

Attack Surface Management: State of the Art and Operational Challenges

Martin Husák
Institute of Computer Science
Masaryk University
Brno, Czech Republic
husakm@ics.muni.cz

Lukáš Sadlek
Institute of Computer Science
Masaryk University
Brno, Czech Republic
sadlek@mail.muni.cz

Abstract—In this paper, we approach the topic of ASM, place the task in the context of cybersecurity operations, review the current methods, and discuss their issues and challenges. We outline an ASM pipeline consisting of common tasks and review the usability of the existing tools. We pinpoint a trade-off between the scope and precision of the existing tools that should be considered, namely in medium to large networks. Finally, we formulate and discuss the issues and challenges for emerging network environments, including those involving IoT or OT, volatile environments, or extensive use of cloud computing. Each of these emerging technologies brings novel issues that need to be approached by ASM, be it improved fingerprinting in IoT and OT, precisely timed scans in volatile environments, or coverage of external services in the cloud.

I. INTRODUCTION

Attack surface management (ASM) is a crucial part of cybersecurity operations. It consists of understanding the attack surface, i.e., knowing which hosts and services are facing the Internet or are otherwise reachable. This understanding allows cybersecurity teams to design network defenses and estimate the risks associated with successful attacks. Common tasks of ASM include enumerating running hosts and services in the computer network, fingerprinting them and finding vulnerabilities on all known assets, and enumerating subdomains or even mapping available information on public websites. A good practice is minimizing the attack surface, i.e., leaving only the well-known hosts and services to face the Internet and restricting access to others. Nevertheless, updated enumerations of cyber assets are a key to successful ASM, which is best obtained by continuous network scanning.

As with any other technique or task in cybersecurity, ASM is evolving and needs to constantly react to changes in the technologies we use or the threat landscape. For example, attacks on weak authentication (for example, on services like SSH) call for mapping all hosts with running SSH server or TCP port 22 open. However, when social engineering attacks like phishing are of major concern, it is also vital to know whether there is any contact information on the organization's personnel available online (attackers can use those to harvest email addresses for phishing campaigns) or accompanied by organization's structure (which can be leveraged to spear phishing attacks against highly-ranked personnel). Discovery and publication of a new severe vulnerability (as in cases of

Heartbleed or Shellshock) often calls for prompt response by the security team that has to find out how many assets in their network are potentially vulnerable and how threatening the vulnerability is to the organization.

In this paper, we provide two contributions. First, we outline an ASM system developed as a part of the Resilmesh system. The system design follows the best practices and procedures collected by cybersecurity teams and illustrates the state of the art in the field. As we show in the paper, ASM is not only about mapping the networks but also about doing it persistently or at a reasonable time on demand. Persistent network mapping requires solid orchestration and careful settings based on the knowledge of the local environment. The tasks should also not take too long, or the results should be updated often to achieve timeliness of the data. Second, we survey and discuss open and emerging issues of ASM. We point out the need for dedicated scanners for IoT or OT environments where common ASM tools are not sufficient. The volatile environment of virtual hosts and containerized services also calls for different approaches to their scanning or logging of parts of the attack surface at specific times. We also highlight the need for mapping not only the external attack surface (i.e., what is reachable from the public Internet) and internal attack surface (i.e., what is visible to the attacker already inside the network) but also mapping external resources and assets, such as those hosted in the cloud or outsourced to third parties. We also comment on the low accuracy of fingerprinting and the large false positive rate in vulnerability scanning, which are long-term issues of ASM. Finally, we comment on recent changes, such as the new version of CVSS scores.

The remainder of this paper is structured as follows: Section II surveys the state-of-the-art and technical background. Section III outlines the tasks and workflows of ASM. Section IV presents the design of our solution. Section V discusses the open issues. Section VI concludes the paper.

II. BACKGROUND AND RELATED WORK

Herein, we present a technical background and related works of research on ASM. The background consists of an overview of monitoring approaches used in ASM and the data sources on vulnerabilities.

A. Background

ASM must obtain data about protected assets from active network monitoring using scanners, passive network monitoring using IP flows, and from application and system logs. Active network scanners inject artificial traffic into the network to obtain responses that can be parsed to reveal information about network services, hosts, and operating systems [1]. IP flows consist of sets of packets with common properties (source IP address, destination IP address, source port, destination port, and protocol) captured during a specific time window [2]. Just like active scanners, passive monitoring may use pattern libraries to fingerprint operating systems and services [3].

