PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns

Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong, Gilles Guette

▶ **To cite this version:**

Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong, Gilles Guette. PWNJUTSU: A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns. IEEE Transactions on Network and Service Management, 2022, Special Issue on Recent Advances in Network Security Management, 19 (4), pp.5252-5264. 10.1109/TNSM.2022.3183476 . hal-03694719

HAL Id: hal-03694719
https://inria.hal.science/hal-03694719

Submitted on 14 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PWNJUTSU: A dataset and a semantics-driven approach to retrace attack campaigns

Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong, and Gilles Guette

***Abstract*—Identifying patterns in the *modus operandi* of attackers is an essential requirement in the study of *Advanced Persistent Threats*. Previous studies have been hampered by the lack of accurate, relevant, and representative datasets of current threats. System logs and network traffic captured during attacks on real companies' information systems are the best data sources to build such datasets. Unfortunately, for apparent reasons of companies' reputation, privacy, and security, such data is seldom available. This article proposes an alternative approach to such issues involved with collecting data. It first presents a formal model of an attacker's tactical progression during their network propagation phase. Such a progression is expressed according to the attacker's state, called muSE, which specifies their propagation area, collected secrets, and knowledge of the environment. The new model wields the operational semantics of attack techniques proposed in this article. The semantics formally define a transition relation between attackers' states. Hence, it can be used to describe an entire attack scenario. This formalization allows the ability to describe the PWNJUTSU experiment unequivocally. In this experiment, 22 *Red Teamers* attacked the vulnerable infrastructure to compromise machines and steal secret *flags*. Each *Red Teamer* operated on a dedicated instance. Sensors captured system logs and network traffic on each of these instances. This article's second contribution is the public release of the PWNJUTSU dataset.**

***Index Terms*—Advanced Persistent Threat, Tactics Techniques Procedures, Red Team, SIEM, Dataset**

I. Introduction and motivation

ACADEMIC research in the field of system and network security regularly needs datasets containing identified attacks, malware, and any malicious activities. Unfortunately, very few researchers published the datasets from their experiments [1]. In [2], Grajeda *et al.* highlight the difficulty in finding suitable datasets, particularly in the field of cybersecurity. The authors reviewed 715 scientific papers published in various conferences and journals from 2010 to 2015. 49% of these papers employed datasets; among those, only 36% relied on real-world datasets. The authors point out the lack of real-world datasets available to the digital forensics community, a conclusion also reached by Baggili *et al.* in [3] and Stojanovic *et al.* in [4]. Currently published datasets are also varied in nature: Grajeda *et al.* reported 70 different types of datasets such as images datasets, malware, raw dumps, emails, *etc*.

A. Berady and V. Viet Triem Tong are with CentraleSupélec, Inria, Univ Rennes, CNRS, IRISA, F-35042 Rennes, France (e-mails: {aimad.berady}{valerie.viet_triem_tong}@centralesupelec.fr)

M. Jaume is with Sorbonne Université, CNRS, LIP6, F-75005 Paris, France (e-mail: mathieu.jaume@lip6.fr)

G. Guette is with Univ Rennes, Inria, CentraleSupélec, CNRS, IRISA, F-35042 Rennes, France (e-mail: gilles.guette@univ-rennes1.fr)

This article focuses on datasets related to *Advanced Persistent Threat* (APT) attacks. These surgically targeted attacks are stealthy and led by **advanced** attackers, who constantly adapt their *Tactics, Techniques, and Procedures* (TTP). These attacks are **persistent** since the intruder can remain active for months in the system. These attacks target strategic data or services and are therefore real **threats** to their victims [5].

In [4], Stojanovic *et al.* state that APT attack detection is a challenging and popular area of research in the scientific community and that the lack of reference datasets is a significant challenge. They propose a full review of APT datasets in the literature but only describe five datasets. Sharafaldin *et al.* already point out in [6] that these APT datasets are rare: the availability of such datasets is limited by confidentiality issues, and the others are highly anonymized and do not reflect current trends, or they lack specific statistical characteristics.

The issues raised by previous authors can be addressed by regularly publishing attack datasets in order to remain representative of current threats. According to [4], the most popular intrusion detection dataset in literature is still the very old KDDCup'99 [7]. We are convinced that the utility of a dataset also strongly depends on the quality of its description. In the context of APT datasets, however, the targeted architecture and the attack scenario are unfortunately either missing or described approximately.

In order to exploit these architectures and scenarios, we need to know the precise description of the targeted network, including the targeted machines with their operating systems, services, legitimate users, and interconnections with the other network machines.

We agree with the *Attack Flow* [8] concept recently introduced by the *Center for Threat-Informed Defense* (MITRE Engenuity), which highlights that understanding attack technique sequences and contexts makes defense more effective.

As stated in [9], the *Cyber Threat Intelligence* experts need formalization at every stage of attacks to understand manipulated tools and data as well as to produce relevant, entirely usable, and comparable datasets.

In this article, we compensate for the lack of raw data that is used to study APT actors' TTP by proposing a dataset obtained by observing 22 professional *Red Teamers* on an architecture representative of a vulnerable subnetwork. In this context, a *Red Teamer* is an individual (or a group) who plays the role of adversaries to assess a company's security. For this dataset to benefit the community, it was necessary to describe the target architecture and different steps of attacks unequivocally.

This article presents two main contributions:

1) First, we present a **formal framework** which allows us to describe an entire attack campaign. It consists of: (1) a model to describe the victim's network through the description of both what it hosts (services, users, assets) and its topology; (2) a model to describe the attacker's state that symbolizes the attacker's knowledge, control, and perception of this network; and finally (3) the operational semantics of attack techniques that an attacker can exploit to progress towards their objectives. Figure 1 presents the main elements involved in an attack and how they are related to each other in our formal framework.
2) This formal representation is challenged to describe attacks achieved during an experiment on the scale of 22 professional *Red Teamers* attacking a vulnerable architecture. This experiment has produced **the PWNJUTSU dataset**, which consists of 16 million event logs and 172 GB of network traces, which forms the second main contribution of this article.

The remainder of this paper is structured as follows. Related publications about APT datasets building are presented in Section II. The generic infrastructure model and that designed for the PWNJUTSU experiment are described in Section III. The attacker perception of the targeted network and the PWNJUTSU experiment panel are described in Section IV The description of the attacker progression thanks to the operational semantics of attack techniques is described in Section V. An instantiation of the model with a given participant, extracted from the released PWNJUTSU dataset, is presented in Section VI. Finally, PWNJUTSU dataset data records resulting from the experiment and results of the survey conducted are detailled in Section VII.

II. Related works

To study APT attacks, it would be useful to access real event logs of freshly compromised companies by APT actors. This opportunity appears inconceivable to date due to the risks incurred by the company in publicly exposing its data. This lack of feasibility is also due to the non-disclosure necessity for *Cyber Threat Intelligence* (CTI) experts to guarantee the currentness of the adversary-related data collected. Thus, when scientists need a novel dataset to study modern APT attacks, several alternatives could be explored [10].

Repository of publicly-available reports

The description of APT attacks is still done through weakly-structured technical reports. These APT reports are mainly published on blogs by independent experts, on websites of computer security companies, or by CERTs on their institutional websites. For instance, APTnotes [11] is a collection of those publicly-available reports related to malicious activities or malware discovered in the wild, that have been attributed to APT groups. These reports detail different aspects of the campaigns, such as the intended targets, the attack vectors, the delivery method, main TTPs, the effects observed on the system, and even some Indicators of Compromise (IOCs). CTI experts need these reports to have a global understanding of APT attacks. However, it is difficult to extract actionable CTI from the reports, because their primary purpose is to expose the incriminated attackers publicly and thus promote the expertise of the company that conducted the investigation.

Malware referenced during APT campaign

In 2017, Laurenza *et al.* evaluated to what extent an incoming malicious sample could have been developed by an APT-actor [12]. A part of their studied dataset consisted of malware discovered during investigations of APT campaigns. This kind of dataset is highly interesting because it allows acquiring knowledge about attackers, such as their technical level, part of their *modus operandi*, and even identifying their C&C infrastructure. However, malware used by attackers is only a part of the attack procedures that attackers can automate and offers only a partial view of APT campaigns.

Logs collection on real information systems

System and network logs observed on a real system during an attack are an interesting source of information. In 2017, Turcotte *et al.* [13] [14] published their third dataset consisting of network flow and host log events collected from the Los Alamos National Laboratory enterprise network over a random period of 90 days. To protect the company's cybersecurity, they de-identified some values before making the dataset public. In 2021, Ho *et al.* presented in "Hopper: Modeling and Detecting Lateral Movement" [15], a system for detecting lateral movement in enterprise event logs. They used a private dataset of over 780 million lines collected on Dropbox internal servers. While building the dataset, a Red Team was engaged to simulate a common APT scenario. The company also used automatic commercial security products and custom tools to stress their systems.

This kind of dataset offers the advantage of proposing real data and thus exposing attack procedures used by attackers. That said these datasets are nevertheless incomplete. For example, some of the logs may have been modified for confidentiality reasons, or the detection probes may have missed events because they were poorly configured. Above all, these datasets are not coupled with a detailed description of the targeted system architecture; the full context of the attack is thus difficult to understand and complicates its reconstruction from the attacker's perspective.

Adversarial simulation, TTP mimicry or attack replay

Gianvecchio *et al.* proposed in "Closing the Gap with APTs Through Semantic Clusters and Automated Cybergames" [16] a fully automated cybergaming environment called BRAWL. This experiment involved "Blue Bot" (defender) and "Red Bot" (attacker). The attacker conducted offensive actions automatically generated, using CALDERA, and targeting hosts on a gameboard. At the same time, the defender extracted event logs and applied detection rules. The attacker's actions on the gameboard allowed them to produce a system logs dataset.

