university of groningen / faculty of science and engineering

# Process Log Generation for Exploring Security Vulnerabilities and Violations in Business and Scientific Workflows

Bachelor Thesis

Nikiforos Kyparos

August 1, 2023

**First Supervisor:**
Prof. Dr. Dimka Karastoyanova

**Second Supervisor:**
Nafiseh Soveizi

**Abstract**

There is a growing need for security solutions to protect against anomalies that can occur in cloud-based business processes. Businesses are increasingly utilizing the cloud space for their data handling operations due to its scalable and flexible nature. Consequently, anomalies that range from inefficient procedures to malicious attacks have become ever more prevalent. Currently, businesses apply various methods to their processes to protect against abnormal behavior, including machine learning models that are trained on event log files and are capable of detecting anomalies. These files are extracted from the execution of the business process. However, based on our research, there is no comprehensive log file targeting the specific characteristics and requirements for detecting security and privacy violations. To address this gap, our focus is on exploring malicious attacks on the user's side of the process and generating event log files. In this research, we introduce a novel approach to generating log files for the user tasks of the business processes, aiming to enhance the robustness and accuracy of detection models.

Contents

# 1 Introduction

The sustainability and prosperity of a modern business depend heavily on the security and robustness of its software infrastructure. However, a recurring point of failure in the chain of operations is the human element. As a result, monitoring user behavior in the business processes is of paramount importance to any complex organization.

Business processes commonly adopt process models to streamline their operations. This model describes the structure of a series of tasks and sets requirements for their completion. In most cases, a business process may involve multiple participants and many contributing systems. Consequently, the points of entry for a malicious individual are also numerous. Such an intrusion could result in a critical attack on the infrastructure of a business process. Therefore, the development of tools capable of detecting and defending against such attacks becomes crucial.

One solution to address this challenge is the generation of log files used to train machine learning models. These log files contain the events that occur during the execution of a process, enabling the construction of the desired logs. Then, machine learning algorithms can be applied to these files to construct a detection model. To that end, many solutions have been proposed, and there is a growing interest in this area of research.

The aim of our research is to investigate the behavior of users that actively participate in a business process and collect logs based on their activity. Therefore, it is of particular importance to collect and log information that can aid in distinguishing between legitimate and malicious users. Information about the time of activity, the number of hits or calls from a specific address, payloads, IP reputation, and geo-location are all useful metrics to include in our logs.

Similarly to the range of metrics that should be collected, the potential types of attacks on user behavior vary significantly. There are four overarching branches that will be considered, and under which all attacks fall. The first category, called Distributed Denial-of-Service, concerns attacks that affect the amount of user traffic. The second, called Probe,

describes attempts to gather information about a system. The third is Remote-to-Local, which covers infiltrations to the network from outside sources. Finally, the fourth category is User-to-Root, and it concerns malicious system privilege escalations. As these attacks pose a serious threat to any business process, a solution has to be found that provides detection and safety enhancements.

In section 2, the context of the research and its key goals will be established. Section 3 will explore current approaches to the topic that also influenced this paper. In section 4, the implementation of the research into a working program will be described, and in section 5, the final results from the program will be displayed and analyzed. In section 6, the future steps for this research will be discussed. Finally, section 7 concludes the paper.

## 2 Background

### 2.1 The business process

Business processes are defined as a specification of the set of business activities required to achieve a business objective, as well as the information and resources they use [10]. The Business Process Modeling Notation (BPMN) is a graphical representation commonly used for depicting business processes, as illustrated in Figure 1. These processes are widely used by companies and offer streamlined workflows, well-defined procedures, and great agility.

In the context of business processes, there are two overarching types of activities. Firstly, the services, which are automated, can be handled by software applications or can be outsourced to the cloud, offering cost savings, scalability, and flexibility. On the other hand, there are users that interact through their own console APIs with the services. These users also need to be monitored in order to extract helpful information. The monitoring of user actions will be the main focus of this paper.

The activities that need to be performed by a user in a process are called user tasks. These are manual and non-automated steps in which an individual is tasked with carrying out a specific operation with the help of some software application. Examples of such tasks are approving requests,

producing reports, and filling out forms. These tasks are susceptible to a wide range of attacks that can be performed by intruders. Despite the large number of possible attacks, these intrusions follow some general patterns and leave a signature about their behavior. These signatures have been studied extensively and as a result, they can be classified according to their characteristics [6].

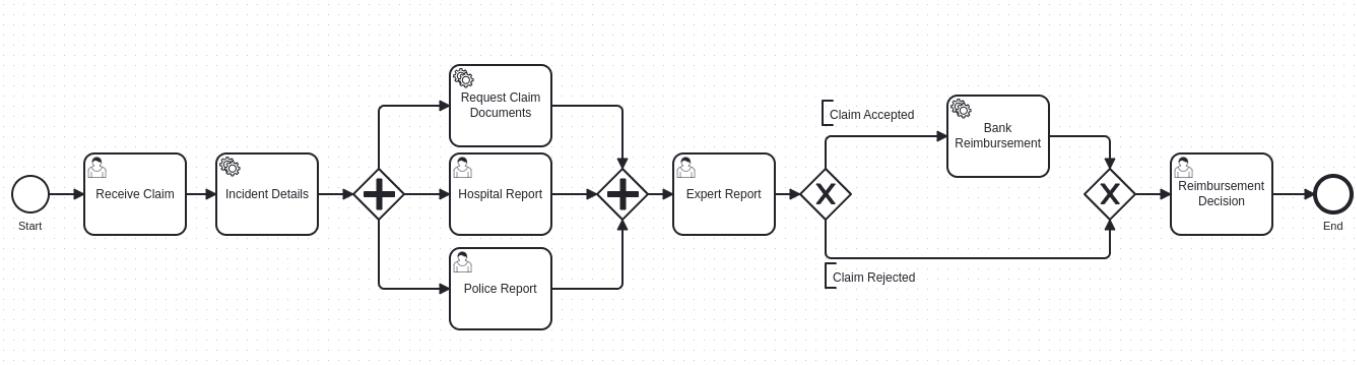


Figure 1: An example of a BPMN

### 2.2 Different Types of Attacks

The attacks covered in our research can be separated into four broad categories:

1. **(Distributed) Denial-of-Service (DDoS):** In DDoS attacks, an influx of requests is sent to the network with the goal of obstructing legitimate requests from being fulfilled [19]. Although it does not pose a threat to sensitive data it can cause significant damage to a company's resources.

