RESEARCH ARTICLE OPEN ACCESS

Smart Platform for Investigation of Log Files

Mohamed Ayari
Faculty of Computing and Information Technology, Northern Border University –Kingdom of Saudi Arabia; Syscom Laboratory, National Engineering School of Tunis, University of Tunis El-Manar, Tunisia.

ABSTRACT
The Internet is the land of a new crime that is growing more and more in recent years. Some have not lost time to exploit the many flaws of the Web for malicious purposes. And thus, to protect themselves from these attacks circulating via e-mail or on the Internet, most companies and organizations have adopted strict security policies and have continued to invest in their update. In this context, we will develop and implement an investigative smart platform for Web log files. This platform should enable more of the investigation as identifying the types of detected attacks and cyber criminals and for all legal proceedings and offering recommendations and possibilities to prevent every detected attack. The aim is to raise awareness of ways to protect victims.

Keywords - Investigation; smart platform; Web Log Files; Cybercrimes

Date of Submission: 09-04-2022 Date of Acceptance: 26-04-2022

I. INTRODUCTION

The investigative platform for Web log files is considered as an efficient way to automate incident analysis to identify cyber criminals and for possible legal action. This is primarily to determine the possible scenarios of an incident and model data attacks on several fronts. This platform should obviously ensure a cooperative organizational structure and consider the following main investigation phases:

1- Introduction and treatment: During this phase a user authenticates through a Web interface to provide some information and evidence in relation to the incident report he wishes. Among the items of evidence, we find the log files such as access-log web server, ssh logs, VPN logs, etc.

After collecting the information, we can identify different scenarios and assumption in relation to the incident.

2- Analysis and validation: To perform this phase, the investigator must manually intervene to validate the results of automatic processing. His work is mainly to eliminate the scenarios presenting false positives.

3- Report writing and publication: This step is to produce a detailed report that shows the different test results and any assumptions with respect to the incident.

Methodology

The development method could be a series of steps that control the project's progress and giving users better visibility of some goods results and thus guarantee the suitability of the result. Each step of the strategy is punctuated by creating a model of the system studied. A model is an abstract representation in an exceedingly certain formalism of reality that excludes certain details of the important world.

To choose the method of development associated with our project we reviewed the possible methods.

Several development processes exist and offer different approaches. A comparative study between these approaches are going to be appreciated within the subject of the method and technology community.

In [1], the authors deal with a vital and important issue in computer world. It is concerned with the software management processes that examine the area of software development through the development models, which are known as software development life cycle. The have presented five of the development models namely, waterfall, Iteration, V-shaped, spiral and Extreme programming. These models have advantages and disadvantages as well.

Therefore, the main objective of this research is to represent different models of software development and make a comparison between them to show the features and defects of each model. Despite its weaknesses, we believe that V cycle [2] will be the most appropriate method to our project.

The principle of this model, as depicted in figure 1.1, is that with any decomposition must be described redial and any description of a component is accompanied by tests to make sure it matches the description.

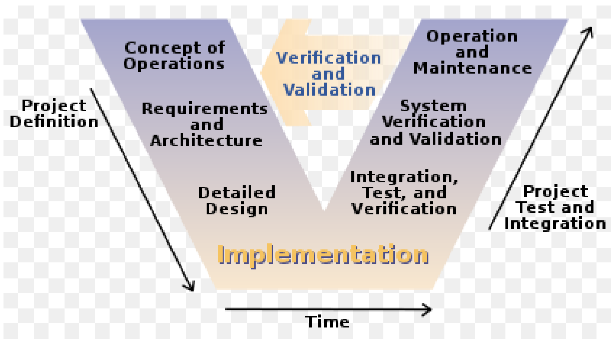
Concept of Operations
Verification and Validation
Operation and Maintenance
Project Definition
Requirements and Architecture
System Verification and Validation
Integration, Test, and Verification
Detailed Design
Project Test and Integration
Implementation
Time

Figure1.1. The "V" Model [3]

This type of organization makes it possible to prepare the development of the upstream stages and move forward on each step without it requires a lot of client-side investments. This makes final stages of preparation explicit (validation, verification) by the first (software construction) and so avoids stating a property it is impossible to verify objectively after completion.

