Article

Not peer-reviewed version

Digital Forensics Readiness in Big Data Wireless Networks: A Novel Framework and Incident Response Script for Linux-Hadoop Environments

Cephas Mpungu *, Carlisle George, Glenford Mapp

Posted Date: 23 July 2024

doi: 10.20944/preprints202407.1803.v1

Keywords: Wireless networks; digital forensics; digital forensics readiness; incident response; big data; Hadoop

Article

Digital Forensics Readiness in Big Data Wireless Networks: A Novel Framework and Incident Response Script for Linux-Hadoop Environments.

Cephas Mpungu *, Carlisle George and Glenford Mapp

Middlesex University, C.George@mdx.ac.uk (C.G.); G.Mapp@mdx.ac.uk (G.M.)
* Correspondence: CM1677@live.mdx.ac.uk; Tel.: +447487550360

Abstract: The surge in big data and analytics has catalysed the proliferation of cybercrime, largely driven by organisations' intensified focus on gathering and processing personal data for profit, often overlooking security considerations. Hadoop and its derivatives are prominent platforms for managing big data, however, investigating security incidents within Hadoop environments poses intricate challenges due to scale, distribution, data diversity, replication, component complexity, and dynamicity. This paper proposes a digital forensics readiness framework and an incident response script for Linux-Hadoop systems, streamlining preliminary investigations. The framework offers a novel approach to digital forensics in the domains of big data and Hadoop environments. A prototype implementing important aspects of the DFR framework was developed and evaluated through comprehensive functionality and usability testing. The results demonstrated robust performance and efficacy.

Keywords: wireless networks; digital forensics; digital forensics readiness; incident response; big data; Hadoop

1. Introduction

In today's data-driven landscape, the Hadoop platform and its derivatives such as Apache Spark, Apache Hive, and Cloudera have emerged as the cornerstone of big data storage and analytics [1,3,22]. With their ability to efficiently manage vast amounts of data across distributed clusters, these platforms have become the preferred choice for most organisations seeking to harness the power of big data for insights and decision-making [21]. From e-commerce giants to healthcare providers, the adoption of Hadoop-based solutions has proliferated across various sectors, fuelling the expansion of business intelligence (BI) capabilities and transforming industries.

BI, facilitated by the analysis of large datasets, has revolutionised how organisations operate, enabling them to uncover valuable insights and optimise processes [16]. For instance, retail companies utilise customer data to personalise marketing campaigns, while financial institutions leverage data analytics to detect fraudulent activities. Such examples underscore the critical role of big data platforms like Hadoop in driving innovation and competitiveness across diverse domains.

Moreover, the exponential growth of big data has also led to an alarming increase in cybercrime[19,23] and data privacy issues[4,14]. As organisations amass significant volumes of sensitive data, they become prime targets for cybercriminals aiming to exploit weaknesses in data storage and processing systems [23]. This trend can exacerbate data privacy concerns, particularly because most organisations often prioritise data analytics for BI over transparency and comprehensive data protection efforts [4]. Furthermore, the intricate nature of Hadoop infrastructures may obscure these risks, making it challenging for digital forensics practitioners to effectively investigate security incidents within these environments [8].

To address these gaps, this research proposes a comprehensive digital forensics readiness (DFR) framework tailored for both the cloud and on-premises environments. DFR is defined as the proactive steps and preparations taken by an organisation to effectively perform digital investigations when

 doi:10.20944/preprints202407.1803.v1

faced with an incident such as a cyberattack, security breach, or legal investigation [2,12,20]. The implementation framework consists of a standard Hadoop cluster running on a Linux operating system (OS), configured with BI data nodes and two forensic nodes. The first forensic node is dedicated to evidential data collection whilst the second one is configured to capture all relevant security policy management and compliance information, guaranteeing an audit trail and simplifying digital investigations. Nevertheless, it is crucial to recognise that this research does not delve into the specifics of implementing security policy configurations. The proposed framework will serve as a foundational tool that organisations can tailor to their specific needs and requirements.

Additionally, a novel incident response script (IR-script) was developed utilising the powerful scripting capabilities of the Linux Bash shell, standard Linux commands, and Hadoop commands relevant to DFR. Linux scripts help automate tasks, manage system operations, and facilitate various administrative activities. The IR-script offers a robust solution for first responders initiating investigations on Linux-Hadoop Cluster setups. By combining automation with meticulous evidential data collection, the script empowers investigators to swiftly gather essential preliminary artefactual information required for effective forensic analysis. Its user-friendly design caters to both seasoned investigators and those with limited knowledge of Linux and Hadoop cluster environments, ensuring accessibility and usability in diverse investigative scenarios.

The primary objective of this research is to introduce a standard and customisable DFR framework designed for big data wireless networks, alongside a customisable IR-script specifically designed for Linux-Hadoop systems. Due to resource and time constraints in setting up and evaluating the entire wireless big data environment, this research paper focuses solely on developing and evaluating the effectiveness of the IR-script. To evaluate its efficacy, functionality, and usability, tests were conducted on two separate Hadoop systems: Amazon Web Services (AWS) and Oracle VirtualBox (VB).

The rest of this paper is organised as follows: Section 2 Summarises the focus of this research and introduces the research questions. Section 3 presents a thorough literature review, examining existing research and identifying gaps that this research aims to address. Section 4 details the research methodology employed in this study, providing a comprehensive overview of the approaches and techniques utilised to achieve the research objectives. Section 5 discusses the proposed system, including its design, architecture, and the theoretical underpinnings that inform its development. Section 6 elaborates on the prototype configuration and evaluation, describing the implementation process, testing environments, and the results obtained from the evaluations. Finally, Section 7 concludes the paper, summarising the key findings, implications, and potential areas for future research.

2. Research Focus

The focus of this research was to propose a comprehensive approach to enforcing DFR within a standard Linux-Hadoop cluster environment. The proposed framework emphasises the collection and preservation of digital forensic artefacts from both the underlying Linux operating system and the Hadoop cluster. Additionally, a novel incident response script was developed, offering a robust solution for first responders initiating investigations on Linux-Hadoop cluster setups.

The research addressed two primary research questions (RQs):

- **RQ1**: How can DFR be effectively implemented in an existing Linux-Hadoop big data wireless network without disrupting its operations?
- **RQ2**: Can digital forensics investigations within a Linux-Hadoop cluster be simplified for both experienced and inexperienced investigators, considering the various configurations of different Hadoop environments?

3. Literature Review

This section focuses on an extensive examination of existing research within the fields of digital forensics, digital forensics readiness, big data, business intelligence, Linux, and Hadoop environments. Guided by the research questions raised in Section 2, the objective is to gather insights

 doi:10.20944/preprints202407.1803.v1

and identify gaps in current knowledge and practices and to inform the identification of requirements for the proposed framework and the IR-script.

3.1. Background

In the rapidly evolving landscape of information technology, the advent of big data has revolutionised the way organisations manage and analyse vast volumes of data [3,5,16,21]. According to Phillip Russom [16], big data analytics denotes the use of advanced techniques on large data sets, combining the vast scale of big data with sophisticated analytics to drive a major trend in BI.

Big data technologies [21], particularly within wireless networks, present both opportunities and challenges that make wireless networks inherently more susceptible to attacks compared to their wired counterparts, due to factors such as signal interception and unauthorised access [10,12,25]. As organisations increasingly rely on big data for critical operations, ensuring the security and integrity of these networks becomes paramount.