2. **Probe:** In this method, the attacker deliberately sends a request that will be flagged as malicious and then analyzes the system's response in order to evaluate its detection capabilities [18]. Probe attacks can often constitute the first step of a much larger attack, where the malicious entity gains insight into the system architecture and then launches the more dangerous steps of the attack while leveraging this knowledge.

3. **Remote-to-Local (R2L) attack:** Here a non-local entity sends a packet over the network to a computer in order to obtain local user privileges by exploiting vulnerabilities in the target system's

authentication protocols [14]. After having obtained user privileges, the attacker can then operate on the network virtually undetected.

4. **User-to-Root (U2R) attack:** A U2R attack starts off with the attacker having access to a regular user account and then exploiting vulnerabilities to gain access to the root user [14]. Similarly to R2L, a malicious individual making a U2R will attempt to exploit system resources in order to execute actions that they would otherwise be unauthorized to execute.

### 2.3 Event logs

During the execution of a business process, various tasks and activities are carried out that provide insight into the specific actions performed. The collection of this data is typically done through Logging, resulting in an event log consisting of records documenting process events. In our research, we are interested in collecting this information about user tasks.

After obtaining log files containing the activities of the users during their tasks, we can compare these data points with the data that would have been produced by an attacker. Then, we can make decisions on whether there are any signs of malicious activity in the executed process. This idea forms the basis for a novel machine learning model that would be trained on these logs and would be able to accurately detect intrusions. Although this step is outside the scope of this thesis, it is important because it relies on the end product of this research, which is the user task logs, to conform to some predefined guidelines.

## 3 State of the Art

### 3.1 Business process

The field of business intelligence has seen a rise in popularity in recent years. Consequently, the interest in security and specifically in anomaly detection has increased sharply. This has led to the proposal of many new tools to serve this purpose.

#### 3.1.1 Log utilization

One area of research for the development of detection tools is log utilization. This involves researchers using the logs that business processes generate as the basis for their proposals. Considering that our paper also falls under this category, it is important to evaluate contemporary solutions.

Current approaches, such as the one by Nolle et al. [13], utilize event log data to detect anomalies. Specifically, the authors do not stop at the case-level anomalies, instead, they also make use of activity attribute data, which allows for greater coverage not available at the case level. The data are obtained from the executed service and user tasks in the form of a log file and then used in the training of a model. This has led to some promising results, as can be seen from the reported accuracy scores of these models. However, there exist additional features that can be considered during the creation of the log file.

In business workflows, Nolle et al. [12] propose a detection method for anomalies using autoencoders, which does not require prior knowledge about the given process and does not rely on a clean (i.e. containing no anomalies) data set for its training phase. Similarly to [13], the same authors extract both real-life event logs and synthetic event logs from processes and then transform them before using them to train an autoencoder. The logs contain at least three columns which are essential for identifying each trace. These columns are "Trace ID", "Timestamp" and "Activity" which distinguish between different activities in the process. There is also the option to add event attributes such as a "User" column. The main benefit of the autoencoder is that it provides a detailed breakdown of detected anomalies, which unlike previous approaches can identify the specific anomalous event within a sequence. Furthermore, it is able to analyze which attribute of the event (e.g. the user) is anomalous rather than the entire event.

One proposal that attempts to predict anomalies is by Rekik et al. [15]. It involves a context-aware system that predicts peak load times using business process duration (KPI) thresholds. To achieve this, it utilizes a decision tree technique on the execution log. To achieve an accurate prediction, it is required that the business process is monitored and logged for at least one

years worth of data. When applied to multiple generated time stamps, the system correctly identified the peak load period and then reduced the number of violated business process instances of a simulated activity. This work, however, does not offer any options for future adaptations.

Focusing on what caused anomalies rather than trying to detect them, Chouchan et al. [7], introduce model-agnostic explanations of process anomalies using linguistic summaries. These anomalies are obtained from the process event logs. The main idea involves isolating anomalous cases and then comparing them to similar normal cases to identify the differences. To measure the similarity, a function of edit distance and length of traces is created. Finally, to generate the summary a truth value is provided to each one. Additionally, there is a threshold value that needs to be reached. Only the summaries with the highest values that are also above the threshold are displayed. The effectiveness of the explanations is evident in the paper, however, as the authors note, there are more aspects that need to be explored, especially in regard to activity levels. Although this paper does not offer any proposal on how to use these findings for further detection purposes, it is highly relevant to our research as it also focuses on extracting information from the logs.

#### 3.1.2 Other existing anomaly detection mechanisms

Besides utilizing the process logs, there are also other proposals that attempt to address the same issue. One such proposal is Lima et al. [9]. The authors present BP-IDS, which is a specification-based detection system. Specification-based systems rely on pre-existing models of acceptable behavior to compare observations against. In this paper, acceptable behavior is modeled as a set of business processes. Using sensors, information on the execution of activities is collected. Then, if any abnormality in the received data is detected the system raises an alarm. Abnormalities are considered as any activities that cannot be attributed to any existing business process. The results showed that BP-IDS is capable of detecting exploits in the software as well as in the business logic. It also proved that it can scale up, regardless of the size and number of activities.

Another approach to anomaly detection is proposed by Sarno et al [17].