Currently there are two main approaches in the design of information systems: the functional approach and the object-oriented approach [4].

The functional approach: Collects the system as an entity carrying out a decomposable global function into sub-functions.

The object approach: It grasps the system as a set of objects that interact.

In our study we have opted for the approach object that provides software evolution and reuse of objects, which is not guaranteed with the functional approach. Therefore, we must also choose a modeling language and a development language that is adequate compared this approach.

Web Attacks [5-6]

To someone eager to force an entry your site, it's imperative they find some extent of entry they will exploit (this is understood as an attack vector). These attack vectors are available a spread of forms, today you regularly attribute these to two main categories: Access Control and Software Vulnerabilities.

In addition, the most important software vulnerabilities are:

SQL Injection (SQLi)

Injection vulnerabilities [10] are rated as the number one problem on the list of top 10 security issues put out by Open Web Application Security Project (OWASP) and continue to be a major source of concern for application and web developers looking to utilize the benefits of storing usable information in a local database. Due to the predictable nature of these types of applications, an attacker can craft a string using specific Structured Query Language (SQL) commands and know it can be used to force the database to give up the goods. These strings can be entered in places like search boxes, login forms, and even directly into a URL to negate simple client-side security measures on the page itself.

Why is this so dangerous? The database keeps the most important and lucrative space on a system, and can not only be coaxed to give up usernames, passwords, and credit card numbers, but can also be attacked in a way that can give an attacker a foothold to gain access to the entire system, and to every other database instance housed there for other websites and applications.

A real-world example of the severity of this issue can be seen in the recent Drupal security release in which the platforms core was vulnerable to a SQLi vulnerability.

Cross-Site Scripting (XSS)

Often misunderstood, and even more often underestimated, XSS is a style of attack where the front of the website acts as a launching point for attacks on other users visiting the website. This happens when developers don't properly test their code for the possibility of allowing scripts to be injected. The scripts can then be executed without the site's original functionality intending them to be.

If an XSS vulnerability is present on a website, then an attacker can craft code that executes when other users open the same website. This causes the new users to interact with the malicious background entity created by the attacker. Once a connection has been initiated, usually via social-engineering tactics convincing a user to do something they shouldn't, the attacker is able to infiltrate your website visitors' computers.

Inclusion Vulnerabilities: LFI and RFI

As a result of insecure coding, malicious users can find functionality within a web application, and use the underlying mechanics to execute their code. The two variations of this action can be to either execute code already on the system or execute code that is located off the system.

Local File Inclusion (RFI)

By targeting 'include' parameters in PHP code, intruders can request an alternative file be used in the specified request instead of the file meant to go along with the program. This can lead to unintended access to internal files and logs. Where a script should work like this – http://site.com/web-app.php?nextStep=goodfile.php – a vulnerable application can be changed to target a sensitive system file, or worse, something that is infected – http://site.com/web-app.php?nextstep=/etc/passwd

Where this can get even messier is when dealing with a highly sophisticated intruder that knows how to manipulate internal files. By sending malicious payloads to the site, without intending for them to work, a hacker can load log files with their own code. By pointing a vulnerable include parameter to a code injected log file by using an LFI technique, a devastating attack can be launched.

Remote File Inclusion (RFI)

A very sneaky method of running malicious software on a victim's server is by simply asking it to go somewhere else on the Internet to find a dangerous script, and then run it from that location. This scary scenario is called a Remote File Inclusion (RFI) attack. An RFI can occur when functions are improperly crafted, allowing users to modify the URL parameters when web apps are launching components for their own purposes.

By changing the intended process in order to activate a faraway malicious payload sitting on a public server, the attacker may be able to activate a piece of code that will give them a shell through a held connection between the victim site and the remote server that holds the designated file. Including a script in this way opens several dangerous options that a hacker can use against you.

Log Files

Current software application often produces (or can be configured to produce) some auxiliary text files known as log files. Such files are used during various stages of software development, mainly for debugging and profiling purposes. Use of log files helps testing by making debugging easier. It allows to follow the logic of the program, at high level, without having to run it in debug mode.

