Automating Network Perimeter Threat Prevention for Decentralized Network Administration

Nikolas Wintering, Eric Lanfer, and Nils Aschenbruck
Institute of Computer Science, Osnabrück University
Friedrich-Janssen-Str. 1
49076 Osnabrück, Germany
{nwintering, lanfer, aschenbruck}@uos.de

***Abstract*—Decentrally administered networks consist of a plethora of hosts providing various services to the Internet and several administrators are responsible for mitigating software vulnerabilities. Therefore, these networks potentially expose a large attack surface depending on the capabilities and workload of administrators. In this paper, we present the automateD nETwork pERimeter thREat pRevention System (DETERRERS). It automatically scans hosts at the network perimeter, assesses the risk of exposed vulnerabilities, and performs actions based on the assessed risk. This supports administrators in their work and enables faster reaction to software vulnerabilities. Additionally, host-based security policies can be configured in a modular user interface by system administrators and firewall configurations can be generated automatically. We deploy our system in a university network with decentralized administration and evaluate the risk assessment process, the influence on the attack surface of the network, and the time-to-remediate from vulnerabilities.**

***Index Terms*—Decentralized Management, SOAR, Threat Prevention, Vulnerability Management**

I. Introduction

Public institutions such as universities whose networks are commonly administered decentrally, are continuously targeted by cyber-attacks. From June 1, 2022 to June 30, 2023 alone, there were 23 successful ransomware attacks against German education and research institutions [1]. These networks expose a large attack surface to the Internet. At the same time, they handle personal data of university members and valuable research data. Therefore, they present a lucrative target for cyber criminals who may try to steal this data or encrypt it to demand ransom.

In a decentralized network, organizational departments manage their own systems, like websites and servers, within a shared network. Different individuals oversee administration across these departments, while a central network operations center is tasked with securing the overall network perimeter. Consequently, network security involves multiple personnel across the organization [2].

Network security operations involve various tools and techniques. System administrators often manage a local Firewall (FW), which requires maintenance, while security administrators use multiple tools, including a perimeter FW and Vulnerability Scanner (V-Scanner). Based on reports provided by the V-Scanner, risk assessment and actions are performed manually, requiring high levels of effort by the personnel. In recent years, the concept of Security Orchestration, Automation, and Response (SOAR) has emerged in research and industry [3]. It aims at automating processes, workflows, and applications to reduce the workload of security administrators. However, we observe a lack of research regarding SOAR for decentralized network administration.

Our main contributions are as follows:

1) We present an open-source tool[1] for decentralized network administration which automates (i) interactions between system administrators, security administrators, a V-Scanner, and a perimeter FW, (ii) a rule-based approach to vulnerability risk assessment, and (iii) the configuration of host-based FWs.
2) We decrease the attack surface of a decentrally administered university network by deploying our tool.
3) We quantify the Time-to-Remediate (TTR) from vulnerabilities at the network perimeter and gain insights into the vulnerability lifetime during deployment.

Section II provides background and gives references to related work. Next, Section III describes the architectural design of DETERRERS. Afterward, in Section IV we evaluate the system. Finally, Section V draws conclusions, discusses limitations, and presents future work aspects.

II. Background

A. Technical Prerequisites

DETERRERS is embedded into a certain technical landscape. Firstly, it interacts with a network-based V-Scanner which scans single machines or entire networks for software vulnerabilities. The V-Scanner automatically gathers information about the target systems and reports the severity of publicly disclosed vulnerabilities by means of Common Vulnerability Score System (CVSS) scores [4]. The expressiveness of a scan depends on the quality and quantity of Network Vulnerability Tests (NVTs) in a test corpus. Secondly, DETERRERS interacts with a perimeter FW (also called network FW). In contrast to perimeter FWs, host-based FWs run directly on host systems and monitor all traffic to and from a certain host.

B. Related Work

A literature research specifically on the interaction between V-Scanners and FWs did not yield much work directly comparable to our application scenario. The closest related work by [5]

[1] https://github.com/UOS-RZ/deterrers

978-3-903176-66-9

proposes a game-theoretic approach of improving interactions between FWs, V-Scanner, and Intrusion Detection Systems (IDSs) from an economic point of view.

In recent years, a new paradigm called Security Orchestration, Automation, and Response (SOAR) has emerged [3]. SOAR systems streamline cyber defense operations by integrating different security applications and processes. Examples are incorporating artificial intelligence into SOAR [6] or integrating honeypots into SOAR systems [7]. However, most approaches assume that security administrators have complete knowledge and control over network and systems. This is not the case for decentral network administration where responsibilities are federated among multiple departments of an organization. We did not find any previous research in this area. Hence, we propose DETERRERS as a first approach at incorporating decentral network administration into the SOAR paradigm. Next, we survey related work that is comparable to parts of our approach.