As for the vulnerabilities and information on them, Common Vulnerabilities and Exposures (CVE) [4] contains publicly disclosed vulnerabilities discovered in hardware, operating systems, and applications. The National Vulnerability Database (NVD)¹ contains CVEs with related information, such as descriptions, references, and identifiers of vulnerable software and hardware products. The identifiers are expressed using Common Platform Enumeration (CPE) [5]. CPE match strings in the NVD have the following structure: `cpe:2.3:<part>:<vendor>:<product>:<version>:<update>:<edition>:<language>:<sw_edition>:<target_sw>:<target_hw>:<other>`. Names of individual parts are mostly self-explaining. The part that follows immediately after 2.3 contains an indication of whether the identifier belongs to an application, an operating system, or hardware.

Common Vulnerability Scoring System (CVSS) aims to capture characteristics of vulnerabilities, such as attack complexity, necessary privileges, and impact [6]. The NVD uses three versions of CVSS, from the second to the fourth version. Its final score is computed from values of exploitability and impact metrics.

B. Related work

Even though ASM is frequently discussed among practitioners and mentioned in research articles, there are not that many research articles focusing on this topic. As Levy and Maldonado [7] pointed out, even the terminology and metrics are not widely agreed on or used consistently. To the best of our knowledge, no survey or literature review is available. The research in this field mostly focuses on designing ASM tools or understanding the needs of ASM in specific domains. For example, Rezvi et al. [8] discuss ASM in the domain of IoT, while Ashley et al. [9] do the same for the OT domain. Wang et al. [10] or Curbo and Falco [11] discuss the issues of ASM in air and space domains. The ASM, with a focus on attack surface minimization, in the context of containerized services was the research objective of Yand et al. [12].

Otherwise, the research presents lessons learned or interesting findings in conducting the tasks, such as in the work of Gelenter et al. [13], who published a study of the external attack surface of top-100 enterprises in Europe and provided recommendations to particular sectors. They also claim that

their work is the first in several aspects. In addition to related research work, there exist several commercial ASM solutions, such as Halo Security² and Falcon Exposure Management by CrowdStrike³.

This work leverages graph-based structuring and interconnection of heterogeneous data on computer network implemented in the CRUSOE toolset [14]. Mahbub et al. [15] further explored the graph representation of relationships between vulnerabilities and vendors, which finds interesting application is ASM.

III. ASM WORKFLOWS AND TASKS

ASM consists of many tasks and workflows with numerous options and a plethora of tools available. An overview can be seen in Figure 1, which is further discussed in this section.

A. Scope and Existing Data

The first step in ASM is defining the scope, just like when a CSIRT team defines its constituency. The scope may cover, for example, an IP address range or domain. The scope will most likely include asset discovery and subdomain enumeration. Other frequent options are the enumeration of assets relevant to social engineering attacks (e.g., data on personnel or organization structure) and identification of critical infrastructure and its dependencies.

Any existing database used in the organization is an interesting resource and should be used for building the initial enumeration or setting up further plans. Tools like asset inventories and asset management systems, configuration databases (CMDB), and user management systems are all vital sources of information to export and use for ASM.

B. Network Scanning and Fingerprinting

The first active task of ASM is typically network scanning. Even if the cyber assets are well documented in the asset inventory, it is still important to scan for all assets, namely to discover forgotten or even rogue devices and undocumented services (open network ports).

Typical tools for active network scanning include Nmap, OWASP Amass, MASSCAN, and other tools. The key features to watch when choosing the most suitable one are the speed and configurability of scanning. Some scanners can be very fast due to transmitting network packets and reducing the time required for scanning even large networks (or even the whole Internet). However, it is vital to also allow for traffic limiting to avoid congestion, especially in low throughput parts of the network. It is also important to consider where to run the scan from – running it from a machine outside the protected network will map only the external attack surface, while running it within the network will add the notion of an internal attack surface. Still, it is worth comparing the internal and external surfaces and keeping the notion of which assets appear in them. There might also be a need to run a scan from multiple workers located within various private networks. Nevertheless, running an active scan is a simple task.

