True Attacks, Attack Attempts, or Benign Triggers? An Empirical Measurement of Network Alerts in a Security Operations Center

Limin Yang[1*], Zhi Chen[1*], Chenkai Wang[1], Zhenning Zhang[1], Sushruth Booma[1], Phuong Cao[2], Constantin Adam[3], Alexander Withers[2], Zbigniew Kalbarczyk[1], Ravishankar K. Iyer[1], Gang Wang[1]
[1]University of Illinois Urbana-Champaign [2]NCSA [3]IBM Research

Abstract

Security Operations Centers (SOCs) face the key challenge of handling excessive security alerts. While existing works have studied this problem qualitatively via user studies, there is still a lack of *quantitative understanding* of the impact of excessive alerts and their effectiveness and limitations in capturing true attacks.

In this paper, we fill the gap by working with a real-world SOC and collecting and analyzing their network alert logs over 4 years (115 million alerts, from 2018 to 2022). To further understand how alerts are associated with true attacks, we also obtain the ground truth of 227 successful attacks in the past 20 years (11 during the overlapping period). Through analysis, we observe that SOC analysts are facing excessive alerts (24K–134K per day), but only a small percentage of the alerts (0.01%) are associated with true attacks. While the majority of true attacks can be detected within the same day, the post-attack investigation takes much longer time (53 days on average). Furthermore, we observe a significant portion of the alerts are related to "attack attempts" (attacks that did not lead to true compromises, 27%), and "benign triggers" (correctly matched security events but had business-justified explanations, 49%). Empirically, we show there are opportunities to use rare/abnormal alert patterns to help isolate signals related to true attacks. Given that enterprise SOCs rarely disclose internal data, this paper helps contextualize SOCs' pain points and refine existing problem definitions.

1 Introduction

A Security Operations Center (SOC) is an organization's central unit responsible for detecting, analyzing, and responding to security threats [82]. A key challenge faced by SOCs is to locate signals of *true attacks* from the noisy, unfiltered, and uncorrelated security alerts [4]. Evidenced by a real-world example, during the data breach of Target [26], security alerts were triggered before the attack, but the alerts were unheeded (missed) by the SOC which failed to stop the attack in time.

Recently, researchers have conducted user studies with SOC members to understand what affects the efficiency and effectiveness of SOCs [4, 20, 24, 37, 69, 70]. The results echo that the overwhelming number of alerts generated by security monitoring tools is a common concern [4, 37], which often leads to alert fatigue or burnout of SOC analysts [70, 71]. While such user studies provide a qualitative view of people's perceptions, there is still a lack of *quantitative understanding* of the impact of the excessive alerts, and their effectiveness in capturing true attacks.

A key challenge for quantitative measurements, however, is to obtain real-world data due to its sensitive nature. While a few works have included SOC logs in their study [7, 12, 61, 75], these datasets only cover a short period of time (e.g., weeks). More importantly, the data is only used for "black-box" evaluations of their proposed detectors—often cases, researchers did not (cannot) provide analyses of the data itself. The most related paper is a recent study on a private rule engine from a security company [76] to report statistics of rule changes and false alerts. However, due to privacy concerns, it did not provide detailed analyses of specific (successful) attacks within its client networks (or how these true attacks are detected and investigated).

Research Questions. In this paper, we fill this gap by empirically analyzing the *network logs within a real-world SOC* to explore the answers to the following research questions. *First*, what are the key bottlenecks in the SOC for threat detection, and how do attackers exploit/respond to such bottlenecks? *Second*, how excessive are the security alerts, and what are the common reasons behind the alert triggering? *Third*, how effective are the alerts to correlate or indicate true/successful attacks?

To answer these questions, we obtained a network alert dataset from the SOC of the National Center for Supercomputing Applications (NCSA) via collaboration (approved by IRB). The alert dataset contains 115,617,526 alerts fired by the SOC's network monitoring tool in the past 4 years (2018–

*The authors contribute equally to this paper (co-first authors).

2022). To further correlate network alerts with true attacks, we obtained "ground truth" incident reports from the SOC's forensics analysis. The ground-truth set includes detailed forensics reports of 227 true attacks[1] in the past 20 years (2002–2022), and 11 of them occurred in the overlapping period (2018–2022). We focus on network monitoring tools since they serve a key role in analyzing attackers' reconnaissance, compromise, and data exfiltration activities.

Key Findings. Our analysis returns several key findings. *First*, we observe that *humans* are still the bottleneck, which is reflected in various aspects of the SOC workflow. For instance, the majority of the true attacks (65%) took *more than one analyst* to investigate. While most of the true attacks were detected on the same day (66%), the post-attack investigation (primarily done manually) can take 53 days on average.

Second, our measurement shows that SOC analysts are facing excessive alerts (24K–134K per day), but only a small percentage of the alerts (0.01%) are associated with true attacks. A significant portion of the alerts is related to "attack attempts" (at least 27%) that failed to cause damages. In addition, a large portion of alerts (49%) are benign triggers [37] and can be safely ignored (i.e., correctly matched security events with business-justified explanations[2]). We argue that such "attack attempts" and "benign triggers" are essentially noises in the data labels, and therefore pose a major challenge to learning-based systems that aim to detect true attacks. To the best of our knowledge, no public benchmark datasets are actively measuring and modeling "attack attempts" and "benign triggers" beyond simply classifying network traffic as malicious or benign.

Third, by associating alert logs with true attacks, we find that while the excessive alerts (false positives) are problematic, the *false negative* problem is also concerning. Relying on network monitoring alone has limited capability in capturing the initial host compromise or stealthy post-attack activities. For instance, 4 out of 11 true attacks investigated cannot be detected by the network alerts. In addition, we empirically show there are opportunities to use rare/abnormal alert patterns to help isolate true attacks from other alerts.

Our analysis has positively impacted operational rules at the SOC (e.g., adding new routing policies, §7.3). More importantly, in §8, we discussed key gaps in existing SOC practices (e.g., log linkability, opportunities for automation), and made recommendations to current and future ML-based research that use SOC data to train detectors.

In summary, we present an empirical measurement of network alerts within a SOC based on a dataset of 115 million network alerts over four years. By associating alerts with true attacks, we quantify the impact of excessive alerts and measure the effectiveness of alerts to indicate true attacks. While acknowledging the limitations and biases of the data (i.e., data from a single SOC), we discuss the generalized lessons from our work and provide recommendations for future research on SOC tooling (especially ML-based approaches). To facilitate future research and encourage result reproduction, we release our code and a sample of the alert dataset [78].

[1] We define "true attacks" as successful compromises that lead to damage to the target network. This is different from "attack attempts" which refer to malicious attacks that failed to achieve their goals.

[2] An example can be alerts fired correctly due to a vulnerable Java version, but the SOC chooses to ignore it due to legacy systems [4].

2 Background and Related Work

Security Operations Center (SOC). SOC is an in-house or outsourced unit responsible for monitoring information technology (IT) infrastructures and responding to security incidents [4, 37]. SOCs have three important aspects: people, processes, and technology [4, 82]. First, people: SOCs have a team of security professionals with various duties and roles (e.g., threat hunters, security analysts, and team managers). Second, processes: SOCs involve various workflows in their routine tasks such as vulnerability scanning and handling alerts from Intrusion Detection Systems (IDS) or Security Information and Event Management (SIEM) systems. SOCs typically maintain a *playbook* [69] to document a standard procedure for communicating about and escalating alerts. Third, technology: SOCs may deploy various technical tools to monitor the network and hosts to detect attacks [4, 16, 42, 53, 57, 80].

User Studies with SOC Members. Researchers have studied various challenges faced by SOC analysts by conducting *user studies*. Existing studies have explored SOC analysts' perceptions of IDS usage [24], effectiveness of playbooks [69], system misconfigurations [3], malware detection strategies [1], SOC performance evaluation metrics [1], the "burnout" issues of analysts [70], and how analysts resolve contradictions between people and tools [72].

In particular, Kokulu et al. [37] conducted an interview with 18 SOC members who expressed concerns about the high volume of unfiltered/uncorrelated alerts. However, surprisingly, participants were not concerned about "false positives" (FPs) of the detection tools. Alahmadi et al. [4] further interviewed 20 SOC members to explore the reasons. They revealed that a common perception is that the majority of the "false positives" are caused by *benign triggers*. Benign triggers are correctly fired alerts but can be explained by legitimate behaviors in the organization's context (and thus are safe to ignore).

Our study complements and advances existing literature (which are primarily qualitative user studies) by providing a *quantitative* lens into the problems of security alerts and attack investigation using real-world SOC logs. Also, with empirical data across multiple years, we provide a longitudinal view of how these problems evolve over time.

SOC Tooling. Researchers have proposed technical solutions for threat detection and forensics for SOCs. These solutions broadly include network intrusion detection systems (NIDS) [34, 59, 68, 77, 79], host-based IDS [15, 21, 25,

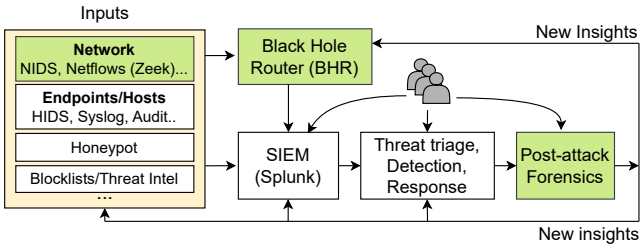


Figure 1: **SOC Workflow**—The highlighted green boxes are the main data sources for our paper. SIEM stands for "Security Information and Event Management."

27, 43, 56], alert triangle [5, 11, 12, 28], verification [38], correlation [61, 63, 66, 75, 80] and prioritization [30, 40] methods, provenance-based threat hunting systems [8, 49], and attack prediction models [7, 67]. The goal of our work is not to develop specific tools but to use real-world data to understand the gaps between existing tools and the needs of SOCs.

A recent paper studied the rule engine used by a Managed-Security-Service-Provider (MSSP) [76] to analyze rule changes and (false) alerts triggered. The key differences are: the prior study [76] was focused more on the relationship between alerts and rule sets, whereas our study was focused on the reasons for alert triggering (distinguishing benign triggers, true attacks, and attack attempts). In addition, [76] did not provide a detailed analysis of successful attacks within the target networks/SOCs, possibly for privacy reasons. Our paper provides a complementary view by diving deeper into one network to analyze its successful attacks and correlate them with alerts.

3 Datasets

We obtained and analyzed the data by partnering with the cybersecurity team at the National Center for Supercomputing Applications (NCSA), located at the University of Illinois Urbana-Champaign. NCSA has been supporting open science research by hosting the first sustained-petascale supercomputer, Blue Waters; and now operates the DeltaAI supercomputer supporting a wide range of workloads for scientific discoveries. The center has thousands of physical nodes, tens of thousands of computing cores, and 10+ petabytes of storage interconnected internationally via a terabit network. The SOC monitors the network/system and user activities leveraging a wide spectrum of internal, open-sourced, and commercial tools to detect and respond to security incidents. Figure 1 shows a simplified view of the process. The SOC analysts use a Splunk dashboard for Security Information and Event Management (SIEM) [41], which provides a real-time view of security events collected by network monitoring tools, system logs, Linux kernel audit logs, and other internal and external data sources (e.g., blocklists, honeypots, threat intelligence).

