

Mitigating Self-Propagating Ransomware using Software Defined Networking

Haruna Saidu¹, Muhammad Aminu Ahmad², Ahmad Abubakar Aliyu³ and Saadatu Abdulkadir⁴
^{1,2,3,4} Department of Computer Science, Kaduna State University, Kaduna

Abstract

Self-propagating ransomware remains a major threat to network security because of its ability to spread rapidly across vulnerable hosts, encrypt critical files, and disrupt organizational operations. Existing host-based solutions often require installation on individual devices, while many network-based and SDN-based approaches mainly focus on detecting suspicious traffic or isolating entire infected hosts, which may disrupt legitimate services. This study developed an enhanced Software-Defined Networking-based mitigation technique for detecting and containing self-propagating ransomware through behavioral analysis and selective service-level isolation. The proposed technique monitors abnormal traffic patterns associated with ransomware propagation and uses a POX-based SDN controller to dynamically install OpenFlow rules that block infected services while allowing unaffected services on the same host to remain operational. The technique was evaluated in a virtualized SDN testbed using BadRabbit, WannaCry, and GandCrab ransomware samples under different threshold values. Performance was assessed using accuracy, precision, recall, F1-score, detection time, and containment time. The results showed that the proposed technique achieved detection accuracy of 95% for BadRabbit and 93% for both WannaCry and GandCrab. It also achieved strong F1-scores across the evaluated ransomware variants and reduced average containment time to 7 seconds compared with 30 seconds reported in the benchmark SDN-based method. These findings demonstrate that selective service-level isolation can provide faster ransomware containment while preserving network availability. The study contributes a practical SDN-based mitigation approach for real-time containment of self-propagating ransomware in networked environments.

Keywords: cybersecurity, ransomware mitigation, software-defined networking, service-level isolation, behavioral analysis, network security

1. Introduction

The rapid growth of Internet-enabled technologies has transformed how individuals, organizations, and governments store, access, and exchange information. Modern digital devices such as smartphones, tablets, servers, and personal computers now support essential services across education, banking, healthcare, governance, and business operations (Jawad & Ahmed, n.d.). As Internet adoption continues to increase globally, more users and organizations depend on connected systems for communication, data storage, financial transactions, and service delivery (Petrosyan, 2023). However, the expansion of Internet usage has also increased exposure to cyber threats and created more opportunities for cybercriminal activities (FBI, 2022). Cybercrime remains one of the most serious risks associated with digital transformation. It involves the use of computers, networks, and Internet-based systems to commit illegal activities, compromise data, disrupt services, or extort victims. Recent reports have shown that cybercrime continues to impose

significant financial and operational costs globally and locally, including in Nigeria (Cybercrime Magazine, 2025; Robinson, 2024). Among the different forms of cybercrime, ransomware has become particularly damaging because it directly targets data availability and business continuity. Ransomware is a type of malware that restricts access to systems or encrypts users' files and demands payment before access is restored. It is commonly categorized into crypto-ransomware and locker-ransomware. Crypto-ransomware encrypts files and demands payment for decryption, while locker-ransomware prevents users from accessing devices or applications. The use of cryptocurrencies for ransom payment often makes transactions difficult to trace, thereby increasing the attractiveness of ransomware to cybercriminals (Su et al., 2018). Ransomware campaigns have continued to evolve in sophistication, affecting different sectors and causing substantial financial losses (Ventura, 2018). Self-propagating ransomware presents an even greater threat because it combines encryption-based extortion with worm-like spreading behavior. Once a vulnerable host is infected, the ransomware can scan the network, identify other vulnerable systems, and spread rapidly without direct user interaction. This propagation capability can result in widespread network compromise within a short period, especially in environments with unpatched systems, weak segmentation, or limited real-time monitoring (Berrueta et al., 2019; Morato et al., 2018a). Existing ransomware mitigation approaches are generally classified into host-based and network-based techniques. Host-based solutions monitor processes, file system activities, application behavior, or registry changes on individual devices. Although they can provide detailed endpoint visibility, they must be deployed and maintained on each host. This makes them difficult to scale in large or heterogeneous networks. In addition, host-based tools may be bypassed, disabled, or delayed if the ransomware executes before detection mechanisms respond effectively (Morato et al., 2018a; Al-rimy et al., 2018). Network-based solutions monitor traffic patterns, ports, protocols, packet headers, and communication flows to identify suspicious behavior. These methods are useful because they can observe ransomware propagation across the network. However, many existing network-based approaches focus mainly on detection, alerting, or blocking known malicious addresses. They may not provide fast and selective containment of infected services, especially where ransomware uses new communication patterns or rapidly changes its propagation behavior (Al-rimy et al., 2018; Alsmadi & Zarour, 2017). Software-Defined Networking offers a promising approach for ransomware mitigation because it separates the control plane from the data plane and enables centralized network programmability. Through an SDN controller, network traffic can be monitored and flow rules can be dynamically installed on switches to allow, redirect, or drop suspicious traffic. This makes SDN suitable for real-time ransomware containment, especially in networks where rapid response is required to stop propagation (Alsmadi & Zarour, 2017; Aladaileh et al., 2020). Previous studies have demonstrated the usefulness of SDN in ransomware detection and mitigation. Akbanov et al. (2019) proposed an SDN-based mechanism for ransomware with worm-like behavior, using WannaCry as a case study. Their approach focused mainly on identifying ransomware communication patterns, especially communication with command-and-control infrastructure. However, it did not fully address selective blocking of propagation services within the local network. Similarly, Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021) proposed an SDN-based intrusion detection and prevention system for self-propagating ransomware such as BadRabbit, using modules such as deep packet inspection, ARP scanning detection, packet header inspection, honeypot, and SMB checking. Although their method

improved detection, it still largely emphasized detection and prevention modules rather than selective service-level isolation that preserves unaffected services on the infected host. Therefore, the major research problem addressed in this study is the lack of a fast and selective ransomware containment mechanism that can block infected services without unnecessarily isolating the entire host from the network. Full host isolation may stop ransomware spread, but it can also disrupt legitimate services running on the same host. In critical environments, this disruption can affect availability, productivity, and continuity of operations. Hence, there is a need for a mitigation approach that detects ransomware propagation behavior and applies targeted containment at the service level. This study develops an enhanced SDN-based ransomware mitigation technique that uses behavioral analysis and selective service-level isolation to contain self-propagating ransomware. Instead of fully disconnecting an infected host from the network, the proposed technique dynamically blocks the infected service, port, or suspicious traffic flow through SDN flow rules, while allowing unaffected services on the same host to continue operating.