Security incidents and data privacy issues within big data ecosystems have garnered significant attention in recent years [14]. The sheer volume, variety, and velocity [16] of data in big data environments make them attractive targets for cybercriminals seeking to exploit vulnerabilities and compromise sensitive information. Furthermore, the prioritisation of BI needs and profits over personal data privacy by some big tech organisations remains a cause for concern [4]. Examples of big tech organisations that have faced criticism in the past for prioritising BI needs and profits over personal data privacy include Facebook, Google, Amazon, Apple and Microsoft [4]. Incidents such as unethical practices [4], data breaches, unauthorised access, and insider threats pose significant risks to organisations, highlighting the importance of implementing robust security measures and proactive monitoring mechanisms [14,19,23].

DFR plays a crucial role in mitigating the impact of security incidents and ensuring auditable accountability within organisations [17] as well as big data environments. DFR is derived from the term digital forensics. According to Sachowski [17] (pp 11), digital forensics is denoted as scientific methodologies, principles, and techniques applied to law to ensure the admissibility of digital evidence in a court of law. The researcher also discussed Locard's exchange principle [17] (pp 11-12). The principle states that anything or anyone interacting with an environment (including digital environments) takes something or leaves behind something (evidence). A forensically ready digital environment (DFR) would thus be well suited to capture this evidence to aid in future investigations.

DFR involves setting up policies, procedures, and technical systems that allow for the swift and accurate collection, preservation, and analysis of digital evidence [12,17]. By proactively preparing for potential security breaches and establishing forensic capabilities, organisations can effectively investigate and respond to incidents, thereby minimising damage and preserving evidential data for legal proceedings.

Aside from security breaches, Sachowski [17] (pp 65-66) discussed other "multiple business scenarios where digital forensics can be applied to manage business risk". These include cybercrime impact validation, solving disputes (internal and external), compliance adherence, electronic discovery (e-discovery) support, and support for contractual and commercial agreements. However, conducting digital forensics investigations within complex big data ecosystems, particularly those built on Apache Hadoop, still presents unique challenges as discussed by the researchers Asim et al.[3], Harshany et al.[8], Oo [15] and Thanekar et al.[24]

Apache Hadoop's distributed file system (HDFS) has emerged as the most preferred platform for processing and analysing big data [3,9] due to its scalability, fault tolerance, and cost-effectiveness. Its ecosystem of tools and frameworks, including Apache Spark, offers unparalleled capabilities for distributed computing and data processing. However, the distributed nature of Hadoop environments, coupled with the sheer volume of data stored across multiple nodes, complicates digital investigations [8,15]. Extracting, correlating, and analysing data from disparate sources within a Hadoop cluster requires specialised skills and tools, making forensic investigations a daunting task.

Additionally, the Linux OS is the most preferred choice for Hadoop deployments [1,22] due to its robust performance, security features, and compatibility with open-source software [11,13]. Linux

 doi:10.20944/preprints202407.1803.v1

offers a stable and customisable environment that aligns well with the needs of large-scale data processing. Its native support for Hadoop's underlying technologies, such as the Java Runtime Environment (JRE) and file system utilities, ensures seamless integration and optimal performance. Moreover, the open-source nature of Linux complements the Hadoop ecosystem, allowing organisations to tailor their infrastructure without the constraints of proprietary software. This synergy between Linux and Hadoop enhances the efficiency and reliability of big data operations, further solidifying their dominance in the field of data analytics and processing.

In this context, this research focuses on implementing DFR within a Linux-Hadoop environment. However, it is also important to acknowledge that Hadoop can run on Windows and other Unix-like systems, even if these platforms are less commonly used in production environments.

3.2. Review of Related Work

Given the complexity of conducting digital investigations within Linux-Hadoop clusters, Thanekar et al [24] conducted a study titled "A Study on Digital Forensics in Hadoop." Their work highlighted the importance of understanding Hadoop's internal mechanisms for digital forensics. It emphasised the relevance of different files generated during Hadoop's processes for forensic analysis. The study focused on leveraging these files for digital forensics and described their roles. Using the open-source tool Autopsy, the study demonstrated efficient methods for identifying key files relevant to Hadoop digital forensics.

Additionally, Harshany et al [8] focused on extracting forensic artefacts from HDFS metadata. Their study explored the effectiveness of using specific metadata from the HDFS data storage layer to reconstruct file system operations and link data to its physical storage locations. By mapping this data, evidence can be prioritised and targeted for preservation or further analysis.

The motivation for the current research paper builds upon the work of Thanekar et al. [24] and Harshany et al. [8] to propose an enhanced Linux-Hadoop DFR framework. It is important to note that the researchers focused on extracting digital forensics artefacts from the Hadoop environment with less focus on the underlying Linux OS. However, in Linux-Hadoop cluster setups, the confidentiality, integrity, and availability (CIA) of a Hadoop cluster big data environment may also be compromised through the underlying Linux OS without necessarily altering HDFS metadata. This can be done by exploiting various vulnerabilities and attack vectors specific to the OS, such as privilege escalation, direct disk access, and manipulation of non-HDFS components like MapReduce and YARN. In that regard, it is important that the effective implementation of DFR within a Linux-Hadoop environment should always consider both the Linux OS and Hadoop environments.

3.3. Conclusion

The reviewed literature underscores the critical importance of ensuring security and DFR in big data environments, particularly within Linux-Hadoop clusters. The advent of big data has transformed organisational data management and analysis, presenting both significant opportunities and challenges. Big data wireless networks, being more vulnerable to attacks, necessitate robust security and DFR measures. Key findings of the literature review highlighted the increasing challenges posed by big data environments on forensic investigations, the prevalent use of Hadoop and its derivatives (like Apache Spark) for BI tasks, and the need for proactive DFR measures. The review also highlighted the predominant use of Linux-based environments for Hadoop installations due to their stability and security.

As discussed above, previous research studies by [8] and [24] have highlighted the role of HDFS metadata in forensic investigations and the complexities involved in such analyses. This research builds on these findings to propose a holistic DFR framework for Linux-Hadoop clusters. The framework aims to enhance the collection and preservation of forensic artefacts, ensuring organisations can effectively respond to security incidents and maintain the CIA of their data.

4. Research Methodology

 doi:10.20944/preprints202407.1803.v1

The research methodology adopted for this research paper entails two phases:

1. Prototype development.
2. Prototype evaluation.

A detailed discussion of the three phases is as follows:

4.1. Prototype Development

According to [6], different types of prototyping include rapid prototyping, slow, probing (throwaway), exploratory (proof of concept), developmental (live), incremental, and evolutionary. Each of these types of prototyping serves different purposes and approaches to developing and testing prototypes. The exploratory (proof of concept) method was selected for the development of this research prototype, followed by a functional and usability evaluation. This method was chosen because it allows for the preliminary demonstration of the feasibility and viability of a proposed idea, concept or technology. Proof of concept (POC) prototyping denotes the preliminary demonstration of the feasibility and viability of a proposed idea, concept, or technology. It involves developing a prototype to test key functionalities and validate assumptions before committing to full-scale development [6,7,18].

For this research, the focus was on developing and testing the IR-script, a critical tool for extracting and summarising key locations of digital forensics artefacts within a standard Linux-Hadoop environment during the initial stage of a digital forensics investigation. A comprehensive network diagram of the framework and a prototype implementing important aspects of the DFR framework were developed as discussed in Section 5.