In this paper, an ontology-based modelling solution is put forward. The first step in this procedure is capturing the anomalies from the event logs using mining techniques. Then, conformance checking is applied to the anomalies by comparing the ontology graph of standard business processes with the ontology graph of the event logs. Finally, a method called multi-level class association rule learning is used to exploit the collected anomalous data for detection purposes.

The vast majority of proposals that focus on the control-flow perspective, similar to those that have been discussed in this paper so far, originate from the field of business process management and process mining [8]. However, we are specifically interested in anomaly detection in user behavior. For such contributions, a valuable example is the research of Myers et al. [11]. The authors of this paper, conduct anomaly detection focusing on user behavior, by collecting the process logs and constructing a model of expected behavior, before applying a conformance checking activity. Similarly to Myers et al, there is also the research of Alizadeh et al. [3]. In this paper, there is a focus on detecting non-conforming user behavior. This is achieved by comparing the expected behavior of a system with the user behavior that was produced in order to identify deviations.

#### 3.1.3 Log Generation Techniques

A critical part of our research involves the procedure for generating the log files. There have been a few contributions in recent years that propose an approach to this problem. One such example is the paper by Remy et al. [16], which provides valuable insight into the process of log generation. The authors describe the experiences and challenges of creating an event log for a warehouse of a large real-world health system. This served as a useful sample for understanding log generation in the health system domain but also in other areas outside of it.

After reviewing these papers (Table 1), we can conclude that one efficient method for detecting attacks is by applying machine learning techniques to log files. However, in the context of this study, we found that there is no suitable log file available for exploring security vulnerabilities specifically related to user behavior in business processes. This highlights the need for

further research in this area to develop appropriate log generation techniques that consider the unique characteristics of user behavior and its impact on security.

### 3.2 Scientific process

Solutions in scientific workflows appear to be more adaptive in nature. In [20, 21], Wang et al. describe an intrusion-tolerant system, where sub-tasks are executed in parallel across multiple Virtual Machines. All tasks are given a confidence score based on the Lagged Decision Mechanism. This mechanism will wait a specified amount of time in order to collect data from multiple VMs before calculating the confidence score that is shown in the logs. The adaptive aspect stems from its ability to react by preserving intermediate data and subsequently re-executing sub-tasks that received a low confidence score.

The idea of rescheduling anomalous tasks during run-time is further expanded upon in [1], while [22] introduces a workflow management framework that can be applied to a federated cloud. This framework also supports a dynamic rescheduling method. In contrast to the previously discussed literature, [2] proposes a proactive adaptation strategy that predicts future resource load. This enables it to control resource load fluctuation, therefore increasing the accuracy of future failure detection.

Table 1: Summary of mentioned papers.

| Reviewed Papers | | |
|---|---|---|
| Paper | Type of Process | Proposed Method |
| [1] | Scientific | Reschedule tasks that are designated as high-risk |
| [2] | Scientific | Predict future resource load |
| [3] | Business | Conformance checking |
| [7] | Business | Model-agnostic explanations of process anomalies using linguistic summaries |
| [9] | Business | Attempt to attribute activities to any pre-existing model of acceptable behavior |
| [11] | Business | Conformance checking |
| [12] | Business | Extract event logs and construct detection model using autoencoders |
| [13] | Business | Extract event logs reaching the attribute level and construct detection model |
| [15] | Business | Predict peak load times using duration thresholds |
| [17] | Business | Conformance checking is applied to the anomalies by comparing the ontology graph of standard business processes with the ontology graph of the event logs |
| [20] | Scientific | Assign confidence score to all tasks |
| [21] | Scientific | Assign confidence score to all tasks and schedule replicas of sub-tasks |
| [22] | Scientific | Dynamic rescheduling of uncompleted services |

# 4 Implementation

## 4.1 Process handling

The first step in implementing the proposed ideas that were discussed earlier is finding a way to create, modify and deploy processes. For this purpose, the Camunda [5] workflow engine was selected. Camunda provides many useful tools for business processes, but for our purposes, two are the most important. The first tool is Camunda Modeler which allowed us to interact with a BPMN by assigning tasks, monitoring the flow, and adding new behavior to the tasks. An example of this graphical user interface is shown in Figure 2. The second tool is a framework provided by Camunda that can be deployed together with a bpmn and run without any adjustments needed. This framework is built in Java using the Spring Boot application. With these tools, introducing new behavior becomes relatively straightforward, as we can create a new Java class in our framework that produces the desired behavior and then add the reference to that class in the Modeler.

