Journal of Integrated
SCIENCE & TECHNOLOGY

Digital forensic analysis of attack detection and identification in private cloud environments for databases

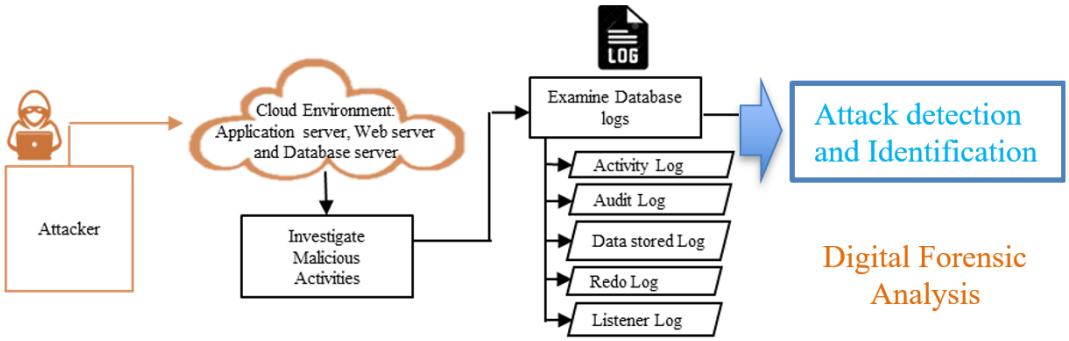
Varshapriya Jyotinagar,* Bandu B Meshram

Veermata Jijabai Technological Institute, Mumbai, Maharashtra. India.

Received on: 03-Nov-2023, Accepted and Published on: 24-Jan-2024

Article

ABSTRACT



Today, a growing number of applications are transitioning to the cloud due to its numerous benefits and user-friendly nature. Data plays a pivotal role in the migration, and its security is of paramount importance. Database forensics is a field dedicated to investigating malicious activities carried out by attackers, whether they occur on the front-end or back-end. These activities can be traced through the analysis of various database logs, with network details aiding in the identification of the attacker. In this paper, the authors explore the activity indicators by examining and analysing activity logs, database logs and network packets. Cloud forensics presents the digital investigation for the detection of cyber crimes on cloud. This could include data breaches or identity thefts or modification of databases and attacks on the application server and webserver.

Keywords: Attack, SaaS, Database, Forensic, Cloud

INTRODUCTION

Cloud computing presents a range of services, including Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). This technology operates through accessible data centers over the internet. Among these services, SaaS relies on databases as foundational components for data storage and retrieval, facilitating seamless functionality.[1]

Regrettably, malicious actors have recognized the value of databases and seek to exploit them for harmful purposes. Constantly devising new techniques, they compromise and manipulate databases within computer systems and applications, driven by malevolent intent. Consequently, it becomes crucial to thoroughly comprehend database vulnerabilities and develop effective countermeasures to prevent and detect such criminal activities.[2]

A critical field in addressing these issues is database forensics, which investigates malicious activities executed by attackers. Forensic experts heavily rely on analyzing diverse database logs, including Redo Logs, Activity Logs, and Audit Logs. This analysis unravels the evidence left behind by attackers, providing valuable insights into their actions. Complementing database forensics is network forensics, which plays a pivotal role in uncovering malicious activities traversing networks. This specialized branch of digital forensics concentrates on investigating malevolent actions conducted over network infrastructures. Wireshark, a notable tool for network information determination, aids in analyzing packet movement across networks.[3,4] By analyzing packet dumps obtained during network sessions, forensic analysts delve deeper into attackers' activities and extract crucial network details.

The research presented here delves into database and network forensics, aiming to advance understanding and capabilities in detecting, investigating, and mitigating malicious activities targeting databases and networks.[5,6] Leveraging insights from database logs and network packet analysis, the goal is to develop effective forensic methodologies and tools, enhancing the accuracy and efficiency of investigations in both cloud and traditional

*Corresponding Author: Varshapriya Jyotinagar
Tel: +91-22-24198167 Email: varshapriyajn@ce.vjti.ac.in

Cite as: *J. Integr. Sci. Technol., 2024, 12(4), 798.*
URN:NBN:sciencein.jist.2024.v12.798

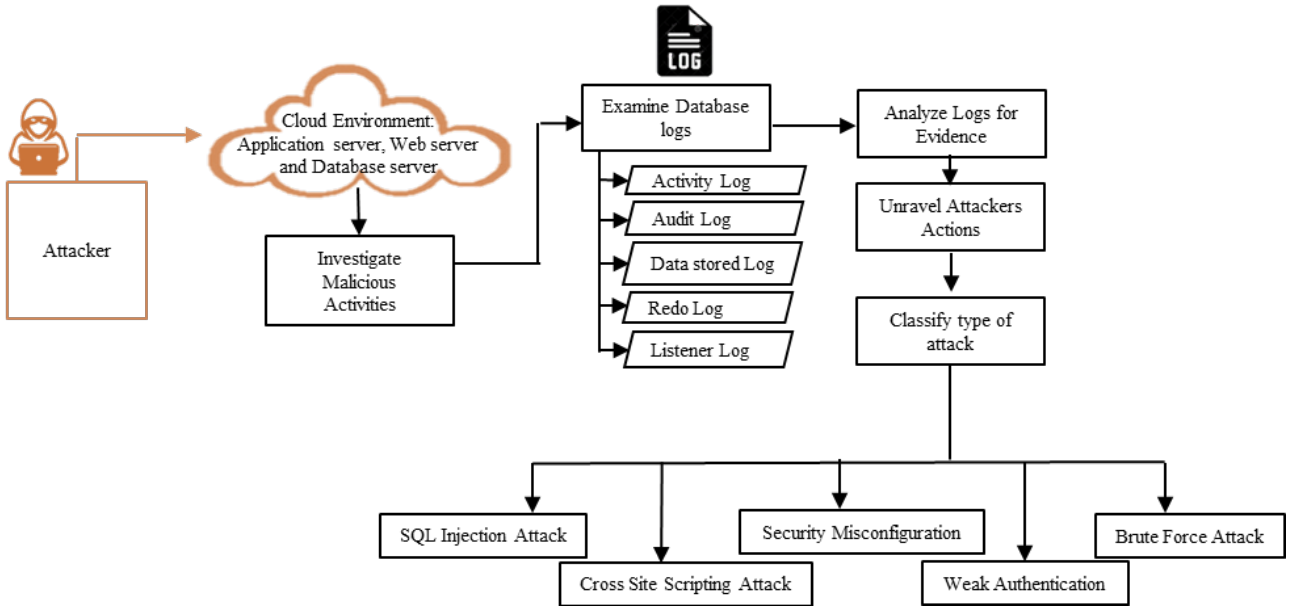


Figure 1: Designed Forensic Process

computing environments. (See Figure 1 for an illustration of the forensic process.)

