ATLAS v2: An Open-Source Dataset for Intrusion Detection Research

Distributed REsearch Apprenticeships for Master's Final Project

The Secure and Transparent Systems Lab, University of Illinois Urbana-Champaign

Andy Riddle
University of Illinois Urbana-Champaign
rriddle2@illinois.edu

Kim Westfall
University of Illinois Urbana-Champaign
kw26@illinois.edu

***Abstract*—Audit logs are a key part of computer system security and provide detailed information allowing intrusion detection systems to alert cyber analysts of suspicious activity. However, the volume of alerts most systems generate can cause time-constrained analysts to miss true attacks. Recent work looks to use machine learning techniques to isolate alerts and system log events that represent genuine attack behavior and minimize false positives of traditional IDS. One significant obstacle in developing and evaluating such systems is the quality of current, publicly available system logging datasets.**

In this work we examine existing systems for evaluating security logs and the limitations of the training and testing data available to these systems. Additionally, we produce an improved system log dataset capturing events from Microsoft Windows Security Auditing, Microsoft SysMon, Carbon Black Cloud Endpoint Detection and Response System, Firefox application system logging, and DNS logging data from Wireshark. This dataset covered multiple days of benign system use from two researchers and included an attack phase that incorporated ten different advanced attacks in both single host and multiple host configurations.

Finally, we perform an analysis of the data generated in the ATLASv2 engagement, In particular, we focus on the volume of alerts generated over the course of the engagement. This highlights the need for more effective systems to provide support to security analysts in prioritizing which alerts require further investigation.

1. Introduction

Intrusion detection systems (IDS) have been heavily relied on to protect businesses and organizations from unauthorized access, malicious activity, and internal threats. However, these systems are known to generate high rates of false alarms, often resulting in missed true attacks [1]. Additionally, as new, sophisticated cyber-attacks emerge, these systems are becoming increasingly less reliable. Most threatening is the prolonged, stealthy style of Advanced Persistent Threats (APTs) [2]. Traditonal IDSs are unable to detect the zero-day vulnerablities commonly used by APTs to maintain anonymity [3]. To evade these attacks, researchers have developed new, more robust threat detection models.

Though machine learning techniques utilized in new threat detection models vary significantly [4], current research demonstrates a distinct shift to the use of system audit logs in the development of these systems [5]. System logs capture timestamped reports of various system operations, such as detecting a service's origin (i.e., device identification number or IP address), documenting the action type of an object (i.e., read, delete), or identifying the actor responsible for modifying an object. The detailed evidence of activities on a machine provided by these logs are needed to train systems to detect threats. However, one significant challenge to developing and evaluating such systems is the quality of current, publicly available system logging datasets.

In this work we examine current threat detection models that operate on system logs (§2) and present an overview of our new dataset, `ATLASv2`. Then, we outline the technical details of our experimental setup used to generate `ATLASv2` (§3–4). Lastly, we present our evaluation of `ATLASv2` (§5) and conclude (§6).

2. Background and Motivation

Auditing is fundamental to system security. Early work on operating system security [6] identified effective auditing as an essential component to detect system compromise and attempted compromise. When defense measures such as authentication and authorization fail, security auditing provides system administrators with the opportunity to detect and remediate security breaches earlier, reducing the overall damage of the breach.

New threat detection models based on system audit logs demonstrate their potential for future security developments [7]. Their increased use is attributed to data provenance [5], or, the evaluation of system audit logs to

construct a complete attack story. This includes *backward tracing*: analyzing events leading up to the attack and *forward tracing*: following the events after to evaluate the effects.

Many new intrusion detection systems analyze data provenance derived from audit logs. One such system, SLEUTH, was designed to provide analysts real-time alerts with a visual summary of the activity promptly after an attack is detected [8]. SLEUTH suggests the use of graph databases to support rapid analysis. Another system, NoDOZE, combined this causality graph concept with an "anomaly score", effectively decreasing the high false-alarm rates that previously plagued intrusion detection systems [9]. SLEUTH, NoDOZE, and other systems like UNICORN [10], ATLAS [11], and THREATRACE [12] utilizing system logs and graph representations have demonstrated audit logs power for detecting APT attacks. Unfortunately, the quality of current, publicly available system log datasets can be a significant obstacle to developing and evaluating new intrusion detection systems. There are few publicly available datasets and those that do exist suffer from common issues, such as being outdated [13] and not reflecting realistic computer usage encountered in real-world settings [14].