2. Related Work

Ransomware detection and mitigation techniques are commonly classified into host-based, network-based, and SDN-based approaches. Host-based techniques monitor activities directly on endpoint devices, while network-based techniques analyze traffic behavior across the network. SDN-based approaches extend network-based detection by using centralized control and programmable flow rules to support dynamic mitigation. This section reviews these approaches and identifies the research gap addressed by this study.

2.1 Host-Based Detection

Host-based ransomware detection techniques operate on individual endpoint devices by monitoring file system activities, process behavior, registry changes, system calls, application programming interface calls, and other local indicators of compromise. These techniques are useful because ransomware typically performs observable activities on the host, such as file encryption, unusual file access, abnormal process execution, and unauthorized modification of user data. Rawaa et al. (2024) presented an Android ransomware detection approach using deep learning. The study used a feedforward neural network model with densely connected layers to classify ransomware instances and achieved high detection accuracy. Similarly, Mohamed et al. (2024) developed an Enhanced Joint Mutual Information method for early ransomware detection during the pre-encryption phase. Their method improved feature selection by ranking cryptographic and non-cryptographic application programming interface features, thereby enhancing the identification of ransomware behavior before encryption occurs. Abutu et al. (2024) proposed DeepCodeLock, a deep learning-based ransomware detection framework that integrates convolutional and recurrent neural network components to analyze behavioral signatures. The system was designed for real-time environments and achieved strong detection accuracy when compared with traditional signature-based and heuristic approaches. Panaras et al. (2024) introduced a cooperative clustering framework for proactive ransomware detection and mitigation using machine learning. Their work emphasized collaboration among multiple detection agents and demonstrated the usefulness of distributed behavioral analysis. Urooj et al. (2023) addressed behavioral drift in ransomware

detection through a weighted generative adversarial network framework, which generated synthetic samples and adapted feature weighting to improve early detection of evolving ransomware variants. Although host-based methods provide detailed visibility into endpoint behavior, they have limitations when applied to self-propagating ransomware. First, they must be installed, configured, and maintained on every endpoint in the network, which can be difficult in large or heterogeneous environments. Second, their effectiveness depends on the state and availability of the host; once ransomware disables security services or executes before detection, the endpoint may already be compromised. Third, host-based detection may respond too late to prevent lateral movement because self-propagating ransomware can scan and infect other systems within a short time. Therefore, while host-based methods are valuable for identifying local ransomware behavior, they are often insufficient as a standalone containment strategy for fast-spreading network-based ransomware.

2.2 Network-Based Detection

Network-based ransomware detection techniques focus on identifying malicious behavior by analyzing traffic flows, packet headers, communication frequency, port usage, entropy patterns, command-and-control communication, and abnormal scanning activities. Unlike host-based solutions, network-based approaches can observe propagation behavior across multiple devices without requiring installation on every endpoint. This makes them useful for monitoring lateral movement and detecting ransomware activity at the network level. Batalov et al. (2024) developed a hybrid ransomware detection framework that combined Isolation Forest for anomaly detection with Long Short-Term Memory networks for sequence modeling of network traffic. The approach was designed to identify deviations from normal traffic behavior and detect temporal patterns associated with ransomware attacks. Wiles et al. (2024) proposed a ransomware detection model using Generative Adversarial Networks to learn benign network traffic distributions and identify deviations caused by ransomware activity. Kayess et al. (2024) developed a hierarchical entropy-based framework for detecting ransomware in encrypted network traffic. The study used multi-layer entropy analysis to detect anomalous traffic patterns without requiring decryption. Gupret et al. (2024) proposed a dual-layer ransomware classification approach that combined opcode similarity with network traffic patterns, thereby integrating static and dynamic indicators. Williams et al. (2024) also applied entropy-based network traffic analysis to detect ransomware by observing changes in packet size, inter-arrival time, and source-destination communication patterns. Venne et al. (2024) introduced the Pattern-Entropy Segmentation Analysis framework, which uses real-time entropy analysis of network traffic to identify early-stage ransomware behavior. Anderson et al. (2024) examined ransomware command-and-control communication by tracing network traffic patterns to identify infrastructure used by ransomware operators. Lumazine et al. (2024) proposed a hybrid model that combined Convolutional Neural Networks and Isolation Forest for ransomware detection in network traffic. These studies demonstrate that network-based approaches are effective for observing ransomware behavior beyond individual endpoints. However, many of them focus mainly on detection, classification, or alert generation. In practical ransomware incidents, detection alone is not sufficient because self-propagating ransomware requires immediate containment to prevent lateral spread. Some network-based systems may identify suspicious activity but still depend on separate security tools or manual intervention to block the attack. Others block entire

hosts or known malicious communication channels, which may disrupt legitimate services or fail against unknown propagation patterns. Therefore, there remains a need for a network-based mitigation strategy that not only detects ransomware behavior but also actively blocks infected services in real time.

2.3 SDN-Based Ransomware Mitigation

Software-Defined Networking has introduced new possibilities for ransomware mitigation because it allows centralized visibility, programmable traffic control, and dynamic enforcement of security policies. In an SDN environment, the controller maintains a global view of the network and can install flow rules on switches to allow, deny, redirect, or isolate specific traffic. This makes SDN suitable for ransomware mitigation, especially where rapid containment is required. Akbanov et al. (2019) proposed an SDN-based ransomware detection and mitigation mechanism using WannaCry as a case study. Their work showed that SDN can be used to detect ransomware with worm-like propagation behavior and respond to malicious traffic. However, the approach focused primarily on communication patterns associated with command-and-control activities and did not sufficiently address selective service-level containment within an infected host. Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021) developed an SDN-based Intrusion Detection and Prevention System for self-propagating ransomware, particularly BadRabbit. Their system included modules such as Deep Packet Inspection, ARP Scanning Detection, Packet Header Inspection, Honeypot, and SMB Checker. The study demonstrated that some modules were effective in detecting ransomware activity with lower resource consumption and faster detection times than more intensive inspection methods. However, the approach still emphasized detection and prevention modules rather than selective isolation of only the infected service while maintaining unaffected host services. Umar et al. (2021) proposed an SDN-based mitigation strategy for Sodinokibi ransomware in cloud networks. Their approach combined static and dynamic malware analysis with an intrusion detection system integrated into an SDN environment. The study showed that SDN can improve the detection and mitigation of ransomware threats in cloud-based environments. However, like many existing SDN-based methods, the emphasis was on identifying and mitigating ransomware activity at a broader traffic or host level rather than preserving unaffected services through service-level isolation.

