




# A process mining-based method for attacker profiling using the MITRE ATT&CK taxonomy

Marcelo Rodríguez   [ Universidad de la República | [marcelor@fing.edu.uy](mailto:marcelor@fing.edu.uy) ]

Gustavo Betarte  [ Universidad de la República | [gustun@fing.edu.uy](mailto:gustun@fing.edu.uy) ]

Daniel Calegari  [ Universidad de la República | [dcalegar@fing.edu.uy](mailto:dcalegar@fing.edu.uy) ]

 Instituto de Computación, Facultad de Ingeniería, Universidad de la República. Julio Herrera y Reissig 565, 11300 Montevideo, Uruguay.

**Received:** 05 December 2023 • **Accepted:** 17 April 2024 • **Published:** 01 August 2024

**Abstract** Cybersecurity intelligence involves gathering and analyzing data to understand cyber adversaries' capabilities, intentions, and behaviors to establish adequate security measures. The MITRE ATT&CK framework is valuable for gaining insight into cyber threats since it details attacker tactics, techniques, and procedures. However, to fully understand an attacker's behavior, it is necessary to connect individual tactics. In this context, Process Mining (PM) can be used to analyze runtime events from information systems, thereby discovering causal relations between those events. This article presents a novel approach combining Process Mining with the MITRE ATT&CK framework to discover process models of different attack strategies. Our approach involves mapping low-level system events to corresponding event labels from the MITRE ATT&CK taxonomy, increasing the abstraction level for attacker profiling. We demonstrate the effectiveness of our approach using real datasets of human and automated (malware) behavior. This exploration helps to develop more efficient and adaptable security strategies to combat current cyber threats and provides valuable guidelines for future research.

**Keywords:** Cybersecurity, process mining, attacker behavior, threat intelligence, MITRE ATT&CK Framework

## 1 Introduction

According to the National Institute of Standards and Technology (NIST) [Center, 2015], any malicious activity that aims to collect, disrupt, deny, degrade, or destroy a system's resources or information is considered a cybersecurity attack. Response teams face immense challenges with the surging number of attacks and the growing complexity of cyber adversaries. Cybersecurity intelligence has emerged as a critical data analysis discipline to combat these challenges. It involves obtaining and analyzing data to identify, track, and predict cyber capabilities, intentions, and activities, aiding decision-making processes [Dehghantanha *et al.*, 2018].

Security experts can rely on monitoring and logging tools to investigate potential threats and generate alerts. By skillfully filtering, grouping, and combining audit logs, various security approaches can be implemented to safeguard systems. According to NIST, "threat modeling" is a technique that effectively captures the functioning of a system to identify and comprehend potential threats, along with the objectives and tactics of threat agents. By doing so, security controls can be established to mitigate potential issues. According to Messe [Messe *et al.*, 2020], models can be categorized into two types: those focusing on assets and those focusing on the attacker. Asset-focused models prioritize risk levels based on the sensitivity of the data and its value to potential attackers. Meanwhile, attacker-focused models create profiles of attackers based on their characteristics and skill set to define and implement an appropriate mitigation strategy based on specific exploits that an attacker may execute.

The MITRE ATT&CK framework [Strom *et al.*, 2018] has

become an essential asset in the battle against cyber-attacks. It provides cybersecurity intelligence by offering insight into the various threats, including detailed information on the tactics, techniques, and procedures employed by attackers during the attack process. These attackers can be human or automated (malware). In the second case, the MITRE ATT&CK provides detailed descriptions of various types of malware. These descriptions cover the use of techniques that have been publicly reported, e.g., for the WannaCry ransomware<sup>1</sup>. Ransomware permanently blocks access to the victim's data unless a ransom is paid.

Although individual tactics, techniques, and procedures are of utmost importance to identify an attack, they must be connected to understand an attacker's behavior fully. It could be essential for more structured attackers such as malware. In this context, Process Mining (PM) techniques [van der Aalst, 2016] could be valuable. PM allows analyzing the event logs associated with executing a system's processes, being a process of a set of coordinated tasks to achieve an objective. Process discovery techniques support building (process) models that best describe the behavior inferred from the event logs. Many discovery algorithms, like the *Inductive Miner* [Lee-mans *et al.*, 2014], can cope with infrequent behavior and large event logs. Several tools, e.g., ProM [van Dongen *et al.*, 2005], provide automated support to perform PM-based analysis of systems behavior.

<sup>1</sup>WannaCry tactics and techniques:  
<https://attack.mitre.org/software/S0366/>

In a previous work [Rodríguez *et al.*, 2021], we examined the behavior of automatic cyber attackers, specifically the WannaCry ransomware, using PM techniques. Our analysis focused on low-level system process events. It uses the ATT&CK Framework as a reference guide, not in the context of a systematic method.

Our current research proposes a four-stage method that employs PM to uncover process models of observed attack strategies. We view these processes as an attacker's actions to compromise a specific target. Our process discovery techniques enable us to identify process models that describe the behavior inferred from event logs. To increase the level of abstraction for attacker profiling, we use the MITRE ATT&CK framework to semantically lift events [Azzini *et al.*, 2013], associating low-level system process events with suitable semantic objects, in this case, tactics.

We presented the method in [Rodríguez *et al.*, 2023] and evaluated its effectiveness using human attackers' information from the PWNJUTSU experiment [Berady *et al.*, 2022]. The present paper constitutes a substantially extended and thoroughly revised version of [Rodríguez *et al.*, 2023] by contributing the following:

1. a comparison between the behavioral model manually described for a human attacker in the PWNJUTSU experiment and the automatically discovered models we obtain (Section 4.4.3);
2. a further evaluation of our method using automated attackers' information based on the WannaCry ransomware (Section 5), also comparing existing behavioral models with the ones we automatically discover;
3. a discussion comparing the method's application to human and automated attacker profiling (Section 7).

The rest of the paper is organized as follows. Section 2 presents some background on PM and the MITRE ATT&CK framework. Section 3 describes our proposed approach for discovering attacker profiles. In Section 4, we present the results of performing a non-trivial experiment using human attackers' information. In Section 5, we present the results of performing a second experiment using automated attackers' information. We present related work in Section 6. In Section 7, we discuss many issues concerning the method and the experiments. Finally, in Section 8, we provide some conclusions and describe future work.

## 2 Background

In what follows, we provide some background on PM and the MITRE ATT&CK framework.

### 2.1 Process mining

Process Mining (PM) [van der Aalst, 2016] applies to a wide range of domains, offering a fact-based vision derived from actual data that helps audit, analyze, and improve existing business processes in those domains.

It allows for analyzing the records (logs) of events associated with executing processes in information systems. Event logs, which should not be confused with application log files,

are organized according to the following principles [van der Aalst, 2016]:

- A process comprises cases (traces) representing a process instance execution from start to finish. Each case has a case ID, allowing one to identify a case among others uniquely.
- A case comprises events, i.e., an action performed in the process, associated with precisely one case.
- Events can have activity, time, cost, and resource attributes. At least one attribute must be present that represents the activity carried out, i.e., its activity name.
- Events have a timestamp, which determines a partial order between events of the same case.

There are three types of PM: discovery, conformance, and enhancement. Discovery uses event logs to create a process model (visual representation) from the causal dependencies between events. Conformance implies verifying the correspondence of the enacted business processes concerning the expected one (through a reference model). Finally, enhancement uses additional information to improve the one provided by the process model, obtaining measures such as the duration of the processes, bottlenecks, or the underuse of resources, among others. Many supporting tools, such as ProM [van Dongen *et al.*, 2005], are freely accessible and provide automated support to perform analysis of systems behavior.

In this work, we perform a data-driven exploration of data focused on discovery and conformance checking. Many discovery algorithms exist, such as the Inductive Miner [Lee-mans *et al.*, 2013] that can cope with infrequent behavior and large event logs. Its main characteristics lie in being one of the few mining algorithms that guarantee anomaly-free (e.g., deadlocks, loops) or sound process models. It is also characterized by generating models with reliable fitness and precision measurements (the two principal quality dimensions for assessing the behavior allowed by a discovered model). Experimental studies showed that the results obtained using Inductive Miner were the most suitable for detecting cyber attacks [Myers *et al.*, 2017; Konsta *et al.*, 2023].

### 2.2 MITRE ATT&CK

The MITRE ATT&CK framework, as proposed by [Strom *et al.*, 2018], provides a comprehensive classification system for attackers' behavior during an attack. In particular, the framework puts forward a taxonomy to describe the behavior of attackers throughout the lifecycle of an attack. It is structured around three key concepts: tactics, techniques, and procedures (TTP), all grounded in real-world observations. The framework is used for multiple purposes, including reactive threat investigation and proactive control evaluation.

In the current version (v13), there are three separate matrices: Enterprise (attacks against IT networks and enterprise cloud), Mobile (attacks targeting mobile devices), and Industrial Control Systems (ICS) (attacks targeting ICS). Although our method is generic, we consider the Enterprise matrix for the application example. Its taxonomy is divided into the 14 tactics depicted in Table 1, each of which includes a subset of more specific techniques (196) and sub-techniques (411) or concrete procedures.

A tactic refers to *what* an attacker does to achieve a goal (discovery, initial access, persistence). A technique is a way of carrying out an activity, representing *how* an adversary achieves a tactical objective by acting. The techniques are individual or discrete actions; tactics are the way of combining those actions. A procedure refers to a series of well-defined steps, a particular instance of using a specific technique, and describes how an adversary implements that technique. For example, to carry out a discovery tactic (TA0007, whose goal is to gain knowledge about the target), an attacker could gather information from exposed systems using network service discovery techniques (T1046). These techniques assist adversaries in gaining knowledge of the network environment and positioning themselves before determining their next course of action. A procedure for carrying out a network discovery could describe the necessary steps to execute the *Nmap* tool [Lyon, 2023]. In the CostaRicto campaign (D: C0004), specified by Mitre Attack, threat actors employ *Nmap* to scan target environments.

Table 1. ATT&CK Enterprise tactics

ID	Name	Description
TA0043	Reconnaissance	Gather information
TA0042	Resource Development	To support operations
TA0001	Initial Access	Get into your network
TA0002	Execution	Run malicious code
TA0003	Persistence	Maintain their foothold
TA0004	Privilege Escalation	Higher-level permissions
TA0005	Defense Evasion	Avoid being detected.
TA0006	Credential Access	Steal account data
TA0007	Discovery	Get environment
TA0008	Lateral Movement	Move in environment
TA0009	Collection	Gather data of their goal
TA0011	Command and Control	Communication
TA0010	Exfiltration	Steal business data
TA0040	Impact	Manipulate or destroy

### 3 A method for attacker profiling

Our approach to discovering attacker profiles using PM and the MITRE ATT&CK taxonomy consists of four steps that follow the standard PM methodology, as shown in Figure 1.

The first step, **Enactment**, involves executing attack strategies within the targeted software system. Next, in the **Extraction** step, data is extracted, integrated, and loaded to create event logs, which serve as the primary source of information for analysis. These logs are classified and labeled using the MITRE ATT&CK taxonomy. The **Discovery** step focuses on discovering a process model that represents the attackers' behavior through process discovery algorithms. Finally, the **Analysis** step interprets and evaluates the discovery results. Expert analysis is used to identify patterns based on the attackers' behavior. The model can reveal various aspects, including techniques that consume more time for the attacker or activities on the critical path.

Next, we provide more detail on extracting, discovering, and analyzing information.

### 3.1 Extraction

This process involves gathering and arranging events, categorizing them according to the MITRE ATT&CK classification system, and constructing the event log.

#### 3.1.1 Event collection and organization

For effective event collection and organization, it is recommended to centralize the information by sending it to an SIEM (Security Information and Event Management) system. This approach offers a complete situation overview and simplifies extracting and correlating events. In a previous article [Rodríguez et al., 2021], we proposed a logging infrastructure based on Elastic Stack [Elastic, 2023], which proved highly effective. As another option, in [Rodríguez et al., 2023], we used an existing dataset that streamlines the event collection process.