To support new intrusion detection systems, we present ATLASv2, a new dataset for intrusion detection systems. With ATLASv2 we look to provide a dataset with realistic user activity that will enable improved training of systems for security log analysis. This will help increase the robustness of such systems, and improve performance in real-world settings.

3. Overview

ATLASv2 is based on a previously generated dataset presented in ATLAS [11]. The original ATLAS dataset is comprised of Windows Security Auditing system logs, Firefox logs, and DNS logs collected via WireShark. In ATLASv2, we aim to enrich the ATLAS dataset with higher quality background noise and additional logging vantage points. This work replicates the ten attack scenarios described in ATLAS, extends the logging to include Sysmon logs and events tracked through VMware Carbon Black Cloud, and adds four days of high-quality benign activity that captures actual usage data on the victim machines.

The main contribution of ATLASv2 is to improve the quality of the benign system activity and the integration of the attack scenarios and to increase the number of logging vantage points available in a single engagement. Instead of relying on automated scripts to generate activity (a common pitfall of public datasets), two researchers use the victim machines as their primary work stations throughout the course of the engagement. The researchers assigned to victim machines carried out their normal daily activities while generating system logs of their typical user behaviors.

	#Endpoints	*CVE*	*Attack Vector*
S1	Single-Host	CVE-2015-5122	Adobe Flash
S2	Single-Host	CVE-2015-3105	Adobe Flash
S3	Single-Host	CVE-2017-11882	MS Word
S4	Single-Host	CVE-2017-0199	MS Word
M1	Multi-Host	CVE-2015-5122	Adobe Flash
M2	Multi-Host	CVE-2015-5119	Adobe Flash
M3	Multi-Host	CVE-2015-3105	Adobe Flash
M4	Multi-Host	CVE-2018-8174	MS Word
M5	Multi-Host	CVE-2017-0199	MS Word
M6	Multi-Host	CVE-2017-11882	MS Word

TABLE 1: **Attack Scenarios**. Common names of the attack scenarios carried out in ATLASv2 and their associated CVEs.

Additionally, the researchers conducted the attacks in a lab setup allowing the integration of the attack into the work flow of the victim user. This allows the ATLASv2 dataset to provide realistic system logs that mirror the system log activity generated in real-world attacks.

The addition of the Carbon Black Cloud End-Point Detection & Response system logging through the engagement provides an important additional vanatage point on the systems through the benign and attack phases. This provides additional opportunity for researchers to compare their system alerts with those of a current state-of-the-art system. In addition, this allows training of systems for additional tasks such as alert triage, and rule rewriting for expanding the efficacy of rule-based EDR systems.

3.1. Attack Behavior

During the five-day data collection period for ATLASv2, we execute ten real-world attack scenarios as described in ATLAS. The attacks include six unique CVEs, four of which are executed in both single-host (s1-s4) and multi-host (m1-m6) environments. The attack common names and the CVE for each can be found in Table 1.

CVE-2015-5122 [15] This attack is a use-after-free vulnerability found in the DisplayObject class in Adobe Flash Player 13.x through 13.0.0.302. It leverages improper handling of the opaqueBackground property and allows remote attackers to execute arbitrary code.

CVE-2015-5119 [16] This attack is another use-after-free vulnerability in the ByteArray class in Adobe Flash Player 13.x through 13.0.0.296 and 14.x through 18.0.0.194 that allows remote attackers to execute arbitrary code or cause a denial of service using crafted Flash content that overrides a valueOf function.

CVE-2015-3105 [17] This attack exploits a vulnerable Adobe Flash Player before 13.0.0.292 and 14.x through 18.x before 18.0.0.160. It allows attackers the use of unspecified vectors to execute arbitrary code or cause a denial of service.

CVE-2018-8174 [18] Windows 7, Windows 8.1, Windows 10, and various Windows Server editions contained a vulnerability from the way VBScript engine handles objects. Remote code execution is possible through Microsoft Word on affected systems.