Vulnerability Risk Assessment: Vulnerability reports of the V-Scanner contain CVSS vectors for each vulnerability that is found. The most naive risk assessment approach, would be CVSS based isolation. However, other factors (e.g., importance of services provided by the host, etc.) might be relevant for the assessment, too. A Bayesian Belief Network is proposed in [8] that estimates risk levels based on impact and frequency of vulnerabilities derived from CVSS vector attributes. In order to mimic the human expert decision-making process in the classical manual assessment of vulnerabilities, [9] introduces a neural network based approach that learns from these experts' decisions. CVSS has also spawned critique for its focus on technical severity and lack of focus on actual risk [10]. For this reason, approaches assessing and prioritizing the severity of vulnerabilities have been proposed [11], [12], [13]. Especially deep learning approaches analyzing the vulnerability descriptions have raised attention in the past years. This eliminates the need for manually deriving a severity score, as is the case with CVSS.

Host-Based Firewall Configuration and Rule Generation: For the generation of FW rules different approaches have been proposed in literature. A survey of approaches for the automatic translation of high-level security policies to low-level FW rules is conducted in [14]. One of these approaches which uses information for the modeling of security policies that is similar to the available information in DETERRERS, can be found in [15]. The authors propose a FW rule generation approach for SDN by using a logical service graph to model the rules. A Domain-Specific Rule Generation algorithm is introduced in [16]. It uses different kinds of network traffic logs and security logs to formulate generalized rules for anomaly checking or supplementing existing rule sets. Another approach, is a Model-Driven Development (MDD) framework for the "design, development and maintenance" of FW rule sets, introduced in [17]. However, with regards to our application scenario, an MDD framework would introduce unnecessary overhead.

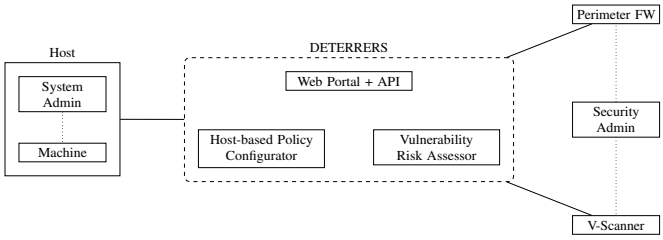


Fig. 1: Schematic overview of DETERRERS' architecture.

III. Architectural Design

DETERRERS is the central building block in automating interactions between a V-Scanner, perimeter FW, system and security administrators (see Fig. 1). On the one side, system administrators access DETERRERS via web interface or API and receive notifications via email. Moreover, they may use the Host-based Policy Configurator to generate rules for their host-based FWs. On the other side, DETERRERS controls a V-Scanner to schedule scan tasks. Hosts which provide services to the Internet are frequently scanned for vulnerabilities and scan results are processed by the Vulnerability Risk Assessor. Based on assessments, the perimeter FW is configured by DETERRERS. It is always up-to-date about which hosts expose which services to the Internet and it allows only traffic belonging to these services. Furthermore, the perimeter FW can be configured to isolate vulnerable hosts from the Internet. Security administrators are kept out of the loop and may monitor the V-Scanner and perimeter FW but do not interact directly with DETERRERS.

In this section, we describe requirements and implementation details of DETERRERS in four steps.

A. Inventory of Hosts and Services at the Perimeter

System administrators should be able to register hosts that provide services to the Internet, either by using a graphical self-service portal or by a web-based API. As part of the process, administrators specify an *Internet Service Profile* which defines services provided, examples are HTTP/S or SSH profiles. DETERRERS configures the perimeter FW automatically by grouping hosts with the same Internet Service Profile and applying FW policies to these groups. All traffic deviating from this profiles can be denied by default. However, it depends on system administrators to specify the most restrictive service profile to keep the attack surface small.

B. Vulnerability Scan Orchestration

When a host is registered, DETERRERS automatically triggers a vulnerability scan for this host. Moreover, it plans periodical scans of all registered hosts. By this we ensure that (1) hosts are not initially vulnerable when they enter the network perimeter and (2) the exposed hosts are constantly monitored for novel vulnerabilities.

C. Automated Risk Assessment

Vulnerability scan reports are collected by DETERRERS for an automated assessment, combining scan results with context-

and host-specific information. Depending on context, different actions should be taken: With a high risk of Internet exposure, a host may be blocked by the perimeter FW preemptively. With a medium risk, notifying the system administrator is sufficient. Meanwhile, with a high risk of exposure to the internal network, notification of host or security administrators is the most effective measure in this case.