2.4 Comparative Summary and Research Gap

The reviewed studies show that host-based, network-based, and SDN-based techniques have contributed significantly to ransomware detection and mitigation. Host-based methods provide detailed endpoint visibility and can detect ransomware activities such as file encryption, abnormal process behavior, and suspicious application programming interface calls. However, they are limited by deployment complexity, endpoint dependency, and delayed response in fast-spreading ransomware incidents. For self-propagating ransomware, relying only on host-based detection may be inadequate because the ransomware can move laterally before endpoint-level responses are completed. Network-based methods improve visibility across the network and can detect propagation behavior, command-and-control communication, scanning activities, and abnormal traffic patterns. Nevertheless, many network-based techniques mainly classify or detect ransomware traffic without providing immediate and targeted containment. Detection without

active mitigation may not stop ransomware from spreading, particularly in high-speed network environments. SDN-based methods provide better mitigation potential because they allow centralized monitoring and dynamic flow-rule enforcement. However, existing SDN-based ransomware studies often focus on detecting suspicious communication, blocking traffic broadly, or isolating infected hosts. Full host isolation can prevent further propagation, but it may also interrupt legitimate services running on the infected host. This creates an availability problem, especially in environments where continuity of service is important. To address this gap, the present study proposes an SDN-based ransomware mitigation technique that uses behavioral analysis and selective service-level isolation. Unlike approaches that only detect ransomware or isolate the entire host, the developed technique dynamically blocks the infected service, port, or malicious traffic flow while allowing unaffected services on the same host to continue operating. In addition, the technique is evaluated using three ransomware variants: BadRabbit, WannaCry, and GandCrab. This multi-ransomware evaluation provides broader evidence of effectiveness than studies limited to a single ransomware case. The technique also demonstrates faster containment time when compared with the benchmark SDN-based approach, thereby improving ransomware response while preserving network availability.

3. Methodology

This study adopted an experimental research design to develop and evaluate an enhanced Software-Defined Networking-based technique for detecting and mitigating self-propagating ransomware. The experimental approach was considered suitable because previous ransomware mitigation studies have commonly evaluated detection and containment mechanisms in controlled SDN or virtualized network environments (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021; Umar et al., 2021). The use of a controlled testbed allowed the study to safely observe ransomware propagation behavior, detect abnormal network activities, and evaluate the effectiveness of the proposed service-level isolation technique without exposing external systems to malware-related risks. The methodology involved four major stages: design of the SDN-based ransomware mitigation architecture, implementation of the detection and containment mechanisms, configuration of the experimental testbed, and evaluation of the developed technique using selected ransomware variants. The evaluation focused on accuracy, precision, recall, F1-score, detection time, and containment time, which are commonly used performance measures in ransomware detection and network security evaluation studies (Alotaibi & Vassilakis, 2021; Urooj et al., 2023; Batalov et al., 2024).

3.1 Ransomware Mitigation Architecture

The architecture of the developed ransomware detection and mitigation technique was designed to use the centralized control and programmability of SDN to identify and contain ransomware propagation in real time as shown in Figure 1.

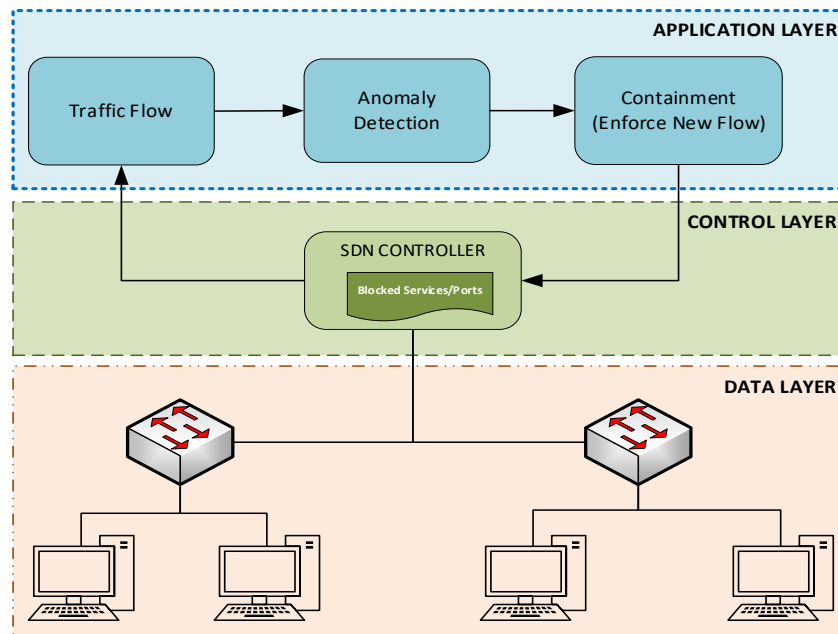


Figure 1: Architecture of the ransomware mitigation approach

SDN separates the control plane from the data plane, allowing the controller to maintain a global view of the network and dynamically enforce traffic rules through programmable switches (Alsmadi & Zarour, 2017; Aladaileh et al., 2020). This makes SDN suitable for ransomware mitigation because malicious flows can be blocked or redirected without manually reconfiguring individual network devices. The architecture consists of three major layers: the infrastructure layer, the control layer, and the application layer. The infrastructure layer consists of virtual hosts and Open vSwitch, which forwards traffic based on flow rules received from the SDN controller. The control layer consists of the POX controller, which manages network traffic, receives packet-in events, analyzes traffic behavior, and installs flow rules on the switch. Similar SDN-based controller-switch designs have been used in previous ransomware mitigation studies to support centralized monitoring and dynamic traffic control. The application layer contains the traffic monitoring, anomaly detection, and containment modules. These modules work together to identify suspicious traffic patterns and apply mitigation actions when ransomware-like behavior is detected. The traffic monitoring module collects flow-level information from the SDN switch, including source and destination IP addresses, MAC addresses, port numbers, protocol type, packet frequency, and connection attempts. These features were selected because ransomware propagation and command-and-control communication often produce observable network indicators such as repeated connection attempts, abnormal scanning behavior, unusual port usage, and suspicious communication flows. The main novelty of the architecture is its selective service-level isolation mechanism. Instead of isolating the entire infected host, the system identifies the suspicious service, port, or traffic flow and applies a targeted blocking rule. This differs from existing approaches that primarily detect ransomware traffic, monitor command-and-control communication, or block broader host-level activity (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). The purpose is to prevent ransomware propagation while preserving legitimate communication from unaffected services.

