

A project report on

INTRUSION DETECTION SYSTEM USING HONEYPOT IN A CLOUD ENVIRONMENT

Submitted in partial fulfillment for the award of the degree of

B.Sc (Computer Science)

by

Mohammad Shidhan NO (21BCS0024)

Abhay v(21BCS0041)



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF COMPUTER SCIENCE ENGINEERING AND
INFORMATION SYSTEMS**

May, 2024

TITLE OF THE PROJECT REPORT

Submitted in partial fulfillment for the award of the degree of

B.Sc (Computer Science)

by

NAME OF THE CANDIDATE (Reg. No.)

<Times New Roman, Font 16, Bold, CAPS>



VIT[®]

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF COMPUTER SCIENCE ENGINEERING AND
INFORMATION SYSTEMS**

May, 2024

DECLARATION

I here by declare that the thesis entitled “INTRUSION DETECTION SYSTEM USING HONEYPOT IN A CLOUD ENVIRONMENT” submitted by me, for the award of the degree of B.Sc(Computer Science) is a record of bonafide work carried out by me under the supervision of Prof.Karthikeyan J.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date:

Signature of the
Candidate

CERTIFICATE

This is to certify that the thesis entitled “INTRUSION DETECTION SYSTEM USING HONEYPOT IN A CLOUD ENVIRONMENT” submitted by Abhay v(21BCS0041), Mohammed Shidhan NO(21BCS0024), School of Computer Science Engineering And Information Systems, Vellore Institute of Technology, Vellore for the award of the degree B.Sc(Computer Science) is a record of bonafide work carried out by him/her under my supervision.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The Project report fulfils the requirements and regulations of VELLORE INSTITUTE OF TECHNOLOGY VELLORE and in my opinion meets the necessary standards for submission.

Signature of the Guide

**Signature of the
HoD**

Internal Examiner

**External
Examiner**

Date:

CERTIFICATE BY THE EXTERNAL GUIDE

This is to certify that the project report entitled “**Intrusion Detection System Using Honeypot In A Cloud Environment**” submitted by Abhay v (21BCS0041), Mohammed Shidhan NO(21BCS0024) to Vellore Institute of Technology in partial fulfillment of the requirement for the award of the degree B.Sc of in Computer Science is a record of bonafide work carried out by him/her under my guidance. The project fulfills the requirements as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

<Times New Roman, Font 12>

<Signature of the External Supervisor>

<Name> <Times New Roman, Font 12, Bold >

**EXTERNAL
SUPERVISOR**

<Times New Roman, Font 12, Bold

>

<Title of the Supervisor >

<Full address of the Institution / organization with e-mail id, phone no.>

<Seal of the Institution / Organization>

<Times New Roman, Font

12>

ABSTRACT

Cloud computing has fundamentally transformed the landscape of modern business operations, offering unparalleled convenience, scalability, and accessibility to computing resources and services via the Internet. However, this very accessibility also renders cloud environments enticing targets for cyber intruders seeking to exploit vulnerabilities and compromise sensitive data. As organizations increasingly rely on cloud infrastructure for storing data, running applications, and conducting critical business functions, safeguarding network-accessible cloud resources against a myriad of cyber threats has become an urgent priority. Traditional network-based Intrusion Detection Systems (NIDS) play a crucial role in monitoring and securing network traffic by identifying patterns or signatures indicative of known attacks. However, these rule-based systems are inherently limited in their ability to detect emerging or zero-day attacks that do not match predefined signatures. Additionally, anomaly detection-based IDS, which detect deviations from normal behavior, often suffer from a high rate of false positives, inundating security teams with alerts and potentially obscuring genuine threats amidst the noise. To address these challenges and bolster the security posture of cloud computing environments, researchers have proposed innovative approaches that integrate IDS with cloud infrastructure. By harnessing the scalability, flexibility, and computational power of the cloud, organizations can enhance their ability to detect and respond to cyber threats in real-time. One such approach involves the strategic deployment of honeypots within cloud environments as part of the IDS architecture. Honeypots are decoy systems or services intentionally designed to mimic legitimate assets within the network, enticing attackers to interact with them and revealing their tactics, techniques, and procedures (TTPs). In the context of cloud computing, honeypots serve as bait, diverting suspicious activity away from genuine assets and providing organizations with valuable insights into the methods employed by adversaries. By monitoring interactions with honeypots, security teams can gather intelligence on emerging threats, identify attack vectors, and develop proactive defense strategies to mitigate future risks. The integration of honeypots into cloud IDS design offers several distinct advantages. Firstly, it enhances threat detection capabilities by providing a proactive defense mechanism against potential attacks. By diverting malicious activity to honeypots, organizations can identify and respond to threats at an early stage, minimizing the risk of data breaches or system compromise. Moreover, the presence of honeypots acts as a deterrent to would-be attackers, dissuading them from targeting genuine assets within the cloud environment. Furthermore, the use of honeypots can significantly reduce the number of false positives generated by traditional IDS, thereby alleviating alert fatigue and enabling security teams to focus their efforts on genuine threats.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to prof.Karthikeyan J, Associate professor sr., School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, for his constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavor. My association with him is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of .

I would like to express my gratitude to DR.G.VISWANATHAN, Chancellor VELLORE INSTITUTE OF TECHNOLOGY, VELLORE, MR. SANKAR VISWANATHAN, DR. SEKAR VISWANATHAN, MR.G V SELVAM, Vice – Presidents VELLORE INSTITUTE OF TECHNOLOGY, VELLORE, Dr. V. S. Kanchana Bhaaskaran, I/c Vice – Chancellor, Dr. Partha Sharathi Mallick, Pro-Vice Chancellor and Dr. S. Sumathy, Dean, School of Computer Science Engineering And Information Systems,, for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to Dr.Iyapparaja M, HoD/Professor, all teaching staff and members working as limbs of our university for their not-self-centered enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Vellore

Date:

Name of the student

TABLE OF CONTENTS

LIST OF FIGURES.....	ix
-----------------------------	-----------

LIST OF TABLES.....	xi
----------------------------	-----------

LIST OF ACRONYMS.....	xii
------------------------------	------------

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND.....	1
----------------------------	----------

1.2 MOTIVATION.....	4
----------------------------	----------

1.3 PROJECT STATEMENT.....	7
-----------------------------------	----------

1.4 OBJECTIVES.....	7
----------------------------	----------

1.5 SCOPE OF THE PROJECT.....	8
--------------------------------------	----------

CHAPTER 2

LITERATURE SURVEY

2.1 SUMMARY OF THE EXISTING WORKS

2.2 CHALLENGES PRESENT IN EXISTING SYSTEM

CHAPTER 3

REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

3.2 SOFTWARE REQUIREMENTS

3.3 BUDGET

3.4 GANTT CHART

CHAPTER 4

ANALYSIS & DESIGN

4.1 PROPOSED METHODOLOGY

4.2 SYSTEM ARCHITECTURE

4.3 MODULE DESCRIPTIONS

CHAPTER 5

IMPLEMENTATION & TESTING

5.1 SAMPLE OUTPUT

CHAPTER 6

RESULTS

6.1 RESEARCH FINDINGS

6.2 RESULT ANALYSIS & EVALUATION METRICS

CONCLUSIONS AND FUTURE WORK

REFERENCES

Chapter 1

Introduction

Cloud computing stands as a transformative model revolutionizing the landscape of modern computing by facilitating convenient, on-demand network access to a shared pool of configurable computing resources, including servers, storage, applications, and services, that can be rapidly provisioned and released with minimal management effort or service provider interaction. This paradigm shift has emerged as a major boon, particularly for startups and burgeoning enterprises, offering a pathway to deploy online applications and services without the need for substantial investments in storage, web infrastructure, or computing resources. Leveraging well-established Internet protocols, standards, and formats, cloud computing exposes a comprehensive set of consumable services to end-users and consumers. These services span a broad spectrum, ranging from fundamental computing utilities to sophisticated platforms for application development and deployment. By abstracting the complexities of infrastructure management and provisioning, cloud computing empowers organizations to focus their resources and attention on innovation, product development, and business growth, rather than grappling with the intricacies of hardware procurement, maintenance, and scalability. Additionally, cloud computing democratizes access to advanced technologies and computational

resources, leveling the playing field for businesses of all sizes and industries. This democratization fosters innovation, creativity, and competition, driving continuous advancements in technology and fueling economic growth and prosperity on a global scale. Furthermore, the inherent scalability and elasticity of cloud computing enable organizations to seamlessly adapt to fluctuating demands and market conditions, ensuring optimal resource utilization and cost efficiency. As cloud computing continues to evolve and mature, it is poised to reshape industries, disrupt traditional business models, and unlock new opportunities for collaboration, efficiency, and value creation. However, alongside its myriad benefits, cloud computing also presents unique challenges and considerations, including data security, privacy, regulatory compliance, and vendor lock-in. Addressing these challenges requires a comprehensive approach encompassing robust security measures, effective governance frameworks, and transparent communication between stakeholders. In conclusion, cloud computing represents a paradigm shift in the way organizations leverage technology to drive innovation, agility, and growth. By embracing the principles of cloud computing and harnessing its transformative potential, businesses can navigate the complexities of the digital age and thrive in an increasingly interconnected and dynamic global marketplace.

1.1 Background

Emergence of Intrusion Detection System

The emergence of the "Intrusion Detection System Using Honeypots in a Cloud Environment" is a response to the evolving landscape of cybersecurity threats that coincide with the widespread adoption of cloud computing. As organizations increasingly transition their infrastructure and services to cloud platforms to leverage benefits like scalability, cost-effectiveness, and accessibility, the need for robust security measures becomes paramount. However, traditional security solutions often struggle to adapt to the dynamic and distributed nature of cloud architectures, leaving gaps that malicious actors exploit. In this context, the integration of honeypots into intrusion detection systems (IDS) emerges as a proactive and innovative approach to bolstering cloud security.

Honeypots have gained recognition as a proactive defense mechanism against cyber threats by mimicking legitimate services and enticing malicious actors to interact with them. This deception-based approach enables organizations to detect and gather intelligence on potential threats before they can cause harm. By strategically deploying

honeypots within a cloud environment, organizations can complement their existing security measures and gain valuable insights into the tactics, techniques, and procedures (TTPs) employed by adversaries. This proactive stance not only enhances threat detection capabilities but also empowers organizations to respond effectively to emerging cyber threats.

Current Detection Challenges

Despite the benefits offered by an IDS using honeypots in a cloud environment, several challenges persist:

1. **Scalability:** Adapting traditional IDS architectures to the scale and elasticity of cloud environments can be challenging. Ensuring seamless deployment and management of honeypots across dynamic cloud infrastructures requires scalable and automated solutions that can adapt to changing workload demands.
2. **Integration Complexity:** Integrating honeypots into existing cloud security architectures may pose integration challenges, particularly in multi-tenant environments or hybrid cloud setups. Compatibility issues, resource constraints, and varying security requirements need to be addressed to ensure seamless operation and effective threat detection.
3. **False Positives:** Honeypots have the potential to generate false positives, triggering unnecessary alerts and impacting operational efficiency. Fine-tuning detection mechanisms, minimizing noise, and distinguishing between genuine threats and benign activities are essential to maintaining the effectiveness of the IDS.
4. **Evasion Techniques:** Sophisticated attackers may employ evasion techniques to bypass honeypots and evade detection. Continuous monitoring, updating of honeypot configurations, and incorporating advanced detection capabilities are necessary to stay ahead of evolving threats and minimize the risk of evasion.

Addressing these challenges requires ongoing research, collaboration, and innovation in the field of cloud security. By overcoming these obstacles, organizations can harness the full potential of an IDS using honeypots to enhance the resilience of their cloud infrastructure against cyber threats and safeguard their critical assets and data.

Need for IDS

The need for an intrusion detection system using honeypots in a cloud environment is driven by several factors:

1. **Advanced Threat Landscape:** The modern threat landscape is characterized by increasingly sophisticated and persistent cyber threats. Traditional security measures, such as firewalls and antivirus software, are no longer sufficient to protect against these evolving threats. Honeypots offer a proactive defense mechanism that enables organizations to stay ahead of adversaries by actively engaging and monitoring their activities.
2. **Cloud Security Challenges:** Cloud environments introduce unique security challenges due to their dynamic nature, shared resources, and multi-tenancy. Traditional IDS solutions may struggle to adapt to the complexities of cloud architectures, leaving organizations vulnerable to attacks. Implementing an IDS using honeypots addresses these challenges by providing granular visibility into cloud-based threats and enabling rapid response to security incidents.
3. **Compliance Requirements:** Regulatory requirements and industry standards mandate robust security measures to protect sensitive data and ensure regulatory compliance. An IDS with honeypots demonstrates a proactive approach to cybersecurity, helping organizations meet compliance requirements and mitigate the risk of regulatory penalties or fines.
4. **Threat Intelligence Gathering:** Honeypots serve as valuable tools for gathering real-time threat intelligence by capturing and analyzing malicious activities. Integrating honeypots into a cloud-based IDS enhances organizations' ability to detect and respond to emerging threats effectively. By leveraging the data collected from honeypots, organizations can enhance their threat intelligence capabilities, improve incident response strategies, and strengthen overall security posture.