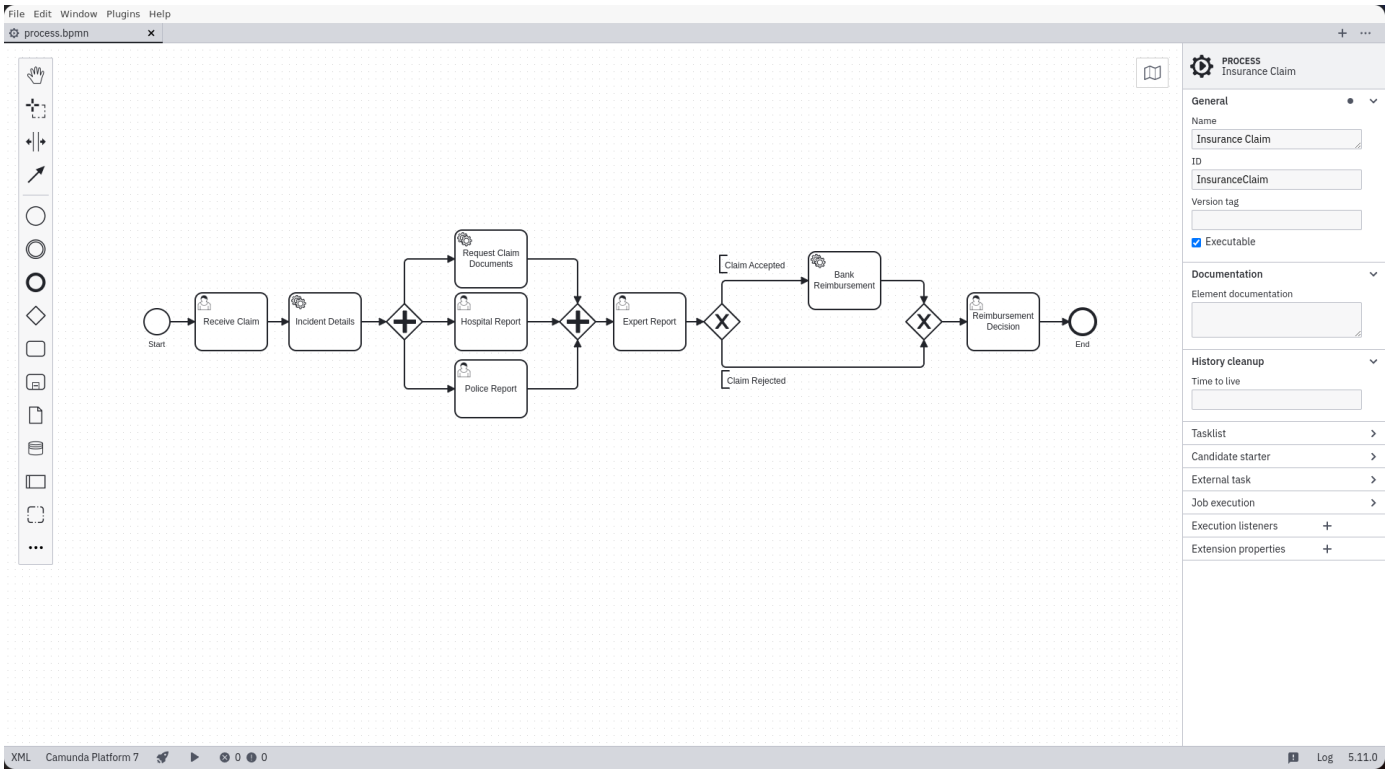


Figure 2: Camunda Modeler GUI

## 4.2 Simulation

### 4.2.1 Legitimate behavior

As mentioned before, in this project, our focus is on user tasks, where a user must manually claim and complete the task, resulting in slow throughput time. However, to generate sufficiently comprehensive event logs, a significant number of executions is required. To solve this issue, we decided to simulate user tasks using service tasks. Essentially, the user tasks were converted to service tasks linked to a Java class that would replicate the behavior of a legitimate user. By applying this simulation, the process execution is now automated and can be performed thousands of times in a few seconds without the need for human intervention.

### 4.2.2 Malicious behavior

Since the user behavior was simulated, a similar approach was chosen for the malicious behavior. The types of attacks analyzed in Section 2 have some unique characteristics that indicate their occurrence. Therefore, we created some additional classes in our Java project to replicate each attack. Thresholds were set for each attack to decrease false positives.

For the DDoS attack, the deciding factors are the IP location of active users as well as any unexpected sharp increases in incoming traffic. This is because such traffic fluctuations, particularly from many different addresses, could indicate someone is attempting to flood the network.

Regarding the probe attack, we considered the duration of a user's task and the number of calls made. If the duration is significantly shorter than expected, it could be a hint that the user assigned to the task is not performing it as expected and is instead quickly gathering information about the system's responses. This information can be used for other malicious purposes.

For the R2L attack, the most important indicators are the user's IP location and their authentication status. Using these metrics, we can identify if someone is accessing the system from addresses outside the expected network or attempting to create a new user remotely.

Finally, for the U2R attack, we assessed whether an existing user has root privileges by checking if they are an admin. If a non-administrator user attempts to perform an admin-level action or escalate the privileges of a regular user account to that of an admin, it is deemed suspicious.

Additionally, we set equal weights for each attack and then randomly

selected one attack during execution. The data generated by the attacks are mixed with the data from legitimate users.

#### 4.2.3 Reusability

The simulated behavior is triggered whenever the associated task is executed. This is achieved by adding a listener to the user tasks[1], as shown in Figure 3. These listeners are the main connection between the Java classes and the tasks, requiring minimal adjustment to the Camunda platform. In order to adapt the code to different processes and tools, only the basic framework needs to change. Then, the classes that replicate the legitimate and malicious behavior can be added as they are. Consequently, we believe that our approach offers great reusability capabilities.

