Identification of Attack Paths Using Kill Chain and Attack Graphs

Lukáš Sadlek*†, Pavel Čeleda*†, Daniel Tovarňák*
†Faculty of Informatics, Masaryk University, Brno, Czech Republic
*Institute of Computer Science, Masaryk University, Brno, Czech Republic
sadlek@fi.muni.cz, celeda@ics.muni.cz, tovarnak@ics.muni.cz

***Abstract*—The ever-evolving capabilities of cyber attackers force security administrators to focus on the early identification of emerging threats. Targeted cyber attacks usually consist of several phases, from initial reconnaissance of the network environment to final impact on objectives. This paper investigates the identification of multi-step cyber threat scenarios using kill chain and attack graphs. Kill chain and attack graphs are threat modeling concepts that enable determining weak security defense points. We propose a novel kill chain attack graph that merges kill chain and attack graphs together. This approach determines possible chains of attacker's actions and their materialization within the protected network. The graph generation uses a categorization of threats according to violated security properties. The graph allows determining the kill chain phase the administrator should focus on and applicable countermeasures to mitigate possible cyber threats. We implemented the proposed approach for a predefined range of cyber threats, especially vulnerability exploitation and network threats. The approach was validated on a real-world use case. Publicly available implementation contains a proof-of-concept kill chain attack graph generator.**

***Keywords*—kill chain, attack graph, threat identification, CVE, MITRE ATT&CK, STRIDE**

I. Introduction

The estimated loss from cybercrime in 2020 amounts to 945 billion dollars representing approximately 1% of global gross domestic product (GDP) [1]. Therefore, we focus on timely identification of relevant threats before the attackers exploit defense weaknesses. The identification of cyber threats is related to items of business value for the organization's services called assets, e.g., *information*, *people*, *technologies*, and *facilities* [2]. The weakness is any bug or flaw in the functionality of used technologies, and vulnerability is its specific instance. Publicly known vulnerabilities, general weakness categories, and types of threats can be obtained from enumerations and knowledge bases. On the other hand, internal information about assets is provided by the organization's asset management.

Multi-step attacks can be modeled using the military kill chain concept. A well-known example is the *cyber kill chain* (also known as *intrusion kill chain*) which consists of ordered phases describing the attacker's progress in achieving actions on objectives [3]. Another threat model are *attack graphs* that depict all possible chains of attacker's actions and their concrete materialization within the organization's network.

978-1-6654-0601-7/22/$31.00

Security issues revealed from alerts or vulnerability scanning correspond to only one step from a sequence of attacker's actions. Simple defense mechanisms (e.g., patch management) may not eliminate all network attack paths. Custom rules are often used for creating these sequences, and the attack paths do not represent possible attacks modeled by the kill chain.

Therefore, our research goal is to identify sequences of adversarial actions that form multi-step threat scenarios which can become attacks. We deal with the following research question:

> *Can we merge kill chains and attack graphs to determine targeted cyber threats that jeopardize protected infrastructure and defense against them?*

Our contribution can be summarized as follows. We proposed the kill chain attack graph as a new data structure that merges attack graphs and kill chain concepts. We prepared a ruleset for creating attack graphs based on a standardized knowledge base of attack techniques. We proposed a method for chaining these attack techniques into attack paths using violated security properties on assets. We implemented the approach and validated it on a real-world phishing use case.

The paper is organized as follows. Section II discusses the current state of the art related to cyber threat identification. Section III defines the new kill chain attack graph. Section IV describes the implementation of attack graph generation. Section V provides a validation of the approach. Section VI concludes the paper.

II. Related Work

The related work consists of two parts focused on cyber threat identification. These parts are related to the kill chain and the attack graphs.

A. Kill Chains

The cyber kill chain models attacks as sequences of steps. It assumes that the attacker selects suitable targets first, prepares necessary deliverables, and transmits them to the environment. Then, the deliverables enable to exploit system or application vulnerabilities, install backdoor or trojan, establish a command-and-control channel, and finally act on objectives [4].