Advanced Solution

An advanced solution for the "Intrusion Detection System Using Honeypots in a Cloud Environment" involves leveraging cutting-edge technologies and methodologies to enhance threat detection, response capabilities, and overall security posture. This solution integrates state-of-the-art techniques for deploying and managing honeypots within a cloud environment, along with advanced analytics and automation capabilities to effectively detect, analyze, and respond to cyber threats. Here's an outline of the advanced solution:

1. Dynamic Honeypot Deployment:

- Implement dynamic honeypot deployment mechanisms that automatically scale and adapt to changes in the cloud environment, such as the creation, deletion, and migration of virtual instances.
- Utilize infrastructure-as-code (IaC) tools like Terraform or CloudFormation to automate the provisioning and configuration of honeypot instances, ensuring consistency and repeatability.

2. Diverse Honeypot Types:

- Deploy a variety of honeypot types, including low-interaction and high-interaction honeypots, to simulate different services and protocols (e.g., web servers, databases, SSH, FTP).
- Utilize container-based honeypots (e.g., Docker containers) to encapsulate honeypot environments, enabling lightweight and efficient deployment across cloud clusters.

3. Intelligent Traffic Analysis:

- Implement advanced traffic analysis techniques, such as protocol fingerprinting, payload analysis, and behavior-based anomaly detection, to identify and categorize suspicious activities.
- Utilize machine learning algorithms for anomaly detection and pattern recognition, enabling the system to adapt and evolve based on observed network behavior.

4. Real-time Threat Intelligence Integration:

- Integrate with threat intelligence feeds and external sources to enrich honeypot data with real-time threat indicators, such as known malicious IP addresses, domains, and signatures.
- Leverage threat intelligence platforms (TIPs) and APIs to automate the ingestion, correlation, and enrichment of threat intelligence data, enhancing the system's ability to identify emerging threats.

5. Automated Response and Orchestration:

- Implement automated response mechanisms to dynamically block or quarantine malicious actors and mitigate potential threats in real-time.
- Utilize security orchestration, automation, and response (SOAR) platforms to streamline incident response workflows, enabling faster decision-making and remediation actions.

6. Behavioral Analysis and Deception Tactics:

- Employ behavioral analysis techniques to identify patterns of malicious behavior and detect sophisticated attacks, such as lateral movement and privilege escalation.
- Leverage deception tactics, such as honeytokens and breadcrumbs, to lure attackers into revealing their tactics and techniques, providing valuable insights for threat hunting and attribution.

7. Scalable and Resilient Architecture:

- Design a scalable and resilient architecture that can handle high volumes of traffic, accommodate dynamic workloads, and withstand distributed denial of service (DDoS) attacks.
- Utilize cloud-native services and serverless computing (e.g., AWS Lambda, Azure Functions) to dynamically scale honeypot infrastructure based on demand and optimize resource utilization.

8. Continuous Monitoring and Evaluation:

- Implement continuous monitoring and evaluation mechanisms to assess the effectiveness of the intrusion detection system and honeypot deployments.
- Utilize metrics, logs, and telemetry data to measure key performance indicators (KPIs) and identify areas for improvement, such as detection accuracy, false positive rates, and response times.

By implementing this advanced solution, organizations can enhance their ability to detect, analyze, and respond to cyber threats in cloud environments effectively. This comprehensive approach leverages the latest technologies and best practices to stay ahead of adversaries and safeguard critical assets and data against evolving security threats.

Conclusion

In conclusion, the implementation of an "Intrusion Detection System Using Honeypots in a Cloud Environment" represents a proactive and innovative approach to enhancing cybersecurity in the era of cloud computing. The emergence of this solution is driven by the need to address the evolving threat landscape, unique challenges posed by cloud architectures, and regulatory compliance requirements. By integrating honeypots into intrusion detection systems within cloud environments, organizations can gain granular visibility into potential threats, gather real-time threat intelligence, and enhance their incident response capabilities.

However, the adoption of this advanced solution is not without its challenges. Overcoming scalability issues, integration complexities, false positives, and evasion techniques requires ongoing research, collaboration, and innovation. By leveraging

cutting-edge technologies such as dynamic honeypot deployment, intelligent traffic analysis, automated response, and behavioral analysis, organizations can stay ahead of adversaries and bolster their defenses against cyber threats.

Furthermore, the continuous monitoring, evaluation, and optimization of the intrusion detection system are essential to maintaining its effectiveness over time. By measuring key performance indicators and incorporating feedback loops, organizations can identify areas for improvement and enhance the resilience of their cloud infrastructure against emerging security threats.

In essence, the implementation of an IDS using honeypots in a cloud environment represents a proactive investment in cybersecurity that enables organizations to detect, analyze, and respond to threats in real-time. By staying vigilant, adaptive, and collaborative, organizations can mitigate risks, safeguard critical assets and data, and maintain the trust and confidence of their stakeholders in an increasingly interconnected and digital world.

1.2 Motivation

In today's rapidly evolving cybersecurity landscape, the need for robust and effective intrusion detection systems (IDS) has never been more critical. With cloud computing becoming the backbone of modern business operations, organizations face unprecedented challenges in safeguarding their network-accessible resources from an array of sophisticated cyber threats. Traditional IDS solutions, while valuable, often struggle to keep pace with the dynamic nature of cloud environments, relying on static signatures or heuristics to detect malicious activity.

However, the integration of honeypots into IDS architecture presents a compelling solution to this challenge, offering a proactive defense mechanism that is tailor-made for the unique characteristics of cloud computing. By strategically deploying honeypots within cloud environments, organizations can create decoy systems or services designed to lure potential attackers, thereby providing valuable insights into their tactics, techniques, and procedures (TTPs). This proactive approach not only enhances threat detection capabilities but also reduces the number of false positives generated by traditional IDS, enabling security teams to focus their efforts on genuine threats.

Furthermore, the use of honeypots in a cloud environment offers additional benefits, such as deterring attackers, gathering threat intelligence data, and facilitating research and analysis efforts. Overall, the integration of honeypots into an IDS framework represents a significant advancement in cybersecurity, empowering organizations to strengthen their

network defenses, mitigate risks, and ensure the integrity and confidentiality of their cloud-hosted data and services. As the adoption of cloud computing continues to grow, the need for innovative security solutions like honeypot-based IDS will only become more pronounced, making it imperative for organizations to embrace this technology to stay ahead of emerging threats and safeguard their digital assets.

1.3 Project Statement

Project Statement:

The project aims to develop an advanced intrusion detection system (IDS) tailored for cloud environments, leveraging the integration of honeypots to enhance threat detection capabilities and mitigate risks associated with cyber attacks. In today's rapidly evolving cybersecurity landscape, organizations increasingly rely on cloud computing for their critical business operations, making the protection of network-accessible resources a top priority. Traditional IDS solutions often struggle to keep pace with the dynamic nature of cloud environments, leading to challenges in effectively detecting and responding to emerging threats. The proposed project seeks to address these challenges by designing and implementing an IDS framework that incorporates honeypots strategically deployed within cloud infrastructure. Honeypots serve as decoy systems or services designed to lure potential attackers, providing valuable insights into their tactics, techniques, and procedures (TTPs). By analyzing interactions with honeypots, the IDS can proactively identify and respond to malicious activity, reducing the risk of data breaches, system compromise, and service disruptions.

Key objectives of the project include designing an architecture for integrating honeypots into the IDS framework, tailored specifically for cloud environments, developing algorithms and mechanisms for deploying and managing honeypots within cloud infrastructure, ensuring seamless integration with existing security controls, implementing detection mechanisms to analyze interactions with honeypots and identify potential security threats in real-time, evaluating the performance and effectiveness of the IDS using realistic attack scenarios and benchmarking against traditional IDS solutions, and providing documentation, training, and support for deploying and maintaining the IDS framework in cloud environments. Ultimately, the project aims to empower organizations to strengthen their network defenses, mitigate risks, and ensure the integrity and confidentiality of their cloud-hosted data and services. By leveraging honeypot technology within the IDS framework, the project will contribute to advancements in cloud security and support the adoption of innovative security solutions in the face of evolving cyber threats.

1.4 Objectives:

Enhance Security Posture:

Strengthening the security posture of a cloud environment is paramount in today's digital landscape where cyber threats are constantly evolving. By adopting proactive strategies and leveraging advanced technologies, organizations can bolster their defenses and mitigate the risk of cyber attacks. Let's delve deeper into each aspect of enhancing security posture.

Early Threat Detection:

Identify and Respond to Security Threats Early:

Early detection of security threats is crucial for minimizing the impact of attacks and reducing the likelihood of data breaches or service disruptions. Implementing continuous monitoring and real-time alerting mechanisms enables organizations to detect suspicious activities promptly. By leveraging threat intelligence feeds, anomaly detection algorithms, and behavioral analysis techniques, organizations can identify potential security incidents at an early stage and initiate timely response actions.

Behavioral Analysis:

Analyze Network Traffic and System Activity Patterns:

Behavioral analysis involves examining network traffic and system activity patterns to identify deviations from normal behavior. By establishing baseline behavior profiles for users, devices, and applications, organizations can detect anomalous activities indicative of unauthorized access attempts, reconnaissance activities, or malware infections. Advanced analytics tools, such as machine learning algorithms, can analyze vast amounts of data to uncover subtle indicators of compromise and facilitate proactive threat hunting efforts.

Threat Intelligence Gathering:

Gather Actionable Threat Intelligence:

Threat intelligence gathering involves actively monitoring and analyzing attackers' behavior and tactics within a controlled environment, such as a honeypot. By deploying decoy systems and services designed to attract adversaries, organizations can gather valuable insights into emerging threats and attack techniques. This intelligence can inform security decision-making processes, enhance threat detection capabilities, and enable proactive defense measures against evolving cyber threats.

Risk Mitigation:

Mitigate the Risk of Data Exfiltration and Unauthorized Access:

Effective risk mitigation strategies are essential for safeguarding sensitive data and preventing unauthorized access to critical resources. By implementing robust access controls, encryption mechanisms, and intrusion detection and prevention systems (IDPS), organizations can minimize the risk of data exfiltration, data loss, and unauthorized access. Regular security assessments, vulnerability scanning, and penetration testing help identify and remediate security weaknesses before they can be exploited by malicious actors.

Security Incident Analysis:

Conduct Thorough Analysis of Security Incidents:

In the event of a security incident or breach, conducting thorough analysis is crucial for understanding the nature and scope of the attack. Forensic analysis techniques, such as disk imaging, memory analysis, and malware reverse engineering, enable organization to identify the root causes of security incidents and assess the extent of the compromise. By documenting findings, preserving digital evidence, and implementing remediation measures, organizations can strengthen their incident response capabilities and mitigate the risk of future incidents. Enhancing the security posture of a cloud environment requires a comprehensive and proactive approach that addresses various aspects of threat detection, risk mitigation, and incident response. By investing in advanced technologies, employee training, and collaboration with industry peers, organizations can stay ahead of emerging cyber threats and protect their assets from malicious actors. Continuous monitoring, analysis, and improvement of security practices are essential for maintaining a strong defense posture and safeguarding against evolving cyber threats in today's dynamic threat landscape.

1.5 Scope Of The Project

Cloud Environment Selection:

Criteria for Choosing the Appropriate Cloud Platform Selecting the right cloud platform for deploying a honeypot infrastructure requires careful consideration of various factors, including scalability, availability, and ease of deployment. Each criterion plays a crucial role in ensuring the effectiveness and efficiency of the honeypot environment.

Scalability and Elasticity:

Scalability and elasticity are fundamental characteristics of any cloud environment specially when deploying a honeypot infrastructure. The ability to scale resources dynamically based on workload demands is essential for accommodating fluctuating traffic patterns and evolving security requirements.

Importance of Scalability and Elasticity:

1. Accommodating Fluctuating Workloads: Honeypots experience variable levels of traffic depending on factors such as network activity and threat detection. A scalable cloud platform can adjust resource allocation to handle sudden spikes in traffic effectively.
2. Meeting Evolving Security Requirements: As security threats evolve, the demands on the honeypot infrastructure may change. Scalability ensures that additional resources can be provisioned quickly to implement new security measures or respond to emerging threats.