Another approach is to generate a dataset from atomic offensive actions previously detected and analyzed by defenders.

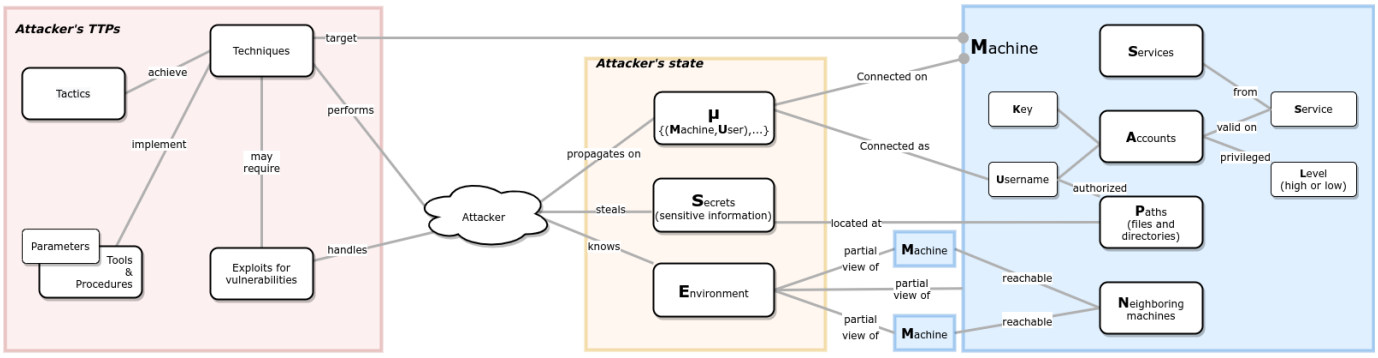

Fig. 1. Model overview

The project "Security Datasets" [17] led by Rodriguez aims to record system traces following the execution of well-known techniques (*i.e.*, custom procedures) or the usage of popular offensive tools like those used by APT actors. These traces can then be exploited individually to create detection rules for SIEM, for example.

APT-like actors observation on dedicated platform

DARPA's project "Operationally Transparent Cyber (OpTC)" [18] is a large dataset shared with the scientific community in 2020. It consists of 17.4 billion events [19] (network and host logs) collected on dedicated infrastructure with 500 to 1000 client machines. During three days, randomly chosen machines were attacked by a *Red Team*.

In [20], Myneni *et al.* proposed a dataset to study APT and for the development of machine learning technologies suitable for APT attack detection. During the period of the logs' capture, an internal *Red Team* has been engaged. They performed attacks on the targeted network, especially related to Reconnaissance, Foothold Establishment, Lateral Movement, and Data Exfiltration. The infrastructure used is not clearly defined. Moreover, the authors use only one internal *Red Team* that may introduce biases in the dataset due to the lack of diversity of TTPs and that does not express the richness of the attack possibilities and their variability.

Datasets allowing the study of APT attacks are thus valuable from a scientific point of view because they are precise and allow for the reproducibility of experiments with a standard reference. However, currently available datasets remain insufficient, poorly documented (description of the architecture, attack procedures), and limited in diversity (*i.e.*, only one attack per architecture).

This article presents an experiment with APT-like actors on a dedicated platform. For this operation, we specifically hired *Red teamers* to whom we gave access to our platform, and we asked them to compromise machines left vulnerable on purpose. The promised financial compensation as the participant reaches objectives adds the motivation aspect, characteristic of APT actors.

In order for this dataset to be used efficiently in future works, it comes with a formal description of the targeted infrastructure. The intuition of our proposal is that an attack on a target system can be understood as an execution of a program on an architecture. In programming language security, the semantics of a program allow describing the processes that a computer follows when executing a program. Such tools do not exist today to describe an attack formally. The MITRE ATT&CK matrix is a significant step in this direction [21]. It allows distinguishing the attack procedures according to the attack pattern, the underlying architecture, and the attacker's intention. This understanding has already allowed Choi *et al.* in [22] to propose an automatic generation method for various attack sequences. We think that it is necessary to go even further in the formalization. Thus, our contribution is a framework allowing the targeted infrastructure description and the attacker's evolution during their campaign.

III. Infrastructure Model and Architecture

In this section, we present the infrastructure deployed on each instance in the PWNJUTSU experiment. First, we provide a generic formal description of the infrastructure. Its genericity and flexibility allow us to describe different infrastructures in which attackers can operate actions to achieve their objectives.

The purpose of this formalization is to highlight the key elements necessary to describe an architecture that is likely to be the target of an APT attacker. In the rest of this section, we describe the practical deployment of this architecture in the context of the PWNJUTSU experiment.

A. Infrastructure

As shown on Figure 1, we represent the targeted infrastructure (*i.e.*, the victim's network) through the machines, network connectivities between the machines, the user accounts, the files and directories, and services that this infrastructure contains and that the attacker can exploit to progress towards their objectives. The network is represented through a graph of machines which we describe in the following.

A machine is a tuple $\mathsf{M} = (\mathbb{S}_\mathsf{M}, \mathbb{P}_\mathsf{M}, \mathbb{A}_\mathsf{M}, \mathbb{N}_\mathsf{M})$ where:

- $\mathbb{S}_\mathsf{M}$ is the set of services provided by machine M. We write `svc_name(logical_identifier : value)` the service $\mathsf{s} \in \mathbb{S}_\mathsf{M}$ provided by M. The logical identifier can be `port` from a network perspective, `process` (name or identifier) from a system perspective or `file` (path, filename or hash) from a filesystem perspective. For example, `apache(port : 443)` stands for the network service `apache` listening on the TCP port 443.

- $\mathbb{P}_M$ is the set of all pairs $(p, \mathcal{U})$ such that p is a path for a file or a directory on the filesystem of M. The content of this file or directory is denoted by $*p$ as a pointer can be. The set $\mathcal{U}$ contains all users u allowed to access this file path. Among these files, some are sensitive data that the attacker covets.
- $\mathbb{A}_M$ is a set of accounts specifying secret key k (*i.e.*, passwords, keypairs, tokens, *etc*) and level of privilege ℓ (low, low*[1] or high) of the user u for a specific service s of the machine M. Thus, $\mathbb{A}_M$ is a set of tuples (u, s, k, ℓ). If the service s does not require an authentication then k = `None`. Users, services and privileges of PWNJUTSU network are described in Table II.
- $\mathbb{N}_M$ is the set of neighboring network machines that can be reached from M. Thus, nodes of the network's graph are machines, and there is an edge from a machine M_1 to a machine M_2 iff $M_2 \in \mathbb{N}_{M_1}$.

The PWNJUTSU dataset was obtained from monitoring systems installed on our dedicated infrastructure. The 22 attackers of the experiment had an isolated network that they needed to compromise. We described this network according to our model by four machines whose complete description according to this model is given in Table I. For the sake of readability in this table, "..." stands for shortened paths, and the "⊙" symbol stands for files containing secrets.

B. Deployment

The PWNJUTSU infrastructure has been deployed on a virtualized infrastructure, installed on a dedicated server, and hosted by the french company OVHcloud in a data center. Specifications of this server were:

- CPU AMD EPYC 7371 16-Core;
- RAM 256 GB DDR4;
- HDD 2x4TB; and
- OS VMware ESXi 7

To automate virtual machines preparation and deployment, we used *DevOps* software: Packer, Vagrant, and Ansible.

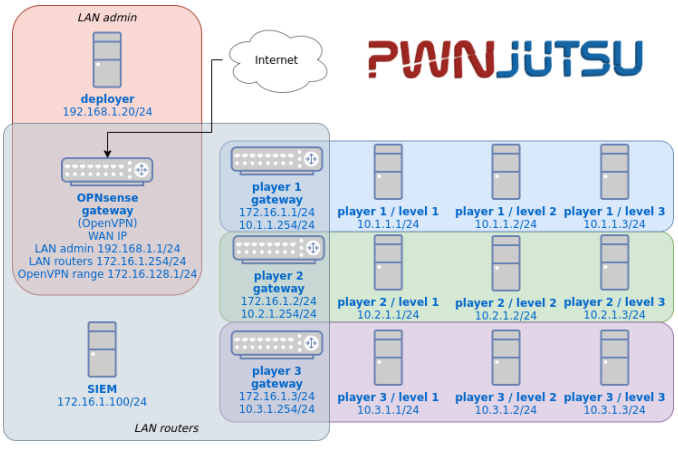

Fig. 2. PWNJUTSU experiment infrastructure

The infrastructure was made up of three types of zones, presented in Figure 2:

- **Administration zone** with a machine called "Deployer", whose role was to prepare participants' instances and to manage their deployment in the playing zone.
- **Routers zone** in which vulnerable zones' gateways have a network interface. It communicates with SIEM (Security Information and Event Management), itself located in this zone. Moreover, VPN clients can communicate with authorized routers in this zone. It was the entry point for experiment participants.
- **Vulnerable zones** in which participants progress individually through three vulnerable machines. Each participant has their dedicated vulnerable zone.

1) Monitoring systems: To accurately observe participants' actions, we installed several monitoring devices that focused on "network" and "system".

On Linux machines, we:

- installed the tool `Snoopy`[2], that logs in `auth.log` all program executions, and their command lines;
- configured the Linux component `auditd`, following the "Best Practice Auditd Configuration"[3];
- added specific file monitoring to these rules to detect read access to the flags hidden in the filesystem;
- enabled logs for all requests received by apache2 (`access.log` and `error.log`) and for all queries received by MySQL;
- also patched SSH server in order to log in `auth.log` login and passwords for all connection attempts.