4.2. Prototype Testing and Evaluation

To evaluate the robustness and compatibility of the prototype, particularly its IR-script, functionality and usability testing were chosen to ensure that the IR-script performs its intended functions accurately and is user-friendly and efficient in practical scenarios. Testing was conducted within two virtual environments: AWS using Ubuntu Linux and Oracle VB using CentOS Linux. This approach mirrors the autonomous functional and usability evaluation methods used by Thanekar et al. [24]. Employing these two environments not only facilitates a thorough evaluation of the IR-script's performance across varied platforms but also validates the autonomous functional and usability testing approach. This methodology enhances the reliability and generalisability of the findings by providing unbiased insights into the user experience. Furthermore, the source code of the IR-script will be made open-source to encourage further improvement by other researchers.

5. Proposed Framework

5.1. Requirements Analysis

To ensure that the system effectively meets the needs of digital forensics investigations, a comprehensive requirements analysis was conducted, guided by previous work [3,8,12,14,17,19,23,24]. This analysis was crucial to identify and define the required functionalities and features that the proposed system must possess to address the specific challenges and requirements of DFR in big data environments. Additionally, the analysis also considered four key factors discussed below:

- Architecture: Research on DFR and big data frameworks informed the identification of necessary system components and their functionalities.
- Operational needs: The practical needs of digital forensics investigations, such as data collection, secure storage, efficient processing, and access control, were key drivers in defining the system requirements.
- Technological capabilities: Leveraging existing technologies like Hadoop and Apache Spark, and integrating advanced hardware like graphical processing units (GPUs), ensured that the system could meet the performance and efficiency demands of both BI and forensics analytics.

 doi:10.20944/preprints202407.1803.v1

- Compliance and security: Ensuring compliance with legal standards and maintaining robust security measures were paramount in defining the system requirements.

For the proposed system, seven key DFR requirements were identified and are outlined as follows:

1. Data Collection and Storage: The system must be capable of capturing and securely storing network access logs, firewall logs, IDS logs, SIEM logs, and Hadoop cluster data. It should centralise the storage of all relevant forensic data to facilitate easy management and analysis.
2. Network Segmentation and Security: Network segmentation and security of both the big data network and its DFR segment must be implemented.
3. Efficient Data Processing: The system must leverage technologies that effectively enforce both BI and forensics analytics, ensuring efficient processing and analysis of large data volumes.
4. Metadata Management: The system should provide efficient metadata management through a secure centralised node, crucial for precise and efficient forensic analysis.
5. Access Control and Compliance: Strict access control measures must be enforced to ensure that only authorised personnel can interact with the forensic data. Furthermore, the system should log all security policies and maintain an up-to-date terms and conditions big data collection policy.
6. Time Synchronisation: Effective synchronisation with the network's network time protocol (NTP) server is essential to adhere to digital forensics admissibility requirements.
7. Resource Allocation: The forensic nodes must have sufficient storage and processing capabilities to handle the volume of data collected.

5.2. Framework Overview

The proposed big data DFR framework, as shown in Figure 1, incorporates essential components of a standard big data wireless network, including a router, access points, a firewall, Security Information and Event Management (SIEM) systems, Intrusion Detection Systems (IDS), and a robust big data storage and processing framework. Notably, the big data framework central to this research is Hadoop. To ensure effective segmentation of network systems, a VLAN (Virtual Local Area Network) switch(s) is integrated into the framework.

VLAN switches enhance network security and enforcement of security policies. They do this by logically isolating traffic and facilitating the logging of segmented DFR data directly into the Hadoop cluster forensic nodes. This integration allows for efficient capture, aggregation, and analysis of vast amounts of forensic data, improving the detection, investigation, and response capabilities within the big data wireless network.

Moreover, as depicted in Figure 1, by placing the BI DataNodes and forensic nodes (DFR node and BI compliance node) on separate network segments (i.e. BI segment and DFR segment respectively), VLANs ensure the security of the Forensic nodes, thereby providing a secure environment for storing and processing forensic data. It ensures that sensitive forensic data is securely stored and managed independently from regular data processing activities, thus preventing unauthorised access and potential data breaches. This approach not only bolsters the overall security of the big data environment but also optimises the storage and retrieval processes within the Hadoop cluster. This setup is also beneficial for the effective storage of big data within the designated BI nodes of the cluster. The proposed framework is depicted in Figure 1 below and is further discussed in section 5.3.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 July 2024 doi:10.20944/preprints202407.1803.v1

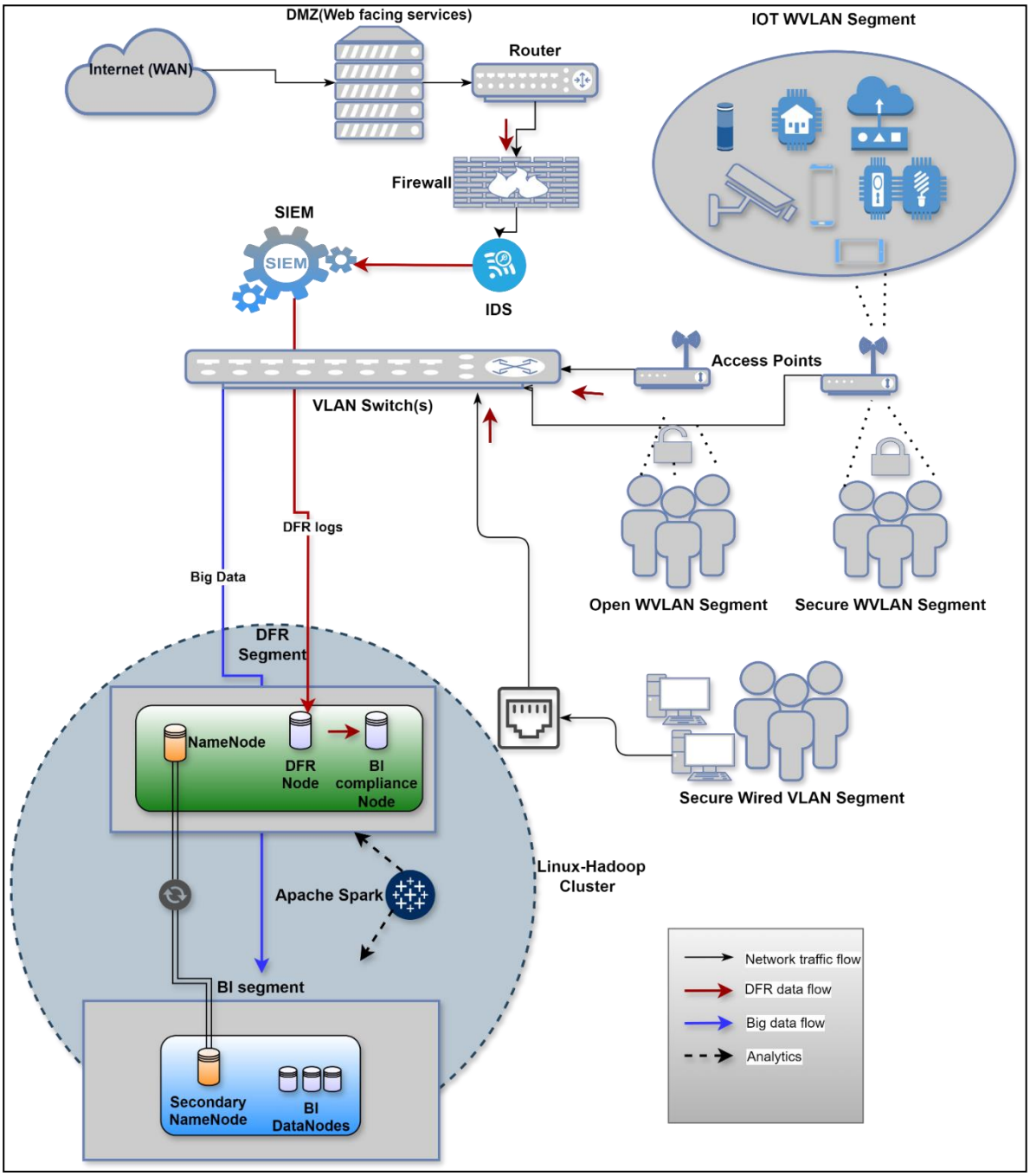

Figure 1. A Linux-Hadoop DFR framework for big data wireless networks.