¹<https://nvd.nist.gov/>

²<https://www.halosecurity.com/>

³<https://www.crowdstrike.com/platform/exposure-management/easm/>

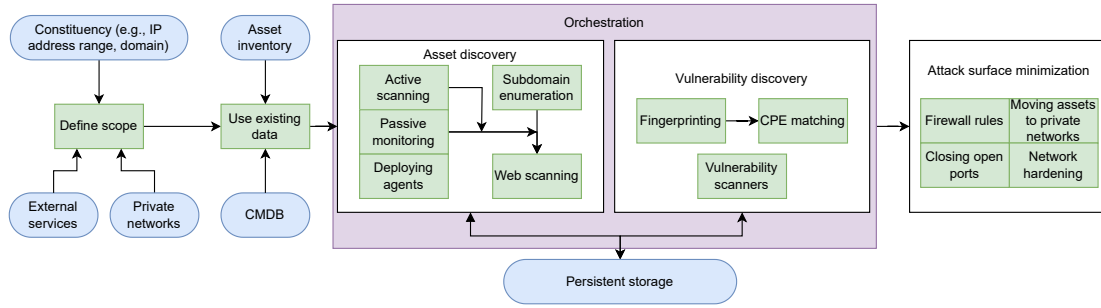


Fig. 1. The workflow of ASM, from defining scope of ASM to accomplishing defense measures based on asset and vulnerability discovery.

A viable alternative to active scanning is network traffic monitoring, such as using NetFlow or IPFIX [2]. In such a case, network traffic is observed by dedicated probes or routers, and each network connection is logged. Setting up network monitoring infrastructure is more complex, especially in large networks, but continuous monitoring offers several benefits. First, any active network device will sooner or later communicate over the network and thus be discovered by the monitoring. This includes even hosts and services that would block network scans or be placed behind a firewall.

Regardless of whether the assets were discovered by active scanning or passive traffic measurement, they can be fingerprinted [16]. Hosts' operating systems can be recognized by characteristic features in the TCP/IP headers (e.g., default TTL or Window size) or passively by extracting User-Agent fields from HTTP traffic. Techniques of banner grabbing allow recognizing software (often including at least its major version, sometimes even minor version) behind network services in active scanning. Passive monitoring can use service fingerprint libraries like Cisco's NBAR2.

An important standard in this field is the CPE [5]. CPE allows for formally identifying a certain piece of software, including its vendor, major and minor versions, edition, language, and other features. CPE strings are used in both asset management and vulnerability databases and thus allow for looking up all vulnerabilities that may be present on a host or service identified by a certain CPE. However, it is difficult to retrieve a complete fingerprint, so CPE-CVE matching is only vague and rather informative, so additional vulnerability discovery and confirmation procedures should be taken [16].

Finally, it is worth noting that there are specialized scanners for web services that allow for fingerprinting Content Management Systems (CMS) and other building blocks of the web. Since HTTP(S) is a dominant application-layer protocol and a major part of the attack surface, it is highly recommended to be included in web scanning. Since the differences from network scanners mentioned earlier, this task is preceded by a subdomain enumeration task – the goal is to enumerate all the subdomains of a given domain, namely to scan them with web scanners.

C. Vulnerability Assessment

Matching CPEs is often only a rough estimate that the vulnerability may be present on a cyber asset, so a more elaborate approach is required [16]. A viable option is to use dedicated vulnerability scanners. Tools like OpenVAS or Nessus allow users to discover vulnerabilities in the network much better than by linking CPEs to CVEs. Unlike network scanners, these tools primarily aim to discover vulnerabilities and enable more interaction with the targets. They are also capable of discovering weaknesses, such as weakly authenticated services. However, dedicated vulnerability scanners are complex and difficult to maintain, thus mostly provided as commercial pieces of software. With the few open-source or free exceptions like OpenVAS, the pricing of vulnerability scanners can reach thousands of dollars, often depending on the number of scanned hosts, which may deter many potential users, namely those managing large networks.