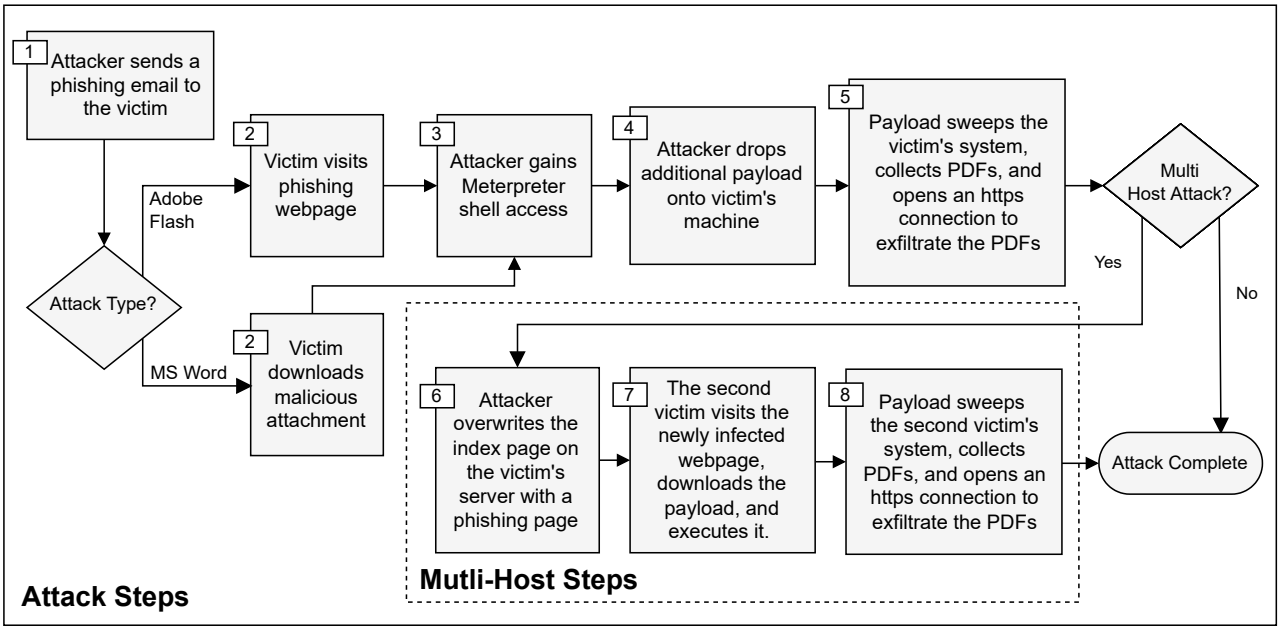

Figure 1: **Steps for executing attack scenarios found in `ATLASv2`.**

CVE-2017-0199 [19] Various Windows Systems allow remote attackers to execute arbitrary code through a crafted document. This attack is known more commonly as "Microsoft Office/-WordPad Remote Code Execution Vulnerability w/ Windows API."
CVE-2017-11882 [20] Microsoft Office Service Packs fail to properly handle objects in memory, allowing attackers to run arbitrary code.

Attack Steps To begin, the attacker runs a python script to setup an exfiltration server to collect PDFs from the victim machine. Next, the attacker launches Metasploit and sets the exploit according to the attack scenario (listed in Table 2) and the reverse https payload.

A URL or a Word document is generated, according to the attack scenario. The attacker crafts a phishing email using the URL or document and sends it to the victim.[1] Next, the first victim user opens the malicious email from a browser equipped with the vulnerable Adobe Flash version. When the victim clicks the malicious link in the phishing email, a new connection is established to fetch and inject the Meterpreter HTTPS payload to the current Firefox process. The attacker uses this access to gather information about the system by running simple shell commands, such as `getuid`, `pwd`, `ps`, and `ipconfig` and metasploit modules, such as `post/windows/gather/usb_history` and `post/windows/gather/enum_shares`. Then, the attacker infects the system with an additional payload (payload.exe). The malicious payload sweeps the system to collect high-profile PDF files then opens an HTTPS connection to exfiltrate them. The single-host attacks end here.

1. When executing the attacks, Gmail flagged the Word documents as malicious. To evade this issue, we sent the documents as encrypted zip files.

In the multi-host attack scenarios, the victim machine is running an HTTP server that hosts a simple web portal. The attacker continues from the single host attack and sets up a path for lateral-movement by overwriting the victim's index webpage with a poisoned phishing webpage. The attack on the first victim is now complete and the scenario moves on to the second victim machine, which accesses the infected portal website. The new victim downloads and executes the same malicious payload from the website hosted by victim one. This payload then sweeps the new system to collect targeted PDF files. Finally these PDFs are sent back to the exfiltration server run by the attacker.