Nowadays, log files are commonly used also at customers installations for the purpose of permanent software monitoring and/or ne-tuning. Log files became a standard part of large application and are essential in operating systems, computer networks and distributed systems. Log files are often the only way how to identify and locate an error in software, because log file analysis is not affected by any time-based issues known as probe effect. This is an opposite to an analysis of a running program, when the analytical process can interfere with time-critical or resource-critical conditions within the analyzed program as shown in figure1.2.

Log files are often very large and can have complex structure. Although the process of generating log files is quite simple and straightforward, log file analysis could be a tremendous task that requires enormous computational resources, long time and sophisticated procedures. This often leads to a common situation, when log files are continuously generated and occupy valuable space on storage devices, but nobody uses them and utilizes enclosed information.

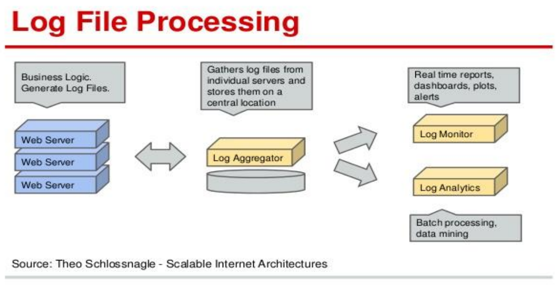
Log File Processing
Business Logic. Generate Log Files.
Gathers log files from individual servers and stores them on a central location
Real time reports, dashboards, plots, alerts
Web Server
Web Server
Web Server
Log Aggregator
Log Monitor
Log Analytics
Batch processing, data mining
Source: Theo Schlossnagle - Scalable Internet Architectures

Figure1.2. Log File Processing

Server Log File

Server traffic logs are files generated by the server in order to provide data about requests to the server for data. When a computer connects to a site, the computer, browser, and network will deliver some data to the site's server itself to create a record that a file was requested.

Of all the information in the log entry, only the time & date, HTTP Request, and the response information should be regarded as accurate. The IP address, referrer, and user-agent string should be regarded as unreliable, as it can be faked in some way by the user. For example, Netscape 8 gives the user the option to identify itself as Internet Explorer during the setup process and many other browsers supply this ability in their "Options" or "Preferences" menus.

Analysis Tools

Many organizations use an analysis program to parse server log files so that they're much easier to understand. Imagine trying to cull anything of value from a huge text file of entries like the one above when hundreds of thousands (or even millions) of entries are present in the log file! Essentially, the analysis tool treats the log file as a flat-file database, processes it, and generates the "statistics" that are discussed throughout the rest of this commentary. In other words, "Web Analytics" software does little more than provide its own interpretation on data contained in the log file which, as has been stated above, could be at least partially faked.

Problem Statement

The analyzers log files provide only statistics and frequencies. In this case, the problem is as follows:

Which is the manager of an attack? How to identify a cybercriminal? And yet how to determine the type of attack and the possible scenarios of the incident?

Manual analysis seems to be inevitable in order to answer all these questions. But at what price? Did it rather automate this manual analysis and take full advantage of a log file information?

Proposed solution

The realization of an investigative platform of log files is probably the solution. The investigation itself is a very important step is to check and analyze the collected data evidence to lead to judicial evidence affirming or denying the occurrence of a computer crime and it determines the type of attack and the possible scenarios and starting traces found (evidence in the log files). Conducting a proper investigation platform log files is thus indispensable

II. INVESTIGATIVE PLATFORM

At the beginning from any application development process, it is imperative to start by listing the functional requirements that must be addressed. This phase is essential to determine exactly what is required. Indeed, it is the construction of an abstract computer representation of a set of user requirements. In what follows, we show the different specific requirements of the referred investigation platform.