A highly interesting option is using the Nuclei tool⁴. Nuclei is a modern vulnerability scanner designed to use templates – each template describes the procedures to discover a particular vulnerability. The templates are available in a curated, community-driven, and frequently updated library. Thus, Nuclei allows the users to check for the presence of even recently discovered vulnerabilities.

The templates also allow for defining elaborated or more fitting workflows than vulnerability scanners. For example, scanning a large number of hosts and services takes a significant amount of time, which could be optimized by collaborating with asset discovery tools. In such a case, the network scanners will periodically enumerate active hosts and services on a weekly basis, for example. Simultaneously, all Nuclei templates from the past week are downloaded. Known assets, i.e., hosts and services discovered in previous weeks, are only scanned using the new templates, while newly found assets are scanned using all the templates. Thus, integrating multiple tools and providing persistent storage can significantly reduce the time required for vulnerability scans.

D. Data Storage

As outlined earlier, storing the data for later use can be advantageous – keeping the record of past scans allows for

⁴<https://docs.projectdiscovery.io/tools/nuclei/overview>

checking the differences and optimizing the tasks. Checking a frequently updated database of asset and vulnerability scan results can also be faster and more convenient than running scans on demand.

An emerging trend is defining ontologies and building the knowledge graphs. Ontology defines the entities and relationships in a particular domain, such as attack surface or cybersecurity in general. The knowledge graph can be seen as an instance of the ontology in the form of a graph or stored in a graph database. Tools like CyGraph or CRUSOE pioneered the use of graph databases to store information about computer networks and their security posture, namely due to their ability to extend the data models on the fly and interlink heterogeneous data from various sources. In the near future, we are likely to see the use of Retrieval-Augmented Generation (RAG), which allows for building the knowledge graph from unstructured data and prompting the data using Large Language Models (LLMs).

E. Toolsets

Even though there are many tools and toolsets to conduct ASM, there is no one-size-fits-all solution. Certain aspects, like orchestration and persistent data storage, are not usually implemented. For example, various popular bundles like bbot, Sn1per, or Viper allow mapping the attack surface. The main advantage of such tools is that they can be executed by a single command. The toolsets then run network scanning, subdomain enumeration, vulnerability scanning, and other tasks autonomously. All such tools have vast capabilities of exporting the results or generating reports but no persistent storage or orchestration capabilities. It is also worth noting that such toolsets follow the trends in the field and often use the same tools for particular tasks (e.g., subfinder and amass), so their results are often very similar.

IV. SAMPLE ASM IMPLEMENTATION IN RESILMESH

In order to fill the gap, we designed an ASM system as a part of the Resilmesh project and framework⁵, specifically its *Situation Assessment* plane [17]. It is also based on the components of an earlier system for cyber situational awareness called CRUSOE [14], namely its Observe and Orient parts. Our aim is not to improve the particular tasks but rather to define a complex workflow covering all the aspects of ASM and integrating it with other cybersecurity tools and processes. Individual components can be run separately, but we would like to illustrate orchestration and collaboration.

The proposed ASM system contains three components – Infrastructure and Service Information Model (ISIM), Cyber Attack Surface Management (CASM), and Situation and Network Awareness Consolidated Dashboard (SACD). ISIM is a central database that holds information about the attack surface and provides various APIs for fetching or updating the data. An important parts of ISIM are the definition of the data model and the constraints and clean-up routines that keep

the data in correct format and actual. CASM consists of an orchestration engine that runs attack surface scanning tools and stores the results into ISIM. SACD is a dashboard that visualizes the data stored in ISIM. All components can be deployed in docker containers, which facilitates their use and maintenance without any additional prerequisites.

A. Data collection and its orchestration

There is a plethora of tools for conducting ASM, including network scanners, passive monitoring tools, and agents. In our case, we chose the active network scanning approach facilitated by using EasyEASM⁶. EasyEASM is a bundle of several tools that disclose the external attack surface together. EasyEASM first enumerates the subdomains of a given domain using the subfinder tool, then scans all the enumerated hosts with amass and provides additional tasks like fingerprinting web services via Wappalyzer. The *fast* mode of EasyEASM's scanning was sufficient for our implementation of ASM. The advantage of EasyEASM is that everything can be run in a single command, but it might be advantageous to rather use all its components separately, especially in larger networks, or to achieve finer granularity of tasks in orchestration.