We derive important features for risk assessment from the scan result schema. Firstly, the *Quality of Detection (QoD)* metric measures the reliability of a scan result. The *CVSS base score* and vector measure the technical severity. However, they are unaware of any context a vulnerability is found in. If an exploit mainly affects the availability of a host, blocking it is no effective countermeasure. In order to account for this, an *adapted CVSS score* is computed by setting the "Availability Impact"-metric to "None". See [4] for more detailed information on how CVSS scores are computed. *Remote Exploitability* is another important characteristic we extract from the CVSS-"Attack Vector"-metric. Finally, there is no risk of Internet exposure if the vulnerable *Port* and *Protocol* are not in the Internet Service Profile.

Fig. 2a shows the decision tree to decide if a host should be blocked and Fig. 2b shows the decision tree if a system administrator should be notified about a vulnerability. A host should be blocked if the risk of Internet exposure is high. In case this risk is medium or the risk of exposure towards the internal network is high, administrators should just be notified.

Risk assessment process and actions are completely automated by DETERRERS. Merely the threshold parameters `QoD-T`, `CVSS-H-T`, and `CVSS-M-T` have to be chosen manually and no extra workload is introduced for security administrators.

D. Host-based Firewall Policy Configuration

Independent of the steps described above, system administrators can configure policies for their host-based FW and generate configuration scripts for different FW tools. This optional feature assists system administrators that only have limited time or little experience with host-based FWs. Therefore, our goal is to provide a simple and intuitive interface to enable configuring policies with little effort. We decide to focus on the current default FW tools of frequently-used server operating systems (i.e., *ufw*, *firewalld*, and *nftables*). DETERRERS automatically generates configuration scripts for all these tools, regardless of the different syntax. A default-allow policy for outgoing and a default-deny policy for incoming traffic are set. The interface for configuring policies enables users to define exceptions to the default-deny policy on incoming traffic. The generated configuration scripts can be downloaded and executed on the corresponding host. We decide against a remote deployment of the FW configurations because some system administrators oppose the idea of enabling remote access.

IV. Evaluation

A. Description of a Test Deployment

For the evaluation, we deploy DETERRERS in selected segments of a university network. We argue that this is a typical example of decentralized network administration. Therefore, our results should be representative for all kinds of decentrally administered networks. The selected segments correspond to a maximum of 2048 unique target IPv4 addresses, where approximately 600 hosts responded to a port scan before the test, and about 30 administrators are responsible for these hosts. We deploy a Greenbone Enterprise 450 Appliance as network-based V-Scanner which builds on the open-source project Open Vulnerability Assessment Scanner (OpenVAS)[2]. The perimeter FW is a Palo Alto PA-5220.

This is the first real test of our system and blocking hosts after periodical scans might have serious negative implications in case of false positives during the risk assessment. Therefore, hosts are only isolated upon high risks during registration. If a high risk vulnerability is found during periodical scans, system administrators are just notified. This is easily achieved by configuring the system thresholds described in Section III-C.

B. Effect on Attack Surface

In order to assess the reduction of the attack surface we distinguish two viewpoints: Towards the internal network and towards the Internet. We conduct two port scans of the selected network segments at daytime on working days. Inexperienced or overloaded system administrators may have a lack of knowledge or time to configure their systems' host-based FWs appropriately. This leads to exposed services on a variety of ports. By comparing the amount and distribution of open ports we are able to draw conclusions about unnecessarily exposed services. The scans are performed by a host inside the university network where no security tool interferes with the scans.

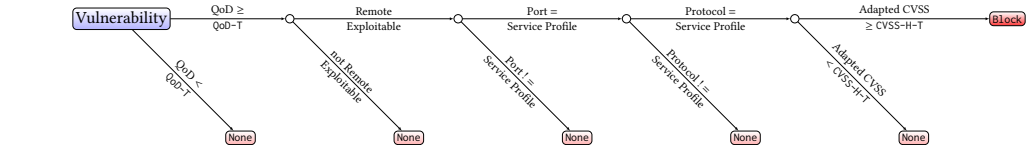
The first scan is performed before the test deployment to get a baseline and the second one after about nine month of deployment. Moreover, we conduct scans to verify differences regarding the time of day, however, we observed same numbers of hosts during day and night. We use the following command to conduct the scans:

```
nmap -vvv --noninteractive -Pn
    --top-ports 3500 -sS -oA <filename>
    -T4 --min-hostgroup 256 <ip_ranges>
```

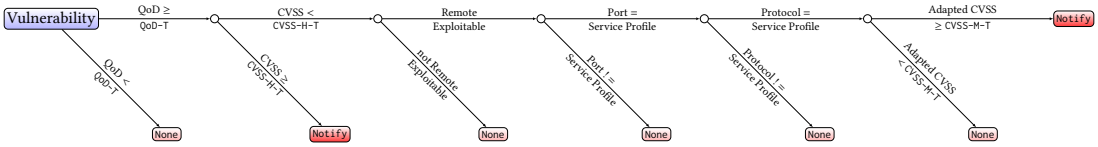
Details on the configuration parameters can be found in [18].