Network Alert Dataset (2018–2022). Our preliminary dataset contains the *network-level* alerts fired between April 2018 and July 2022 by the SOC's network monitoring tool Zeek (a widely-used open-source tool [81]). As shown in Figure 1, the SOC maintains a customized rule engine that analyzes a variety of network and system logs to detect events that match the pre-defined rules. The network monitoring tool is one of the main sources of alerts. There are other alerts from endpoint/host-IDS, partner sites, and other commercial tools, to which we don't have access. In other words, the dataset represents a lower bound of alerts from this SOC.

Term	Definition
True attacks	The attacker successfully compromised the system and/or caused damages
Attack attempts	The attacker tried but failed to compromise the system or cause damages
False positives	Benign activities that are incorrect determined as security attacks
Benign triggers	Correctly matched security events but have business-justified explanations [4]

Table 1: **Definitions**—Key terms used in the paper.

In total, the dataset contains 115,617,526 network-level alerts. Each alert is characterized by a timestamp, involved IPs, an alert category, and other metadata. Starting from 2020, the SOC implemented a Black Hole Router (BHR) [39] (see Figure 1) to automatically handle a subset of these Zeek alerts without human involvement (e.g., blocking IPs associated with the alert) in real-time. As such, our dataset contains the BHR label from 2020 to 2022.

Ground-Truth: True Attacks. To understand how effective these network alerts are in informing the SOC of true attacks, we also gather data about the ground truth of successful attacks and compromises in the past two decades. This dataset is from "post-attack forensics", as illustrated in Figure 1. The definition of *true attack* (see Table 1) is based on whether the attacks have caused major degradation/interruption to system operation or whether there was evidence of data leakage or system compromise. This dataset contains 227 true attacks from January 2002 to March 2022. Among them, 11 occurred during the overlapping period with the alert dataset (2018–2022). For each attack, SOC analysts collected a set of attack artifacts (e.g., logs, binaries, VM images), conducted a detailed investigation, and produced a report describing their investigation process and conclusions. The reports were written in an unstructured format, describing details about the user accounts, IP addresses, or machines related to and/or responsible for the attacks.

Is the Dataset Biased? We believe the answer is "yes." Given the dataset was collected from a single SOC, we do not expect the result to generalize to all SOCs. Instead, our goal is to present an in-depth case study for a SOC-type that is complementary to (different from) other existing studies. This SOC's network has thousands of servers with 17,000+ active users/researchers, and it represents a typical moderate/large-

sized research facility. The SOC has 9 members, which is among the most common SOC team sizes (2–10) [64]. Note that this SOC is SOC-2-Type-2 [2] certified, which testifies to the legitimacy of its security infrastructure. The SOC's network is *segmented* where certain parts are heavily guarded while other parts are configured as "open networks." The open-network setting allows both users within the research center and researchers outside of the network to access the computation nodes. This presents a unique contrast with enterprise/private networks that usually have a strict firewall to secure the perimeter (i.e., the access to internal resources is usually not openly shared with external parties). In this open network, both malicious and legitimate connections may come from anywhere in the world, making it difficult to flag attacks. Such open-network configuration is common for open research infrastructures. The SOC may not represent those of government agencies or large companies, but one useful context is this research institution not only supports scientific computing but also runs government/military applications. The incidents observed by this SOC are diverse because it not only monitors open networks (recording popular incidents such as volumetric DoS, password guessing, or database ransoms) but also monitors special network segments of closed systems that work with highly regulated industry partners (HIPAA in healthcare) and CUI data with national labs (recording sophisticated incidents such as zero-day exploit privilege escalation and lateral movement to national labs).

To better interpret our results, we note the following limitations of this data. First, the data is from a specific SOC, and thus the result may be biased due to the specific SOC type, size, certification, and network characteristics. These factors should be considered for our result discussion (e.g., in §8). Second, the alert dataset is focused on network alerts which is a lower bound of all alerts in the SOC. The results and observations may not be applicable to host-level alerts or Endpoint Detection and Response (EDR) systems.

What's the Value of Analyzing Private Data? Considering that real-world SOCs rarely share their internal attack details, we believe the data can provide a unique opportunity to quantify the problem related to alert filtering and investigation and study the longitudinal changes of a SOC's threat investigation practice. Based on our analysis, we believe some of the observations and findings have broader/generalized implications, which will be further discussed in §8.

A key challenge of analyzing private datasets is the *reproducibility of results*. This is a long-standing challenge to the community. We believe this type of analysis is beneficial with strong evidence demonstrated by prior works. For example, our community has published papers based on private data (from a single organization) [22, 30, 31, 74, 76], providing new insights into established problems, revealing emerging threats, as well as challenging existing false beliefs and assumptions.

To mitigate the concerns about reproducibility, we plan to publish all the code developed in the project. We also have worked with the SOC and obtained the approval to release a sample of the network alert dataset. Details are in §8.3.

Ethical Considerations. The study was reviewed and approved by our IRB. Given the sensitive nature of the data, the dataset has been stored on the SOC's internal server and the researchers can access the server to perform analysis. One SOC member oversees the data analysis process to ensure compliance with the SOC's policies. The log data does not contain user-identifiable information except for the IP addresses. For most analyses, we used the hashed IPs (accessible by the whole research team). Only researchers who worked closely with the SOC team can access the IPs with the last octet removed, which is only for case studies.

Positionality Statement. Throughout this project, we carefully reflected on our position as researchers and inspected how our backgrounds may have influenced the analysis and results. The authors of this paper have the collective knowledge and expertise in network/system security research, security management and operations, incident response, and machine learning. The team is diverse, including both academic researchers and industry practitioners, and one of the co-authors is affiliated with this SOC. We acknowledge that our background and position may introduce biases. For example, during *manual investigation* of a given attack (and their alerts), the investigation process can be influenced by the team member's knowledge of the target SOC and their prior experience with similar attacks. Another example is related to *benign trigger* identification (see §6.4) which is highly dependent on the knowledge of the SOC's alert system and the organizational context.

Roadmap. In the following, we start our analysis with *a case study* of one true attack by associating attacker steps with the triggered alerts (§4). This is to illustrate how the dataset can be used for such analysis. Then we perform a more systematic analysis of true attacks (§5), triggered alerts (§6), and the attack-alert association (§7).

4 Case Study: Postgres Compromise

In November 2021, an attacker successfully compromised an internal host through a misconfigured Postgres (PostgreSQL database management system). We analyze the incident report, and the alert dataset to recover the attack process in Figure 2.

Attack Process. On the day before the actual attack, the attacker made several attempts to connect to different hosts (ports 8081, 9001, and 9999) without success. Eventually, at 03:42 AM of November 10, the attacker found a vulnerable port (5432) that allowed the attacker to connect to the victim server *D* via Postgres. Immediately at 03:44 AM, the attacker downloaded three files to server D (at least one of the files was later confirmed to be malicious). After the server was compromised, the attacker used the compromised host to scan external IPs. They first scanned at least 250 hosts on port

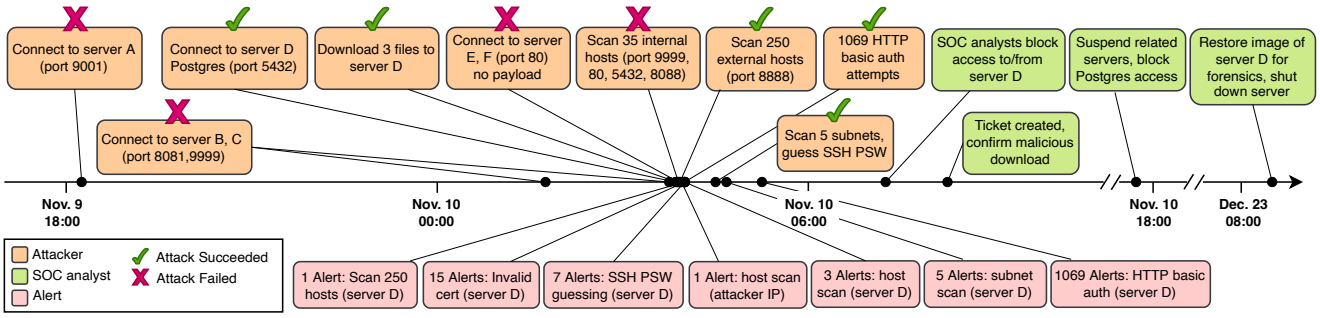


Figure 2: **Postgres Case Study**—The attacker successfully compromised an internal server D using a Postgres vulnerability. After the compromise, server D was used to scan and perform password guessing against external hosts.

8888, then attempted 1,069 HTTP basic access authentications. Meanwhile, they also scanned 110 subnets, and run password guessing over 5,164 connections. The scanning activities started shortly after the host compromise until 07:11 AM (when the SOC took the host offline).

During 03:37–03:59 AM, the attacker attempted to find other vulnerable hosts by scanning (at least) 35 hosts within the organization's network on ports 5432, 9999, 80, and 8088. These scanning activities were picked up by Zeek and then were automatically blocked by BHR.

Alerts Triggered. No alert was triggered before or during the host compromise. The first alert was sent at 03:44 AM because the victim host scanned 250 unique hosts on port 8888 within 5 seconds. Between 03:44 and 07:11 AM, there were 1,069 alerts fired on HTTP basic access authentication, 15 alerts on invalid certification, 7 alerts on SSH password guessing, 5 alerts on subnet scanning, and 4 alerts on host scanning. In total, 1,101 alerts were triggered including 1,100 alerts on host *D* and 1 alert on the attacker IP. The alert on the attacker IP was triggered at 03:59 AM when the attacker scanned 35+ unique internal hosts.

Our analysis confirms that the number of alerts on host *D* is abnormally high. As a reference, we compute the average number of alerts per host within a 4-hour window during 12/2020–07/2022, which is only 0.14 with a standard deviation of 12.65. Host D has 1,100 alerts during the 4-hour attack window, which is more than three standard deviations away from the mean value.

SOC Response. Although the alerts were triggered shortly after the compromise, it took some time for the SOC analysts to effectively respond to the attack. At 07:11 AM, an analyst recognized the incidents and immediately blocked access to and from the victim host *D*. Around 08:10 AM, one of the downloaded files was confirmed to be malicious through third-party vendors. During the day, the SOC analysts extracted the attacker IPs from the log. Around 06:00 PM of the same day, the SOC blocked external access to Postgres for certain servers. It took more time to gather artifacts (logs) and recover the snapshot of server *D* (which requires assistance from other teams). A month later (December 23), the snapshot was created for forensics analysis. The post-attack investigation has a long delay (which is common, see §5).

Root Causes. While the Postgres server running on the victim host was publicly accessible, the root cause was misconfigured authentication. One of the usernames had no configured password and the attacker was able to use this username to gain access and then download malicious files. It is a combination of improper security practices of users (open Postgres, a lack of authentication) and SOC issues (insufficient internal scanning/screening).

Attacker IP Analysis. We further searched the attacker IP in the network connection log for the entire year of 2021 to examine whether the attacker has attempted to connect to any internal hosts *before* the attack date. We find that, about *one month before the attack*, the attacker made five separate scans (on at least 175 hosts, including the later compromised host) on five different days (which raised 5 alerts). The result indicates that the alert system may have captured the early attacker activities before the actual attack but the signals are not strong enough to raise attention.