EasyEASM obtains the list of IP addresses, open ports, domain names, and underlying software versions, all with timestamps, which are ready to be stored in ISIM after a serialization into a JSON file. In addition, we prepared a script that automatically converts the identified names of software and its versions into CPE strings, wherever possible. Apart from EasyEASM, we implemented a CVE connector, a tool that connects to the NVD's REST API and downloads the information on recent CVEs. CPE strings from the CVE details are then matched against the CPE strings already present in the database, and potentially vulnerable hosts and services are linked to vulnerabilities.

CASM is orchestrated via Temporal⁷. For this purpose, we prepared Temporal's workflow and activities that validate scan parameters on the input, execute the scanner, and store data in the Neo4j database. Inputs for the scan consist of a list of domain names and the mode of scanning. There was a need to implement a Temporal worker, which is, however, a rather simple wrapper that straightforwardly calls EasyEASM to conduct the scan.

However, it is required to implement storing the scan results into Neo4j. Domain names, IP addresses, open ports, and network services are stored in the databases and interconnected by relationships. Some types of relationships use start timestamps captured when a new relationship is created. They are also used to transitively track old IP addresses and domain names because relevant relationships have their end timestamps set. Each IP address and domain name receives an end timestamp at the end of the Temporal task when it was discovered by network scanning.

From the user's perspective, a workflow can be started using a prepared client from the command line. Schedules

⁵<https://resilmesh.eu/>

⁶<https://github.com/g0ldencybersec/EasyEASM>

⁷<https://temporal.io/>

and individual workflows can be created and viewed using a web user interface. The prepared workflow class serves as a template for creating a scanning workflow.

B. Data storage and visualization

ISIM is using the Neo4j graph database to store data on the attack surface. The use of a graph database is advantageous since organizing network data as graphs allows focusing on relationships of network entities, which positively influences performance. The database is organized according to the data model based on CRUSOE [14] and further enhanced and updated to fit the needs of Resilmesh. Since ISIM and Resilmesh have a larger scope than just ASM, we do not provide a deeper explanation. For the ASM, the crucial entities and relationships are domain names translated to IP addresses, network services hosted on IP addresses, software versions detected on IP addresses (e.g., OS version) or network services (e.g., HTTP server version), vulnerabilities identified by CVE associated to CPE string describing software versions, and IP addresses belonging to subnets (IP address ranges) and linked to contacts on administrators or users.

ISIM does not allow the user to interact with the database content directly but offers two APIs that provide data quality and consistency checks when modifying the data. The REST API primarily consumes structured data containing information on network assets, including hosts, devices, applications, software versions, open ports, protocols, IP addresses, domain names, URIs, subnets, contacts, and organization units. Their relationships expressed which entities should be connected in the Neo4j database, such as devices and their applications. Scanning tools typically produce their results in a JSON file, which is then processed by the REST API. The GraphQL API primarily serves the dashboard and other data consumers. The graph-based API allows us to keep the graph structure of data for visualization or further processing, which is highly advantageous due to the natural comprehensibility of graph-based data in visual form.

The dashboard (SACD) visualizes the content of the ISIM database in a user-friendly and intuitive manner without any additional training. The dashboard is composed of several panels, each implementing a different view of the data. For example, one panel allows for listing all the contextual information on a given asset, while another panel visualizes how many hosts are vulnerable to a given CVE per subnet. These panels are taken from the CRUSOE Orient dashboard [14]. A novel content consists of interactive panels that allow the user to list assets, filter newly found ones, and manually annotate them (e.g., if a newly found asset is known and benign or suspicious).

V. ISSUES AND CHALLENGES

Herein, we outline the issues of ASM that we identified. If possible, we formulate the research or development gap and outline the approaches for future work.

A. Scope and heterogeneity of data

The first issue is rooted in defining the scope of ASM. Organizations are more and more reliant on cloud services, thus not extending the potential attack surface to cloud services hosted by third parties. Network visibility is one of the issues related to the quality of obtainable data. Each data source has some disadvantages. Manual input cannot be applied on a large scale. Passive network monitoring does not reveal hosts that do not communicate. Network scanning ignores hosts that do not respond or are not up. Application and system logs can have various formats and need to be obtained by installed agents. Another category of assets with low visibility includes hosts in private networks. While these are not part of the external attack surface, we are seeing an increasing number of attacks using lateral movement and also having to map the internal attack surface. Nevertheless, this can be approached by using multiple workers (i.e., scanning hosts located at different locations, such as inside or outside the network, in the private network).