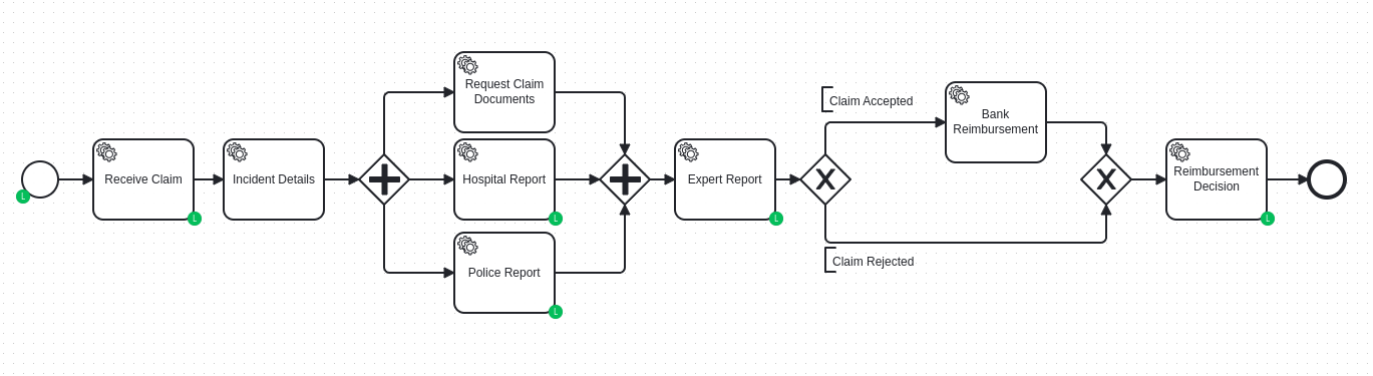


Figure 3: An example of a BPMN with simulated user tasks and execution listeners

### 4.3 Collected event logs

After achieving the desired simulation, the next step was logging all the relevant execution data. In order to log information, we used the Simple Logging Facade for Java (SLF4J) and Logback. For the user tasks, it is important to monitor data that describe the actions taken by users and the corresponding context. Therefore, we identified the following fields to be included in our logs:: "Instance ID", "Process ID", "Task Name", "Assignee", "Used IP", "Day", "Duration", "Number of valid", "Number of invalid", "Number of calls", "Activity label" and "Label justification". This is not an exhaustive list of useful metrics that can be logged, but only a subset that was selected for this project. Adding or removing fields from the logs is a simple procedure of generating the data and then calling the log

[1]The user tasks are simulated as service tasks in the BPMN.

function, as all the logging is done in the same class that handles user task triggers. The data assigned to these fields is simulated and logged in the listener class that is executed together with the user tasks. The last two fields are the conclusion of each log entry. "Activity label", will be assigned either a value of 'Normal' or one of the different types of attacks ('DDoS', 'Probe', 'R2L', 'U2R'). In order to reach a decision, all the previous data points for the specific task are considered against a model signature for each of the attacks. Then, the "Label justification" will provide the reasoning behind this labeling.

Expanding on the previous description of the log fields, they are a combination of real process metrics and synthetic data. "Instance ID", "Process ID" and "Task Name" are extracted from the process using the JavaDelegate class offered by Camunda. "Assignee" is produced by randomly selecting a name from a list using a random generator class. "Used IP", "Day", "Duration", "Number of valid", "Number of invalid" and "Number of calls" follow a similar randomization approach, but each with its own restrictions. "Used IP" follows the IPv4 standard and its network part is restricted to addresses between 100 and 256 to create a range of allowed addresses and a range of foreign addresses. "Day" is obtained by randomly indexing a list of seven entries, each for one day of the week. "Duration" is considered in minutes and has a range of values between 0 and 1000. "Number of valid" has a range between 0 and 100, "Number of invalid" between 0 and 10, and "Number of calls" also has the same range between 0 and 10. The "Activity label" and the "Label justification" are produced last after considering the previous data that have been generated. Their possible values will be discussed in the next section. Finally, we executed the entire process multiple times to produce a sufficient number of entries and then wrote all logged data to an event log file named 'process.log', which was saved in the directory of the Java project.

Table 2: The log fields and their descriptions.

| Log fields | |
|---|---|
| Field | Description |
| Instance ID | The random ID that was generated for the process instance |
| Process ID | The name that was given to the process |
| Task Name | The name of the executed user task |
| Assignee | The name of the user that executed the user task |
| Used IP | The IP address of the user that executed the user task |
| Day | The day of the week that the user task was executed |
| Duration | The number of minutes it took for the user to complete the user task |
| Number of valid | The number of valid calls made by the user |
| Number of invalid | The number of invalid calls made by the user |
| Number of calls | The total number of calls made by the user |
| Activity label | The type of activity that was indicated |
| Label justification | The reason behind the decision on the type of activity |

## 5 Results

When the Java project is running, the BPMN file that is included in the directory is executed and the 'process.log' file is created. This file is considered the process event log file. It contains lines of log entries along with their label