On the Windows machine, we:

- installed the service `Sysmon`[4], and set the popular configuration template maintained by SwiftOnSecurity[5];
- configured Windows Audit to monitor read access to specific files to detect activities on hidden *flags*;
- collected event logs (System, Application, and Security).

All of the above logs were forwarded and indexed centrally in our SIEM (Splunk) in real-time. In addition, network probes were deliberately set with a high level of verbosity so as not to lose any trace of the participants' activity during the experiment, even if they performed *Defense Evasion* techniques. Moreover, an indiscriminate network capture was recorded for inbound and outbound traffic of each virtual machine. One PCAP file was produced per virtual machine with the native ESXi tool `pktcap-uw`.

2) Targeted machines: We only implemented vulnerabilities that were obvious and easy to exploit. Thus, participants did not lose any time to gain access to a machine, and their effective technical skills did not become a constraint. This choice was made possible because the participants have been pre-selected based on their rankings on an offensive security platform. This aspect of the scenario is essential; if, as a condition, participants were required to overcome technical challenges in order to compromise machines, the scenario would have favored a few elite participants, each in their respective specialty.

[1] `low*` stands for users allowed to elevate their privileges (e.g. *sudoers*)

[2] https://github.com/a2o/snoopy
[3] https://github.com/Neo23x0/auditd/blob/master/audit.rules
[4] https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon
[5] https://github.com/SwiftOnSecurity/sysmon-config

TABLE I
THE FOUR MACHINES COMPOSING THE PWNJUTSU TARGET ARCHITECTURE

P12 VPN address - 172.16.128.112			
n12-gateway - 172.16.1.12, 10.12.1.254			
$\mathbb{S}_{n12-gateway}$ (Services)	$\mathbb{P}_{n12-gateway}$ (Paths)	$\mathbb{A}_{n12-gateway}$ (Accounts)	$\mathbb{N}_{n12-gateway}$ (Neighboring)
∅	∅	Refer to Table II	n12-vm1
n12-vm1 - 10.12.1.1 - Linux Ubuntu 14.04			
$\mathbb{S}_{n12-vm1}$ (Services)	$\mathbb{P}_{n12-vm1}$ (Paths)	$\mathbb{A}_{n12-vm1}$ (Accounts)	$\mathbb{N}_{n12-vm1}$ (Neighboring)
continuum(port:8080) ssh(port:22) apache(port:80) unrealircd(port:6697) rails(port:3500) mysql(port:3306)	(/home/boba_fett/flag.txt$^{\odot}$, {root, boba_fett}) (/root/flag.txt$^{\odot}$, {root}) (/home/han_solo/flag.txt$^{\odot}$, {root, han_solo}) (/home/lando_calrissian/flag.txt$^{\odot}$, {root, lando_calrissian}) (/home/vagrant/flag.txt$^{\odot}$, {root, vagrant}) (/opt/unrealircd/Unreal3.2/flag.txt$^{\odot}$, {root, boba_fett}) (/opt/apache_continuum/.../flag.txt$^{\odot}$, {root}) (/opt/readme_app/flag.txt$^{\odot}$, {root, chewbacca}) (/etc/shadow, {root})	Refer to Table II	n12-vm2
n12-vm2 - 10.12.1.2 - Windows 2008			
$\mathbb{S}_{n12-vm2}$ (Services)	$\mathbb{P}_{n12-vm2}$ (Paths)	$\mathbb{A}_{n12-vm2}$ (Accounts)	$\mathbb{N}_{n12-vm2}$ (Neighboring)
tomcat(port:8080) webdav(port:80) elasticsearch(port:9200) ssh(port:22) wmi/dcom(port:135) netbios(port:139) smb(port:445) winrm(port:5985) rdp(port:3389)	(C:/Users/Administrator/.../password_flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C:/Windows/System32/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C:/Program Files/.../boba_fett/flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, boba_fett}) (C:/Program Files/.../lando.../flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, lando_calrissian}) (C:/Program Files/.../tomcat/.../flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C:/wamp/www/uploads/.flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator, NT AUTHORITY/LOCAL SERVICE}) (C:/Program Files/elastic.../flag.txt$^{\odot}$, {NT AUTHORITY/SYSTEM, Administrator}) (C:/Windows/System32/config/SAM,{NT AUTHORITY/SYSTEM, Administrator}) (C:/Program Files/.../boba_fett/id_rsa,{NT AUTHORITY/SYSTEM, Administrator, boba_fett})	Refer to Table II	n12-vm1 n12-vm3
n12-vm3 - 10.12.1.3 - Linux Ubuntu 20.04			
$\mathbb{S}_{n12-vm3}$ (Services)	$\mathbb{P}_{n12-vm3}$ (Paths)	$\mathbb{A}_{n12-vm3}$ (Accounts)	$\mathbb{N}_{n12-vm3}$ (Neighboring)
vsftpd(port:21) ssh(port:22) apache(port:80) vnc(port:5900)	(/var/www/html/drupal/flag.txt$^{\odot}$, {root, www-data}) (/usr/share/vsftpd/flag.txt$^{\odot}$, {root, ftp}) (/home/captain/Desktop/flag.txt$^{\odot}$, {root, captain}) (/home/boba_fett/flag.txt$^{\odot}$, {root, boba_fett}) (/root/final_flag.txt$^{\odot}$, {root}) (/etc/shadow, {root}) (/var/www/html/backup.tar.gz, {root, www-data}) (/home/captain/.bash_history, {root, captain})	Refer to Table II	n12-vm1 n12-vm2

TABLE II
SERVICES, USERS, KEYS AND PRIVILEGES ($\mathbb{A}_M$) ON MACHINES

$\mathbb{A}_{n12-gateway}$			
u	s	k	ℓ
anonymous	network	None	low
$\mathbb{A}_{n12-vm1}$			
u	s	k	ℓ
root	continuum(port:8080)	[redacted]	high
vagrant	ssh(port:22)	[redacted]	low*
han_solo	ssh(port:22)	[redacted]	low*
lando_calrissian	ssh(port:22)	[redacted]	low
boba_fett	ssh(port:22)	[redacted]	low
www-data	apache(port:80)	None	low
mysql	mysql(port:3306)	None	low
boba_fett	unrealirc(port:6697)	[redacted]	low
chewbacca	rails(port:3500)	[redacted]	low
$\mathbb{A}_{n12-vm2}$			
u	s	k	ℓ
tomcat	tomcat(port:8080)	[redacted]	high
NT AUTHORITY\LOCAL SERVICE	webdav(port:80)	None	low*
NT AUTHORITY\SYSTEM	elastic(port:9200)	None	high
anonymous	smb(port:445)	None	low
boba_fett	smb(port:445)	[redacted]	low
lando_calrissian	smb(port:445)	[redacted]	low
Administrator	smb(port:445)	[redacted]	high
boba_fett	wmi/dcom(port:135)	[redacted]	low
lando_calrissian	wmi/dcom(port:135)	[redacted]	low
Administrator	wmi/dcom(port:135)	[redacted]	high
boba_fett	netbios(port:139)	[redacted]	low
lando_calrissian	netbios(port:139)	[redacted]	low
Administrator	netbios(port:139)	[redacted]	high
boba_fett	ssh(port:22)	[redacted]	low
lando_calrissian	ssh(port:22)	[redacted]	low
Administrator	ssh(port:22)	[redacted]	high
Guest	rdp(port:3389)	None	low
boba_fett	rdp(port:3389)	[redacted]	low
lando_calrissian	rdp(port:3389)	[redacted]	low
Administrator	rdp(port:3389)	[redacted]	high
$\mathbb{A}_{n12-vm3}$			
u	s	k	ℓ
ftp	vsftpd(port:21)	[redacted]	low*
root	ssh(port:22)	[redacted]	high
www-data	apache/drupal(port:80)	None	low
captain	vnc(port:5900)	[redacted]	low

Thus, we would not have been able to collect data logs and preserve the spirit of this experiment (*e.g.*, the network propagation procedures related to lateral movement, credential access, privilege escalation, discovery, and persistence).

The machines `n12-vm1` and `n12-vm2` are based respectively on Linux and Windows of the project Metasploitable 3 that we have refined to keep only services and accounts that were relevant for the PWNJUTSU experiment. This choice was motivated by the fact that we wanted to observe participants' behavior during the *Network Propagation* phase. Thus, some tactics (*i.e.*, stages of kill chains) were not applicable because attackers used these tactics to reach their goals during earlier or later operational phases [23], for example *Reconnaissance*, *Initial Access*, *Asset Dominance*, and even those related to the victim's *Network Exploitation* phase in order to achieve final objective.

Machine `n12-vm3` is a custom Linux Ubuntu system on which we have implemented the following vulnerabilities:

- `vsftpd` (CVE-2011-2523)
- `drupal` (CVE-2018-7600)
- `ssh` (weak root password)
- `vnc` (weak passphrase)

Finally, on each machine, flag files have been hidden inside *home directories* or inside *working directories* of vulnerable services. Participants were asked to collect these files identifiable by filename ending with `flag.txt`. We considered them as indicators that testify to the participant's tactical progress. These files are the attacker's objectives for the PWNJUTSU experiment. They are part of the secrets from sensitive files discovered by a successful attacker.

IV. Attacker progression model and panel

We propose a progression model of an APT-like attacker in a compromised network. We also briefly present attackers enrolled in the experiment. The model has led us to propose semantics of attack techniques. It has allowed us to formally and precisely represent the complete attack campaign of an attacker in the PWNJUTSU experiment, explained in Section VI. We use here the classical notations used in semantics, which are based on the definition of transition relations between states of the system. These generic notations are independent of any implementation and therefore compatible with various tools which are dedicated to specific applications, such as PDDL [24], Factored MDPs, LPMLN, *etc*). Using a mathematical formalism to describe the semantics of the techniques performed by the attacker makes it possible to have a complete and unambiguous specification of these techniques and makes it possible to build models on which real reasoning can be conducted.