A possible solution to the disadvantages of data sources is to combine them. Security platforms using ELK (Elasticsearch, Kibana, and Logstash)⁸ can utilize many modules provided by ELK and gathering cloud logs, containers' logs, operating system logs, and application logs (e.g., web servers). ELK can also ingest IP flow data in the form of NetFlow and IPFIX records, which are two protocols for transmitting flow data. ELK can also store custom data, such as the results of network scans. Further research is needed to cope with the amount of network traffic and logs, synchronization of data from various sources, and encryption of network traffic [18].

A vital practical issue arose with vulnerabilities like Log4Shell. In this particular case, the vulnerable component was a Java library Log4j used by software providing network services. Even though we could fingerprint the network services and the software behind them, we had to check whether the piece of software uses Log4j or not and in what version – we had to either check the software in question or experimentally exploit it. Thus, if a vulnerability exploitable over the network appears in the library (or elsewhere in the supply chain), the ASM procedures have only limited capabilities for finding vulnerable hosts.

B. Volatile environments and emerging technologies

Dynamic environments represent a limitation for network scanning. When hosts and network services are up only for a short time (e.g., several hours), the scanning will not perform optimally. It also holds for continuous scanning, which executes another scan immediately after the previous one ends to approximate complete network visibility. Other methods, such as passive network monitoring, should be considered an alternative to scanning.

The emergence of IoT devices calls for designing novel techniques for their discovery and fingerprinting. Most of the existing scanners or monitoring tools are designed for generic IT networks and can recognize operating systems and

⁸<https://www.elastic.co/elastic-stack/>

services on common workstations, servers, or mobile devices. However, simple IoT devices often fail to be fingerprinted properly – we can fingerprint their OS (e.g., a variant of Linux) but not much else. We need to differentiate between various device types (e.g., CCTV cameras, various sensors, smart TVs, and other appliances), which require either more elaborated device signatures for scanning or passive monitoring for a longer period of time. A very similar situation is with operational technology (OT) and cyber-physical systems (CPS). A related issue is the need to consider different communication protocols used by IoT or OT as such devices do not necessarily use TCP/IP protocols, especially when communicating over radio, which calls for designing additional tools and procedures. Nevertheless, both issues are the subject of excessive research, and even though we do not have many ready-to-use tools at hand, current research suggests they are ready to hit the market.

C. Vulnerability information

CVE currently contains more than 270,000 vulnerabilities in total [4]. This count of vulnerabilities can represent computational overhead for software that attempts to use all of them. Using only a specific time range, such as the last years, may not be the best for applying in practice because the experience reveals that even old vulnerabilities can remain unpatched. When having a limited asset inventory, a better option is to filter vulnerabilities according to their CPEs. Each software and hardware asset has only a limited count of vulnerabilities.

One of the challenges is the adoption of CVSS v4.0 in practice, namely due to the increased number of metrics compared to previous versions. However, when using only base metrics, such as in the NVD, their use is simple. CVSS v4.0 has two times more impact metrics to express that a vulnerability can affect not only a vulnerable system but also other systems. Searching NVD on March 21, 2025, returned only 76 vulnerabilities that have some CVSS v4.0 in comparison with 171,685 vulnerabilities with CVSS v3.x.

The NVD plans to use its new REST API instead of data feeds. In the past, it was necessary to download large JSON files containing vulnerabilities for a specific year. The REST API allows the downloading of only relevant vulnerabilities for individual CPE names, time ranges, CVSS, and other properties.

VI. CONCLUSION

In this paper, we discussed the issues of ASM, which is routinely done in practice but not often discussed in research. Even though there is a plethora of tools available for each task in ASM and even toolsets enabling one to run it with a single command, there are unaddressed issues like orchestration, persistent data storage, or scalability for large networks. Therefore, we deconstructed the tasks and workflows in ASM and discussed the research and development gaps. Subsequently, we designed the ASM toolset as a part of the Resilmesh system that approaches emerging needs and allows integration with other cybersecurity tasks and processes. Finally, we commented on the open issues and outlined the expected trends in future work.