3.2 Detection Mechanism

The detection mechanism is based on behavioral analysis of network traffic. Self-propagating ransomware typically generates abnormal traffic patterns when scanning vulnerable hosts, attempting lateral movement, establishing command-and-control communication, or spreading through network services (Berrueta et al., 2019; Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). Therefore, the developed system monitors network behavior and compares observed traffic patterns against defined threshold values. The monitored indicators include repeated connection attempts from a single host to multiple destinations, unusual traffic frequency within a short time interval, suspicious use of vulnerable services or ports, abnormal SMB or HTTP communication patterns, repeated failed connection attempts, and traffic patterns associated with known ransomware propagation behavior. These indicators are consistent with network-level ransomware detection features used in previous studies on ransomware traffic analysis and SDN-based mitigation. When the alert score generated from these indicators exceeds the configured threshold, the traffic is classified as suspicious and passed to the containment module. The detection mechanism is summarized in Algorithm 1.

Algorithm 1: Ransomware Detection Algorithm

```
BEGIN

INITIALIZE monitoring interval

INITIALIZE alert score

INITIALIZE threshold value

WHILE monitoring interval is active DO

    CAPTURE traffic flow from SDN switch

    EXTRACT source IP, destination IP, source port, destination port,
    MAC address, protocol type, packet frequency, and connection pattern

    IF packet is a normal controller or management packet THEN

        IGNORE packet

    ELSE

        ANALYZE traffic behavior

        IF traffic shows suspicious propagation behavior THEN

            INCREASE alert score

            LOG suspicious activity
```

3.3 Threshold Selection and Tuning

Threshold values were used to determine the point at which observed traffic behavior should be classified as suspicious. In this study, three threshold values, namely 30, 50, and 100, were selected to evaluate the sensitivity of the detection mechanism under different ransomware propagation patterns. The threshold values were chosen to represent low, medium, and high detection sensitivity levels. A lower threshold allows the system to detect suspicious behavior earlier, but it may increase false positives because some benign traffic may be incorrectly classified as malicious. A higher threshold reduces false positives by requiring stronger evidence of malicious behavior before triggering containment, but it may delay response time. This threshold-response relationship is important in anomaly-based ransomware detection because overly sensitive detection may increase false alarms, while delayed detection may allow fast-spreading ransomware to propagate further. The tuning process involved observing ransomware and benign traffic patterns in the virtualized testbed and evaluating how each threshold affected detection accuracy, false positives, false negatives, detection time, and containment time. The selected threshold values were therefore used to examine the trade-off between sensitivity and response speed across BadRabbit, WannaCry, and GandCrab ransomware variants. Similar threshold-based experimentation has been used in network security and SDN-based ransomware studies to evaluate how detection parameters influence system performance.

3.4 Service-Level Isolation Using SDN Flow Rules

The containment mechanism applies selective service-level isolation through dynamic SDN flow-rule updates. When the detection module identifies ransomware-related traffic, the containment module extracts the relevant traffic attributes, including the source IP address, destination IP address, source MAC address, destination MAC address, protocol type, and infected service port. These attributes are then used to create a flow rule that matches only the suspicious service or traffic flow. The SDN controller installs an OpenFlow rule on the switch to drop traffic that matches the identified malicious service or flow. OpenFlow-based rule enforcement allows SDN controllers to define packet-matching conditions and corresponding actions, such as forwarding, dropping, or redirecting traffic. In this study, the rule is designed to drop only ransomware-related service traffic rather than disconnecting the entire host. For example, if ransomware propagation is detected through a specific service port, the controller installs a rule that drops traffic from the infected host on that port while allowing unrelated traffic from the same host to continue. This approach extends prior SDN-based ransomware mitigation work by introducing targeted service-level containment instead of broad host-level isolation or detection-only monitoring. The general structure of the service-level isolation rule is as follows:

```
START
  RECEIVE Alert from Detection System
  WHILE TRUE DO
    GET Data (Source IP, Port, MAC)
    IF Source IP is in Block List THEN
      CONTINUE
    ELSE
      ADD Source IP to Block List
      ADD Source MAC to Block List
      ADD Port to Block List
      DROP PACKETS Port, Source IP and MAC Address
      SEND Blocking Information to SDN Controller
    END IF
    IF TTL is Exceeded THEN
      DELETE Entry from Block List
    END IF
    UPDATE SDN Rules
    LOG Blocked IPs and MACs
  END WHILE
END
```

This selective blocking approach differs from full host isolation. In full host isolation, all network traffic from the infected host is blocked, which may disrupt legitimate services. In the proposed service-level isolation approach, only the suspicious service or flow is blocked. This preserves network availability and reduces the operational impact of ransomware containment.

3.5 Experimental Testbed

The developed mitigation technique was evaluated in a virtualized SDN testbed using VirtualBox, Open vSwitch, a POX controller, Wireshark, and selected ransomware samples. Virtualized testbeds are commonly used in ransomware and SDN security studies because they allow controlled experimentation, repeatability, traffic observation, and safe restoration of infected systems after each test. The testbed consisted of six virtual machines, each configured for a specific role in the experiment. VM1 and VM2 were configured as vulnerable Windows 7 SP1 hosts. These machines represented potential victim systems that could be infected by ransomware. VM3, also configured with Windows 7 SP1, was used to launch the ransomware infection. VM4 was configured with Ubuntu 12.04 and hosted the POX controller. VM5 hosted Open vSwitch and served as the SDN switch for managing network traffic. VM6 was configured with Kali Linux and used to provide DNS and HTTP services required for observing ransomware communication behavior.