Auto-Scaling Mechanisms:

Implementing auto-scaling mechanisms within the cloud environment is critical for achieving scalability and elasticity. These mechanisms automatically adjust resource allocation based on predefined criteria, such as CPU utilization or network traffic.

Optimizing Cost-Efficiency and Performance:

Effective auto-scaling not only ensures that the infrastructure can handle workload fluctuations but also optimizes cost-efficiency and performance. Resources are allocated dynamically, minimizing underutilization and reducing unnecessary expenses.

Best Practices for Scalability and Elasticity:

1. **Use of Managed Services:** Leveraging managed services provided by the cloud platform, such as AWS Auto Scaling or Azure Auto scale, simplifies the implementation of auto-scaling mechanisms.
2. **Monitoring and Metrics:** Regular monitoring of key performance metrics enables proactive scaling based on real-time data, ensuring optimal resource utilization.
3. **Horizontal and Vertical Scaling:** Employing both horizontal (adding more instances) and vertical (increasing the size of existing instances) scaling strategies provides flexibility in managing resource allocation.

Honeypot Deployment:

Deploying a honeypot infrastructure within the chosen cloud environment requires careful planning and consideration of various factors. From selecting appropriate honeypot types to designing detailed architectural blueprints, each step contributes to the successful deployment and operation of the honeypots.

Selection of Honeypot Types:

Choosing the right honeypot types depends on factors such as the level of interaction required, resource utilization, and security implications. Common honeypot types include low-interaction honeypots (e.g., Honeyd) and high-interaction honeypots (e.g., Dionaea). Each type offers different levels of complexity and realism in simulating network services and vulnerabilities.

Detailed Architectural Designs:

Developing detailed architectural designs is essential for defining the structure and components of the honeypot infrastructure. This includes:

1. **Network Configuration:** Configuring network settings to ensure proper isolation and segmentation of the honeypot environment from production networks.
2. **Virtual Machine Instances:** Provisioning virtual machine instances with the necessary operating systems and software to simulate target systems and services.
3. **Storage Requirements:** Determining storage requirements for capturing and storing data collected by the honeypots, such as network traffic logs and malware samples.
4. **Access Controls:** Implementing access controls to restrict unauthorized access to the honeypot environment and prevent tampering with the infrastructure.

Security Considerations:

Ensuring the security of the honeypot deployment is paramount to prevent unauthorized access or exploitation by malicious actors. This includes:

1. **Hardening Configuration:** Implementing security best practices to harden the configuration of the honeypot instances and minimize potential attack vectors.

2. **Regular Patching and Updates:** Keeping the honeypot software and underlying operating systems up-to-date with the latest security patches and updates to address known vulnerabilities.
3. **Monitoring and Alerting:** Setting up monitoring and alerting mechanisms to detect suspicious activity or unauthorized access attempts in real-time.

Continuous Improvement:

Deploying honeypots is an ongoing process that requires continuous monitoring, maintenance, and optimization. Regular reviews of the honeypot deployment help identify areas for improvement and ensure alignment with evolving security requirements.

Traffic Monitoring and Analysis:

Once the honeypot infrastructure is deployed, monitoring mechanisms are implemented to capture and analyze incoming traffic directed at the honeypots. This involves deploying network monitoring tools, such as intrusion detection systems (IDS), packet capture utilities, and log analysis frameworks.

Importance of Traffic Monitoring:

1. **Detecting Security Threats:** Monitoring incoming traffic allows for the detection of security threats, such as unauthorized access attempts, malware infections, or network reconnaissance activities.
2. **Gathering Threat Intelligence:** Analyzing traffic patterns and attack vectors provides valuable insights into the tactics, techniques, and procedures (TTPs) used by potential adversaries, enhancing threat intelligence capabilities.

3. **Supporting Incident Response:** Real-time monitoring facilitates rapid incident response by alerting security teams to suspicious activity or anomalous behavior that may indicate a security incident.

Network Monitoring Tools:

Deploying network monitoring tools enables comprehensive visibility into network traffic and activity. Commonly used tools include:

1. **Intrusion Detection Systems (IDS):** IDS solutions analyze network traffic for signs of malicious activity or policy violations and generate alerts or notifications when potential threats are detected.
2. **Log Analysis Frameworks:** Log analysis frameworks aggregate and analyze log data from sources, including network devices, servers, and applications, to identify patterns or anomalies indicative of security incidents.

Advanced Analytics Techniques:

In addition to traditional network monitoring tools, advanced analytics techniques can be employed to enhance threat detection capabilities:

1. **Machine Learning:** Machine learning algorithms can analyze large volumes of network data to identify patterns or anomalies that may indicate security threats, such as unusual traffic patterns or suspicious behavior.
2. **Anomaly Detection:** Anomaly detection techniques identify deviations from normal behavior based on historical data or predefined baselines, enabling the detection of unusual or potentially malicious activity.

Custom Algorithms and Models:

Developing custom algorithms and models tailored to the specific characteristics of the honeypot environment enhances the accuracy and effectiveness of threat detection:

1. Feature Engineering: Extracting relevant features from network traffic data, such as packet headers, payload content, or behavioral attributes, to feed into machine learning models for analysis.
2. Training and Evaluation: Training machine learning models on labeled datasets and evaluating their performance using metrics such as accuracy, precision, recall, and F1-score to ensure robust detection capabilities.

Integration with Incident Response:

Integrating traffic monitoring and analysis capabilities with incident response processes enables a coordinated and timely response to security incidents:

1. Alert Triage: Prioritizing and triaging alerts generated by the monitoring tools based on severity, relevance, and potential impact to facilitate rapid incident response.
2. Forensic Analysis: Using traffic data captured by the monitoring tools for forensic analysis and investigation of security incidents, including root cause analysis and attribution of malicious activity.

Incident Response and Forensics:

Provisions for incident response and forensic analysis capabilities are essential for investigating security incidents and breaches effectively. Detailed incident response playbooks, forensic procedures, and chain of custody protocols ensure proper handling and preservation of digital evidence.

Incident Response Playbooks:

Developing incident response playbooks outlines the procedures and workflows for responding to

security incidents systematically:

1. Incident Identification: Defining criteria for identifying security incidents based on predefined indicators of compromise (IOCs) or suspicious activity detected through monitoring.
2. Response Coordination: Establishing roles, responsibilities, and communication channels for coordinating incident response efforts across relevant stakeholders, including IT teams, personnel, and management.
3. Containment and Mitigation: Implementing measures to contain the impact of the incident and mitigate further damage or unauthorized access while investigations are underway.

Forensic Procedures:

Establishing forensic procedures ensures the proper collection, preservation, and analysis of digital evidence:

1. Evidence Collection: Documenting and collecting digital evidence in a forensically sound manner to maintain its integrity and admissibility in legal proceedings.
2. Chain of Custody: Establishing a chain of custody protocol to track the handling and movement of evidence throughout the investigative process, ensuring its integrity and reliability.
3. Forensic Analysis: Conducting detailed forensic analysis of digital evidence, including disk imaging, memory analysis, and timeline reconstruction, to reconstruct the sequence of events and identify the root cause of the security incident.

Legal and Compliance Considerations:

Adhering to legal and compliance requirements is essential when conducting incident response and forensic investigations:

1. **Data Privacy Laws:** Complying with data privacy regulations, such as GDPR or CCPA, when handling sensitive information or personally identifiable data during investigations.
2. **Chain of Custody Documentation:** Maintaining accurate and detailed records of the chain of custody for digital evidence to ensure its admissibility and credibility in legal proceedings.
3. **Collaboration with Law Enforcement:** Collaborating with law enforcement agencies, when necessary, to report security incidents, share evidence, and support criminal investigations into cyberattacks or data breaches.

Continuous Improvement:

Incident response and forensic capabilities should be continuously evaluated and refined to adapt to evolving security threats and regulatory requirements:

1. **Post-Incident Review:** Conducting post-incident reviews and lessons learned sessions to identify areas for improvement in incident response procedures, forensic techniques, and overall security posture.

Incident Response Tabletop Exercises: Conducting regular tabletop exercises or simulations to test incident response plans, validate response procedures, and enhance coordination among stakeholders.

2. **Training and Skills Development:** Providing ongoing training and skills development opportunities for incident responders and forensic analysts.

CHAPTER 2

LITERATURE SURVEY

2.1 SUMMARY OF THE EXISTING WORKS

TITLE	AUTHOR NAME	ABSTRACT
An IoT Honeynet Based on Multiport Honeypots for Capturing IoT Attacks	WeizheZhang , Bin Zhang Ying Zhou , Hui He, Zeyu Ding	The article introduces a specialized honeynet system designed for IoT devices, aiming to address their susceptibility to attacks due to limited network resources and complex operating systems. Three types of honeypots are implemented: a medium-high interaction honeypot emulating router UPnP services affected by CVE-2017–17215, a high-interaction honeypot utilizing actual IoT device firmware matching vulnerabilities, and a multiport honeypot targeting exposedSOAP service ports. Docker is employed for streamlined deployment, while a centralized control center facilitates management. In practical deployment, the system effectively captures previously unidentified malicious attacks, demonstrating its efficacy compared to traditional security measures like Virus Total.
Tracing MIRAI Malware in Networked System	Yao Xu , Hiroshi Koide Danillo Vasconcellos , Kouichi Sakurai	The paper delves into the threat posed by the Mirai malware amidst the proliferation of Internet of Things (IoT) devices, projected to reach 30 billion in 2021. Mirai's ability to compromise vulnerable IoT devices with default credentials

		<p>for launching Distributed Denial of Service (DDoS) attacks underscores the urgent need for improved IoT security. To understand and combat Mirai effectively, the paper conducts a combined static and dynamic analysis of the malware.</p> <p>Leveraging insights from this analysis, the paper introduces Threat Tracer, an information system simulator initially designed to combat Advanced Persistent Attacks (APT). By simulating Mirai's operational processes within Threat Tracer, the paper aims to provide a comprehensive understanding of Mirai's behavior and vulnerabilities. Additionally, this simulation can help predict the behavior of future Mirai variants, aiding in proactive threat mitigation</p>
<p>A prototype network monitoring information system: modelling, design, implementation and evaluation</p>	<p>Sarandis Mitropoulos, Vassilis Toulas, Christos Douligeris</p>	<p>This paper presents the development and assessment of a prototype network monitoring information system. It begins with an overview of network management systems, emphasizing the importance of visualizing network elements for effective monitoring. The necessity of network monitoring for network administrators and relevant users is discussed, outlining key reasons for employing network management tools. The paper then details the functional and architectural design and implementation of the proposed network monitoring tool, incorporating</p>

		<p>advanced standards and technologies such as TMF608 multi- technology network management, NML-EML interface, and the JBoss enterprise application platform. A notable aspect of the prototype is its graphical representation of telecommunication elements and managed objects, including real-time alarms. The system enables users to acknowledge alarms and provides operational demonstrations. Evaluation results are discussed, followed by conclusions and suggestions for futurework.</p>
<p>Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification</p>	<p>Matthew Roughan ,Subhabrata Sen , Oliver Spatscheck ,Nick Duffiel</p>	<p>This paper addresses the challenge of providing Quality of Service (QoS) guarantees for different types of traffic in IP networks, particularly in enterprise settings. Despite the existence of various QoS mechanisms, widespread deployment remains limited. The authors argue that a key barrier to adoption is the lack of effective methodologies for mapping traffic from different applications to appropriate QoS classes. This is exacerbated by the difficulty of identifying applications in enterprise networks and the limitations of port- based classification. The paper proposes a solution framework for measurement-based traffic classification using statistical application signatures. These signaturesare designed to be protocol-agnostic and focus on how applications are used(e.g.,</p>

		<p>interactive or bulk data transport).The resulting signatures are then used to determine the Class of Service (CoS)for individual IP datagrams. Evaluationusing traffic traces demonstrates the feasibility and potential of the proposedapproach, highlighting its practical relevance in addressing QoS challengesin IP networks.</p>
<p>A real time mobile traffic classification approach based on timing sequence flow</p>	<p>LIU YI , SONG TIAN , LIAO Le-Jian</p>	<p>This paper addresses the challenges posed by the rapid expansion of mobileInternet, particularly in the realms of network security, measurement, and quality of service. Researchers aiming to delve deeper into mobile Internet characteristics require efficient methods for classifying mobile traffic amidst traditional network traffic. The paper introduces a novel approach that combines lightweight flow table and deep packet inspection (DPI) technology for real-time mobile network traffic classification. By expanding network flow into sequence flow segments based on interval-time relationships and employing DPI on thefirst N packets within these segments, the approach achieves accurate mobile traffic classification while minimizing DPI overhead and flow table scale. Real-time experiments demonstrate a classification accuracy rate of 91.55%, with DPI overhead averaging only 20 packets and a flow table scale reductionto 0.21%.</p>