	Metasploit Exploit
S1	exploit/multi/browser/adobe_flash_opaque_background_uaf
S2	exploit/multi/browser/adobe_flash_shader_drawing_fill
S3	exploit/windows/fileformat/office_ms17_11882
S4	exploit/windows/fileformat/office_word_hta
M1	exploit/multi/browser/adobe_flash_opaque_background_uaf
M2	exploit/multi/browser/adobe_flash_hacking_team_uaf
M3	exploit/multi/browser/adobe_flash_shader_drawing_fill
M4	exploit/windows/fileformat/cve_2018_8174
M5	exploit/windows/fileformat/office_word_hta
M6	exploit/windows/fileformat/office_ms17_11882
	Metasploit Payload
*	payload windows/meterpreter/reverse_https

TABLE 2: **Metasploit Exploits and Payload**. Exploits and payload from Metasploit used in `ATLASv2`.

Type	*Known Attack Footprint*	*Assoc. Attacks*
Domain	`ortrta.net`	all
Program	`payload.exe`	all
IP	`10.193.66.115:9999`	all
File	`s3take2.zip`	s3
File	`s4-at-night.zip`	s4
File	`m4.zip`	m4
File	`m5-2.zip`	m5
File	`m6.zip`	m6

TABLE 3: **Known Attack Entities**. Artifacts known to be malicious within `ATLASv2`.

Attack	*Machine Running Times*	*Initial Attack Step*
s1	`2022-07-19 13:12:00-13:40:00`	`13:26:00`
s2	`2022-07-19 13:45:00-14:20:00`	`14:07:00`
s3	`2022-07-19 14:20:00-15:05:00`	`15:36:00`
s4	`2022-07-20 00:31:00-01:00:00`	`00:52:00`
m1	`2022-07-19 16:00:00-17:50:00`	`17:20:00`
m2	`2022-07-19 19:32:00-20:02:00`	`19:44:00`
m3	`2022-07-19 20:06:00-20:40:00`	`20:28:00`
m4	`2022-07-19 22:31:00-23:04:00`	`22:49:00`
m5	`2022-07-19 23:16:00-23:46:00`	`23:38:00`
m6	`2022-07-19 23:54:00-00:27:00`	`00:10:00`

TABLE 4: **Attack Timestamps**. UTC Timestamps for attack windows and initial attack step.

4. Experimental Setup

Machine Instrumentation We execute `ATLASv2` using two machines running Windows 7 32-bit VMs with VMware Workstation Pro. We commonly refer to these two machines as host 1 (h1) and host 2 (h2). During the attack scenarios on the fifth day of the data collection period, the user of the host 2 machine runs Kali Linux outside of the Windows 7 VM to execute the attack scenarios.

The Windows VMs were instrumented with Mozilla Firefox 52.0, Microsoft Office Professional Plus 2010, Adobe Flash Player 17.0.0.188 and 18.0.0.194 (varied by attack scenario), and included Carbon Black sensors v.3.8.0.627 with data-forwarders set to an AWS S3 bucket.

Benign Data Generation The first four days of the engagement records two researchers using the Windows VMs as their primary workstations approximately 8-hours per day, simulating a normal work day. At the conclusion of each workday, the VMs are left running overnight. As discussed in (§2), a primary focus of this work was the generation of *realistic* benign user data. This is accomplished by the two researchers assigned to the VMs manually generating all system activity through normal daily use. We assume all activity over this four-day period is benign.

Over the course of the benign period, the two researchers used a variety of applications from their usual work behavior such as web browsing, attending Zoom meetings, working on projects using Microsoft Office applications, sending and receiving e-mail attachments, chatting on Discord, watching videos, and other normal user behavior. The user on the h1 machine also ran a SimpleHTTP server implemented in Python to simulate an internal web portal. This server is then exploited in the multi-host attack scenarios to achieve lateral movement by the attacker to leverage the initial compromise of the h1 machine to then compromise the h2 machine.

Malicious Data Generation Malicious activity begins with the initial attack step for each attack scenario. The timestamp of each initial attack step is recorded in Table 4. We deploy each attack scenario over a set time ranging from 30 minutes to 2 hours. During each attack scenario, both users continue manually generating benign system use. We randomize the attack initiation for each time period. When an attack begins, the user of the h2 machine briefly transitions to the Kali Linux machine and executes the steps according to Figure 1. When the attack on h1 has been fully carried out, the user of h2 transitions back to the Windows VM and continues benign use. At another random time prior to the scheduled end of the scenario, the user of h2 visits h1's webpage and the attack is carried out on h2.

