CasinoLimit: An Offensive Dataset Labeled with MITRE ATT&CK Techniques

Sébastien Kilian
CentraleSupélec, Inria, IRISA
Rennes, France
sebastien.kilian@inria.fr

Valérie Viet Triem Tong
CentraleSupélec, Inria, IRISA
Rennes, France
valerie.viettriemtong@inria.fr

Jean-François Lalande
CentraleSupélec, Inria, IRISA
Rennes, France
jean-francois.lalande@inria.fr

Frédéric Majorczyk
DGA, CentraleSupélec, Inria, IRISA
Rennes, France
frederic.majorczyk@def.gouv.fr

Alexandre Sanchez
Inria, CentraleSupélec, IRISA
Rennes, France
alexandre.sanchez@inria.fr

Natan Talon
Hackuity, CentraleSupélec, Inria, IRISA
Rennes, France
natan.talon@centralesupelec.fr

Pierre-Victor Besson
CentraleSupélec, Inria, IRISA
Rennes, France
pierre-victor.besson@inria.fr

Hélène Orsini
CentraleSupélec, Inria, IRISA
Rennes, France
helene.orsini@centralesupelec.fr

Pierre Lledo
DGA, CentraleSupélec, Inria, IRISA
Rennes, France
pierre.lledo@centralesupelec.fr

Pierre-François Gimenez
Inria, CentraleSupélec, IRISA
Rennes, France
pierre-francois.gimenez@inria.fr

***Abstract*—Cybersecurity exercises are a common way to train and evaluate the skills of cybersecurity professionals. These exercises also provide a unique opportunity to generate datasets with realistic attack traces on non-sensitive systems. Nevertheless, the collected logs are unlabeled, and deciding which logs are related to pentesters is a difficult problem. In this paper, we present a novel methodology to label efficiently both system and network logs using MITRE ATT&CK techniques. To demonstrate the effectiveness of our approach, we introduce CasinoLimit, a dataset generated from a pentest exercise that has been played by 114 participants where we collected 540 GB of attack data. We apply our methodology to accurately label these logs with a semi-automatic approach: labels are inferred from the shell sessions and propagated to the network sessions, and eventually corrected by a junior analyst. An expert analyst has manually reviewed all the labels that have been computed to ensure the quality of the labeling process. The results of the pentest exercise are deeply discussed. We show the variability of players' behaviors and that players can be distinguished by their command line habits. In addition, the high level of granularity of labels coupled with the number of participants enables multiple other applications. With this paper, we release the full dataset and the associated labeling tool, MANATEE, which can be used to browse the logs and labels. To support the generalization of our approach, we made it possible to load other datasets with this tool.**

***Index Terms*—Cybersecurity, Dataset, Labeling, MITRE ATT&CK, Pentest**

I. INTRODUCTION

Attack data refers to information that captures the actions an attacker has performed on an object, a machine, or an information system. Such data are crucial for detecting, analyzing and preventing malicious activities. They are typically collected through the monitoring of the targeted object or network and include application logs, system logs, and network traffic.

However, researchers and practitioners have limited access to real-world attack data for several reasons. First, attack data are often incomplete, either because the system was only partially monitored or because the attacker successfully evaded detection. Additionally, precisely identifying all the attacker's activities is challenging, as malicious actions can easily be mistaken for benign or legitimate actions [2]. Finally, even when these challenges are overcome, organizations and administrations that possess such attack data may be reluctant to share them due to confidentiality or privacy concerns.

Despite these limitations, a few datasets containing attack data are available and are highly valuable to the scientific community [31], [28], [4]. These datasets typically include either network and/or system logs from fully automated botnet-type attacks or logs of attacks conducted by a limited number of human actors with varying technical expertise, ranging from cybersecurity students to professional ethical hackers [19]. Many applications of cybersecurity datasets such as attacker profiling [3] require them to be accurately labeled to have a deep understanding of the attacker's actions [13].

Labeling attack data remains a fundamental challenge [17], [20]. It requires specialized knowledge and a deep understanding of attack techniques, making the process highly time-consuming. The availability of qualified experts is limited, as they are often engaged in other responsibilities. In addition,

when an attack on a monitored system is completely unknown, labeling becomes technically impossible. Consequently, researchers often reproduce attacks in controlled environments, as this simplifies the work of experts responsible for labeling the logs [35]. However, the scale of the information system used in these experiments is usually limited to a few hosts, and the simulated attacks are often scripted [11]. This lack of variability constrains the range of actions performed and, consequently, the diversity of the collected data.

Ultimately, the research community faces three fundamental challenges: (1) the lack of datasets that capture the full spectrum of attacker expertise and techniques, (2) the difficulty of ensuring accurate and precise labeling of attack data, and (3) the lack of a standardized labeling methodology that ensures semantic consistency across datasets.

In this paper, we address these issues by introducing CasinoLimit a novel attack dataset extracted from an infrastructure intrusion challenge conducted during the largest in-person CTF event in France. This challenge was played by 120 teams of five players each, allowing us to observe multiple attackers attempting to compromise the same infrastructure. For ensuring the non-interference between players, we deployed an isolated infrastructure for each team in the cloud. Since it is not possible to know how many players were engaged in the challenge for each team because of privacy reasons, we refer to them as "the player". Nevertheless, we know that at least one (and usually only one) person played in each of the 114 selected infrastructures, allowing us to collect 540 GB of attack data during the event.

We labeled the collected dataset using a hybrid methodology, combining automated and manual review. System logs were labeled automatically, with ambiguous cases reviewed manually, while labels on network log were inferred using timestamps and target information. Rather than relying on a coarse-grained benign/malicious classification, we employ the MITRE ATT&CK matrix [21] as a semantic framework to annotate observed actions. This framework provides a standardized vocabulary for describing techniques, i.e. unitary actions that can be performed by an attacker on a system, grouped by high-level objectives called tactics.

In total, $4{,}896{,}255$ events, defined as individual actions or occurrences captured in the logs, were labeled with a double-review from a junior and senior expert. These labels cover 98.45% of the attackers' actions in shell sessions and consequently, mark 11.70% of network flows. Beyond dataset creation, we also present an in-depth analysis of player behavior, examining attacker profiles, stealthiness, and the various strategies employed throughout the competition. We leverage this dataset to construct an empirical Cyber Kill Chain that reveals discrepancies with traditional representations of linear attack sequences. Finally, we provide empirical evidence that players can be distinguished based on the usage of a minimal set of shell commands.

The remainder of this paper is structured as follows. Section II reviews existing labeled attack datasets and highlights their limitations. Section III introduces the CasinoLimit challenge, the controlled attack environment and data collection process. Section IV describes our labeling methodology and the validation process. Section V gives a general analysis of the collected attack data. Section VI explores the feasibility of attacker fingerprinting based on tool usage patterns. Section VII discusses the limitations of our approach and potential biases in the dataset.

II. Related Work

Datasets containing attack data serve multiple purposes, including the evaluation of intrusion detection systems (IDS) [29], [22], attacker profiling [8], [3] and attack path predictions [34]. In [12], Goldschmidt *et al.* conduct a systematic comparative survey of these public datasets published from 1989 until the end of 2023. They identify over 89 datasets, the vast majority of which contain only network traffic, typically provided in the form of raw packet captures (PCAPs), which are well-suited for evaluating Network Intrusion Detection Systems (NIDS). Only 18 of these datasets capture attacker activity from both the network (packets or flows) and end-host systems (e.g., logs, system calls, process trees). However, comprehensive understanding of adversarial behavior requires visibility into both levels—network and host—to accurately reconstruct attack scenarios and analyze the progression of multi-stage intrusions.

Among these, only DAPT-2020 [22], CUPID [18], PWNJUTSU [4], Unraveled [23], and the CyberForce scenarios [33] document human-driven attack activity—that is, non-botnet and non-simulated behaviors—and provide insight into how attackers operate within the target information system.

In all these datasets, attacker behavior originates from penetration testers, ethical hackers, or Capture the Flag (CTF) participants acting in controlled or competitive settings. Although the main goal of such exercises is often educational or competitive [25], they have also proven useful for generating labeled attack data. For example, Taylor *et al.* [32] propose a framework to design CTF challenges and collect participant activity, which can later be used for profiling attackers [3] or building threat models.

The main characteristics of these datasets are compared in Table I.

These datasets become substantially more valuable when they are precisely and accurately labeled [17], since annotations are essential for training supervised learning models and for conducting reliable performance evaluations. CUPID and DAPT-2020 use binary labels to distinguish malicious from benign activities, whereas Unraveled relies on fine-grained annotations based on MITRE ATT&CK techniques to characterize attackers behaviors.