A. Attacker description

As an attacker evolves in a compromised network, their knowledge about the network matures throughout the attack time span. The attacker's knowledge of the network is initially imperfect. This knowledge is refined through the use of discovery techniques that increase the attacker's knowledge of the network's organization, data, services, and user accounts [25]. To capture this idea, we use the notion of partial knowledge about a machine M. We write it with the tuple:

$$\lfloor \mathsf{M} \rfloor = (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor)$$

specifying that the attacker knows part of services, paths, accounts, and neighboring machines. These sets may contain some elements from the infrastructure or elements added by the attacker into the victim's network, which is only known by the attacker (*i.e.*, backdoors, illegitimate accounts, *etc*). Moreover, $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor$ may contain tuples $(\mathsf{u}, \mathsf{s}, \mathsf{k}, \ell)$ in which s, k, or ℓ can be $\perp$ if this information is unknown.

When updating the partial knowledge of a machine M by considering a set P of new paths, we write $\lfloor \mathbb{P}_{\mathsf{M}} \rfloor \oplus P$ the set of paths obtained by adding for each $(\mathsf{p}, \mathcal{U}) \in P$, new users of $\mathcal{U}$ if p already occurs in $\lfloor \mathbb{P}_{\mathsf{M}} \rfloor$ and by adding path $(\mathsf{p}, \mathcal{U})$ to $\lfloor \mathbb{P}_{\mathsf{M}} \rfloor$ otherwise.

For example,

$$\{(\mathsf{p}_1, \{\mathsf{u}_1, \mathsf{u}_2\})\} \oplus \{(\mathsf{p}_1, \{\mathsf{u}_2, \mathsf{u}_3\}), (\mathsf{p}_2, \{\mathsf{u}_1\})\} = \{(\mathsf{p}_1, \{\mathsf{u}_1, \mathsf{u}_2, \mathsf{u}_3\}), (\mathsf{p}_2, \{\mathsf{u}_1\})\}$$

Similarly, given a set A of new accounts, we write $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor \oplus A$ the set of accounts obtained by replacing $\perp$ values occurring in accounts of users u already contained in $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor$ with new values provided by A for u and by adding to $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor$ accounts in A which do not already occur in $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor$.

For example,

$$\{(\mathsf{u}, \mathsf{s}_1, \perp, \ell)\} \oplus \{(\mathsf{u}, \mathsf{s}_1, \mathsf{k}_1, \ell), (\mathsf{u}, \mathsf{s}_2, \mathsf{k}_2, \ell)\} = \{(\mathsf{u}, \mathsf{s}_1, \mathsf{k}_1, \ell), (\mathsf{u}, \mathsf{s}_2, \mathsf{k}_2, \ell)\}$$

We choose here to represent an attack as the progression of the attacker's conquest over the network. At each stage of progression, this conquest (or simply the attacker) is defined through their *muSE*: a tuple

$$(\mu, \mathcal{S}, \mathcal{E})$$

where:

- μ is a set containing all pairs (M, u) such that the attacker owned an access on M with the user identity u
- $\mathcal{S}$ is the set of file contents from files located at paths and containing secrets that are part of the attacker's objective (*i.e.*, all $*\mathsf{p}$ such that $\mathsf{p}^{\odot}$).
- $\mathcal{E}$ is the environment knowledge acquired by the attacker and is defined by a mapping providing a partial view, written $\lfloor \mathsf{M}' \rfloor_{\mathcal{E}}$ for each machine M'.

During a campaign, the attacker can use their tactical capabilities, which are, on the one hand, the attack techniques that they master and, on the other hand, the vulnerabilities for which they have an exploit. An exploit is a code that takes advantage of a software vulnerability or security flaw[6]. In the following, we will note `Exploits` the database of exploits mastered by the attacker (for example, it could be the public Exploit-DB database[7]). Techniques will be detailed in section V and illustrated by an example in section VI.

B. PWNJUTSU panel

To conduct our experiment, we needed attackers who were technically able to behave like an APT. Thus we looked for professional *Red Teamers* within the YesWeHack platform's TOP 100[8]. Then, we hired 22 attackers from nine nationalities and asked them to participate in this experiment. To motivate them, as *Threat Actors* could be, we rewarded them individually according to a progressive pay scale based on the number of machines they compromised. This reward became the participant's effective gain. We gave them intermediate objectives to claim their reward: collecting "flag" files.

To beta test our architecture, about ten participants have played with the PWNJUTSU scenario. This first phase allows us to enhance the quality of the experiment scenario and log systems. During the second phase of the experiment, 22 participants attacked the stable version of the infrastructure. They finished the entire scenario by reaching the final flag. PWNJUTSU dataset is the concatenation of those 22 participants' traces.

The typical profile of a participant from the panel is:

- 25-35 years old (63%)
- Level 6-7 diploma (91%)
- Ethical hacking certified (64%)
- Self-taught in offensive security (100%)

In section VI, we use our formal model to detail step by step the attack performed by one of these attackers and give an overview of data produced during their attack.

[6] https://www.trendmicro.com/vinfo/us/security/definition/exploit
[7] https://www.exploit-db.com
[8] https://yeswehack.com

V. Semantics of attack techniques

To progress towards their objectives, the attacker performs attack techniques that achieve tactical goals (*e.g.*, obtain new accesses, unveil account credentials, gain new privileges, acquire knowledge). Each technique execution leads to a change in the attacker's perception, or even to improve their control over the victim's network. The MITRE ATT&CK® [21] gives a comprehensive list of attack techniques known to date. "Hacking tools" and malware contain concrete implementations of those techniques called procedures. There are, therefore, several procedures for the same technique. Before their instantiation, techniques are parameterized according to their target in the victim's network (*e.g.*, services, machines, users) [26]. The different techniques allow the attacker to gain control over different accounts and privileges (evolution of μ_i in our model of the attacker), discover new secrets (evolution of $\mathcal{S}_i$), or refine their knowledge of the targeted network (evolution of $\mathcal{E}_i$). We define here a formal operational semantics of techniques that can be viewed as specifications for procedures.

An attack progression is a sequence of transitions:

$$(\mu_i, \mathcal{S}_i, \mathcal{E}_i) \xrightarrow{\mathsf{t}(params)} (\mu_{i+1}, \mathcal{S}_{i+1}, \mathcal{E}_{i+1})$$

starting from an initial state $(\mu_0, \emptyset, \mathcal{E}_0)$ where $\mu_0 = \{(\mathsf{M}_0, \mathsf{u}_0)\}$ meaning that the attacker has obtained initial access over the machine M_0 as the user u_0 and has already acquired knowledge about the infrastructure in $\mathcal{E}_0$.

The execution of a technique t with parameters *params* leads the attacker from a state $(\mu_i, \mathcal{S}_i, \mathcal{E}_i)$ to a state $(\mu_{i+1}, \mathcal{S}_{i+1}, \mathcal{E}_{i+1})$ where the sets $\mu_i, \mathcal{S}_i$ or $\mathcal{E}_i$ are updated according to the expected effect of the technique executed if the preconditions on $(\mu_i, \mathcal{S}_i, \mathcal{E}_i)$ are met. To formally define the update of $\mathcal{E}$, given a machine M', we write

$$\mathcal{E}' = \mathcal{E}[\mathsf{M}' \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor)]$$

the knowledge environment $\mathcal{E}'$ such that for all machine M:

$$\lfloor \mathsf{M} \rfloor_{\mathcal{E}'} = \begin{cases} (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor) & \text{if } \mathsf{M} = \mathsf{M}' \\ \lfloor \mathsf{M} \rfloor_{\mathcal{E}} & \text{otherwise} \end{cases}$$

Such a transition relation is formally defined in the following and corresponds to the operational semantics of techniques used to describe an attack progression. In the long run, this work may allow the identification of all possible attack paths on a given architecture through model checking techniques.

We provide the semantics for 13 techniques, achieving five different relevant tactics. We chose these tactics for two reasons: not only do they make sense in the *Network Propagation* phase on which the PWNJUTSU's scenario focused, but they also allowed us to collect related traces thanks to deployed sensors. Among all tactics described in the MITRE ATT&CK matrix, we focused on those unavoidable for an attacker in the network propagation phase. However, the model proposed in this paper is flexible enough to specify techniques that achieve other tactics, which could be employed by attackers in other operational phases of APT campaigns.

A. Lateral Movement

MITRE defines the *Lateral Movement* as a tactic consisting of techniques that attackers use to enter and control remote systems on a network. In other words, the use of this tactic allows the attacker to gain access to a M' machine under the user identity u'. In our semantics, this is expressed as an evolution of μ_i. We describe here two famous techniques achieving *Lateral Movement*:

- *using Remote Services (T1021)*
- *using Exploitation of Remote Services (T1210)*

For the first technique (T1021), the attacker can gain access to a machine M' by using a secret k' for user u' on service s, if the attacker knows the user u'. These four elements are part of the parameters needed to perform this technique. Moreover, three preconditions have to be satisfied:

- the machine M and the user u have to be in the propagation area of the attacker, meaning that they have access to this machine as this user. In other words, (M, u) is the "location" from where the attacker uses the technique.
- the targeted machine M' has to be reachable (network) from M, and the attacker has to be aware of it.
- attacker has to be aware of the existence of the account $(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell)$ on the machine M'.

After executing this technique, the attacker will expand their propagation area with the pair $(\mathsf{M}', \mathsf{u}')$. This first technique is formally described in Table III.

