.com
- play.googleapis.com
- playstoregatewayadapter-pa.googleapis.com
- prod-lt-playstoregatewayadapter-pa.googleapis.com
- play-lh.googleusercontent.com
- mtalk.google.com
- www.googleapis.com
- firebaseinstallations.googleapis.com
- android.apis.google.com
- notifications-pa.googleapis.com
- clienttracing-pa.googleapis.com
- geller-pa.googleapis.com
- xgapromomanager-pa.googleapis.com
- ... (70 total)

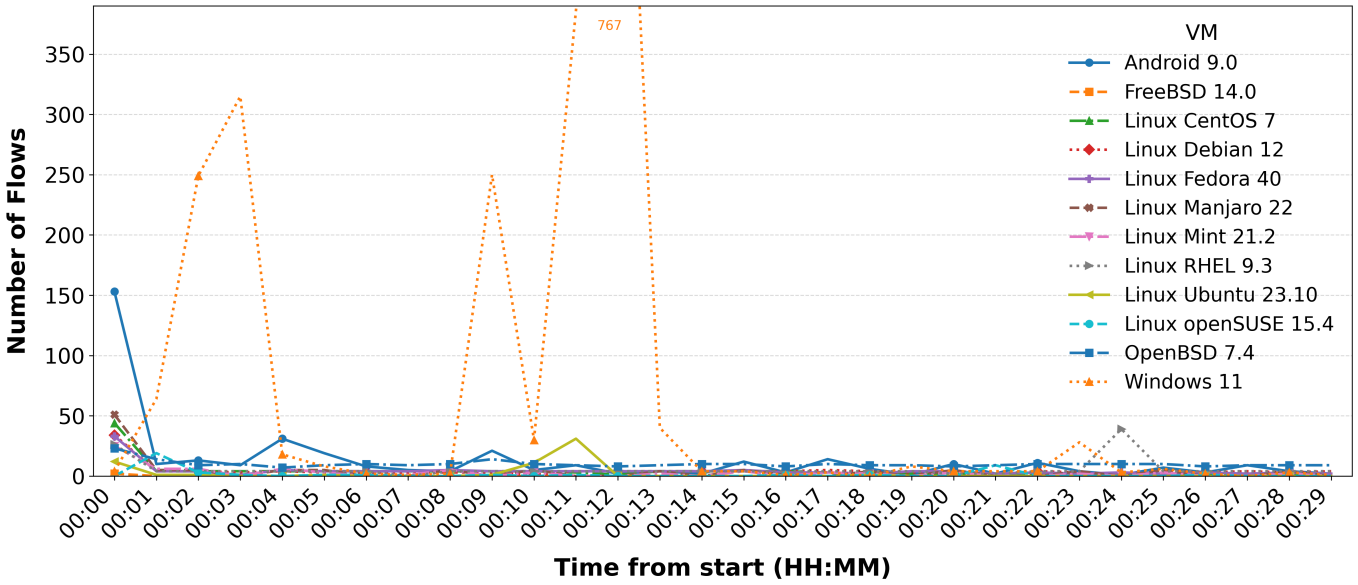

Fig. 2: Fine-grained flow activity of selected VMs during initial 30 minutes (1-minute aggregation).

c) Temporal Traffic Distribution: We also examine how idle traffic is distributed over time. Figures 2 and 3 illustrate the number of connections observed during the first 30 minutes and over a 24-hour period, respectively. Most operating systems exhibit a burst of connections shortly after boot, followed by a quieter steady state. However, the duration and intensity of these phases vary significantly. For example, Windows 11 shows traffic peaks during the 2nd–3rd minute and again at the 9th and 12th minute, with the latter likely corresponding to update-related activity. Android maintains a high connection rate for more than an hour after boot and continues to generate a relatively high number of flows even in the steady state. Conversely, OpenBSD and several Linux variants, such as CentOS 7 and openSUSE, generate significantly fewer connections overall, making them among the quietest systems in terms of idle network activity.