However, it was often criticized for its limited usability, e.g., focus on malware and insufficient support for phishing attacks [5]. Therefore, Paul Pols introduced the unified kill

chain consisting of 18 tactics, i.e., phases [5]. The tactics are related to establishing a foothold, network propagation, and actions on objectives. The concept uses confidentiality, integrity, and availability as compromised security properties.

MITRE ATT&CK [6] is a knowledge base for attack modeling based on kill chain. ATT&CK is a matrix consisting of columns and rows. Its tactics (columns) can be viewed as phases of a multi-step attack and techniques (rows) as particular attack steps. Some techniques may not appear in a single multi-step attack and the same order. A recent effort called MITRE D3FEND specifies countermeasures [7] that can be applied for some ATT&CK techniques.

B. Attack Graphs

Attack graphs were introduced by Phillips and Swiler in 1998 [8]. These graphs depict the attacker's paths through the network. The attacker usually accomplishes a sequence of attack steps where each attack step leads to some privileges on protected assets. The popularity of attack graphs manifests in their joint application with other cybersecurity concepts, e.g., Bayesian networks [9].

Research papers usually do not have any standardized attack steps. Exceptions are papers that utilize some well-known enumerations and knowledge bases. A threat modeling language that utilized ATT&CK was proposed in [10]. ATT&CK and CVE (Common Vulnerabilities and Exposures [11]) were utilized in [12]. Aksu et al. used CVE vulnerabilities from the NVD (National Vulnerability Database) [13]. ATT&CK is usually applied as a detailed taxonomy of the attacker's actions and not as a kill chain. To the best of our knowledge, attack graphs cannot represent sequences of attacker's actions mapped to kill chain phases [14].

Generators of attack graphs were already designed and implemented. A well-known attack graph generator is called MulVAL. It takes as input data advisories, host configuration, network configuration, information about users, the interaction of components, and a policy [15]. However, Kaynar pointed out that the generators can create unrealistic attack paths, e.g., by considering reachability only among hosts and simplifying relationships among applications [14].

III. Kill Chain Attack Graph

Herein, we introduce a new *Kill Chain Attack Graph (KCAG)* and a description of adjustments for the attack graph generation process. *KCAG* is a specific type of attack graph that combines the kill chain and the attack graph concepts. It allows representing chains of attacker's actions divided into kill chain phases.

A. Definition of the KCAG

KCAG is an ordered triple (G, P, f) where $G = (V, E)$ denotes a directed graph with vertices V and edges E. A set P contains kill chain phases, and a function f assigns kill chain phases to attack techniques.

Vertices are of five types:

1) attacker's level of control over an asset,
2) property of an asset,
3) countermeasure,
4) attack technique,
5) attack goal.

The first type of vertices expresses the attacker's level of control over an asset. We focus on four categories of assets – processes, people, technologies, and data. The process is a sequence of actions accomplished by *actors* who utilize technologies to work with data. The process is not accomplished when the security requirements of its related assets are violated.

Each person owns a collection of accounts and *technologies*, has allowed *actions*, and works with data. Technologies are secondary assets that support processes, e.g., software installed on a network host. Further, we deal with *data at rest* and *data in transit*.

The attacker can have three levels of control over assets. Level zero corresponds to the state when the attacker does not know about the asset's existence. At the first level, the attacker knows about the asset's existence but does not have any rights or cannot violate asset's security properties. At the highest level, the attacker can violate the asset's security requirements.

The asset type determines potential security properties. Data requires confidentiality, integrity, and availability. Actors are related to authentication and non-repudiation. On the other hand, actions and secondary assets require all security properties from the STRIDE threat model (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege). These properties are authentication, integrity, non-repudiation, confidentiality, availability, and authorization.

Asset type called *external actor* is always a leaf representing the attacker's default position (level number zero). Final attack paths contain levels of the attacker's control. Therefore, a list of assets that the attacker accessed or compromised during an attack can be obtained from the attack path vertices. The attacker can also gain control related to the second level from level zero, i.e., skip the first level.

Properties of assets are the second type of vertices. They contain information about network services, vulnerable applications, and user accounts. *Countermeasures* are anti-prerequisites for attack techniques. For example, employing a strong password policy hinders the brute force technique. Properties of assets and countermeasures are always leaves.