Figure 1 illustrates the proposed forensic approach to counteract a malicious entity attempting to compromise a cloud instance hosting a Software-as-a-Service (SaaS) application. Initially, the attacker launches various types of attacks, including SQL injection, weak authentication, Cross-Site Scripting (XSS), and security misconfiguration.

The ”Cloud Instance (Running a SaaS Application)” refers to the virtual environment where the SaaS application is hosted. The attacker’s primary target is this cloud instance, aiming to gain unauthorized access or execute malicious actions. During these attacks, the system generates multiple log files, such as the redo log, activity log, audit log, datastore log, and listener log files, to record the activities.[7]

The compromised SaaS application generates additional log files to document the attacker’s actions. These log files play a pivotal role in understanding the specific actions the attacker took, the vulnerabilities they exploited, and the data they accessed or modified. By meticulously analyzing the log files, the investigator can accurately classify the attacks into specific types, such as SQL injection, weak authentication, XSS, and security misconfiguration. This classification helps in building a comprehensive understanding of the attack methods employed by the malicious entity, assisting in formulating an effective response and mitigation strategy.

LITERATURE SURVEY

Cloud computing has revolutionized the IT landscape, enabling the widespread adoption of various services such as web applications and learning platforms. While cloud providers integrate robust security measures, the responsibility for data security ultimately rests with clients. An influential report highlights the primary challenges in cloud security, with data loss and data privacy emerging as significant concerns, followed closely by compliance issues and accidental credential exposure.[8]

As organizations increasingly transition their workloads to the cloud, Security Operations Centers (SOCs) confront considerable hurdles in maintaining continuous security and ensuring visibility into system security. The establishment of consistent security policies across both cloud and on-premises environments and addressing the scarcity of skilled security personnel become paramount.[9]

Among the vulnerabilities within cloud security, unauthorized access and insecure interfaces are notable threats, along with misconfigurations of cloud platforms and the hijacking of accounts.

In this context, this paper emphasizes the pivotal role of database structures as foundational components in organizational setups. With the surge in database attacks, the exposure of sensitive data to potential attackers becomes a critical concern. The paper delves into various types of database attacks and explores their implications for data security and forensics. Most of the organization invest in self analysis forensic servers and remaining do forensic analysis after any illegal activity happens.

Numerous studies have investigated database attacks and their associated log file evidence. Particularly, SQL injection attacks have garnered significant attention, wherein attackers exploit vulnerabilities in web applications to inject malicious SQL queries, leading to unauthorized access and data manipulation.

Addressing insider threats has also been a focus of research, centering on individual with legitimate database access who misuse their privileges for unauthorized activities.The analysis of log files empowers forensic investigators to pinpoint attack vectors, track the actions of malicious actors, and reconstruct the sequence of events, facilitat ing effective incident response and mitigation strategies.[10]

In the experimental phase, a cloud environment was constructed using open-source cloud data center software or popular public cloud platforms such as AWS and Google Cloud. Previous studies

underscore the importance of comprehensive cloud infrastructures, including web servers, application servers, and database servers, to address associated challenges. Different assaults on databases are talked about in this paper. Notably, Wang et al. (2020)[1] proposed techniques to optimize server configurations in a cloud environment, enhancing resource allocation and overall system performance.

A range of techniques and algorithms have been developed to enhance cloud security. Notably, Hao et al., 2011[2] introduced a cloud security storage system, and Barati et al., 2014[3] provided an efficient method to detect attacks in encrypted networks. Qin et al., 2016[4] proposed Secure Scalar Invariant Feature Transform (SecSIFT) for privacy-preserving SIFT feature detection using the MapReduce frame- work. Additionally, Deng et al., 2017[5] leveraged the MapReduce framework to enhance algorithm performance.This paper concludes by highlighting the significance of various approaches to data protection, ranging from partition techniques[6] to blockchain- enabled distributed security frameworks,[7] in addressing threats such as DDoS attacks,[8] intrusion detection,[9] and zero-trust security frameworks.[10,11]

Apart from techniques and algorithms to enhance cloud security, various aspects of data analysis, machine learning, and visualization have been reported in literature.[12-30] The amalgamation of these contributions underscores the continuous evolution of cloud security strategies and the persistent pursuit of robust data protection measures. A significant portion of operations, including web services,[26] applications, and learning processes, is executed in the cloud. The basic anchor of this system lies in the database structure, representing a vital capacity for numerous organizations. Consequently, attacks targeting databases are on the rise, posing a significant threat.[12-14] Such attacks present a extreme form of intrusion, as they expose critical and valuable information to the aggressor.

This paper delves into various types of database attacks and analyzes the challenges for discussion in the next sub section.

ATTACKS AND THREATS OF DATABASE

SQL Injection: This is one of the most severe attacks. In an SQL injection attack, malicious queries are executed on the server, leading to the manipulation of the database. Attackers typically insert unauthorized database commands into an insecure SQL data channel.[17,19]

Cross Site Scripting (XSS): In XSS attacks, malicious scripts are injected into web applications. Attackers use these scripts to send harmful code, often in the form of a browser-side script, to other users, enabling them to gain unauthorized access to databases.[17,19]

Password Sniffing: This method involves capturing password information by monitoring network traffic. Attackers may use their associate's or a nearby machine to intercept data while the authentication server responds, capturing login credentials.[6,8]

URL redirection attacks: divert victims from a legitimate webpage to a fraudulent one, typically a phishing page. Attackers use social engineering techniques, such as phishing emails, to trick victims into visiting malicious sites, a tactic known as URL redirection attack.[17,18]

Brute Force Attack: This is a cryptographic hack that involves guessing various combinations of passwords until the correct one is found. Weak passwords are easy targets for attackers, emphasizing the importance of strong password policies in organizations.[18]

Excessive Privileges: When users or applications have more database privileges than necessary for their roles, they can misuse these privileges to access confidential information. This is a significant security risk, as many attacks on company data are initiated by insiders.[18]

Privilege Abuse: Users with legitimate data access privileges may misuse them for unauthorized purposes. Studies show that a substantial percentage of unlawful activities are carried out using privileges.[18]

Weak Authentication: Weak authentication methods allow attackers to impersonate legitimate database users. Attack strategies may include brute force attacks and social engineering.[18]

Security Misconfiguration: Vulnerable and poorly configured databases are common targets for attackers. Exploiting these vulnerabilities is a typical tactic used by attackers against organizations.[18]

ANALYZING CHALLENGES

Research indicates that the analysis of attacks is continually improving. Some advantages and drawbacks are associated with forensic data analysis.