MITRE introduced the ATT&CK matrix in 2013—a knowledge base of adversary tactics and techniques, grounded in real-world observations, that systematically documents and categorizes attacker behavior across the attack lifecycle [30]. Tactics denote why an attacker performs an action—that is, the short-term objectives guiding adversarial behavior—while techniques describe how these objectives

TABLE I: Attack-focused datasets comparison

	CasinoLimit (Ours)	**Unraveled** [23]	**DAPT-2020** [22]	**CUPID** [18]	**PWNJUTSU** [4]	**CyberForce scenarios** [33]
Different attackers	**114**	3	1	10	22	1
Tactics	**14**	5	4	Not applicable	5	4
Techniques	**67**	15	16	Not applicable	13	10
Label granularity	MITRE ATT&CK	MITRE ATT&CK	Binary	Binary	None	None
Attack duration	10h	**4 weeks**	4 days	Multiple days	24 hours	2.5 hours

are achieved. ATT&CK also provides examples of procedures, which are specific implementations of techniques. This framework has gained significant traction in the cybersecurity community, serving as a common language for describing adversarial behavior. Thus, labeling data using the ATT&CK framework allows for a large variety of applications [1].

One way to label logs from the monitoring of attacked infrastructures is to use predefined log filtering rules [26]. This approach relies on the fact that some attacks have a specific pattern that can be detected by looking for specific signatures in the logs. However, this approach is limited to known attacks and the signature should carefully be written to match the payload in any configuration of the infrastructure. This is a major drawback when considering real attackers who use new techniques or variants.

When precise information about the attack is available, it becomes possible to use this information to label the logs. In [16], the authors differentiate malicious activities from benign activities when the time periods of the attacks have been previously identified. This approach works well when the nature of the attack is known in advance, or when the experimenter controls the start and end of the attack, for example using scripts [11]. By generating additional reports during the attack phase, the logs can be labeled based on the timestamps and the nature of the attack.

In this paper, we adopt a similar strategy but under more realistic conditions: attack paths are executed by 114 competitors participating in a pentest exercise, our dataset is precisely labeled with MITRE ATT&CK techniques through a rigorous and reproducible methodology assisted by a dedicated tool.

III. The CasinoLimit Challenge

In this section, we present how we designed the CL challenge to address the limitations of discussed datasets of Table I. We aim to have a large variety of attack behaviors by having an attack scenario that requires players to display a wide range of tactics and techniques. Additionally, replicating the challenge across multiple teams not only enhances the variety of observed behaviors but also enables their comparison.

a) Ethical considerations: Before the challenge, all participants were informed about the monitoring of their activities and the collection of logs. Then, they explicitly consented to be monitored and to accept that their data would be published as part of scientific research. This process was reviewed and approved by the legal department of our institution. Additionally, any personally identifiable information (PII) was removed from the dataset to ensure the privacy of participants.

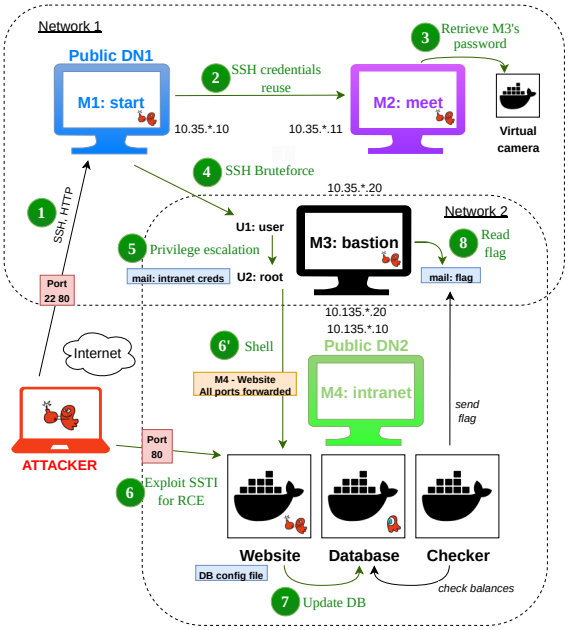

Fig. 1: Overview of the CasinoLimit Scenario.

b) Deployment: A total of 700 virtual machines were deployed (1120 CPU, 4.5 TB RAM) in the cloud for this challenge. Each of the 120 teams was allocated **an instance** *i.e.* an isolated environment, ensuring identical conditions across all participants. An instance is composed of a gateway, 2 hosts running Debian 12 (*start* and *meetingcam*), and 2 hosts running Ubuntu 22.04 (*bastion* and *intranet*). These hosts were provisioned with a specific set of services and vulnerabilities. This was made possible using a dedicated software for creating these challenges [6] from a given set of vulnerabilities. To facilitate further research and reproducibility, the entire infrastructure used in this challenge is fully re-deployable. All necessary materials, including deployment scripts, configuration files, and documentation, are publicly available[1].

c) Scenario overview: The story that supports this challenge is based on a casino that does not respect the right to have personal data erased. The objective assigned to the participants was to infiltrate the casino's network from a compromised host, gain access to a web application, and successfully remove the data of a single designated user without

[1] https://gitlab.inria.fr/pirat-public/casinolimit

TABLE II: Techniques required to solve the scenario

Machine	Required Technique	Attacker action
Start	T1021: Remote Services	connects to the machine using SSH
Start	T1021: Remote Services	reuses the password to connect to the meetingcam machine
Meetingcam	T1125: Video Capture	uses the webcam to take a picture of parts of the password for tocean@bastion
Meetingcam	T1021: Remote Services	bruteforces the missing characters of the password for tocean@bastion and connects to the bastion machine
Bastion	T1068: Exploitation for Privilege Escalation	exploits a vulnerability on the bastion machine to obtain root access
Intranet	T1190: Exploit Public-Facing Application	exploits an SSTI vulnerability on the intranet site to update the database
Intranet	T1485: Data Destruction	deletes the data of a specific user from the database
Bastion	T1114: Email Collection	reads the flag in the emails

disrupting the main functionalities of the web application and deleting other users' data. The scenario required players to navigate through four different machines and to impersonate multiple users with varying levels of privileges.

This challenge was structured around two interconnected networks, as illustrated in Figure 1:

- Network 1: a perimeter network containing a webcam service.
- Network 2: an intranet network hosting a website accessible from the internet.

The penetration testing aspect of the challenge was designed to be performed entirely through command-line interactions on Linux-based systems, without any graphical user interfaces other than the website. Teams were given only their final goal, but the environment itself provided clues to guide them through the infiltration process. These hints were embedded in various artifacts within the system, such as a verbose yet imprecise penetration test report, photos from a motorized webcam, and dynamic email exchanges. Similar to a real information system, participants had to extract actionable intelligence to determine valuable actions.

d) Step-by-step attack progression:

1) **Initial access on** `start` **(M1).** Participants were provided with the URL of the `start` host (M1) in Network 1, which had a public IP address accessible from the internet. Using the supplied credentials for the user `tbenedict`, players could establish an SSH connection to the M1 machine.
2) **Lateral movement to** `meetingcam` **(M2).** On `start` (M1), players needed to discover the presence of two additional hosts: `meetingcam` (M2) and `bastion` (M3) through network scanning. Once identified, they could reuse the same credentials to access `meetingcam`.
3) **Intelligence gathering on** `meetingcam` **(M2).** Players discovered that `meetingcam` hosted a motorized camera system that was filming a physical whiteboard in a meeting room. By inspecting system configuration files, they determined how to interact with the camera and take snapshots. The challenge required them to capture an image of the whiteboard and transfer it back to their machine, since no graphical interface was available. They obtained two pieces of information : 1) A partially erased password for the user `tocean` on `bastion` (M3); 2) A note stating that the kernel of `bastion` was outdated and should be patched, hinting that it was vulnerable to CVE-2023-0386 (privilege escalation vulnerability).
4) **Lateral movement to** `bastion` **(M3).** Since the password for `tocean` was incomplete, players needed to brute-force the missing character to successfully authenticate on `bastion` via SSH. The SSH service was deliberately misconfigured, making it vulnerable to brute-force attacks. Consequently, players had access to the second internal network (Network 2) and the email service on `bastion`.
5) **Privilege escalation on** `bastion` **(M3).** To escalate privileges, participants exploited CVE-2023-0386, a Linux kernel vulnerability that allowed privilege escalation to root. With root access, they gained access to emails explaining how to create new user accounts on the website.
6) **Web exploitation on** `website` **(M4).** Inside the web application, an SSTI (Server-Side Template Injection) vulnerability enabled the players to have access to the website's server and execute arbitrary commands. Optionally, they could also establish a reverse shell to interact with the system.
7) **Data deletion.** Using this backend access but with few tools due to the docker environment, players had to craft SQL queries to delete the data of a specific user from the database, which would trigger the system to send an email containing the flag to the `admin` user.
8) **Flag retrieval.** Finally, players had to access the email service on `bastion` to retrieve the email sent to the `admin` user that contains the confirmation of data deletion and the flag.

As a summary, we report in Table II the seven main actions that should be performed in the scenario. For each global action, we associate the most relevant MITRE ATT&CK technique [21].

e) Attack data collection: All of the 700 hosts were monitored to capture the actions performed by players. However, as shown in Figure 3, not all teams were actively engaged in the challenge. Even if only 99 teams out of 120 effectively interacted with their instance, a substantial amount of data was collected. Because some players had issues or performed actions that blocked their progression in the challenge, we provided 15 fresh instances to them. In total, we collected logs from 114 instances from which we collected: 1) System logs: Kernel and user-space events recorded via *auditd* and *syslog*. Particularly, all processes executed interactively or by a script are logged with their associated arguments; 2) Network traffic: Full packet captures and intrusion detection logs using *Suricata*, deployed in monitoring mode.