Our work aims to uncover and model tactical information about attackers, especially their *modus operandi*. Therefore, the log entries must be part of an activity carried out in the context of an attack. The entries corresponding to benign behavior should be minimal; ideally, all traces should pertain to attacker activity. This point necessitates isolating the environments and allowing access only to attackers. In the realm of cybersecurity, such scenarios are expected to be encountered. Honeypots [Spitzner, 2002], for instance, are tools that have long been utilized to divert and learn from malicious activities in an infrastructure or information system.

To effectively identify attacker profiles when they compromise a host, network, or application, it is essential to group events that accurately describe their behavior from different viewpoints. For instance, entries from the same IP address can be grouped to showcase the attacker's perspective, while entries with the same destination IP address can represent the target's perspective.

Although it is assumed that everything recorded in the log is part of an attack, not everything done by the attacker has the same relevance for the model. For example, once a host has been compromised, the attacker may explore directories for information, generating specific behavior dependent on the compromised host. Similarly, events that are executed autonomously by the system to ensure its proper functioning may not be significant in identifying the behavior of an attacker. In contrast, the mechanism used to exploit a particular vulnerability, which leads to the compromise of the host, may be very relevant.

The attributes of every event included in the entry must be considered to construct the attacker model. Selecting features that provide relevant information is crucial to uncover strategies. Specialized tools are necessary to gather this type of data. In this work, we use System Monitor (Sysmon for short) [Microsoft, 2023], a service developed by Microsoft's Sysinternals that operates at the driver level within the operating system. With Sysmon, we can obtain pertinent information about activity on a Windows system, such as system process creation or termination, network connections, changes in the Windows registry, and file creation timestamps. It is important to note that Sysmon is not a security tool and does not analyze the events it generates. Its sole purpose is to

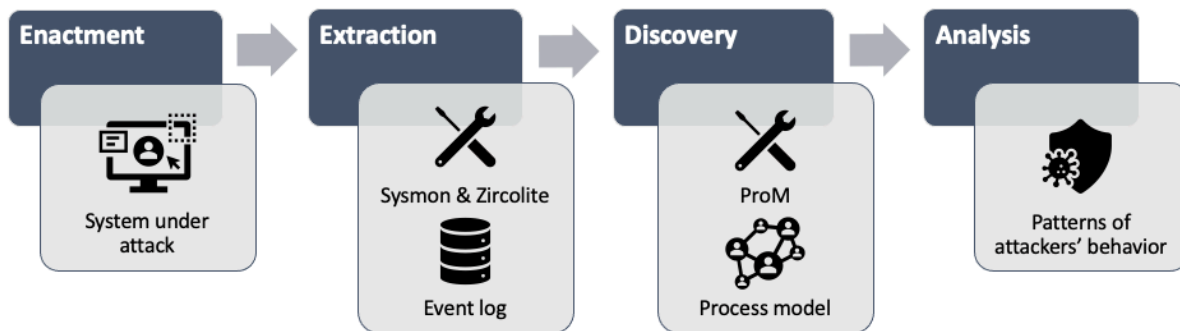


Figure 1. Proposed method

record system activity as long as it is installed and configured accordingly. A configuration file is required to determine which events to collect.

Various projects aim to develop Sysmon configuration templates prioritizing security, including SwiftOnSecurity [M., 2021]. These templates prove helpful in identifying significant events related to attacker behavior, enabling the user to define exceptions and avoid generating irrelevant events. Utilizing this tool and its configurations makes it easier to analyze the activities of automated attackers, as we have demonstrated in our previous work [Rodríguez *et al.*, 2021].

### 3.1.2 Labeling of events

Labels associated with events triggered by low-level system processes often make it challenging to grasp the process discovered from those events correctly. We perform a “semantic lifting” of events [Azzini *et al.*, 2013], associating to relevant events labels from the MITRE ATT&CK taxonomy. It allows for gaining better knowledge of some properties of the overall process.

The classification process can be carried out in various ways, from manual intervention by an expert to an automated process aided by tools. In our case, we utilize **Zircolite** [Wagabat, 2023]. This tool detects anomalies in operating system logs, both Linux and Windows, as it allows us to identify events that result from malicious activities. We use Zircolite with a set of rules from the SIGMA project [Roth and Patzke, 2022], an open, generic, and technology-independent metalanguage designed for creating rules that can detect significant events in a set of log traces. The project aims to enhance the capabilities of detecting specific events by supporting the definition, sharing, and collection of rules. Additionally, the project offers converters that translate rules into query languages specific to different types of SIEM. Consequently, a query generated from a Sigma rule can be entered into the SIEM search engine to verify the occurrence of defined events.

In Listing 1, we present a summary of a SIGMA rule; that rule includes the authors, title, description, and identifier, as well as a tags field that associates the rule with a MITRE ATT&CK tactic or technique as proposed by the author of the rule. The rule definition consists of conditions that must be met for the rule to take effect. In the example, the rule verifies that the event was generated by **Sysmon** using the **Channel** and **EventID** fields and then verifies the context in which the **whoami** command is executed.

Zircolite is a versatile tool that can process input files in several formats, such as MS Windows EVT-X format (EVT-X, XML, and JSON), Auditd logs, Sysmon for Linux, and EVT-Xtract logs. When given a set of events and a set of rules, Zircolite iterates through the event set and generates a new output event if the conditions of a rule are met. The output format can be customized using Zircolite’s templates.

The output of Zircolite is structured into two main sections: rule information and information for each event that matches that rule. The first section lists information such as the rule ID and name, the tag related to the tactic or technique of MITRE ATT&CK, and the event’s impact on the system. The number of events that match that rule is recorded, and for each match, execution environment data and specific detection data are registered.

Listing 1: Rule sigma - Whoami

```

title: Run Whoami as SYSTEM
status: experimental
author: <name>
id: 80167ada-7a12-41ed-b8e9-aa47195c66a1
description: Detects a whoami.exe
description: This may be a sign of successful
description: local privilege escalation
tags: [attack.privilege_escalation]
level: high
rule_level: [
  "SELECT * FROM logs WHERE ((EventID = '1' AND
    Channel = 'Microsoft-Windows-Sysmon/
    Operational') AND (User LIKE '%AUTHORI%'
    ESCAPE '\\') OR User LIKE '%AUTORI%'
    ESCAPE '\\') AND (OriginalFileName = '
    whoami.exe' OR Image LIKE '%\\whoami.
    exe' ESCAPE '\\'))"
]

```

Then, the number of events that match that rule is tracked, and for each match, execution environment data and specific detection data are recorded. This information is obtained from the original event detected by Sysmon. Some examples of environment data are EventID, Computer, User, and Timestamp. The specific detection data depends on the type of event, and it may vary accordingly.

### 3.1.3 Building the event log

To create a suitable event log for PM, we utilize Zircolite’s exporting capabilities to convert the labeled log entries from the previous step. The log is exported as a CSV file, each row representing an event with columns containing event attributes, including the case ID, activity name, and timestamp.

Identifying the activity associated with each event is crucial, as it represents the nodes in the attack model that explain

the steps executed in the attack flow. It helps visualize and identify the sequences and dependencies of the attacks in the model. As per the labeling process, the attacker's techniques and tactics are considered the activities performed since they represent the attacker's intent to compromise the target. The timestamp of the event indicates when it occurred and the order of events, and we use Sysmon's timestamp as the most direct option.

Furthermore, organizing and grouping events within a case is necessary, and this information must be defined as part of the values recorded in the Zircolite output. We use the red team name as the case ID for each process instance we attempt to discover.

### 3.2 Discovery

The event log obtained from the earlier extraction process is utilized to discover a process model. Process discovery techniques take this event log and generate a process model based solely on the observed events without additional information. The discovered model is then represented using process modeling languages like Petri-net, BPMN, or process-tree.

Before selecting the best-suited discovery algorithm, it is crucial to understand the peculiarities of the domain. In our domain, cases may include repeated activities in sequence or duplicate tasks with the same taxonomy signature. Moreover, when the attacker is an automated process like a bot, the time between tasks can be milliseconds, creating a false perception of concurrent activities if the timestamp is not precise enough. Furthermore, the data may contain noise, incomplete or undefined start and end tasks, and hidden activities, leading to an inadequate model that does not accurately represent reality.

In this work, we use ProM [van Dongen *et al.*, 2005], which is freely accessible and provides discovery algorithms and specialized monitoring panels for analysis. Specifically, we utilize the Inductive Miner algorithm.

Multiple filters can be applied to refine the process model, removing incomplete cases, outliers, and more. An iterative approach filters partial behavior unsuitable for a specific study, such as selecting process variants for deeper analysis.

In PM, the measures of fitness and precision [van der Aalst, 2016] are used to evaluate how well a set of traces fits a model. The goal of fitness analysis is to evaluate whether the behavior in the log is captured in the model, e.g., we have 100% fitness if all log traces match the model. In practice, the process model may show more or less behavior than recorded in a particular event log. The analysis and detection of these different behaviors is called precision. It means that if the model only considers the behavior observed in the particular event log, the precision is 100%. In this way, precision can help discover alternative branches never used when executing a particular process instance. We use these metrics and variant analysis to analyze specific behaviors described by groups of traces.

In this work, we use the "Visualize deviations on Process tree" PromM plugin, which performs conformance checking concerning a set of traces and highlights the paths taken by such traces to visually compare behavior and calculate fitness and precision metrics.

### 3.3 Analysis

Once a model is discovered, it is imperative to interpret and analyze it to determine if it contains evidence of an attacker's behavior and, if possible, characterize such behavior. This step is strongly attached to Discovery since findings could require iterative refinements of the discovered models for further analysis.

An identification and interpretation of patterns is crucial in this analysis. In this sense, we can use specific statistical metrics, analyze the results with the help of experts in the problem domain, and even compare the findings with previous results if available. Further research is required to understand to what extent it could be possible to build automated assistance tools.

We divide the analysis phase into two: data analysis and model analysis. In this context, the ProM tool provides certain statistical metrics and the discovered models that facilitate such evaluations.

During data analysis, we statistically evaluate the results derived during the model discovery phase. Through expert analysis, we compare the tactics included in the generated model with the low-level data captured in the logs of each experiment. It is important to remember that the tactics included in the model are derived from the labeling carried out during the extraction step of the proposed methodology, according to the MITRE taxonomy.

During model analysis, we focus on identifying and interpreting the patterns that emerge from models. Again, we rely on expert analysis. When previous results exist, e.g., there is an expected behavior of an attacker (as with the PWNJUTSU experiment) or even an existent flow model (as with the WannaCry ransomware), it is possible to go deeper in the analysis and validate if the discovered models present such attacker behaviors by comparing both models. In this work, we added a section related to model analysis describing such a deeper analysis.

## 4 Profiling human attackers: the PWNJUTSU experiment

To assess the effectiveness of the suggested approach, we use the unprocessed monitoring events produced by human attackers in the PWNJUTSU experiment [Berady *et al.*, 2022], a publicly available dataset used for monitoring attack campaigns. In PM, extracting and preparing event logs can be complex. It requires identifying relevant data and using timestamps to maintain order in activities. System logs are not designed for this purpose. PWNJUTSU is a set of 16 million events where 22 red teams perform independent activities on vulnerable machines, recording each event with its corresponding timestamp. In this sense, each red team can be considered a separate case of the process.

We will now outline the implementation of the method by following the four steps elaborated in Section 3.



## 4.1 Step 1: Enactment

The PWNJUTSU dataset was collected through sensors installed in an infrastructure purposely created for attack testing. The research team, which owns the infrastructure, later reported the experiment's results. The dataset logs the activities of the 22 professional hacker participants during the investigation.