Advantages:

Comprehensive data recording.

Systematic categorization and distributed storage for easy cross-referencing.

Enhanced physical security due to undisclosed server and database locations.

Drawbacks:

Overwhelmingly large data volume.

Complex pattern analysis and identification.

Ongoing vigilance required for monitoring and detecting illegal activities.

To tackle these challenges, we propose a solution that leverages the power of the pandas DataFrame in digital forensics.[13-15] Pandas, a well-regarded open-source Python library, is renowned for its high-performance data manipulation and analytical capabilities.[20]

METHODOLOGY

The utilization of the cloud for application deployment, owing to resource limitations, introduces potential vulnerabilities for attackers to exploit and compromise data secu rity, leading to service disruptions. In response to such incidents, forensic investigations are crucial in extracting essential attack details, including the date, time, and extent of database damage. These investigations involve the retrieval of database logs and utilizing default facilities provided by the database for analysis.

This research employs a qualitative analysis to examine various types of cyber-attacks prevalent in cloud environments, with a specific focus on SQL Injection, Cross-Site Scripting (XSS), Brute Force Attacks, Weak Authentication, and Security Misconfiguration.

In our literature review, we explored various attack mechanisms employed by malicious actors and examined the corresponding traces they leave within a system. Leveraging this understanding, we extract pertinent data along the affected path, as illustrated in Figure 2. These traces encompass a range of activities, including illegal actions, user privilege issues, cross-site scripting (XSS), security misconfigurations, and notably, alterations to the database. By capturing network packets, we comprehensively gather data flowing to and from other nodes within the network

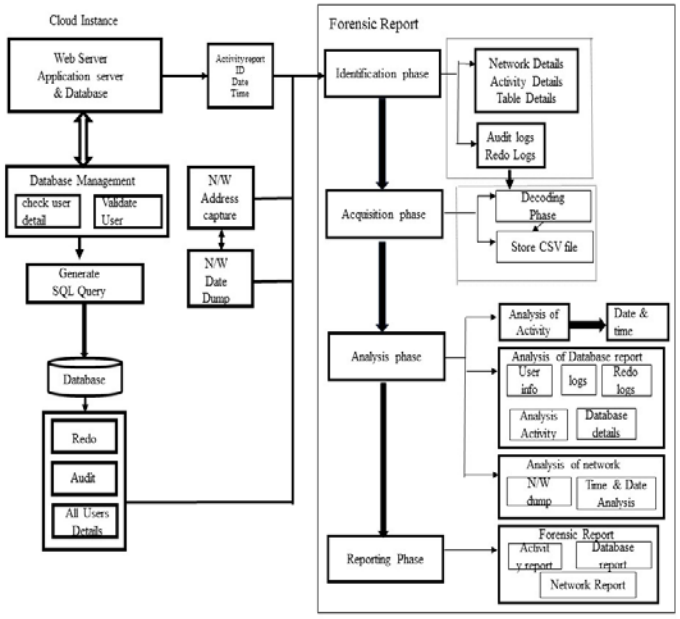


Figure 2: Block Diagram of Cloud Attack Environment Setup and Forensic Investigation Workflow for Cloud-Based SaaS Security

The block diagram presented in Figure 2 illustrates the data flow and information exchange from the client-side cloud instance to the forensic investigation side, where four distinct phases of forensic investigation are carried out. These phases encompass Identification, Acquisition, Analysis, and Reporting. The Identification phase involves the recognition of various traces, such as illegal activities, SQL injection, XSS, Security Misconfiguration, and significant Database Changes. This phase entails identifying crucial components like network dumps, activity logs, database tables, audit data, and redo logs. In the Acquisition phase, relevant data is collected and acquired, followed by decoding and storage in the required CSV file format. This step ensures that the necessary information is accessible for further examination. The Analysis phase encompasses three key aspects: analyzing activity with its corresponding date and time, studying the database logs, and investigating network packets. These analysis contribute to a comprehensive understanding of the events and potential security breaches. Finally, the Reporting phase involves generating detailed activity, database, and network reports. The ultimate objective of this process is to identify any suspicious or unauthorized activities, extract relevant time and date information from the Activity report, and establish correlations between the activity data, database logs, and Network logs. Furthermore, a dedicated analysis tool is created to facilitate the examination of this data. Overall, this workflow aims to uncover and address any secu rity incidents effectively, ensuring the integrity and security of the cloud-based SaaS environment.

In database, different logs are connected with each other. Mainly we have four log files having essential features which are important while linking traces in forensic analysis.

In digital forensic analysis, the interconnection between audit logs, redo logs, activity logs, and network dump logs plays a crucial role for linking traces in forensic analysis and reconstructing events within a computing environment. The examination of existing literature led us to the conclusion that if an attacker acquires authenticated ID and password for the database, it often manifests as legitimate access in the database logs. To address this, we propose a data analysis approach that involves mapping different logs obtained from diverse cloud services.

Figure 3 depicts the software architecture diagram for the proposed forensic tool. The forensic tool is designed based on the forensic investigation workflow for Cloud-Based SaaS Security. The process begins by identifying instances of illegal activity, extracting relevant time and date information from the Activity report, and subsequently mapping this data with both database logs and network logs. The outcome is a tool designed for the