Remark 1: This case study has two takeaways: (1) While the alert system have helped to capture the attack, it still missed the initial host compromise. If the attacker had chosen to stay dormant after the compromise, it could be difficult to detect. (2) The attackers made multiple attack attempts way before the actual attack, which presents an opportunity for early interventions.

5 Ground Truth of True Attacks

The case study above (§4) is only based on one true attack. In this section, we analyze the broader set of *ground truth* of true attacks based on the forensics reports. As shown in Table 1, our definition of *true attacks* refers to those that lead to successful compromises or damage to the target network. This differs from *attack attempts* that fail to achieve the attack goals. Through this analysis, we examine what types of attacks have led to successful compromises, and the key bottlenecks in the threat detection/investigation process.

Available Info in Report	# of Reports (%)
Total	227 (100%)
Break-in Method Identified	178 (78%)
Consequence Identified	203 (89%)
Attacker IP(s) Identified	151 (67%)
Attack Start Time Identified	123 (54%)
Attack Discovery Time Identified	167 (74%)
Involved Analysts Identified	107 (47%)

Table 2: **True Attacks**—Attack reports where certain attack details were recovered and documented.

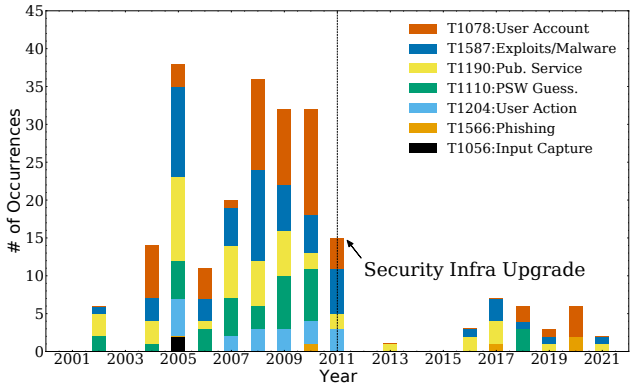


Figure 3: Occurrences of MITRE Technique IDs per year related to the break-in methods.

Given that reports are largely unstructured, we have two researchers manually annotate the reports which took 1–2 months to complete. Due to space limit, we describe the detailed coding method in Appendix A. For each attack, we extract the attacker/victim IPs, attack start time (t_1), attack detection time (t_2), and investigation end time (t_3). Here, "attack start time" is the earliest timestamp of the attacker activity that is directly associated with the compromise. The determination was made by the SOC analysts based on their investigation. We also report the number of analysts involved in the investigation (if the information is available). For each attack, we refer to the MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) Framework [51] to annotate the high-level techniques of attackers. For brevity, we only identify the MITRE techniques that best describe (1) how the attacker broke into the system, and (2) the attack consequences. Not all incident reports are conclusive and contain the full attacker information. As shown in Table 2, for 78% of the true attacks, the SOC analysts reached the conclusion about the attackers' break-in methods. For 73% of the attacks, the reports had information about the attack consequences. The attacker IP(s) were identified in 67% of the reports. Also, the timing information (e.g., when the attack started) was not always easy to confirm (available in 54% of the reports).

True Attacks Over Time. Figure 3 shows the occurrences of MITRE tags from 2002 to 2022 for the break-in methods. A similar figure is plotted for "attack consequences" in the supplementary materials [78]. Overall, the most common break-in method is using (stolen) credentials from existing valid user accounts (T1078), followed by using malware or exploiting vulnerabilities (T1587). For attack consequences, the most common consequence is the compromise of servers/hosts (T1584). Also, attackers may steal more user credentials (T1586), perform denial of service attacks (T1498), and use the compromised servers to scan other hosts (T1595). More detailed results are in the supplementary materials [78].

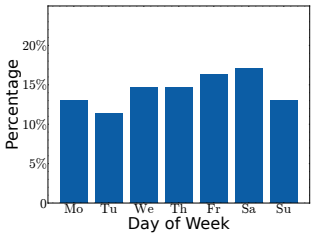


Figure 4: Day of week for *attack start* time.

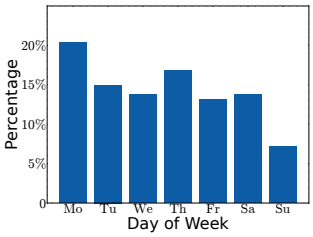


Figure 5: Day of week for *attack discovery* time.

We find that the overall attack trend correlates with the known changes of the SOC infrastructure. Before 2011, the SOC's security infrastructures/policies were not well established. During 2010–2011, the SOC had a major upgrade on its security infrastructure by adding new capacities in logging, monitoring, and threat response, recruiting new members to the SOC team, and implementing two-factor authentication (2FA) for accessing the computing nodes. This may correlate to the different number of true attacks before and after 2010. However, we want to emphasize that this should only be interpreted as a correlation, and does not necessarily reflect causality [9]. There are other possible explanations such as the underlying changes in attack trends, and the improved cooperation proficiency after reorganization. We don't have the data for a full causal analysis to rule out other factors.

Analysts Involved. As shown in Table 2, 107 of the reports have clearly documented the analysts involved in the investigation. Using this set, we find that the majority (65%) of attacks need two or more SOC analysts to work together for attack investigation. This echoes prior works that describe the SOC pipeline as a collaboration process [24]. Some of the attacks involved a large number of people (e.g., 6–7), including people outside of the core SOC team at other institutions (e.g., network administrators) who helped to provide extra context information about the attack.

Time took for Post-Attack Analysis. While it is generally quick for the SOC to identify most of the attacks (e.g., 66% are within the same day, see Figure 15), it takes a long time to investigate and understand the attack. If we compute the investigation time (only 28 attacks after 2016 had related timestamps), the average post-attack investigation time is 53.2 days with a maximum of 253 days.

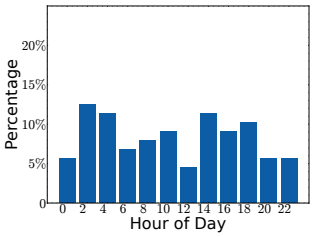


Figure 6: Hour of day for *attack start* time.

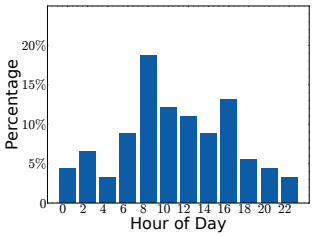


Figure 7: Hour of day for *attack discovery* time.

Attack during Off-Hours. We further examine the "day of the week" pattern for *attack start time* and *attack discovery time*, respectively. We consider all the true attacks that can provide corresponding timestamps, and all timestamps are based on the SOC's local time. As shown in Figure 4, the attack start time is fairly evenly distributed, slightly skewing to Saturdays. Figure 5 shows the attack discovery time by the SOC, which indicates that Sundays are the least common days to detect attacks. In contrast, Monday is the most common day to discover attacks when the full team is available.

Figure 6 and Figure 7 further illustrate the "hour of the day" pattern. This analysis needs more fine-grained timestamps (hour-level), which further limits the number of qualified attacks (88 and 91, respectively). The data shows that more attacks *started* during the off-hours of the SOC (i.e., midnight to 8 AM, and then 6 PM to midnight) than during the working hours. Note that we don't have evidence that attackers *intentionally* target the SOC's off-hours. An alternative explanation is that certain attackers' working hours happen to overlap with the SOC's off-hours, due to time zone differences. In Appendix A, we attempt to convert the attack start time based on the attacker IP's local time to provide further insights. In contrast, most of the attacks were *discovered* by the SOC during working hours (peaking at 8–10 AM). While the SOC operates 24/7, there are more people available during working hours than off hours. Overall, the result suggests that attack discovery is more aligned with human analysts' working hours. As such, attacks that happened during SOC off-hours can have extra delays for detection.

Remark 2: Human analysts are still the bottleneck: (1) it usually takes more than one analyst to work on a single attack; (2) Attack detection is more aligned with the SOC analysts' working hours, which can cause delays to attack discovery.

6 Network Alerts

In this section, we explore how alerts are fired by the network monitoring tool and answer the following questions: How often are the alerts associated with malicious activities? To what extent can the SOC analyze and take actions on these alerts automatically? What are the common reasons behind the fired alerts and the challenges to reason the alerts?

Time Range	Days	# Alerts	Alert/Day	Uniq. IPs
04/18–08/20	751	101,187,992	134,738	17,364,456
12/20–07/22	578	14,429,534	24,965	1,869,671
Total	1,329	115,617,526	86,996	19,040,194

Table 3: **Alert Dataset**—The alert dataset covers two time periods. During the first period, the SOC did not have an auto-block mechanism yet. During the second period, the auto-block mechanism (BHR) helped to automatically block IPs that triggered certain alerts (6,474,204 blocked alerts, 44.9%).

6.1 Excessive Alert Volume

Table 3 summarizes the alert dataset which covers 1,329 days and contains 115,617,526 alerts (about 86,996 alerts per day). Figure 8 plots the number of daily alerts (in millions) over time. We observe that, during the first period (April 2018 to August 2020), the average number of daily alerts is 134,000, which is 5-6 times higher than the 24,000 alerts per day in the second period (December 2020 to July 2022). In particular, there was an abnormally large number of alerts during early 2018, when the network was faced with intensive address scanning from external IPs (from a new variant of Mirai botnet). Also, major peaks showed up in early 2022 which were caused by the scanning for the Log4j vulnerability[1] and alerts fired by internal logging systems. The excessive number and the high variance of daily alerts make staff planning and workload allocation a challenging problem.

Automated Alert Handling. In the second period, the SOC implemented a Black Hole Router (BHR) [39] and firewall rules to automatically handle a subset of alerts without human involvement in real time (e.g., blocking IPs associated with the security alert). 6,474,204 alerts are handled by BHR during the second period (44.9%) without involving human analysts. Many of these alerts are associated with attack attempts that match high-confident signatures and some may involve known benign triggers (e.g., scanning from third-party security vendors, see §6.2). Even so, after BHR filtering, there are still a large number of alerts that need to be handled and processed by analysts (about 13,764 alerts per day).

Remark 3: About 44.9% of the alerts can be handled/blocked automatically without human involvement. However, after auto-block, there is still a large volume of daily alerts that cannot be handled automatically.

[1] Log4j is a high-severity zero-day vulnerability that involves remote code execution in Log4j (disclosed in November 2021) [18].

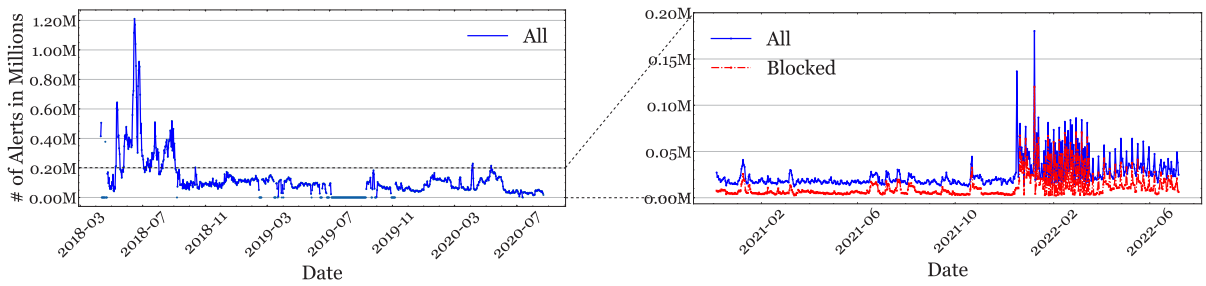


Figure 8: **Number of Daily Alerts**—We plot the two time periods separately under different y scales. Note that the first period (2018–2020) has a significantly higher number of daily alerts than that of the second period (2021–2022).