In this experiment, participants were required to perform intrusive tasks to achieve the agreed-upon objective with the infrastructure owners. Each participant was provided with an independent instance of identical infrastructure. After the experiment was completed, the research team compiled the raw dataset and publicly shared it.

To accurately observe the actions of participants, monitoring devices were installed to focus on the network and systems. For the Windows machine, Sysmon was configured based on SwiftOnSecurity, and Windows auditing was set up to collect logs of systems, applications, and security events. The Snoopy tool is installed in the Linux machines to log all program executions and their command lines in auth.log. Additionally, the Linux *auditd* component was configured according to security best practices to monitor all requests received by Apache2 (access.log and error.log) and all queries received by MySQL and observe SSH accesses. Network traffic was closely monitored for all activity traces, logging each machine's incoming and outgoing traffic. A packet with network data (PCAP file) was generated for each machine involved in the scenario. The system and network logs were centrally forwarded and indexed in a Splunk SIEM.

## 4.2 Step 2: Extraction

Of the three extraction activities, event collection and organization, labeling of events, and building of the event log, the first one was done by the PWNJUTSU experiment.

### 4.2.1 Event collection and organization

In the PWNJUTSU experiment, three hosts were used: n\*-vm1 with Linux Ubuntu 14.04, n\*-vm2 with Windows 2008, and n\*-vm3 with Linux Ubuntu 20.04 (where "\*" corresponds to Red Team number). The system logs of these vulnerable machines were collected in JSON format files, resulting in more than 16 million event records, of which n\*-vm1 contributed 9.2 million events, n\*-vm2 contributed 50,000 events, and n\*-vm3 contributed 7.2 million events.

The Windows host (n\*-vm2) was installed with Windows 2008 and configured with Sysmon using the SwiftOnSecurity security template, which is particularly relevant for this research. In addition to recording events with Sysmon, this host meets the premise that all entries in the log are part of some activity carried out in the context of an attack.

The dataset provides event clustering (grouping). A file containing system logs from all infrastructure hosts is supplied for each participant (Red Team). In our experiment, we only work with events from Windows hosts (n\*-vm2) in the dataset.

### 4.2.2 Labeling of events

The events from Windows machines of the PWNJUTSU experiment were transformed into a JSONL format compatible with Zircolite. After that, we run Zircolite to process and tag the events. An example of its output, in pseudo-JSON format, is displayed in Listing 2.

This log was generated by an event logged in the n35-vm2 host (used by Team 35). The output is divided into two sections: rule information and event-data information. The rule information section contains the corresponding data for the rule, as shown in Listing 1, and also keeps track of the number of events that matched the rule (attribute: count). The event details are added to the corresponding section for each event that matches the rule. In this example, we look at the illustrative event data in the 'matches' section, which involves the execution of the 'whoami' command initiated by the 'cmd.exe' process and performed by the system user (NT AUTHORITY LOCAL SERVICE). While the 'whoami' command is not malicious, its execution in this context raises suspicion. In general terms, the input that generates this output from Zircolite refers to the event data of the event.

Listing 2: Output Zircolite

```

title: Run Whoami as SYSTEM
status: experimental
author: <name>
id: 80167ada-7a12-41ed-b8e9-aa47195c66a1
description: Detects a whoami.exe
description: This may be a sign of successful
description: local privilege escalation
tags: [attack.privilege_escalation]
rule_level: high
rule: [
  "SELECT * FROM logs WHERE ((EventID = '1' AND
  Channel = 'Microsoft-Windows-Sysmon/
  Operational') AND (User LIKE '%AUTHORI%'
  ESCAPE '\\' OR User LIKE '%AUTORI%' ESCAPE
  '\\') AND (OriginalFileName = 'whoami.exe'
  OR Image LIKE '%\\\\whoami.exe' ESCAPE
  '\\'))"
]
count: 1
matches: [
{
  row_id: 30
  Keywords: None
  EventID: 1
  OpCode: Info
  Computer: n35-vm2
  EventRecordID: 10956
  Channel: Microsoft-Windows-Sysmon/Operational
  Description: whoami-logged on user information
  User: NT AUTHORITY\LOCAL SERVICE
  LogonId: 0x3e5
  Product: Microsoft Windows Operating System
  Type: Information
  ParentCommandLine: cmd.exe /c \"whoami\"
  UtcTime: 2021-06-09 05:56:08.830
  ParentProcessId: 4956
  ProcessId: 4204
  OriginalFileName: whoami.exe
  CommandLine: whoami
  CurrentDirectory: C:\wamp\www\uploads\
  text: Process Create
  SidType: 0
  Company: Microsoft Corporation
  TaskCategory: Process Create
  ParentImage: "C:\Windows\System32\cmd.exe"
  Image: "C:\Windows\System32\whoami.exe"
  IntegrityLevel: System
  EventType: 4
  UserID: S-1-5-18
  OriginalLogfile: n35.json
}
]

```

When tagging the events, we noticed that for some Zircolite rules, the author did not suggest tags for MITRE ATT&CK-related tactics. It caused some labels to be left blank in the event labeling step when using the proposed methodology, and the behavior model was inadequately defined according to MITRE's taxonomy. Nevertheless, the event was identified as malicious behavior because it met the conditions of a rule, even though that rule was associated with a tactic. When encountering rules without a proper tag, we manually labeled them based on the rule conditions and our experience.

### 4.2.3 Building the event log

We built a CSV file using a specific output template for Zircolite. Table 2 illustrates the basic information included in our event log.

**Table 2.** Fragment of example event log

Case ID	Time	Activity
n11-vm2	2021-05-10-09:00:31	persistence
n11-vm2	2021-05-10-11:12:18	privilege_escalation
n12-vm2	2021-05-10-00:30:00	defense_evasion
n21-vm2	2021-05-15-00:15:52	execution

As mentioned, the activity of the process is determined by the tactics carried out by the attacker. The order of the event sequence is based on the Sysmon timestamp. The process instance or Case ID refers to a specific instance of the process. In this study, we are interested in observing the security attack process by the Red Teams on Windows hosts (n\*-vm2). Therefore, the security events registered in each host instance refer to a particular case, and the CaseID will be a hostname. Thus, by having several instances of the same host (one for each Red Team), the PM tool can compare multiple executions of the process, one for each case.

The log also contains other fields of interest, e.g., Timestamp, ComputerName, RuleID, User, and RuleTag. Listing 2 presents a detailed overview of fields in the event-data section. Although unused in this work, the EventType and TaskCategory fields are of interest since Sysmon uses them to indicate actions related to the registered event, such as ProcessCreate, FileCreate, RegistryEvent, NetworkConnect.

After the labeling step, 22 process instances were obtained. Each corresponds to an attack by a Red Team participant in the experiment. Of the 22 process instances, 21 were used to discover a process model based on the 7265 events extracted. Since the activity of team 12 was analyzed in detail in [Berady *et al.*, 2022], we decided to exclude the events corresponding to their activity when discovering the model and to perform an individual deviation analysis of that activity. Deviation analysis refers to the fact that the execution of a process does not conform to the normative model of the process [Depaire *et al.*, 2013].

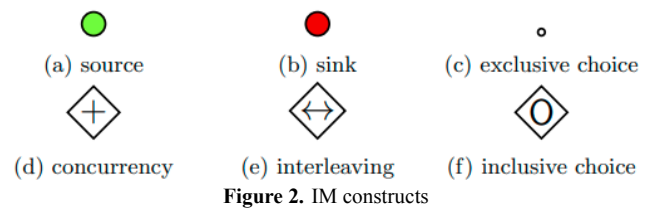
The set of activities (class) shown in Table 3 refers to all actions performed by 21 participants in the experiment (not including team 12) and mapped to MITRE ATT&CK. In other words, during the attack, it is determined that the behavior of these 21 attackers is represented by the tactics expressed in the Class column of the log summary. Additionally, the

frequency with which these tactics occur is provided in the information contained in the analyzed log.

On the other hand, the activities detected in the individual behavior of Team 12 are detailed in Table 5. It is noteworthy that Table 5 contains a subset of the activities (classes) present in Table 3, indicating that the behavior of Team 12 is included in the behavior observed by the rest of the 21 participating teams.

### 4.3 Step 3: Discovery

The Inductive Miner algorithm operates internally on process trees, hierarchical structures composed of multiple nodes with children. The leaves represent individual activities, while non-leaf nodes are operators that determine how their children are combined. In ProM, the Inductive Miner (IM) plugin uses the constructors shown in Figure 2.



**Figure 2.** IM constructs

Figure 3 shows the discovered process model that describes the actions of 21 attack teams out of 7,265 events organized into 12 activities. The IMf algorithm (a variant of the original algorithm that filters infrequent behaviors using a particular heuristic) was used to generate this model, which allows for noise filtering by removing infrequent routes and highlighting the most common ones. The model uses start, end, activity nodes, directed edges, and operators that split and join edges into branches.

The ProM plugin uses path-slider control to determine the level of noise filtering. This control can be adjusted to enable unrestricted filtering or complete filtering. After experimenting with different settings to get a better view, this study applied a noise filter of 0.3 by setting the path slider to 0.7.

We use the “Visualize deviations on Process tree” plugin for performing conformance checking and highlighting the paths a group of traces takes for visually comparing behavior. Figure 4 depicts (colored) the behavior of Team 12 concerning the process discovered from the remaining teams (given in Figure 3).

We got a 95% fitness and 40% precision for the traces of Team 12, concerning the model created by the traces of the other 21 red teams (Figure 3). The fitness of 95% means that the model covers practically all the behavior performed by Team 12 during the attack. On the other hand, given the 40% precision, the model is also helpful in quantifying the degree of difference between the tactics used by different attackers (in the experiment, red teams). A precision value of 40% is helpful because it can indicate several things. For example, if the attacker achieved his goal, his attack was more optimal if we consider the number of tactics used to measure optimization. Using fewer tactics may allow the attacker to evade detection mechanisms. On the other hand, if the attacker failed to complete the task, this could be a measure that helps identify the tactics the attacker could have used to

achieve a successful attack. Of course, this analysis depends on determining whether the attacker could complete the attack successfully.

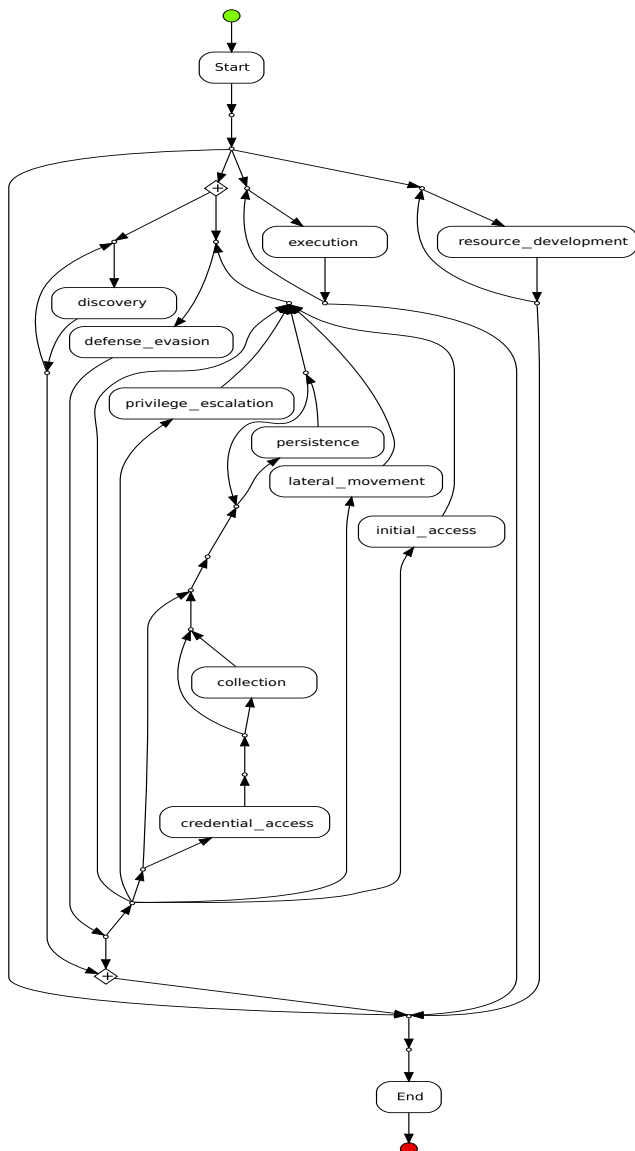


Figure 3. PWNJUTSU Red Team's-Inductive Miner (ProM)

#### 4.4 Step 4: Analysis

With PM tools, it is possible to perform data analysis of the generated tagged record and model analysis of the discovered process models. These two types of analysis are presented in what follows.