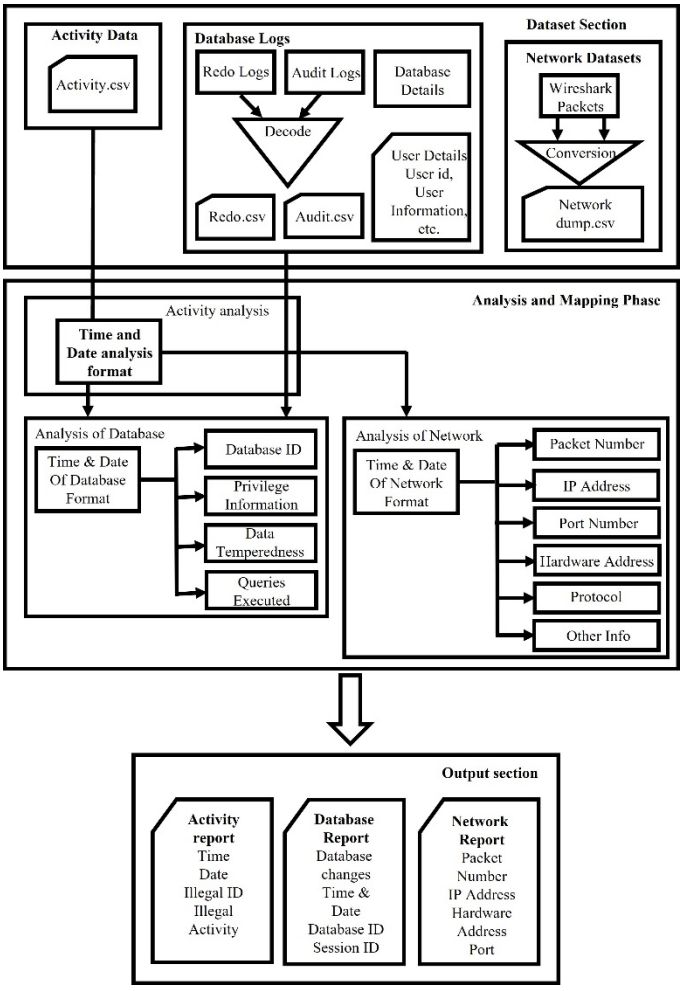


Figure 3: Software architecture diagram for Forensic tool

comprehensive analysis of such data. This involves converting various types of database logs into CSV files for thorough analysis. The methodology involves three main sections: Datasets , Analysis and Mapping .Output. In the Dataset Section, all data collected from the attacked machine is aggregated. The Analysis and Mapping Section comprises three modules: Module 1 for Activity file analysis, Module 2 for database logs analysis, and Module 3 for Network packet dump analysis. These modules are designed to conduct in-depth analysis of their respective data sources. The Output Section generates the report and findings based on the results obtained from the analysis modules. The report authoring system, integral to the forensic tool, is implemented to compile and present the findings effectively. By following this methodology, we aim to systematically analyze and interpret the collected data, enabling us to draw meaningful conclusions and provide valuable insights in our forensic investigation.

IMPLEMENTATION

Based on the experimentation research on AWS Cloud database forensic software architecture, the database forensic framework is proposed for network traffic analysis and database forensic analysis. The software framework of the forensic tool is designed to identify instances of illegal activity by examining and analysing database logs and network logs, providing information about the time and date when such activities occurred. The proposed software framework consists of Activity file analysis , database logs analysis and Network packet dump analysis. Implementation is discussed in the following categories Cloud Environment Setup, Infected Paths and Logs, Utilized Programming Languages and Tools in the Research, Forensic workflow.

CLOUD ENVIRONMENT SETUP:

Amazon Web Services (AWS) is utilized to create the cloud environment, employing a Windows server machine. A Student Management system website portal is deployed on this cloud server to facilitate registration and basic features for analysis purposes. These features include new student registration, course registration, updating marks, checking details, and editing information. To ensure security, the system maintains two types of logins: normal user login and Admin login. The backend of the system is powered by an Oracle database, leveraging its default activity monitoring service, auditing reports, redo logs, and other built-in functionalities. Network activity data is collected using network miner tools on the cloud. Additionally, an API is developed to enable activity detection on the website. When a user logs in as either a normal user or an admin, a notification is sent to their respective personal Gmail accounts. Based on their response, an activity report is generated.

INFECTED PATHS AND LOGS:

A website is built and deployed on the AWS cloud. Any attacking activity triggers distinct symptoms, which are recorded in various system logs. The logs and their meanings are analyzed to identify different types of attacks. Table 1 presents a comprehensive list of attacks, their names, and the corresponding traces found in the logs.

Table 1 Attacks and their paths

Logs	Attack Names	Path
Network Dump	Man in the Middle Denial of service BruteForce Attack	C:/NETWORK/DUMP
Auditing log	SQL Injection Cross Site Scripting (XSS)	F:/APP/MARK2/ADMIN/ORCL/ADUMP
DataStored	XSS(Cross site Scripting) Security Misconfiguration Insecure Logging	F:/APP/MARK2/ORACLE/ORADATA/ORCLI/
Activity.log	Illegal Activity Identification	https://www.google.com/forms/about/
Redo Log	Database Changes	F:/APP/MARK2/ORACLE/ORADATA/ORCLI/

In this research, the focus is on analyzing attacks and their paths by examining various log files related to database and network activities. The following log files are utilized:

Redo Logs:

These logs store all changes made to the database through queries. The critical fields include Timestamp (time and date records), Operation (start, update, commit), SQL Redo (SQL queries), Start SCN (System Change Number), Commit Timestamp (date for committed SQL queries), and Commit SCN (change in the database for a specific command).

Data Stored:

This log contains data stored in the database for each user and their respective tables. To visualize data in this location, we connect databases with SQL developer for better visualization.

Auditing Log:

Oracle generates this log to track activities on the tables. The essential fields in this log are Number (serial number), SQL Text (SQL queries), DB ID (Database ID), Transaction ID (generated for each transaction), and Commented Text (additional information such as the protocol used, port number, and authenti cation details).

Activity Log:

This log is generated by clients and consists of responses from authen ticated users connected to the AWS cloud. The fields in this log include Timestamp (time and date records), Are you sign In (indicating whether the user signed in or not), ID (user ID), Time (input, asking about time), and Date (input, asking about date).

Listener Log:

The listener log file is used to record details of connections made to the Oracle listener. The listener log file contains information such as the time of connection, the client IP address, and the service name requested. The location of the listener log file is specified in the listener.ora file.By default, the listener log file is named listener.log, but this can be changed in the listener.ora file. The listener log file is useful for troubleshooting connectivity issues in Oracle database environments.

Network Dump:

This log stores all network packets. The important fields in this log include Packet Number (a unique number given to each packet), Time (date and time information), Source (source IP), Destination (destination IP), Protocol (protocol used), Length (packet length), Info (information like sequence number, acknowledgment, or application data), Des Port (destination port address), mac Source (hardware address of the source), and mac Destination (hardware

address of the destination).Attacks with their names and their traces paths in table 1. Table 1 provides a comprehensive overview of the log files used in the research, their file paths.

UTILIZED PROGRAMMING LANGUAGES AND TOOLS IN THE RESEARCH:

The research employs a diverse range of programming languages and tools for various purposes, including development, database management, and data analysis. The main programming languages and frameworks used are as follows:

Python 3.0: This language is utilized for tool development, and the TKinter GUI framework is employed to create a user-friendly graphical interface.