The baseline scan before the deployment was performed on February 2, 2023, with 595 IP addresses responding. The second on November 14, 2023, with 714 IPs responding. Both scans took approx. 2 hours. We cannot perform a port scan from an external point because the perimeter FW would detect and block this. However, before our test, there was no default-deny policy at the perimeter and virtually all incoming traffic was allowed. Hence, we argue that the interior port scan had a similar point of view as an external scan would have had.

[2]https://www.openvas.org/



(a) Block action



(b) Notify action

Fig. 2: Decision trees for which action to take based on the risk assessment.

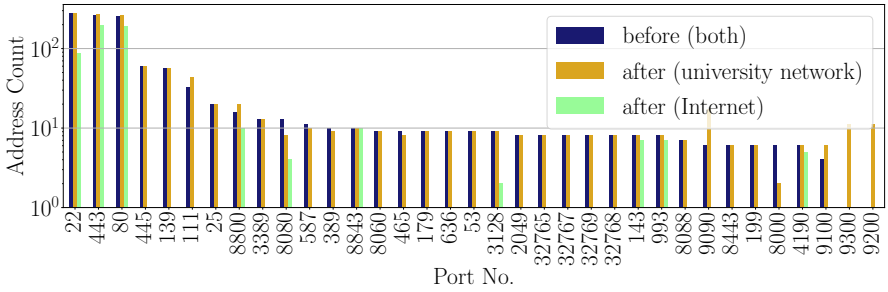


Fig. 3: Number of IP addresses per open TCP port before and after deployment. Point of view in parentheses. $n = 505$ hosts. Excluded 98 ports which are open on $< 1\%$ of IPs.

The external viewpoint post-deployment is estimated by combining the interior port scan with host-specific data from DETERRERS. The Internet Service Profile indicates which ports are exposed by the perimeter FW for each host. We believe that we can accurately infer the external viewpoint from the interior scan using this data. However, local FW rules may affect the results. Additionally, there is a significant difference in the number of hosts before and after deployment, caused by the addition and removal of hosts during the 9-month testing period. In the evaluation, we focus only on the 505 hosts present in both port scans.

Fig. 3 shows the numbers of IP addresses that have a respective port open during the scans before and after our test. The ports that are open most often during both scans are 22, 443, and 80. About 50% of all IP addresses have these ports open to the university network before and after the deployment. There are many ports which are open only on a smaller number of IPs. The differences between the port scans before and after the test deployment are minimal for the university network viewpoint. We conclude that there is no visible reduction of interior attack surface regarding individual ports.

In contrast, the exterior attack surface is reduced noticeably. Ports 443 and 80 remain being exposed on nearly 200 hosts. However, the number of IP addresses exposing port 22 is reduced by nearly 75%. Moreover, the number of IP addresses exposing more uncommon ports is also reduced drastically. This shows that DETERRERS is effective at deploying a default-deny policy at the perimeter FW so that only well-defined services are exposed to the Internet. Security-sensitive services, e.g. SMB on port 139 and 445, are thereby effectively isolated.

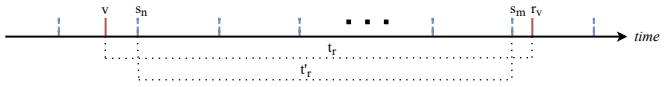


Fig. 4: Schematic approach for approximating the TTR.

C. Vulnerability Lifetime

We evaluate the Time-to-Remediate (TTR) from vulnerabilities by collecting scan results of perimeter-scans in the test deployment for nearly ten months. As visualized in Fig. 4, we approximate the TTR t_r by measuring the time t'_r between the first scan s_n after vulnerability v occurs and the last scan s_m before a remediation r_v is performed. This approximation becomes more accurate by reducing the time between scans. We schedule the periodical scans once a week. However, it should be noted that rarely scans get canceled due to high utilization of the V-Scanner.