This research paper aims to address the RQs raised in section 2: Research Focus:

RQ 1: How can DFR be effectively implemented on an existing Hadoop big data wireless network without disrupting its operations?

RQ 2: Can digital forensics investigations within a Hadoop cluster be simplified for both experienced and inexperienced investigators, considering the various configurations of different Hadoop environments?

The research questions are answered in the subsequent sections.

5.3. Research Question 1 Insights

To address RQ 1, the proposed framework recommends integrating DFR on the same wireless network where a company's BI operations are already running. Specifically, it recommends adding

 doi:10.20944/preprints202407.1803.v1

two dedicated forensic nodes (DFR node and BI compliance node) to the existing Hadoop cluster. As shown in Figure 1, these nodes are segmented from the BI DataNodes but strategically placed on the same network segment as the cluster's NameNode. This placement ensures low-latency access to the metadata and file system information stored within the NameNode, which is crucial for efficient forensic analysis. It also reduces the time needed to retrieve and process metadata, thus accelerating forensic tasks. Additionally, it is crucial to synchronise the NameNode and forensic nodes effectively with the NTP server to adhere to digital forensics admissibility requirements. Adjustments to the NTP settings may be necessary to ensure alignment with these requirements.

Effective metadata management is crucial for Hadoop's HDFS, offering significant advantages for digital forensics investigations. HDFS relies on a central master server, the NameNode, to handle all distributed file system metadata. This centralised management provides a comprehensive overview of the file system's structure, including file and directory attributes, data storage location mapping, and namespace hierarchy. Such detailed metadata facilitates precise and efficient forensic analysis, enabling investigators to trace and reconstruct data with greater accuracy and speed.

Furthermore, this setup enforces secure access control by ensuring that only authorised personnel can interact with the DFR segment, enhancing the security and integrity of the forensic data. By dedicating specific nodes for forensic purposes, the setup also mitigates resource contention with regular data processing tasks, ensuring that forensic analyses can be conducted without performance degradation. This approach not only optimises the efficiency of forensic operations but also maintains the performance of the ongoing BI operations [8].

Additionally, the logic of Apache Spark, which was initially configured for BI analytics, is then reconfigured to also perform forensics intelligence (FI) analytics as shown in Figure 1. This dual configuration of Apache Spark enables it to handle both BI and forensic workloads effectively. By leveraging Spark's in-memory processing capabilities and parallel computing framework [21], the system can efficiently process and analyse large volumes of data for both BI and FI. This reconfiguration ensures that the forensic analytics processes are as robust and efficient as the BI analytics, providing a comprehensive solution that supports both operational and forensic data analysis requirements.

To further enhance the performance and efficiency of the analytics processes, GPUs should be utilised. While Hadoop traditionally uses central processing units (CPUs) for its data processing tasks, integrating GPUs into a Hadoop environment can offer significant performance improvements for certain types of workloads, particularly those that benefit from parallel processing, making them ideal for both BI and forensic analytics.

To add the new forensic nodes without disrupting the big data operations, the following steps are followed:

- First, reconfigure the wireless network to place the forensic nodes on the same segment as the NameNode using VLANs or subnetting to achieve network segmentation.
- Next, prepare the Forensic nodes by installing Linux OS, Java, and Hadoop, setting up SSH access, and updating host resolution.
- Install and configure Hadoop on the forensic nodes, and subsequently synchronise configuration files from the NameNode. Adjust these configurations accordingly to accurately reflect their role as forensic nodes.
- Start necessary Hadoop services on the forensic nodes, such as DataNode and any additional services related to forensic readiness.
- Finally, verify the integration by ensuring the forensic nodes are recognised by the NameNode and can communicate efficiently, and by testing data retrieval and forensic tool operations to confirm the setup.

The first node (DFR node) is dedicated to the collection and secure storage of wireless network digital evidential data. This includes network access logs, firewall access logs, IDS logs, SIEM logs, and the entire Hadoop cluster information. By centralising all relevant forensic data, this node will facilitate easier management and analysis while ensuring the secure storage of sensitive information, which is crucial for any potential investigations. However, careful attention must be given to resource

 doi:10.20944/preprints202407.1803.v1

allocation, ensuring this node has sufficient storage and processing capabilities to handle the volume of data.

The second node (BI compliance node) is configured to capture essential functions related to security policy management and compliance within the big data environment. Its primary role includes logging all current and historical security policies of the system. This encompasses policies governing activities such as personal data anonymisation, data encryption, and password management. By maintaining these logs, the node provides a comprehensive record of the security measures implemented at any point in time. These records are invaluable for auditing, ensuring compliance with regulations, and supporting forensic investigations when necessary.

Furthermore, the BI compliance node should also maintain an up-to-date terms and conditions big data collection policy. This should capture the rules and regulations regarding the collection, storage, and processing of big data (especially personal data) within the system. By ensuring that these logging policies are current and actively enforced, organisations can better account for the management of their big data in accordance with legal and ethical standards, reducing the risk of data breaches or misuse.

As earlier discussed, the proposed Linux-Hadoop configuration leverages Apache Spark's capabilities [21] for both BI and FI. This is because Apache Spark is already utilised for BI tasks, such as analysing customer churn, targeted advertising, and maximising sales. The same infrastructure can be extended to process forensic data, perform anomaly detection, and generate necessary reports from the collected evidential data.

To optimise the implementation, a phased strategy is recommended for smooth integration. Firstly, the DFR node is set up to collect data from a subset of the network while monitoring and validating the overall configuration and performance. Gradual expansion to cover the entire network then ensues. Next, the establishment of the BI compliance node and subsequent migration of all relevant data and policies is done. Finally, the integration of BI and FI processes using Apache Spark to ensure seamless operation is done.

Thorough testing and validation are essential. Functional testing ensures the new nodes operate correctly without negatively impacting the existing BI Hadoop cluster, while usability testing ensures the system meets the needs of both BI and FI teams. However, these tests are beyond the scope of this research. Lastly, adequate training programs for IT and security personnel, alongside comprehensive documentation, will support the effective use and maintenance of the new nodes.

5.4. Research Question 2 Insights

To address RQ 2, an IR-script was developed. This script offers a comprehensive solution for streamlining digital forensics investigations when executed either on a Linux-Hadoop cluster's NameNode or an image of the NameNode. It is designed to be user-friendly for investigators of all experience levels, regardless of the diverse configurations of Linux-Hadoop environments. Developed using Bash shell scripting, standard Linux utilities, and Hadoop-specific commands, the script automates the retrieval and cataloguing of crucial forensic data, making it easier to initiate investigations by abstracting the complexities of Hadoop environments. Additionally, the script is a valuable tool for network administrators and other IT personnel in their day-to-day tasks and troubleshooting, as will be demonstrated in its usability and functional evaluation.