TABLE III
Lateral movement with Remote Services

Technique	$T$1021: Remote Services
Tactic	Lateral movement
Description	Gain access to a machine M' by using secret key k' for user u' on service s.
Parameters	M, u, M', u', k', s
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$ and $(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu', \mathcal{S}, \mathcal{E})$ where $\mu' = \mu \cup \{(\mathsf{M}', \mathsf{u}')\}$
Variants	T1021.001 (s = `RDP`); T1021.002 (s = `SMB`); T1021.003 (s = `DCOM`); T1021.004 (s = `SSH`); T1021.005 (s = `VNC`); T1021.006 (s = `WinRM`);

TABLE IV
Lateral movement using Exploitation of Remote Services

Technique	$T$1210: Exploitation of Remote Services
Tactic	Lateral movement
Description	Gain access to a machine M' by exploiting remotely a vulnerability x on an exposed service s.
Parameters	M, u, M', s, x
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$ and $\mathsf{x} \in \mathtt{Exploits}(\mathsf{s})$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu', \mathcal{S}, \mathcal{E})$ where $\mu' = \mu \cup \{(\mathsf{M}', \mathsf{u}')\}$ with $(\mathsf{u}', \mathsf{s}, \mathsf{k}, \ell) \in \mathbb{A}_{\mathsf{M}'}$
Variants	Post-authenticated vulnerability requires additional parameters u'' and k'' such that $(\mathsf{u}'', \mathsf{s}, \mathsf{k}'', \ell'') \in \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$ (Note: u'' can be u')

The second technique (T1210), formally described in Table IV, is related to the remote exploitation of a vulnerability to a remote service. Similar to the first technique, this one has exceptions: credentials u'' and k'' are mandatory only for post-authenticated vulnerabilities. The attacker has to specify the exploit x as a parameter used for this technique execution.

The attacker relies on `Exploits(s)`, which is the set of *exploits* for vulnerabilities applicable on the service s that they master. This set has to be, at least, non-empty.

After executing this technique, the attacker takes the identity of a user u' on the machine M', while they do not know the secret key k' for this account.

B. Credential access

MITRE defines the *Credential Access* as a tactic consisting of techniques for stealing credentials like account names and passwords. In our model, the attacker uses this tactic when they collect credentials of a service s. We describe here two techniques achieving *Credential Access*:

- using *Unsecured Credentials (T1552)*
- using *Brute Force (T1110)*

TABLE V
CREDENTIAL ACCESS USING UNSECURED CREDENTIALS

Technique	$T1552$: Unsecured Credentials
Tactic	Credential access
Description	Collect all credentials linked to service s on M.
Parameters	M, u, s
Preconditions	$(\mathsf{M},\mathsf{u}) \in \mu$, $(\mathsf{u},\mathsf{s},\mathsf{k},\ell) \in \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu,\mathcal{S},\mathcal{E}) \hookrightarrow (\mu,\mathcal{S},\mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{u}',\mathsf{s},\mathsf{k}',\ell') \in \mathbb{A}_{\mathsf{M}}\}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}})]$
Variants	$T1552.001$ (with s = `filesystem`); $T1552.003$ (with s = `bash`); $T1552.004$ (with s = `filesystem` on private key file); $T1003.008$ (with $\mathsf{s} = \mathtt{OS}_{\mathtt{Linux}}$ and ℓ = `high`); $T1003.002$ (with $\mathsf{s} = \mathtt{OS}_{\mathtt{Windows}}$ and ℓ = `high`);

When they execute the first one (T1552), formally described in Table V, the attacker collects all the credentials linked to service s on the machine M where they are connected as the user u. To execute this technique, preconditions have to be satisfied:

- the machine M and the user u have to be in the propagation area of the attacker, meaning that they have access to this machine as this user;
- the account related to the user identity u has to be a valid account for the service, but it works even if the attacker does not know the secret key for this account (*i.e.*, $\mathsf{k} = \bot$).

After executing this technique, the attacker will increase their knowledge of the environment $\mathcal{E}$ for the machine they targeted with the new account's credentials.

For the second technique (T1110), the attacker knows the existence of user u' for service s' on machine M' and they can reach this machine from their execution point (*i.e.*, from (M,u)).

After executing this technique, the attacker will discover the aspired secret key and capitalize it in their knowledge of the targeted machine's environment. This technique is formally described in Table VI.

TABLE VI
CREDENTIAL ACCESS USING BRUTE FORCE

Technique	$T1110$: Brute Force
Tactic	Credential Access
Description	Get the secret k' of user u' of service s' on machine M'.
Parameters	M, u, M', u', s'
Preconditions	$(\mathsf{M},\mathsf{u}) \in \mu$, $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$ and $(\mathsf{u}',\mathsf{s}',\bot,\ell') \in \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$
Transition	$(\mu,\mathcal{S},\mathcal{E}) \hookrightarrow (\mu,\mathcal{S},\mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M}' \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{u}',\mathsf{s}',\mathsf{k}',\ell') \in \mathbb{A}_{\mathsf{M}'}\}, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor_{\mathcal{E}})]$
Variants	$T1110.001$ with k obtained by guessing; $T1110.002$ with k obtained by cracking;

C. Privilege Escalation

MITRE defines the *Privilege Escalation* as a tactic consisting of techniques that adversaries use to gain higher-level permissions on a system, according to our model, when the attacker gets high user privileges u' from a lower privilege user u. We describe one technique that allows achieving Privilege Escalation using *Abuse Elevation Control Mechanism (T1548)*.

This technique can be executed if the user u with whom the attacker is connected on the machine M has a privilege level set at `low*`. It means that the user is authorized to escalate their privilege (*i.e.*, `sudo` command).

After the execution of this technique, the attacker is still connected to the machine M, but with a user u' who has high privileges on the machine. This technique is formally described in Table VII.

TABLE VII
PRIVILEGE ESCALATION USING ABUSE ELEVATION CONTROL MECHANISM

Technique	$T1548$: Abuse Elevation Control Mechanism
Tactic	Privilege escalation
Description	Get the high privileges of u' from a low privilege user u on M.
Parameters	M, u
Preconditions	$(\mathsf{M},\mathsf{u}) \in \mu$, $(\mathsf{u},\mathtt{OS},\mathsf{k},\mathtt{low*}) \in \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu,\mathcal{S},\mathcal{E}) \hookrightarrow (\mu',\mathcal{S},\mathcal{E})$ where $\mu' = \mu \cup \{(\mathsf{M},\mathsf{u}')\}$ with $(\mathsf{u}',\mathtt{OS},\mathsf{k}',\mathtt{high}) \in \mathbb{A}_{\mathsf{M}}$
Variants	$T1548.003$ with $\mathsf{s} = \mathtt{OS}_{\mathtt{Linux}}$ (`sudo` command)

D. Discovery

MITRE defines the *Discovery* as a tactic consisting of techniques an attacker may use to gain knowledge about the system and internal network. In our model, it leads to enriching the attacker's knowledge of the environment $\mathcal{E}$. This knowledge can be related to services $\lfloor \mathbb{S}_{\mathsf{M}} \rfloor$, files and directories paths $\lfloor \mathbb{P}_{\mathsf{M}} \rfloor$, accounts $\lfloor \mathbb{A}_{\mathsf{M}} \rfloor$, or neighboring machines $\lfloor \mathbb{N}_{\mathsf{M}} \rfloor$.

We described five techniques achieving *Discovery*:

- File and Directory Discovery (local and remote) (T1083)
- Network Service Scanning (T1046)
- Remote System Discovery (T1018)
- Account Discovery (local and remote) (T1087)
- System Service Directory (T1007)

Table VIII and Table IX present the same technique (T1083), but our semantics lead us to differentiate its local version from its remote version. In the first case (T1083), the attacker discovers files and directories on the filesystem of the local machine M at file path p. The current machine M and the current user u have to be in the attacker's propagation area μ. The user u also has to be in the set of users $\mathcal{U}$ who are allowed to access the file at path p. After executing this technique, the attacker improves their knowledge of the environment $\mathcal{E}$ with the files and directories discovered. If applicable (*i.e.*, if $\mathsf{p}.^{\odot}x$) they also collect new secrets in $\mathcal{S}$.

In the second case (T1083'), the difference is that the targeted machine M' is remote and that the attacker must provide a valid account (with a user u' and a key k') to connect remote network service s.

TABLE VIII
DISCOVERY USING FILE AND DIRECTORY DISCOVERY (LOCAL)

Technique	$T$1083: File and Directory Discovery (Local)
Tactic	Discovery
Description	Get all files and directories on the filesystem of machine M at path p.
Parameters	M, u, p
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $(\mathsf{p}, \mathcal{U}) \in \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}$ and $\mathsf{u} \in \mathcal{U}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}', \mathcal{E}')$ with $\mathcal{S}' = \mathcal{S} \cup \{*\mathsf{p}.x \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathsf{M}}, \mathsf{u} \in \mathcal{U}' \text{ and } \mathsf{p}.^{\odot}x\}$ and $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{p}.x, \mathcal{U}') \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathsf{M}} \text{ and } \mathsf{u} \in \mathcal{U}'\}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}})]$
Variants	$T$1083', described below, is the same technique but for remote filesystems.

TABLE IX
DISCOVERY USING FILE AND DIRECTORY DISCOVERY (REMOTE)

Technique	$T$1083': File and Directory Discovery (Remote)
Tactic	Discovery
Description	Get all files and directories on the remote filesystem of machine M' at path p.
Parameters	M, u, M', u', k', p, s
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $(\mathsf{p}, \mathcal{U}) \in \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$ and $\mathsf{s} \in \lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$, s is network service (e.g., Apache, SMB, *etc*), $(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$ with $\mathsf{k}' \neq \perp$, $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$ and $\mathsf{u}' \in \mathcal{U}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}', \mathcal{E}')$ with $\mathcal{S}' = \mathcal{S} \cup \{*\mathsf{p}.x \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathsf{M}'}, \mathsf{u}' \in \mathcal{U}' \text{ and } (^{\odot}\mathsf{p}.x)\}$ and $\mathcal{E}' = \mathcal{E}[\mathsf{M}' \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{p}.x, \mathcal{U}') \mid (\mathsf{p}.x, \mathcal{U}') \in \mathbb{P}_{\mathsf{M}'} \text{ and } \mathsf{u}' \in \mathcal{U}'\}, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor_{\mathcal{E}})]$
Variants	$T$1135 where p is the path of a network share.