Java version 1.8: Java is used for programming and interacting with the Oracle Database Express Edition.

SQL: SQL is used in conjunction with Oracle SQL Developer for managing and query- ing the Oracle database.

Visual Studio Code Editor: This tool serves as the integrated development environ- ment (IDE) for coding tasks and script writing.

For data analysis, the following libraries and tools are used:

Python Pandas Library: The Python Pandas library is employed for data manipulation, analysis, and processing tasks.

Network Analysis with Wireshark Version 3.0.3/Network miner: Wireshark is utilized for network packet analysis and examining network traffic.

The research effectively combines this diverse set of programming languages and tools to address different aspects of the study, ranging from tool development using Python and Java to data analysis using Python Pandas and network analysis with Wire- shark/Network miner.

To implement our proposed software tool, we have developed three specific algorithms for analyzing activity, database, and network logs. These algorithms aim to acquire and analyze data from the logs effectively.

In the domain of database forensics, four main logs are emphasized, namely audit logs, data stored logs, and listener logs, and redo logs. Each of these logs plays a crucial role in detecting and investigating potential security breaches and attacks on the database.

FORENSIC WORKFLOW

Figure 3 shows a software architecture diagram for forensic tool. We first identify when incident has occured and extract time and date from the Activity report and do mapping with the database logs and Network logs and create a tool for analysis of such data. Different types of logs are converted into csv files and analysis is done. To implement our proposed software tool, we have developed three specific algorithms for analysing activity, database, and network logs. These algorithms aim to acquire and analyse data from the logs effectively. Each of the logs plays a crucial role in detecting and investigating potential security breaches and attacks on the database. In figure 3 We create three sections Datasets, Analysis and Mapping and Output. The Dataset Section contains all data collected from the compromised machine which contains activity logs, database logs from the database server and network related dataset like network packets captured from wieshark tool or network miner. The activity log gives the information of activity data done by the user. Database logs contain redo logs, audit logs and database files which may not be in the readable format. Hence it has to be decrypted. The Analysis and Mapping Section, there are three modules Module 1 is for Activity file analysis, Module 2 is for database logs analysis and Module 3 is for Network packet dump analysis. The analysis of activity.csv involves time and date analysis. The analysis of database logs involves time and date of database format. The database format may contain database ID, privilege information, data tamperdness and queries executed. Analysis of network involves analysis of date and time of network format. The network format may include packet number, IP address, port number, hardware address, protocol and other information. The Output Section gives our report and findings respectively. The report authoring system is shown in implementation part of the forensic tools.

Extraction of Activity Data

Purpose: The purpose of this algorithm is to extract data from an activity log file. Variables Used: Activity.csv (the activity log file)

Input: The location of the activity data.

Output: The Activity.csv file extracted from the specified input location. Start:

Go to the admin Google Form at https://www.google.com/forms/about/.

Sign in with the Admin account.

Select ”Event Registration” from the options.

Navigate to the response section.

Choose the option to download the report in CSV format.

Save the downloaded file as ”Activity.csv”.

End

To Analyse Activity report:

This algorithm aims to extract time, date, and illegal IDs associated with illegal activities from an input Activity.csv file and generate an activity report.

Variables Used: file name, event, d, idx, date, time, id Input: Activity.csv file containing activity data

Output: Activity report with time, date, and illegal IDs

Start

1. Identify the location of the 'Activity.csv' file and store its name as file_name.

2. Create a Pandas DataFrame named 'event' for the data in file_name.

3. Create an empty DataFrame called 'd'.

4. Commence a For Loop.

5. Perform a Sequential Search for instances where 'Are you Sign IN' is 'No'.

6. Set the index (idx) to the current loop iteration number.

7. Gather the date associated with the instance.

8. Gather the time associated with the instance.

9. Collect the ID related to the instance.

10. Add the gathered data to the DataFrame.

11. End the loop.

12. Generate the Activity report based on the collected data.

End

Extracting Data from Audit Log File and Collecting into CSV

The purpose of this algorithm is to extract data from an audit log file stored at a specific location and collect it into a CSV file for further analysis and investigation.

Variables Used:. Audit.csv

Input:Location of audit og 'F:\APP\MARK2\ADMIN\ORCL\ADUMP'

Output :Audit.csv

Start:

1. Launch SQL Developer application.
2. Access the Oracle database console.
3. Connect to the admin panel.
4. Input the ID and Password credentials for authentication.
5. Execute the SQL command: 'SELECT * FROM dba_audit_trail;'
6. Enter the command for exporting data.
7. Assign the name 'Audit.csv' to the exported file.
8. : Audit.csv has been successfully generated.

End

Analyzing Extracted Data from Auditing Logs

Module 1

This algorithm analyze the data extracted from the auditing logs stored in the Audit.csv file and retrieve the row index that matches the provided date and time.

Variables used: date,time,counter, r, idx, p2, sample,file name2,Log sample2,

Input:Audit.csv file,time,date

Output:Required index value in Audit.csv

Start

1. Read date, time
   date = dt.get()
   time = ti.get()
2. Split time into minute and hour
   t = time.split()
3. Set Counter = 0
4. Get location of audit.csv to root.file_name2
5. Create dataframe Log_sample2 = dataframe(audit.csv)
6. Loop start reversed on Log_sample2
7. Store sample = date and time for ith index (idx)
8. Check User given date with sample date
9. If found, Counter = 1
10. Store p2 = idx
11. End Loop

End

To Extraction log data if changes are made in database

Module 2

This algorithm collect Database changes to analyse attacks Variable Used:p2, idx, sample, temp frame2, report2.

Input p2, Log sample2

Output Report generation database changes

Start

1. Make an empty dataframe report2 for saving results
   report2={'Date':[],'Time':[],'Executed SQL Statement':[], 'Session ID': [], 'DB ID': []}
2. Loop through Log_sample2
3. Continue until reaching index p2 in Audit.csv
4. Sample = get time and date from the p2-th row
5. Store split in audie_t
6. Store date, time, sql_queries, session_id, DB_id into report2
   report2 = {
   'Date': [Log_sample2[p2].date()],
   'Time': [Log_sample2[p2].time()],
   'Executed SQL Statement': [Log_sample2[p2].sql()],
   'Session ID': [Log_sample2[p2].Sessionid()],
   'DB ID': [Log_sample2[p2].id()]
   }
7. End Loop
8. Generate report Database_changes
   report2.to_html("Database_changes.html")

End

Extraction of redo logs

This algorithm is used to extract the redo logs and get Decrypted redo logs

Variable Used: None

Input Redo log location 'F:\app\mark2\oradata\orcl\ .