Attack techniques are vertices with incoming edges from prerequisites – previous asset control levels, asset properties, and not applied countermeasures. Each attack technique violates some security properties imposed on the assets (see Table I). The result of the applied attack technique is a different level of control over another asset, a higher control level over the same asset, or a final attack goal. Only some combinations of input and output asset types are allowed. These combinations are depicted in Table II. Allowed combinations are denoted by checkmarks and the forbidden by dashes.

An external actor can only take an action that allows him to interact with the protected network infrastructure. No

TABLE I
SELECTED ATT&CK TECHNIQUES MAPPED TO TACTICS AND VIOLATED SECURITY PROPERTIES

Tactic (Kill Chain Phase)	ATT&CK ID	Technique Name	Violated Property
Initial Access	T1190	Exploit Public-Facing Application	Authorization
Initial Access	T1133	External Remote Services	Authentication
Initial Access	T1566.002	Spearphishing Link	Authentication
Initial Access, Privilege Escalation	T1078.001	Default accounts	Authentication
Execution	T1059.008	Network Device Command Line Interpreters	Authorization
Execution	T1204.001	User Execution - Malicious link	Authentication
Execution	T1203	Exploitation for Client Execution	Authorization
Privilege Escalation	T1068	Exploitation for Privilege Escalation	Authorization
Credential Access	T1110	Brute Force	Authentication
Discovery	T1046	Network Service Scanning	Authentication
Discovery	T1018	Remote System Discovery	Authentication
Lateral Movement	T1021	Remote Services	Authentication
Lateral Movement	T1210	Exploitation of Remote Services	Authorization
Collection	T1005	Data from Local System	Confidentiality
Impact	T1499.004	Endpoint DoS – Application or System Exploitation	Availability
Impact	T1498	Network Denial of Service	Availability
Impact	T1489	Service Stop	Availability
Impact	T1486	Data Encrypted for Impact	Integrity, Availability
Impact	T1565.001	Data Manipulation - Stored Data Manipulation	Integrity, Availability
Impact	T1485	Data Destruction	Integrity, Availability

attack step leads to the default position again. The attacker cannot gain control over an actor from another actor without processing some data via actions or by secondary assets. Besides, we do not allow the attacker to gain control over data immediately after gaining control over another data. The rules for asset types were inspired by the data flow diagram used for the STRIDE threat modeling [16].

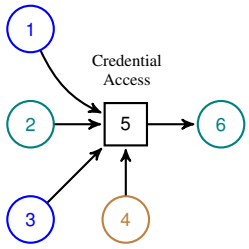
Each technique is mapped to the kill chain phase. For this purpose, we use set P and function f from the *KCAG* definition. $P = \{1, ..., n\}$ is a set of kill chain phases $1, ..., n$, which are called tactics in MITRE ATT&CK. Function f maps set of attack techniques T to their kill chain phases, i.e., $f : T \rightarrow 2^P - \emptyset$. It holds that each technique can be mapped to one or more phases.

Attack goals are vertices representing mission-critical assets as the attacker's objectives. These vertices have only incoming and no outgoing edges in the output graph. They can be expressed using violated security properties of assets on a specified level of control over assets.

Figure 1 contains an attack graph excerpt for the brute force technique (black vertex) belonging to the credential access kill chain phase. Green vertices are levels of asset control, and the blue vertices are asset properties. The technique requires

TABLE II
ALLOWED INPUT AND OUTPUT ASSET TYPES FOR ATTACK TECHNIQUES

Input \ Output	External Actor	Actor	Secondary Asset	Action	Data
Ext. Actor	–	–	–	✓	–
Actor	–	–	✓	✓	–
Sec. Asset	–	✓	✓	✓	✓
Action	–	✓	✓	✓	✓
Data	–	–	✓	✓	–



ID	Description
1	A user account on SSH service running on the server.
2	Violated authentication of SSH network connection to the server.
3	SSH service on the server accessible on TCP port 22.
4	Organization does not use a strong password policy.
5	T1110 - Brute Force.
6	Violated authentication of SSH service user account on the server.