The technique (T1046) presented in Table X allows an attacker to discover all network services of a remote machine by scanning its network ports (between 0 and 65535).

To perform this technique, the targeted machine M' has to be reachable from the machine M, which must be in the attacker's propagation area μ. After executing this technique, the attacker improves their knowledge $\mathcal{E}$ of the environment, particularly of services installed on the targeted machine $\mathbb{S}_{\mathsf{M}'}$.

TABLE X
DISCOVERY USING NETWORK SERVICE SCANNING

Technique	$T$1046: Network Service Scanning
Tactic	Discovery
Description	Get all network services of remote machine M' by browsing the network ports' namespace $\Delta \subseteq \{0, \cdots, 65535\}$.
Parameters	M, u, M', Δ
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M}' \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}} \cup \{\mathsf{s}(\text{port} : i) \mid i \in \Delta\}, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor_{\mathcal{E}})]$

Thanks to the technique (T1018) presented in Table XI, the attacker can discover neighboring machines of the current machine M, which is in their propagation area μ.

After this execution, the attacker improves their knowledge $\mathcal{E}$ by adding the set of all neighboring machines $\mathbb{N}_{\mathsf{M}}$.

TABLE XI
DISCOVERY USING REMOTE SYSTEM DISCOVERY

Technique	$T$1018: Remote System Discovery
Tactic	Discovery
Description	Get all the neighboring machines of machine M.
Parameters	M, u
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}} \cup \mathbb{N}_{\mathsf{M}})]$
Variants	$T$1049 (System Network Connections Discovery): is for neighboring machines discovery thanks to established network connections.

Table XII and Table XIII present two forms of the technique (T1087) for local or remote execution respectively. These techniques allow the attacker to get all service accounts from s on machine M (local) or machine M' (remote).

To execute this technique, preconditions have to be satisfied by the attacker:

- the machine M must be in their propagation area μ;
- they have to know the actual existence of the service s on the targeted machine; and
- they need access to the service s with the current user u.

TABLE XII
DISCOVERY USING ACCOUNT DISCOVERY (LOCAL)

Technique	$T$1087: Account Discovery (local)
Tactic	Discovery
Description	Get all service accounts from s on machine M.
Parameters	M, u, s
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}$ and $(\mathsf{u}, \mathsf{s}, \mathsf{k}, \ell) \in \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{u}', \mathsf{s}, \perp, \ell) \mid (\mathsf{u}', \mathsf{s}, \mathsf{k}, \ell) \in \mathbb{A}_{\mathsf{M}}\}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}})]$
Variants	$T$1087', described below, is the same technique but for remote services.

The difference with the alternative techniques (T1087') for a remote machine M' is that the attacker needs to be aware of the network service s. The attacker also needs to have access to this service thanks to an account with the user u' and the key k'. The machine M' must be reachable from the machine M.

TABLE XIII
DISCOVERY USING ACCOUNT DISCOVERY (REMOTE)

Technique	$T1087'$: Account Discovery (Remote)
Tactic	Discovery
Description	Get all service accounts from s on machine M'.
Parameters	M, u, M', u', k', s
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $\mathsf{s} \in \lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$, s is network service (e.g., Apache, SMB, *etc*) and $(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}}$ with $\mathsf{k}' \neq \perp$ and $\mathsf{M}' \in \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M}' \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}'} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}'} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{u}', \mathsf{s}, \perp, \ell) \mid (\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell) \in \mathbb{A}_{\mathsf{M}'}\}, \lfloor \mathbb{N}_{\mathsf{M}'} \rfloor_{\mathcal{E}})]$

After executing these techniques, the attacker improves their knowledge $\mathcal{E}$ with all service accounts configured in the service s. It is essential to specify that these techniques allow accounts discovery and not keys. To discover the keys related to the accounts, the attacker must perform a *Credential Access* technique (Section V-B).

Finally, the technique formalized in Table XIV allows the attacker to discover all the services installed on the machine M. The precondition to perform this technique is that machine M and the user u must be in the attacker's propagation area μ to perform this technique.

After executing this technique, the attacker improves their knowledge of machine M with the set containing all existing services of the machine $\mathbb{S}_{\mathsf{M}}$.

TABLE XIV
DISCOVERY USING SYSTEM SERVICE DISCOVERY

Technique	$T1007$: System Service Discovery
Tactic	Discovery
Description	Get all services of machine M.
Parameters	M, u
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}} \cup \mathbb{S}_{\mathsf{M}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}})]$
Variants	$T1057$ (Process Discovery): is for service discovery through their process names.

Some other *Discovery* techniques like:

- System Information Discovery (T1082)
- System Network Configuration Discovery (T1016)
- System Owner/User Discovery (T1033)
- Internet Connection Discovery (T1016.001)

are often performed by attackers. They are helpful for them to acquire knowledge about the current machine (part of the situational assessment process). However, their inputs do not contribute to the attacker's tactical progression. That is why we do not describe these techniques with our semantics.

E. Persistence

MITRE defines the *Persistence* as a tactic consisting of techniques that attackers use to retain access to systems or when the attacker manipulates the victim's environment to be able to maintain their access for a long time. We described in Table XV the technique Create Account (T1136), used to achieve *Persistence*.

To perform this technique, the attacker must provide the 4-tuple $(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell')$ that will be created thanks to privileges of the user u, on the local machine M.

Preconditions to execute this technique are:

- the existence of the machine M;
- and the user u in the attacker's propagation area μ.

Moreover, the attacker must have access to a high-privilege account on the machine. After the execution of this technique, the provided account will be created and, because the attacker knows it, it improves their knowledge of the environment $\mathcal{E}$.

TABLE XV
PERSISTENCE USING CREATE ACCOUNT

Technique	$T1136$: Create Account
Tactic	Persistence
Description	Create an account for local service s on machine M.
Parameters	M, u, u', k', ℓ', s
Preconditions	$(\mathsf{M}, \mathsf{u}) \in \mu$, $(\mathsf{u}, \mathsf{s}, \mathsf{k}, \texttt{high}) \in \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}}$
Transition	$(\mu, \mathcal{S}, \mathcal{E}) \hookrightarrow (\mu, \mathcal{S}, \mathcal{E}')$ with $\mathcal{E}' = \mathcal{E}[\mathsf{M} \leftarrow (\lfloor \mathbb{S}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{P}_{\mathsf{M}} \rfloor_{\mathcal{E}}, \lfloor \mathbb{A}_{\mathsf{M}} \rfloor_{\mathcal{E}} \oplus \{(\mathsf{u}', \mathsf{s}, \mathsf{k}', \ell')\}, \lfloor \mathbb{N}_{\mathsf{M}} \rfloor_{\mathcal{E}})]$

VI. PWNJUTSU P12 ATTACK CAMPAIGN

The attacker model (Section IV), as well as the semantics of the attack techniques (Section V), allows us to describe the progression of attackers during their attack campaigns unequivocally. The dataset contains traces of 22 participants, and we have semi-automatically reconstructed the steps of the attack campaign performed by one of those participants during the PWNJUTSU experiment. The reconstruction of this kind of attack is possible thanks to the collected logs exploitation. Then, in order to identify techniques performed by the attacker, we used a tool[9] that allows us to extract *Events of Interest* (EoI) [27] from the dataset. In the following, we detail actions performed by the participant #12, denoted P12. We have chosen to take this participant as an example because their performance is akin to a "smash and grab" robbery. No hesitation, intuitive choices, and goals achieved.

P12 attack can be deconstructed into eighteen steps, which are implementations of the eight techniques that form their campaign-exposed TTP. In Figure 3 we have represented the sequence of techniques carried out by this participant: their operational flow. The figure suggests that the participant relies on a routine of attacks since they tend to perform similar techniques in a similar environment. This representation allows us to identify the following attack pattern.

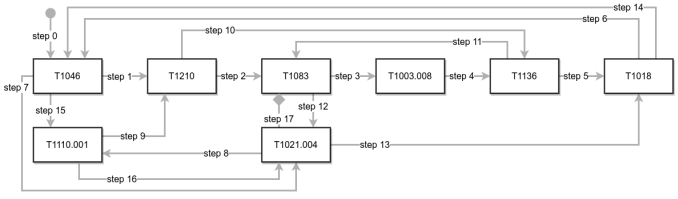

Fig. 3. PWNJUTSU P12 operational flow

[9] https://github.com/wagga40/Zircolite

P12 starts by scanning ports, then they try to exploit a vulnerability to gain access to the next machine. After that, they persist on the machine by dropping a private key, and then they move laterally using SSH. Finally, they collect the coveted secret flags and they start this loop again.

Each step of their attack is transcribed with their procedure's parameters (for the first ones) in Table XVI. In this table, we have also shown some event logs related to these steps, as we can find them in the PWNJUTSU dataset. For example, at `Step 2`, P12 performs, from the machine `n12-gateway`, the technique "Exploitation of Remote Services" on the service `Apache Continuum` which listens on network port `8080` on the machine `n12-vm1`. For that, they used the appropriate exploit from the public database ExploitDB. It produced a trace in the network capture. This trace reveals an `HTTP POST` request to `/continuum/saveInstallation.action`, which is the vulnerable endpoint of the web application.