5. Evaluation

The five day `ATLASv2` engagement generated 154 GB of data. This includes raw logs from the five logging frameworks in the instrumentation on each machine. All log files were copied from the VM to the host machine at the conclusion of each work day during the engagement. All Carbon Black Cloud data was sent to an AWS S3 bucket using data forwarding throughout the course of the engagement.

5.1. Volume of System-Level Security Logs

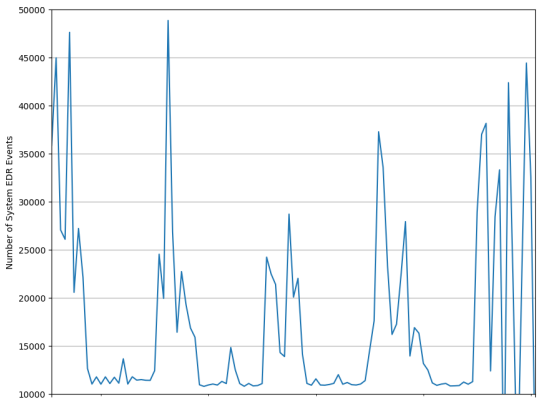

Figure 2: **Total EDR System Call Events Over 5-day Engagement.** View of the total number of system call events logged by the EDR system during the engagement.

One interesting result from the `ATLASv2` engagement is the volume of system-level security auditing data. Over

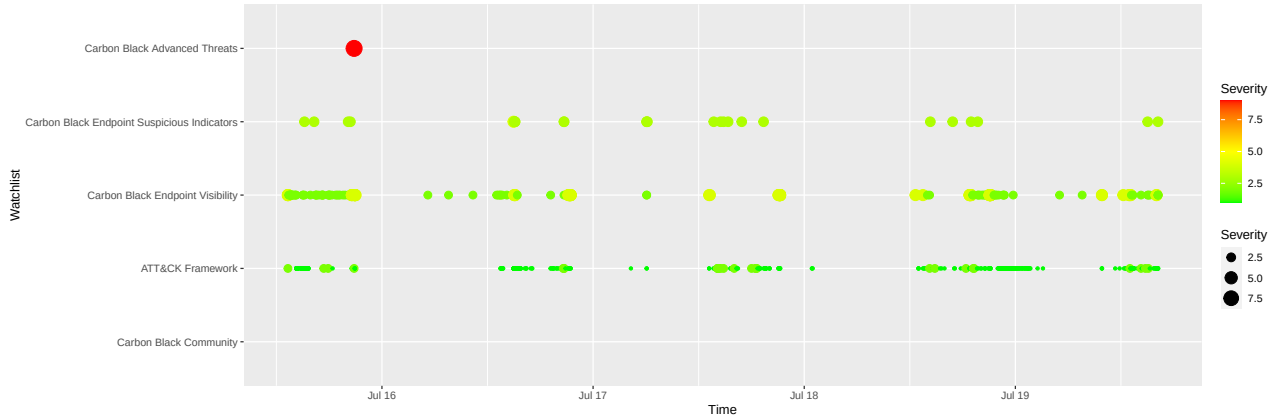

Figure 3: **Benign Period Watchlist Hits.** Frequency of triggered watchlist queries and their severity scores for both hosts during the four-day benign period.