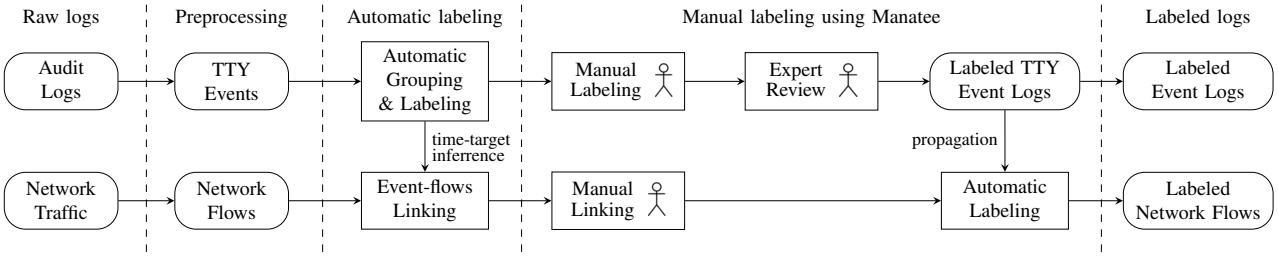

Fig. 2: Labeling process overview.

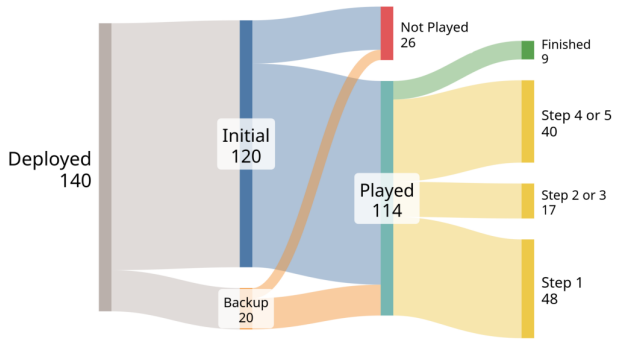

Fig. 3: Deployed challenge instances compared to the number of players that reached each step.

TABLE III: Summary of the log preprocessing

		Raw logs	Preprocessing
Audit Logs	processes	3,357,781	404,637
	events	25,325,145	5,037,467
Network Traffic	flows	99,075,828	52,340,202
	packets	1,171,224,555	369,669,619

These logs were collected from the four hosts of each instance, resulting in a total of 40 GB of system logs and 500 GB of network traffic. Even if this volume is small compared to other cybersecurity datasets, the data contain a large number of logs related to players' actions. This large quantity of attack activities is not usual and requires a specific methodology to be analyzed.

Time	Command	Label
02:34:12	ss -natp	T1049: System Network Connections Discovery
02:35:52	./linpeas.sh	T1082: System Information Discovery
02:35:52	grep -i password	
02:35:52	(...)	
02:36:47	nmap -sC -sV 10.25.141.0/24	T1046: Network Service Discovery
02:36:53	nmap -sC -sV 10.25.141.0/24	
02:39:18	mutt –help	T1518: Software Discovery
02:39:23	mutt -help	

Log 1: Example of the labeling process

IV. Labeling Methodology

In this section, we present our proposed labeling methodology for the CasinoLimit dataset. The goal of this methodology is to provide a comprehensive and accurate labeling of the attack data, which is crucial for several applications. Our objective is to add a high level of detail to the labels, going beyond a simple "malicious/benign" classification, which would have been almost trivial given that nearly all observed activities originated from players actively attacking the infrastructure. The only benign traffic was coming from the internet and automated background services (*e.g.*, NTP traffic). Instead, we adopted a fine-grained labeling approach based on MITRE ATT&CK techniques, ensuring that system and network logs are categorized at the technique level. This methodology enhances the dataset's reusability for future research, including the evaluation of detection models and the training of learning-based approaches.

To achieve this, we introduce a labeling methodology that combines both automated and manual processes. This approach aims to label 100% of user actions originating from a terminal and to maximize the labeling coverage of network events.

a) Overview: The overview of our methodology is summarized in Figure 2. Raw logs are first pre-processed to extract events related to terminal interactions. Events are also grouped into processes, which simplifies greatly the labeling procedure. This divides by 12 the number of objects to label, as shown in Table III. Also, network traffic has been represented as flows, which are more convenient to label than raw packets, and packets that are not in the time frame of the exercise are not considered for labeling.

Then, the processes are labeled automatically by recognizing known commands. Eventually labels on network events are added if the command generates network flows. Finally, a two-phases manual review is performed by a junior analyst (29h) and then a senior analyst (8h) who can investigate the tricky cases. The link between a process and its events is kept so that process labels can be conveniently associated with events. Similarly, network flows can be traced back to their constituent packets using the same approach.

b) System Logs Labeling: Labeling auditd logs is a challenging task because of the large volume of diverse events. In order to take into consideration relevant events, only events that are related to the shell sessions are kept since they are

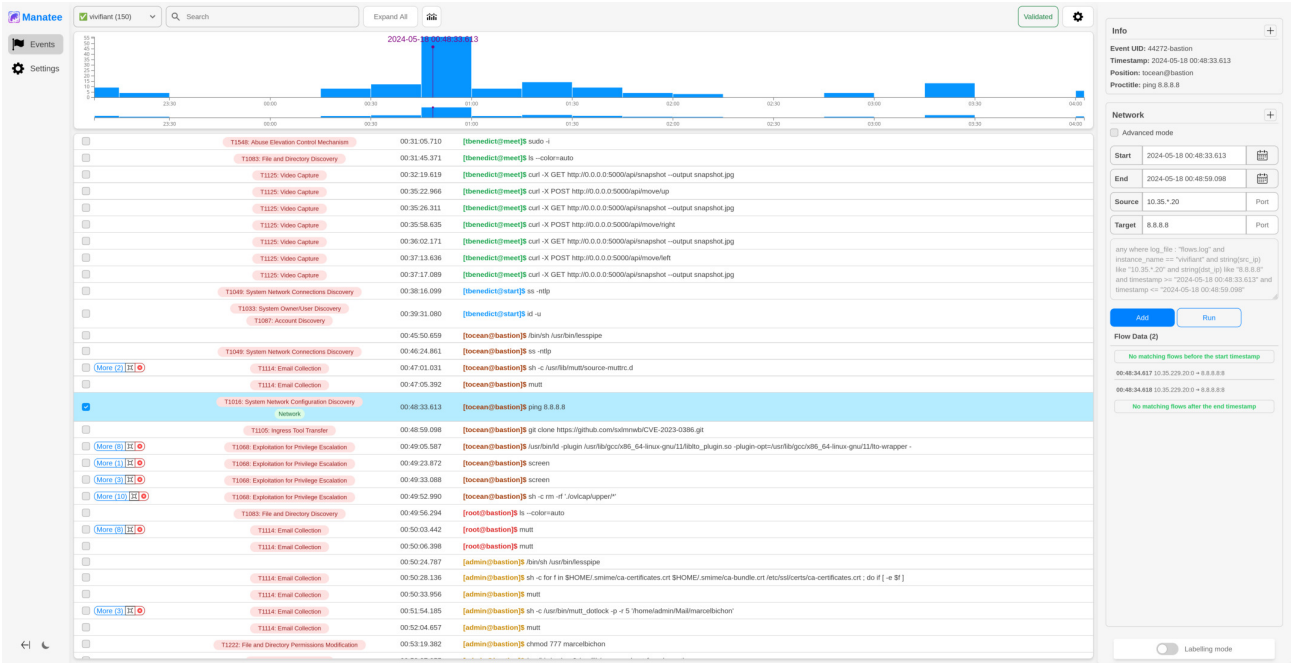

Fig. 4: Web interface of MANATEE, a tool for labeling system and network events with our methodology.

sufficient to understand the behavior of the attacker. These events are named TTY events and are the result of the player's actions. Since the players start the challenge on a monitored machine, we can consider that all players' actions will be monitored. For example, when performing a network scan, even if the consequences of the scan on the target machine will not be considered and labeled, the scan itself will be labeled on the sender machine. Then, these events are grouped by processes, allowing all the events generated by a command to be associated with a single process. This results in a sequence of commands which is an efficient representation of what players did. The next step is to add a semantic label to these commands with the corresponding technique.

MITRE ATT&CK techniques are first automatically inferred using a mapping between a command and its corresponding technique depicted in Table VII in Appendix. Then, the labels are manually reviewed to ensure the quality of the labeling in a tool specifically designed for this purpose: MANATEE[2].

As shown in Figure 4, a junior analyst can browse the system logs (center of the figure), identify the host where they occurred and with which user identity (`tocean` for the highlighted line), the time, and the proposed MITRE technique (red). Their task is to review any errors in the matching by understanding the intent behind the command. For example, in Log 1, the mutt command is labeled automatically with T1114 (Email Collection), but the analyst decides that the usage of the parameter *–help* requires labeling this command with T1517 (Software discovery).