Furthermore, our model allows representing the propagation area of this attacker in the PWNJUTSU architecture. Figure 4 represents this propagation area through all the pairs (M, u) they control. In other words, the propagation area represents all the connection points from which the attacker can perform techniques. This area is the Holy Grail for an *Incident Response* team that aims to eradicate the threat permanently.

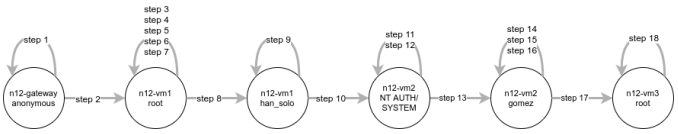

Fig. 4. PWNJUTSU's P12 propagation area

VII. PWNJUTSU BENEFITS TO THE COMMUNITY

In this article, we have chosen to present semantics that make it possible to formally describe both an attack flow and an evolution of an attacker's propagation area. However, we are convinced that many works can also use this dataset. Thus, we have chosen to distribute the raw data collected publicly so that the entire scientific community can take advantage of them with their projects.

A. Dataset records

Sensors deployed on PWNJUTSU infrastructure recorded events, as described in section III-B1. We extracted those events, and we made them available to the scientific community in the form of a downloadable dataset[11]. This data can also be consulted on our site using the search engine provided.

In this dataset, for each of the 22 participants of the experiment, sensors produced the following files:

- JSON Lines file containing system logs from the three vulnerable machines. Overall this represents a total of more than 16 million event logs (`n*-vm1`: 9.2M events, `n*-vm2`: 50k events, `n*-vm3`: 7.2M events).

[10] https://www.exploit-db.com/exploits/39945
[11] https://pwnjutsu.irisa.fr

TABLE XVI
DETAIL ABOUT THE PWNJUTSU P12 ATTACK

Step 0	P12 got an initial access to `n12-gateway`.
Step 1	P12 performed network service scanning (T1046) from `n12-gateway` to `n12-vm1`.
Parameters	M = `n12 – gateway`, u = `anonymous` M' = `n12 – vm1`, Δ = {`top1000portsnmap`}
Trace (net)	`21887 2021-05-09 20:07:49,695678 172.16.128.112 10.12.1.1 TCP 60 42548 → 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1357`
Step 2	P12 exploited remote service (T1210) Apache Continuum (port 8080) from `n12-gateway` to `n12-vm1`.
Parameters	M = `n12 – gateway`, u = `anonymous` M' = `n12 – vm1` s = `continuum(port : 8080)` x = `EDB-ID : 39945`[10]
Trace (net)	`170511 2021-05-09 20:28:15,698503 172.16.128.112 10.12.1.1 HTTP 1411 POST /continuum/saveInstallation.action HTTP/1.1`
Step 3	P12 got a secret flag file (T1083) on `n12 – vm1`.
Parameters	M = `n12 – vm1`, u = `root` p = `/opt/apache_continuum/.../flag.txt`
Trace (sys)	`May 9 20:28:54 n12-vm1 snoopy[1566]: [uid:0 sid:1309 tty:(none) cwd:/opt/apache_continuum/... filename:/bin/cat]: cat flag.txt`
Step 4	P12 got all credentials (T1003.008) of `n12 – vm1` OS.
Parameters	M = `n12 – vm1`, u = `root`, s = $\mathrm{OS}_{\mathrm{Linux}}$
Trace (sys)	`May 9 20:29:51 n12-vm1 snoopy[1569]: [uid:0 sid:1309 tty:(none) cwd:/opt/apache_continuum/... filename:/bin/cat]: cat /etc/shadow`
Step 5	P12 added a private key (from the root user's folder) for the user han_solo on the service SSH (T1136) on machine `n12 – vm1`.
Parameters	M = `n12 – vm1`, u = `root`, u' = `han_solo`, k' = `SSHprivatekey`, ℓ' = `low*`, s = `ssh(port : 22)`
Trace (sys)	`May 9 20:36:16 n12-vm1 snoopy[1593]: [uid:0 sid:1309 tty:(none) cwd:/root filename:/bin/mv]: mv .ssh /home/han_solo/`
Step 6	P12 discovered remote system (T1018) `n12-vm2` from `n12-vm1` using ARP table.
Parameters	M = `n12 – vm1`, u = `root`, M' = `n12 – vm1`
Trace (sys)	`May 9 20:39:45 n12-vm1 snoopy[1622]: [uid:0 sid:1309 tty:(none) cwd:/home/han_solo/.ssh filename:/usr/sbin/arp]: arp -an`
Step 7	P12 performed network service scanning (T1046) from `n12-vm1` to `n12-vm2` as user `root`.
Parameters	M = `n12 – vm1`, u = `root` M' = `n12 – vm2`, Δ = {`top1000portsnmap`}
Trace (sys)	`May 9 20:44:15 n12-vm1 snoopy[1634]: [uid:0 sid:1309 tty:(none) cwd:/home/han_solo/.ssh filename:/usr/bin/nmap]: nmap -sS -vv -Pn -n 10.12.1.2-3`
Step 8	P12 got an access using SSH (T1021.004) service on `n12-vm1` as user `han_solo` with the previously added private key.
Parameters	M = `n12 – gateway`, u = `anonymous` M' = `n12 – vm1`, u' = `han_solo` k' = `SSHprivatekey`, s = `ssh(port : 22)`
Trace (sys)	`May 9 20:44:37 n12-vm1 sshd[1636]: Accepted publickey for han_solo from 172.16.128.112 port 41134 ssh2`
Step 9	P12 bruteforce by guessing (T1110.001) the service `Tomcat/axis2` (port 8080) with the default username `admin` on `n12 – vm2`.
Parameters	M = `n12 – vm1`, u = `han_solo`, M' = `n12 – vm2`, u' = `admin`, s = `tomcat(port : 8080)`
Trace (net)	`182610 2021-05-09 21:02:25,553027 10.12.1.1 10.12.1.2 HTTP 307 POST /axis2/axis2-admin/login HTTP/1.1`
Step 10	P12 exploited post authenticated remote service (T1210) `Tomcat` (port 8080) from `n12 – vm1` to `n12 – vm2`.
Step 11	P12 added an account for the user `gomez` on the service SSH (T1136) on machine `n12 – vm2`.
Step 12	P12 got a secret flag file (T1083) on `n12 – vm2`.
Step 13	P12 got an access using SSH (T1021.004) service on `n12-vm2` as user `gomez`.
Step 14	P12 discovered remote system (T1018) `n12-vm3` from `n12-vm1` using ARP table.
Step 15	P12 performed network service scanning (T1046) from `n12-vm2` to `n12-vm3`.
Step 16	P12 bruteforce by guessing (T1110.001) the service `SSH` (port 22) with the username `root` on `n12 – vm3`.
Step 17	P12 got an access using SSH (T1021.004) service on `n12-vm3` as user `root`.
Step 18	P12 got a secret flag file (T1083) on `n12 – vm3`.

- PCAP files containing raw network traffic captured on both inbound and outbound interfaces. Overall this represents a total of 172 GB of raw data and 17 GB of Zeek[12] analysis results, corresponding to 45 million lines.

Additionally, we produced a reference (`n99`) which is the same instance infrastructure monitored but without any malicious activity.

B. Questions to panel members

Immediately after they participated in the experiment, we questioned panel members through a short survey about their operational habits and tactical preferences. All the PWNJUTSU participants (n = 22) answered this survey.

1) Pivoting: First, we looked at attackers' needs to pivot to progress in-deep in the victim's network. Thus, we asked panel members which features or protocols they used to reach `vm2` and `vm3` machines during the experiment. Multiple answers were allowed. It is important to specify that each of these features or protocols were available. Figure 5 represents their answers. We learned that panel members mostly prefer to use two features among the four proposed: **Proxy SOCKS and Port Forwarding**. We want to emphasize that those features are part of SSH. This information is interesting to consider for real-world information systems when implementing countermeasures, such as protocol bastions machines, or even on detection rules specific to these protocols and features.

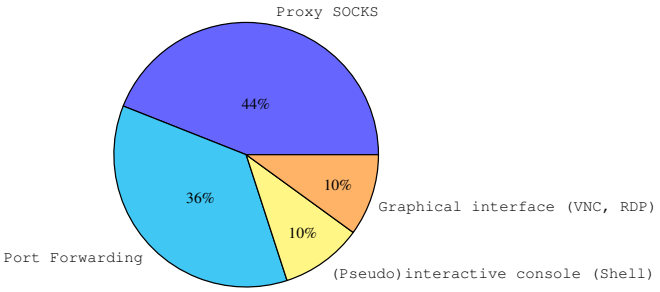

Fig. 5. Pivoting features and protocols used to reach `vm2` and `vm3`

2) Target's Operating System influence: Then, we sought to identify the possible influence that the Operating System (OS) could have on the attacker's behavior. We asked this question to panel members. **95% of them answered "yes"**. It can be interpreted as the lack of OS-agnostic offensive tools or procedures in attackers' arsenals. In the context of an active defense approach (*i.e.*, *Cyber Deception*), this also makes it possible to deploy several kinds of OS to be able to lure attackers into decoys more effectively.

3) Preferred techniques: Finally, we asked panel members to order by preference, during tactical progression, their attack techniques for several tactics:

- *Lateral Movement*,
- *Credential Access*,
- *Privilege Escalation*,
- *Discovery*.

[12] https://zeek.org/

Table XVII presents their answers. We enriched these subjective answers with what we observed on their PWNJUTSU instances. It is interesting to note that the preferred technique to perform *Lateral Movement* actions is SSH, which corresponds to the preferred features and protocol observed in Section VII-B1. We hope these results, which ultimately are priority lists, will motivate the community to enrich collaborative detection initiatives such as the popular Sigma project[13] with new pragmatic rules.