##### 4.4.1 Data analysis

Table 3 summarizes the information contained in the labeled event log. Almost 79.88% of the activity is concentrated in 3 types of events: **execution**, **defense\_evasion**, **persistence**. Our approach generates a high-level tactics-based model which summarizes the behavior of an attacker. The main goal is to map the activities of the host logs that correspond to the attacker's malicious activities. How tactic iden-

tification contributes to a better understanding and generalization of attacker behavior is discussed in Section 7.

The execution tactic [MITRE, 2023] involves using techniques and procedures to enable an adversary to execute malicious code on a local or remote system. These techniques are often combined with methods from other tactics to achieve more complex objectives, such as network exploration or stealing data. For instance, an adversary can use a tool from a compromised system to perform remote system discovery and explore new targets. In the PWNJUTSU dataset, most cases are related to running tools like `net.exe`, `powershell.exe`, and `ping.exe`. The Windows tool `net.exe` allows for configuration and system adjustments through a command prompt or batch files, while `ping.exe` is used for diagnosing and discovering network systems. The tool `Powershell.exe` executes malicious code; the Red Team 32 extensively uses this tool to establish network connections with other hosts in the infrastructure.

A defense evasion tactic [MITRE, 2023] is used by attackers to evade detection during a compromise. These tactics may involve disabling or uninstalling security software or obfuscating data and scripts. When analyzing the dataset, anomalies in executing system processes from unexpected locations are the main factor in detecting these tactics.

Finally, the persistence tactic [MITRE, 2023] refers to the set of techniques that adversaries use to maintain access to systems, even after reboots or changes of credentials. These techniques include installing backdoors, creating scheduled tasks, and modifying configuration files. We observed that these tactics are detected through unusual processes accessing `desktop.ini`. Adversaries can use this to alter how Windows File Explorer displays the contents of a folder. We also identified techniques related to the creation of local users.

Table 3. Log Summary (ProM)

#### Log Summary

Total number of process instances: 21

Total number of events: 7265

#### Event Name

Event classes defined by Activity

#### All events

Total number of classes: 14

class	Occur. (abs.)	Occur. (rel.)
execution	3270	45.01%
defense_evasion	1627	22.39%
persistence	907	12.48%
resource_development	805	11.08%
discovery	441	6.07%
privilege_escalation	64	0.88%
lateral_movement	47	0.65%
initial_access	25	0.34%
Start	21	0.29%
End	21	0.29%
command_and_control	19	0.26%
impact	10	0.14%
credential_access	6	0.08%
collection	2	0.03%

Although the detection of each of these tactics individually may be a warning signal, identifying an ordered sequence of



these three techniques indicates successful malicious behavior. In a typical attack, an adversary seeks to compromise a device to use it as an entry point into the network. It can be achieved by exploiting a public vulnerability or using compromised credentials, i.e., through the execution tactic. Once inside the network, the goal is to avoid being detected while malicious activities are carried out, for example, by applying defense evasion tactics. In all cases, maintaining access to the compromised device is crucial (persistence tactic). In short, detecting a process with this sequence of events indicates that the attacker has managed to obtain initial access, attempted to hide, and seeks to persist in maintaining access. Additionally, adversaries often use other tactics in a successful attack, such as privilege escalation (privilege escalation tactic) or lateral movement through the network (lateral movement tactic). These tactics were observed, albeit to a lesser extent, as shown in Table 3.

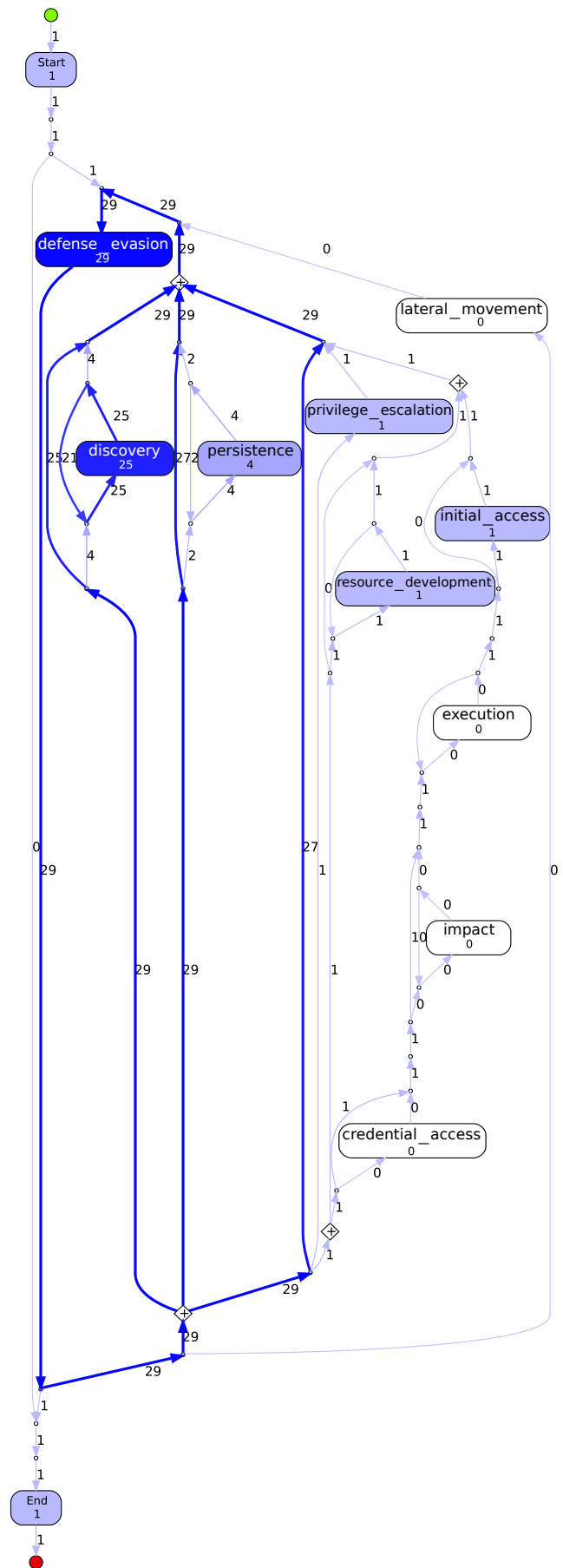
**Table 4.** Cisco: Critical severity IoCs (from [Nahorney, 2020])

MITRE ATT&CK Tactic	% of IoCs seen	% point change
Execution	55	+14
Defense Evasion	45	-12
Persistence	38	+27
Lateral Movement	22	+18
Credential Access	21	+17
Command and Control	8	-3
Impact	7	+6
Collection	5	+4
Discovery	0.4	-2
Privilege Escalation	0.3	-7

Our results align with the experiments by [Nahorney, 2020] studying data from *Cisco’s Endpoint Security*. Based on Indicators of Compromise (IoCs)<sup>2</sup> analysis, it shows that code execution, defense evasion, and persistence are the main tactics used in critical attacks (Table 4).

**4.4.2 Model analysis**

Initially, in Figure 3, we note that attackers choose one of three mutually exclusive lines of action: a) resources\_development, b) execution, or c) carrying out a set of actions concurrently. The concurrent actions involve c.1) discovery, c.2) defense\_evasion, and c.3) choosing one of the following tactics exclusively: initial\_access, lateral\_movement, privilege\_escalation, or persistence. In this step, a flow can also be followed where tactics of credential\_access or collection are applied before persistence. It can also be seen that the model includes tactics that can be used in a loop, such as discovery, execution, persistence, defense\_evasion, etc. This model reflects behavior consistent with the intuition that attackers usually try to identify their target first (information gathering), evade defenses, and achieve initial access. The model reflects this behavior based on the tactics of discovery, defense\_evasion, and initial\_access, respectively. In this path, some of the tactics used may be in the form of a loop, which reflects that they are activities in which the attacker, through trial and error, makes several attempts on one or more targets. Once they



**Figure 4.** PWNJUTSU Team 12 - Deviations (ProM)

<sup>2</sup>IoC: An Indicator of Compromise is a piece of information that indicates a potential security breach or cyberattack.

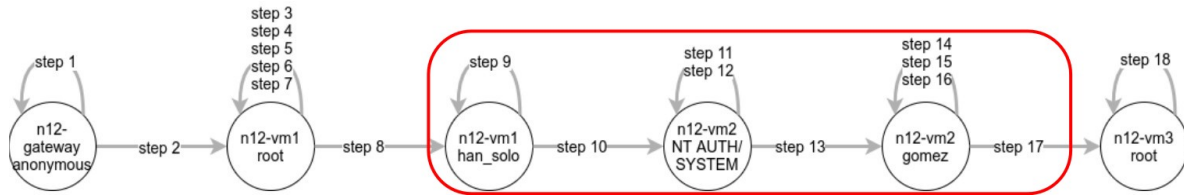


Figure 5. Propagation space of Team 12 during its campaign during the PWNJUTSU experiment

have collected information on the target or gained initial access, attackers may attempt to execute actions such as execution, privilege escalation, lateral movement, or persistence in the system. Each step of the model indicates that, depending on the attackers’ choice, each path leads to a different attack flow, and in turn, a new flow may start that includes the same activities or tactics.

Table 5. Log Summary (Team 12)

Log Summary - Team 12		
Total number of process instances: 1		
Total number of events: 70		
<b>Event Name</b>		
Event classes defined by Activity		
<b>All events</b>		
Total number of classes: 8		
class	Occur. (abs.)	Occur. (rel.)
defense_evasion	29	41.43%
discovery	26	37.14%
initial_access	7	10.00%
persistence	4	5.71%
privilege_escalation	1	1.43%
Start	1	1.43%
resource_development	1	1.43%
End	1	1.43%

In PM, a deviation is a change in activities that affects how the process runs. Figure 4 shows Team 12’s deviation model. In this model, it can be seen that team 12 did not carry out all the activities of the general model that was discovered for the other teams (Figure 3), which means that they did not carry out some of the tactics that the others did it. On the other hand, all activities of Team 12 are included in the general model, and it does not necessarily indicate anything about the effectiveness of the attacker. Given these characteristics, provided only the event logs of the machine attacked by Team 12 and the general model, it would be possible to assert, based on PM-compliance techniques, that the activity performed on this machine is at least suspicious.

#### 4.4.3 PWNJUTSU operational flow

As part of the PWNJUTSU experiment, the authors represented the sequence of techniques carried out by Team 12, called the “operational flow”. Table 6 summarizes the attack steps described by the authors. The table only details the steps related to the Windows machine (n12-vm2), which is the subject of our study. The phases relevant to the Windows machine range from steps 9 to 16 within the total steps outlined by the authors. Events occurring between steps 0 and 8 and step 18 are beyond the scope of this analysis because

they do not involve the machine n12-vm2. Figure 5 shows a diagram of Team 12’s propagation space during their participation in the PWNJUTSU experimental campaign.