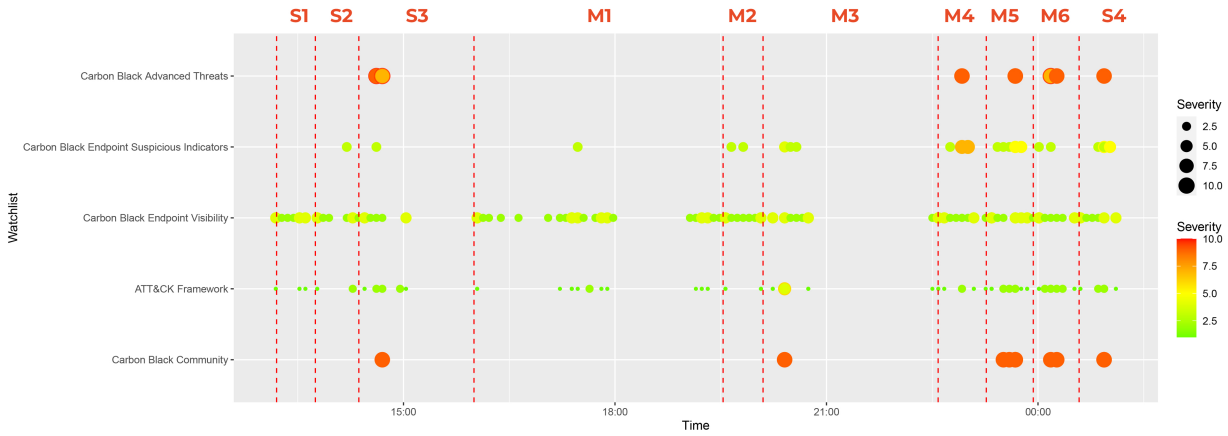

Figure 4: **Attack-Day Watchlist Hits.** Frequency of triggered watchlist queries and their severity scores for both hosts across the ten attack scenarios.

the five-day engagement on the two host machines, the volume of security log data illustrates the need for effective tools to analyze and manage the data. Extracting relevant attack behavior from such a high volume of data is a challenge, and having high-quality public datasets to aid in the creation of such tools helps to improve researchers' ability to assess the quality of their tools.

In Figure 2 we have an illustration of the system calls made during the entire ATLASv2 engagement period under a single logging framework (the Carbon Black Cloud EDR system). This logging framework contained almost two million system security events over the course of the engagement.

5.2. Case Study: ATLASv2 For EDR Analysis

Security analysts who use EDR systems may encounter a number of challenges and difficulties. One common issue is the large volume of alerts and notifications that these systems can generate, which can make it difficult for analysts to prioritize and focus on the most important threats. Additionally, EDR systems may generate false positives, where the system identifies a potential threat that is actually harmless, which can waste analysts' time and effort. To prioritize their time for alert triage, analysts can use a variety of techniques, such as prioritizing alerts based on their severity or potential impact and using automated tools to filter and classify alerts.

One important use for the ATLASv2 dataset is for analyzing these EDR system alerts. One of the logging frameworks included in the ATLASv2 machine instrumentation was the Carbon Black Cloud EDR system which provides system-level logging events as well as alerts from the Carbon Black attack watchlists. These watchlists correspond to known attack behaviors seen in attacks and in government and third-party databases.

Watchlist Classication The watchlist hits from the Carbon Black Cloud system are used to alert security analysts to potential attack behaviors on sytems with the EDR sensor installed. There are five different categories the CBC system draws from:

- Carbon Black Advanced Threats
- Carbon Black Ednpoint Suspicious Indicators
- Carbon Black Endpoint Visibility
- MITRE ATT&CK Framework
- Carbon Black Community

These watchlists give a rough indication of the likely severity of the alert, with alerts on the Carbon Black Advanced Threats considered the most urgent. However, each alert is also given a severity score to help analysts assess which alerts to investigate.

False Positives During Benign Activity To get an idea of the volume of alerts the system generates, in Figure 3 we consider the alerts generated during the benign activity of ATLASv2. By assumption, all of these are false alarms, since all activity during this period is benign.

The data indicate a high volume of alerts in three of the Carbon Black watchlist categories (Suspicious Indicators, Endpoint Visibility, and the ATT&CK Framework). Examining these alerts in the ATLASv2 dataset gives an indication of the problem of alert fatigue. The four days of benign data contain over 400 watchlist alerts. The overall alert severity scale employed by Carbon Black is a nine-point scale, with the most severe alerts raised during benign activity registering at as a five.

Wathclist Hits on Attack Day Next, we examine the volume and severity of watchlist alerts fired by the Carbon Black system during the fifth and final day of the ATLASv2 engagement.
Figure 4 shows the alerts fired during the attack phase of the engagement. The number of high severity alerts is clearly higher, indicating the attack behavior triggers alerts from the EDR system. However, we can see that several of the attacks (S1, S2, M1, M2, and M3) generated very few high severity alerts as measured by the severity scale, or by their inclusion on the Carbon Black Advanced Threats watchlist.

This suggests that for a security operations center to investigate behavior associated with all attacks, they must investigate lower severity alerts as well as high severity alerts. This demonstrates the need for automated tools to help improve the alert triage task and improve the accuracy of severity indicators to help security analysts determine which EDR alerts are more likely to indicate actual attack behavior. Without such tools, many potentially true positives will remain not be investigated because of the volume of false positives these systems produce [9].