TABLE XVII
PWNJUTSU PANEL PREFERRED TECHNIQUES DURING PROGRESSION

Preferred techniques for *Lateral Movement* actions:
1) **[T1021.004] SSH**
(preferred by 80% of participants, observed[14] for 95%)
2) [T1021.002] SMB
3) [T1021.001] RDP
4) [T1021.006] WinRM
5) [T1021.005] VNC
6) [T1021.003] DCOM

Preferred techniques for *Credential Access* actions:
1) **[T1552] Unsecured Credentials (Bash History, Creds in Files)**
(preferred by 59% of participants, observed[15] for 81%)
2) [T1003] OS Creds Dumping (LSASS, /etc/shadow, SAM)
3) [T1110] Bruteforce (Guessing, Spraying)
4) [T1557] MITM capture / AiTM (ARP, LLMNR)
5) [T1056.001] Input Capture (Keylogging)

Preferred techniques for *Privilege Escalation* actions:
1) **[T1078] Valid Accounts (default, domain, local)**
(preferred by 68% of participants)
2) [T1548] Abuse Elevation Control Mechanism (Sudo, Bypass UAC)
3) [T1068] Vulnerability Exploitation for Privilege Escalation
4) [T1053] Scheduled Task/Job (At, Cron)
5) [T1055] Process Injection (ptrace, DLL, Proc Memory)

Preferred techniques for *Discovery* actions:
1) **[T1046] Network Service Scanning**
(preferred by 68% of participants, observed[16] for 100%)
2) [T1016] System Network Configuration Discovery (ip, arp)
3) [T1018] Remote System Discovery (/etc/hosts, references in files)
4) [T1049] System Network Connections Discovery (netstat, ss)
5) [T1040] Network Sniffing

VIII. Conclusion

This article contributes to the study of the *modus operandi* of actors conducting APT-like attack campaigns. The primary purpose of this work is the PWNJUTSU experiment in which 22 *Red Teamers* attacked their own instance of the same vulnerable infrastructure with the same attack objectives. Nevertheless, there is still no description language or model available to describe elements manipulated during this experiment formally. Therefore, the first contribution of this paper is a formal framework to express the evolution of the attacker's progression and their perception of a compromised network during their attack campaign.

[13] https://github.com/SigmaHQ/sigma

[14] any accepted connection (with password or key file) on SSH services.

[15] any event related to the `password_flag.txt` file on `vm2` or to the `.bash_history` file on `vm3`

[16] any attempt to reach (`SYN`) the TCP port 1723 which is among the 10 most common ports tried by the popular network scanner `nmap`. Moreover, this port makes no sense in our scenario.

More precisely, we have proposed to model an attacker through the state of their knowledge and their control over the compromised network. This control and knowledge evolve as the attacker progresses in the network. We proposed an operational semantics of the attack techniques described in the MITRE ATT&CK matrix to describe this evolution formally. These semantics allowed us to delineate one participant's attack campaign conducted during the PWNJUTSU experiment. In addition, these semantics are sufficiently generic to formalize other attack techniques and the description of other scenarios or real attack patterns. We believe that our formalization of ATT&CK techniques will, in the long term, contribute to the automation of some time-consuming processes that aim to fight against modern threats.

Furthermore, another significant contribution of this article is the PWNJUTSU dataset which consists of 16 million event logs and 172 GB of network traces of the PWNJUTSU experimentation. This dataset has been made public for the benefit of the scientific community. It can already be used to feed *Cyber Threat Intelligence* teams or to teach *Threat Hunting* from several perspectives.

Beyond the contributions of this paper, we believe that it will be engaging in the future to propose automatic methods for reconstructing attack campaigns by the exploitation of collected logs, based on the formalization proposed here. The PWNJUTSU dataset could also be used to compare different attack approaches implemented by APT-like actors who have the same operational objectives. For example, each of these attackers has different implementations of their procedures. For those who rely on native systems tools (*Living-Off-the-Land*), we think that it would be interesting to assess their uniqueness to be able to distinguish a hunted attacker from a legitimate user. The PWNJUTSU dataset should also make it possible to identify the attack tools or bruteforce approaches (*i.e.*, wordlists) used by the attackers. It would be interesting to characterize and attribute artifacts of these attack tools, which are also those used by *Advanced Persistent Threats*.

Acknowledgment

The authors would like to thank IRSN (*Institut de Radioprotection et de Sûreté Nucléaire*) for trusting them by supporting the PWNJUTSU project, but also for their active participation in its preparation. They are also grateful to YesWeHack for allowing them to work with high-level *Red Team* experts from their international community.

References

[1] M. Zheng, H. Robbins, Z. Chai, P. Thapa, and T. Moore, "Cybersecurity research datasets: Taxonomy and empirical analysis," in *11th USENIX Workshop on Cyber Security Experimentation and Test (CSET 18)*, 2018.

[2] C. Grajeda, F. Breitinger, and I. Baggili, "Availability of datasets for digital forensics – and what is missing," *Digital Investigation*, 2017.

[3] I. Baggili and F. Breitinger, "Data sources for advancing cyber forensics: What the social world has to offer," *AAAI Spring Symposium*, 2015.

[4] B. Stojanovic, K. Hofer-Schmitz, and U. Kleb, "APT datasets and attack modeling for automated detection methods: A review," *Computers & Security*, vol. 92, 2020.

[5] M. K. Daly, "The Advanced Persistent Threat." USENIX, 2009.

[6] I. Sharafaldin., A. Habibi Lashkari., and A. A. Ghorbani., "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP*, 2018.

[7] S. D. Bay, D. Kibler, M. J. Pazzani, and P. Smyth, "The UCI KDD Archive of Large Data Sets for Data Mining Research and Experimentation," *SIGKDD Explor. Newsl.*, 2000.

[8] A. Applebaum, D. Beck, M. Haase, and J. Baker, "Attack Flow – Beyond Atomic Behaviors," 2022. [Online]. Available: https://medium.com/mitre-engenuity/attack-flow-beyond-atomic-behaviors-c646675cc793

[9] D. Schlette, M. Caselli, and G. Pernul, "A comparative Study on Cyber Threat Intelligence: The Security Incident Response Perspective," *IEEE Communications Surveys & Tutorials*, 2021.

[10] B. Stojanović, K. Hofer-Schmitz, and U. Kleb, "APT datasets and attack modeling for automated detection methods: A review," *Computers & Security*, 2020.

[11] "APT Notes," 2022. [Online]. Available: https://github.com/aptnotes/

[12] G. Laurenza, L. Aniello, and R. Lazzeretti, "Malware triage based on static features and public APT reports," in *International Conference on Cyber Security Cryptography and Machine Learning*, 2017.

[13] M. J. M. Turcotte, A. D. Kent, and C. Hash, "Unified host and network data set," in *Data Science for Cyber-Security*, 2018.

[14] A. D. Kent, "Comprehensive, multi-source cyber-security events data set," 2015. [Online]. Available: https://www.osti.gov/biblio/1179829

[15] G. Ho, M. Dhiman, D. Akhawe, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner, "Hopper: Modeling and detecting lateral movement," in *30th USENIX Security Symposium*, 2021.

[16] S. Gianvecchio, C. Burkhalter, H. Lan, A. Sillers, and K. Smith, "Closing the Gap with APTs Through Semantic Clusters and Automated Cybergames," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 2019.

[17] R. Rodriguez and J. L. Rodriguez, "Security Datasets," 2021. [Online]. Available: https://securitydatasets.com/

[18] R. Arantes, "Operationally transparent cyber (OpTC)," 2021. [Online]. Available: https://dx.doi.org/10.21227/edq8-nk52

[19] M. M. Anjum, S. Iqbal, and B. Hamelin, "Analyzing the Usefulness of the DARPA OpTC Dataset in Cyber Threat Detection Research," in *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, 2021.

[20] S. Myneni, A. Chowdhary, A. Sabur, S. Sengupta, G. Agrawal, D. Huang, and M. Kang, "DAPT 2020 - constructing a benchmark dataset for advanced persistent threats," in *Deployable Machine Learning for Security Defense. MLHat 2020*, 2020.

[21] "MITRE ATT&CK Matrix for Enterprise," 2021. [Online]. Available: https://attack.mitre.org/matrices/enterprise/

[22] S. Choi, J.-H. Yun, and B.-G. Min, "Probabilistic attack sequence generation and execution based on MITRE ATT&CK for ICS datasets," in *Cyber Security Experimentation and Test Workshop - CSET*, 2021.

[23] A. Berady, V. Viet Triem Tong, G. Guette, C. Bidan, and G. Carat, "Modeling the Operational Phases of APT Campaigns," in *IEEE Conf. on Computational Science and Computational Intelligence*, 2019.

[24] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, *PDDL — The Planning Domain Definition Language*, Yale Center for Computational Vision and Control, 1998.

[25] S. Roy, N. Sharmin, J. C. Acosta, C. Kiekintveld, and A. Laszka, "Survey and taxonomy of adversarial reconnaissance techniques," 2021.

[26] F. Maymi, R. Bixler, R. Jones, and S. Lathrop, "Towards a definition of cyberspace tactics, techniques and procedures," in *2017 IEEE International Conference on Big Data*, 2017.

[27] A. Berady, M. Jaume, V. Viet Triem Tong, and G. Guette, "From TTP to IoC: Advanced Persistent Graphs for Threat Hunting," *IEEE Transactions on Network and Service Management*, 2021.

PWNJUTSU dataset
The dataset is available online at https://pwnjutsu.irisa.fr. It contains all participants' event logs (system and network), including the reference environment's event logs. A search engine is also provided to peek into the dataset.