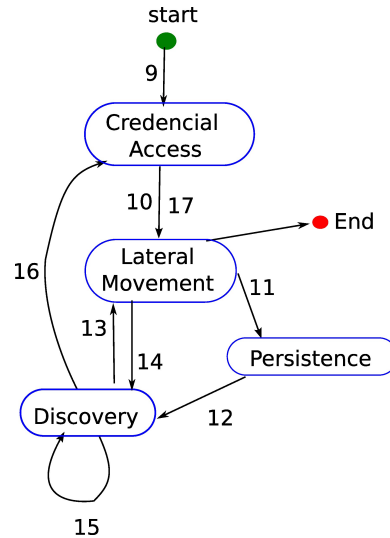


Figure 6. Operational flow of Team 12 during the PWNJUTSU experiment

It is important to note that the propagation space and techniques described in the original paper are analyzed from an external perspective of the hosts involved in the experiment. The authors specifically indicate that among all the attack techniques detailed in the MITRE ATT&CK matrix, they focus on those essential for an attacker trying to advance in the operational phase of propagation in the network. On the other hand, in our work, we concentrate on an internal perspective of the host, particularly the Windows host. It allows us to observe the consequences and events related to the attacks from a complementary point of view.

In Table 6, we have only the core information and the corresponding tactic for each technique identified in the original paper. It allows for a direct comparison with our study, where we use the MITRE ATT&CK tactics for modeling.

Figure 6 illustrates the progression of Team 12 within the PWNJUTSU scenario as discovered by the authors. This plot provides an external viewpoint and uses ATT&CK tactics. Similarly, Figure 4 shows the behavioral model identified through PM to provide an internal perspective.

Below, we analyze the behavior exhibited on the n12-vm2 machine from both an external perspective (the network) and an internal perspective using Symon logs on Windows.

The external behavior reveals that steps 9 and 10, executed by the attacker within the Tomcat service, are categorized under credential access and lateral movement tactics. Internally, the exploitation of the Tomcat service leaves a trail that can be categorized into three tactics: initial access, resource development, and discovery.

**Table 6.** Team 12 - Attack campaign tactics details (n12-vm2)

Step	Techniques	Tactic	Procedure
9	T1110.001	Credential Access	Bruteforce by guessing the service Tomcat
10	T1210	Lateral Movement	Exploited post authenticated remote service Tomcat
11	T1136	Persistence	Add an account for the user gomez on the service SSH
12	T1083	Discovery	Got a secret flag file on n12-vm2.
13	T1021.004	Lateral Movement	Got an access using SSH on n12-vm2 as user gomez
14	T1018	Discovery	Discovered remote system n12-vm3 from n12-vm1 using ARP table
15	T1046	Discovery	Performed network service scanning (T1046) from n12-vm2 to n12-vm3
16	T1110.001	Credential Access	Bruteforce by guessing the service SSH (user=root) on n12-vm3
17	T1021.004	Lateral Movement	Got an access using SSH service on n12-vm3

The creation of the user “gomez” (step 11 of the external perspective) is also detected from the internal perspective. However, unlike the external perspective, it is classified using multiple tactics (Table 7). Part of the creation is classified as a discovery because it was performed using the Windows net.exe command [Fisher, 2022], which can have multiple purposes. On the other hand, other log entries related to the creation of the user are classified as persistence tactics, just as from the external perspective, since the creation of files related to the user’s Windows profile is observed.

Some event logs were recorded in the internal system logs but not classified as malicious, e.g., the SSH accesses in steps 13, 16, and 17. Although a brute force attack was performed, the classification mechanism currently used in our research evaluates each log entry independently. It means the overall context is lost, e.g., many access attempts in a short period are not detected. In this sense, an SSH access attempt viewed independently is considered normal behavior.

The ARP command (step 14) is observed in the dataset obtained from the system trace. Although some Sigma rules at the Zircolite level allow the execution of such a command to be classified as malicious, in our case, it was not so classified due to a technicality related to the detection rules. The dataset used recorded the execution of the command “arp -a” but with a double space between the command (arp) and its parameter (-a). The detection sigma rule [Peacock, 2022] has a single space between the command and its parameter.

Steps 12 (getting the file with the secret flag) and 15 (network scan), although there are several executions of the Windows cmd.exe command and the Metasploit attack tool that could be related to the events, there is no precise evidence to identify the behavior from an internal perspective.

Table 7 lists examples of evidence found in the internal records of the n12-vm2 machine for steps 9, 10, and 11, which correlate with the tactics observed in Figure 4.

We note that the models shown in Figures 4 and 6 complement each other in several ways. The perspective provided by the internal model makes it easier to understand fine details and identify evidence left by the attacker on a particular computer. On the other hand, observation from an external position, such as the network, provides a global view of the attacker’s movements throughout the environment.

From the perspective of profiling an attacker’s behavior, we understand that both approaches have similarities, each from its viewpoint. The similarities are that both model the attacker by focusing on the “symptoms” reflected in the underlying system (Figures 4 and 6). For example, when ana-

lyzing in isolation the behavior of machine 12 on a specific computer (n12-vm2), unusual network traffic on the Tomcat service is detected, signaling through tactics anomalous behavior leading to the compromise of the machine (Table 6). The underlying system includes the machine in question and the network connections to and from the machine.

Observation from the network alone may not reveal the internal impact on the system. However, by examining the internal behavior, clear indicators of malware execution are identified in Table 7, proving the techniques used to create a new user and identifying the vulnerability. In this context, the term “underlying system” refers to internal system activity, as evidenced by operations with system processes, network connections, file actions, and changes to the Windows registry.

In summary, we understand that PM is a valuable tool for automatically identifying attacker behavior.

## 5 Profiling automated attackers: the WannaCry ransomware

As a second experiment to assess the effectiveness of the suggested approach, we explore the behavior of automated attackers. In particular, we use the WannaCry ransomware, whose attack was a global epidemic in May 2017.

The first approach to this experiment was carried out in the work [Rodríguez et al., 2021]. Although the task of modeling Wannacry’s behavior was addressed then, there was no consolidated methodology. In this section, we revisit the previous work and use the technological infrastructure defined therein, especially during the enactment phase, where data are re-acquired for this new experiment. From this point on, the remaining phases are applied to this new dataset according to the proposed methodology.

The massive increase in ransomware attacks has generated significant losses for different institutions since it captures information or blocks systems and then demands large sums of bitcoin as ransom. Attacks are not only limited to individuals; nowadays, they are mainly aimed at large organizations. A ransomware attack is the set of stages necessary to infect a system, considering that it starts with the distribution and infection of the device involved, followed by communication, search for files to infect, file encryption, and ransom demand. There are several approaches to detecting and preventing this type of attack [Aslan and Samet, 2020][Davies et al., 2021].

**Table 7.** Team 12 - Example of evidence found in the event log of the n12-vm2 machine

Step	Tactic	Evidence
9 & 10	Resource Development	Tomcat create file: \spawn3186357984909711176.tmp.dir\DNBAaBdn.exe
9 & 10	Initial access	Tomcat execute (log4j exploit): C:\system32\cmd.exe
9 & 10	Discovery	Tomcat execute (metasploit.Payload): tasklist.exe /v /fo csv /nh
11	Discovery	Tomcat execute: net localgroup administrators gomez /add
11	Persistence	Files Create: C:\Users\gomez\AppData\Roaming\*

The main parameters considered in detection-level investigations are registry keys, input and output activity of system files, process activity, operating system API function calls, and network activity [Malin *et al.*, 2012].

We will now outline the implementation of the method by following the four steps elaborated in Section 3.

## 5.1 Step 1: Enactment

This case study uses the dynamic analysis technique, explicitly running the malware, to observe its activity and the changes in the underlying system as it runs. Performing this type of analysis requires a secure environment isolated from production environments. The advantage of performing dynamic analysis is that observing changes in the system makes it possible to indirectly study how the malware works by monitoring changes to files, registry entries, processes, and network communications, among other things.

In this context, for the enactment phase, a controlled test environment is built on which the ransomware is executed. Through the generation of system events that are recorded in logs, the behavior of the artifact is studied so that a dataset is built with the information collected. The deployment of a secure environment is based on the use of virtualization. In [Rodríguez *et al.*, 2021], we presented the scenario (sandbox) developed to construct the dataset related to the analysis of the Wannacry ransomware. The layout is as follows: Machine 1 contains the installation of a SIEM based on the ELK stack [Elastic, 2023], and Victim 1 is installed with Windows 10 and the Sysmon tool [Microsoft, 2023]. Sysmon provides detailed information about process creation, network connections, file creation, and temporary variations.

## 5.2 Step 2: Extraction

Initially, the information collected from the environment is considered raw data. In the extraction phase, this data is filtered to determine candidate characteristics to build the required dataset. Unlike the previous work [Rodríguez *et al.*, 2021], where the data was extracted from the SIEM, in this case study, the data is obtained directly from the victim machine, i.e., from the system's log files. Zircolite directly supports Windows EVT X files for processing [Charter, 2008].

### 5.2.1 Event collection and organization

In the knowledge extraction phase of malicious code, one of the critical tasks is to identify events of interest and organize them systematically. In the context of Windows systems, during execution, ransomware interacts with the host

system from four main perspectives: processes, file system, registry, and network activity [Kao and Hsiao, 2018].

**Processes** Processes can be direct indicators of malware execution. Process monitoring includes process inheritance (parent and child processes), parameters used, image paths, and loaded dynamic libraries (DLLs).

**Registry** In malware analysis, registry scanning provides valuable information to understand various aspects, such as changes made by specific programs, signs of infection on the computer, and artifacts related to persistence mechanisms.

**File system** Fluctuations in the number of files indicate that the malware may be deleting associated files, modifying specific files, or deleting artifacts created to disguise themselves. For example, encryption of user files will generate frequent input and output operations on the file system.

**Network activity** Network activity is critical to the malware life cycle. Implementing network functions is essential to verify domain names, propagate worms, or manage command and control (C&C) servers.

### 5.2.2 Labeling of events

As in the PWNJUTSU experiment, the events generated by the Windows machines during the execution of the ransomware were tagged using the Zircolite tool. However, in this experiment, it is worth noting that there was no need to transform the events into JSON format, as Zircolite can directly process Windows event logs in EVT X file format.

For example, the result of Zircolite event labeling for one Wannacry run is shown in Figure 7.

**Figure 7.** Zircolite GUI - WannaCry Tactics labels

It is important to note that no additional adjustments were required due to the rule set pre-treatment performed in PWNJUTSU and the manual addition of MITRE ATTACK tactic labels (when none were available). The same rule sets that were previously customized were used.

### 5.2.3 Building the event log

To conduct this experiment, we based the event log generation on creating a CSV file using an output template designed for Zircolite. A summary of the representation of critical information in the event logs associated with the WannaCry ransomware is provided in Table 8.

The sequence of events is organized according to the Sysmon timestamp. On the other hand, the process instance or CaseID refers to a specific execution of the ransomware in the experimental environment. After processing the data for each execution, this field was manually added to the dataset. In short, the CaseID is an identifier associated with each ransomware run. Usually, event logs contain noisy, infrequent, missing, or incorrect process information. These are commonly referred to as outliers. Repeated attacks on the same machine help mitigate the problems caused by this anomaly since such anomalies may be occasional and do not refer to the attack process but to the operating system's behavior. On the other hand, the very nature of PM algorithms requires several instances to build the process model. Thus, by having multiple executions of the same ransomware, the PM tool can distinguish between cases and compare numerous process instances.

The event log contains other relevant fields, such as "computer name, username, rule description, etc". In this context, it is crucial to note that the static fields in the environment, such as computer name and username, are consistent across each execution of the ransomware. This is because, in the experiment, each instance of the ransomware is executed under the same experimental conditions, resulting in the consistency of these values.

## 5.3 Step 3: Discovery

To discover the patterns and action flows followed by the ransomware, we applied PM to the different datasets on five independent instances of Wannacry execution, which, in our case, represents the action of an automated attacker.

The models were obtained using Inductive Miner (IM), where different noise filtering settings were applied. The generated model in Figure 9 is the result of setting the noise to 0.2, where, similar to the case of PWNJUTSU, the slider of the PROM plugin (path slider) was set to 0.8.