6.2 Complex Reasons for Triggering Alerts

Next, we use the alert datasets to understand the reasons why alerts are triggered. For this analysis, *we focus on the second period (2020–2022)*, because (1) the rule sets are more optimized to reduced alerts; and (2) only the second period has auto-blocking BHR, which is important for identifying attack attempts. At a high level, we categorize the reasons into several categories (Table 4).

True Attacks. True attacks refer to incidents where the attacker successfully compromised the system and/or caused damages. Since successful attacks are rare (see §5), the number of alerts is also small (1,114, 0.01%). Further analysis is presented in §7.

Attack Attempts. Attack attempts are those that (1) were initiated by malicious attackers, but (2) did not manage to compromise the system or cause damage. For our analysis, we rely on the auto-blocking mechanism (BHR) to flag ground-truth attack attempts. Recall that BHR is designed to be "conservative" to only block attack attempts that match high-confidence signatures/patterns. Through our analysis, we find that not all the BHR alerts represent attack attempts (many of them turned out to be benign triggers, see §6.4). In total, 44.9% of the alerts are BHR-blocked—after excluding benign triggers in them (about 17.50%), we located 27.37% alerts that are associated with attack attempts (see §6.3).

Benign Triggers. Alerts that are neither "true attacks" nor "attack attempts" cannot be simply marked as false positives (FP). Many of them can be benign triggers. Benign triggers [4] are *correctly fired alerts*, explained by business-justified, acceptable behaviors in the organization's environment. As such, analysts may choose to ignore them. An example would be an alert fired correctly because a weak SSL key is detected, but SOC can ignore the alert knowing the host is retiring and disconnected from the public network. This definition is different from the generic "false positives" which usually refer to alerts that are *incorrectly* fired on benign behaviors (e.g., due to bad rules, errors, or misconfigurations). Identifying and attributing benign triggers is a non-trivial task. We found evidence that 48.91% of the alerts had benign triggers, and we detail the challenges encountered in §6.4.

Alert Type	# of Alerts	Percentage
True Attacks	1,114	0.01%
Attack Attempts	3,948,645	27.37%
Benign Triggers	7,057,012	48.91%
Unknown	3,422,763	23.72%
Total	14,429,534	100%

Table 4: **Reason of Alerts**— We categorize different reasons for triggering alerts during the period of 12/2020–07/2022.

Unknown. The remaining alerts (23.72%) are marked as "unknown" since we cannot easily/confidently determine their causes. They could be a mixture of attack attempts, unknown benign triggers, false alerts, or even true attacks that are missed by the SOC team. We will further analyze them by statistically comparing them with other alert categories, especially true attack alerts, in §7.2.

6.3 Understanding Attack Attempts

First, we focus on the 3,948,645 alerts from the second period (2020–2022) that are associated with the attack attempts blocked by BHR, to characterize attacker behaviors.

Geolocation of Attacker IPs. We start by analyzing where the attacker is coming from. For this analysis, we identify *external* attackers that are blocked by BHR, i.e., the source IP is an external IP (outside of the research center) and the destination IP is an internal IP of the research center. The blocked external IP is then labeled as an *attacker IP* and the internal IP is labeled as the *victim IP*. In total, we identified 721,169 attacker IPs. By mapping the IPs to their country/region codes (using Maxmind database [46]), we find that the attacker IPs are from all over the world (223 countries/regions). The top five countries are China, the US, India, Brazil, and Germany (see more details in the supplementary materials [78]).

Types of Attacks. Table 5 lists the top five alert categories associated with the blocked attack attempts. We observe that the vast majority of the blocked attack attempts are related to network scanning (3 out of 5 categories, 96.7% of the alerts). The other 2 out of 5 are related to brute-force SSH password guessing and suspicious SSH clients.

Alert Description	Count (%)
Address Scan	3,368,776 (85.3%)
Random Scan	384,247 (9.7%)
Port Scan	63,676 (1.6%)
SSH PSW Guess	58,879 (1.5%)
Bad SSH Client	40,747 (1.0%)

Table 5: **Top 5 Alerts**—Descriptive categories for the top 5 alerts since we cannot present the exact alert name from the SOC.

Setting	# Alerts	# Alerts w/ Victim Identified	Uniq. # of Victim IPs
Before IP Recovery	14,429,534	7,531,038	18,115
After IP Recovery	49,478,207	37,378,465	69,356

Table 6: **Victim IP Recovery**—For network scanning related alerts, the victim IP is not directly available in the alert due to alert aggregation. We recover victim IPs from net connection logs.

Victim IP Recovery. To further analyze attacker behavior, we need to map the attacker-victim pairs based on the alert. Unfortunately, to reduce the volume of *scanning alerts*, Zeek has suppressed and aggregated multiple scanning alerts from the same attacker IP into a single alert (as a summary). For example, if an attacker scanned 200 IPs, a single alert is recorded in the log to summarize the activity *but the list of 200 scanned hosts is omitted* from the alert. This creates difficulties for attributing the victim hosts. As shown in Table 6, among the 14.4 million attack-attempt alerts, we can only identify victim IPs for 7.5 million (52%).

To recover the victim IPs, we worked with the SOC to correlate the alert data with network connection logs. The network connections logs are several orders of magnitude larger than the alert log, which can only be stored for two years. As such, we can only perform this recovery analysis for 2020–2022 period. Given an alert, we search the connection log of the same day to identify connections with a matching source IP (attacker IP) and the destination port for each alert. This means a single alert summary will be expanded to multiple alerts (one for each victim host). As a result (see Table 6), the number of alerts is increased to 49 million, and the number of victim IPs is increased from 18K to 69K. This recovery method is not necessarily sustainable because the connection logs will be deleted after two years due to its enormous storage requirement. We will use this extended dataset for the following analysis.

Remark 4: The SOC applies alert suppression and aggregation to reduce alert volume; however, this process also leads to information loss (e.g., victim IPs, timing), which creates difficulties for attack analysis and victim attribution. Improvements are needed to preserve key metadata during alert aggregation.

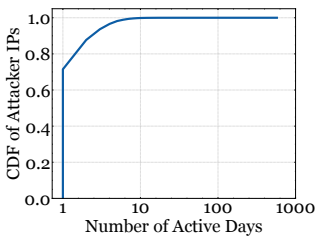


Figure 9: # of active days per attacker IP.

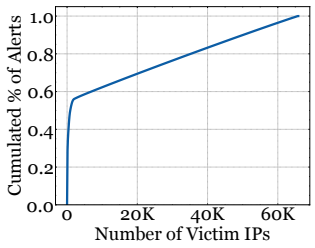


Figure 10: Top victim IPs vs. accumulated % of alerts.

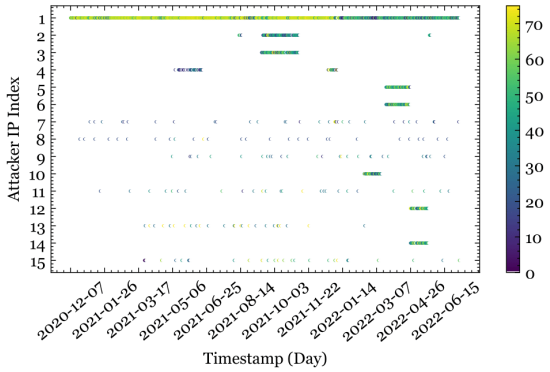


Figure 11: **Activities of the Most Active Attackers over Time**—Top 15 attackers with the highest number of active days. Each row (y-axis) represents one attacker. The color of the dot represents the number of alerts from this attacker for a given day.

Recurrent Attackers. Next, we examine how persistent the attackers are. In other words, once the attacker IP is blocked by BHR (for some period), how likely will they make more attack attempts? Here, for each attacker, we define their *active days* as the number of days when they made attack attempts. Figure 9 shows the Cumulative Distribution Function (CDF) for the number of active days per attacker IP. We observe that about 75% of the attacker IPs are only active for one day, indicating they are short-lived. However, a small portion of them (0.3%, 28 IPs) have more than a week of active attempts, indicating a high level of persistence.

To understand the activity patterns of top attackers, we plot the heatmap in Figure 11 for the top 15 most active attackers (that had at least 26 days of active attack attempts). The top-1 IP is making active attempts for 573 days. After investigation, we discover that this is an (undocumented) external scanner set up by one SOC member to test the BHR system. Other IPs are attacker IPs. Some attackers (e.g., ID=2, 3, 4, 5, 6) had continued with their attack attempts for a concentrated period of time (e.g., several weeks or even months) before stopping. Other attackers (e.g., ID=7, 8, 9) spread their attack attempts across time, and persistently return after certain time gaps.

Uneven Alert Distribution of Victims. Finally, we find that not all victim IPs are evenly targeted. In Figure 10, we sort the victim IPs based on the number of associated alerts

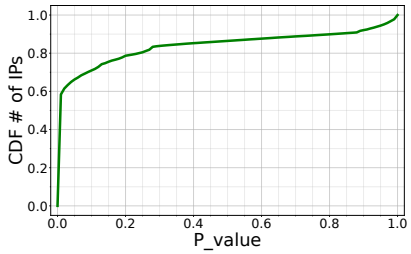


Figure 12: ***P*-value from Hypothesis Test to Detect Bots** — The null hypothesis is that the attack connection occurred at random intervals. A lower p value allows us to reject the null hypothesis with statistically significant confidence. $p < 0.05$ indicates regularity of connection timing, which is a strong indicator of bots.

on the x-axis and report the accumulated percentage of alerts on the y-axis. We observe that a very small portion of victim IPs (1,576, 2.3%) contribute 55% of the total alerts. Then the rest of the IPs (67,780, 97.7%) contributed the rest of 45% of the alerts. This represents an uneven distribution of alerts across hosts. We confirmed with SOC that some of these most-targeted hosts were running services with open access via the public Internet (i.e., more alerts). The skewness is also reflected by the number of alerts per host—while the maximum number is 388,502 alerts, the median is only 203 alerts per host (over 2 years). Prior works have proposed per-host models for anomaly detection or alert/event prediction [10, 14, 32, 67]. The skewness of the event distribution can create challenges for such models.