[2]Available at https://gitlab.inria.fr/pirat-public/manatee

Even after filtering for TTY events and grouping by processes, some commands can still produce a lot of subsidiary sub-processes. As an example, if an attacker executes a specialized script from a single command (one process), numerous events will occur, as shown in Log 1 for the execution of *./linepeas.sh* (several sub-processes). Thus, we chose to visually group processes if the time between events is less than 1 second on the MANATEE interface. A group of processes can be unfolded in order to label them individually if needed, or selected to label them all at once.

c) Network Traffic Labeling: Given that the network data has a much larger volume than the system logs, we propose a different approach to label them. The objective is to associate the network events with the system commands that generated them, in order to propagate the labels from the system to the network events. Among the commands used by the attackers, those that produce network events are automatically identified and tagged using regular expressions. Additionally, analysts can add additional commands that our regular expressions have missed. For each of these commands, a temporal query is generated in EQL format [10] to select relevant network flows from an ElasticSearch server. The query is composed of the time window, the source IP, and the destination IP or IP range. On the one hand, the source IP is often not provided in the command, so this information is deduced from the machine where the command was executed. On the other hand, the destination IP or its hostname can be found most of the time in the command directly. This can also be IP ranges or regular expressions, which are conveniently handled by EQL queries. The time window is automatically set to start

TABLE IV: Summary of created labels, labeled processes and labeled events in the dataset

		Automatic Labeling	After manual labeling: Junior	After manual labeling: Expert Add	Change	Sub	Number of labels	Labeled at least once	Input data	Coverage
Audit Logs	labels	5152	8872	+465	438	-94	9243	9243		
	processes	5152	410,861	+3056	2810	-1416	412,501	399,532	404,637	98.74%
	events	4,063,695	6,937,978	+211,108	106,766	-61,050	7,088,036	4,896,255	5,037,467	97.20%
Network Traffic	labels	453	7118	+172	19	-35	7255	952		
	flows	11,721,688	11,927,000	+966,647	14,799	-40,531	12,853,116	6,122,221	52,340,202	11.70%

at the timestamp of the command and end at the timestamp of the next command. As an example, on the right part of Figure 4, MANATEE automatically inferred the EQL query for the command `ping 8.8.8.8`. The source IP is a regular expression matching the IP of *bastion* where the command was executed, and the destination IP is *8.8.8.8*. The timestamps are set to the time of the command and the next command, which is the `git` command in this case. These parameters are the most important, and can easily be modified using the MANATEE interface. Then, the analyst can visualize the selected network logs matched by this query and check if the time window is correct, as displayed on Figure 4.

Thanks to this association, network flows are labeled with the same technique as the command that generated them. Additionally, a senior analyst is responsible for confirming the labels and checking the time window of the flows. They can also edit the EQL query directly and modify its structure if a more complex request should be performed. In difficult cases where the used command is unknown by the analyst and where the time window is difficult to estimate, the analyst should remove the label from the network logs to avoid any false positives. Even if most of the work is done automatically, this process still remains time-consuming.

d) Personal Identifiable Information (PII) removal: In order to ensure the privacy of the participants, all PII information, such as URLs to personal websites, is tagged by the analysts and removed from the logs. This process is integrated into the labeling tool.

e) Results: Table IV summarizes the volume of labeled events at each step of our methodology. A total of 9243 labels are assigned to sets of processes, 412,501 labeled processes, and 7,088,036 labeled events. Since some processes receive multiple labels, we compute the coverage of the labeling by counting the number of processes with at least one label. 98.74% of processes are labeled, leading to a coverage of 97.20% for TTY events. When considering the initial raw events, the coverage reaches 19.32% (cf. Table III). The obtained coverage for network flow labels derived from event labeling is 11.70%. This low coverage is explained by the fact that the attack data is only a small part of the network traffic. In addition, 67% of the network traffic originates from outside the challenge because two of the machines were exposed to the internet, either from players connecting to the challenge or from automated scans. This artificially inflates the number of network flows that are not related to the scenario and thus not labeled.

V. A DEEP DIVE INTO THE CASINOLIMIT DATASET

A. Diversity of the Collected Data

This dataset was generated by observing players in a competitive environment, where they operated on identical instances with the same objectives but without any constraints on the tools, code, or frameworks they could use. As a result, this dataset captures a broad diversity of attack techniques and procedures. In total, 114 players used 67 different techniques, distributed across all the tactics of MITRE $ATT\&CK$ matrix. Among them, only 6 tactics (*Discovery*, *Collection*, *Credential Access*, *Reconnaissance*, *Privilege Escalation*, and *Exfiltration*) were really necessary to effectively achieve the challenge's objective.

Figure 5 illustrates the distribution of actions taken by players throughout different stages of the attack scenario. The x-axis represents instances sorted by their total activity duration, while the y-axis shows the number of recorded actions per instance. Each bar is color-coded to distinguish between different steps in the attack progression. The figure highlights the diverse attacker behaviors present in the dataset. Some players abandoned their attempts early, while others demonstrated remarkable efficiency, reaching the final stage in less than two hours. In contrast, other players spent over nine hours in the reconnaissance phase of the challenge, executing more than 400 actions, yet never progressing beyond the first step. Additionally, some instances crashed due to players' actions, requiring them to be assigned new environments. These cases can be identified in the figure as bars that contain actions from Steps 3 and 4 but lack those from Steps 1 and 2, indicating that players resumed their attack mid-scenario rather than starting from the initial phases.

B. Prevalence of Discovery Tactic

Table V presents an overview of the tactics, techniques, and labeled events observed in the CasinoLimit dataset. The analysis of this table leads to the following observations.

First, the *Discovery* tactic is significantly more prevalent in player behavior. The target environment was designed to resemble a real-world information system, ensuring that the observed attack data closely reflect realistic adversarial behaviors. In this environment, as in real-world attacks, neither the player nor the attacker has prior knowledge of the system they are navigating. Consequently, they must gather information continuously to make progress. This necessity largely explains the frequent use of the *Discovery* tactic, which accounts for

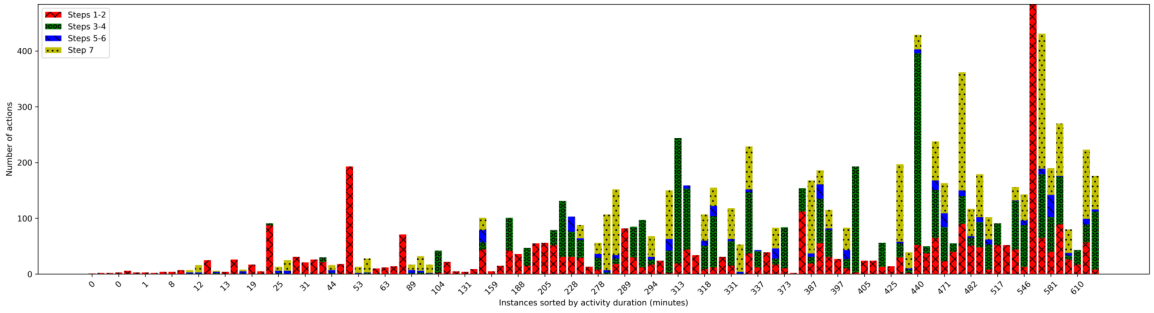

Fig. 5: Number of techniques performed by instance, sorted by activity duration.

TABLE V: Observed tactics, techniques, and labeled events in the CasinoLimit dataset

Tactic	Required	Techniques Used	Labels	Labeled Events	Labeled Flows
Reconnaissance	✓	3	902	2,074,904	5,906,628
Resource Development		2	178	1046	0
Initial Access		1	2	6	0
Execution		3	81	32,154	0
Persistence		2	24	212	0
Privilege Escalation	✓	2	160	23,140	4
Defense Evasion		9	622	63,912	0
Credential Access	✓	7	90	73,096	0
Discovery	✓	**18**	5239	4,303,664	6,843,254
Lateral Movement		3	784	3092	95,404
Collection	✓	7	679	23,972	7758
Command and Control		5	343	488,221	66
Exfiltration	✓	3	133	594	2
Impact		2	6	23	0
Total	**6**	**67**	**9,243**	**7,088,036**	**12,853,116**

the largest number of labeled flows (6,166,677) and labeled events (4,303,664).

Second, certain constraints in the challenge design introduced a bias in the diversity of observed behaviors. Specifically, some steps were highly guided, limiting players' ability to explore alternative attack strategies. For instance, privilege escalation on the Bastion machine was explicitly structured, leading all players to follow the expected sequence of actions without deviation. As a result, while the dataset captures a wide range of techniques in less constrained phases, certain key steps exhibit low behavioral variability due to predefined attack paths: Persistence, Impact, Initial Access.

C. Stealthiness of Offensive Tools

Several offensive tools exist to automate all or part of the kill chain. Table VI provides an overview of tool usage across the different tactics employed. Once again, Discovery is the tactic for which players used the most tools. These tools can be categorized based on their purpose:

- Network Discovery: `nmap`, `ping`, `dig`, `curl`, `ifconfig`, `ngrok`, which are used to identify active hosts, network configurations, and exposed services.
- System Discovery: `find`, `linpeas`, `lse.sh`, `pspy64`, `PwnKit`, which are leveraged to enumerate files, processes, system configurations, and potential local privilege escalation vectors.