The script facilitates the retrieval of essential digital forensics details, including Linux OS information, uptime, load averages, promiscuous mode state, potential rootkits, file system configuration, mount points, partitions, open ports, disk usage, network configuration, Linux log directories, and critical Hadoop cluster specifics such as Hadoop version, configuration files, HDFS version, Hadoop log directories, environment variables, SSH configuration, DNS configuration, detailed cluster metrics, and running processes. Additionally, it includes functionalities to scan the Linux-Hadoop environment for installed IDS systems and extract log information. This automated approach streamlines the investigation process, ensuring consistency and accuracy in data collection across various Hadoop configurations.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 July 2024 doi:10.20944/preprints202407.1803.v1

Moreover, the IR-script's output format, comprising well-titled documents and execution logs, enhances readability, comprehensibility, and transparency. By organising the retrieved information and redirecting it into accessible text documents, titled 'Hadoop_Info.txt', 'OS_Info.txt', and script_log.txt, investigators can efficiently navigate through the collected data, regardless of their familiarity with Hadoop environments. The configuration code for the IR-script is given in Appendix A.

6. Prototype Configuration and Evaluation

As previously mentioned in the Introduction and Research Methodology sections, due to resource and time constraints, setting up and evaluating a prototype of the entire wireless Hadoop big data environment was not feasible for this research. Instead, this paper focused solely on developing and assessing the effectiveness of a prototype of the IR-script. To evaluate its efficacy, the research conducted functional and usability testing on two separate Linux-Hadoop systems: Amazon Web Services (AWS) with Ubuntu Linux and Oracle VB with CentOS Linux. The setup was such that the AWS cluster primarily uses a Command Line Interface (CLI), while the Oracle VB one provides both Graphical User Interface (GUI) and CLI interfaces. The network configurations on both systems are depicted in Figure 2 below.

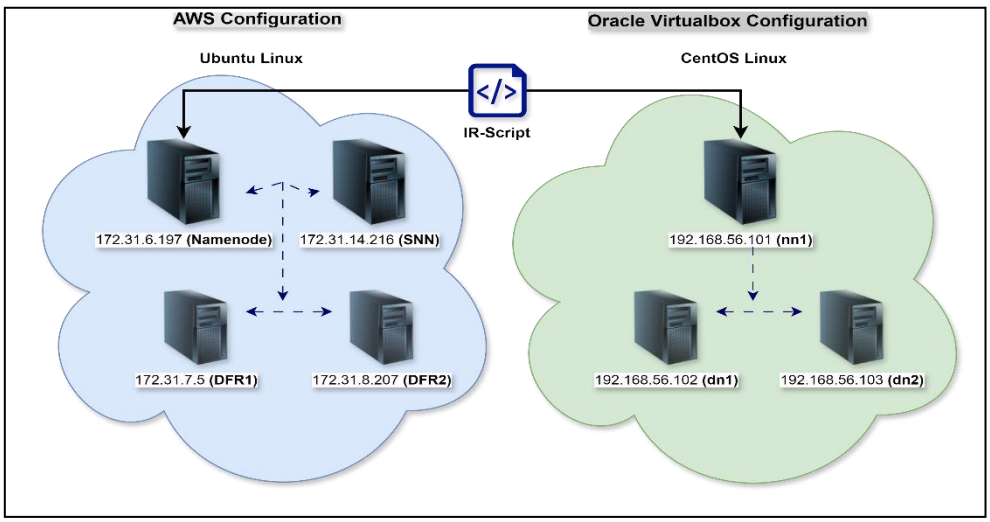

Figure 2. Network configuration of test environments.

6.1. AWS Configuration.

The first configuration was set up on AWS cloud and constituted an Ubuntu-Hadoop Cluster with the following configuration:

- OS details- Ubuntu Server 22.04 LTS
- Virtualisation environment- Amazon Web Services (AWS).
- Four EC2 instances namely NameNode, Secondary NameNode(SSN) and two data nodes (DFR1 and DFR2)
- All instances were configured to run on the same subnet to ensure network connectivity and efficient data transfer within the Hadoop cluster.
- A Key Pair was created to enable a secure connection to the AWS instances. This key pair was used to SSH into the instances.
- A security group was configured to allow necessary traffic:
- IAM Configuration: IAM role was configured and attached to instances.
- Java and Hadoop were downloaded and installed on each instance.
- Password-less SSH set up on all EC2 nodes.
- The /home/ubuntu/.bashrc shell script was modified in all the nodes to configure the required environment for Java and Hadoop.

 doi:10.20944/preprints202407.1803.v1

- Hadoop core files on the NameNode were configured and copied to the other nodes using the scp command.
- The Hadoop Filesystem on the NameNode was then formatted and Hadoop daemons started.
- Lastly, the IR-script was developed within the Ubuntu environment, tested and improved iteratively. The results are discussed in the evaluation section 4.1 below.

6.2. Oracle VB Configuration

The second configuration was set up on Oracle VB and constituted a CentOS-Hadoop Cluster with the following configuration:

- OS details- CentOS 8
- Virtualisation environment- Oracle VB installed on Windows 11 Dell laptop.
- Three virtual machines (VMs) were created namely nn1 and two data nodes (dn1 and dn2)
- All VMs were configured to run on the same internal network using host-only adaptors for internal communication and network address translation (NAT) adaptors for effective external communication.
- Password-less SSH was then set up on all VMs.
- The /home/hadoop/.bashrc shell script was modified in all the nodes to configure the required environment for Java and Hadoop.
- Hadoop core files on the NameNode (nn1) were configured and copied to the other nodes.
- HDFS on nn1 was formatted and Hadoop daemons started.
- The IR-script was then executed and the results were examined as discussed in the evaluation section below.

Detailed configuration documents, including screenshots, for both test environments are readily available.

6.3. IR-script Overview.

The provided hadoop-forensics.sh script is a comprehensive Bash script designed to automate the collection of essential digital forensics information from a Linux OS with a Hadoop cluster setup. Its primary purpose is to simplify digital forensics investigations by automating the retrieval and cataloguing of crucial forensic data on a Hadoop cluster's NameNode or an image of the NameNode. The script is user-friendly, making it accessible for investigators of all experience levels. Additionally, it is valuable for network administrators and IT personnel in their day-to-day tasks.

Key features of the script include gathering detailed OS information, specific details related to the Hadoop cluster, status and logs of various Intrusion Detection Systems (IDS) and performing scans using rootkit detection tools. For OS information, the script collects data such as OS details, kernel details, users with sudo privileges, logged-in users, system uptime and load averages, disk usage and partition information, network configuration, and default Linux log files.

Regarding Hadoop cluster information, the script gathers data on Hadoop configuration files, HDFS version, Namenodes and secondary Namenodes, Hadoop log files, cluster environment variables, detailed Hadoop cluster metrics, running Hadoop and Java processes, and cluster nodes. This ensures comprehensive coverage of all essential Hadoop-specific details that might be relevant in a forensic investigation.

The script also checks the status and logs of various IDS systems, including Snort, Suricata, and OSSEC, if they are installed on the system. This provides a quick overview of the security monitoring tools and their current state. Additionally, the script includes rootkit detection capabilities, utilising tools like chkrootkit, rkhunter, and unhide to perform scans and gather results, which is crucial for identifying potential rootkit infections.

The script is structured with several key functions. The usage() function displays usage information for the script, while the handle_error() function manages errors by displaying an error message and exiting the script. The command_exists() function checks if a command exists on the system. The script validates the output directory to ensure it is provided and exists, and defines paths for output files: OS_Info.txt, Hadoop_Info.txt, and script_log.txt. All script output is redirected to a log file for comprehensive logging, ensuring that the entire process is documented.

 doi:10.20944/preprints202407.1803.v1

To add structure to the output files, the script includes functions like add_section_header() and add_command_output() to insert section headers and command outputs into the respective files. The main functions for gathering information are get_os_info() for OS details, get_hadoop_info() for Hadoop cluster specifics, get_ids_info() for IDS information, and get_rootkit_info() for rootkit detection.