Results are shown in Fig. 5. The majority of closed vulnerabilities has a CVSS base score of less than or equal to 5.0. No vulnerabilities with a CVSS base score of 0.0–2.0 were remediated, as such scores are typically rare (see Figure 6a). Furthermore, scores of 3.0–4.0 are also uncommon, raising questions about the distribution of CVSS base scores in practice. However, we deem further evaluation of this issue as out of

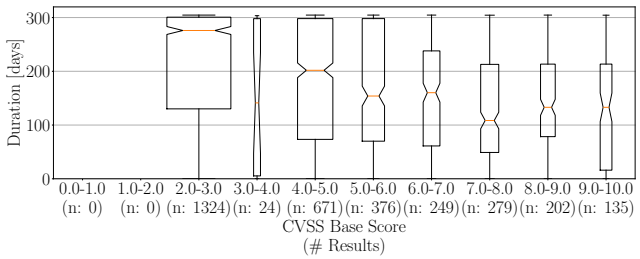


Fig. 5: TTR of vulnerabilities per CVSS base score with $QoD >= 70$. Boxwidth indicates the number of samples.

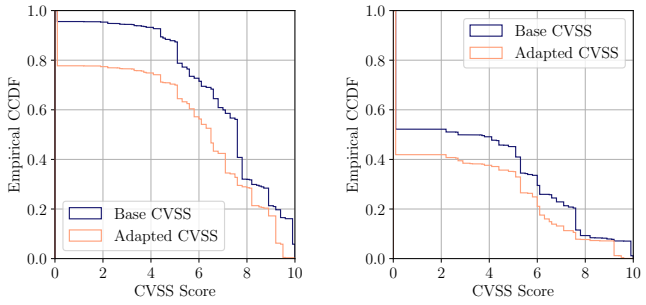


Fig. 6: ECCDF of CVSS scores discretized in 100 bins of equal size for **(a)** all NVTs in the Greenbone Feed and **(b)** scan results before the deployment.

scope for this paper since we are not mainly concerned with qualitative characteristics of the CVSS metric.

Overall, it can be seen that the mean TTR for lower CVSS base scores is longer than the mean TTR for higher CVSS base scores. This suggests that prioritization of vulnerabilities happens successfully. The mean TTR is shortest for scores of 7.0-8.0 with approximately 110 days. For scores of 8.0-10.0 the mean TTR is a bit longer but still less than 140 days. It should be noted that confidence intervals in this area are increasing due to decreasing number of samples. The longest mean TTR is approximately 280 days for CVSS base scores of 2.0-3.0. This indicates that system administrators do not bother fixing vulnerabilities with such low scores.

All scores have a maximum TTR of approximately 310 days which is similar to the measurement duration of 10 month. We argue that those scores are mainly caused by false positives which are continuously detected by vulnerability scans even though they are not present in reality. Developing a mechanism for suppressing continuous false positive notifications is seen as future work.

All in all, we are able to quantify the TTR at the network perimeter for the first time with this approach. This was not easily possible before the deployment of DETERRERS.

D. Risk Assessment

To evaluate the risk assessment process, we examine the adapted CVSS score by calculating it for all 161,436 NVTs in the Greenbone Enterprise Feed as of October 25, 2023. Figure 6a visualizes the Empirical Complementary Cumulative Distribution Function (ECCDF) of CVSS base and adapted scores, showing the proportion of scores exceeding various thresholds, starting at 1.0 since all scores are ≥ 0.0. The drop in the function for base and adapted CVSS is different in magnitude. While the proportion of base scores greater than 0.1 still is above 0.95, the corresponding proportion of adapted scores is less than 0.8. This vertical difference of 0.2 persists for CVSS scores of up to 5.0 approximately. This means that there are 20% less adapted scores that exceed a threshold of 5.0 compared to base scores. Finally, the ECCDF of adapted CVSS scores drops to 0.0 at a score of 9.5 while the ECCDF of base scores is still at nearly 0.2. We conclude that the number of NVTs which exceed some CVSS threshold is potentially 20% smaller for adapted CVSS than for base CVSS. This means that our adapted CVSS potentially reduces the number of vulnerabilities that trigger the risk assessment.

The distribution of CVSS scores among NVTs does not necessarily correlate with the distribution of CVSS scores among vulnerability results from practice. In Fig. 6b, we consider the ECCDF of CVSS scores found in the test network segments before our deployment. The total number of vulnerability results is 68915 found for 876 unique IP addresses. As mentioned in Section IV-A, we only reached approximately 600 IPs during a port scan before the test deployment. We argue that this difference is because during our port scan some hosts were not reachable by the machine we ran the port scan on. Furthermore, the vulnerability scanner as a security tool is known to system administrators and might be whitelisted in host-based FW configurations. Meanwhile, the machine from which we ran our port scan is unknown to system administrators and denied more often by host-based FWs.