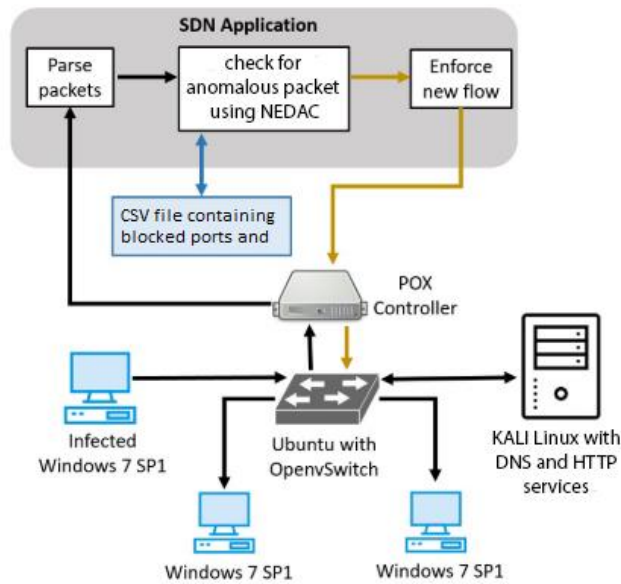


Figure 2: SDN Application

3.6 Justification for Operating System Selection

Windows 7 SP1 was selected for the vulnerable host machines because the ransomware variants used in the experiment, particularly WannaCry and BadRabbit, are associated with vulnerabilities and propagation behaviors commonly observed in older Windows environments (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). Using Windows 7 SP1 therefore allowed the study to reproduce realistic infection and propagation conditions in a controlled laboratory setting. Ubuntu 12.04 was selected for hosting the SDN controller and Open vSwitch because of its compatibility with the POX controller and SDN experimentation tools used in the study. Kali Linux was included because it provides network analysis, DNS, HTTP, and security testing utilities that were useful for setting up supporting services and monitoring ransomware communication behavior. The operating systems were selected strictly for experimental compatibility and controlled evaluation; they were not intended to represent current best-practice production environments.

3.7 Experimental Procedure

The evaluation was conducted using three ransomware variants: BadRabbit, WannaCry, and GandCrab. Each ransomware variant was tested under three threshold values: 30, 50, and 100. This produced a total of nine experimental scenarios. Multi-scenario testing was used to evaluate whether the developed technique could maintain performance across different ransomware families and threshold configurations, rather than being limited to a single ransomware case study as seen in some previous SDN-based works (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). The experiments were conducted in two stages. In the first stage, ransomware samples were executed without the developed mitigation technique to observe baseline infection and propagation behavior. In the second stage, the experiments were repeated with the developed SDN-based mitigation technique installed on the POX controller. During each experiment, network traffic was monitored

and recorded using Wireshark. The detection module analyzed the traffic and generated alerts when suspicious behavior exceeded the configured threshold. The containment module then installed service-level flow rules to block the infected service or malicious traffic flow.

Table 1: Summary of Evaluation Experiments

Experiments	Ransomware	Threshold Values
1	BadRabbit	30
2		50
3		100
4	WannaCry	30
5		50
6		100
7	GandCrab	30
8		50
9		100

3.8 Evaluation Metrics

The performance of the developed mitigation technique was evaluated using confusion matrix-based metrics. The metrics include accuracy, precision, recall, and F1-score. These metrics are widely used in ransomware detection and intrusion detection studies because they measure the ability of a system to correctly classify malicious and benign traffic. Accuracy measures the proportion of correctly classified ransomware and benign traffic instances. Precision measures the proportion of traffic classified as ransomware that was actually ransomware. Recall measures the proportion of actual ransomware traffic correctly detected by the system. F1-score provides a harmonic mean of precision and recall. The equations used are presented as follows:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

$$F1-Score=2*\frac{Precision*Recall}{Precision+Recall} \quad (4)$$

Where TP represents true positives, TN represents true negatives, FP represents false positives, and FN represents false negatives. In addition to the confusion matrix metrics, detection time and containment time were recorded. Detection time refers to the time taken by the system to identify ransomware-related behavior after suspicious activity begins. Containment time refers to the time taken to apply SDN flow rules and block the infected service after detection.

3.9 Assumptions and Limitations of the Experimental Setup

The experimental setup was based on several assumptions. First, it was assumed that ransomware propagation behavior could be observed through measurable network indicators such as packet frequency, connection attempts, port activity, and communication patterns. This assumption is consistent with previous network-based ransomware detection studies that used traffic behavior to identify suspicious activity (Batalov et al., 2024; Williams et al., 2024; Venne et al., 2024). Second, it was assumed that the POX controller and Open vSwitch could enforce flow rules quickly enough to contain malicious traffic in real time, as demonstrated in SDN-based mitigation studies. Third, it was assumed that the selected ransomware samples represented relevant self-propagating ransomware behavior for evaluating the developed mitigation technique. Despite these assumptions, the study has some limitations. The experiments were conducted in a virtualized environment, which may not fully represent the complexity, traffic volume, and heterogeneity of large enterprise networks. The study used three ransomware variants, namely BadRabbit, WannaCry, and GandCrab; therefore, the results may not generalize to all ransomware families. The threshold values were evaluated within the scope of the testbed and may require adjustment in real-world networks with different traffic patterns. In addition, the testbed used specific operating systems and SDN tools, which may affect reproducibility in environments with different configurations.

3.10 Ethical Handling of Malware Samples

The ransomware samples used in this study were handled strictly within an isolated and controlled laboratory environment. Controlled virtualized environments are commonly used in malware and ransomware research to prevent unintended spread and to allow safe restoration of infected machines after experimentation (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). The virtual machines used for malware execution were separated from external production networks to prevent accidental spread or unauthorized communication. The ransomware samples were used only for academic research and evaluation of the developed mitigation technique. No malware sample was distributed, shared, or executed outside the controlled environment. After each experiment, affected virtual machines were reverted to clean snapshots to remove infection traces and restore the testbed to a safe state. This ensured that the study followed responsible malware research practice and minimized risk to external systems.

4. Results and Discussion

This section presents the results obtained from the evaluation of the developed SDN-based ransomware mitigation technique. The evaluation was conducted using BadRabbit, WannaCry, and GandCrab ransomware variants under three threshold values: 30, 50, and 100. The results are presented using confusion matrix values, detection performance metrics, detection time, and containment time. The findings are also compared with the benchmark SDN-based approach of Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021).

4.1 Baseline Ransomware Propagation Without Mitigation

Before activating the developed mitigation technique, each ransomware variant was executed in the virtualized testbed to observe its baseline propagation behavior. The baseline experiment showed that BadRabbit successfully communicated with its command-and-control server and encrypted vulnerable hosts within five minutes. WannaCry successfully infected vulnerable hosts within ten minutes, while GandCrab infected vulnerable hosts within fifteen minutes. These baseline observations confirmed that the testbed was suitable for evaluating ransomware propagation and containment. They also demonstrated the need for rapid mitigation because self-propagating ransomware can compromise vulnerable systems within a short period when no active containment mechanism is deployed. The observed baseline behavior is consistent with previous studies that identified self-propagating ransomware as a high-risk threat due to its ability to spread quickly across vulnerable networked systems (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021).