		Comparative analysis against POF indicates a significant improvement in accuracy with the proposed approach.
Early detection of DDoS attacks against SDN controllers	Seyed Mohammad Mousavi , Marc St-Hilarie	A Software Defined Network (SDN) is a new network architecture that provides central control over the network. Although central control is the major advantage of SDN, it is also a single point of failure if it is made unreachable by a Distributed Denial of Service (DDoS) Attack. To mitigate this threat, this paper proposes to use the central control of SDN for attack detection and introduces a solution that is effective and lightweight in terms of the resources that it uses. More precisely, this paper shows how DDoS attacks can exhaust controller resources and provides a solution to detect such attacks based on the entropy variation of the destination IP address. This method is able to detect DDoS within the first five hundred packets of the attack traffic.
A honeypot for arbitrary malware on USB storage devices	Sebastian Poeplau Jan Gassen	Malware is a serious threat for modern information technology. It is therefore vital to be able to detect and analyze such malicious software in order to develop countermeasures. Honeypots are a tool supporting that task - they collect Unfortunately, existing honeypots concentrate on malware that spreads over

		<p>networks, thus missing any malware that does not use a network for propagation. A popular network- independent technique for malware to spread is copying itself to USB flash drives. In this article we present Ghost, a new kind of honeypot for such USB malware. It detects malware by simulating a removable device in software, thereby tricking malware into copying itself to the virtual device. We explain the concept in detail and evaluate it using samples of wide- spread malware. We conclude that this new approach works reliably even for sophisticated malware, thus rendering the concept a promising new idea.</p>
<p>Design of Intrusion Detection Honeypot Using Social Leopard Algorithm to Detect IOT Ransomware Attacks</p>	<p>S.Sibi Chakkaravarthy , D.Sangeetha , Meenalosini Vimal Cruz, V.Vaidehi , Balasubramanian Raman</p>	<p>In recent years, ransomware has emerged as a major cyber threat affecting various sectors including individuals, enterprises, healthcare, and the Internet of Things (IoT). Conventional security measures such as Intrusion Detection and Prevention System (IDPS) and Antivirus (AV) software, operating as singular monitoring agents, often prove inadequate in detecting ransomware complexity and time-consuming nature. To address these shortcomings, a robust solution in the form of an Intrusion Detection Honeypot (IDH) is proposed. The IDH comprises three main components: Honeyfolder, Audit Watch, and Complex</p>

		<p>Event Processing(CEP). Honeyfolder serves as a decoy folder designed using the Social Leopard Algorithm (SoLA), enticing attackers and functioning as an early warning system for detecting suspicious file activities. AuditWatch, an Entropy module, assesses the entropy of files and folders to identify anomalies. The CEP engine aggregates data from diverse security systems to confirm ransomware behavior and promptly respond to attacks. Experimental testing conducted in a secured testbed using over 20 variants of recent ransomware validates the efficacy of the proposed IDH. Results indicate significant improvements in ransomware detection time, detection rate, and accuracy compared to existing ransomware detection models</p>
--	--	---

2.2 CHALLENGES PRESENT IN EXISTING SYSTEM

In the existing system of "Intrusion Detection System Using Honeypots in a Cloud Environment," several challenges present themselves, hindering its efficacy and efficiency.

Firstly, scalability poses a significant challenge. Traditional intrusion detection systems may struggle to adapt to the dynamic and elastic nature of cloud environments. The rapid provisioning and deprovisioning of resources in cloud environments can lead to difficulties in maintaining consistent coverage and detection capabilities across the entire infrastructure. As the number of virtual instances fluctuates in response to varying workloads and demand, ensuring that honeypots are adequately deployed and managed to cover the entire cloud environment becomes a complex task. Without scalable solutions in

place, organizations risk gaps in coverage and may overlook potential threats, undermining the effectiveness of the intrusion detection system.

Integration complexity is another challenge in the existing system. Integrating honeypots into the existing cloud security architecture requires careful planning and coordination. Cloud environments often consist of diverse technologies, platforms, and services, each with its own set of security mechanisms and configurations. Compatibility issues, resource constraints, and varying security requirements can complicate the integration process, leading to delays and operational challenges. Additionally, ensuring seamless communication and data exchange between honeypots and other security components, such as firewalls and SIEM (Security Information and Event Management) systems, further exacerbates the integration complexity.

False positives present a significant challenge in intrusion detection systems using honeypots. Honeypots are designed to attract and deceive attackers, mimicking legitimate services and assets within the network. However, this deception can sometimes lead to false alarms or alerts triggered by benign activities or legitimate users. Distinguishing between genuine threats and false positives requires sophisticated detection mechanisms and advanced analytics capabilities. Without effective filtering and analysis techniques in place, security teams may waste valuable time and resources investigating false alarms, leading to alert fatigue and decreased responsiveness to genuine security incidents.

Moreover, evasion techniques employed by sophisticated attackers pose a formidable challenge to intrusion detection systems using honeypots in cloud environments. Attackers may employ evasion tactics to bypass honeypots and evade detection, such as obfuscating malicious payloads, encrypting communication channels, or exploiting vulnerabilities in honeypot configurations. Detecting and mitigating these evasion techniques require continuous monitoring, regular updates to honeypot configurations, and the implementation of advanced detection mechanisms, such as behavior-based analysis and anomaly detection. Without robust evasion detection capabilities, organizations risk overlooking stealthy attacks and failing to protect their cloud assets effectively.

In summary, the existing system of "Intrusion Detection System Using Honeypots in a Cloud Environment" faces several challenges, including scalability, integration complexity, false positives, and evasion techniques. Addressing these challenges requires innovative solutions, such as scalable deployment mechanisms, seamless integration with existing security architectures, advanced analytics capabilities, and robust evasion detection techniques. By overcoming these challenges, organizations can enhance the effectiveness and resilience of their intrusion detection systems and better protect their

cloud environments against emerging cyber threats. Mitigating these challenges will ensure that intrusion detection systems using honeypots remain a valuable asset in the defense against cyber threats in cloud environments, bolstering security and safeguarding critical assets and data.

CHAPTER 3

REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

- Processor – AMD/INTEL
- Clock Speed – 2.9 GHz or above
- Storage – 150 gb Minimum
- Peripherals – Mouse and Keyboard
- Monitor – Any Monitor

3.2 SOFTWARE REQUIREMENTS

- AWS
- Linux
- Python
- Bash
- Snort

3.3 BUDGET

Procured Items/Components for the Project work	Total Cost
Item 1	0
Item 2	0
Item 3	0
Total Budget (INR)	0

Note: Please Refer the Appendices A for more details and select suitable items as per your project requirements.

CHAPTER 4

ANALYSIS & DESIGN

4.1 PROPOSED METHODOLOGY

Implementing an intrusion detection system (IDS) using honeypots within a cloud environment involves a structured methodology aimed at maximizing security efficacy while leveraging the scalability and flexibility of cloud infrastructure.

Firstly, defining clear objectives is paramount. Organizations must delineate the specific threats they aim to detect, the level of visibility required, and the desired outcomes of the IDS deployment. This initial step provides a guiding framework for subsequent decisions and actions.

Selecting an appropriate cloud provider is the next crucial step. Factors such as scalability, reliability, compliance capabilities, and the availability of necessary services for IDS deployment must be considered. Choosing a provider that aligns with the organization's requirements ensures a solid foundation for implementing the IDS using honeypots.

Designing the honeypot architecture follows suit, wherein organizations map out the deployment plan for honeypots within the cloud environment. This involves determining the types of honeypots to deploy (e.g., low-interaction or high-interaction) and their strategic placement within the network architecture to maximize visibility and lure potential attackers effectively.

Once the architecture is outlined, the deployment of honeypots begins. This step entails the actual implementation of the designed honeypot architecture within the cloud environment, ensuring meticulous configuration and isolation to prevent interference with production systems while minimizing security risks.

Integrating the IDS with the deployed honeypots comes next. This involves configuring the IDS to collect and analyze data from the honeypots, detect suspicious activities, and generate alerts for further investigation. The seamless integration of the IDS with honeypots is essential for effective monitoring and analysis of potential threats within the cloud environment.

Fine-tuning the detection rules of the IDS is a critical step in optimizing its effectiveness. Organizations must adjust thresholds, filters, and signatures based on the specific characteristics of the deployed honeypots and anticipated attack scenarios to accurately differentiate between legitimate and malicious activities.

Continuous monitoring and analysis of network traffic captured by the honeypots and processed by the IDS are imperative. This ongoing surveillance enables organizations to identify patterns, trends, and anomalies indicative of unauthorized access attempts, exploitation, or reconnaissance activities within the cloud environment.

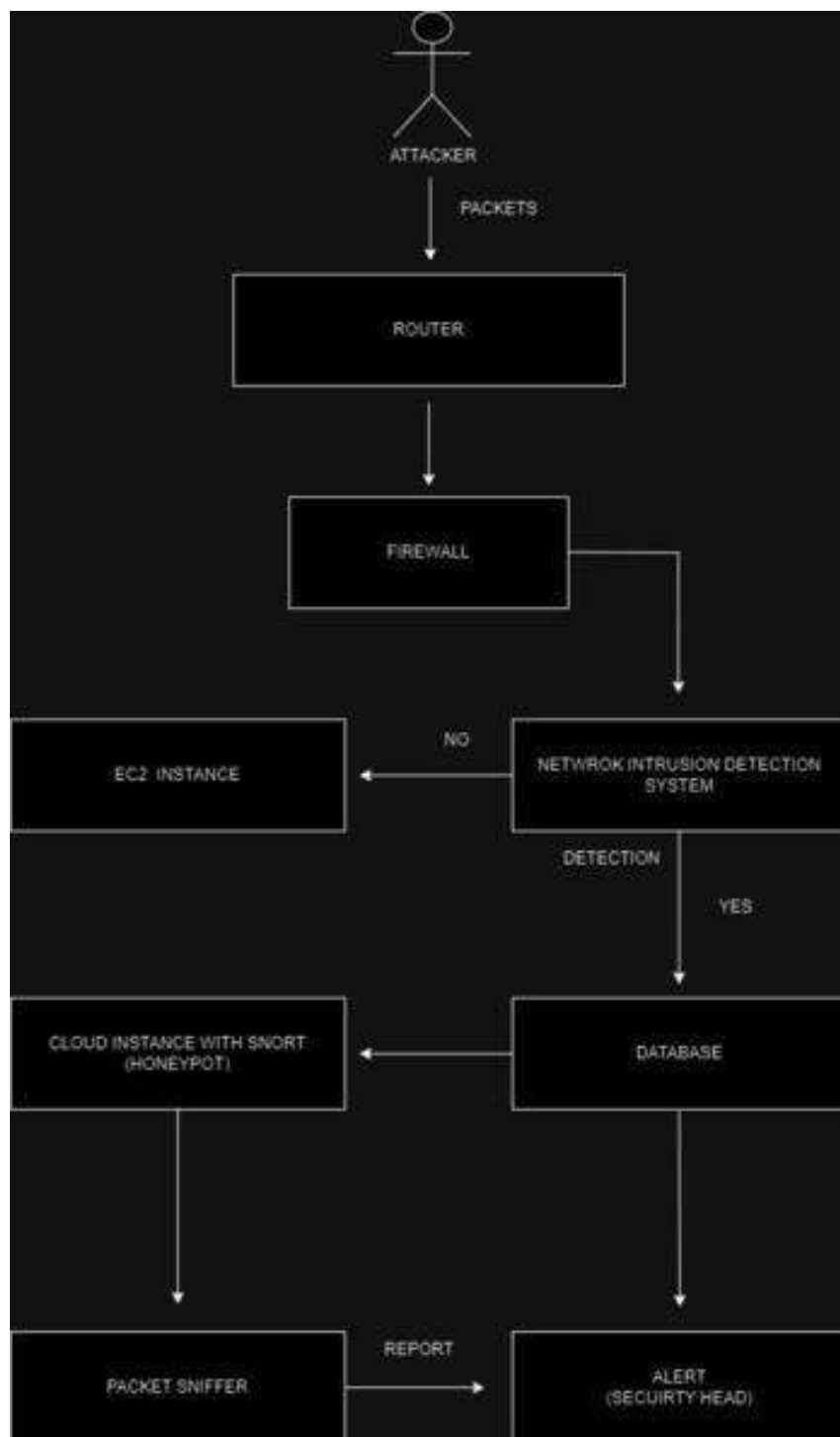
Configuring the IDS to generate alerts for detected security events and defining appropriate response actions is essential for incident management. Establishing clear escalation procedures for responding to alerts ensures timely incident triage, investigation, and mitigation steps, thereby minimizing potential damage.