The generated model consisted of 7 activities - in this context, MITRE Att&CK tactics - performed by the ransomware. Among them are the tactics referred to as defense evasion, resource development, execution, privilege escalation, lateral movement, persistence, and impact. These findings are discussed in the following subsection.

## 5.4 Step 4: Analysis

We analyzed the data and the model generated from 2 perspectives: focusing on the data generated by the SIEM and comparing the events before and after the labeling process. Then, the models obtained from PM are compared with the information obtained from traditional analysis mechanisms of the Wannacry ransomware.

### 5.4.1 Data analysis

After running five instances of the same ransomware, the SIEM-logged events sent by Sysmon during the experiment were analyzed. For this purpose, a custom SIEM panel was used to examine each of the Sysmon events triggered during the execution of the samples. The determination of each sample's start and end time was based on Sysmon events related to the creation (EventID 11) of two specific files. The timestamp of creating the file named "Endermanch-WannaCrypt0r.exe", representing the WannaCry instance used in the experiment, was used to determine the start of the infection process. Similarly, to determine the completion time, we consider the timestamp of the creation of the file "Microsoft-Windows-Sysmon%%4Operational.evtx" in the "tmp" directory, corresponding to the copy of the system event log obtained for the analysis. In this way, a period was established to determine when each ransomware sample was active on the system.

In Figure 8, the Sysmon ID 11, indicating creating a file, was the most frequently triggered event. However, when tagged with Zircolite, only 17 of these events were classified as malicious. The malicious events detected the creation of an executable by another executable (related to resource development tactics) or the detection of files created in the Windows startup directory (related to persistence). Behavior related to file encryption by the ransomware, which typically stores files with the same name but different extensions (e.g., <file\_name>.wncryt), was not detected.

Top Event IDs [Winlogbeat Overview]	
Event IDs	Count
11	7,151
7	5,302
12	4,941
10	4,732
23	3,609
13	2,350
2	896
1	96
1001	84
5	80

Figure 8. Wannacry-Top event SIEM

According to the results compiled in Figure 8, the second most common Sysmon event triggered by the dataset was event ID 12, which started when a registry key or its value was created or deleted. Notice that the Zircolite rules did not identify any of these events as malicious in our experiment.

Another Sysmon event in the top 10 is event ID 2, which is triggered when a process changes the creation time of a file. Although malware sometimes changes file creation times to hide the time of infection, this behavior was also not considered malicious. In addition, some event entries with ID 3, which log TCP/UDP connections on the machine, were not



**Table 8.** Fragment of example WannaCry event log

CaseID	Time	Activity	Computer	UserName	RuleDescription
1	2023-11-10 12:24:18.462	persistence	PM	PM\Userw	Suspicious Conhost Legacy
2	2023-11-10 12:25:06.359	lateral_movement	PM	PM\Userw	WannaCry Ransomware
3	2023-11-10 12:25:06.359	defense_evasion	PM	PM\Userw	File Permissions Modifications
4	2023-11-10 12:24:18.023	execution	PM	PM\Userw	Non Interactive PowerShell

regarded as malicious. However, some were related to using the TOR network [Dingledine, 2023].

Event ID 1 is also noteworthy, although to a lesser extent, as one of the most relevant events triggered when a process is created. About 50% of the events were classified as malicious since defense evasion and lateral movement tactics were detected in connection with creating such processes.

Events with ID 5, triggered at the end of a process, are also present in Figure 8. Although there are instances of these events in the constructed dataset, they were not considered malicious, although they are associated with actions related to ransomware processes.

Another common event triggered in the experimental dataset is event ID 13, which started by setting a registry value. Although at a low rate, malicious behavior was detected, particularly in persistence tactics where the ransomware attempts to add a “New Root or CA or AuthRoot certificate to storage”.

The image load event (ID 7) is logged when a module is loaded in a specific process and is related to defense evasion and execution tactics by detecting suspicious actions such as loading “Advapi31.dll” or “WMI Modules Loaded”.

Event ID 10 reports when a process opens another process. In the generated dataset, this type of event is associated with privilege escalation tactics. This operation often precedes information requests or reads and writes to the target process’s address space. It makes it easier to detect hacking tools that read the memory contents of other processes to steal credentials and use them in pass-the-hash attacks.

Event ID 1001 is related to Windows anti-malware protection, which is disabled in our scenario. Finally, event ID 23 is connected to file deletion actions and was not classified as a malicious operation during the labeling process.

According to the results obtained in this study, the ransomware sample used to build the dataset did not significantly trigger other Sysmon events. On the other hand, Sysmon logged some events that were not classified as malicious during the labeling process.

These results are understandable, considering that the primary function of ransomware is to encrypt files and demand a ransom for their recovery. Its main activity is related to file operations. Using SIEM and performing this type of analysis, valuable information was obtained about which Sysmon events are most frequently triggered by ransomware. This information gives organizations a guide to focusing their analysis efforts when investigating ransomware incidents in their environment. It also helps to understand the actions taken by the ransomware sample on the infected system, providing a starting point for identifying patterns through PM.

## 5.4.2 Model analysis

From a less formal or intuitive perspective, the WannaCry malware can be described as a self-propagating ransomware with worm-like characteristics that spread across internal networks and the public Internet by exploiting a Server Message Block (SMB) protocol vulnerability. Although several variants of this malware exist, WannaCry’s operation is not considered complex or innovative compared to more modern ransomware [Group-IB, 2018].

It commonly enters a compromised computer through diverse methods, like self-propagating via SMB or activation via email attachments. Subsequently, an internal module extracts various embedded application elements: tools for encrypting and decrypting data, files with encryption and configuration keys, a TOR application for command and control communication, and a propagation component facilitating SMB exploitation.

During infection, the malware scans and encrypts files in different formats, making them inaccessible. Simultaneously, it presents a ransom note, demanding payment in Bitcoin for file decryption.

The MITRE ATT&CK provides a series of detailed descriptions of various types of malware, e.g., a description of the WannaCry ransomware. Specifically, 15 techniques are detailed in the enterprise matrix, grouped into eight tactics, as briefly summarized in Table 9.

In the model shown in Figure 8, it is evident that six of the eight tactics described by the MITRE framework are identified in our experimental environment. In addition, the use of the resource development tactic is determined, which is not described in detail by MITRE. As an added value, the model describes the flow of actions using these tactics. Initially, this flow allows the ransomware to perform one of the three actions described below simultaneously: a) resource development, b) defense evasion, and c) execution.

Paths (a) and (b) refer to a set of actions that the ransomware executes continuously and repeatedly throughout its lifecycle. As for path (c), which is dominated by execution tactics, the ransomware enters a sub-cycle that is also executed concurrently and refers to the actions: c.1) persistence, c.2) privilege escalation, and c.3) lateral movement. In the path of (c.1), the ransomware also concurrently performs tasks related to impact techniques.

From a functional perspective, this model reflects behavior consistent with what is expected, where the ransomware attempts to disconnect defenses and then executes the attack, combined with persistence, privilege escalation, lateral movement, and impact tactics are performed. However, the ATT&CK framework describes other types of activity not detected in our experiments. Considering the data analysis performed in Section 4.4.1, where there is much information that

**Table 9.** Tactics and techniques of Wannacry in MITRE ATT&CK

Tactic	Techniques
Execution	Windows Management Instrumentation
Persistence	Create or Modify System Process: Windows Service
Privilege escalation	Create or Modify System Process: Windows Service
Defense evasion	File and Directory Permissions Modification
Discovery	File and Directory Discovery
Lateral movement	Exploitation of Remote Services
Command	Control & Remote Service Session Hijacking
Impact	Data Encrypted for Impact

was not tagged by the labeling process, we understand that this is a direct consequence of the need to improve the tagging tools used and consider specific features of ATT&CK, such as modeling based on techniques instead of tactics.

We see the results as another step in analyzing the activities of malware or, more generally, automated attackers, identifying and modeling advanced threats through PM, and helping to understand malicious actions.

### 5.4.3 WannaCry execution flowchart

Figure 10 illustrates the flow generated from the technical analysis described in [Elastic, 2017]. For our analysis, we will take the flow that starts when Wannacry begins to damage the system. In step (2a), a service called mssecsvc.exe is created and loads Resource tasksche.exe, which extracts the components in a zip file. Then, steps 3, 4, and 5 modify system attributes based on the main configuration file (c.wnry) and set keys for the AES encryption algorithm.

Steps 7, 8, 9, 10, and 11 initiate the system file encryption process. Steps 12 and 14 implement the persistence mechanisms, while steps 13 and 17 attempt to establish communication with the C&C using the TOR network. Step 15 marks the initiation of ransomware demands by creating files on the hard drive and displaying them on the screen. Finally, steps 16 and 18 remove defenses and system backups.

Table 10 shows evidence of the findings made on the logs of the system infected by the malware in our sandbox. At the same time, we can observe the tactics assigned during the tagging phase. On the one hand, these labels give rise to the model shown in Figure 9, and on the other hand, the relationship between the evidence and the execution flow manually defined in Figure 10 can also be seen. Consequently, both models are closely related in terms of the tactics used by the automated attacker.

## 6 Related work

In [van der Aalst and de Medeiros, 2005], the authors describe how to use a process mining algorithm to discover acceptable or typical behavior of a system based on a set of records or audit trails. Malicious behavior is identified by applying compliance verification techniques by identifying patterns that do not correspond to acceptable behavior. The central hypothesis proposes that anomalous audit records differ from typical behavior sequences. The researchers suggest that process mining techniques can assist in this task.

In [Macak et al., 2022], the authors present a systematic review of works combining PM with computer security. The paper explores the potential of PM to help in cybersecurity, collecting existing process mining applications and discussing different research directions to address current challenges. As part of their work, they conclude that adopting a process-oriented approach to the detection and advanced prevention of cyberattacks and incidents would be beneficial.

In [de Alvarenga et al., 2015], the authors use process mining techniques to discover attack strategies from intrusion detection system (IDS) alerts. Their proposed methodology uses process mining techniques to extract the alerts and generate a high-level process model of the attacker's behavior. The authors propose to use the signatures registered in the IDS logs as the activities of the events for process mining. However, this method has a downside: the generated models heavily rely on the IDS signature, which is proprietary. Consequently, the models lack interoperability between different technologies and are strongly linked to the alerts.

Attack Flow [for Threat-Informed Defense, 2022] provides a unified and structured language and tools for describing behavior and sharing manually generated models in a specific attack-flow language. The main challenge of this attack-flow data model is to model attacker behavior accurately and manually once the attack has been studied and understood. Our approach combines the advantages of Attack Flow, which allows an attacker's behavior to be modeled as a sequence of activities, with the added benefit of using PM to automatically extract the attacker's behavior model by analyzing audit logs and applying process discovery techniques.

The article [Berady et al., 2022] presents a formal framework describing attack campaigns. As part of the framework, a model is provided to describe the infrastructure targeted by the attack. The model also describes the knowledge the attacker acquires during the development of the campaign. The operational semantics of the attack techniques described in the MITRE ATT&CK, through which the attacker can achieve his goal, is also contemplated. The article uses a particular attacker as an example to conduct case analysis. Through a flowchart built manually, the authors express that the attacker is based on an attack routine since he tends to perform similar techniques in similar environments. Then, they claim that a specific attack pattern can be identified. PM techniques could automatically obtain a model with characteristics similar to the manually created one they provide.