II.1 Objective

The aim of our project is to realize an investigative platform that allows log files to investigate to identify cyber criminals and for possible legal action, starting trace (evidence or logs) required.
To achieve this overall objective, we must address the following sub-objectives:
Automating the analysis of the investigation: replace the manual investigation procedure by allowing automatic analysis speed in terms of time and efficiency in terms of results.
Identify cyber criminals: determine those responsible for attacks to all legal proceedings.
Identify possible scenarios of the incident.
Data Modeling attacks on several fronts: determine some information about the attack as the IP address of the attacker, the date, the type of attack, and so on.

II.2 Study of the existing

The security team of any organization is responsible among other things to investigate multiple log files from different Web servers. These files are submitted by their clients to this team conducting their investigation manual and based on expertise of its members and their visual capabilities to award various incidents in the log files. This method is without doubt not suitable to analyze large amounts of information. The use of an automated analyzer log files is then adopted by this team in order to have a more comprehensive and correct vision. However, the role of an analyzer is limited to the creation of a report statistics without deeply investigating the file and identify the types of attacks and cyber criminals.

Thus, and from results by automatically analyzing the security team still makes a deeper and more complicated manual analysis to identify attacks and potential cyber criminals. This manual analysis is time consuming and does not guarantee good results because of the size of the log file. The realization of a fully automated investigative platform is thus inevitable.

II.3 Requirements Specifications

This section contains details about all the specific requirements that will be enough for designers to create a system to satisfy those requirements and for testers to test the given. If there is any one thing any project must have in order not to be doomed to failure, that is a sensible and comprehensive collection of both the functional and non-functional requirements. Any project’s requirements need to be well thought out, balanced and clearly understood by all involved, but perhaps of most importance is that they are not dropped or compromised halfway through the project.

The Functional Requirements, as collected from the users, have been categorized to support the types of user interactions the system shall have. Typically, functional requirements will specify a behavior or function,”
Identification of actors:
About our system, there are three principal actors:

i. Administrator:

The user can access all the features of the application (consultation and configuration). It corresponds to a person who has full authority. So, it can:
- Login
- Manage investigators (members of security team are investigating the log files).
- Manage customers (who deposit the log files from organization for a full analysis revealing any risk or vulnerability).
- Manage the dictionary attack (reference dictionary used to identify existing attacks).
- Process log files (the administrator can play the role of an investigator)
- Manage log files (Filter and affect log files to an investigator)
- Track the status of the said incident analysis (pending, in process or completed)
- Viewing Statistics (Statistical reports written / investigators statistics).

The administrator does not have the right to remove an investigator, a client or a log file because it needs all the information to the statistics of the platform.

ii. Investigator:
Corresponds to a person who performs the investigation of log files. So, it can:
- Login
- Look at the log files to process
- Compare automatically part of logs with the dictionary attack
- Validate attack
- Write a report
- Add a new attack in the dictionary attack.

iii. Client:
It is the victim of the incident. This is a minor player because it has no direct interaction with the system. This user has no access to the functionality of the application, but it can trigger events that will change the status of the system:
- Register (Enter general information)
- Report an Incident
- Follow the incident sound processing status
- Modify contact information.

II.4 High-level diagrams
In what follows, we present the adopted analysis such as the High-level use case diagram with all requirement details for the investigative platform as well as class diagram, sequence diagram and activity diagram to show to evolution of the system. These diagrams have been implemented with UML language [7-9].

a- General Use Case Diagram
The following Use Case Diagram shows a detailed view of the system and how the actors would interact with each other and with the system. The explanation for each use case is then provided below the System Use Case (Figure3.1) and helps the user to understand who the actors areas well as giving the description of each use case along with its pre and post conditions that should be satisfied once the use case is implemented in the software.