Regular evaluation and refinement of the IDS deployment are necessary to maintain its effectiveness. Organizations should analyze alert data, incident reports, and feedback from security operations to identify areas for improvement and enhance the detection and response capabilities continuously.

Lastly, maintaining security and compliance is a continual effort. Organizations must implement measures to ensure the ongoing security of the IDS deployment, including regular updates and patches, adherence to security best practices, and compliance with relevant regulatory requirements in the cloud environment.

By following this structured methodology, organizations can effectively leverage honeypots within a cloud environment to bolster their intrusion detection capabilities, detect emerging threats, and fortify their overall cybersecurity posture.

4.2 SYSTEM ARCHITECTURE



4.3 MODULE DESCRIPTIONS

Packet sniffers are indispensable tools in network administration, security analysis, and troubleshooting. Their primary function involves intercepting, capturing, and logging data packets traversing a network, containing crucial information about communication between devices, including source and destination IP addresses, ports, protocols, and payload data. Utilizing techniques like promiscuous mode, they capture packets across network interfaces, allowing real-time analysis or storage for later examination.

Their utility spans various areas:

1. **Network Troubleshooting** : Packet sniffers assist in diagnosing issues like misconfigured devices, network congestion, or abnormal traffic patterns, aiding network administrators in swift problem resolution.
2. **Bandwidth Monitoring** : By analyzing traffic volume and types, administrators can identify bandwidth-intensive applications or users, optimizing network performance and enforcing usage policies.
3. **Security Analysis** : Packet sniffers play a critical role in detecting suspicious or malicious activities like unauthorized access attempts or malware infections. Security analysts use packet headers and payloads to identify threats and take countermeasures.
4. **Protocol Analysis** : They allow in-depth scrutiny of network protocols and communication patterns, aiding in detecting anomalies and ensuring compliance with standards.
5. **Forensic Investigation**: Essential for forensic investigators, packet sniffers reconstruct network sessions and analyze packet contents related to security incidents or cyber attacks, aiding in identifying attackers and gathering evidence.

However, ethical and responsible usage respecting privacy and legal considerations is crucial.

Snort honeypots leverage Snort IDS capabilities to create decoy environments, attracting attackers to reveal their tactics. Configured with custom rulesets, they mimic vulnerable systems, enabling monitoring and analysis of malicious activities. Key components include decoy systems, Snort configuration, logging, alerting, and analysis. Benefits include early threat detection, threat intelligence generation, deception tactics, and research opportunities, though deploying and managing them requires careful planning and security considerations to prevent misuse.

Network Intrusion Detection Systems (NIDS) are crucial for real-time monitoring and analysis of network traffic to detect suspicious activities. Functions include packet capture, signature-based and anomaly-based detection, alert generation, and integration with SIEM systems. NIDS benefits include real-time threat detection, continuous monitoring, enhanced visibility, and regulatory compliance support. However, effective deployment requires meticulous planning, configuration, and maintenance to ensure optimal performance.

AWS EC2 instances running Linux offer scalable and flexible computing resources in the cloud. Key features include scalability, flexibility in choosing Linux distributions, robust security measures, high availability, integration with AWS services, management, monitoring tools, and cost management options. They empower organizations to deploy and manage applications securely without physical hardware constraints.

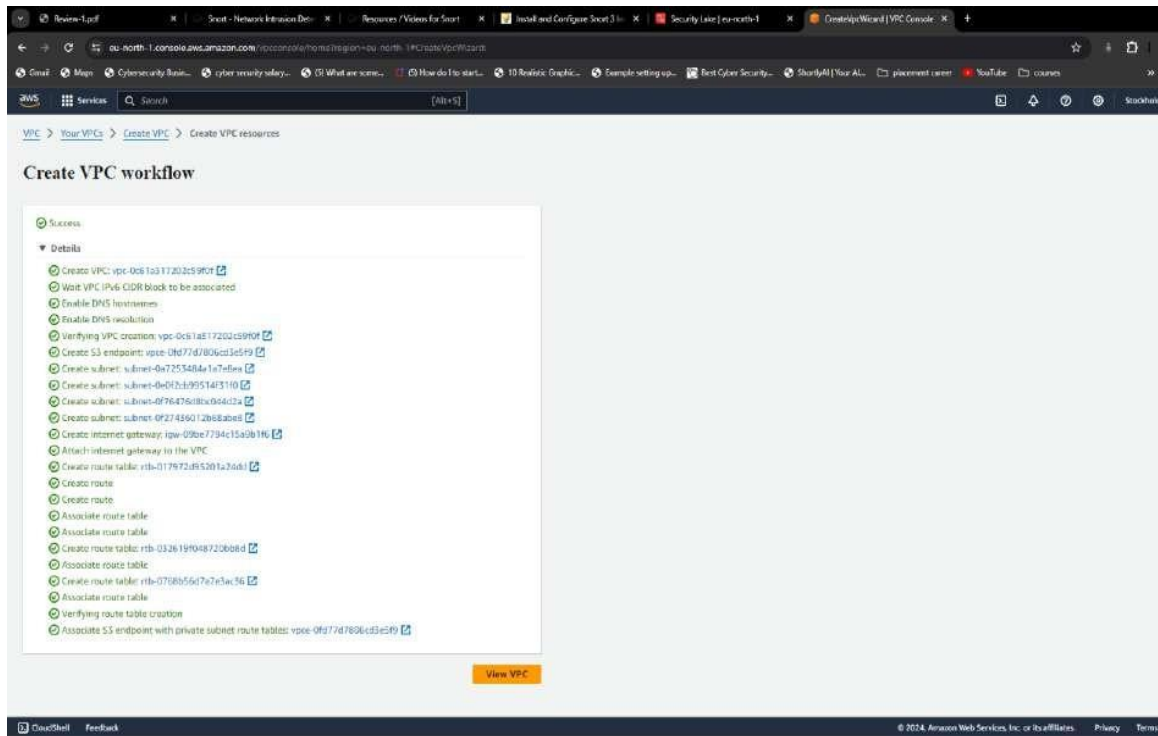
An IDS alert system is vital for generating, managing, and responding to security alerts triggered by IDS. Key components include alert generation, correlation, prioritization, escalation, enrichment, visualization, management, and integration with incident response workflows. Their role in incident detection, analysis, and response is pivotal, aiding organizations in identifying and mitigating security threats effectively.

CHAPTER 5

IMPLEMENTATION & TESTING

5.1 IMPLEMENTATION

Setting up a virtual private cloud in AWS



Opening VPC Feature and creating a vpc in a selected region.

VPC configuration setup

The screenshot displays the AWS VPC console for a VPC named 'vpc-0c61a317202c59f0f / Sentinel-vpc'. The left sidebar shows the navigation menu with categories like Virtual private cloud, Security, DNS Firewall, and Network Firewall. The main content area is divided into 'Details' and 'Resource map' sections.

Details:

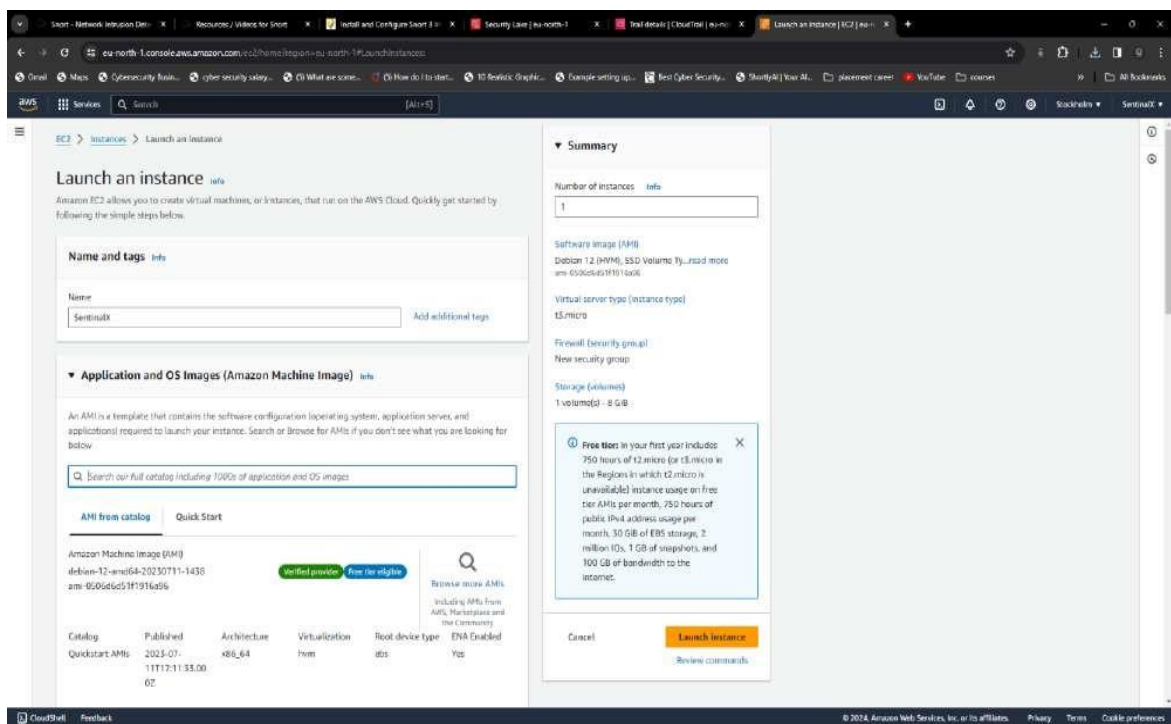
- VPC ID: vpc-0c61a317202c59f0f
- Status: Available
- Tenancy: Default
- Default VPC: No
- Network Address Usage metrics: Disabled
- DNS hostnames: Enabled
- Main route table: rtp-07f9dd8a1cf490deec
- IPv4 CIDR: 10.0.0.0/16
- Route 53 Resolver DNS Firewall rule groups: --
- DNS resolution: Enabled
- Main network ACL: acl-0796dc00d24882d1
- IPv6 CIDR (Network border group): 2a05:0018:a6a:9400::/56 (eu-north-1) Associated
- Owner ID: 739535344629

Resource map:

- VPC:** Sentinel-vpc
- Subnets (4):**
 - eu-north-1a: Sentinel-subnet-public1-eu-north-1a, Sentinel-subnet-private1-eu-north-1a
 - eu-north-1b: Sentinel-subnet-public2-eu-north-1b, Sentinel-subnet-private2-eu-north-1b
- Route tables (4):** Sentinel-rtp-private1-eu-north-1a, rtp-07f9dd8a1cf490deec, Sentinel-rtp-private2-eu-north-1b, Sentinel-rtp-public
- Network connections (2):** Sentinel-vpc-s3, Sentinel-vpc-s3

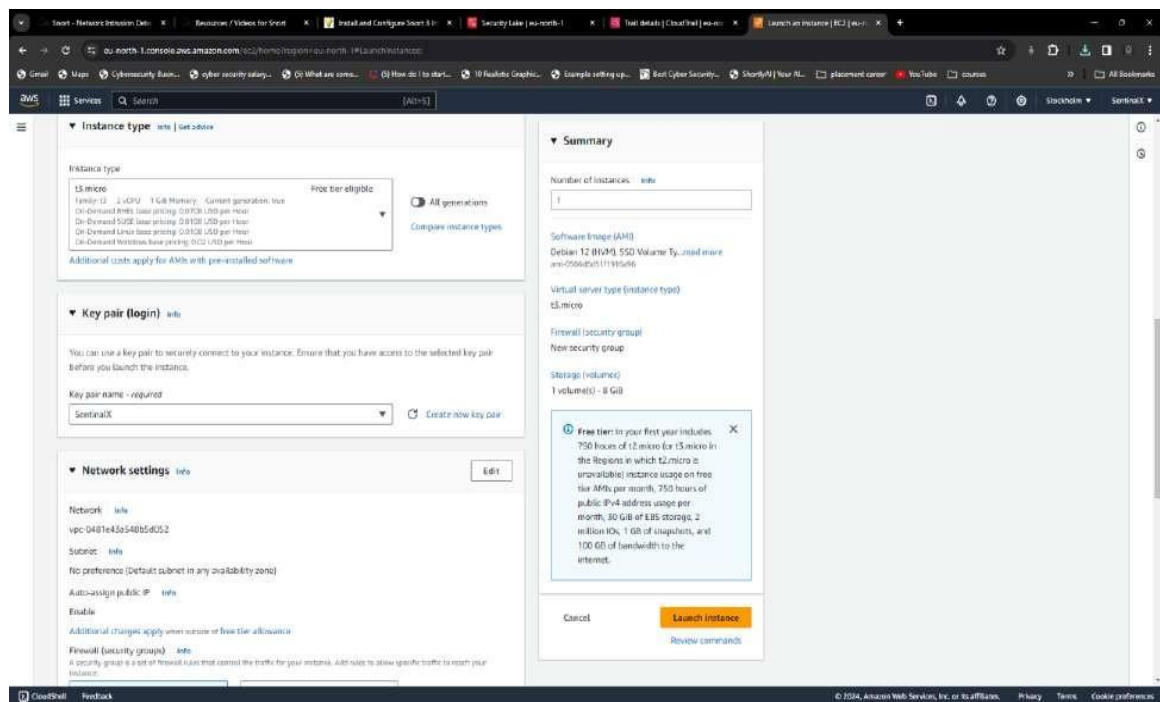
Configuring the vpc and region and the required subnets.