As can be seen in Fig. 6b, both functions drop further in the beginning compared to Fig. 6a. About 50% of all CVSS base scores and about 60% of all adapted scores in the test deployment scan results are smaller than or equal to 0.1. Afterward, the functions behave similar to Fig. 6a in that the ECCDF of adapted scores consistently lies below the base scores. However, the vertical difference between both functions is only about half the size compared to Fig. 6a. Notably, for CVSS scores of approx. 8-9 both functions are close together. We conclude that, for actual scan results in the test deployment, the number of adapted CVSS scores that surpass some threshold is potentially up to 10% less than CVSS base scores.

Next, we quantitatively assess the influence of different parameters on the results of the complete risk assessment process. As described in Section III-C, the risk assessment process involves three thresholds: `QoD-T`, `CVSS-M-T`, and `CVSS-H-T`. We evaluate the influence of these parameters by performing the risk assessment with different discrete parametrizations for scan results from before the test deployment. As mentioned above, these results contain vulnerabilities found on 876 IP addresses of which 873 could be matched to hosts registered in DETERRERS at the point of evaluation.

Fig. 7a shows, in form of a heatmap, the number of hosts in the test deployment which theoretically would be blocked following the risk assessment. As expected, with rising thresholds the number of blocked hosts decreases. As can be seen, the rows for QoDs of 60, 70, and 80% are nearly identical. This indicates that the number of blocked hosts is stable for a `QoD-T` between 60-80%. The recommended QoD by Greenbone for a good trade-off between false positives and false negatives is 70%. Our results align with this recommendation. The CVSS standard version 3.1 [4] classifies scores of 7.0 and above as high. The number of blocked hosts with `QoD-T` of 70% and a `CVSS-H-T`=7.0 is 48. Here, increasing or decreasing `CVSS-H-T` has a bigger impact. While for `CVSS-H-T`=6.0 the number of blocked hosts is twice as high as with `CVSS-H-T`=7.0, for `CVSS-H-T`=8.0 the number is less than half of it.

The number of blocked hosts correlates to the number of vulnerabilities that trigger blocking which is visualized in Fig. 7b. Again, the number of block reasons is stable for a `QoD-T` between 60-80%. For `CVSS-H-T` between 6.0-8.0 the change in block reasons is similar to the change in blocked hosts.

For the evaluation of the risk assessment with regard to the notification about vulnerabilities, we settle on a `QoD-T` of 70% and only vary the `CVSS-H-T` and `CVSS-M-T`. We argue that this threshold for QoD is viable because it is recommended by Greenbone and changing the threshold in the range of 60-80% has nearly no influence on the block-decisions, as shown above. Fig. 7c depicts the number of hosts in the test deployment which theoretically would be notified according to our risk assessment. Since it does not make sense to have a medium CVSS threshold which is equal to or higher than the high CVSS threshold, everything above the diagonal from the top left to bottom right can be disregarded. Both CVSS thresholds have an impact on the number of notified hosts. Generally, increasing any of the two thresholds decreases the number of notified hosts. The CVSS standard version 3.1 classifies scores equal to and above 4.0 as medium. Increasing `CVSS-H-T` above 6.0 does not change the results in general.

Looking at the numbers of vulnerabilities that trigger a notification in Fig. 7d, we see that the `CVSS-H-T` does have more of an impact on the decision process. Here, for increasing `CVSS-H-T`, the number of notify reasons also increases noticeably. This is because the number of block reasons decreases for increasing `CVSS-H-T` and more vulnerabilities start to trigger a notification instead of blocking. In contrast, for increasing `CVSS-M-T`, the number of notify reasons decreases.

V. Conclusion

In this paper, we present the design of an automated network perimeter threat prevention system for decentralized network administration (DETERRERS) and evaluate its performance. As a SOAR prototype for decentral network administration, it integrates and automates workflows between system administrators, security administrators, V-Scanners, and perimeter FWs, reducing the workload of administrators. Its most important features are building an inventory of exposed hosts and services, enforcing a default-deny policy at the network perimeter, and automated scanning for and assessment of vulnerabilities without direct involvement of security administrators.

The evaluation of the attack surface after a test deployment reveals a significant reduction in exposed ports towards the Internet. Furthermore, we are able to estimate the lifetime of vulnerabilities with DETERRERS due to the continuous scanning and assessment of the perimeter. An evaluation in the deployment reveals that the mean TTR decreases for increasing CVSS base scores. Finally, we analyze the influence of different parameters in the risk assessment process.

Based on our results, we see great potential in combining capabilities of V-Scanners and perimeter FWs. However, combining these sources of information manually is laborious. By automating this process with a SOAR tool we are able to decrease the workload of administrators and increase the security level of a network. Until now, the investigation of SOAR for decentral network administration is an overlook field of research and we provide fundamental work with this paper. Future work could investigate the inclusion of other security tools such as IDS.