Some of these tools generate a significant number of traces, either at the network level (e.g., scanning tools like `nmap` and ping) or at the system level (e.g., enumeration tools like find and linpeas). This table reveals several attack strategies. Indeed, while certain tools serve the same purpose, they differ in terms of the number of generated events and system traces. For instance, `nmap` performs active network scanning by sending crafted packets to multiple ports and analyzing responses, which generates a large number of network events. To achieve the same objective, some players adopted a different approach: instead of scanning directly, they first looked up the known hosts in `/etc/hosts`, then used `ping` to verify which ones were reachable, and finally tested a small set of ports manually (e.g., 22, 44, 23, 25, 80). This second approach generates significantly less network traffic because it avoids sending probes to every possible IP address and does not scan all ports indiscriminately. However, this strategy comes with the risk of missing unknown hosts or services.

As expected for exfiltration of data and credential access, the Table VI reports few system events. In real-world attack scenarios where the attacker would collect large amounts of data and where stealth is critical, an adversary must find a balance between the volume and the risk of detection. A more cautious attacker could also favor low-noise techniques, spreading out reconnaissance over time to remain undetected.

Tactic	Tool name	System Labels	System Events	Internal Flows
Reconnaissance	nmap	518	591	5,885,832
	curl	6	8	4002
	dig	4	6	0
	gobuster	1	2	0
Discovery	nmap	597	59,213	6,793,308
	curl	469	348,245	12,802
	find	425	355,698	11
	ping	302	336,252	70
	linpeas	131	338,832	11
	dig	8	878	0
	pspy64	8	8	0
	ifconfig	3	10,609	0
	PwnKit	2	4626	0
	lse.sh	1	869	0
	ngrok	1	2	0
Collection	mutt	154	937	0
	s-nail	22	27	0
	tar	8	20	0
	zip	3	3	0
	strings	3	3	0
	postqueue	3	16	0
	wget	2	3	0
	gzip	2	6	0
	foremost	1	1	0
Privilege escalation	mount	67	3069	0
	github 1	58	820	0
	sudo su	40	47	0
	github 2	8	90	0
	docker	2	2281	0
Defense evasion	chmod	85	85	0
	rm	59	67	0
	chown	23	23	0
Lateral movement	ssh	617	674	0
	nc	106	122	12
	scp	60	111	0
	su	47	61	0
Resource development	nano	135	168	0
	vi	31	70	0
	vim	27	35	0
Exfiltration	scp	72	136	0
	python3	30	37	0
	wget	12	13	0
	curl	11	23	0
	nc	1	2	0
Credential access	tcpdump	14	23	0
	nmap	5	14	0
	ssb	5	8	0
	unshadow	3	7	0
	hashcat	2	6	0
	john	1	1	0
Persistence	passwd	16	16	0
	useradd	2	4	0
Command and control	git	169	412	0
	curl	84	242	1
	wget	60	289	7
	chisel	27	38	13
	ligolo-ng	26	1937	14
	ngrok	18	337	0
Impact	rm	5	5	0
	psql	1	5	0

TABLE VI: Tool usage by tactic and their consequences on the number of raised events and flows

D. Empirical Cyber Kill Chain

Beyond analyzing the frequency of individual tactics, we also examined how these tactics transition from one to another during the challenge. This kind of analysis was already performed by Rodríguez *et al.* [27] using process mining techniques. Figure 6 illustrates the frequencies of tactic transitions with more than 24 occurrences, based on the sequences of tactics associated with labels across the entire dataset. We observed that certain tactics were frequently combined, forming structured sequences rather than occurring at random. Notably, *Discovery* serves as a central pivot in the attack process, with attackers repeatedly returning to it. This behavior is an expected consequence of the exploratory nature of the challenge: since players operate in a completely unknown environment, they progress incrementally by gathering system information before deciding on their next move. The recurrence of discovery and its connections to multiple tactics indicate an iterative approach, where attackers refine their strategy dynamically, possibly testing different paths before committing to an exploitation phase.

Discovery is not only the most used tactic (Table V) but also one of the most automated (Table VI) and the one that generates the most system or network logs (Table V). This implies that monitoring systems should pay close attention to the occurrence of events suggesting the use of legitimate tools or commands that facilitate discovery. For example, commands like `cat /etc/hosts` allow an attacker to enumerate accessible hosts and user accounts from a compromised machine. Many of these commands fall under the category of Living Off The Land Binaries and Scripts, which attackers exploit to blend malicious actions within normal system activity [7]. By closely monitoring such seemingly benign actions, security tools could improve early threat detection.

Our results also suggest that the state-transition system presented in Figure 6 constitutes an empirical kill chain, challenging the overly linear nature of traditional models. Classic representations, such as Lockheed Martin's cyber kill chain [14], assume a mostly sequential attack progression, whereas our observations highlight a more iterative and non-linear process. This aligns with the insights from [5], which suggested that reconnaissance is not a one-time initial phase but rather a recurring step throughout the attack lifecycle.

VI. Players Behaviors

This scenario was played by participants with various levels of expertise and different strategies. All of them follow the same attack path and the infrastructure was designed accordingly. Players can explore irrelevant areas (such as the content of the machines), but they cannot access newly discovered machines unless these accesses align with the initially planned attack path. For example, the initial user, `tbenedict`, could not connect to the `bastion` machine even if the player had discovered it. Two advantages can be noted: 1) it avoids players from losing too much time in too complex dead ends; 2) it enables from our side a relevant comparison between players.

Time	Position	Command	Label
02:29:23	tbenedict@start	nano dic	T1587: Develop Capabilities
02:30:02	tbenedict@start	nmap -p 22 –script ssh-brute –script-args userdb=user,passdb=dic –script-args ssh-brute.timeout=4s 10.35.128.20	T1110: Brute Force
02:32:21	tbenedict@start	nmap -p 22 –script ssh-brute –script-args userdb=user.lst,passdb=dic.lst –script-args ssh-brute.timeout=4s 10.35.128.20	T1110: Brute Force
02:32:58	tbenedict@start	nmap -p 22 –script ssh-brute –script-args userdb=user.lst,passdb=dic.lst 10.35.128.20	T1110: Brute Force
02:34:07	tbenedict@start	ssh tocean@10.35.128.20	T1021: Remote Services
02:35:24	tbenedict@start	scp poc tocean@10.35.128.20:/home/tocean/poc	T1570: Lateral Tool Transfer
02:35:39	tocean@bastion	./poc	T1068: Exploitation for Privilege Escalation
02:35:39	root@bastion	/bin/bash	

Log 2: List of commands used by the player on the *fantaisie* instance for steps 4-6

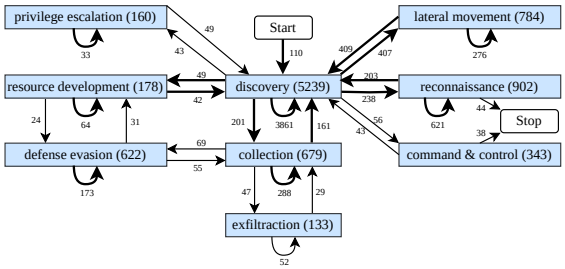

Fig. 6: Empirical cyber kill chain computed through most frequent transitions between tactics.

In this section, we highlight the variety of player behaviors, and we show how the collected data and labels can be processed in an understandable summary, which can be used for example by a security analyst. Finally, we also show that players can be recognized by matching their session against a profile built on top of their command usages.

A. Players Comparison

For comparing players we analyzed the sessions of commands and we computed two main indicators: their stealth, measured by the number of generated system/network logs with assigned labels, and their efficiency, which can be assessed through the number of steps or commands used to reach their objective.

Logs 2 and 3 illustrate the behavior of two players assigned with two chosen words as instance names: `moderne` and `fantaisie` instances. Both logs show the player at the same stage of the scenario when connected to the `start` machine under the user account `tbenedict` and searching to enter the second machine `bastion`. They had to use the non-privileged account `tocean` after bruteforcing the SSH service, and subsequently escalate privileges to root on `bastion`. Both players start at approximately 02:30:00. However, their approaches and efficiency differ significantly: The *fantaisie* player escalates to root on `bastion` in approximately 5 minutes, using only 8 commands. In contrast, the *moderne* player reaches the same objective after 1 hour and 15 minutes, executing 63 commands.

B. Towards Attackers Identification

We explore in this section whether players can be identified based on their usage of the specific command-line tools:

- Network inspection (`ss`)
- Network scanning (`nmap`, `nc`)
- Text editing (`vi`, `vim`, `nano`)

Our hypothesis is that each player or attacker develops personal habits in the way they use these tools, particularly in the parameters they select and the order in which they provide them. Behavioral fingerprinting techniques have been successfully applied in other security contexts, such as browser fingerprinting [9], keystroke dynamics [15], and SSH behavior analysis [24]. These works demonstrate that users exhibit consistent behaviors even in technical interactions, which suggests that command-line usage patterns may also serve as behavioral identifiers. For example, while one player might habitually execute `ss -ltnp`, another might prefer `ss -nltpu`, introducing unique patterns in command usage. If such habits exist, they could serve as identifiers for distinguishing different players.