Fig. 1. The brute force technique against SSH network service credentials.

previously violated authentication of a network connection, i.e., scanned network services. In this example, a user account exists on the SSH service accessible on the TCP port 22. The brown vertex represents that a countermeasure (a strong password policy) was not applied. Last, the authentication of the SSH user account is violated as a result.

B. Benefits of the Proposed Approach

To the best of our knowledge, attack graphs cannot represent sequences of attacker's actions mapped to kill chain phases [14]. On the other hand, the kill chains do not capture all possible sequences of attack steps, such as attack graphs. Minor fixes to the current state (such as changed input files) will not help. Therefore, the joint application of these concepts is incorporated directly into *KCAG*.

It is complicated to determine the right level of detail for actions in attack graphs. Our KCAG can be realistic enough

because the kill chain enforces proper modeling of particular attack steps. In addition, we use a standardized knowledge base of the attacker's actions (MITRE ATT&CK).

Kill chain models suggest eliminating strategic places in sequences of steps [5]. However, they do not provide theoretical apparatus for such a task. Discussion about defense measures for network hardening is usually accomplished after the graph was generated [14]. Therefore, countermeasures of attack techniques are considered directly within the KCAG's definition and during its generation. The most strategic countermeasures can remove the largest count of attack paths.

IV. Implementation

We implemented the proposed approach using the multi-step KCAG generation workflow in Figure 2. One input file describes the organization and the second one contains rules based on ATT&CK techniques. The attack graph is consequently generated using MulVAL [15]. Finally, the KCAG generator creates the *KCAG* as a result.

A. Requirements

The *KCAG*'s design and methodology for creating its vertices must depict the kill chain and the attacker's progress through its phases with their proper ordering. We must automatically determine strategic kill chain phases and countermeasures that can be employed.

We decided to fulfill the first requirement by utilizing a ruleset containing selected ATT&CK techniques and their kill chain phases. The ordering of phases is determined by merging the rules into attack paths. Strategic techniques and phases appear in a maximum count of attack paths. These techniques should be addressed by defense measures.

B. Organization's Description

The organization's description in a YAML file contains necessary information about secondary assets (i.e., technologies – hosts, routers, operating systems, software, and network services), network topology, and the organization's missions. Each organization's mission determines requirements on hosts, services, and files. The file also contains a list of identities and user accounts for hosts, services, and domains of hosts.

Vulnerabilities are specified by their CVE IDs. Their impact is categorized into categories specified by our previous work in [17]. These categories are related to code execution, privilege escalation, confidentiality loss, integrity loss, and availability loss. Vulnerabilities are processed differently based on their attack vector from CVSSv3 [18]. Locally exploitable vulnerabilities can be exploited by someone who has access to the system. The attack graph generator considers only vulnerabilities with existing exploits.

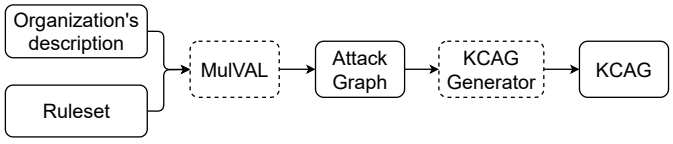


Fig. 2. Kill chain attack graph generation workflow.

Countermeasures specified in the YAML file conform to the D3FEND knowledge graph [7] or were obtained directly from ATT&CK. Last, the input file contains general information about the organization, e.g., whether employees' emails are publicly available on websites.

Information about technologies, network topology, and user accounts can be obtained automatically, e.g., from asset inventory, using network monitoring, and from application and system logs. Vulnerabilities are obtained from NVD and countermeasures from D3FEND or ATT&CK. General information about websites can be obtained using web scraping. Last, mission requirements can be obtained manually or automatically depending on the presence of missions in its inventory.

C. Ruleset

Ruleset can be created in a generic way using the following steps. First, it is necessary to list assets that the defender wants to consider in these categories:

- actors (e.g., external actor and user accounts),
- actions (e.g., sending an email and network connection),
- secondary assets (e.g., operating systems and applications),
- data (e.g., file and email message).