descriptions. The number of lines in the file is decided by the value that we set in the loop that will execute the process. A sample from an event log file that was generated by executing an insurance claim procedure (Figure 1) can be seen in Figure 4 below. In this example, the number of executions was set to 50,000.

```
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9b7c542-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Larry Irwin, 119.235.109.250, Sun, 20, 61, 5, 0, Normal, Expected behavior
b9b7c542-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Alex Orwig, 209.67.64.13, Tue, 737, 15, 9, 0, Normal, Expected behavior
b9b7c542-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Victor Schickowski, 116.116.122.146, Tue, 474, 17, 7, 1, Normal, Expected behavior
b9b7c542-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Monte Ory, 118.42.39.15, Mon, 597, 81, 5, 5, Normal, Expected behavior
b9b7c542-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Thomas Ferry, 169.155.49.89, Sun, 473, 14, 8, 1, Normal, Expected behavior
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9b8399b-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Larry Ulrich, 130.234.135.90, Sat, 278, 97, 4, 3, Normal, Expected behavior
b9b8399b-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Walter Chambers, 167.9.175.230, Sat, 140, 80, 7, 8, Normal, Expected behavior
b9b8399b-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Fred Schickowski, 227.64.42.52, Tue, 213, 90, 8, 8, U2R, User does not have admin authorization
b9b8399b-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Victor Caswell, 135.179.38.164, Wed, 941, 71, 0, 9, Normal, Expected behavior
b9b8399b-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Victor Ebner, 173.215.194.150, Sat, 20, 77, 9, 7, DDoS, Sharp increase in network traffic
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9b8aef7-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Victor Linde, 250.73.162.2, Sun, 423, 48, 4, 7, Normal, Expected behavior
b9b8aef7-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Monte Wagner, 239.46.187.51, Mon, 394, 36, 7, 7, Probe, Too many calls
b9b8aef7-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Larry Schiebel, 145.140.241.110, Sat, 102, 73, 9, 3, Normal, Expected behavior
b9b8aef7-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Steve Eastman, 102.60.184.40, Wed, 125, 50, 4, 9, R2L, Unusual time of activity
b9b8aef7-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Matthew Tapia, 102.195.63.178, Wed, 629, 1, 3, 6, DDoS, IP addresses coming from outside authorised network
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9b92450-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Adam Woo, 249.133.69.183, Thu, 552, 77, 1, 7, R2L, Unusual user location
b9b92450-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, John Bongard, 132.48.1.29, Sun, 34, 45, 9, 7, R2L, Unusual user location
b9b92450-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Thomas Reardon, 253.177.69.143, Wed, 844, 98, 0, 4, R2L, Unusual time of activity
b9b92450-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Dan Valente, 213.58.25.147, Fri, 991, 27, 1, 1, R2L, Unusual time of activity
b9b92450-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Ike Dinkins, 254.107.176.44, Mon, 256, 97, 2, 1, Normal, Expected behavior
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9b999a9-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Paul Jagtap, 252.74.110.12, Sun, 299, 49, 8, 9, Probe, Too many invalid calls
b9b999a9-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Alex Boyd, 210.114.125.141, Wed, 253, 69, 3, 6, Normal, Expected behavior
b9b999a9-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Alex Ferry, 162.154.248.63, Wed, 540, 31, 8, 7, Normal, Expected behavior
b9b999a9-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Fred Linde, 105.128.215.114, Tue, 445, 89, 5, 0, Normal, Expected behavior
b9b999a9-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Joe Wakefield, 181.194.201.17, Wed, 67, 31, 2, 2, Normal, Expected behavior
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9ba0f02-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Joe Paiser, 153.223.82.89, Mon, 416, 16, 1, 1, Normal, Expected behavior
b9ba0f02-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, George Miller, 201.162.162.233, Thu, 199, 68, 7, 3, Probe, Very short connection duration
b9ba0f02-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Thomas Christensen, 229.141.42.178, Sat, 934, 81, 6, 2, Normal, Expected behavior
b9ba0f02-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Hank Nelson, 235.93.36.225, Sat, 275, 0, 5, 4, Normal, Expected behavior
b9ba0f02-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Aaron Hancock, 132.183.103.47, Wed, 974, 61, 4, 4, Normal, Expected behavior
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9ba845e-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Otto Weinstein, 166.93.153.212, Wed, 143, 14, 3, 0, U2R, Suspicious privilege escalation
b9ba845e-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Roger Ziegler, 245.248.102.116, Sat, 831, 2, 8, 4, R2L, Unusual user location
b9ba845e-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Mark Ziegler, 246.170.180.60, Mon, 644, 92, 7, 9, R2L, Unusual time of activity
b9ba845e-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Ben Schwager, 105.219.147.43, Thu, 59, 50, 7, 6, Probe, Too many calls
b9ba845e-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Dan Norquist, 130.116.237.154, Mon, 303, 10, 0, 8, DDoS, IP addresses coming from outside authorised network
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9baf8ba-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Roger Mills, 239.126.73.232, Tue, 210, 50, 8, 7, DDoS, IP addresses coming from outside authorised network
b9baf8ba-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Victor Sagar, 133.145.19.224, Fri, 768, 30, 0, 4, Normal, Expected behavior
b9baf8ba-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, John Baxster, 175.142.74.53, Wed, 24, 76, 2, 2, Normal, Expected behavior
b9baf8ba-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Hank Lewis, 128.69.25.53, Mon, 82, 19, 8, 2, R2L, Unusual time of activity
b9baf8ba-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Peter Schickowski, 105.26.115.87, Fri, 953, 61, 1, 9, R2L, Unusual user location
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
b9bb6e13-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Alex Dugelman, 194.158.205.37, Wed, 617, 18, 6, 2, Normal, Expected behavior
b9bb6e13-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Peter Dinkins, 211.59.22.24, Mon, 595, 19, 3, 4, R2L, Unusual user location
b9bb6e13-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Carl Ferro, 140.194.92.59, Sat, 611, 33, 1, 3, Normal, Expected behavior
b9bb6e13-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, John Schuster, 188.170.171.60, Tue, 100, 36, 5, 3, Normal, Expected behavior
```

Figure 4: A sample from process.log

All log files generated by our program will have the same structure as the insurance claim log file of Figure 4. Using the insurance claim process (Figure 1) as our example, it comprises of 5 user tasks, resulting in 6 line entries for each completed process execution when combined with the label descriptions. Considering a loop of 50,000 iterations, there are 250,000 entries each of which describes a task execution using the fields outlined in Table 2. With this information, we can assess the frequency of each simulated attack and the factors influencing it.

In order to obtain more accurate results, we ran the program 5 times and averaged the occurrences for each of the attacks. We considered a total of 250,000 instances for further evaluation. In the case of the DDoS attack, the log files contained 22,105.8 instances (≈8.84%). Among these, 15,256 (≈6.1%) instances were due to "IP addresses coming from the outside authorized network" and 6,849.8 (≈2.74%) instances were caused by a "Sharp increase in network traffic". Probe attacks occurred on average 19,666.8 times (≈7.87%). 3,768.2 (≈1.51%) were attributed to a "Very short connection duration", 5,229.8 (≈2.09%) to "Too many calls" and 10,668.8 (≈4.27%) to "Too many invalid calls". The number of R2L attacks was 42,594.8 (≈17.04%). This included 14,302 (≈5.72%) for "Unusual time of activity", 23,773.8 (≈9.51%) for "Unusual user location" and 4,519 (≈1.81%) for an "Attempt to create new unauthorized user". Finally, U2R attacks occurred an average of 19,869.2 times (≈7.95%), of which, 10,444.2 (≈4.18%) were due to "User does not have admin authentication", while the other 9,425 (≈3.77%) due to "Suspicious privilege escalation". Below are example instances of each of the attacks in the log files. The relevant data for each Figure is highlighted.

```
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
8c8248df-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Ben Tapia, 178.116.181.8, Sun, 253, 46, 8, 6, Probe, Too many invalid calls
8c8248df-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Monte Reyes, 154.134.124.183, Thu, 997, 92, 8, 4, DDoS, Sharp increase in network traffic
8c8248df-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Otto Kalleg, 176.72.93.158, Wed, 196, 85, 4, 3, Normal, Expected behavior
8c8248df-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Nathan Vanderpoel, 210.128.121.102, Sun, 760, 73, 6, 8, Normal, Expected behavior
8c8248df-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Aaron Wagle, 237.84.154.80, Tue, 609, 68, 4, 1, Normal, Expected behavior
```