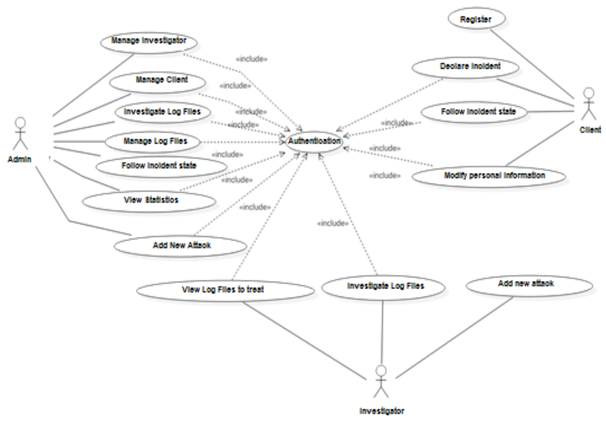

Figure2-1 General Use Case Diagram of the system

b. Sequence Diagram: Declaration of an incident
The following figure shows the sequence diagram for the reporting by the client of an incident. Indeed, the customer fills out the declaration form of an incident by inserting the necessary information. Once validated, the information will be stored in the database and the transaction sucked message will be displayed

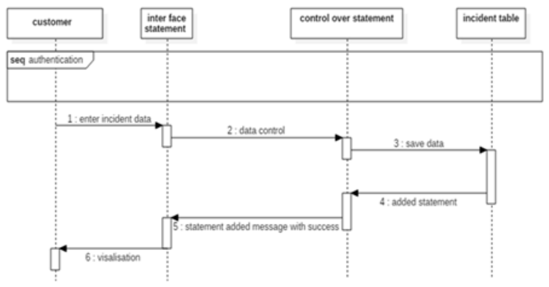

Figure2.2. Declaration of an incident SD

c. Sequence Diagram: Investigation on the log files
This diagram illustrates the steps of the investigation on the log files. Since the two actors (Director and Investigator) are entitled to the same functionality, we illustrate the case of an investigator.

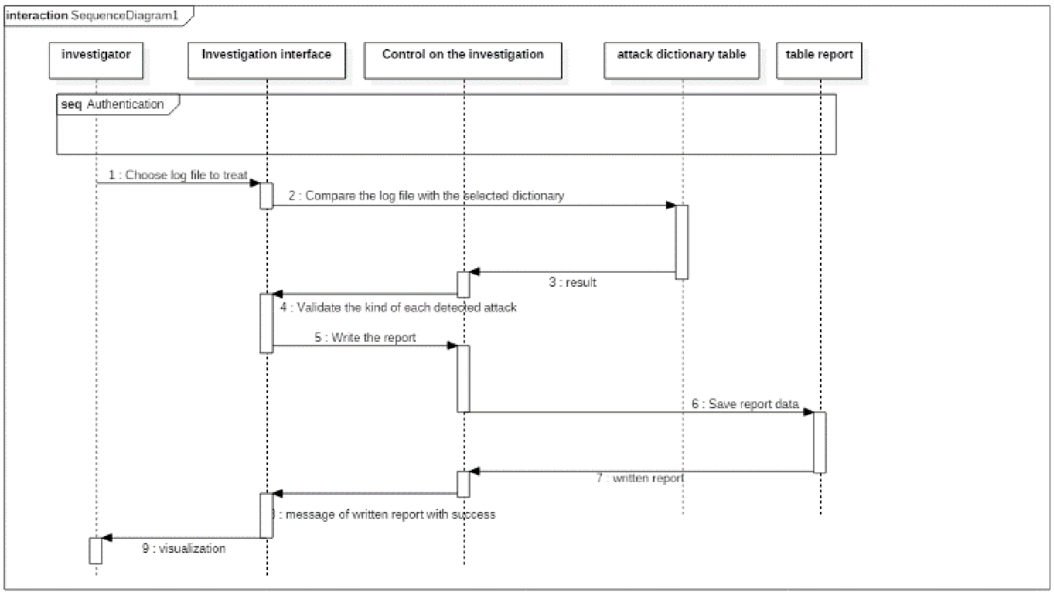
interaction SequenceDiagram1
investigator
Investigation interface
Control on the investigation
attack dictionary table
table report
seq Authentication
1 : Choose log file to treat
2 : Compare the log file with the selected dictionary
3 : result
4 : Validate the kind of each detected attack
5 : Write the report
6 : Save report data
7 : written report
: message of written report with success
9 : visualization