Botnets vs. Manual Attackers. A particular question of interest is whether the attack attempts are made by human attackers or botnets. This is a difficult question to answer for two reasons. First, many attack attempts are "short-lived" (or quickly blocked by BHR; see Figure 9), resulting in insufficient data for our investigation. Second, manual attackers may still use attack software which may exhibit signals of automation. For this analysis, we consulted a SOC analyst and decided to rely on known heuristics of botnets, i.e., the time intervals of botnet connections often exhibit a high level of regularity. As such, we formulate attackers' connection time intervals into time series, and run a statistical test to determine their randomness. We select the Ljung–Box statistical test [44], which is widely used in time series analysis. Ljung–Box test helps determine whether the autocorrelation coefficients at various lags in the time series are significantly different from zero. To make sure the statistical test returns meaningful results, we select attacker IPs with at least 5 connections (37.17% qualified IPs)—the rest of the attacker IPs unfortunately are deemed to have insufficient data for this analysis. The distribution of p-value is reported in Figure 12. Time series with a p-value near 0 represent attacker IPs that have highly regular time intervals (i.e., rejecting the null hypothesis), indicating bot behavior. We take $p < 0.05$ as the threshold, which is the commonly used value in statistical tests. We estimate that the majority (66.22%) of the attacks that made more than 5 connections are botnet-driven.

Category	Description of Alerts	# of Alerts
Internal Scan	Legitimate scan	194,518
External Pen-test	Various alerts	2,694,851
DNS	External domain w/ local IP	1,407,151
	External DNS server	28,698
	Excessive DNS queries	26,555
SSL	Weak key	30,570
	Weak cipher	6,909
	Old SSL version	2,339
Non-Attack	Logging failure	2,553,794
	New IP address space	138,630
	New host added	3
Services w/ Exceptions	Various alerts	260
Total:		7,057,012

Table 7: **Benign Triggers**— Types of benign triggers and the number of alerts. The alerts under different categories may overlap (e.g., internal scan alerts may include DNS/SSL-related alerts), and the total number is the *union* of these alerts.

Remark 5: We show that the vast majority of attack attempts are short-lived but persistent attack attempts exist. The highly uneven distribution of security alerts among hosts may create challenges to developing per-host security prediction models. Many recurrent attackers display characteristics consistent with botnets.

6.4 Understanding Benign Triggers

Prior works conducted *user studies* to study how SOC analysts perceive false positives (FP) [4, 37], and concluded that most FPs are benign triggers [4]. In this section, we examine benign triggers *quantitatively*. Identifying benign triggers requires specific knowledge about the organization's network and context [4]. As such, we worked with two SOC analysts to go through each alert category and apply rules/heuristics to filter out alerts of benign triggers. The rules are intentionally conservative as we only flag those with clear evidence of benign triggers (i.e., a lower bound). This process requires extensive manual efforts. The identification either requires manually constructing an allowlist of IPs where certain alerts are safe to ignore based on organizational context or by filtering candidate alerts and manually confirming their legitimacy with additional inquiries (to users or admins). Certain benign triggers can be filtered automatically (e.g., logging failures) without manual inspection. Due to the space limit, the detailed methodology is documented in Appendix C and Table 9. In total, we identified 7.05 million benign triggers (48.91% of the total alerts during 2020–2022) and a breakdown of specific reasons is shown in Table 7.

Internal Scan. The first common cause of benign triggers is scheduled internal scanning to proactively check for vulnerabilities and misconfigurations. These scans have similar behavior patterns as malicious scans, but the alerts can be safely ignored. We attempted to identify internal scans based on the dedicated scanner IPs. This returns 194,518 alerts.

Surprisingly, we observed another set of 384 internal IP addresses within the research center that have triggered network scanning alerts (scanning other internal or external hosts). Since they are not immediately attributed to benign triggers, we regard them as "unknown" for manual investigation. The detailed investigation is documented in the supplementary materials [78]. Anecdotally, we confirm there are web scrapers/crawlers running on internal IPs and legitimate internal scanners set up by other units of the institution (that are not well communicated with the SOC), or scanners set up by SOC members (that are not well-documented as those in Table 7). These all create challenges for tracking benign triggers.

External Pen Test. The SOC also worked with third-party (external) security vendors to perform penetration tests and the annual compliance auditing. The corresponding alerts are identified based on known vendor IPs (2.7 million alerts). While the external scanners have triggered varied types of alerts such as address scan, port scan, and vulnerability scan, the majority (92%) are related to Log4j in early 2022.

DNS Configurations. We identified benign triggers related to DNS services (1.4 million), and the vast majority are triggered due to external domains hosted under a local/internal IP. The detection rules are designed to capture internal host compromises. However, the research center often has collaborative projects with other organizations and there are websites (with external domain names) hosted on internal IPs. Similarly, benign triggers related to "external DNS server" happen when an internal host changes its default DNS server to unknown external DNS servers. This is supposed to capture DNS tunneling attacks (for data exfiltration), but alerts are triggered by these collaborative projects. Finally, "excessive DNS queries" means the number of DNS queries surpasses a daily threshold. We locate benign triggers as those that exclusively query the internal DNS (i.e., low risk).

SSL and Legacy Systems. Benign triggers related to SSL (e.g., weak key, weak cipher, old version) are mostly due to legacy systems—SOC determined they are either difficult to change or safe to ignore. This is consistent with what is reported in prior work [4]. Their detection is based on (manually) mapping out the IPs of internal legacy systems and then matching the alert types.

Alerts for Information Purposes. More than 2.5 million alerts are not related to attacks but are generated for information/context purposes. For example, most of such alerts are related to logging failures from Zeek. Others are related to new IP ranges or hosts added to the network.

Services with Exceptions. Finally, SOC analysts provided an allowlist of IPs that host special services with legitimate reasons for triggering certain alerts. For example, certain provisioning services are allowed to move large data around the network. Certain legitimate devices are known to trigger ArpScan alerts. This category only has 260 alerts.

Remark 6: We quantitatively show that benign triggers count for a significant portion of alerts (at least 48.91%). However, benign triggers are caused by complex reasons which are challenging to attribute. (1) Not all benign triggers are well-documented by the SOC and it took significant (manual) efforts to gather evidence and craft rules to flag them. (2) Even after these efforts, a major portion of alerts still can not be determined (23.77% unknown).

7 Linking Alerts with True Attacks

So far, we analyzed the large volume of alerts (§6) and the relatively small number of true attacks (§5). In this section, we focus on the overlapped periods of the two datasets (2018–2022) to correlate them. We aim to answer the following questions: are alerts successfully triggered during true attacks? Did the SOC use the alerts to identify the attacks? What characteristics of alerts would indicate true attacks?

7.1 Analyzing True Attacks

A case study of alert-attack association has been presented in §4. Here, we expand the analysis to examine the general effectiveness of alerts, we perform a similar analysis for all of the true attacks during the overlapping period (11 true attacks). As shown in Table 8, out of the 11 successful attacks, 7 attacks involve host compromise via existing user accounts, 2 involve host compromise via vulnerabilities, 1 case involves exploiting an open port to run reflection DoS attacks, and 1 case involves internal policy violation (crypto mining).

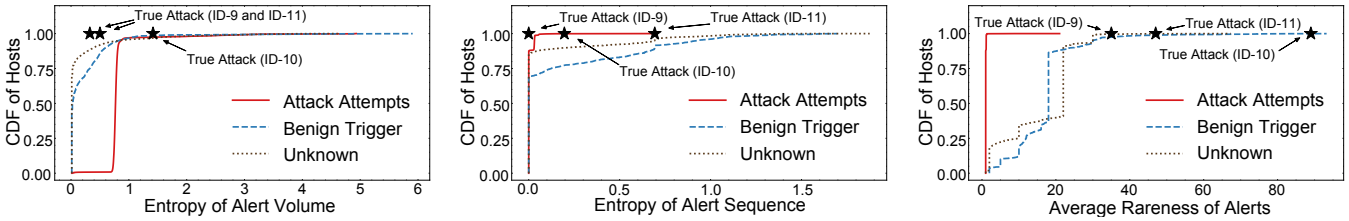
In Table 8, we attempt to map *related alerts* to these true attacks (under the guidance of the SOC analysts). Given a true attack (and its detailed incident report), we first identify the attacker IPs as well as the IPs of compromised internal hosts (if any), and use the IPs to match with the alert dataset. The mapping is further constrained by the timestamp (i.e., the same day). In addition to alerts related to these true attacks, we also report other alerts triggered on the same day of the attack to show the level of distraction that the SOC was facing. For true attacks that happened after 2020 (after BHR was implemented), we can also report the number of attack attempts (blocked by BHR) during the attack day.

Out of the 11 true attacks, 7 true attacks have triggered alerts. However, for one of them (case 1), the alerts (associated with the victim IP) were not directly related to the attack itself. For cases 4 and 6, the incident reports suggest that Zeek alerts were triggered but we cannot recover the exact

ID	Attack Info (Brief)	# Related Alerts	Alert Type	How Detected?	# of Benign Triggers	# of Attack Attempts	# of Other Alerts	# of Total Alerts
1	Acct. compromised; credential leak	2†	1	User	3,168	N/A	52,329	55,499
2	Acct. compromised; scanning	22	4	Zeek	19,167	N/A	144,689	163,878
3	Acct. compromised; data leak	0	0	Other	17,993	N/A	64,966	82,959
4	Acct. compromised; spam	0*	0*	Zeek	18,330	N/A	79,215	97,545
5	0-day; scanning	18	2	Zeek	19,401	N/A	104,735	124,154
6	Acct. compromised; outbound SSH	0*	0*	Zeek	0*	N/A	0*	0*
7	Acct. compromised; phishing	0	0	User	19,037	N/A	115,810	134,847
8	Acct. compromised; crypto mining	0	0	Other	2,958	N/A	28,480	31,438
9	Open port; DoS reflection	12	0	Zeek	6,628	5,540	5,766	17,946
10	Postgres compromise; scanning	1,101	5	Zeek	7,093	4,028	5,617	17,839
11	Internal account crypto mining	1	1	Zeek	14,657	3,947	6,316	24,921

∗: the alert log is missing.
†: the alerts are triggered but are not relevant to the attack.

Table 8: **Alert-Attack Association Results**—True attacks between 2018 and 2022. We briefly describe how the attacker breaks in and the attacker's actions after the compromise. We report *related alerts* triggered by the attack as well as other alerts triggered on the same day of the attack (e.g., benign triggers, attack attempts) to show the level of distraction that the SOC was facing.



(a) Entropy of Daily Alert Volume (b) Entropy of Alert Sequence (c) Alert Rareness Per Host Per Day

Figure 13: **Per-Host Alert Characteristics**—We show the CDF plots of per-host behavioral metrics for internal hosts in the network. The "stars" in the figure represent "true attacks."

alerts due to missing log entries. A closer examination shows these alerts were triggered mostly due to attackers' activities *after* the compromise, especially activities that involve networking (e.g., scanning, password guessing). On the contrary, the alerts were ineffective if the attacker only stole a small amount of data such as credentials and meeting notes (cases 1 and 3), or sent a small number of phishing emails (case 7). From the SOC's perspective, valid accounts may log in from networks outside of the organization (e.g., researchers from other countries). This made it challenging to generate alerts on account compromise if the user did not follow the policy to enable 2FA. For the four attacks where Zeek alerts were not triggered, they were found either by the account owners/users (cases 1 and 7) or by other parties/tools, e.g., network admin mailing list (cases 3 and 8). To reduce false negatives, we observe that SOC has been actively updating its rules based on past attacks. For instance, in case 8, an existing account was compromised which then installed crypto mining programs to the computation servers. After the incident, the SOC implemented new rules to alert on such behaviors. Later in case 11, when an internal account was used to run Bitcoin mining, an alert was successfully triggered to capture it. Overall, as shown in Table 8, the number of alerts related to the true attacks is much smaller in comparison with other alerts (i.e., distractions) triggered on the same day. In the next section (§7.2), we explore possible ways to distinguish true attack alerts from other categories.