VI. Use Cases

The presented dataset is designed to support research in multiple areas of networking and cybersecurity. While its primary motivation was to provide a reliable source of idle traffic traces for operating system fingerprinting, the dataset also offers value for a broader range of applications. Below we outline several representative use cases.

1) Operating System Fingerprinting: Fingerprinting techniques aim to identify the operating system of a device based

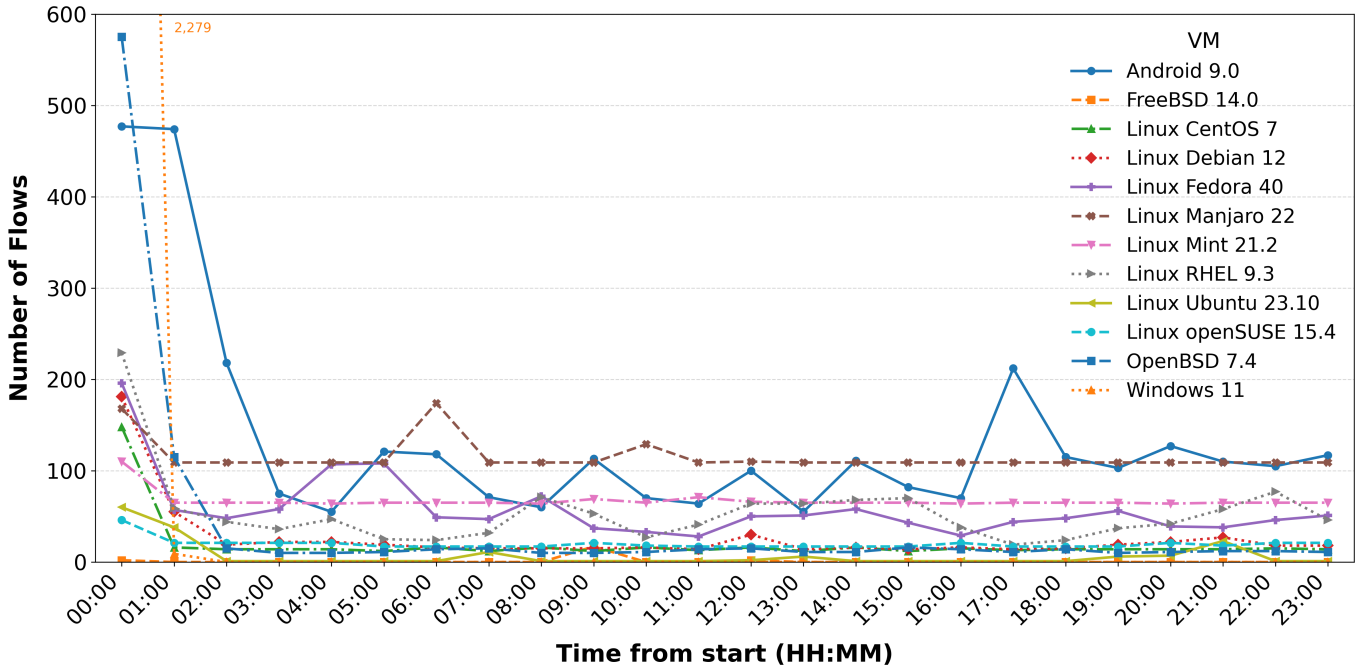

Fig. 3: Flow activity of selected VMs over the first 24 hours (1-hour aggregation).

TABLE III: Examples of HTTP User-Agents used by selected OSes.

Windows 11
`Microsoft NCSI`
`Windows-Update-Agent/1216.2402.13012.0 \`
`  Client-Protocol/2.80`
`Microsoft-Delivery-Optimization/10.1`

Ubuntu 23.10
`Debian APT-HTTP/1.3 (2.7.3ubuntu0.1) \`
`  non-interactive`

FreeBSD 14.0 `pkg/1.21.3`

on its network behavior. Idle traffic is particularly suitable for this purpose, since it is reproducible, system-specific, and not influenced by user activity. The dataset provides structured views of flows, HTTP requests, and TLS sessions, enabling the development of new machine learning–based fingerprinting methods and the benchmarking of existing ones.