To use the script, the user must provide the path to the output directory as an argument, as demonstrated in the usage example: ./hadoop-forensics.sh /path/to/output_directory. Upon execution, the script systematically gathers OS information, Hadoop cluster information, IDS information, and rootkit detection results, concluding with a completion message. This automation ensures consistency and accuracy in data collection, making the forensic investigation process more efficient and reliable across different Hadoop configurations.

6.4. IR-Script Evaluation.

The evaluation of the IR-script was done using functional and usability approaches.

Functionality in this context refers to the ability of the script to perform its intended tasks accurately and efficiently. In evaluating the IR-script, functionality evaluation was conducted to ensure that the script successfully collected and organised the necessary forensics artefacts. The evaluation was done in two distinct Linux-Hadoop environments: AWS with Ubuntu Linux and Oracle VB with CentOS. A key objective was to assess the script's effectiveness across different Linux-Hadoop configurations. The script was evaluated for its capability to gather comprehensive OS and Hadoop-specific information, including OS details, disk usage, network configuration, and Hadoop cluster metrics.

The functionality evaluation confirmed that the IR-script met 99% of the expected outcomes in both environments, demonstrating its robustness and reliability across different setups. Each command executed as intended, and the output files contained all the relevant information, verifying the script's functional accuracy. However, the intrusion detection system (IDS) check was inconclusive because there were no IDS systems installed within the test environment, as this was outside the scope of this research. Documents have been attached showing the detailed configuration of both environments as well as the functionality test results within both environments. Table 1 below shows a summary of the functional evaluation in both Linux-Hadoop environments.

Table 1. Functional Evaluation Results.

IR-script Generated Files	Forensics artefacts	AWS	Oracle VB	Results Summary
OS_Info.txt	1. Operating System Details 2. Users with Sudo Privileges 3. Logged-In Users 4. Uptime and Load Averages 5. Disk Usage and Partition Information 6. Mount Points and Filesystem Config 7. Network Configuration 8. Default Linux Log File List 9. Promiscuous mode check 10. Detect IDS and check logs 11. Check for rootkits	√ √ √ √ √ √ √ √ √ ✗ √	√ √ √ √ √ √ √ √ √ ✗ √	All the target OS digital forensics artefacts on both systems were effectively retrieved. However, IDS checks were inconclusive (see conclusion section for details)
Hadoop_Info.txt	1. Hadoop Configuration Files 2. HDFS Version	√ √ √	√ √ √	All the target Hadoop cluster digital

 doi:10.20944/preprints202407.1803.v1

	3. NameNodes and Secondary NameNodes 4. Default Hadoop Log File List 5. Hadoop Cluster Network Config 6. Cluster Environment Variables Config 7. Detailed Hadoop Cluster Info/Metrics 8. Current Running Hadoop Processes 9. Hadoop Cluster Nodes	√ √ √ √ √ √ √	√ √ √ √ √ √ √	forensics artefacts on both systems were effectively retrieved.
Script_Log.txt	1. Time-stamped script operations account	√	√	Time-stamped log files on both systems were generated.

Usability pertains to the ease with which users (investigators) can effectively utilise the script to achieve their goals. Usability evaluation of the IR-script was performed to assess how user-friendly the script is for digital forensic investigators and network administrators. The evaluation revealed that while the script is thorough in its data collection, it can only be executed via the Command Line Interface (CLI). This limitation may pose a challenge for users who are not familiar with CLI operations or who prefer graphical interfaces. Despite this, the script performed fairly well, as it organised the collected data into clear and accessible text documents.

However, enhancing the script with a more user-friendly interface or additional guidance could improve its usability, making it more accessible to a broader range of users, including those with less technical expertise. Documents have been attached showing the usability steps and results of IR-script execution within both environments.

7. Conclusions

In the research titled "Digital Forensics Readiness in Big Data Wireless Networks: A Novel Framework and Incident Response Script for Linux-Hadoop Environments," the proposed framework and IR-Script solve the problem of efficiently gathering essential preliminary information required for digital forensics investigations in Linux-Hadoop environments. This task, typically performed manually, is time-consuming, prone to errors, and requires significant expertise in both Linux and Hadoop systems.

Currently, investigators address this problem through manual data collection, executing numerous commands to obtain system and Hadoop-specific details. This manual approach is inefficient and can result in inconsistent data collection, affecting the accuracy and reliability of forensic investigations.

The evaluation of the IR-Script confirms that the prototype performs its intended function effectively. Tested in two different Linux-Hadoop environments, i.e., AWS with Ubuntu Linux and Oracle VB with CentOS, the script successfully collected and organised comprehensive Linux OS system and Hadoop-specific information. However, the IDS check was inconclusive because there were no IDS systems installed within the test environment, as this was outside the scope of this research. Additionally, despite being primarily CLI-based, the script's thorough documentation and organised output enhance its usability. Therefore, the IR-Script automates and simplifies the initial stages of digital forensics investigations, proving to be a novel and valuable tool for investigators and network administrators.

 doi:10.20944/preprints202407.1803.v1

Future work will involve testing the IR-Script on other Linux distributions, integrating AI to enhance its capabilities, and developing a user-friendly interface using Python to improve the accessibility and usability of IR-Script functionalities. These advancements aim to further streamline digital forensics investigations and broaden the script's applicability across diverse environments.

Author Contributions: Conceptualization, C.M., C.G. and G.M.; methodology, C.M. and C.G; software, C.M.; validation, C.M.; investigation, C.M.; resources, C.M.; data curation, C.M.; writing—original draft preparation, C.M.; writing—review and editing, C.M., C.G. & G.M.; visualization, C.M., C.G. and G.M.; supervision, C.G. and G.M.; project administration, C.G. and G.M.; funding acquisition, C.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding and was self-funded.

Data Availability Statement: Data are available upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

```
# Script: hadoop-forensics.sh
# This script is designed for first responders to collect essential preliminary information required to initiate a digital forensics investigation on a Linux OS with a Hadoop Cluster setup.
# Author: Cephas Mpungu

# Function to display usage information
usage() {
    echo "Usage: $0 <output_directory>"
    echo "Example: $0 /path/to/output_directory"
}

# Error handling function
handle_error() {
    echo "Error: $1" >&2
    exit 1
}

# Function to check if a command exists
command_exists() {
    command -v "$1" >/dev/null 2>&1
}

# Check if output directory argument is provided
if [ $# -ne 1 ]; then
    usage
    exit 1
fi

# Check if the output directory exists
if [ ! -d "$1" ]; then
    handle_error "Output directory does not exist."
fi

# Output file paths
OUTPUT_DIR="$1"
OS_INFO_FILE="$OUTPUT_DIR/OS_Info.txt"
HADOOP_INFO_FILE="$OUTPUT_DIR/Hadoop_Info.txt"
LOG_FILE="$OUTPUT_DIR/script_log.txt"
```