Figure2-3. Investigation on the log files SD

d. Activity Diagram

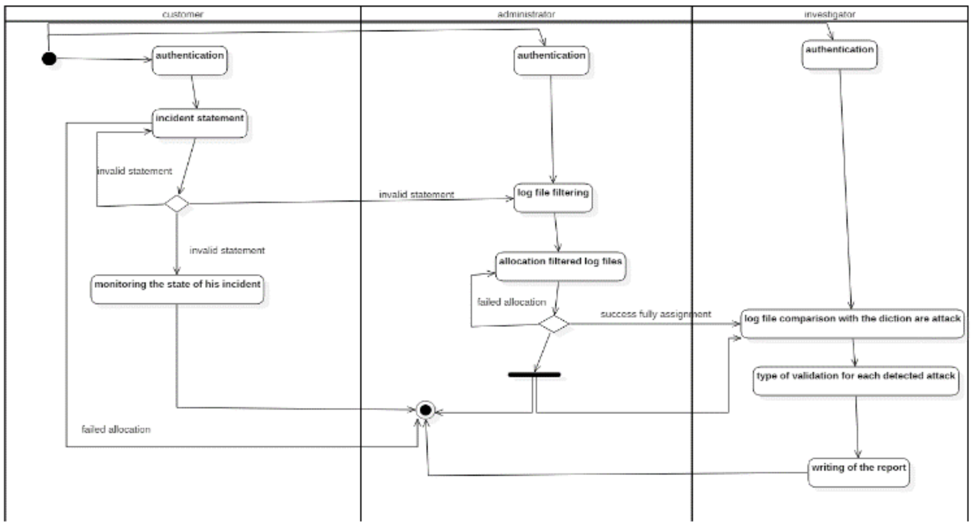

Figure2.4 Investigation process AD

The flow begins when the user starts the web desktop application. A login screen appears in the web browser, prompting the users to enter their ID and password. After the user logs in to the system, the system decides whether the user is an Admin or a normal user and then shows the appropriate user webpage. Users have a defined set of options for navigation once they log into the system.

e. Class Diagram

To facilitate the implementation of the proposed system, we consider the following classes to build our smart platform:

Incident class: contains information related to an incident.
Log: This is the class that contains information related to a log file.
Customer class: contains the coordinates of a client who made the declaration of an incident.
Administrator: The administrator class contains information specific to the administrator.
Investigator: this class contains specific information the investigator.
Report: The class report which includes the necessary contact details of a report by investigator
Dictionary Attack: The "Dictionary Attack" class contains information related to a dictionary attack.

Recommendation: Class recommendation that contains the necessary recommendations found in a report by type of attack.