Second, we need to prepare syntactic constructs expressing levels of control over these assets. We prepared predicates in the ruleset file. Third, attack techniques should be mapped to kill chain phases and security properties from the STRIDE model as in Table I. Consequently, countermeasures for the selected techniques (e.g., using D3FEND) should be listed.

The final step is to prepare rules for attack techniques. The rules should express previous levels of asset control, properties of assets, and possible countermeasures. A result of the attack technique is a level of control over some asset of the type allowed in Table II. We manually created a ruleset based on ATT&CK [6]. Its automated creation would require an approach from natural language processing.

Listing 1 contains a rule for the brute force technique. The technique requires a connection to a host *H*. The attacker obtained the second level of its control and violated its authentication by scanning network services. Predicates *networkService* and *hasAccount* identify network service running on the host *H* and the network service's account owner (*Identity*). The strong password policy and multifactor authentication are not used. As a result, the attacker violated the authentication of the network service's user account, i.e., obtained its credentials.

Listing 1. Rule for the brute force technique against network service accounts.

```
account(2, authentication, User, Identity, H,
    Software) :-
    networkConnection(2, authentication, H,
        Protocol, Port),
    networkService(H, Software, Protocol, Port,
        _),
    hasAccount(Identity, User, H, Software),
    strongPasswordPolicy(no),
    multifactorAuthentication(no).
```

D. Generation of the Attack Graph

MulVAL creates the attack graph from the organization's description (i.e., network topology, attack goals, and facts about the organization) and the ruleset. The rules describe how the attacker can escalate control over one asset or move laterally. The generator tries to build an attack path from goals to the initial vertex (in reversed direction) by using rules. The external actor is always the beginning of the attack path.

The ruleset is processed by MulVAL, which determines concrete variable assignment. For example, MulVAL identifies that the brute force rule in Listing 1 is applicable for SSH network service (see Figure 1). MulVAL creates the attack graph but does not determine *KCAG*'s vertices types and kill chain phases. We also do not deal with the probabilities of edges. The approach is as scalable as MulVAL's attack graph generation for generating large attack graphs [15].

E. KCAG Generator

The KCAG generator post-processes the attack graph in Python and outputs the *KCAG* as a result. The KCAG generator assigns to each vertex its label according to its type (e.g., countermeasure) and kill chain phases for attack techniques using the mapping function and the set of phases (see Section III).

The mapping function contains allowed kill chain phases for each technique. Attack paths contain alternating levels of asset control and attack techniques. Therefore, we process subsequent pairs of these attack techniques and their possible phases. We check that assigned phases adhere to their partial ordering. For example, *privilege escalation* cannot be followed by *initial access*. Therefore, if an attack technique belongs to several phases (e.g., *Default Accounts* in Table I), the KCAG generator allows only phases that adhere to the ordering.

The implementation outputs the attacker's strategic techniques and phases belonging to the maximum count of attack paths. The countermeasures recommended by the KCAG generator can be utilized to destroy all attack paths in the created *KCAG*.

V. Validation

In this section, we validate that *KCAG* can be utilized in practice. The section describes a real-world multi-step attack containing phishing with malicious attachment and exploitation of two vulnerabilities. Supplementary materials contain all files necessary to reproduce the validation use case [19].

A. Use Case Setup

Our setup consists of the organization's description and the ruleset files. The organization's description is available in file *organization.yml*. We chose for clarity a simple setup where the organization's network contains a personal computer connected to the Internet. Windows 8.1 and MS Office are installed on the computer. Their vulnerabilities were obtained from the NVD.

The integrity of files on the computer was specified as a mission requirement. The organization publishes emails and employees' names on their public websites. We specified for each considered countermeasure whether the organization employs it. An employee owns an email account and a user account on the PC.

The file *ruleset.P* was created manually. The first part of the ruleset contains predicates that represent instances of *KCAG* vertices except for attack techniques. The attack techniques are specified by rules. MulVAL processes the ruleset and depicts the attack graph in the file *AttackGraph.pdf*.

B. Kill Chain Attack Graph

The final kill chain attack graph is depicted in Figure 3, and its vertices are listed in Table III. It describes a multi-step attack with the setup mentioned above. The attacker obtains information about the employees' emails from the organization's website. The attacker sends a convincing phishing email with a malicious MS Word document.