ACKNOWLEDGMENT

This project has been funded by the European Union as part of the Horizon Europe Framework Program (HORIZON), under the grant agreement 101119681, Resilmesh.

REFERENCES

- [1] E. Bou-Harb, M. Debbabi, and C. Assi, “Cyber scanning: A comprehensive survey,” *IEEE Communications Surveys and Tutorials*, vol. 16, pp. 1496–1519, 2014.
- [2] R. Hofstede, P. Čeleda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, “Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.
- [3] 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.
- [4] The MITRE Corporation, “CVE: Common Vulnerabilities and Exposures (CVE),” 1999/2025, accessed: Mar 12, 2025. [Online]. Available: <https://www.cve.org/>
- [5] B. A. Cheikes, D. Waltermire, and K. Scarfone, “Common platform enumeration: Naming specification version 2.3,” National Institute of Standards and Technology, Tech. Rep., 2011. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir7695.pdf>
- [6] FIRST – Forum of Incident Response and Security Teams, “Common Vulnerability Scoring System version 4.0: Specification Document,” 2023, accessed: Mar 19, 2025. [Online]. Available: <https://www.first.org/cvss/v4.0/specification-document>
- [7] M. Levy and F. Maldonado, “Attack surface measurement: A weird machines perspective,” in *Proceedings of the 2024 European Interdisciplinary Cybersecurity Conference*, ser. EICC ’24. New York, NY, USA: Association for Computing Machinery, 2024, p. 90–94.
- [8] S. Rizvi, R. Orr, A. Cox, P. Ashokkumar, and M. R. Rizvi, “Identifying the attack surface for iot network,” *Internet of Things*, vol. 9, p. 100162, 2020.
- [9] T. Ashley, S. N. G. Gouriseti, N. Brown, and C. Bonebrake, “Aggregate attack surface management for network discovery of operational technology,” *Computers & Security*, vol. 123, p. 102939, 2022.
- [10] H. Wang, X. Heng, and W. Li, “Research on attack surface management technologies and telecom operators’ strategies for space-air-ground networks,” in *2024 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, 2024, pp. 541–546.
- [11] J. Curbo and G. Falco, “Attack surface analysis for spacecraft flight software,” in *2024 IEEE 10th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, 2024, pp. 22–30.
- [12] S. Yang, B. B. Kang, and J. Nam, “Optimus: association-based dynamic system call filtering for container attack surface reduction,” *Journal of Cloud Computing*, vol. 13, no. 1, p. 71, 2024.
- [13] N. Gelernter, H. Schulmann, and M. Waidner, “External attack-surface of modern organizations,” in *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS ’24. New York, NY, USA: Association for Computing Machinery, 2024, p. 589–604.
- [14] M. Husák, L. Sadlek, S. Špaček, M. Laštovička, M. Javorník, and J. Komárková, “CRUSOE: A toolset for cyber situational awareness and decision support in incident handling,” *Computers & Security*, vol. 115, p. 102609, 2022.
- [15] M. Mahbub, M. S. A. Khan, T. Hamid, and M. S. Mia, “A novel vulnerability exploit prediction system using the relational vulnerability-vendor network,” *Digital Threats*, Mar. 2025, Just Accepted.
- [16] M. Laštovička, M. Husák, and L. Sadlek, “Network Monitoring and Enumerating Vulnerabilities in Large Heterogeneous Networks,” in *IEEE/IFIP Network Operations and Management Symposium*, 2020.
- [17] J. B. Bernabé, B. Lee, M. Husák, L. Sadlek, B. Stojanović, M. Somma, J. I. de Oliveira, E. P. Nieto, P. F. Saura, A. Skarmeta, and V. H. La, “The Resilmesh Architecture: Situation Aware Enabled Cyber Resilience for Dispersed, Heterogenous Cyber Systems,” in *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*, 2025.
- [18] S. Špaček, P. Velan, P. Čeleda, and D. Tovarňák, “HTTPS Event-Flow Correlation: Improving Situational Awareness in Encrypted Web Traffic,” in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, 2022.