In this section, we have described some of the efforts related to the use of PM in the field of cybersecurity and also described the actions based on the operational semantics

**Table 10.** Example of evidence found in the event log of WannaCry experiment

Step	Tactic	Evidence
3,4 & 5	Defense evasion	conhost.exe 0xffffffff -ForceV1; "attrib +h ."
	Defense evasion	conhost.exe 0xffffffff -ForceV1; "icacls . /grant Everyone:F /T /C /Q"
	Privilege escalation	C:\tmp\taskdl.exe; "C:\tmp\Endermanch-WannaCrypt0r.exe"
7,8,9,10 & 11	Lateral movement	C:\tmp\Endermanch-WannaCrypt0r.exe\; taskdl.exe
	Privilege escalation	C:\tmp\taskdl.exe;"0x1ffff"
	Resource development	C:\tmp\@WanaDecryptor@.exe";C:\tmp\taskdl.exe
12	Persistence	cmd.exe /c reg add HKLM...\Run /v \fdpdfroadcstuf441\ /t REG_SZ /d
13 & 17	Lateral movement	C:\tmp\TaskData\Tor\tor.exe
	Resource development	@WanaDecryptor@.exe co; TaskData\Tor\taskhsvc.exe
16 & 18	Defense evasion	cmd.exe /c vssadmin delete shadows /all /quiet 0026 wmic shadowcopy
	Defense evasion	wmic shadowcopy
	Defense evasion	bcdedit /set default bootstatuspolicy ignoreallfailures
	Defense evasion	wbadmin delete catalog -quiet

of attacks based on MITRE ATT&CK. Our work combines PM techniques and tools with the taxonomies provided by MITRE ATT&CK with two specific objectives: to discover and give a characterization to a flow of actions that help to detect better and understand attacker behavior inspecting large amounts of data and to express that behavior using a standardized language.

## 7 Discussion

Our analysis demonstrates that models generated by PM can help understand attacker behavior and provide valuable insights for analyzing it. Additionally, PM makes it easy to share these models with the broader community.

The attacker's objectives, knowledge, and modus operandi significantly impact a cyber attack, dictating the tactics and techniques used and the order in which they are applied. In the following sections, we will discuss some outcomes and unresolved issues related to this approach, leading to further research opportunities.

### 7.1 Using the ATT&CK taxonomy

We use the MITRE ATT&CK taxonomy to classify attackers. By focusing on tactics, we can understand their behavior and detect their actions.

In Section 4.4.3 and Section 5.4.3, we compared manually designed operational flows and automatic models created using PM techniques. We found that they were mostly comparable, but there were differences in how events were classified. Flow diagrams generally use less detailed representations, while specific techniques and procedures from MITRE can provide a more in-depth understanding of attackers' behavior.

Focusing on various techniques can result in the creation of more detailed models. However, this also makes them complex to analyze. To address this issue, additional research is required to tackle the challenges of labeling events of interest, filtering uninteresting behavior from logs, and keeping the dataset as optimal as possible. One way to simplify analysis is to apply a stepwise refinement method with the help of automated tools that can focus on specific regions.

### 7.2 Event classification

During our event classification phase, we label events according to Zirconite rules based on the Sigma project. However, this process has limitations. For instance, an event may meet the criteria for multiple Sigma rules, which leads to duplicate output events with different tactical tags. This outcome is unsatisfactory since it fails to represent real-life events accurately. To resolve this issue, we have assigned the first tactic tag defined in the rule to avoid generating multiple events for the duplicate registry entry.

Although Sigma rules are highly respected and used by the community across multiple SIEMs, they have disadvantages. The rules do not require labeling tactics or techniques related to the MITRE taxonomy, which means it is left to the author's discretion, introducing subjectivity. Therefore, it can be challenging to establish a direct link between a given event within a record and a specific tactical or technical attribute. On the other hand, Zirconite is unique among open-source tools because it automates a complex task.

Sometimes, the individuals responsible for creating rules might forget to assign a tactical label. In such cases, we have the necessary expertise to modify rules and include the relevant taxonomy tag, as discussed in Section 4.2.2. Although it is possible to exclude such events from the model using PM tools and filtering techniques, this approach could result in rejecting suspicious behavior. An alternative solution is the TTP-Based Hunting methodology [Daszczyszak *et al.*, 2019], which concentrates on detecting malicious activity. Nevertheless, there are limited tools available to support this approach, and further research is necessary to carry out the tagging process effectively, which is a crucial aspect.

Our results are limited by how the tactics are associated with ATT&CK. The labeling process relies on how the techniques are connected to each of the rules used during the extraction phase and the formality of the mapping process. Although our experiments are based on rules subject to the scrutiny of an expert cybersecurity community, it is essential to acknowledge the possibility that some sightings may have been incorrectly mapped. This consideration is crucial to understanding our results' limitations and potential biases.

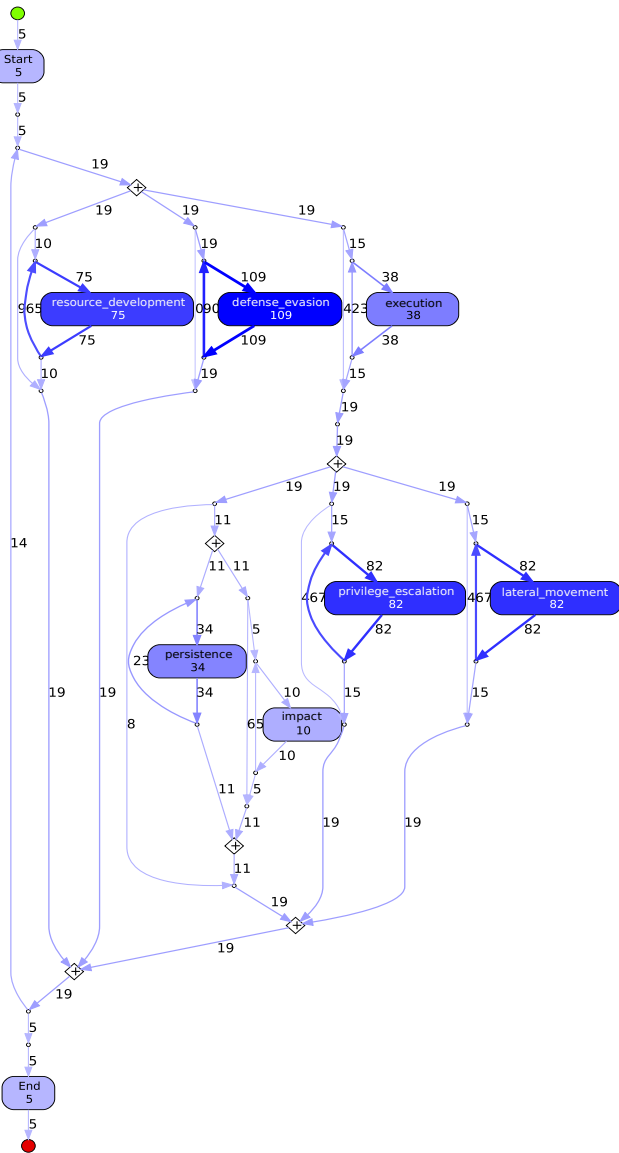


Figure 9. Wannacry-Inductive Miner model (ProM)

### 7.3 Human vs automated attackers

It is essential to differentiate between human and automated attackers’ characteristics. Although the attacks studied may not be directly related, human attackers tend to engage in attacks spread over time, making the chronology of events appear more dispersed. The sequence of events generated by human attackers often spans minutes, hours, or even days. On the other hand, automated attackers manage a sequence of events that can occur within milliseconds due to their nature. This last aspect can be challenging when applying PM techniques since these techniques assume that a business process is represented as a sequence of ordered events in time. In the case of automated attacks, if the timestamp’s precision is insufficient to distinguish a deterministic sequence of events, we could face difficulties in modeling the activities carried out.

It is also emphasized that human behavior is much more heterogeneous, resulting in variations of models. This aspect can accentuate the differences, especially when using MITRE ATT&CK techniques for behavioral representation and modeling or when relying on procedures that instantiate a specific attack technique. In this context, having more cases

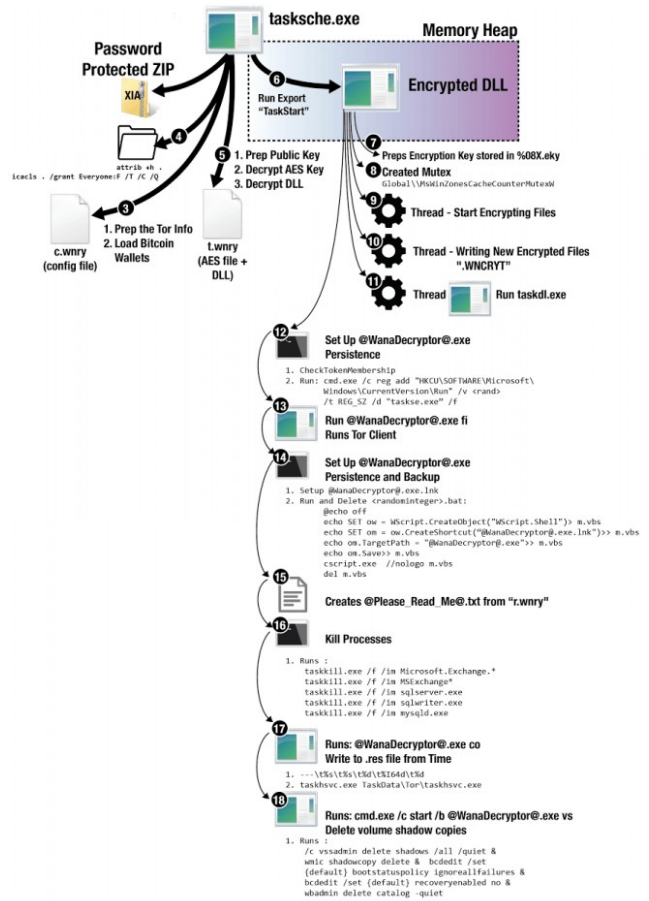


Figure 10. Wannacry execution flowchart from [Elastic, 2017]

for modeling allows models to be more closely matched to reality or the underlying security issues, improving the diversity of procedures depending on the activities performed. PM enhancement techniques could be instrumental in this regard.

Automated attackers, on the other hand, operate algorithmically, performing actions without conscious thought. In this context, the main feature where PM enhancement techniques could be instrumental is to identify variations related to changes in the execution environment, which in turn can influence the behavior of the automatic attacker.

In summary, we understand that these situations become evident when defining activity-based models with more detail. In our case, activities are related to MITRE ATT&CK tactics, which define a higher level of abstraction; techniques and procedures are the elements with a lower level of abstraction and greater detail.

### 7.4 Research limitations

It is crucial to consider multiple perspectives when analyzing the results of our experiments. We focus on building process models that align with the application domain, specifically, creating attacker behavioral profiles. To do this, we have conducted a qualitative assessment of the results, which may be subjective and open to interpretation. However, this approach enables us to make observations that validate our method. Nevertheless, qualitative assessments pose challenges when comparing our results with other methods, making it challenging to assess the effectiveness of our proposed

mechanisms. As a result, we plan to conduct new experiments based on expert knowledge, allowing us to obtain numerical estimates that facilitate comparison with other work. By applying PM-conformance techniques and algorithms in different scenarios, we can obtain comparable results on the main PM quality dimensions: fitness and precision.

We have applied the method to one case of human attackers and one case of automated attackers. Thus, findings rely on a relatively small sample size to test our assumptions. Nevertheless, findings showed that PM is a valuable tool for attacker modeling and complements other existing strategies, such as generating attack flows and understanding attacker behavior expressed in natural language, which is often difficult for non-experts to understand. This limitation of experimentation is mainly due to the difficulty in obtaining or generating realistic datasets. However, further work can be developed, e.g., using and extending our controlled test environment developed for enacting ransomware.

Finally, PM is still an evolving discipline, particularly concerning its application in the cybersecurity domain. As described in Section 6, significant efforts in this area have taken an ad hoc approach to addressing specific problems rather than building models that capture attacker knowledge. Although this lack of comparative works makes it difficult to perform meaningful qualitative comparisons, it also opens new research opportunities that must be further developed.