Remark 7: While excessive alerts (and false positives) are problematic, false negatives are equally concerning. More specifically, network-level alerts are highly limited in capturing the initial host compromise or stealthy activities after the compromise.

7.2 Comparing Different Alert Categories

Next, we explore behavioral statistics that may help to distinguish true attacks from the other alert categories. We focus on individual internal hosts and plot a series of per-host behavioral metrics. We select the more distinguishing metrics in

Figure 13 and present the additional ones (e.g., alerts per day) in the supplementary materials [78]. Given this analysis will require connection logs (for victim IP recovery, see §6.3), we only focus on the period of 2020-2022 (covering true attacks 9, 10, 11).

Figure 13a shows the entropy of *daily alert volume* per host, and Figure 13b shows the entropy of the *alert sequence* of a host. A lower entropy means the data has a higher regularity (i.e., more predictable). Recall that the true attacks (cases 9, 10, 11) each only lasted for one day, and thus we compute their entropy based on the victim host's recent daily alerts in the past 10 days. As shown in Figure 13a, for "benign trigger" and "unknown", there are 45.7% and 70.4% of the hosts with an entropy near 0. This is because these hosts did not have corresponding alerts triggered on most days. In comparison, almost all the internal hosts have encountered attack attempts (over 99%). Overall, the attack attempts' daily volume is more random but their alert sequence is more predictable. This makes sense since most attack attempts are related to scanning and password guessing (§6.3). On the contrary, unknown and benign trigger alerts are more unpredictable on alert sequence but more stable on volume. For true attacks, case 10 is in the outlier area for alert volume (Figure 13a) and both cases 10 and 11 are in the outlier area for alert sequence (Figure 13b).

Figure 13c further shows that true attacks tend to have more rare combinations of alerts. Here, given a host and a date, we compute a *rareness score*. This is done by first ranking all the alerts based on their overall frequency in the data. Due to the skewness of the frequency, we did not use the frequency value as the rareness score but used the *ranking* as the normalized score. A higher score value means the alert type is rarer. The final rareness score is the *sum* of the score of each unique alert on the host in a day. Figure 13c shows that all three true attacks are located in the outlier area.

Remark 8: There are opportunities to use host-level abnormal alert characteristics (e.g., rare combinations of alerts) to identify and prioritize those that indicate true attacks.

7.3 Other Impact: New Rules at the SOC

Our analysis of the alerts and true attacks has led to the creation of new BHR rules at the SOC. Appendix B provides an example of such rules regarding blocking HTTP basic access authentication (based on the analysis of case 10).

8 Discussion and Conclusion

8.1 Implications to ML-Security Research

Our results have implications for active research directions that develop ML-based security systems for SOC. More specifically, getting access to real-world network data for research has been historically difficult due to privacy and legal complications [6, 29, 60]. As a result, researchers developed their ML models primarily based on public benchmark datasets with clearly labeled "benign" and "attack" traffic [17, 19, 23, 45, 50, 65, 73] and such attack traffic is often generated by simulated attacks on a controlled network. Our analysis indicates that *the problem may have been oversimplified* and should be revisited.

Impact of Attack Attempts and Bening Triggers. Most existing models either train on benign traffic for anomaly detection (semi-supervised or unsupervised) [13,33,50,52,68] or train with benign and malicious traffic supervised attack classification [35, 36, 79]. However, this problem formulation may not cleanly match real-world data.

§6.2 shows that about 27.37% of the alerts (3.9 million) represent the large number of "attack attempts" faced by the network. The remaining alerts may contain (undetected) attack attempts, and benign triggers (correctly matched security events with a legitimate explanation). The network traffic associated with "attack attempts" and "benign triggers" can complicate the setup for ML models. On the one hand, the attack attempts or benign triggers exhibit patterns of malicious activities and should not be simply grouped into the "benign" class. On the other hand, such attack attempts/benign triggers did not actually manifest into successful attacks or cause damage (i.e., should be ignored). If they are simply labeled to the "malicious" class, it is possible the ML model would pick up their patterns and produce even more irrelevant alerts. Alternatively, researchers may pre-filter network traffic associated with attack attempts or benign triggers before training with the data; however, the same treatment is needed during the model deployment, which can be a challenge (§6).

Beyond Classifying Malicious from Benign. Recently, researchers have worked on ML tools to handle other tasks in SOC such as correlating or ranking security alerts to identify true attacks [21, 40, 61, 75, 80]. We believe this type of work is highly needed, given the excessive number of alerts that SOC has (§6.1). At the same time, we notice key challenges through our analysis. First, to develop and evaluate ML models for alert correlation, it would require the "ground truth" of true attacks (specifically, successful attacks) that are worth human attention. Unfortunately, the ground truth of successful attacks is rare in comparison with the large volume of alerts (e.g., 0.01% in our dataset). We notice that existing benchmark datasets usually have more balanced attack-to-benign ratios [17, 65, 73] because attacks are simulated within concentrated time windows. Second, there are other problems beyond ranking alerts. For example, in §7, we show that alerts may fail to fire during the initial compromise (i.e., false negative problems) and there are opportunities to identify attackers' early attempts before the actual compromise to deploy interventions.

Our Recommendations. First, constructing more realistic benchmark datasets by clearly defining the "true positives".

Specifically, "attack attempts", and "benign triggers" should be distinguished from "successful attacks" in the data construction and evaluation process. Second, proactively experimenting with noisy labels since attack attempts or benign triggers may introduce inaccurate labels. Third, consider extreme positive-to-negative ratios (e.g., 0.01%) in their evaluation.

Finally, researchers may consider nuanced problem definitions for ML. For example, one interesting direction is to model the likelihood of attack attempts turning into real compromises based on their early signals. Another direction is to improve alert generation by jointly modeling *commonly-appeared* and *rarely-appeared* alert sequences for alert prioritization to identify true attacks.

8.2 Human Bottleneck vs. Automation

We show concrete evidence that *human analysts* are still the bottleneck in the SOC pipeline. While automation helps, many challenges remain to be addressed.

Post-attack analysis is time-consuming since it primarily relies on human efforts. Post-attack investigation of one true attack can take an average of 53 days. We show that an investigation often involves more than one analyst (§5). A particular issue noted in our analysis is the difficulty to *link different logs* to attribute victims and recover attacker behaviors (§6.3). For instance, due to alert aggregation, network scanning alerts may miss key metadata for victim IPs. Also, certain large logs (e.g., network connection logs) may become unavailable as they will be deleted periodically due to limited storage space.

Automated alert handling reduces human workload, but reasoning the remaining alerts is still challenging. We confirm that the SOC is faced with a large number of alerts (24K–134K daily for our dataset). It is possible for the SOC to use automated methods to handle a portion of the alerts (44.9%) using high-confidence rules (BHR). There still remain a large number of alerts (about 13K per day) that may require human attention. We demonstrate that these alerts can be triggered by complex reasons. Benign triggers are a key contributor to excessive alerts (49%), which confirms a prior user study [4]. Compared with [4], we further show isolating benign triggers is non-trivial due to the difficulty of extracting contexts from logs and a lack of SOC documentation.

Our Recommendations: First, to speed up the post-attack investigation and improve the efficiency of analysts, SOCs need efficient ways to store, link, and query different logs. Future work may consider including key metadata (e.g., victim IPs) in the alert log to make it easier to link alerts to other raw logs (e.g., network connection logs, and even host logs). In addition, future work may explore more storage-efficient log representations such that historical logs can be stored for a longer time.

Second, for benign trigger identification, our experience (§6.4) shows that significant manual efforts are needed for constructing and curating various allowlists. This process can be potentially improved with more systematic asset management, especially documentation of allowed exceptions. Future work may also look into automated methods (e.g., machine learning) to extract rules for benign trigger identification by analyzing related logs and documents.

Third, for threat response, automatons are needed to handle alerts, especially during off-hours when human analysts have limited availability. Our analysis shows the promise of investigating *rare combinations of alerts or sequences* to find true attacks (see §7). Finally, large language models (LLM) such as GPT-4 [55] and chatGPT [54] have demonstrated the ability to assist humans with a variety of technical tasks (e.g., developing websites, debugging, and fixing bugs [62]). A possible direction is to train LLM-like foundation models to help with threat response, guiding human analysts in investigating alerts and taking action [48]. During off-hours, it might be difficult for ML models to reason the attack automatically, but it may be possible to contain the attack (e.g., and its damage) before humans can investigate it. However, given the limitations of such models [58], it is still an open question regarding how to set the boundaries in the decision space.

8.3 Generalizability and Data Sharing.

Our dataset is collected from a single SOC, which may contain biases due to the specific SOC size, type, and network characteristics (§3). As a result, we want to be cautious about result generalization. Certain results should be discussed considering the SOC-specific contexts. For example, we show that attacks that occurred during the off-hours of the analysts may lead to delayed response (§5). This observation may not generalize to larger organizations with multiple SOC sites in different time zones (where analysts take shifts 24/7), but can be more serious for less resourceful SOCs. However, even with these factors considered, we believe certain lessons from our analysis have broader implications. For example, the issue of "attack attempts" and "benign triggers" can produce misleading labels for network traffic data for general ML models (which has been largely overlooked by the research community); the issue of information loss during log aggregation is a generic challenge for threat response using NIDS; and the use of rareness of alert combinations to inform filtering can be explored in broader contexts.

To facilitate future research and result-reproduction, we publish the code developed in the project, which is designed for the highly popular Zeek log format. The code should work for other network datasets formulated with Zeek. In addition, we have worked with the SOC to release one-month alert data during the month of the Postgres attack discussed in §4. The dataset (see [78]) contains ground-truth labels for true attacks, attack attempts, and benign triggers. For privacy

considerations, IP addresses or any network payload will not be included in the dataset.

Acknowledgments

This work is supported in part by the National Science Foundation (NSF) under grants 2029049, 2319190, 1935966, 1548562, 1547249, 1535070, 2229876, 2055233, 2055127, the IBM-ILLINOIS Discovery Accelerator Institute (IIDAI), and a gift from Nokia Bell Labs Core Research. We thank the NCSA cybersecurity team and staff (e.g., Phuong Cao, Christopher Clausen, and Alexander Withers) for their assistance with providing: 1) a curated security dataset, 2) insights, and 3) a testbed infrastructure for data analysis.

References

[1] Enoch Agyepong, Yulia Cherdantseva, Philipp Reinecke, and Pete Burnap. Challenges and performance metrics for security operations center analysts: a systematic review. *Journal of Cyber Security Technology*, 2020.

[2] AICPA. SOC 2: Audit & assurance. hhttps://www.aicpa.org/topic/audit-assurance/audit-and-assurance-greater-than-soc-2, 2023.

[3] Olusola Akinrolabu, Ioannis Agrafiotis, and Arnau Erola. The challenge of detecting sophisticated attacks: Insights from soc analysts. In *Proc. of ARES*, 2018.

[4] Bushra A. Alahmadi, Louise Axon, and Ivan Martinovic. 99% false positives: A qualitative study of SOC analysts' perspectives on security alarms. In *Proc. of USENIX Security*, 2022.