 doi:10.20944/preprints202407.1803.v1


```
# Redirect all script output to log file
exec > >(tee -a "$LOG_FILE")
exec 2>&1

# Function to add section headers to output files
add_section_header() {
    echo "$1" >> "$2"
    echo "==================================" >> "$2"
}

# Function to add command output to output files
add_command_output() {
    echo "$1" >> "$2"
    echo >> "$2"
}

# Start of script
echo "Starting Hadoop cluster information gathering script"
echo "$(date) - Starting Hadoop cluster information gathering script" >> "$LOG_FILE"

# Detect the operating system
if [ -f /etc/os-release ]; then
    . /etc/os-release
    OS=$NAME
else
    OS=$(uname -s)
fi

# Function to get OS-specific information
get_os_info() {
    add_section_header "Operating System Information" "$OS_INFO_FILE"

    if command_exists lsb_release; then
        add_command_output "Operating System: $(lsb_release -d | cut -f 2)" "$OS_INFO_FILE"
    elif [ -f /etc/os-release ]; then
        add_command_output "Operating System: $(grep -w NAME /etc/os-release | cut -d= -f2 | tr -d '\"')" "$OS_INFO_FILE"
    elif [ -f /etc/centos-release ]; then
        add_command_output "Operating System: $(cat /etc/centos-release)" "$OS_INFO_FILE"
    else
        add_command_output "Operating System: $(uname -s)" "$OS_INFO_FILE"
    fi

    add_command_output "$(uname -a)" "$OS_INFO_FILE"
    add_command_output "Users With Sudo Privileges:" "$OS_INFO_FILE"
    add_command_output "$(getent group sudo | cut -d: -f4 | tr ',' '\n')" "$OS_INFO_FILE"
    add_command_output "$(getent group wheel | cut -d: -f4 | tr ',' '\n')" "$OS_INFO_FILE"
    add_command_output "Logged In Users:" "$OS_INFO_FILE"
    add_command_output "$(w)" "$OS_INFO_FILE"
    add_command_output "Uptime:" "$OS_INFO_FILE"
    add_command_output "$(uptime | awk '{print $3, substr($4, 1, length($4)-1)}')" "$OS_INFO_FILE"
    add_command_output "Load Averages:" "$OS_INFO_FILE"
```

 doi:10.20944/preprints202407.1803.v1


```
        add_command_output "$(cat /proc/loadavg | awk '{print "Load averages (1min/5min/15min): " $1 ", " $2 ", " $3}')" "$OS_INFO_FILE"
        add_command_output "Disk Usage:" "$OS_INFO_FILE"
        add_command_output "$(df -h)" "$OS_INFO_FILE"
        add_command_output "Partition Information:" "$OS_INFO_FILE"
        add_command_output "$(lsblk)" "$OS_INFO_FILE"
        add_command_output "Mount Points:" "$OS_INFO_FILE"
        add_command_output "$(mount)" "$OS_INFO_FILE"
        add_command_output "Filesystem Configuration:" "$OS_INFO_FILE"
        add_command_output "$(cat /etc/fstab)" "$OS_INFO_FILE"
        add_command_output "Network Configuration:" "$OS_INFO_FILE"

        if command_exists ifconfig; then
            add_command_output "$(ifconfig)" "$OS_INFO_FILE"
        else
            add_command_output "$(ip addr)" "$OS_INFO_FILE"
        fi

        add_command_output "$(netstat -tuln)" "$OS_INFO_FILE"
        add_command_output "$(cat /etc/hosts)" "$OS_INFO_FILE"
    # Check for promiscuous mode
        add_section_header "Promiscuous Mode Check" "$OS_INFO_FILE"
        if command_exists ip; then
            add_command_output "IP link Results:" "$OS_INFO_FILE"
            add_command_output "$(ip link | grep PROMISC)" "$OS_INFO_FILE"
        else
            add_command_output "IP command not found, using alternative:" "$OS_INFO_FILE"
            add_command_output "$(ifconfig | grep PROMISC)" "$OS_INFO_FILE"
        fi
        add_command_output "$(ls /var/log/)" "$OS_INFO_FILE"
    }

    # Function to get Hadoop-specific information
    get_hadoop_info() {
        add_section_header "Hadoop Cluster Information" "$HADOOP_INFO_FILE"
        add_command_output "Hadoop Configuration Files:" "$HADOOP_INFO_FILE"
        add_command_output "$(find $(dirname $(which hdfs))/../etc/hadoop -type f -name '*.xml')" "$HADOOP_INFO_FILE"
        add_command_output "Hadoop HDFS Version:" "$HADOOP_INFO_FILE"
        add_command_output "$(hdfs version)" "$HADOOP_INFO_FILE"
        add_command_output "All Cluster Namenodes:" "$HADOOP_INFO_FILE"
        add_command_output "$(hdfs getconf -namenodes)" "$HADOOP_INFO_FILE"
        add_command_output "All Cluster Secondary Namenodes:" "$HADOOP_INFO_FILE"
        add_command_output "$(hdfs getconf -secondaryNameNodes)" "$HADOOP_INFO_FILE"
        add_command_output "Default Hadoop Log File List:" "$HADOOP_INFO_FILE"
        add_command_output "$(ls -l $(dirname $(which hdfs))/../logs)" "$HADOOP_INFO_FILE"
        add_command_output "Hadoop Cluster Network Config:" "$HADOOP_INFO_FILE"
        add_command_output "$(cat /etc/hosts)" "$HADOOP_INFO_FILE"
        add_command_output "Hadoop Cluster Environment Variables Config:" "$HADOOP_INFO_FILE"
        add_command_output "$(hdfs envvars)" "$HADOOP_INFO_FILE"
        add_command_output "Detailed Info/Metrics About Hadoop Cluster:" "$HADOOP_INFO_FILE"
        add_command_output "$(hdfs dfsadmin -report)" "$HADOOP_INFO_FILE"
        add_command_output "Current Running Hadoop Processes:" "$HADOOP_INFO_FILE"
```

 doi:10.20944/preprints202407.1803.v1


```
        add_command_output "$(ps aux | grep hadoop)" "$HADOOP_INFO_FILE"
        add_command_output "$(ps aux | grep java)" "$HADOOP_INFO_FILE"
        add_section_header "Hadoop Cluster Nodes" "$HADOOP_INFO_FILE"
        add_command_output "Namenodes:" "$HADOOP_INFO_FILE"
        add_command_output "$(cat $(dirname $(which hdfs))/../etc/hadoop/masters)" "$HADOOP_INFO_FILE"
        add_command_output "Datanodes:" "$HADOOP_INFO_FILE"
        add_command_output "$(cat $(dirname $(which hdfs))/../etc/hadoop/slaves)" "$HADOOP_INFO_FILE"
    }

    # Function to get IDS information
    get_ids_info() {
        add_section_header "Intrusion Detection Software (IDS) Information Logs" "$OS_INFO_FILE"
        local ids_detected=0

        if command_exists snort; then
            add_command_output "Snort Status:" "$OS_INFO_FILE"
            add_command_output "$(systemctl status snort)" "$OS_INFO_FILE"
            add_command_output "Snort Log Files:" "$OS_INFO_FILE"
            add_command_output "$(ls /var/log/snort/)" "$OS_INFO_FILE"
            ids_detected=1
        fi

        if command_exists suricata; then
            add_command_output "Suricata Status:" "$OS_INFO_FILE"
            add_command_output "$(systemctl status suricata)" "$OS_INFO_FILE"
            add_command_output "Suricata Log Files:" "$OS_INFO_FILE"
            add_command_output "$(ls /var/log/suricata/)" "$OS_INFO_FILE"
            ids_detected=1
        fi

        if command_exists ossec; then
            add_command_output "OSSEC Status:" "$OS_INFO_FILE"
            add_command_output "$(systemctl status ossec)" "$OS_INFO_FILE"
            add_command_output "OSSEC Log Files:" "$OS_INFO_FILE"
            add_command_output "$(ls /var/ossec/logs/)" "$OS_INFO_FILE"
            ids_detected=1
        fi

        if [ "$ids_detected" -eq 0 ]; then
            add_command_output "No IDS detected." "$OS_INFO_FILE"
        fi
    }

    # Function to get rootkit detection information
    get_rootkit_info() {
        add_section_header "Rootkit Detection" "$OS_INFO_FILE"
        local rootkit_detected=0

        if command_exists chkrootkit; then
            add_command_output "chkrootkit Scan Results:" "$OS_INFO_FILE"
            add_command_output "$(chkrootkit)" "$OS_INFO_FILE"
            rootkit_detected=1
```