When developing a central IT security tool which is used by many administrators spread across an entire organization, we should also discuss risks that are implied by it. User authentication and authorization for accessing DETERRERS should be handled by appropriate tools such as identity providers and access management systems. Furthermore, all the tasks performed by DETERRERS can still be performed or reverted manually by security administrators. Therefore, our system does not introduce a single point of failure.

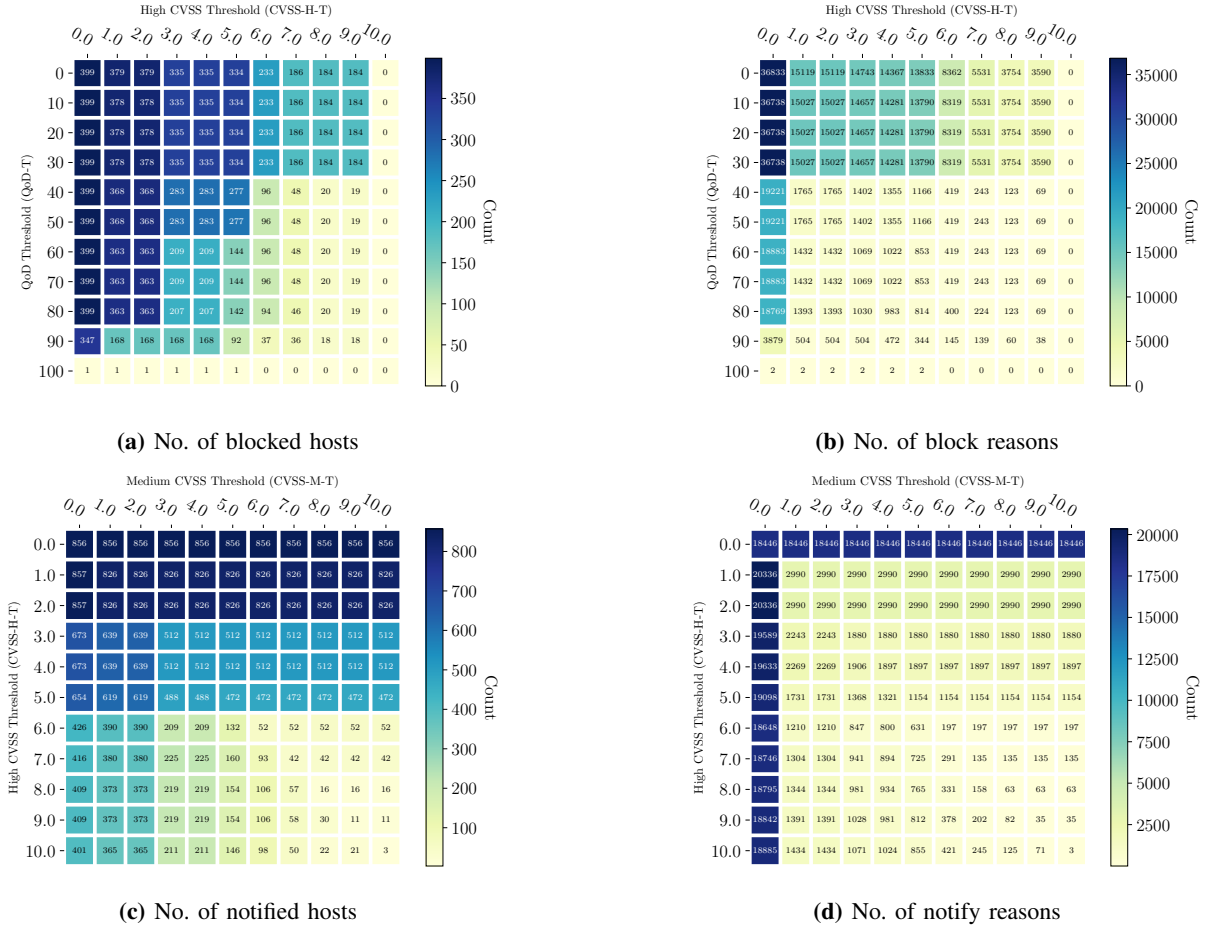
We limited our evaluation of the risk assessment process to a quantitative analysis of the parameters and their influence on results in a test deployment. Future work could focus on the qualitative evaluation of risk assessment results by incorporating domain expert knowledge. Moreover, irregularities in the distribution of CVSS base scores raise concerns about its suitability (see Section IV-C). Other risk assessment approaches which are not based on CVSS could be implemented and compared to each other. Regarding future features, DETERRERS could be extended by a module that checks for the presence of a host-based FW on systems. This would provide a better overview of the security measures at the network perimeter and increase the incentive to apply host-based FWs further. Finally, our evaluation is limited to a single test deployment. Evaluation in other contexts might reveal further requirements for SOAR in decentral network administration.