[5] Ahmed AlEroud and George Karabatis. Beyond data: Contextual information fusion for cyber security analytics. In *Proc. of SAC*, 2016.

[6] Mark Allman and Vern Paxson. Issues and etiquette concerning use of shared measurement data. In *Proc. of IMC*, 2007.

[7] Luca Allodi and Fabio Massacci. Security events and vulnerability data for cybersecurity risk estimation. *Risk Analysis*, 2017.

[8] Abdulellah Alsaheel, Yuhong Nan, Shiqing Ma, Le Yu, Gregory Walkup, Z. Berkay Celik, Xiangyu Zhang, and Dongyan Xu. ATLAS: A sequence-based learning approach for attack investigation. In *Proc. of USENIX Security*, 2021.

[9] Naomi Altman and Martin Krzywinski. Points of significance: Association, correlation and causation. *Nature methods*, 2015.

[10] Mohammad Samar Ansari, Václav Bartoš, and Brian Lee. Gru-based deep learning approach for network intrusion alert prediction. *Future Generation Computer Systems*, 128:235–247, 2022.

[11] André Årnes, Fredrik Valeur, Giovanni Vigna, and Richard A. Kemmerer. Using hidden markov models to evaluate the risks of intrusions. In Diego Zamboni and Christopher Kruegel, editors, *Proc. of RAID*, 2006.

[12] Tao Ban, Ndichu Samuel, Takeshi Takahashi, and Daisuke Inoue. Combat security alert fatigue with ai-assisted techniques. In *Proc. of CSET Workshop*, 2021.

[13] Monowar H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys & Tutorials*, 2014.

[14] Haibo Bian, Tim Bai, Mohammad A Salahuddin, Noura Limam, Abbas Abou Daya, and Raouf Boutaba. Host in danger? detecting network intrusions from authentication logs. In *CNSM*, 2019.

[15] Benjamin Bowman, Craig Laprade, Yuede Ji, and H. Howie Huang. Detecting lateral movement in enterprise computer networks with unsupervised graph AI. In *Proc. of RAID*, 2020.

[16] David J. Chaboya, Richard A. Raines, Rusty O. Baldwin, and Barry E. Mullins. Network intrusion detection: Automated and manual methods prone to attack and evasion. *IEEE Security and Privacy*, 2006.

[17] Gideon Creech and Jiankun Hu. Generation of a new ids test dataset: Time to retire the kdd collection. In *Proc. of WCNC*, 2013.

[18] Cybersecurity & Infrastructure Security Agency. Apache log4j vulnerability guidance. https://www.cisa.gov/uscert/apache-log4j-vulnerability-guidance, 2022.

[19] Robertas Damasevicius, Algimantas Venckauskas, Sarunas Grigaliunas, Jevgenijus Toldinas, Nerijus Morkevicius, Tautvydas Aleliunas, and Paulius Smuikys. Litnet-2020: An annotated real-world network flow dataset for network intrusion detection. *Electronics*, 9(5), 2020.

[20] Constanze Dietrich, Katharina Krombholz, Kevin Borgolte, and Tobias Fiebig. Investigating system operators' perspective on security misconfigurations. In *Proc. of CCS*, 2018.

[21] Feng Dong, Liu Wang, Xu Nie, Fei Shao, Haoyu Wang, Ding Li, Xiapu Luo, and Xusheng Xiao. Distdet: A cost-effective distributed cyber threat detection system. In *Proc. of USENIX Security*, 2023.

[22] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J Alex Halderman. Neither snow nor rain nor mitm... an empirical analysis of email delivery security. In *Proc. of IMC*, 2015.

[23] Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga. Iot-23: A labeled dataset with malicious and benign iot network traffic (version 1.0.0). http://doi.org/10.5281/zenodo.4743746, 2020.

[24] John Goodall, Wayne Lutters, and Anita Komlodi. I know my network: Collaboration and expertise in intrusion detection. In *Proc. of CSCW*, 2004.

[25] Xueyuan Han, Thomas F. J.-M. Pasquier, Adam Bates, James Mickens, and Margo I. Seltzer. Unicorn: Runtime provenance-based detector for advanced persistent threats. In *Proc. of NDSS*, 2020.

[26] Woodrow Hartzog and Daniel Solove. We still haven't learned the major lesson of the 2013 target hack. https://slate.com/technology/2022/04/breached-excerpt-hartzog-solove-target.html, 2022.

[27] Wajih Ul Hassan, Adam Bates, and Daniel Marino. Tactical provenance analysis for endpoint detection and response systems. In *Proc. of IEEE S&P*, 2020.

[28] Wajih Ul Hassan, Shengjian Guo, Ding Li, Zhengzhang Chen, Kangkook Jee, Zhichun Li, and Adam Bates. Nodoze: Combatting threat alert fatigue with automated provenance triage. *Proc. of NDSS*, 2019.

[29] John Heidemann and Christos Papadopoulos. Uses and challenges for network datasets. In *Proc. of CATCH*, 2009.

[30] Grant Ho, Aashish Sharma, Mobin Javed, Vern Paxson, and David Wagner. Detecting credential spearphishing in enterprise settings. In *Proc. of USENIX Security*, 2017.

[31] Danny Yuxing Huang, Doug Grundman, Kurt Thomas, Abhishek Kumar, Elie Bursztein, Kirill Levchenko, and Alex C. Snoeren. Pinning down abuse on google maps. In *Proc. of WWW*, 2017.

[32] Martin Husák, Jana Komárková, Elias Bou-Harb, and Pavel Čeleda. Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Communications Surveys & Tutorials*, 21(1):640–660, 2019.

[33] S. Jacobs, Roman Beltiukov, Walter Willinger, Ronaldo A. Ferreira, Arpit Gupta, and Lisandro Z. Granville. Ai/ml for network security: The emperor has no clothes. In *Proc. of CCS*, 2022.

[34] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. In *Proc. of BICT*, 2016.

[35] Farrukh Aslam Khan, Abdu Gumaei, Abdelouahid Derhab, and Amir Hussain. A novel two-stage deep learning model for efficient network intrusion detection. *IEEE Access*, 2019.

[36] Jiyeon Kim, Jiwon Kim, Hyunjung Kim, Minsun Shim, and Eunjung Choi. Cnn-based network intrusion detection against denial-of-service attacks. *Electronics*, 2020.

[37] Faris Bugra Kokulu, Ananta Soneji, Tiffany Bao, Yan Shoshitaishvili, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. Matched and mismatched socs: A qualitative study on security operations center issues. In *Proc. of CCS*, 2019.

[38] Christopher Krügel and William K. Robertson. Alert verification determining the success of intrusion attempts. In *Proc. of DIMVA*, 2004.

[39] Warren Kumari and Danny McPherson. RFC5635: Remote triggered black hole filtering with unicast reverse path forwarding (urpf). https://www.rfc-editor.org/rfc/rfc5635, 2009.

[40] Jehyun Lee, Farren Tang, Phyo May Thet, Desmond Yeoh, Mitch Rybczynski, and Dinil Mon Divakaran. SIERRA: ranking anomalous activities in enterprise networks. In *Proc. of EuroS&P*, 2022.

[41] Lizzy Li. Splunk dashboard studio: Dashboard customization. https://www.splunk.com/en_us/blog/platform/dashboard-studio-dashboard-customization-made-easy.html, 2021.

[42] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M. Voelker, Stefan Savage, and Kirill Levchenko. Reading the tea leaves: A comparative analysis of threat intelligence. In *Proc. of USENIX Security*, 2019.

[43] Ming Liu, Zhi Xue, Xianghua Xu, Changmin Zhong, and Jinjun Chen. Host-based intrusion detection system with system calls: Review and future trends. *ACM Computing Surveys (CSUR)*, 2018.

[44] Greta M Ljung and George EP Box. On a measure of lack of fit in time series models. *Biometrika*, 1978.

[45] Gabriel Maciá-Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón. Ugr'16: A new dataset for the evaluation of cyclostationarity-based network idss. *Computers & Security*, 73:411–424, 2018.

[46] Maxmind. Maxmind geoip. https://www.maxmind.com/, 2024.

[47] Nora McDonald, Sarita Schoenebeck, and Andrea Forte. Reliability and inter-rater reliability in qualitative research: Norms and guidelines for cscw and hci practice. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW), 2019.

[48] Microsoft. Microsoft Security Copilot. https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-security-copilot, 2023.

[49] Sadegh M. Milajerdi, Birhanu Eshete, Rigel Gjomemo, and V.N. Venkatakrishnan. Poirot: Aligning attack behavior with kernel audit records for cyber threat hunting. In *Proc. of CCS*, 2019.

[50] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: an ensemble of autoencoders for online network intrusion detection. In *Proc. of NDSS*, 2018.

[51] MITRE. MITRE ATT&CK Framework. https://attack.mitre.org/, 2023.

[52] Biswanath Mukherjee, L Todd Heberlein, and Karl N Levitt. Network intrusion detection. *IEEE network*, 1994.

[53] S. Oesch, R. Bridges, J. Smith, J. Beaver, J. Goodall, K. Huffer, C. Miles, and D. Scofield. An assessment of the usability of machine learning based tools for the security operations center. In *Proc. of iThings*, 2020.

[54] OpenAI. ChatGPT. https://openai.com/blog/chatgpt, 2024.

[55] OpenAI. GPT-4. https://openai.com/research/gpt-4, 2024.

[56] Yang-jia Ou, Ying Lin, and Yan Zhang. The design and implementation of host-based intrusion detection system. In *Proc. of IITSI*, 2010.

[57] Vern Paxson. Bro: A system for detecting network intruders in Real-Time. In *Proc. of USENIX Security*, 1998.

[58] Hammond Pearce, Baleegh Ahmad, Benjamin Tan, Brendan Dolan-Gavitt, and Ramesh Karri. Asleep at the keyboard? assessing the security of github copilot's code contributions. In *Proc. of IEEE SP*, 2022.

[59] Antonio Pecchia, Aashish Sharma, Zbigniew Kalbarczyk, Domenico Cotroneo, and Ravishankar K Iyer. Identifying compromised users in shared computing infrastructures: A data-driven bayesian network approach. In *Proc. of SRDS*, 2011.

[60] Phillip Porras and Vitaly Shmatikov. Large-scale collection and sanitization of network security data: Risks and challenges. In *Proc. of NSPW*, 2006.

[61] Elias Raftopoulos, Matthias Egli, and Xenofontas Dimitropoulos. Shedding light on log correlation in network forensics analysis. In Ulrich Flegel, Evangelos Markatos, and William Robertson, editors, *Proc. of DIMVA*, 2013.

[62] Rollbar. How to debug code using chatgpt. https://rollbar.com/blog/how-to-debug-code-using-chatgpt/, 2023.

[63] Saeed Salah, Gabriel Maciá-Fernández, and Jesús E. Díaz-Verdejo. A model-based survey of alert correlation techniques. *Computer Networks*, 2013.

[64] SANS. Sans 2022 soc survey. https://www.sans.org/white-papers/sans-2022-soc-survey/, 2022.

[65] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proc. of ICISSP*, 2018.

[66] Aashish Sharma, Zbigniew Kalbarczyk, James Barlow, and Ravishankar Iyer. Analysis of security data from a large computing organization. In *Proc. of DSN*, 2011.