2) Anomaly Detection and Intrusion Detection: A reliable baseline of "normal" background traffic is essential for detecting anomalies. Although the dataset cannot serve as such baseline by itself, since it only captures traffic generated by the operating system, not by any user-driven activity, it can still serve as an important component. By integrating the dataset into broader traffic collections that include active user behavior, researchers can create more accurate and realistic baselines for anomaly detection. It can also be used to enhance any Intrusion Detection datasets by providing samples of normal traffic.

3) Forensic Analysis: In forensic investigations, it is often necessary to separate expected background activity from suspicious communication. The dataset can serve as a reference for what constitutes normal idle behavior for different operating systems.

4) Education: Both the dataset and the open-source capture toolset can be easily used in teaching environments. Students can reproduce the captures, extend the dataset with new virtual machines, and practice analyzing network data in a controlled setting.

VII. Conclusion

The presented dataset, named *CESNET Idle OS Traffic*, is a publicly available collection of network traffic samples generated by different operating systems running without user interaction. The dataset provides raw packet captures together with L7-enriched flow data, and data extracted from HTTP and TLS headers. Its primary goal is to support operating system fingerprinting research, but we also anticipate other potential use cases.

We also release the toolset used to create the dataset. It enables researchers to recreate the dataset in a different network or with longer capture period, or to extend it with additional operating systems.

By making the dataset and toolset openly accessible, we aim to support the community in developing and benchmarking new methods for traffic analysis, classification, and security research.

REFERENCES

[1] M. Novotná and V. Bartoš, "CESNET Idle OS Traffic," Mar. 2025, [dataset], https://doi.org/10.5281/zenodo.15004765.

[2] M. Zalewski, "Passive os fingerprinting tool," 2003, http://lcamtuf.coredump.cx/p0f.shtml.

[3] S. S. Group, "Passive os fingerprinting tool," *Available at https://web.archive.org/web/20050205014947/http://www.subterrain.net/projects/siphon/*, 2000.

[4] T. Jirsík and P. Čeleda, "Identifying operating system using flow-based traffic fingerprinting," in *Advances in Communication Networking*, Y. Kermarrec, Ed. Cham: Springer International Publishing, 2014, pp. 70–73.

[5] P. Matoušek, O. Ryšavý, M. Grégr, and M. Vymlátil, "Towards identification of operating systems from the internet traffic: Ipfix monitoring with fingerprinting and clustering," in *2014 5th International Conference on Data Communication Networking (DCNET)*, 2014, pp. 1–7.

[6] M. Laštovička, M. Husák, P. Velan, T. Jirsík, and P. Čeleda, "Passive operating system fingerprinting revisited: Evaluation and current challenges," *Computer Networks*, vol. 229, p. 109782, 2023.

[7] D. H. Hagos, A. Yazidi, Ø. Kure, and P. E. Engelstad, "A machine-learning-based tool for passive os fingerprinting with tcp variant as a novel feature," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3534–3553, 2021.

[8] R. Pérez-Jove, C. R. Munteanu, A. Pazos, and J. Vázquez-Naya, "Application of tabular transformer architectures for operating system fingerprinting," *arXiv preprint arXiv:2502.09084*, 2025.

[9] J. Song, C. Cho, and Y. Won, "Analysis of operating system identification via fingerprinting and machine learning," *Comput. Electr. Eng.*, vol. 78, no. C, p. 1–10, Sep. 2019.

[10] M. Lastovicka, T. Jirsik, P. Celeda, S. Spacek, and D. Filakovsky, "Passive os fingerprinting methods in the jungle of wireless networks," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–9.

[11] M. Hulák, V. Bartoš, and T. Čejka, "Transferability of tcp/ip-based os fingerprinting models," in *2025 IFIP Networking Conference (IFIP Networking)*, 2025.