4.2 Confusion Matrix Results

The confusion matrix results of the developed ransomware detection technique are presented in Table 2. The results show the number of true positives, true negatives, false positives, and false negatives recorded across the nine experimental scenarios.

Table 2: Confusion Matrix Results of Ransomware Detection

Experiments	Ransomware	Threshold	TP	TN	FP	FN
1	BadRabbit	30	70	20	3	7
2		50	65	25	2	8
3		100	80	30	1	5
4	WannaCry	30	80	30	4	10
5		50	75	35	3	12
6		100	85	40	2	8
7	GandCrab	30	85	35	5	10
8		50	80	40	3	12
9		100	90	45	2	8

For BadRabbit, the developed technique recorded true positive values of 70, 65, and 80 at threshold values of 30, 50, and 100 respectively. The false positive values decreased from 3 at threshold 30

to 1 at threshold 100. This indicates that a higher threshold improved the system's ability to reduce benign traffic misclassification. The false negative values also reduced at threshold 100, suggesting improved detection stability at the highest threshold. For WannaCry, the system recorded true positive values of 80, 75, and 85 across the three threshold values. The false positive values also decreased as the threshold increased, from 4 at threshold 30 to 2 at threshold 100. However, WannaCry recorded higher false negatives than BadRabbit, especially at threshold 50. This suggests that some WannaCry traffic patterns overlapped with benign traffic or did not consistently exceed the selected detection threshold. For GandCrab, the system recorded true positive values of 85, 80, and 90, while false positives decreased from 5 at threshold 30 to 2 at threshold 100. GandCrab had the highest false positive value at the lowest threshold. This indicates that some GandCrab-related traffic patterns were more difficult to distinguish from benign network activities when the detection sensitivity was high. Overall, the confusion matrix results show that the developed technique was able to detect ransomware traffic across all tested variants. The results also show that increasing the threshold generally reduced false positives and improved detection consistency. However, the relationship between threshold value and detection performance was not completely linear, indicating that ransomware behavior differs across variants and must be evaluated using multiple performance measures rather than accuracy alone.

4.3 Detection Performance

The detection performance of the developed mitigation technique is presented in Table 3. The performance was measured using accuracy, precision, recall, F1-score, detection time, and containment time.

Table 3: Ransomware Detection Performance

Experiments	Ransomware	Threshold	Accuracy	Precision	Recall	F1-Score	DT	CT
1	BadRabbit	30	90%	96%	91%	93%	7s	10s
2		50	90%	97%	89%	91%	5s	7s
3		100	95%	99%	94%	96%	3s	5s
4	WannaCry	30	89%	95%	89%	92%	6s	8s
5		50	88%	96%	86%	91%	5s	7s
6		100	93%	98%	91%	94%	4s	5s
7	GandCrab	30	89%	94%	89%	92%	7s	9s
8		50	89%	96%	87%	91%	5s	7s
9		100	93%	98%	92%	95%	3s	5s

The results show that the developed technique achieved strong detection performance across the three ransomware variants. BadRabbit recorded the highest accuracy of 95% at threshold 100, with precision of 99%, recall of 94%, and F1-score of 96%. This indicates that the system was effective in distinguishing BadRabbit traffic from benign traffic, particularly at the highest threshold. For WannaCry, the best performance was also obtained at threshold 100, with accuracy of 93%, precision of 98%, recall of 91%, and F1-score of 94%. Although the performance was slightly lower than BadRabbit, the results show that the developed technique could still detect and contain WannaCry propagation effectively. For GandCrab, the highest performance was also recorded at threshold 100, with accuracy of 93%, precision of 98%, recall of 92%, and F1-score of 95%. The high F1-score shows that the system maintained a good balance between precision and recall, even though GandCrab generated more false positives at lower threshold values. The performance results suggest that the developed SDN-based technique is effective across multiple ransomware variants. This is important because many earlier SDN-based studies focused mainly on a single ransomware case, such as WannaCry or BadRabbit (Akbanov et al., 2019; Alotaibi & Vassilakis, 2021). By evaluating BadRabbit, WannaCry, and GandCrab, this study provides broader evidence that the proposed service-level isolation approach can generalize across different ransomware behaviors.

4.4 Threshold Value and Response Speed Trade-Off

The results show an important trade-off between threshold value, detection sensitivity, false positives, detection time, and containment time. At lower threshold values, the system is more sensitive and can trigger alerts with fewer suspicious observations. This can be useful for early warning, but it may also increase false positives because some benign traffic may be classified as malicious. At higher threshold values, the system requires stronger evidence of malicious behavior before triggering containment. In this study, threshold 100 produced the best overall performance for all three ransomware variants. It reduced false positives and improved accuracy, precision, recall, and F1-score. It also produced the fastest containment time of 5 seconds across the best-performing experiments. The observed improvement at threshold 100 suggests that the traffic generated by the tested ransomware variants was sufficiently abnormal to exceed a higher detection threshold quickly. Therefore, the higher threshold did not delay response in the testbed; rather, it improved decision quality by reducing false alarms and allowing the controller to apply containment rules more accurately. However, this result should be interpreted with caution. In real-world networks, a very high threshold may delay detection if ransomware spreads slowly, uses stealthy propagation techniques, or generates traffic patterns similar to benign applications. Conversely, a very low threshold may trigger unnecessary blocking of legitimate services. Therefore, threshold tuning should be adapted to the traffic characteristics of the deployment environment. The implication is that threshold selection should not be static in large operational networks. A practical SDN-based ransomware mitigation system should support adaptive threshold tuning based on normal traffic profiles, service criticality, historical false positive rates, and ransomware behavior. This would allow the system to balance early detection with containment accuracy.