Output: Decrypted redo logs

Start

1. Start SQL Developer
2. Navigate to Oracle console
3. Connect using admin login ID and password
4. Add redo log inputs to LogMiner
   a. Execute DBMS_LOGMNR.ADD_LOGFILE(REDO01)
   b. Execute DBMS_LOGMNR.ADD_LOGFILE(REDO02)
   c. Execute DBMS_LOGMNR.ADD_LOGFILE(REDO03)
5. Change pdb_orcl mode to READ_WRITE mode
   ALTER PLUGGABLE DATABASE PDBORCL OPEN READ WRITE;
6. Start LogMiner to decode redo logs
   EXECUTE DBMS_LOGMNR.START_LOGMNR();
7. Retrieve results from V$LOGMNR_CONTENTS
   SELECT * FROM V$LOGMNR_CONTENTS;
8. Access export section
9. Save the results as Redo.csv

End

Extraction of Network Dump using Network Analyzer tool

This algorithm collects the data from network analyzer and store the network_dump .csv data file.

Variable Used None

Input" Wireshark service

Output : network_dump.csv

1. Go to Wireshark service
2. Navigate to the file section
3. Save the capture as a CSV file
   a. Choose "Save As" option
   b. Select CSV format
4. Provide the name "Network_dump"
5. Store the file in a safe location

End

Analyzing Extracted network data from Network_dump.csv

Variable Used date, time, t, audie_t, idx

Input Network_dump.csv, time = (Provided by user), date = (Provided by user)

Output: index number of packet

Start

1. Read date, time
2. T = split time into minute and hour
3. Set Counter = 0
4. Get location of Network_dump.csv to file_name3
5. Create data frame network_sample =dataframe(Network_dump.csv)
6. Loop start reversed on network_sample
7. Store sample = date and time for ith index (idx)
8. Check user-given date with sample date
9. If found, Counter = 1
10. Store p2 = idx
11. End Loop
12. From index p2, retrieve IP, hardware address, port number, packet number
13. Display IP, hardware address, port number, packet number

End

Algorithm To check Database threats

This algorithm Identify three database threats XSS, Security Misconfiguration, Insecure Logging

Variable Used :None

Input Sql Developer Service

Output Identify oracle threat

1. Open SQL Developer
2. Connect to the admin using login ID and Password
3. Iterate through every table properties:
   a. Retrieve a list of user tables: SELECT * FROM USER_TABLES;
   b. For each table in the list:
      i. Retrieve all data: SELECT * FROM <TABLE_NAME>;
      ii. Retrieve table metadata:
      SELECT table_name, column_name, data_type, data_length
      FROM USER_TAB_COLUMNS WHERE table_name = <TABLE_NAME>;
4. Search for specific keywords in the data: input buffer, password, owner
5. If the size of the input buffer > 100, it indicates a potential script injection possibility
6. If the password is not a string, it suggests a higher possibility of a brute force attack
7. Identify if multiple tables belong to a single user; this scenario could indicate a major impact in case of an attack

End

Evidence Report

We showed the result according to date and time. From activity we trace attack time, date and illegal id used. Getting input from the activity report we dig into the database and find all database changes done and network packets at that session.

RESULT

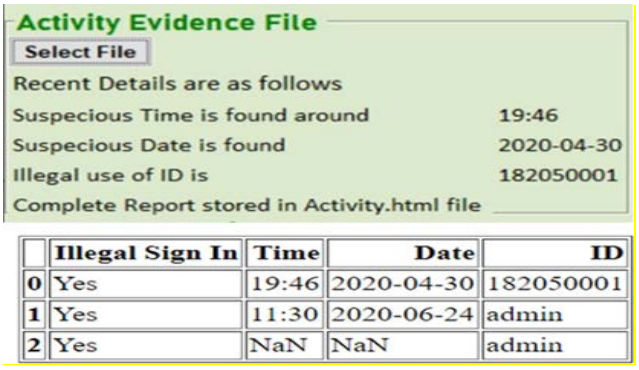
Results are presented in three categories. First we give Time and Date for when Illegal activity is detected, Second we give a Database report that gives a time, date and executed queries from that time And at last we give network related info at that time such as hardware address and ip and packet number from that dump and later all network packets are displayed on the generated report. The evidence collection process involved extracting data from all infected files stored at a particular location and consolidating it for further analysis.

The activity report is generated when ever an illegal activity takes place such as unauthorized access to the SaaS application. The email sent to the valid user to approve the activity or else fill the google form if the activity is not done by the valid user. The data extracted and analysed from the Activity.csv file using an proposed software tool is presented in the report form as shown in the figure 4. The file contained essential information about user interactions and system events.

The activity report flagged a suspicious time of occurrence around 19:46. This time raised concerns as it falls outside the typical operational hours or may coincide with periods of reduced supervision and increased vulnerability. The report also identified a suspicious date as 2020-04-30. This date raised red flags due to its association with potentially critical events, known incidents, or other unusual patterns,events, known incidents, or other unusual patterns. The report highlighted the use of an illegitimate identification number (ID) - ”182050001.” The presence of an unauthorized or fake ID suggested an attempt to bypass security measures or impersonate a legitimate user. The combination of these suspicious activity indicators pointed to possible security breaches, unauthorized access attempts, or insider threats within the digital system. Additionally, we examined the contents of the complete activity report, which were stored in the ”Activity report.html” file, to gain a more comprehensive understanding of the events leading up to and following the flagged activities. By promptly identifying and responding to suspicious activities, organizations can strengthen their security posture and protect their valuable assets from unauthorized access and data breaches.

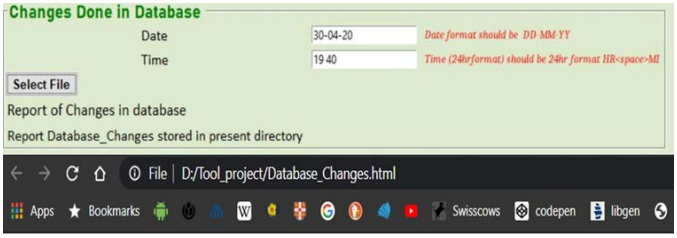
Second we give a Database report as shown in figure 5. It gives a time, date and executed queries from that time. The result focuses in-depth investigation into monitoring and analyzing database activities to identify and track changes within a database system. The software tool examined a comprehensive database report file generated by uploading a database.csv file. The report contained critical information pertaining to database changes and alterations. The database report highlighted recent changes to the database, offering valuable insights into the frequency of alterations made to the database. This file served as a valuable resource, capturing essential details such as dates, times, executed SQL statements, session IDs, and database IDs. This report allowed us to drill down into specific modifications, providing insights into the types of changes made, users responsible for them, and the impact on overall database operations. The ”database report.html” file is analyzed to

extract crucial details such as the sequence of events, timestamps of changes, corresponding SQL statements, and the context of database sessions and IDs.