To test this hypothesis, we first constructed a binary command usage matrix (Figure 7) to visualize which players used which commands. This first figure shows that the usage of these commands is well differentiated among players. To refine this result, we computed a Jaccard similarity matrix, shown in Figure 8, to measure behavioral overlap between players. The similarity scores using the Jaccard index reveal that command usage patterns remain highly distinct, with a minimum similarity of 0.3. The average similarity score between two players is 0.18, and the highest is 0.6 indicating that while some players share command usage habits, no pairs of players are fully identical in their command execution patterns. The heatmap derived from this matrix represents the similarity between two players, with darker colors indicating lower similarity and lighter colors representing greater overlap in command usage. We observe some clusters of players with similar behaviors, but overall, each player's command set remains unique. Note also that even a team for which we gave a backup instance, is distinct from itself. This can be explained by the fact that such a team plays the first part of the scenario on the first instance, and the second part on the backup instance: these two parts of the scenario does not involve the same commands.

Time	Position	Command	Label
02:20:05	tbenedict@meet	nano bruteforce.py	T1587: Develop Capabilities
02:21:53	tbenedict@meet	python3 bruteforce.py	T1110: Brute Force
02:22:37	tbenedict@meet	bash ./tqt.sh	T1110: Brute Force
02:23:01	tbenedict@meet	bash ./tqt.sh -u username.txt -p passwords.txt 10.35.165.15	T1110: Brute Force
02:23:11	tbenedict@meet	bash ./tqt.sh -u username.txt -p passwords.txt -a 10.35.165.15 -d 22	T1110: Brute Force
02:23:28	tbenedict@meet	bash ./tqt.sh -u username.txt -p passwords.txt -a 10.35.165.20 -d 22	T1110: Brute Force
02:27:12	tocean@bastion	id	T1033: System Owner/User Discovery
02:28:27	tocean@bastion	ss -antp	T1049: System Network Connections Discovery
02:29:38	tocean@bastion	curl http://10.135.165.10/login	
02:32:47	tocean@bastion	ssh tocean@intranet	T1021: Remote Services
02:33:05	tocean@bastion	ssh tbenedict@intranet	T1021: Remote Services
02:33:23	tocean@bastion	su admin	T1548: Abuse Elevation Control Mechanism
02:33:59	tocean@bastion	ss -antp	T1049: System Network Connections Discovery
02:34:54	tocean@bastion	/usr/bin/python3 /usr/lib/command-not-found – smtp	T1518: Software Discovery
02:35:21	tocean@bastion	telnet 10.135.165.20 25	T1046: Network Service Discovery
02:41:32	tbenedict@start	ssh tbenedict@meetingcam	T1021: Remote Services
02:41:41	tbenedict@meet	ssh tocean@bastion	T1021: Remote Services
02:43:50	tocean@bastion	nano 1715872830.V801I7e23cM311161.bastion.CasinoLimit .bzh	T1114: Email Collection
02:44:07	tocean@bastion	nano 1715872830.V801I7e240M342889.bastion.CasinoLimit .bzh	T1114: Email Collection
02:45:18	tocean@bastion	curl http://intranet	
02:45:21	tocean@bastion	curl http://intranet/login	
02:45:29	tocean@bastion	curl http://intranet/admin	
02:45:35	tocean@bastion	curl http://intranet/ogin	
02:45:38	tocean@bastion	curl http://intranet/login	
02:46:12	tocean@bastion	curl http://intranet/images/	
02:46:16	tocean@bastion	curl http://intranet/images/login.jpef	
02:46:17	tocean@bastion	curl http://intranet/images/login.jpeg	
02:49:15	tocean@bastion	sh -c /usr/lib/mutt/source-muttrc.d	T1114: Email Collection
02:49:17	tocean@bastion	mutt	T1114: Email Collection
03:02:19	tocean@bastion	whoami	T1033: System Owner/User Discovery
03:04:42	tocean@bastion	curl http://intranet/login	
03:06:56	tocean@bastion	curl -X POST -d username=admin&password=admin http://intranet	T1078: Valid Accounts
03:06:59	tocean@bastion	curl -X POST -d username=admin&password=admin http://intranet/login	T1078: Valid Accounts
03:07:49	tocean@bastion	curl -X POST -d login=admin&password=admin http://intranet/login	T1078: Valid Accounts
03:08:28	tocean@bastion	curl -X POST -d login=tocean&password=kaeCaiSoojie7i http://intranet/login	T1078: Valid Accounts
03:09:13	tocean@bastion	curl -X POST -d login=admin&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:09:33	tocean@bastion	curl -X POST -d login=admin&password=kaeCaiSoojie7i http://intranet/login	T1078: Valid Accounts
03:10:08	tbenedict@start	ssh tbenedict@meetingcam	T1021: Remote Services
03:10:17	tbenedict@meet	id -u	T1033: System Owner/User Discovery
03:10:45	tocean@bastion	curl -X POST -d login=cfragasso&password=kaeCaiSoojie7i http://intranet/login	T1078: Valid Accounts
03:11:07	tocean@bastion	curl -X POST -d login=twiseau&password=kaeCaiSoojie7i http://intranet/login	T1078: Valid Accounts
03:11:24	tocean@bastion	curl -X POST -d login=cfragasso&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:11:39	tocean@bastion	curl -X POST -d login=twiseau&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:12:35	tocean@bastion	curl -X POST -d login=admin&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:12:50	tocean@bastion	curl http://intranet/login	
03:14:41	tocean@bastion	curl -X POST -d login='OR 1=1 – http://intranet/login	T1190: Exploit Public-Facing Application
03:14:55	tocean@bastion	curl -X POST -d login='OR 1=1 – &password='OR 1=1 – http://intranet/login	T1190: Exploit Public-Facing Application
03:15:39	tocean@bastion	curl -X POST -d login=admin&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:15:47	tocean@bastion	curl -v -X POST -d login=admin&password=NEi9g8Bc http://intranet/login	T1078: Valid Accounts
03:16:12	tocean@bastion	s-nail	T1114: Email Collection
03:17:30	tocean@bastion	curl -X POST -d login=camille&password=C@m1lle http://intranet/login	T1078: Valid Accounts
03:17:41	tocean@bastion	ssh camille@intranet	T1021: Remote Services
03:18:42	tocean@bastion	s-nail	
03:19:18	tocean@bastion	curl -X POST -d login=tocean&password=kaeCaiSoojie7i http://intranet/login	T1078: Valid Accounts
03:31:14	tbenedict@start	scp CVE.zip tbenedict@meetingcam:/home/tbenedict/CVE.zip	T1570: Lateral Tool Transfer
03:31:34	tbenedict@start	scp CVE.zip tbenedict@meetingcam:/home/tbenedict/CVE.zip	T1570: Lateral Tool Transfer
03:32:26	tbenedict@meet	scp CVE.zip tocean@bastion:/home/tocean/CVE.zip	T1570: Lateral Tool Transfer
03:40:21	tocean@bastion	/usr/bin/python3 /usr/lib/command-not-found – untar	T1518: Software Discovery
03:40:33	tocean@bastion	tar –help	T1518: Software Discovery
03:40:53	tocean@bastion	tar tf CVE.tar	T1608: Stage Capabilities
03:40:56	tocean@bastion	tar -xvf CVE.tar	T1608: Stage Capabilities
03:41:54	tocean@bastion	./exp	T1068: Exploitation for Privilege Escalation
03:44:59	root@bastion	id	T1033: System Owner/User Discovery

Log 3: List of commands used by the player on the *moderne* instance for steps 4-6

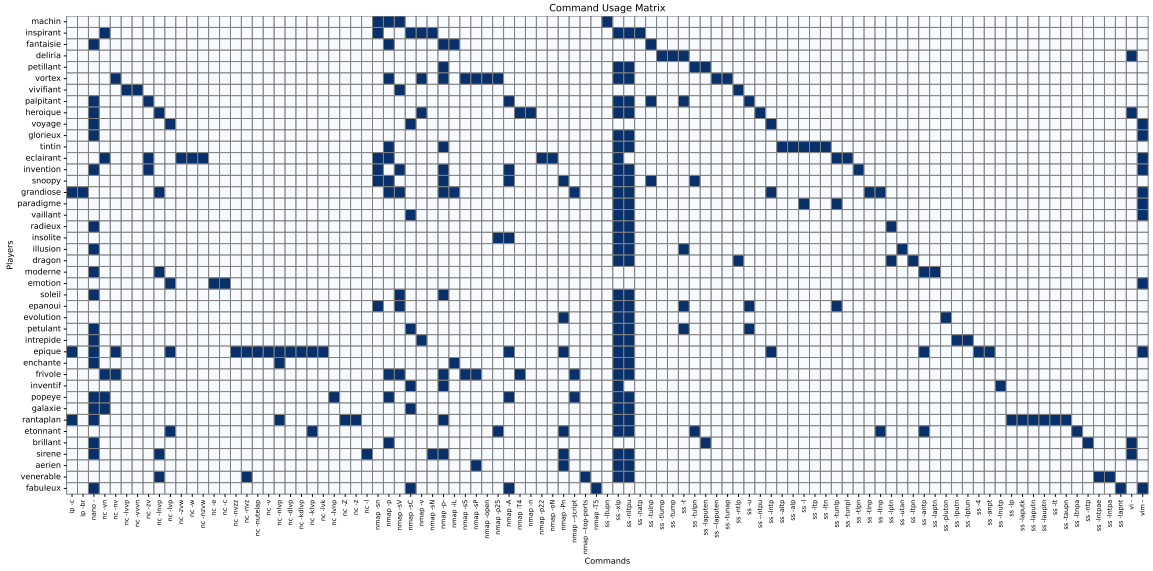

Fig. 7: Commands usage across players.