When the curious employee opens the malicious attachment, a code execution vulnerability (*CVE-2017-0262*) is consequently exploited. Its exploitation may be accompanied by exploiting the operating system's *CVE-2017-0263*. This CVE allows privilege escalation to administrator privileges. Both vulnerabilities can be exploited locally. These vulnerabilities were exploited in the real world, e.g., by APT28 [6].

Finally, the attacker can modify files stored on the personal computer, i.e., encrypt or even destroy them. Modification of files will negatively impact the organization's services. All countermeasures from Figure 3 are suggested as strategic because each one of them can destroy attack paths.

Figure 3 depicts *KCAG* with two attack paths differing in their final attack technique. KCAG generator categorized *KCAG*'s vertices. Moreover, it assigned allowed kill chain phases to attack techniques. The KCAG generator revealed how chains of possible attacker's actions might materialize on protected assets, i.e., used kill chain phases and their ordering.

VI. Conclusion

Our research effort focused on cyber threat identification based on asset management data and data about threats. We proposed a new kill chain attack graph for modeling sequences of the attacker's actions. The kill chain concept only shows the attack lifecycle, and attack graphs do not use any standardized set of actions. Therefore, our adjusted methodology incorporated ATT&CK techniques and security properties from the STRIDE threat model.

Supplementary materials contain the kill chain attack graph generator in Python [19]. The *KCAG*'s implementation was validated on a real-world use case and a set of techniques. The approach gives the security administrators a way to describe the organization, specify attack rules, and reveal the materialization of multi-step network attacks, including strategic attack techniques and possible countermeasures.

Our future work is to implement generation in an imperative language. It would allow defining asset hierarchy and complicated rules for the attack graph generation. We would also like to use strategic attack steps observed by detection systems to assess the severity of possible multi-step cyber attacks.

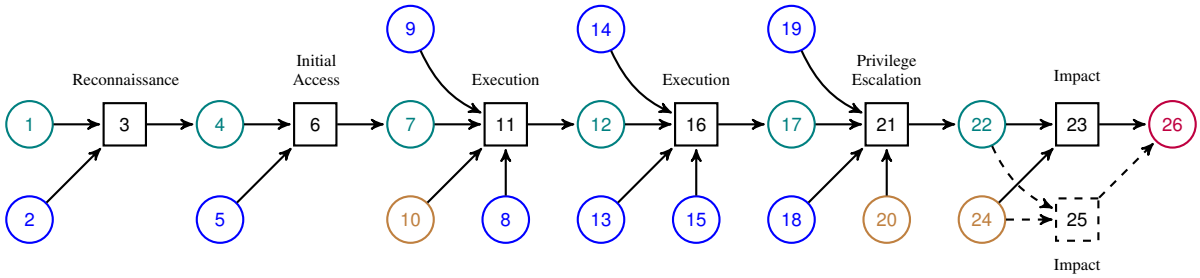


Fig. 3. Example kill chain attack graph for the validation use case. Vertices correspond to attacker's level of control over assets (green), asset properties (blue), countermeasures (brown), attack techniques (black), and an attack goal (dark red). Vertices are described in Table III.

TABLE III
ATTACK GRAPH VERTICES

ID	Description
1	External actor.
2	Email address of an employee was published on a website.
3	T1594 – Search Victim-Owned Websites.
4	The attacker knows that the email address exists.
5	Sender reputation analysis was not accomplished.
6	T1566.001 – Spearphishing Attachment.
7	Authentication of sending an email was violated.
8	The employee can click on the attachment.
9	The employee has a user account on a personal computer.
10	Training of users was not accomplished (countermeasure).
11	T1204.002 – User execution of malicious file.
12	Authentication of opening file action was violated.
13	Microsoft Office opens files.
14	Microsoft Office is installed on the personal computer.
15	Microsoft Office 2016 contains CVE-2017-0262 vulnerability.
16	T1203 – Exploitation for Client Execution.
17	The attacker violated the system's authorization (user rights).
18	Microsoft Windows 8.1 is installed on the personal computer.
19	Microsoft Windows 8.1 contains CVE-2017-0263 vulnerability.
20	Software is not regularly updated.
21	T1068 – Exploitation for Privilege Escalation.
22	The attacker violated the system's authorization (admin rights).
23	T1485 – Data Destruction.
24	Data backup was not accomplished (countermeasure).
25	T1486 – Data Encrypted for Impact.
26	Integrity of a sensitive file was violated.