[67] Yun Shen, Enrico Mariconti, Pierre Antoine Vervier, and Gianluca Stringhini. Tiresias: Predicting security events through deep learning. In *Proc. of CCS*, 2018.

[68] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proc. of IEEE S&P*, 2010.

[69] Rock Stevens, Daniel Votipka, Josiah Dykstra, Fernando Tomlinson, Erin Quartararo, Colin Ahern, and Michelle L. Mazurek. How ready is your ready? assessing the usability of incident response playbook frameworks. In *Proc. of CHI*, 2022.

[70] Sathya Chandran Sundaramurthy, Alexandru G. Bardas, Jacob Case, Xinming Ou, Michael Wesch, John McHugh, and S. Raj Rajagopalan. A human capital model for mitigating security analyst burnout. In *Proc. of SOUPS*, 2015.

[71] Sathya Chandran Sundaramurthy, Jacob Case, Tony Truong, Loai Zomlot, and Marcel Hoffmann. A tale of three security operation centers. In *Proc. of CCS*, 2014.

[72] Sathya Chandran Sundaramurthy, John McHugh, Xinming Ou, Michael Wesch, Alexandru G. Bardas, and S. Raj Rajagopalan. Turning contradictions into innovations or: How we learned to stop whining and improve security operations. In *Proc. of SOUPS*, 2016.

[73] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *Proc. of CISDA*, 2009.

[74] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. Data breaches, phishing, or malware? understanding the risks of stolen credentials. In *Proc. of CCS*, 2017.

[75] Thijs van Ede, Hojjat Aghakhani, Noah Spahn, Riccardo Bortolameotti, Marco Cova, Andrea Continella, Maarten van Steen, Andreas Peter, Christopher Kruegel, and Giovanni Vigna. DeepCASE: Semi-Supervised Contextual Analysis of Security Events. In *Proc. of IEEE S&P*, 2022.

[76] Mathew Vermeer, Michel van Eeten, and Carlos Gañán. Ruling the rules: Quantifying the evolution of rulesets, alerts and incidents in network intrusion detection. In *Proc. of AsiaCCS*, 2022.

[77] Giovanni Vigna and Richard A Kemmerer. Netstat: A network-based intrusion detection system. *Journal of computer security*, 1999.

[78] Limin Yang, Zhi Chen, et al. Data/code sharing and supplementary materials. https://github.com/idashlab/SOC_Measurement_Usenix24_related_material, 2024.

[79] Limin Yang, Wenbo Guo, Qingying Hao, Arridhana Ciptadi, Ali Ahmadzadeh, Xinyu Xing, and Gang Wang. CADE: Detecting and explaining concept drift samples for security applications. In *Proc. of USENIX Security*, 2021.

[80] Ting-Fang Yen, Alina Oprea, Kaan Onarlioglu, Todd Leetham, William Robertson, Ari Juels, and Engin Kirda. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proc. of ACSAC*, 2013.

[81] Zeek. The zeek network security monitor. https://zeek.org/, 2023.

[82] C. Zimmerman. *Ten Strategies of a World-Class Cybersecurity Operations Center*. MITRE Corporation, 2014.

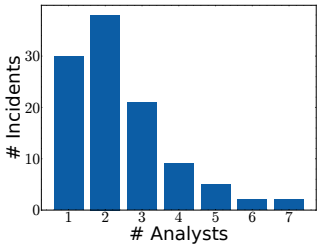


Figure 14: # of Analysts involved in the investigation.

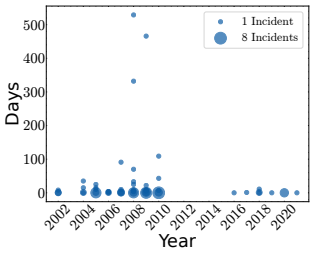


Figure 15: # of Days before the attack was discovered.

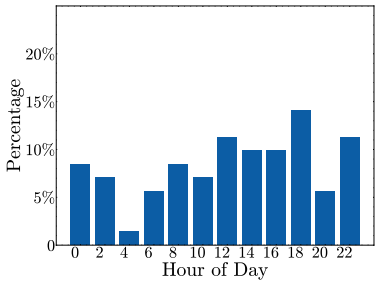


Figure 16: *Attack start* time (attacker time zone).

A Detailed Analysis of Incident Reports

Qualitative Coding Method. Given that incident reports are unstructured, we manually annotate the reports to extract *structured information* for our analysis. The coding process involves extracting structured data fields by reading the incident reports and labeling the incident type. This is different from a typical thematic coding process [47] (e.g., we don't need to develop a codebook, but instead rely on a pre-defined data field list to perform annotation). Two researchers (who have the domain knowledge and expertise) worked on the annotation task, and both annotated all the reports. They frequently met and discussed the results to resolve any disagreement collaboratively along the process.

First, for each incident, we document the incident ID and the timestamp of the incident, and verify the incident is indeed a *true attack*. Here, true attacks are defined as those where the attacker successfully compromised the system and/or caused damages (see Table 1). For the true attacks, we extract further information including the number of analysts involved in the investigation, the attacker/victim IPs, attack start time (t_1), attack detection time (t_2), and investigation end time (t_3). Here, "attack start time" is the earliest timestamp of the attacker activity that is directly associated with the compromise. The determination was made by the SOC analysts who conducted the investigation and wrote the report. Then we refer to the MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) Framework [51] to annotate the high-level techniques used by the attacker. We focus on MITRE tags that best describe (1) how the attacker broke into the system, and (2) the consequences/activities after breaking in. Here, we only seek to assign the most relevant ATT&CK technique to describe each attack, rather than trying to cover all related techniques.

Type of Attacks. Among the 227 true attacks, we can assign MITRE techniques/tags to 220 attacks. For the 7 remaining attacks, we cannot find a MITRE technique that accurately describes them, and thus we create two custom tags, namely "DMCA hosting" and "internal policy violations." More details are reported in the supplementary materials [78].

Analysts Involved. 107 of the reports have clearly documented the analysts involved in the investigation. Using this set, we plot Figure 14. It is more common for two or more SOC analysts to investigate a single attack (65% of the attacks).

Delay of Attack Detection. To examine how long it took for the SOC to discover an attack, it requires two timestamps: the attack start time and the attack discovery time. Out of the 227 reports, 96 of them contain both timestamps. We show the delay of attack detection in Figure 15. The size of the circles represents the number of attacks in a given year. Overall, most of the attack attacks were identified within the same day (63/96, 66%). Since 2011, this has been true for almost all attacks (except for one in 2018 which had a delay of one extra day). Before 2011, the delay varied from 0 days to as long as more than 300 days. We further analyze the attacks with long detection delays and summarize representative reasons: (1) attackers launched DDoS attacks to distract the SOC team from discovering the actual compromise; (2) worms/malware stayed hibernated for a while before launching attacks; (3) attackers scanned the network extremely slowly; and (4) the compromise was noticed by users or system admins accidentally long after the attack. Most of these attacks happened before 2011, which again correlates with the lack of logging capability and the SOC's understaffing problem before 2011.

Category	Identification Method	Required Effort
Internal Scan	Manually compile a list of legit internal scanners; Match alerts with legit scanners' IPs	Semi-Automatic
External Pen-test	Manually map out known external security vendors' IPs; Match alerts with these IPs	Semi-Automatic
DNS (external domain name)	Manually document a list of internal projects that use external domain names; Add to allowlist	Manual
DNS (external DNS server)	Manually document a list of internal projects with customized DNS configs; Add to allowlist	Manual
DNS (excessive queries)	Filter based on alert type, and pattern (internal source IP querying internal DNS)	Automatic
SSL (week key)	Manually map out IPs of internal legacy systems; Match alert types, metadata, and IPs	Semi-Automatic
SSL (week cipher)	Manually map out IPs of internal legacy systems; Match alert types, metadata, and IPs	Semi-Automatic
SSL (old version)	Manually map out IPs of internal legacy systems; Match alert types, metadata, and IPs	Semi-Automatic
Non-Attack (logging failure)	Match alert types	Automatic
Non-Attack (new IP space)	Match alert types; Manually confirm alert details	Semi-automatic
Non-Attack (new host added)	Match alert types; Manually confirm alert details	Semi-automatic
Services w/ Exceptions	Match alerts with allow-listed IPs	Automatic

Table 9: **Benign Triggers Identification**— We describe how we identified the benign triggers in each category.

Attackers' Local Time Analysis. In §5, we show that more attacks *started* during the off-hours of the SOC (based on the SOC's local time). However, we cannot confirm whether attackers were *intentionally* targeting the SOC's off-hours—one alternative explanation is attackers (especially those located in a different time zone) just happened to start during the SOC off-hours. To provide further insights, we try to convert the *attack start time* based on the attacker IPs' local time. Note that not all true attacks have identifiable IPs and thus this analysis can only use a subset of the incident reports (71 attacks). To convert the attack start time, we first obtain the attacker IP's geolocation using the MaxMind GeoIP database [46], and then convert the timestamp to the attacker IP's local time zone. As shown in Figure 16, there are slightly more attacks started between 8 AM and 6 PM (i.e., the attackers' local working hours). As such, it is possible that some of the attackers' local working hours happened to overlap with the SOC's off-hours. That said, one limitation of this analysis is that we cannot ensure these IPs are the attackers' real IPs (e.g., attackers may connect to the SOC's network via other compromised external hosts or botnets). This can only be interpreted as an estimation of the local time of the *directly connected* attacker IPs.

B New Rules/Policies Added to SOC

Our analysis has led to the creation of new rules at the SOC. As a concrete example, during our analysis of the Postgres compromise (§4), we observed interesting patterns in the alerts. More specifically, after the victim host D is compromised, the attacker used the host to scan a large number of external hosts and performed password guessing (which collectively raised 1,100 alerts). Among the alerts, 1,069 were related to HTTP basic access authentication. We observe that *all of the HTTP basic access authentication* requests were made to external hosts with the same username and password encoded with Base64. We then searched this username and password combination in our own alert dataset and found another 3,377 alerts with the exact match. This indicates this username-password has been used by other attackers to attack the internal network too. By ranking all the usernames and passwords used in such attack attempts in the alert log, this identified username ranked number three (right behind "admin" and the empty string ""). After discussing with SOC analysts, new rules are added to the BHR to automatically block such HTTP access authentication attempts based on the alert type and known username and password combinations.

C Benign Triggers Detection Method

Using Table 9, we provide further details regarding the identification process of different benign triggers. As shown in Table 9, the identification of benign triggers either requires manually constructing an allowlist of IPs where certain alerts are safe to ignore based on organizational context or by filtering candidate alerts and manually confirming their legitimacy. Most manual efforts are spent on obtaining such contexts through additional inquiries to network administrators and users, or within the SOC team. For example, identifying legitimate internal scans and external penetration tests requires the knowledge of trusted IP ranges of internal scanners and third-party security vendors. Identifying DNS-related benign triggers requires documenting and verifying the list of internal and collaborative projects that have such violations (paired with the allowlist of certain behavior patterns). Identifying SSL-related benign triggers requires mapping out the IPs of internal legacy systems and filtering certain alert types. Once such allowlists are created and curated, the alert matching process can be coded and executed automatically. Certain benign triggers (such as "logging failures") can be matched based on the alert type alone, without manual efforts.