EC2 Instance launch and setup



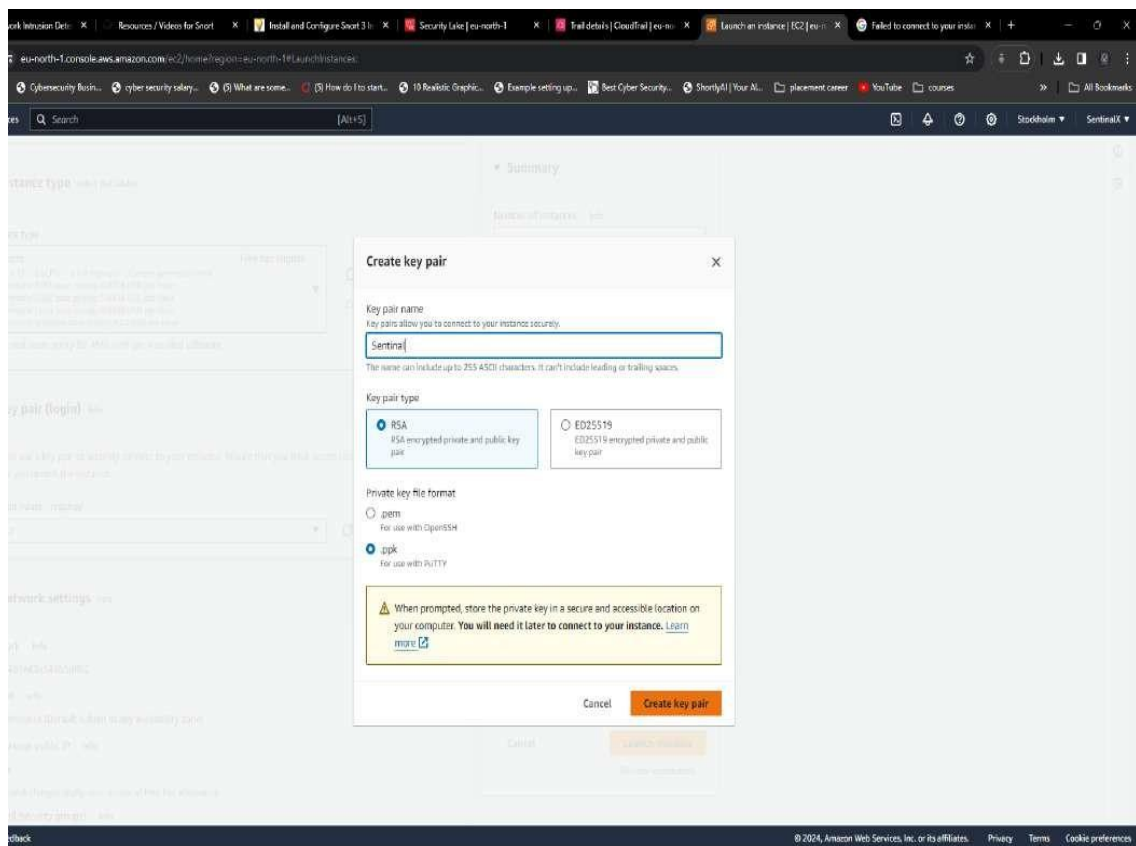
Launching a new ec2 instance within the vpc that's been created

Operating System Selection



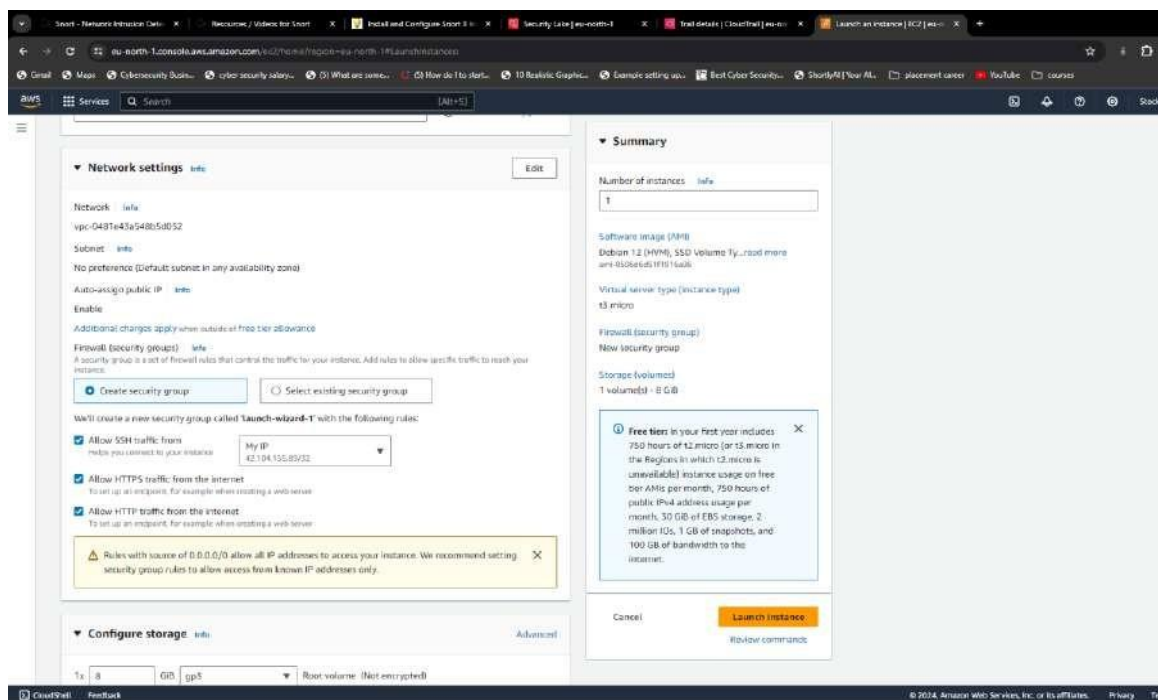
Selecting the basic linux distro(debian) as operating system.

Setting up key pair(RSA and Putty)in.ppk



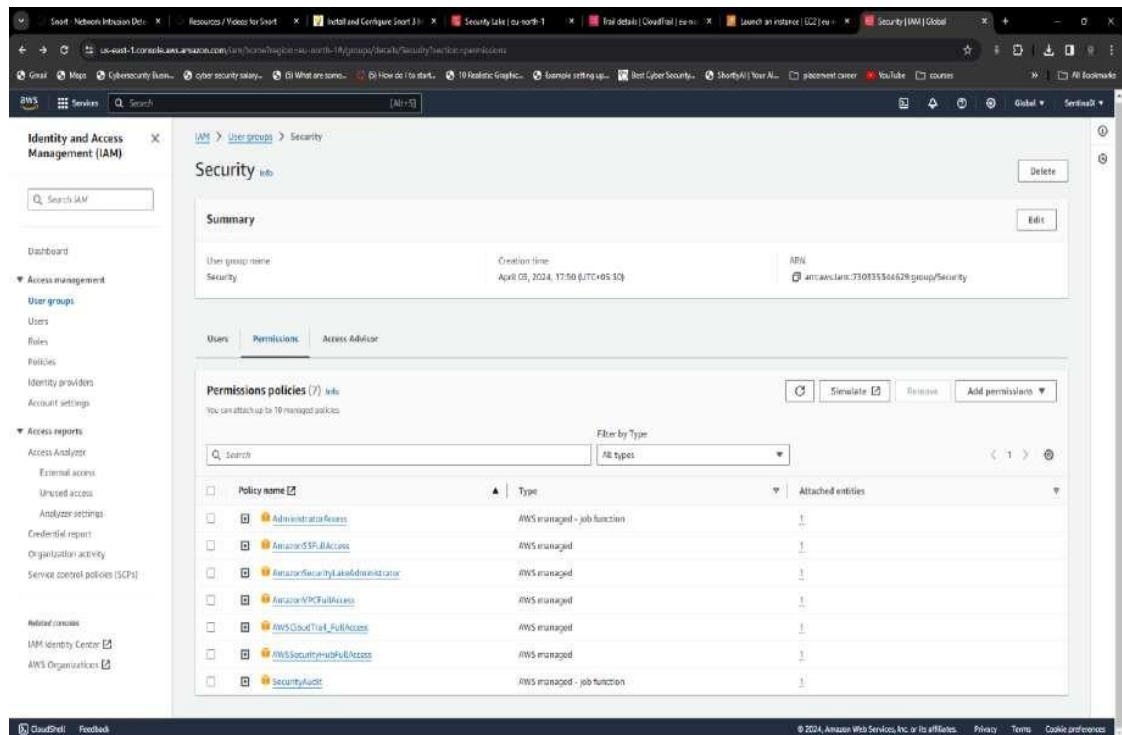
Generating a Key pair containing public and private key in encrypted manner(RSA).

EC2 Instance Network settings Configuration



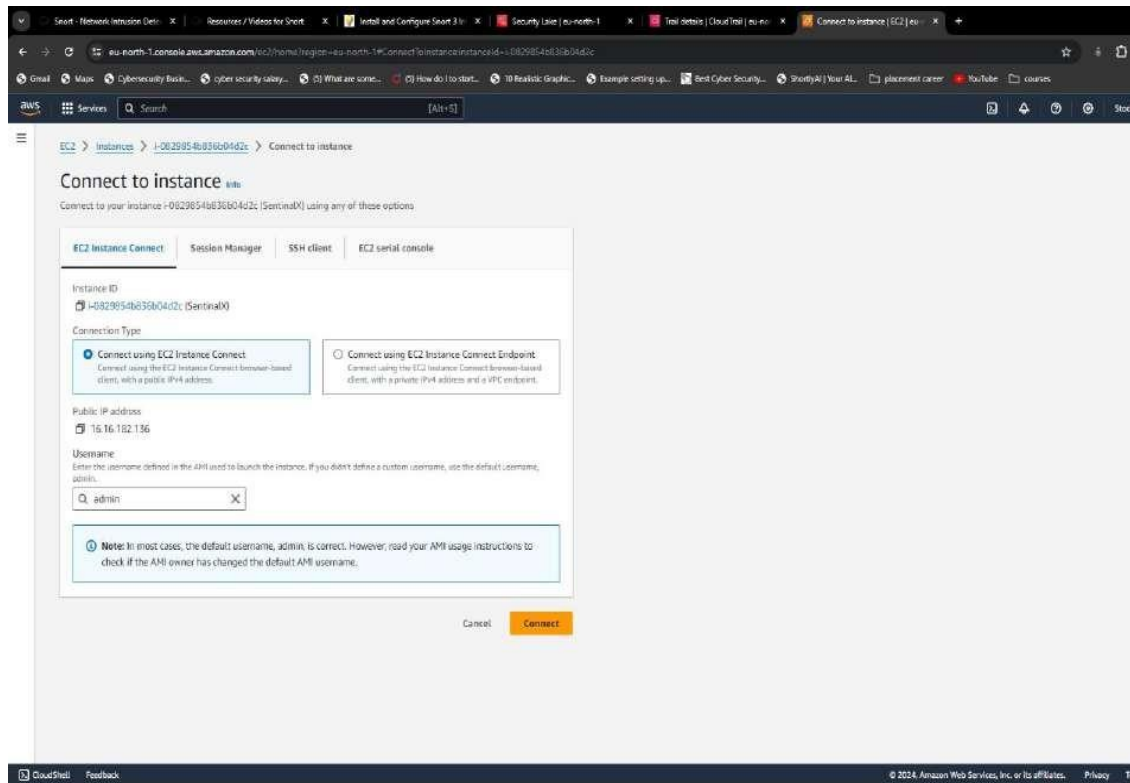
Configuring traffic rules and DHCP Allocations(Network Configuration).

Security policy creation and additions



Security Policy Creation for User and the User group handling the instance.

Launchpad of instance



Initiating and Connecting to the Instance.