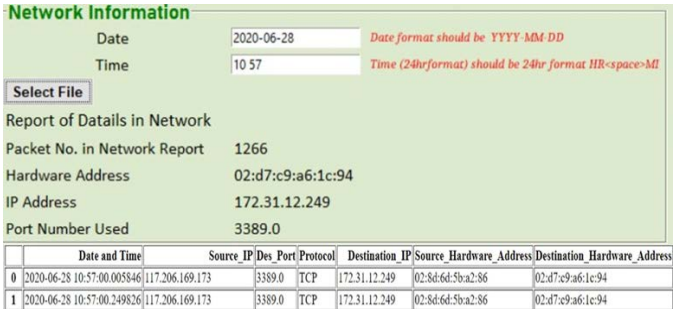
	Illegal Sign In	Time	Date	ID
0	Yes	19:46	2020-04-30	182050001
1	Yes	11:30	2020-06-24	admin
2	Yes	NaN	NaN	admin

Figure 4: Activity Report



	Date	Time	Executed SQL Statement	Session ID	DB ID
0	30-04-20	01:33:43.465000000	select * from login_db where id='18205001'	151500.0	1.563801e+09
1	30-04-20	01:58:37.174000000	select * from login_db where id='asdf¿ or ¿1¿=¿1'	151516.0	1.563801e+09
2	30-04-20	01:59:33.192000000	select * from login_db where id='asdf¿ or ¿1¿=¿1'	151517.0	1.563801e+09
3	30-04-20	02:00:38.830000000	select * from login_db where id='akshay¿ or ¿1¿=¿1'	151524.0	1.563801e+09

Figure 5: Database Report



	Date and Time	Source_IP	Des_Port	Protocol	Destination_IP	Source_Hardware_Address	Destination_Hardware_Address
0	2020-06-28 10:57:00.005846	117.206.169.173	3389.0	TCP	172.31.12.249	02:8d:6d:5b:a2:86	02:d7:c9:a6:1c:94
1	2020-06-28 10:57:00.249826	117.206.169.173	3389.0	TCP	172.31.12.249	02:8d:6d:5b:a2:86	02:d7:c9:a6:1c:94
2	2020-06-28 10:57:00.330268	172.31.12.249	51145.0	TLSv1.2	117.206.169.173	02:d7:c9:a6:1c:94	02:8d:6d:5b:a2:86
3	2020-06-28 10:57:00.359780	172.31.12.249	51145.0	TLSv1.2	117.206.169.173	02:d7:c9:a6:1c:94	02:8d:6d:5b:a2:86
4	2020-06-28 10:57:00.451852	172.31.12.249	51145.0	TLSv1.2	117.206.169.173	02:d7:c9:a6:1c:94	02:8d:6d:5b:a2:86
5	2020-06-28 10:57:00.451911	172.31.12.249	51145.0	TLSv1.2	117.206.169.173	02:d7:c9:a6:1c:94	02:8d:6d:5b:a2:86
6	2020-06-28 10:57:00.514211	172.31.12.249	51145.0	TLSv1.2	117.206.169.173	02:d7:c9:a6:1c:94	02:8d:6d:5b:a2:86
7	2020-06-28 10:57:00.578806	117.206.169.173	3389.0	TCP	172.31.12.249	02:8d:6d:5b:a2:86	02:d7:c9:a6:1c:94
8	2020-06-28 10:57:00.672794	117.206.169.173	3389.0	TCP	172.31.12.249	02:8d:6d:5b:a2:86	02:d7:c9:a6:1c:94

Figure 6: Network Report

And at last we give network related info at that time such as hardware address(MAC address of source and destination) and ip and packet number from that dump and later all network packets are displayed on the generated report as shown. The result shows the significance of robust network traffic analysis. By capturing and examining network details, organizations can proactively identify and mitigate potential security risks. Our primary objective was to develop a robust network detail file that captures essential information about network activities and enables the detection of potential security threats. The file contained critical details about network traffic, including packet numbers, hardware addresses, IP addresses, and port numbers as shown in figure 6. The following key details were captured in the file.

The comparison of the proposed tool with other forensic methods proposed by Sachdeva et al.[33] and Roussev et al.[32] is given below.

Forensic Method	Sources	Execution time	No of attacks (Attack type)	Type of cloud
Sachdeva et al. [33]	Documents store, memory information, and virtual memory information	5 mins 33 secs	3 (Internet Control Message Protocol Attack, Transmission Control Protocol Sync Attack, and User Datagram Protocol Attack)	Traditional forensic
Roussev et al. [32]	Filesystem metadata extraction	2 Min 38 Secs	No attacks performed as such.	Traditional forensic,Non Cloud
Proposed Method	Logs and Network traffic	3 min 50 Secs	(5) SQL injection, Cross site Scripting, Security Misconfiguration, Weak Authentication and Brute force attack	Private cloud

Conclusion

To conduct a forensic process effectively, a comprehensive understanding of various organizational aspects is crucial. This includes knowledge of the complete organizational structure, database design adopted by the organization, facilities utilized, permissions allocated to different user types, and the overall database architecture. For our forensic investigation, we specifically focused on utilizing an Oracle database and its corresponding architecture and newly introduced concept of pluggable database in Oracle 19C version and their hierarchy. In the case of hardware you used to do forensic. The proposed method offers a valuable tool for businesses to enhance their ability to detect and mitigate potential security breaches effectively. By promptly identifying and responding to suspicious activities, organizations can bolster their security posture and protect their valuable assets from unauthorized access and data breaches.

Conflict of Interest Statement

Authors declare that there is no Conflict of interest or competing interests.