 doi:10.20944/preprints202407.1803.v1


```
    fi

    if command_exists rkhunter; then
        add_command_output "rkhunter Scan Results:" "$OS_INFO_FILE"
        add_command_output "$(rkhunter --check --skip-keypress)" "$OS_INFO_FILE"
        rootkit_detected=1
    fi

    if command_exists unhide; then
        add_command_output "unhide Scan Results:" "$OS_INFO_FILE"
        add_command_output "$(unhide proc)" "$OS_INFO_FILE"
        add_command_output "$(unhide sys)" "$OS_INFO_FILE"
        rootkit_detected=1
    fi

    if [ "$rootkit_detected" -eq 0 ]; then
        add_command_output "No rootkits detected." "$OS_INFO_FILE"
    fi
}


# Execute all functions
get_os_info
get_hadoop_info
get_ids_info
get_rootkit_info

# End of script
echo "Hadoop cluster information gathering completed"
```

References

1. Ahmed, Hameeza, Muhammad Ali Ismail, and Muhammad Faraz Hyder. "Performance optimization of hadoop cluster using linux services." In *17th IEEE International Multi Topic Conference 2014*, pp. 167-172. IEEE, 2014
2. Akinbi, Alex Olushola. "Digital forensics challenges and readiness for 6G Internet of Things (IoT) networks." *Wiley Interdisciplinary Reviews: Forensic Science* 5, no. 6 (2023): e1496.
3. Asim, Mohammed, Dean Richard McKinnel, Ali Dehghantanha, Reza M. Parizi, Mohammad Hammoudeh, and Gregory Epiphaniou. "Big data forensics: Hadoop distributed file systems as a case study." *Handbook of Big Data and IoT Security* (2019): pp.179-210
4. Beloume, Amy. 2023. "The Problems of Internet Privacy and Big Tech Companies." The Science Survey. February 28, 2023. https://thesciencesurvey.com/news/2023/02/28/the-problems-of-internet-privacy-and-big-tech-companies/.
5. Elgendy, Nada, and Ahmed Elragal. "Big data analytics: a literature review paper." In Advances in Data Mining. Applications and Theoretical Aspects: 14th Industrial Conference, ICDM 2014, St. Petersburg, Russia, July 16-20, 2014. Proceedings 14, pp. 214-227. Springer International Publishing, 2014.
6. Evaluating Prototypes." https://www.tamarackcommunity.ca/hubfs/Resources/Tools/Aid4Action%20Evaluating%20Prototypes%20Mark%20Cabaj.pdf. (accessed on 20 May 2024).
7. Häggman, Anders, Tomonori Honda, and Maria C. Yang. "The influence of timing in exploratory prototyping and other activities in design projects." In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 55928, p. V005T06A023. American Society of Mechanical Engineers, 2013.
8. Harshany, Edward, Ryan Benton, David Bourrie, and William Glisson. 2020. "Big Data Forensics: Hadoop 3.2.0 Reconstruction." *Forensic Science International: Digital Investigation* 32 (April): 300909. https://doi.org/10.1016/j.fsidi.2020.300909.
9. Joshi, Pramila. "Analyzing big data tools and deployment platforms." *International Journal of Multidisciplinary Approach & Studies* 2, no. 2 (2015): pp. 45-56.

 doi:10.20944/preprints202407.1803.v1

10. Kumar, Yogendra, and Vijay Kumar. "A Systematic Review on Intrusion Detection System in Wireless Networks: Variants, Attacks, and Applications." *Wireless Personal Communications* (2023): pp. 1-58.
11. Messier, Ric, and Michael Jang. *Security strategies in Linux platforms and applications*. Jones & Bartlett Learning, 2022.
12. Mpungu, Cephas, Carlisle George, and Glenford Mapp. "Developing a novel digital forensics readiness framework for wireless medical networks using specialised logging." In *Cybersecurity in the Age of Smart Societies: Proceedings of the 14th International Conference on Global Security, Safety and Sustainability, London, September 2022*, pp. 203-226. Cham: Springer International Publishing, 2023.
13. Nazeer, Sumat, Faisal Bahadur, Arif Iqbal, Ghazala Ashraf, and Shahid Hussain. "A Comparison of Window 8 and Linux Operating System (Android) Security for Mobile Computing." *International Journal of Computer (IJC)* 17, no. 1 (2015):pp. 21-29
14. Olabanji, Samuel Oladiipo, Oluseun Babatunde Oladoyinbo, Christopher Uzoma Asonze, Tunboson Oyewale Oladoyinbo, Samson Abidemi Ajayi, and Oluwaseun Oladeji Olaniyi. "Effect of adopting AI to explore big data on personally identifiable information (PII) for financial and economic data transformation." *Available at SSRN 4739227* (2024).
15. Oo, Myat Nandar. "Forensic Investigation on Hadoop Big Data Platform." PhD diss., MERAL Portal, 2019.
16. Russom, Philip. "Big data analytics." *TDWI best practices report, fourth quarter* 19, no. 4 (2011): pp. 1-34.
17. Sachowski, Jason. *Implementing Digital Forensic Readiness: From Reactive to Proactive Process*, 2nd ed.; Boca Raton, Fl: Crc Press, Taylor & Francis Group, NW, U.S. 2019
18. Sadia, Halima. 2022. "10 Prototype Testing Questions a Well-Experienced Designer Need to Ask." Webful Creations. November 28, 2022. https://www.webfulcreations.com/10-prototype-testing-questions-a-well-experienced-designer-need-to-ask/.
19. Sarker, Iqbal H. AI-Driven Cybersecurity and Threat Intelligence: Cyber Automation, Intelligent Decision-Making and Explainability. Springer Nature, 2024.
20. Shoderu, Gabriel, Stacey Baror, and Hein Venter. "A Privacy-Compliant Process for Digital Forensics Readiness." In *International Conference on Cyber Warfare and Security*, vol. 19, no. 1, pp. 337-347. 2024.
21. Singh, Dilpreet, and Chandan K. Reddy. "A survey on platforms for big data analytics." *Journal of big data* 2 (2015): pp. 1-20.
22. Taylor, Ronald C. "An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics." *BMC bioinformatics* 11 (2010): pp 1-6
23. Thakur, Manikant. "Cyber security threats and countermeasures in digital age." *Journal of Applied Science and Education (JASE)* 4, no. 1 (2024): pp. 1-20
24. Thanekar, Sachin Arun, K. Subrahmanyam, and A. B. Bagwan. "A study on digital forensics in Hadoop." *Indonesian Journal of Electrical Engineering and Computer Science* 4, no. 2 (2016): 473-478.
25. Yaman, Okan, Tolga Ayav, and Yusuf Murat Erten. "A Lightweight Self-Organized Friendly Jamming." *International Journal of Information Security Science* 12, no. 1 (2023): pp. 13-20.