EC2 Instances dashboard

eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#instances

Cybersecurity Busin...cyber security safety...What are some...How do I to start...10 Realistic Graphic...Example setting up...Best Cyber Security...ShortlyAI | Your AI...placement careerYouTubecourses»

Search[Alt+S]

Stackholm

Instances (1/1) info

Find instance by attribute or tag (case-sensitive)All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
SentinalX	i-0829854b836b04d2c	Running	t3.micro	Initializing	View alarms	eu-north-1a	ec2-16-162-136.eu-...	16.16.182.136	-

Instance: i-0829854b836b04d2c (SentinalX)

Details

Status and alarms new

Monitoring

Security

Networking

Storage

Tags

Instance summary info

Instance ID
i-0829854b836b04d2c (SentinalX)

IPv6 address
-

Hostname type
IP name: ip-172-31-30-217.eu-north-1.compute.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
16.16.182.136 [Public IP]

Public IPv4 address
16.16.182.136 [open address]

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-30-217.eu-north-1.compute.internal

Instance type
t3.micro

VPC ID
vpc-0461e43a548b5d052

Private IPv4 addresses
172.31.30.217

Public IPv4 DNS
ec2-16-162-136.eu-north-1.compute.amazonaws.com [open add]

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

© 2024 Amazon Web Services, Inc. or its affiliates. PrivacyTerms

User/Root permissions and group permissions

The screenshot displays the AWS IAM console interface for a user named 'SentinalX'. The top navigation bar shows the user's name and the 'Permissions' tab is selected. The 'Summary' section provides details about the user, including their ARN, creation date, and console access status. Below this, the 'Permissions policies (7)' section lists the policies attached to the user, such as 'AdministratorAccess', 'AmazonSSFullAccess', and 'AmazonSecurityLakeAdministrator'. The table includes columns for Policy name, Type, and Attached via.

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Group Security
AmazonSSFullAccess	AWS managed	Group Security
AmazonSecurityLakeAdministrator	AWS managed	Group Security
AmazonVPCFullAccess	AWS managed	Group Security
AWSCloudTrail_FullAccess	AWS managed	Group Security
AWSSecurityHubFullAccess	AWS managed	Group Security
SecurityAudit	AWS managed - job function	Group Security

User group Permissions and Access Controls.

Security Policies for Group and Roles

Resources / Videos for Sncort x Install and Configure Sncort 3.1 x Security Lake | eu-north-1 x Trail details | CloudTrail | eu-north-1 x Instances | EC2 | eu-north-1 x Security | IAM | Global

aws.amazon.com/iam/home?region=eu-north-1#/groups/detail/Security/section:permissions

cyber security salary... What are some... (5) How do I to start... 10 Realistic Graphic... Example setting up... Best Cyber Security... ShortlyAI | Your AI... placement career YouTube courses

[Alt+S]

IAM > User groups > Security

Security Info

Delete

Summary

Edit

User group name: Security

Creation time: April 03, 2024, 17:50 (UTC+05:30)

ARN: arn:aws:iam::730335344629:group/Security

Users (1) Permissions Access Advisor

Permissions policies (7) Info








You can attach up to 10 managed policies.

Filter by Type

Search

All types

< 1 > ⚙

<input type="checkbox"/>	Policy name	Type	Attached entities
<input type="checkbox"/>	 AdministratorAccess	AWS managed - job function	2
<input type="checkbox"/>	 AmazonS3FullAccess	AWS managed	2
<input type="checkbox"/>	 AmazonSecurityLakeAdministrator	AWS managed	2
<input type="checkbox"/>	 AmazonVPCFullAccess	AWS managed	2
<input type="checkbox"/>	 AWSCloudTrail_FullAccess	AWS managed	2
<input type="checkbox"/>	 AWSSecurityHubFullAccess	AWS managed	2
<input type="checkbox"/>	 SecurityAudit	AWS managed - job function	1

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cook

Adding and Updating Securty Policies necessarily.

Role update and permission alteration

The screenshot displays the AWS IAM console interface for the 'Sec_Head' role. The breadcrumb navigation shows 'IAM > Roles > Sec_Head'. The role's description is 'Allows EC2 instances to call AWS services on your behalf.' The 'Summary' section provides key details: Creation date (April 03, 2024, 17:56 (UTC+05:30)), ARN (arn:aws:iam::730335344629:role/Sec_Head), Instance profile ARN (arn:aws:iam::730335344629:instance-profile/Sec_Head), and Maximum session duration (1 hour). Below the summary, tabs for 'Permissions', 'Trust relationships', 'Tags', 'Access Advisor', and 'Revoke sessions' are visible. The 'Permissions policies (7)' section shows a list of managed policies attached to the role. A search bar and a 'Filter by Type' dropdown (set to 'All types') are present above the table. The table lists seven policies, all of which are 'AWS managed'.

Policy name	Type	Attached entities
AdministratorAccess	AWS managed - job function	2
AmazonEC2FullAccess	AWS managed	1
AmazonS3FullAccess	AWS managed	2
AmazonSecurityLakeAdministrator	AWS managed	2
AmazonVPCFullAccess	AWS managed	2
AWSCloudTrail_FullAccess	AWS managed	2

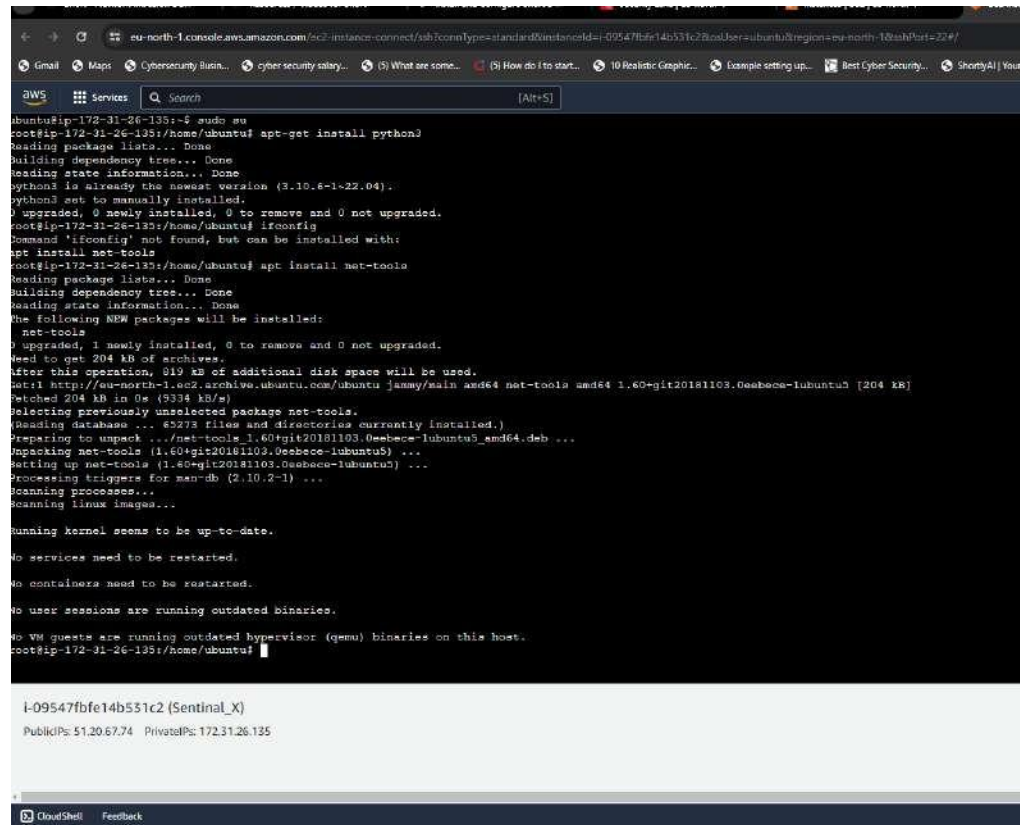
Connecting to instance

The screenshot shows the AWS Management Console interface for connecting to an EC2 instance. The browser address bar shows the URL: `eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#ConnectToInstance:instanceid=i-0829854b836b04d2c`. The console breadcrumb navigation is `EC2 > Instances > i-0829854b836b04d2c > Connect to instance`. The main heading is `Connect to instance` with an `info` icon. Below the heading, it says: `Connect to your instance i-0829854b836b04d2c (SentinelX) using any of these options:`. There are four tabs: `EC2 Instance Connect`, `Session Manager`, `SSH client` (which is selected), and `EC2 serial console`. Under the `SSH client` tab, the `Instance ID` is `i-0829854b836b04d2c (SentinelX)`. A list of steps is provided:

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is `SentinelX.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable:
`chmod 400 "SentinelX.pem"`
4. Connect to your instance using its Public DNS:
`ec2-16-162-136-eu-north-1.compute.amazonaws.com`

 An `Example:` section shows the command: `ssh -i "SentinelX.pem" admin@ec2-16-162-136-eu-north-1.compute.amazonaws.com`. A `Note:` box states: `Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.` At the bottom right of the panel is a `Cancel` button. The footer of the console shows `GetShell Feedback` and `© 2024 Amazon Web Services, Inc. or its affiliates. Privacy Terms`.

Instance Initialized



```
eu-north-1.console.aws.amazon.com/ec2-instance-connect/sb?connType=standard&instanceId=i-09547fbfe14b551c2&osUser=ubuntu&region=eu-north-1&subPort=22#/?
AWS Services Search [Alt+S]

ubuntu@ip-172-31-26-135:~$ sudo su
root@ip-172-31-26-135:/home/ubuntu# apt-get install python3
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.10.6-1~22.04).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@ip-172-31-26-135:/home/ubuntu# ifconfig
Command 'ifconfig' not found, but can be installed with:
apt install net-tools
root@ip-172-31-26-135:/home/ubuntu# apt install net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 204 kB of archives.
After this operation, 919 kB of additional disk space will be used.
Get:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 net-tools amd64 1.60-glibc20181103.0esbeece-1ubuntu3 [204 kB]
Fetched 204 kB in 0s (9334 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 65273 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60-glibc20181103.0esbeece-1ubuntu3_amd64.deb ...
Unpacking net-tools (1.60-glibc20181103.0esbeece-1ubuntu3) ...
Setting up net-tools (1.60-glibc20181103.0esbeece-1ubuntu3) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

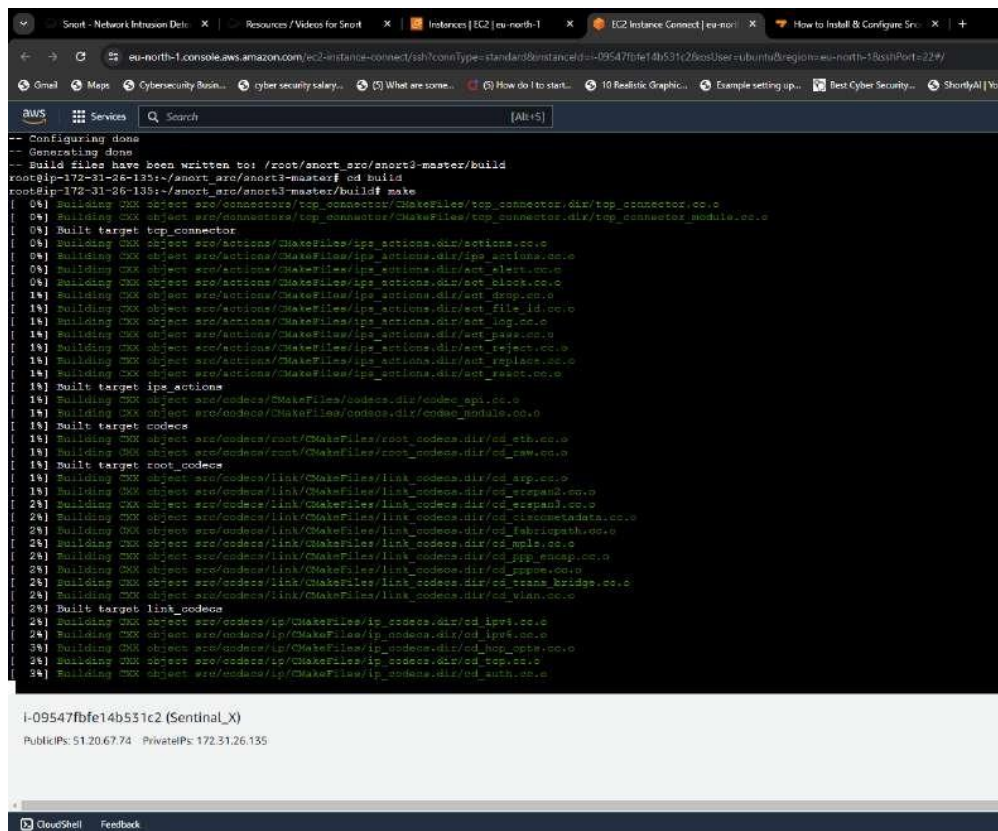
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-26-135:/home/ubuntu#
```

i-09547fbfe14b551c2 (Sentinel_X)