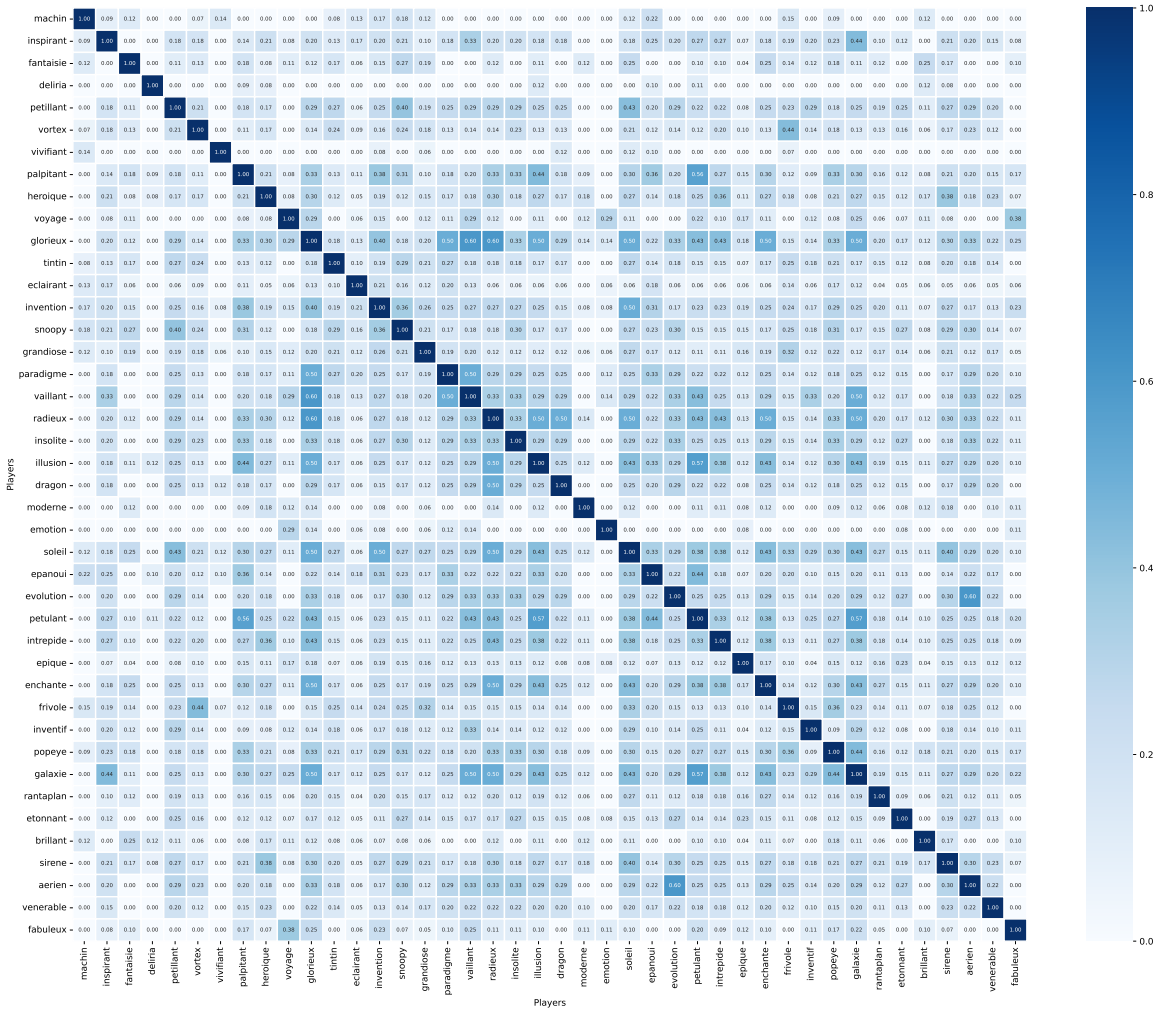

Fig. 8: Behavioral similarity between players based on command usage patterns.

These findings demonstrate that command-line habits are distinctive enough to differentiate players, reinforcing the potential for behavioral fingerprinting in cybersecurity. Such an approach could aid in tracking persistent attackers or detecting anomalous activity in real-world systems.

VII. Threat to Validity

While the CasinoLimit dataset provides a unique opportunity to analyze multiple attacker behaviors in a controlled penetration testing scenario, several factors may impact the generalizability and robustness of our findings.

a) Technical Attack Coverage: This dataset captures a range of attack techniques, allowing us to observe different strategies adopted by attackers. However, certain techniques may be underrepresented due to the predefined scenario structure. For instance, the scenario included a single privilege escalation path, which was explicitly guided within the game design. As a result, only one type of privilege escalation was observed, limiting variability in escalation strategies. Future datasets could expand the range of adversary behaviors by incorporating additional scenarios with diverse objectives to better reflect real-world attack diversity.

b) Lack of Benign Traffic: A key characteristic of this dataset is the absence of significant benign traffic, aside from some automated network services. This simplifies labeling and makes it particularly well-suited for studies on attacker profiling, behavioral modeling, and tool usage analysis. However, this also limits its applicability to research areas that require distinguishing between malicious activity and legitimate user behavior. Future datasets could incorporate realistic background traffic to improve applicability to intrusion detection, threat-hunting scenarios, and security monitoring in production environments.

c) Generalizability Beyond the CTF Scenario: Since the dataset was collected in a controlled CTF environment, players operated within a predefined infrastructure without the need to evade active defensive countermeasures. This likely reduced the effort attackers put into stealth, as they were aware that the monitoring system would not block their actions. Consequently, the dataset may not fully capture evasion strategies used against intrusion detection systems or active defenders. Future work could analyze datasets where attackers engage with dynamic defenses to assess behavioral shifts under adversarial conditions and better understand how real-world attackers adapt their strategies in response to defensive pressure.

d) Limitations of Attacker Fingerprinting: Our analysis demonstrates that players exhibit distinguishable command-line habits, suggesting that behavioral fingerprinting can be used to differentiate attackers based on their tool usage patterns. In this study, we showed that commonly used command-line tools were sufficient to distinguish all players who executed at least four distinct commands. A natural extension of this work would be to examine how many tool usage observations are required to maintain reliable attacker differentiation as the number of distinct adversaries increases. Additionally, future research should evaluate the resilience of behavioral fingerprinting against adversarial adaptation, where attackers may deliberately alter their command usage to evade identification. Testing this approach against automated attack frameworks and scripting techniques could further assess its robustness in detecting real-world adversaries.

VIII. Conclusion

We have presented the CasinoLimit dataset, a labeled dataset of attack data collected from a large-scale cybersecurity challenge[3]. The dataset was generated from 114 controlled environments where players acted as attackers on a vulnerable infrastructure of four hosts, aiming to compromise a target machine and erase sensitive data. The underlying infrastructure prevented players from significantly deviating from the intended attack path. However, since the game design did not explicitly guide their actions, players had to develop their own approach to the attack, which led them to expose their technical methods and actively explore their environment.

The dataset is provided with Manatee, a labeling and log visualization software, which is used to label 98.45% of system logs related to shell sessions and 11.70% network logs using MITRE ATT&CK techniques, providing a fine-grained categorization of attacker activities. Manatee supports the automation of the labeling and the manual part, achieved by a junior and expert analyst. Our analysis shows that the dataset captures a wide range of attacker behaviors, reflecting diverse levels of expertise, strategic approaches, and tooling preferences.

Our findings highlight that *Discovery* actions recur frequently in attacker behaviors and that the kill-chain is not a linear path. The *Discovery* actions are not only the most prevalent but also among the most automated and generate the highest volume of system and network activity. This suggests that improving the monitoring of system and network scanning, including benign enumeration commands, could enhance early threat detection.

Finally, this dataset suggests that attackers exhibit distinct tool usage patterns, indicating that behavioral fingerprinting may be feasible. However, further studies with larger and more diverse datasets are needed to confirm this hypothesis. Despite limitations inherent to the CTF-based design, we believe that CasinoLimit will serve as a valuable resource for advancing research in attack detection, behavioral analysis, and cybersecurity automation.

Acknowledgement

This work has benefited from a government grant managed by the National Research Agency under France 2030 with reference ANR-22-PECY-0007.

[3]The code supporting our analyses is available at https://gitlab.inria.fr/pirat-public/datasets/casinolimit and the necessary data at https://doi.org/10.5281/zenodo.15278062

References

[1] Bader Al-Sada, Alireza Sadighian, and Gabriele Oligeri. Mitre att&ck: State of the art and way forward. *ACM Comput. Surv.*, 57(1), October 2024.

[2] Frederick Barr-Smith, Xabier Ugarte-Pedrero, Mariano Graziano, Riccardo Spolaor, and Ivan Martinovic. Survivalism: Systematic analysis of windows malware living-off-the-land. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1557–1574, 2021.