## 8 Conclusion and future work

Our methodology involves the application of process mining to identify process models associated with observed attack strategies. We analyze by focusing on events that provide evidence of attacks by expert human actors and automated actors. The data corresponding to human attackers were obtained from the experimental set of the PWNJUTSU project. At the same time, we generated the dataset of an automated WannaCry attacker as part of our experimentation. In both cases, our primary focus is on data generated by Windows operating system hosts. To facilitate analysis, we extract raw data from the Windows system, transform it into an event log, and map it for use with process mining tools.

By observing an attacker's behavior as a business process, we can systematically classify the signs of an attack. The method helps us analyze the attack by breaking it down into its components (at potentially many levels of detail) and identifying how they are linked. It enables us to gain a clear understanding of the attack and its underlying processes, as well as the different vulnerabilities involved.

The MITRE ATT&CK framework is a valuable tool for predicting an attacker's actions by analyzing their behavior through a breakdown of tactics and techniques. Experimentation has proven that PM can be an effective tool for defense teams to identify the characteristics of attacker actions, especially the sequence of events. It can help in the development of effective mitigation measures.

Although collaborative projects like Attack Flow exist, generating attack flows requires manual effort. Therefore, PM offers a valuable solution as it can automatically analyze data compiled from affected systems.

A critical aspect of the method is Zircolite's event labeling. Although we identified some limitations, its effectiveness in event tagging has created a beneficial attack model. Data labeling is a crucial aspect of machine learning, which involves tagging and classifying data. A recent study on machine learning for labeling is presented in [Fredriksson *et al.*, 2020]. Our future research aims at using machine-learning techniques to enhance event labeling. Additionally, we plan to develop a ProM plugin that can generate models based on the Attack Flow specification, making model sharing easier and improving their usability in threat analysis.

We plan to use more specific ATT&CK techniques and procedures to improve model development. So far, we have only used PM-discovery techniques. However, implementing our approach in a real system poses several challenges. To address this, we would like to experiment with applying PM-conformance techniques. It will help us assess the potential false positive rate by comparing discovered models with a running model generated from operating system data without being under attack.

## Acknowledgements

Thanks to A. Berady and co-authors for the PWNJUTSU dataset used in this paper.

## Availability of data and materials

The datasets generated and analyzed during the current study are available in GSI-Datasets repository, <https://gitlab.fing.edu.uy/gsi/attacker-profiling-datasets>.

The public release of the PWNJUTSU dataset are available in <https://pwnjutsu.irisa.fr/>

## References

- Aslan, O. and Samet, R. (2020). A comprehensive review on malware detection approaches. *IEEE Access*, 8:6249–6271. DOI: 10.1109/ACCESS.2019.2963724.
- Azzini, A., Braghin, C., Damiani, E., and Zavatarelli, F. (2013). Using semantic lifting for improving process mining: a data loss prevention system case study. In *Proc. of the 3rd Intl. Symp. on Data-driven Process Discovery and Analysis*, volume 1027 of *CEUR Workshop Proceedings*, pages 62–73. CEUR-WS.org. Available at: <https://ceur-ws.org/Vol-1027/paper5.pdf>.
- Berady, A., Jaume, M., Triem Tong, V. V., and Guette, G. (2022). Pwnjutsu: A dataset and a semantics-driven approach to retrace attack campaigns. *IEEE Transactions on Network and Service Management*, 19(4):5252–5264. DOI: 10.1109/TNSM.2022.3183476.
- Center, C. S. R. (2015). Cyber attack: Definition. Available at: [https://csrc.nist.gov/glossary/term/cyber\\_attack](https://csrc.nist.gov/glossary/term/cyber_attack).
- Charter, B. (2008). EVTX and Windows EventLogging. Technical report, SANS Institute. Available at: <https://www.giac.org/paper/gcia/2999/evtx-windows-event-logging/115806>.



- Daszczyszak, R., Ellis, D., Luke, S., and Whitley, S. (2019). Ttp-based hunting. Technical report, Internet Engineering Task Force. The MITRE Corporation. Available at: <http://www.mitre.org/sites/default/files/2021-11/prs-19-3892-ttp-based-hunting.pdf>.
- Davies, S. R., Macfarlane, R., and Buchanan, W. J. (2021). Review of current ransomware detection techniques. In *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, pages 1–6. DOI: 10.1109/ICEET53442.2021.9659643.
- de Alvarenga, S. C., Zarpelão, B. B., Junior, S. B., Miani, R. S., and Cukier, M. (2015). Discovering attack strategies using process mining. In *Intl. Conf. on Telecommunications*, pages 119–125, Brussels, Belgium. International Academy, Research, and Industry Association (IARIA). DOI: 10.13140/RG.2.1.4524.4008.
- Dehghantanha, A., Conti, M., and Dargahi, T. (2018). *Cyber Threat Intelligence*. Springer Cham, Switzerland. DOI: 10.1007/978-3-319-73951-9.
- Depaire, B., Swinnen, J., Jans, M., and Vanhoof, K. (2013). A process deviation analysis framework. In *Business Process Management Workshops*, pages 701–706. Springer. DOI: 10.1007/978-3-642-36285-9\_69.
- Dingledine, R. (2023). The tor project. Available at: <https://www.torproject.org/>.
- Elastic (2017). Wcry/wanacry ransomware technical analysis. Available at: <https://www.elastic.co/blog/wcrywanacry-ransomware-technical-analysis>.
- Elastic (2023). Elastic Stack. Available at: <https://www.elastic.co/elastic-stack/>.
- Fisher, T. (2022). Net command. Available at: <https://www.lifewire.com/net-command-2618094>.
- for Threat-Informed Defense, C. (2022). Attack flow. Available at: <https://center-for-threat-informed-defense.github.io/attack-flow/>.
- Fredriksson, T., Bosch, J., and Olsson, H. H. (2020). Machine learning models for automatic labeling: A systematic literature review. In *Proceedings of the 15th International Conference on Software Technologies - ICSoft*, pages 552–561, Online Event. INSTICC, SciTePress. DOI: 10.5220/0009972705520561.
- Group-IB (2018). the evolution of ransomware and its distribution methods. Technical report, Group-IB. Available at: <https://go.group-ib.com/hubfs/whitepaper/group-ib-the-evolution-of-ransomware-white-paper-2018-en.pdf>.
- Kao, D.-Y. and Hsiao, S.-C. (2018). The dynamic analysis of wannacry ransomware. In *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pages 1–1. DOI: 10.23919/ICACT.2018.8323681.
- Konsta, A. M., Spiga, B., Lluch-Lafuente, A., and Dragoni, N. (2023). A survey of automatic generation of attack trees and attack graphs. *CoRR*, abs/2302.14479. DOI: 10.1016/j.cose.2023.103602.
- Leemans, S. J. J., Fahland, D., and van der Aalst, W. M. P. (2013). Discovering block-structured process models from event logs - a constructive approach. In *Application and Theory of Petri Nets and Concurrency*, pages 311–329, Berlin. Springer. DOI: 10.1007/978-3-642-38697-8\_17.
- Leemans, S. J. J., Fahland, D., and van der Aalst, W. M. P. (2014). Discovering block-structured process models from incomplete event logs. In *Application and Theory of Petri Nets and Concurrency*, pages 91–110. Springer. DOI: 10.1007/978-3-319-07734-5\_6.
- Lyon, G. (2023). Utility for network discovery and security auditing. Available at: <https://nmap.org/>.
- M., S. (2021). A sysmon configuration file for everybody. Available at: <https://github.com/SwiftOnSecurity/sysmon-config>.
- Macak, M., Daubner, L., Fani Sani, M., and Buhnova, B. (2022). Process mining usage in cybersecurity and software reliability analysis: A systematic literature review. *Array*, 13:100120. DOI: 10.1016/j.array.2021.100120.
- Malin, C. H., Casey, E., and Aquilina, J. M. (2012). *Malware Forensics Field Guide for Windows Systems: Digital Forensics Field Guides*. Syngress Publishing. Available at: [https://opac.atmaluhur.ac.id/uploaded\\_files/temporary/DigitalCollection/NDJmMjc3M2IzZmYyMWNmNmU5N2RlZTRjMTg5NjdhNzg2ODNlYWQ0Ng==.pdf](https://opac.atmaluhur.ac.id/uploaded_files/temporary/DigitalCollection/NDJmMjc3M2IzZmYyMWNmNmU5N2RlZTRjMTg5NjdhNzg2ODNlYWQ0Ng==.pdf).
- Messe, N., Chiprianov, V., Belloir, N., Hachem, J. E., Fleurquin, R., and Sadou, S. (2020). Asset-oriented threat modeling. In *19th IEEE Intl. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 491–501, Guangzhou, China. IEEE. DOI: 10.1109/TrustCom50675.2020.00073.
- Microsoft (2023). System monitor tool. Available at: <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>.
- MITRE (2023). Mitre enterprise tactics. Available at: <https://attack.mitre.org/tactics/enterprise/>.
- Myers, D., Radke, K., Suriadi, S., and Foo, E. (2017). Process discovery for industrial control system cyber attack detection. In *ICT Systems Security and Privacy Protection*, pages 61–75, Rome, Italy. Springer. DOI: 10.1007/978-3-319-58469-0\_5.
- Nahorney, B. (2020). Threat trends: Endpoint security. Available at: <https://blogs.cisco.com/security/threat-landscape-trends-endpoint-security>.
- Peacock, C. (2022). Suspicious network command rule. Available at: [https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process\\_creation/proc\\_creation\\_win\\_susp\\_network\\_command.yml](https://github.com/SigmaHQ/sigma/blob/master/rules/windows/process_creation/proc_creation_win_susp_network_command.yml).
- Rodríguez, M., Betarte, G., and Calegari, D. (2023). Discovering attacker profiles using process mining and the MITRE att&ck taxonomy. In *12th Latin-American Symp. on Dependable and Secure Computing, LADC 2023*, pages 146–155. ACM. DOI: 10.1145/3615366.3615372.
- Rodríguez, M., Betarte, G., and Calegari, D. (2021). A process mining-based approach for attacker profiling. In *2021 IEEE URUCON*, pages 425–429, Uruguay. IEEE. DOI: 10.1109/URUCON53396.2021.9647342.
- Roth, F. and Patzke, T. (2022). Sigma: Generic signature format for siem systems. Available at: <https://github.com/SigmaHQ/sigma>.
- Spitzner, L. (2002). *Honeypots: Tracking Hackers*. Addison-Wesley, USA. Available at: <http://www.it-docs.net>

- t/ddata/792.pdf.
- Strom, B., Applebaum, A., Miller, D., Nickels, K., Pennington, A., and Thomas, C. (2018). Mitre att&ck: Design and philosophy. Technical report, The MITRE Corporation. Available at: [https://attack.mitre.org/docs/ATTACK\\_Design\\_and\\_Philosophy\\_March\\_2020.pdf](https://attack.mitre.org/docs/ATTACK_Design_and_Philosophy_March_2020.pdf).
- van der Aalst, W. and de Medeiros, A. (2005). Process mining and security: Detecting anomalous process executions and checking process conformance. *ENTCS*, 121:3–21. Proc. of 2nd Intl. Workshop on Security Issues with Petri Nets and other Computational Models (WISP). DOI: 10.1016/j.entcs.2004.10.013.
- van der Aalst, W. M. P. (2016). *Process Mining - Data Science in Action, Second Edition*. Springer, Netherlands. DOI: 10.1007/978-3-662-49851-4.
- van Dongen, B. F., de Medeiros, A. K. A., Verbeek, H. M. W., Weijters, A. J. M. M., and van der Aalst, W. M. P. (2005). The prom framework: A new era in process mining tool support. In *Applications and Theory of Petri Nets*, pages 444–454, Berlin. Springer. DOI: 10.1007/11494744\_25.
- Waggabat (2023). Zircolite: Sigma-based detection tool. Available at: <https://github.com/wagga40/Zircolite>.