References and Notes

1. Z. Wang, S. Zheng, Q. Ge, K. Li. Online Offloading Scheduling and Resource Allocation Algorithms for Vehicular Edge Computing System. *IEEE Access* **2020**, 8, 52428–52442.
2. L. Hao, D. Han. The study and design on secure-cloud storage system. *2011 International Conference on Electrical and Control Engineering, ICECE 2011 - Proceedings*. 2011, pp 5126–5129.
3. M. Barati, A. Abdullah, N.I. Udzir, et al. Intrusion detection system in secure shell traffic in cloud environment. *J. Comput. Sci.* **2014**, 10 (10), 2029–2036.
4. Z. Qin, J. Yan, K. Ren, C.W. Chen, C. Wang. SecSIFT: Secure image SIFT feature extraction in cloud computing. *ACM Trans. Multimed. Comput. Commun. Appl.* **2016**, 12 (4s), 1–24.
5. S. Deng, A.H. Zhou, D. Yue, B. Hu, L.P. Zhu. Distributed intrusion detection based on hybrid gene expression programming and cloud

computing in a cyber physical power system. *IET Control Theory Appl.* **2017**, 11 (11), 1822–1829.
6. G. Levitin, L. Xing, Y. Dai. Co-residence based data vulnerability vs. security in cloud computing system with random server assignment. *Eur. J. Oper. Res.* **2018**, 267 (2), 676–686.
7. D.V. Medhane, A.K. Sangaiah, M.S. Hossain, G. Muhammad, J. Wang. Blockchain-Enabled Distributed Security Framework for Next-Generation IoT: An Edge Cloud and Software-Defined Network-Integrated Approach. *IEEE Internet Things J.* **2020**, 7 (7), 6143–6149.
8. S. Sambangi, L. Gondi. A Machine Learning Approach for DDoS (Distributed Denial of Service) Attack Detection Using Multiple Linear Regression. In *The 14th International Conference on Interdisciplinarity in Engineering—INTER-ENG 2020*; MDPI, Basel Switzerland, **2020**; Vol. 63, p 51.
9. N.O. Ogwara, K. Petrova, M.L. Yang, L. Tan. Towards the Development of a Cloud Computing Intrusion Detection Framework Using an Ensemble Hybrid Feature Selection Approach. *J. Comput. Networks Commun.* **2022**, 2022, 1–16.
10. S. Bhatt, B. Bhushan. Cyberattacks and Risk Management Strategy in Internet of Things Architecture. *Artificial Intelligence and Cybersecurity*. 2021, pp 51–68.
11. D. Xue. Research on Identification of Illegal Intrusion in Ship Communication Network Based on Depth Learning Algorithm. *J. Coast. Res.* **2020**, 115 (sp1), 127–129.
12. M. Ali, L. Tang Jung, A. Hassan Sodhro, et al. A Confidentiality-based data Classification-as-a-Service (C2aaS) for cloud security. *Alexandria Eng. J.* **2023**, 64, 749–760.
13. H. Chung, J. Park, S. Lee, C. Kang. Digital forensic investigation of cloud storage services. *Digit. Investig.* **2012**, 9 (2), 81–95.
14. A. Pichan, M. Lazarescu, S.T. Soh. Cloud forensics: Technical challenges, solutions and comparative analysis. *Digit. Investig.* **2015**, 13, 38–57.
15. R. Battistoni, R. Di, P. Dipartimento, F. Lombardi. CloRoFor : Cloud Robust Forensics. *Int. J. Adv. Comput. Sci. Appl.* **2013**, 10 (3), 1–12.
16. A. Liu, H. Fu, Y. Hong, J. Liu, Y. Li. LiveForen: Ensuring Live Forensic Integrity in the Cloud. *IEEE Trans. Inf. Forensics Secur.* **2019**, 14 (10), 2749–2764.
17. E. Daniel, S. Durga, S. Seetha. Panoramic view of cloud storage security attacks: An insight and security approaches. In *Proceedings of the 3rd International Conference on Computing Methodologies and Communication, ICCMC 2019*; **2019**; pp 1029–1034.
18. S.B. Mallisetty, G.A. Tripuramallu, K. Kamada, et al. A Review on Cloud Security and Its Challenges. In *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*; IEEE, **2023**; Vol. 9, pp 798–804.
19. M. Malik, T. Patel. Database Security - Attacks and Control Methods. *Int. J. Inf. Sci. Tech.* **2016**, 6 (1/2), 175–183.
20. M.C. Enache. Data Analysis with Pandas. *Annals of Dunarea de Jos University of Galati. Fascicle I. Economics and Applied Informatics*. 2019, pp 69–74.
21. N.C. Rajasekar, C.O. Imafidon. Exploitation of Vulnerabilities in Cloud-Storage. *Gstf Int. J. Comput.* **2011**, 1 (2).
22. W.K. Hauger, M.S. Olivier. The state of database forensic research; University of Pretoria Pretoria, South Africa, **2015**.
23. S. Zawoad, A.K. Dutta, R. Hasan. Towards Building Forensics Enabled Cloud Through Secure Logging-as-a-Service. *IEEE Trans. Dependable Secur. Comput.* **2016**, 13 (2), 148–162.
24. P.Huey, S. Jeloka "Oracle ® Database Security Guide 12c Release 1 (12.1) E48135-19."
25. C. Bellman, P.C. van Oorschot. Systematic analysis and comparison of security advice as datasets. *Comput. Secur.* **2023**, 124, 102989.
26. S.D. Sanap, V. More. Design of efficient S-box for Advanced Encryption Standard. *J. Integr. Sci. Technol.* **2022**, 10 (1), 39–43.
27. D.O. Embarak. Data Analysis and Visualization Using Python; Apress, Berkeley, CA, **2018**.
28. C. Nwankwo, H. Wimmer, L. Chen, J. Kim. Text Classification of Digital Forensic Data. In *11th Annual IEEE Information Technology, Electronics and Mobile Communication Conference, IEMCON 2020*; Electronics, **2020**; pp 661–667.
29. H. Stepanek. The Future of pandas. In *Thinking in Pandas*; Apress, Berkeley, CA, **2020**; pp 157–169.
30. N. Yadav, V. Yadav. Software reliability prediction and optimization using machine learning algorithms: A review. *J. Integr. Sci. Technol.* **2023**, 11 (1), 457.
31. M.E. Schüle, L. Scalerandi, A. Kemper, T. Neumann. Blue Elephants Inspecting Pandas Inspection and Execution of Machine Learning Pipelines in SQL. In *Advances in Database Technology - EDBT*; **2023**; Vol. 26, pp 40–52.
32. V. Roussev, C. Quates, R. Martell. Real-time digital forensics and triage. *Digit. Investig.* **2013**, 10 (2), 158–167.
33. S. Sachdeva, A. Ali. Machine learning with digital forensics for attack classification in cloud network environment. *Int. J. Syst. Assur. Eng. Manag.* **2022**, 13, 156–165.