[3] Masooda Bashir, Colin Wee, Nasir Memon, and Boyi Guo. Profiling cybersecurity competition participants: Self-efficacy, decision-making and interests predict effectiveness of competitions as a recruitment tool. *Computers & Security*, 65:153–165, March 2017.

[4] Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong, and Gilles Guette. PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns. *IEEE Transactions on Network and Service Management*, 19(4):5252–5264, 2022.

[5] Aimad Berady, Valérie Viet Triem Tong, Gilles Guette, Christophe Bidan, and Guillaume Carat. Modeling the Operational Phases of APT Campaigns. In *CSCI 2019 - 6th Annual Conf. on Computational Science & Computational Intelligence*, pages 1–6, Las Vegas, United States, December 2019.

[6] Pierre-Victor Besson, Valérie Viet Triem Tong, Gilles Guette, Guillaume Piolle, and Erwan Abgrall. Ursid: Automatically refining a single attack scenario into multiple cyber range architectures. In Mohamed Mosbah, Florence Sèdes, Nadia Tawbi, Toufik Ahmed, Nora Boulahia-Cuppens, and Joaquin Garcia-Alfaro, editors, *Foundations and Practice of Security*, pages 123–138, Cham, 2024. Springer Nature Switzerland.

[7] Hrushikesh Chunduri, P. Mohan Anand, Sandeep K Shukla, and P. V. Sai Charan. Trusted yet disguised: Analysing the subversive role of lolbins in contemporary cyber threats. In *2024 IEEE International Conference on Big Data (BigData)*, pages 2596–2605, 2024.

[8] Sajjad Dadkhah, Hassan Mahdikhani, Priscilla Kyei Danso, Alireza Zohourian, Kevin Anh Truong, and Ali A Ghorbani. Towards the development of a realistic multidimensional iot profiling dataset. In *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*, pages 1–11. IEEE, 2022.

[9] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10, page 1–18, Berlin, Heidelberg, 2010. Springer-Verlag.

[10] Elastic. Event Query Language (EQL), 2025.

[11] Julie Gjerstad, Fikret Kadiric, Gudmund Grov, Espen Hammer Kjellstadli, and Markus Leira Asprusten. LADEMU: a modular & continuous approach for generating labelled APT datasets from emulations. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 2610–2619, December 2022.

[12] Patrik Goldschmidt and Daniela Chudá. Network intrusion datasets: A survey, limitations, and recommendations, 2025.

[13] Jorge Luis Guerra, Carlos Catania, and Eduardo Veas. Datasets are not enough: Challenges in labeling network traffic. *Computers & Security*, 120:102810, September 2022.

[14] Eric M. Hutchins, Michael J. Cloppert, and Rohan M. Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Lockheed Martin Corporation*, 2011.

[15] Kevin S. Killourhy and Roy A. Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 125–134, 2009.

[16] Max Landauer, Florian Skopik, Markus Wurzenberger, Wolfgang Hotwagner, and Andreas Rauber. Have it Your Way: Generating Customized Log Datasets With a Model-Driven Simulation Testbed. *IEEE Transactions on Reliability*, 70(1):402–415, March 2021.

[17] Maxime Lanvin, Pierre-François Gimenez, Yufei Han, Frédéric Majorczyk, Ludovic Mé, and Eric Totel. Errors in the CICIDS2017 dataset and the significant differences in detection performances it makes. In *CRiSIS 2022 - 17th International Conference on Risks and Security of Internet and Systems*, volume 13857, pages 18–33, Sousse, Tunisia, December 2022. Springer.

[18] Heather Lawrence, Uchenna Ezeobi, Orly Tauil, Jacob Nosal, Owen Redwood, Yanyan Zhuang, and Gedare Bloom. Cupid: A labeled dataset with pentesting for evaluation of network intrusion detection. *J. Syst. Archit.*, 129(C), August 2022.

[19] Jinxin Liu, Yu Shen, Murat Simsek, Burak Kantarci, Hussein T. Mouftah, Mehran Bagheri, and Petar Djukic. A new realistic benchmark for advanced persistent threats in network traffic. *IEEE Networking Letters*, 4(3):162–166, 2022.

[20] Lisa Liu, Gints Engelen, Timothy Lynar, Daryl Essam, and Wouter Joosen. Error prevalence in nids datasets: A case study on cic-ids-2017 and cse-cic-ids-2018. In *2022 IEEE Conference on Communications and Network Security (CNS)*, pages 254–262, 2022.

[21] MITRE. ATT&CK Framework, 2024.

[22] Sowmya Myneni, Ankur Chowdhary, Abdulhakim Sabur, Sailik Sengupta, Garima Agrawal, Dijiang Huang, and Myong Kang. DAPT 2020 - Constructing a Benchmark Dataset for Advanced Persistent Threats. In Gang Wang, Arridhana Ciptadi, and Ali Ahmadzadeh, editors, *Deployable Machine Learning for Security Defense*, pages 138–163, Cham, 2020. Springer International Publishing.

[23] Sowmya Myneni, Kritshekhar Jha, Abdulhakim Sabur, Garima Agrawal, Yuli Deng, Ankur Chowdhary, and Dijiang Huang. Unraveled — A semi-synthetic dataset for Advanced Persistent Threats. *Computer Networks*, 227:109688, May 2023.

[24] Julien Piet, Aashish Sharma, Vern Paxson, and David Wagner. Network detection of interactive SSH impostors using deep learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4283–4300, Anaheim, CA, August 2023. USENIX Association.

[25] Portia Pusey, Mark Gondree, and Zachary Peterson. The outcomes of cybersecurity competitions and implications for underrepresented populations. *IEEE Security & Privacy*, 14(6):90–95, 2016.

[26] Markus Ring, Sarah Wunderlich, Dominik Grüdl, Dieter Landes, and Andreas Hotho. Flow-based benchmark data sets for intrusion detection. In *Proceedings of the 16th European conference on cyber warfare and security. ACPI*, pages 361–369, 2017.

[27] Marcelo Rodríguez, Gustavo Betarte, and Daniel Calegari. A process mining-based method for attacker profiling using the MITRE ATT&CK taxonomy. *Journal of Internet Services and Applications*, 15(1):212–232, August 2024. Number: 1.

[28] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *International Conference on Information Systems Security and Privacy*, 2018.

[29] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1(2018):108–116, 2018.

[30] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. Mitre att&ck: Design and philosophy. In *Technical report*. The MITRE Corporation, 2018.

[31] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6, 2009.

[32] Clark Taylor, Pablo Arias, Jim Klopchic, Celeste Matarazzo, and Evi Dube. CTF: State-of-the-Art and building the next generation. In *2017 USENIX Workshop on Advances in Security Education (ASE 17)*, Vancouver, BC, August 2017. USENIX Association.

[33] CyberVan team. Cyberforce scenario data sets. https://cybervan.peratonlabs.com:9000/cyberforce-scenario-data-sets, 2024. Accessed: 2025-04-22.

[34] Franco Terranova, Abdelkader Lahmadi, and Isabelle Chrisment. Leveraging deep reinforcement learning for cyber-attack paths prediction: Formulation, generalization, and evaluation. In *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses*, RAID '24, page 1–16, New York, NY, USA, 2024. Association for Computing Machinery.

[35] Rafael Uetz, Christian Hemminghaus, Louis Hackländer, Philipp Schlipper, and Martin Henze. Reproducible and adaptable log data generation for sound cybersecurity experiments. In *Annual Computer Security Applications Conference*, ACSAC '21, page 690–705. ACM, December 2021.

APPENDIX

TABLE VII: Association rules between techniques and commands, used in the automated labeling process

Tactic	Technique	Commands
Collection	T1114 - Email Collection T1125 - Video Capture	email, mutt, thunderbird, outlook camera, webcam, snapshot
Defense evasion	T1562 - Impair Defenses T1550 - Use Alternate Authentication Material T1222 - File and Directory Permissions Modification	audit su admin, sudo su chmod, chown
Discovery	T1033 - System Owner/User Discovery T1069 - Permission Groups Discovery T1082 - System Information Discovery T1016 - System Network Configuration Discovery T1083 - File and Directory Discovery T1049 - System Network Connections Discovery T1057 - Process Discovery T1018 - Remote System Discovery T1046 - Network Service Discovery	whoami, who, id, users, /etc/passwd sudo -l systeminfo, lsmod, uptime, env, /etc/issue, uname -a ifconfig, arp, route, netstat, nbtstat, ip, nslookup ls, cat netstat, net use, net session, ss ps, pspy ping nmap, masscan, zmap, unicornscan
Execution	T1129 - Shared Modules T1053 - Scheduled Task/Job T1610 - Deploy Container	dlopen, dlsym, LoadLibrary schtasks, cron docker, kubernetes
Lateral movement	T1021 - Remote Services T1570 - Lateral Tool Transfer	ssh, nc -lvnp scp
Persistence	T1136 - Create Account	useradd
Privilege escalation	T1068 - Exploitation for Privilege Escalation	CVE-2023-0386
Reconnaissance	T1592 - Gather Victim Host Information T1595 - Active Scanning T1590 - Gather Victim Network Information	hostname masscan, nmap masscan