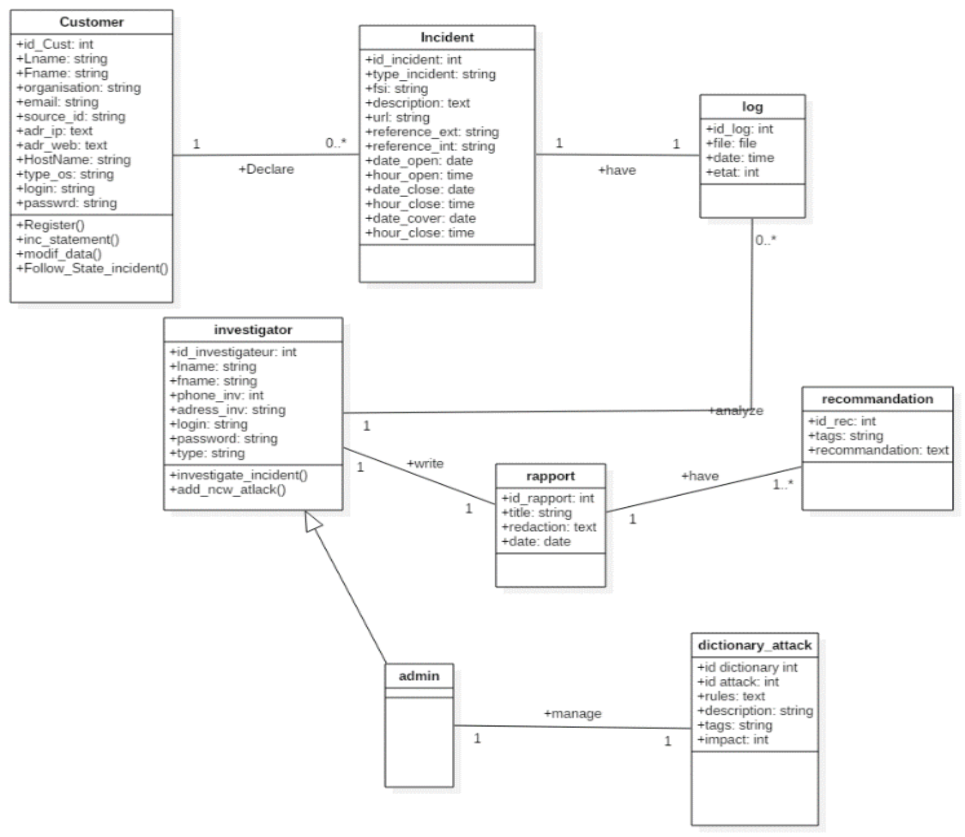

Figure2.5: Class diagram

III. CONCLUSIONS

The incidents are inevitable. Any organization, regardless of maturity in information security, must deal with a regular basis and try to reduce their number. The starting point will be well protected against all forms of attack, but in case the damage is already done we must try to know the types of threats to find the right solution and identify cyber criminals. In this context, a web application has been developed to achieve the establishment of an investigative platform of logs easy to use and automation of process of investigation. To carry out this mission, we have divided our work into two basic parts:

• A first covering the establishment of a dictionary attack with the types of the most common attacks and as a reference to determine any existing attack in a log file.

• A second part is interested to achieve our investigative platform for attack detection and identification of cyber criminals.

The analysis of this system has been successfully developed and presented in this work. The system design based on the uses cases diagram, class diagram, sequence diagram and activity diagram have been successfully implemented. The implementation of this platform required the implementation of several issues of Web security and event logs. Several avenues for future work can be envisaged such as the possibility to migrate this platform to the cloud computing architecture that can reduce investigation time and provide faster service.

REFERENCES

[1]. Munassar, N. M. A., & Govardhan, A. (2010). A comparison between five models of software engineering. International Journal of Computer Science Issues (IJCSI), 7(5), 94.
[2]. Mall, R. (2018). Fundamentals of software engineering. PHI Learning Pvt. Ltd.
[3]. Clarus Concept of Operations. Publication No. FHWA-JPO-05-072, Federal Highway Administration (FHWA), 2005
[4]. Braude, E. J., & Bernstein, M. E. (2016). Software engineering: modern approaches. Waveland Press.

[5]. Anwer, M., Farooq, M. U., Khan, S. M., & Waseemullah, W. (2021). Attack Detection in IoT using Machine Learning. Engineering, Technology & Applied Science Research, 11(3), 7273-7278.
[6]. Pan, Y., Sun, F., Teng, Z., White, J., Schmidt, D. C., Staples, J., & Krause, L. (2019). Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 10(1), 1-22.
[7]. El Yamami, A., Ahriz, S., Mansouri, K., Qbadou, M., & Illoussamen, E. (2017). Representing IT projects risk management best practices as a metamodel. Engineering, Technology & Applied Science Research, 7(5), 2062-2067.
[8]. Laaziri, M., Khoulji, S., Benmoussa, K., & Larbi, K. M. (2018). Information System for the Governance of University Cooperation. Engineering, Technology & Applied Science Research, 8(5), 3355-3359.
[9]. System Analysis and design with UML,4th Edition, 2012- John Wiley & Sons ,Inc.
[10]. Awokola, J. A., Emuoyibofarhe, O. N., Omotosho, A., Emuoyibofarhe, J. O., & Mebawondu, J. O. (2019). Picture Archiving and Communication System. Engineering, Technology & Applied Science Research, 9(5), 4859-4862.

Mohamed Ayari. “Smart Platform for Investigation of Log Files.” *International Journal of Engineering Research and Applications (IJERA)*, vol.12 (04), 2022, pp 01-08.