6. Conclusion

This work proposed ATLASv2, a new (soon-to-be) publically available dataset for intrusion detction research. Audit logs are an important part of computer system security and can provide valuable information for security analysts and for automated tools to detect system breaches. However, the large volume of data and alerts generated by most systems makes it difficult for analysts to effectively prioritize and respond to potential threats. Machine learning techniques, such as deep learning algorithms, can be used to help isolate and classify alerts and system log events, which can improve the accuracy and efficiency of intrusion detection. However, the availability and quality of publicly available system logging datasets is a significant obstacle to the development and evaluation of these systems. High-quality datasets are essential for training and testing machine learning algorithms, and the lack of such datasets can limit the effectiveness of these technologies in intrusion detection.

While ATLASv2 provides an upgrade on existing datasets, we also caution about the limitations on its use. The ATLASv2 dataset still suffers from a great deal of commonality between the attack scenarios it implements. This means that the dataset is ideal for unsupervised anomaly detection and for evaluation of EDR systems on alert triage. However, it should not be used in a supervised training context since the similarity of attack data does not represent the variety of attacks an EDR system will encounter in production.

With ATLASv2 we present a dataset that makes significant upgrades to the existing publicly available datasets, particularly in the area of the realism of benign data, and in the integration of attack scenarios into normal user activity.

References

[1] Georgios P. Spathoulas and Sokratis K. Katsikas. Using a fuzzy inference system to reduce false positives in intrusion detection. In *International Conference on Systems, Signals and Image Processing*, 2009.

[2] Yang Lv, Shaona Qin, Zifeng Zhu, Zhuocheng Yu, Shudong Li, and Weihong Han. A review of provenance graph based apt attack detection:applications and developments. pages 498–505. Institute of Electrical and Electronics Engineers Inc., 2022.

[3] Fargana J. Abdullayeva. Advanced persistent threat attack detection method in cloud computing based on autoencoder and softmax regression algorithm. *Array*, page 100067, 7.

[4] Geeta Kocher and Gulshan Kumar. Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges. *Soft Computing*, 25:9731–9763, 8 2021.

[5] Muhammad Adil Inam, Yinfang Chen, Akul Goyal, Jason Liu, Jason Mink, Noor Michael, Sneha Gaur, Adam Bates, and Wajih Ul Hassan. Sok: History is a vast early warning system: Auditing the provenance of system intrusions. 2023.

[6] James P Anderson. Computer security technology planning study (volume ii), 1972.

[7] Zhenyuan Li, Qi Alfred Chen, Runqing Yang, Yan Chen, and Wei Ruan. Threat detection and investigation with system-level provenance graphs: A survey. *Computers Security*, 106:102282, 2021.

[8] Md Nahid Hossain, Sadegh M. Milajerdi, Junao Wang, Birhanu Eshete, Rigel Gjomemo, R. Sekar, Scott D. Stoller, and V. N. Venkatakrishnan. Sleuth: Real-time attack scenario reconstruction from cots audit data. In *USENIX Conference on Security Symposium*, 2017.

[9] 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. In *Network and Distributed Systems Security Symposium*, 2019.

[10] Xueyuan Han, Thomas Pasquier, Adam Bates, James Mickens, and Margo Seltzer. Unicorn: Runtime provenance-based detector for advanced persistent threats. In *ISOC Network and Distributed System Security Symposium*, 2020.

[11] 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 *USENIX Security Symposium*, 2021.

[12] Su Wang, Zhiliang Wang, Tao Zhou, Xia Yin, Dongqi Han, Han Zhang, Hongbin Sun, Xingang Shi, and Jiahai Yang. threatrace: Detecting and tracing host-based threats in node level through provenance graph learning. *IEEE Transactions on Information Forensics and Security*, 2022.

[13] Robert A. Bridges, Tarrah R. Glass-Vanderlan, Michael D. Iannacone, Maria S. Vincent, and Qian (Guenevere) Chen. A survey of intrusion detection systems leveraging host data. *ACM Comput. Surv.*, nov 2019.

[14] Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers and Security*, 31:357–374, 5 2012.

[15] CVE. CVE-2015-5122. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5122.

[16] CVE. CVE-2015-5119. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5119.

[17] CVE. CVE-2015-3105. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3105.

[18] CVE. CVE-2018-8174. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-8174.

[19] CVE. CVE-2017-0199. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0199.

[20] CVE. CVE-2017-11882. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-11882.