4.5 Analysis of False Positives in WannaCry and GandCrab

The results show that WannaCry and GandCrab generated slightly more false positives than BadRabbit, especially at lower threshold values. This may be due to differences in their propagation behavior, communication patterns, and interaction with network services. WannaCry is known for worm-like propagation behavior and can generate repeated scanning and connection attempts across vulnerable hosts (Akbanov et al., 2019). Some of these traffic patterns may resemble legitimate network activities such as service discovery, SMB communication, repeated connection retries, or administrative scanning. As a result, benign traffic with similar characteristics may be incorrectly classified as ransomware-related, particularly when the detection threshold is low. GandCrab also produced higher false positives at threshold 30. This may be because its traffic behavior in the testbed generated patterns that overlapped with benign HTTP or DNS-related communication. Where ransomware communication resembles normal service traffic, the detection system may classify some legitimate flows as suspicious. This explains why increasing the threshold reduced the false positive values for GandCrab from 5 to 2. The higher false positives for WannaCry and GandCrab indicate that ransomware detection should not depend on a single traffic feature. Instead, it should combine multiple indicators, such as connection frequency, service port behavior, packet timing, destination diversity, and protocol-specific anomalies. This supports the need for behavioral analysis and threshold tuning, as implemented in the developed technique. From a practical perspective, false positives are important because unnecessary blocking may affect legitimate services. However, the proposed service-level isolation mechanism reduces the impact of false positives when compared with full host isolation. Since the developed technique blocks only the suspicious service or flow, the operational disruption caused by a false positive is lower than in approaches that disconnect the entire host.

4.6 Comparison with Benchmark SDN-Based Approach

The developed ransomware mitigation technique was compared with the SDN-based approach of Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021) as shown in Table 4.

Table 4: Average Detection and Containment Time for BadRabbit Ransomware

Ransomware	Detection Time		Containment Time	
	Alotaibi & Vassilakis (2021)	Developed Solution	Alotaibi & Vassilakis (2021)	Developed Solution
BadRabbit	5 seconds	5 seconds	30 seconds	7 seconds

The comparison shows that both approaches achieved an average detection time of 5 seconds for BadRabbit ransomware. However, the developed technique achieved a shorter containment time of 7 seconds compared with 30 seconds reported by Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021). This represents a containment time reduction of approximately 77%. The improvement in containment time can be attributed to the selective service-level isolation mechanism. Instead of

applying broad host-level blocking or relying mainly on detection modules, the developed technique extracts specific traffic attributes and installs targeted SDN flow rules to block only the infected service or malicious flow. This reduces the time required to apply containment and minimizes the disruption of unaffected services. This finding is significant because containment time is critical in self-propagating ransomware incidents. Even when detection is accurate, delayed containment may allow ransomware to spread to additional hosts. Therefore, the ability to reduce containment time while preserving host availability represents an important improvement over detection-focused or full-isolation approaches.

4.7 Discussion of Findings

The results demonstrate that SDN can provide an effective platform for real-time ransomware mitigation because of its centralized visibility and programmable traffic control. The developed technique achieved high detection performance across the three ransomware variants and reduced containment time when compared with the benchmark SDN-based method. The findings also show that selective service-level isolation is a practical improvement over full host isolation. In traditional containment approaches, isolating an entire host may prevent further ransomware spread, but it can also disrupt legitimate services. This may be unacceptable in environments where availability is critical, such as healthcare, financial services, education, and government networks. By blocking only the suspicious service or malicious flow, the developed technique provides a more balanced response that supports both security and service continuity. Another important implication is that ransomware mitigation should be evaluated using both detection metrics and response metrics. Accuracy, precision, recall, and F1-score show how well the system identifies ransomware traffic. However, detection time and containment time show whether the system can respond quickly enough to stop propagation. In this study, the reduction in containment time demonstrates that the proposed approach is not only accurate but also operationally useful. The multi-ransomware evaluation also strengthens the contribution of the study. Testing BadRabbit, WannaCry, and GandCrab provided broader evidence of effectiveness than studies limited to one ransomware variant. Although performance varied slightly across the variants, the developed technique maintained strong results in all experiments. This suggests that behavioral SDN-based detection combined with targeted flow-rule enforcement can support flexible ransomware containment across different ransomware behaviors. Overall, the findings confirm that the developed SDN-based mitigation technique can detect and contain self-propagating ransomware effectively. The results also support the central contribution of the study: selective service-level isolation can provide faster containment while preserving unaffected network services.

5. Conclusion

This study developed an enhanced Software-Defined Networking-based technique for mitigating self-propagating ransomware through behavioral traffic analysis and selective service-level isolation. The study was motivated by the limitation of existing host-based, network-based, and SDN-based approaches that either require deployment on individual endpoints, mainly focus on detection, or isolate entire infected hosts in a way that may disrupt legitimate services. To address this gap, the developed technique used the centralized visibility and programmability of SDN to detect ransomware-related propagation behavior and dynamically install flow rules that block only

the infected service or malicious traffic flow. The key novel contribution of this study is the introduction of an SDN-based selective service-level isolation mechanism that contains ransomware propagation by blocking infected services while allowing unaffected services on the same host to remain operational. The developed technique was evaluated in a virtualized SDN testbed using BadRabbit, WannaCry, and GandCrab ransomware variants. The results demonstrated that the technique achieved strong detection performance across the tested ransomware variants and reduced containment time when compared with the benchmark SDN-based method of Alotaibi and Vassilakis (Alotaibi & Vassilakis, 2021). More importantly, the findings show that effective ransomware mitigation should not only focus on detecting malicious traffic but should also provide fast, targeted, and availability-preserving containment. This is particularly important for organizations where full host isolation may interrupt critical services. The practical implication of the study is that SDN can be used as an active ransomware response platform rather than merely a monitoring infrastructure. By applying targeted flow-rule enforcement, network administrators can contain suspicious services in real time while maintaining access to unaffected services. This makes the developed approach relevant to enterprise networks, university networks, government systems, cloud testbeds, and other environments where network availability and rapid containment are both important. The findings also highlight the importance of threshold tuning in ransomware detection. While higher threshold values improved performance in the experimental setup, real-world deployment may require adaptive threshold configuration based on normal traffic profiles, service sensitivity, and the expected behavior of different ransomware variants. Therefore, threshold values should be treated as tunable security parameters rather than fixed universal settings. Despite the promising results, the study has some limitations. The experiments were conducted in a controlled virtualized environment, which may not fully represent the scale, diversity, and traffic complexity of large production networks. The evaluation was limited to three ransomware variants, namely BadRabbit, WannaCry, and GandCrab. In addition, the threshold values were manually configured and tested within the experimental testbed. These limitations provide opportunities for future improvement. Future work should extend the evaluation to larger and more complex network environments with higher traffic volume and diverse host configurations. Additional ransomware families should also be tested to further validate the generalizability of the technique. Furthermore, machine learning or adaptive thresholding mechanisms may be integrated into the SDN controller to improve the system's ability to learn normal traffic behavior and respond to evolving ransomware patterns. Future studies may also examine the deployment of the technique in cloud-based SDN environments and assess its performance under large-scale ransomware propagation scenarios.