ACKNOWLEDGMENT

This research was supported by the CONCORDIA project that has received funding from the European Union's Horizon 2020 research and innovation programme under the grant agreement No 830927.

REFERENCES

[1] Z. M. Smith and E. Lostri, "The Hidden Costs of Cybercrime," McAfee, San Jose, CA, USA, Tech. Rep., 2020. [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf

[2] R. A. Caralli, J. H. Allen, and D. W. White, *CERT Resilience Management Model - CERT-RMM*. Glenview, IL, USA: Addison-Wesley Educational Publishers Inc, 2016.

[3] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare & Security Research*, vol. 1, no. 1, pp. 80–106, 2011.

[4] T. Yadav and A. M. Rao, "Technical aspects of cyber kill chain," in *Security in Computing and Communications*, J. H. Abawajy, S. Mukherjea, S. M. Thampi, and A. Ruiz-Martínez, Eds. Cham: Springer International Publishing, 2015, pp. 438–452.

[5] P. Pols, "The Unified Kill Chain," accessed: Jan 17, 2022. [Online]. Available: https://www.unifiedkillchain.com/assets/The-Unified-Kill-Chain.pdf

[6] "MITRE ATT&CK," The MITRE Corporation, 2015 – 2021, accessed: Jan 14, 2022. [Online]. Available: https://attack.mitre.org/

[7] "MITRE D3FEND," The MITRE Corporation, 2021, accessed: Jan 14, 2022. [Online]. Available: https://d3fend.mitre.org/

[8] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the 1998 workshop on New security paradigms*, 1998, pp. 71–79.

[9] N. Poolsappasit, R. Dewri, and I. Ray, "Dynamic Security Risk Management Using Bayesian Attack Graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 61–74, Jan 2012.

[10] W. Xiong, E. Legrand, O. Åberg, and R. Lagerström, "Cyber security threat modeling based on the MITRE Enterprise ATT&CK Matrix," *Software and Systems Modeling*, pp. 1–21, 2021.

[11] "CVE - Common Vulnerabilities and Exposures (CVE)," The MITRE Corporation, 1999 – 2022, accessed: Jan 17, 2022. [Online]. Available: https://cve.mitre.org/

[12] M. Inokuchi, Y. Ohta, S. Kinoshita, T. Yagyu, O. Stan, R. Bitton, Y. Elovici, and A. Shabtai, "Design procedure of knowledge base for practical attack graph generation," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, pp. 594–601.

[13] M. U. Aksu, K. Bicakci, M. H. Dilek, A. M. Ozbayoglu, and E. i. Tatli, "Automated generation of attack graphs using NVD," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, 2018, pp. 135–142.

[14] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, vol. 29, pp. 27–56, 2016.

[15] X. Ou, S. Govindavajhala, and A. W. Appel, "MulVAL: A Logic-based Network Security Analyzer," in *USENIX security symposium*, vol. 8. Baltimore, MD, 2005, pp. 113–128.

[16] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, "Threat Modeling – Uncover Security Design Flaws Using the STRIDE Approach," *MSDN Magazine*, 2006. [Online]. Available: https://docs.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach

[17] J. Komárková, L. Sadlek, and M. Laštovička, "Community based platform for vulnerability categorization," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018, pp. 1–2.

[18] "Common Vulnerability Scoring System v3.1: Specification Document," FIRST — Forum of Incident Response and Security Teams, accessed: Jan 17, 2022. [Online]. Available: https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf

[19] L. Sadlek, P. Čeleda, and D. Tovarňák, "Supplementary Materials: Identification of Attack Paths Using Kill Chain and Attack Graphs," 2022, accessed: March 25, 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6367986