Acknowledgment

We would like to thank Greenbone AG for their interest in our work and discussing topics about vulnerability assessment.

References

[1] Fed. Office for Inf. Secur. (BSI), "The State of IT Secur. in Germany in 2023," Standard, 2023. [Online]. Available: https://bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Securitysituation/IT-Security-Situation-in-Germany-2023.html



(a) No. of blocked hosts

(b) No. of block reasons

(c) No. of notified hosts

(d) No. of notify reasons

Fig. 7: Heatmaps of the assessment of 68915 scan results from the test deployment.

[2] G. Stoneburner, A. Goguen, and A. Feringa, *Risk management guide for inf. technology syst.*, special publication 800-30 ed., Nat. Institute of Standards and Technology, 2002.

[3] C. Islam, M. A. Babar, and S. Nepal, "A Multi-Vocal Review of Security Orchestration," *ACM Comput. Surv.*, vol. 52, no. 2, 2019.

[4] "Common Vulnerability Scoring System version 3.1," Forum of Incident Response and Secur. Teams (FIRST), Specification Document Ver. 3.1, 2019. [Online]. Available: https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf

[5] Z. Liu-Rong, S.-E. Mei, and W.-J. Zhong, "An Economic Analysis of the Interaction Between Firewall, IDS and Vulnerability Scan," *Economic Comput. & Economic Cybernetics Studies & Research*, vol. 49, no. 4, 2015.

[6] L. A. Johnson Kinyua, "AI/ML in Security Orchestration, Automation and Response: Future Research Directions," *Intelligent Automation & Soft Computing*, vol. 28, no. 2, 2021. [Online]. Available: http://www.techscience.com/iasc/v28n2/42057

[7] U. Bartwal, S. Mukhopadhyay, R. Negi, and S. Shukla, "Security Orchestration, Automation, and Response Engine for Deployment of Behavioural Honeypots," in *Proc. of the 2022 IEEE Conference on Dependable and Secure Computing (DSC)*. IEEE, 2022.

[8] S. H. Houmb, V. N. Franqueira, and E. A. Engum, "Quantifying secur. risk level from CVSS estimates of frequency and impact," *J. of Syst. and Softw.*, vol. 83, no. 9, 2010.

[9] A. Beck and S. Rass, "Using neural netw.s to aid CVSS risk aggregation — An empirically validated approach," *J. of Innovation in Digital Ecosyst.*, vol. 3, no. 2, 2016.

[10] J. Spring, E. Hatleback, A. Householder, A. Manion, and D. Shick, "Time to Change the CVSS?" *IEEE Secur. and Privacy*, vol. 19, no. 2, 2021.

[11] H. Kekül, B. Ergen, and H. Arslan, "A multiclass hybrid approach to estimating softw. vulnerability vectors and severity score," *J. of Inf. Secur. and Applications*, vol. 63, 2021.

[12] T. H. M. Le, H. Chen, and M. A. Babar, "A Survey on Data-Driven Softw. Vulnerability Assessment and Prioritization," *ACM Computing Surveys*, vol. 55, no. 5, 2022.

[13] R. Sharma, R. Sibal, and S. Sabharwal, "Softw. vulnerability prioritization using vulnerability description," *Int. J. of System Assurance Eng. and Manage.*, vol. 12, 2021.

[14] I. Kovačević, B. Štengl, and S. Groš, "Systematic review of automatic translation of high-level secur. policy into firewall rules," in *Proc. of the 45th Jubilee Int. Convention on Inf., Communication and Electronic Technology (MIPRO)*, 2022.

[15] D. Bringhenti, G. Marchetto, R. Sisto, F. Valenza, and J. Yusupov, "Introducing programmability and automation in the synthesis of virtual firewall rules," in *Proc. of the 6th IEEE Conf. on Netw. Softwarization (NetSoft)*, 2020.

[16] W. Li, H. Wan, and S. Li, "An approach to the generalization of firewall rules," in *Proc. of the IEEE/ACIS 12th Int. Conf. on Comput. and Inf. Science (ICIS)*, 2013.

[17] S. Pozo, R. M. Gasca, A. Reina Quintero, and A. Varela Vaca, "CONFIDDENT: A model-driven consistent and non-redundant layer-3 firewall ACL design, development and maintenance framework," *J. of Syst. and Softw.*, vol. 85, 2012.

[18] G. F. Lyon, *Nmap netw. scanning: The official Nmap project guide*. Nmap Project, 2009, ch. 6. [Online]. Available: https://nmap.org/book/performance.html