References

- Abutu, G., Weissman, D., Hoffmann, P., Bernstein, N., Brennan, O., & Armstrong, T. (2024). Deepcodelock: A novel deep learning-based approach for automated ransomware detection using behavioral signatures. *Authorea Preprints*.
- Akbanov, M., Vassilakis, V. G., & Logothetis, M. D. (2019). Ransomware detection and mitigation using software-defined networking: The case of WannaCry. *Computers and Electrical Engineering*, 76, 111--121. <https://doi.org/10.1016/j.compeleceng.2019.03.012>

- Al-rimy, B. A. S., Maarof, M. A., & Shaid, S. Z. M. (2018). Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions. *Computers & Security*, 74, 144--166. <https://doi.org/10.1016/j.cose.2018.01.001>
- Aladaileh, M. A., Anbar, M., Hasbullah, I. H., Chong, Y.-W., & Sanjalawe, Y. K. (2020). Detection Techniques of Distributed Denial of Service Attacks on Software-Defined Networking Controller--A Review. *IEEE Access*, 8, 143985--143995. <https://doi.org/10.1109/ACCESS.2020.3013998>
- Alotaibi, F. M., & Vassilakis, V. G. (2021). SDN-based detection of self-propagating ransomware: the case of badrabbit. *Ieee Access*, 9, 28039-28058.
- Alsmadi, I. M., & Zarour, M. (2017). Empirical Evidences in Software-Defined Network Security: A Systematic Literature Review. In I. M. Alsmadi, G. Karabatis, & A. Aleroud (Eds.), *Information Fusion for Cyber-Security Analytics* (Vol. 691, pp. 253--295). Springer International Publishing. https://doi.org/10.1007/978-3-319-44257-0_11
- Anderson, N., Li, M., & Evans, M. (2024). Mapping the ransomware ecosystem: Tracing network traffic to command & control centers.
- Batalov, E., Haverstock, P., Anderson, R., Thompson, W., & Wolverton, R. (2024). Ransomware detection via network traffic analysis using isolation forest and lstm neural networks.
- Berrueta, E., Morato, D., Magaña, E., & Izal, M. (2019). A Survey on Detection Techniques for Cryptographic Ransomware. *IEEE Access*, 7, 144925--144944. <https://doi.org/10.1109/ACCESS.2019.2945839>
- Cybercrime Magazine (2025). Cybersecurity Predictions and Statistics For 2024. Retrieved March 3, 2025, from <https://cybersecurityventures.com/cybercrime-to-cost-the-world-9-trillion-annually-in-2024/>
- Farhan, R. I. (2024). An Approach to Android Ransomware Detection Using Deep Learning. *Wasit Journal for Pure sciences*, 3(1), 90-94.
- FBI (2022). Internet Crime Complaint Center Releases 2022 Statistics. Retrieved March 3, 2025, from <https://www.fbi.gov/contact-us/field-offices/springfield/news/internet-crime-complaint-center-releases-2022-statistics>
- Gupret, E., Turner, A., Evans, C., Morgan, R., & Richardson, M. (2024). Dual-layer ransomware classification using opcode and network traffic similarity.
- Jawad, S. F., & Ahmed, N. T. Evaluation Study for Analysis of Quality of Service in the Internet.
- Kayess, R., Ellingworth, G., Alderstone, Z., & Windermere, A. (2024). A novel hierarchical entropy-based framework for ransomware detection in encrypted network traffic.
- Lumazine, A., Drakos, G., Salvatore, M., Armand, V., Andros, B., Castiglione, R., & Grigorescu, E. (2024). Ransomware detection in network traffic using a hybrid cnn and isolation forest approach.
- Mohamed, T. M. H., Al-Rimy, B. A. S., & Almalki, S. A. (2024). A Ransomware Early Detection Model based on an Enhanced Joint Mutual Information Feature Selection Method. *Engineering, Technology & Applied Science Research*, 14(4).
- Morato, D., Berrueta, E., Magaña, E., & Izal, M. (2018a). Ransomware early detection by the analysis of file sharing traffic. *Journal of Network and Computer Applications*, 124, 14--32. <https://doi.org/10.1016/j.jnca.2018.09.013>
- Panaras, A., Silverstein, B., & Edwards, S. (2024). Automated cooperative clustering for proactive ransomware detection and mitigation using machine learning. *Authorea Preprints*.
- Petrosyan, A. (2023). Number of internet and social media users worldwide as of January 2023. Retrieved from Statista database: <https://www.statista.com/statistics/617136/digital-population-worldwide>.

- Robinson S. (2024) Cybercrime and The Challenge of Static Legislations in Nigeria. Forbes Technology Council Post. Retrieved March 3, 2025, from <https://www.forbes.com/councils/forbestechcouncil/2024/04/29/cybercrime-and-the-challenge-of-static-legislations-in-nigeria/>
- Su, D., Liu, J., Wang, X., & Wang, W. (2018). Detecting Android Locker-ransomware on Chinese Social Networks. *IEEE Access*, 7, 20381-20393.
- Umar, R., Riadi, I., & Kusuma, R. S. (2021). Mitigating sodinokibi ransomware attack on cloud network using software-defined networking (SDN). *International Journal of Safety and Security Engineering*, 11(3), 239-246.
- Urooj, U., Al-Rimy, B. A. S., Zainal, A. B., Saeed, F., Abdelmaboud, A., & Nagmeldin, W. (2023). Addressing behavioral drift in ransomware early detection through weighted generative adversarial networks. *IEEE Access*, 12, 3910-3925.
- Venne, S., Clarkson, T., Bennett, E., Fischer, G., Bakker, O., & Callaghan, R. (2024). Automated ransomware detection using pattern-entropy segmentation analysis: A novel approach to network security.
- Ventura, V. (2018). Talos Blog || Cisco Talos Intelligence Group---Comprehensive Threat Intelligence: SamSam---The Evolution Continues Netting Over \$325,000 in 4 Weeks. Retrieved March 3, 2025, from <https://blog.talosintelligence.com/2018/01/samsam-evolution-continues-netting-over.html#more>
- Wiles, A., Colombo, F., & Mascorro, R. (2024). Ransomware detection using network traffic analysis and generative adversarial networks.
- Williams, M., Morales, R., Johnson, K., Martinez, G., & Bennett, J. (2024). Entropy-based network traffic analysis for efficient ransomware detection. *Authorea Preprints*.