Figure 5: An instance of a DDoS attack

```
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
8ce1a69f-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Peter Ulrich, 139.148.18.233, Thu, 87, 18, 6, 8, Probe, Very short connection duration
8ce1a69f-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Nathan Soloman, 167.44.138.190, Thu, 653, 8, 4, 2, DDoS, Sharp increase in network traffic
8ce1a69f-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Alex Schickowski, 167.64.167.204, Thu, 466, 48, 9, 5, Normal, Expected behavior
8ce1a69f-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Ben Sagar, 108.118.55.202, Sun, 463, 0, 7, 2, Normal, Expected behavior
8ce1a69f-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Tim Johnson, 222.253.238.200, Sun, 216, 47, 1, 7, U2R, Suspicious privilege escalation
```

Figure 6: An instance of a Probe attack

```
"Instance ID" "Process ID" "Task Name" "Assignee" "Used IP" "Day" "Duration" "Number of valid" "Number of invalid" "Number of calls" "Activity label" "Label justification"
8dcf891c-0f51-11ee-b933-02426fd28150, Insurance Claim, Receive Claim, Frank Quizoz, 238.15.117.59, Sun, 136, 11, 9, 4, R2L, Unusual user location
8dcf891c-0f51-11ee-b933-02426fd28150, Insurance Claim, Hospital Report, Joe Caswell, 196.111.70.73, Sat, 322, 78, 7, 1, DDoS, Sharp increase in network traffic
8dcf891c-0f51-11ee-b933-02426fd28150, Insurance Claim, Police Report, Alex McCormack, 136.54.61.218, Fri, 16, 50, 3, 8, Normal, Expected behavior
8dcf891c-0f51-11ee-b933-02426fd28150, Insurance Claim, Expert Report, Larry Schwager, 181.82.167.188, Thu, 644, 69, 5, 0, DDoS, Sharp increase in network traffic
8dcf891c-0f51-11ee-b933-02426fd28150, Insurance Claim, Reimbursement Decision, Roger Moore, 212.124.74.56, Thu, 486, 69, 7, 6, Normal, Expected behavior
```

Figure 7: An instance of a R2L attack

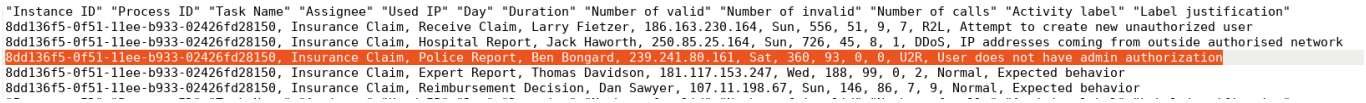


Figure 8: An instance of a U2R attack

| | DDoS | Probe | R2L | U2R |
|---|---|---|---|---|
| Overall | 22,105.8 (~8.84%) | 19,666.8 (~7.87%) | 42,594.8 (~17.04%) | 19,869.2 (~7.95%) |
| IP addresses coming from outside authorized network | 15,256 (~6.1%) | | | |
| Sharp increase in network traffic | 6,849.8 (~2.74%) | | | |
| Very short connection duration | | 3,768.2 (~1.51%) | | |
| Too many calls | | 5,229.8 (~2.09%) | | |
| Too many invalid calls | | 10,668.8 (~4.27%) | | |
| Unusual time of activity | | | 14,302 (~5.72%) | |
| Unusual user location | | | 23,773.8 (~9.51%) | |
| Attempt to create new unauthorized user | | | 4,519 (~1.81%) | |
| User does not have admin authentication | | | | 10,444.2 (~4.18%) |
| Suspicious privilege escalation | | | | 9,425 (~3.77%) |

Figure 9: Summary of attack occurrences in the log files

## 6 Future Work

The produced log files can be improved in the future in multiple ways. Firstly, the authenticity of the log fields that were randomly generated can be increased to better reflect a real-life example. This may involve incorporating a wider range of candidate users, more accurate IP addresses, and more flexible task duration values. Secondly, the log file can be extended to include more fields that offer new insight into the details of the process execution. This way, thereby enhancing the accuracy of the activity labels' decisions. Thirdly, the current log files are restricted to the four types of attacks that were discussed and then to a few possible cases for each of them. Therefore, in the future new types of attacks can be introduced as well as new possible ways for these attacks to manifest. This will increase the coverage of the log files and will improve their descriptive capabilities.

As for the Java application, some contributions can benefit the research. The current results are mostly based on the insurance claim process of Figure 1. Therefore, it will be useful to gather more processes with varying

complexity and deploy them using our framework to enhance the external validity of the logs. For this purpose, we can generate random BPMN models using tools such as PLG2 [4] and then include the generated file in the resources of our project in the same way as the insurance claim process was used. Additionally, the Java code can be refactored in order to have a better file and directory structure as well as more abstraction to our implementation.

This project is a component of a larger project in the Information Systems group of the University of Groningen. Parallel to this research, log files were also generated for the service tasks of a business process, following similar security principles. Therefore, the next step can be merging the results of these two projects into one log file that covers all possible tasks.

Finally, the complete log files for both user tasks and service tasks can be used for security and detection processes using machine learning. Specifically, machine learning algorithms can be applied to the log files to construct a model that will be able to detect incoming attacks or intrusions. The detection rate can either be periodical or ideally in real-time. This model can then be used by businesses to protect their processes against malicious individuals or groups.

# 7 Conclusion

The objective of this project was to simulate user tasks in business processes accurately, incorporating various types of attacks. The motivation behind these simulations was to provide a clear event description in the event of different attacks, enabling the use of the generated data for detection purposes through machine learning.

The generated log of the project shows effective recognition and analysis of different types of attacks, closely resembling real process execution results. Additionally, the automation of user tasks through simulation has provided the opportunity to generate vast amounts of log entries. Therefore, the accuracy of a future machine learning model that will be trained on these files increases.

While the results are promising, there is room for improvement, as the current simulation is relatively basic. Since it is not in the scope of this project, the authenticity of the data produced by the tasks for business purposes has not been extensively explored. Furthermore, the signatures and characteristics of the attacks that were covered are often much more complex and difficult to identify in the real world than were portrayed in these log files.

Despite these limitations, the findings of this project, in their current form, are already valuable for research purposes and hold promising potential for achieving satisfactory detection rates with machine-learning models. However, further research is necessary due to the increasing complexity and evolving nature of the attacks targeted by the project.

## References


[1] Farzaneh Abazari, Morteza Analoui, Hassan Takabi, and Song Fu. Mows: Multi-objective workflow scheduling in cloud computing based on heuristic algorithm. *Simulation Modelling Practice and Theory*, 93:119–132, 2019. Modeling and Simulation of Cloud Computing and Big Data.

[2] Mani Alaei, Reihaneh Khorsand, and Mohammadreza Ramezanpour. An adaptive fault detector strategy for scientific workflow scheduling based on improved differential evolution algorithm in cloud. *Applied Soft Computing*, 99:106895, 2021.

[3] Mahdi Alizadeh, Xixi Lu, Dirk Fahland, Nicola Zannone, and Wil MP van der Aalst. Linking data and process perspectives for conformance analysis. *Computers & Security*, 73:172–193, 2018.

[4] Andrea Burattin. Plg2: Multiperspective process randomization with online and offline simulations. In *BPM (Demos)*, pages 1–6. Citeseer, 2016.

[5] Camunda. Camunda BPM. `https://camunda.com/`. Accessed: 2023-07-17, 2023. Version 7.19.0.

[6] Zouhair Chiba, Noureddine Abghour, Khalid Moussaid, Amina El Omri, and Mohamed Rida. A survey of intrusion detection systems for cloud computing environment. In *2016 International Conference on Engineering & MIS (ICEMIS)*, pages 1–13, 2016.

[7] Sudhanshu Chouhan, Anna Wilbik, and Remco Dijkman. Explanation of anomalies in business process event logs with linguistic summaries. In *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–7, 2022.

[8] Jonghyeon Ko and Marco Comuzzi. A systematic review of anomaly detection for business process event logs. *Business & Information Systems Engineering*, pages 1–22, 2023.

[9] João Lima, Filipe Apolinário, Nelson Escravana, and Carlos Ribeiro. Bp-ids: Using business process specification to leverage intrusion detection in critical infrastructures. In *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 7–12, 2020.

[10] Business Process Model. Notation (bpmn) version 2.0. *OMG Specification, Object Management Group*, 19:52–60, 2011.

[11] David Myers, Suriadi Suriadi, Kenneth Radke, and Ernest Foo. Anomaly detection for industrial control systems using process mining. *Computers & Security*, 78:103–125, 2018.

[12] Timo Nolle, Stefan Luettgen, Alexander Seeliger, and Max Mühlhäuser. Analyzing business process anomalies using autoencoders. *Mach Learn*, 107:1875–1893, 2018.

[13] Timo Nolle, Alexander Seeliger, and Max Mühlhäuser. Binet: Multivariate business process anomaly detection using deep learning. In *Business Process Management*, pages 271–287, Cham, 2018. Springer International Publishing.

[14] Swati Paliwal and Ravindra Gupta. Denial-of-service, probing & remote to user (r2l) attack detection using genetic algorithm. *International Journal of Computer Applications*, 60(19):57–62, 2012.

[15] Mouna Rekik, Khouloud Boukadi, and Hanêne Ben-Abdallah. Towards an autonomic outsourcing to the cloud decision. In *2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 20–25, 2016.

[16] Simon Remy, Luise Pufahl, Jan Philipp Sachs, Erwin Böttinger, and Mathias Weske. Event log generation in a health system: a case study. In *Business Process Management: 18th International Conference, BPM 2020, Seville, Spain, September 13–18, 2020, Proceedings 18*, pages 505–522. Springer, 2020.

[17] Riyanarto Sarno and Fernandes P. Sinaga. Business process anomaly detection using ontology-based process modelling and multi-level class association rule learning. In *2015 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, pages 12–17, 2015.

[18] Vitaly Shmatikov and Ming-Hsiu Wang. Security against probe-response attacks incollaborative intrusion detection. *Association for Computing Machinery*, pages 129–136, 08 2007.

[19] Nikhil Tripathi and Babu Mehtre. Dos and ddos attacks: Impact, analysis and countermeasures. In *2nd International Conference on Advanced Computing, Networking and Security (ADCONS '13)*, pages 1–6, 12 2013.

[20] Yawen Wang, Yunfei Guo, Zehua Guo, Wenyan Liu, and Chao Yang. Protecting scientific workflows in clouds with an intrusion tolerant system. *IET Information Security*, 14(2):157–165, 2020.

[21] Yawen Wang, Yunfei Guo, Wenbo Wang, Hao Liang, and Shumin Huo. Inhibitor: An intrusion tolerant scheduling algorithm in cloud-based scientific workflow system. *Future Generation Computer Systems*, 114:272–284, 2021.

[22] Zhenyu Wen, Rawaa Qasha, Zequn Li, Rajiv Ranjan, Paul Watson, and Alexander Romanovsky. Dynamically partitioning workflow over federated clouds for optimising the monetary cost and handling run-time failures. *IEEE Transactions on Cloud Computing*, 8(4):1093–1107, 2020.