PublicIPs: 51.20.67.74 PrivateIPs: 172.31.26.135

CloudShell Feedback

Installing Dependencies for Snort Honey Pot

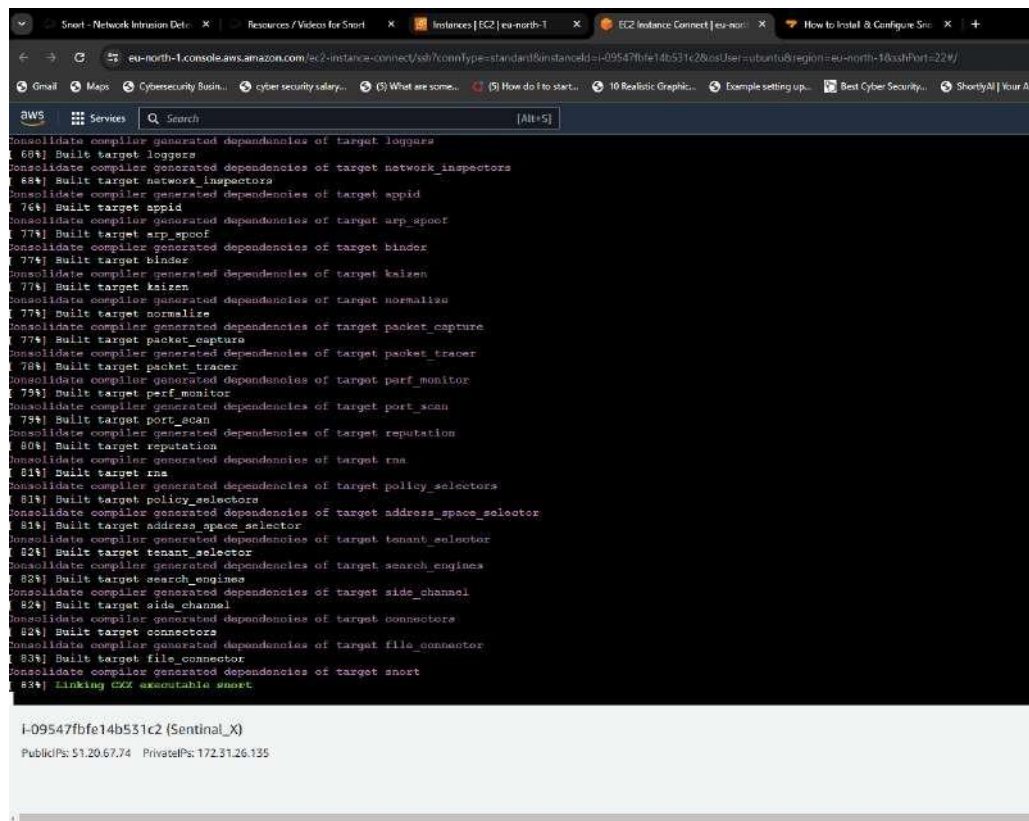


```
-- Configuring done
-- Generating done
-- Build files have been written to: /root/snort_pro/snort3-master/build
root@ip-172-31-26-135:~/snort_pro/snort3-master# cd build
root@ip-172-31-26-135:~/snort_pro/snort3-master/build# make
[ 0%] Building CXX object src/connector/CMakeFiles/top_connector.dir/top_connector.cc.o
[ 0%] Building CXX object src/connector/CMakeFiles/top_connector.dir/top_connector_module.cc.o
[ 0%] Built target top_connector
[ 0%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/actions.cc.o
[ 0%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/ip_actions.cc.o
[ 0%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_elect.cc.o
[ 0%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_block.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_drop.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_file_id.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_log.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_pass.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_reject.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_replace.cc.o
[ 1%] Building CXX object src/actions/CMakeFiles/ip_actions.dir/act_reset.cc.o
[ 1%] Built target ip_actions
[ 1%] Building CXX object src/codecs/CMakeFiles/codecs.dir/codecs_api.cc.o
[ 1%] Building CXX object src/codecs/CMakeFiles/codecs.dir/codecs_module.cc.o
[ 1%] Built target codecs
[ 1%] Building CXX object src/codecs/root/CMakeFiles/root_codecs.dir/ed_ath.cc.o
[ 1%] Building CXX object src/codecs/root/CMakeFiles/root_codecs.dir/ed_raw.cc.o
[ 1%] Built target root_codecs
[ 1%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_arp.cc.o
[ 1%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_ethernet2.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_ethernet3.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_ethernetmetadata.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_fabricpath.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_mpls.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_gpp_encap.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_ipv6.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_ipv6_bridge.cc.o
[ 2%] Building CXX object src/codecs/link/CMakeFiles/link_codecs.dir/ed_vlan.cc.o
[ 2%] Built target link_codecs
[ 2%] Building CXX object src/codecs/ip/CMakeFiles/ip_codecs.dir/ed_ipv4.cc.o
[ 2%] Building CXX object src/codecs/ip/CMakeFiles/ip_codecs.dir/ed_ipv6.cc.o
[ 3%] Building CXX object src/codecs/ip/CMakeFiles/ip_codecs.dir/ed_ip_options.cc.o
[ 3%] Building CXX object src/codecs/ip/CMakeFiles/ip_codecs.dir/ed_top.cc.o
[ 3%] Building CXX object src/codecs/ip/CMakeFiles/ip_codecs.dir/ed_auth.cc.o
```

i-09547fbfe14b531c2 (Sentinal_X)
PublicIPs: 51.20.67.74 PrivateIPs: 172.31.26.135

CloudShell Feedback

Installing Snort

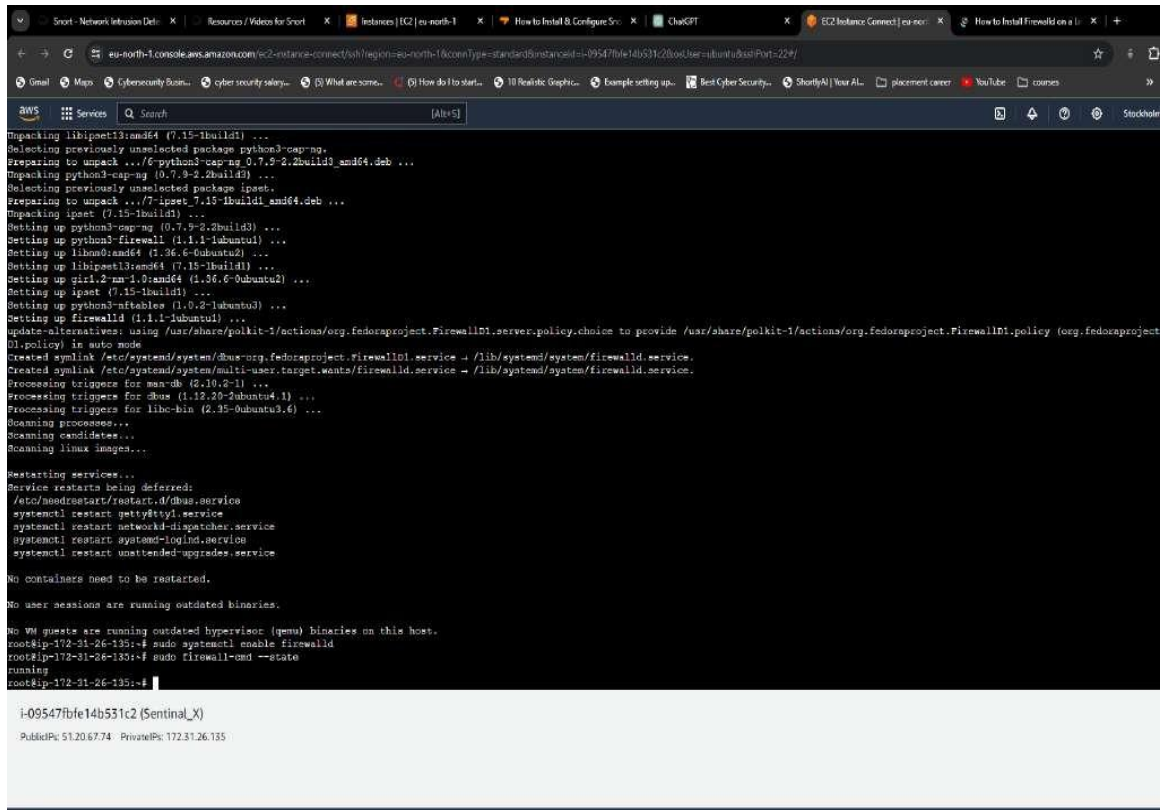


The screenshot shows the AWS Management Console interface for an EC2 instance named 'eu-north-1'. The browser tabs include 'Snort - Network Intrusion Detection', 'Resources / Videos for Snort', 'Instances | EC2 | eu-north-1', 'EC2 Instance Connect | eu-north-1', and 'How to Install & Configure Snort'. The console displays the output of the Snort installation process, showing the compilation and linking of various targets. The progress is as follows:

```
Consolidate compiler generated dependencies of target loggers
[ 68%] Built target loggers
Consolidate compiler generated dependencies of target network_inspectors
[ 68%] Built target network_inspectors
Consolidate compiler generated dependencies of target appid
[ 76%] Built target appid
Consolidate compiler generated dependencies of target arp_spoof
[ 77%] Built target arp_spoof
Consolidate compiler generated dependencies of target binder
[ 77%] Built target binder
Consolidate compiler generated dependencies of target kaizen
[ 77%] Built target kaizen
Consolidate compiler generated dependencies of target normalize
[ 77%] Built target normalize
Consolidate compiler generated dependencies of target packet_capture
[ 77%] Built target packet_capture
Consolidate compiler generated dependencies of target packet_tracer
[ 78%] Built target packet_tracer
Consolidate compiler generated dependencies of target perf_monitor
[ 79%] Built target perf_monitor
Consolidate compiler generated dependencies of target port_scan
[ 79%] Built target port_scan
Consolidate compiler generated dependencies of target reputation
[ 80%] Built target reputation
Consolidate compiler generated dependencies of target rna
[ 81%] Built target rna
Consolidate compiler generated dependencies of target policy_selectors
[ 81%] Built target policy_selectors
Consolidate compiler generated dependencies of target address_space_selector
[ 81%] Built target address_space_selector
Consolidate compiler generated dependencies of target tenant_selector
[ 82%] Built target tenant_selector
Consolidate compiler generated dependencies of target search_engines
[ 83%] Built target search_engines
Consolidate compiler generated dependencies of target side_channel
[ 82%] Built target side_channel
Consolidate compiler generated dependencies of target connectors
[ 82%] Built target connectors
Consolidate compiler generated dependencies of target file_connector
[ 83%] Built target file_connector
Consolidate compiler generated dependencies of target snort
[ 83%] Linking CXX executable snort
```

Below the output, the instance ID is shown as 'i-09547fbfe14b531c2 (Sentinal_X)' and the public IP address is '51.20.67.74'. The private IP address is '172.31.26.135'.

Installed and Running FireWall(Firewalld)



```
eu-north-1.console.aws.amazon.com/ec2-instance-connect/v1/regions/eu-north-1/connections/standard?instanceId=i-09547f14b531c220&user=ubuntu@bss:Port=22#/?
AWS Services Search [Alt+S]
Stockholm

Unpacking libipset13:amd64 (7.15-1build1) ...
Selecting previously unselected package python3-capng.
Preparing to unpack .../6/python3-capng_0.7.9-2.2build3_amd64.deb ...
Unpacking python3-capng (0.7.9-2.2build3) ...
Selecting previously unselected package ipset.
Preparing to unpack .../7-ipset_7.15-1build1_amd64.deb ...
Unpacking ipset (7.15-1build1) ...
Setting up python3-capng (0.7.9-2.2build3) ...
Setting up python3-firewall (1.1.1-1ubuntu1) ...
Setting up libnss0:amd64 (2.36.6-0ubuntu2) ...
Setting up libipset13:amd64 (7.15-1build1) ...
Setting up glib2.0-bin-1.0:amd64 (1.36.6-0ubuntu2) ...
Setting up ipset (7.15-1build1) ...
Setting up python3-nftables (1.0.2-1ubuntu3) ...
Setting up firewalld (1.1.1-1ubuntu1) ...
update-alternatives: using /usr/share/polkit-1/actions/org.fedoraproject.FirewallD1.server.policy.choice to provide /usr/share/polkit-1/actions/org.fedoraproject.FirewallD1.policy (org.fedoraproject.FirewallD1.policy) in auto mode
Created symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service → /lib/systemd/system/firewalld.service.
Created symlink /etc/systemd/system/multi-user.target.wants/firewalld.service → /lib/systemd/system/firewalld.service.
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.6) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Restarting services...
Service restarts being deferred:
/etc/passwd:restart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service

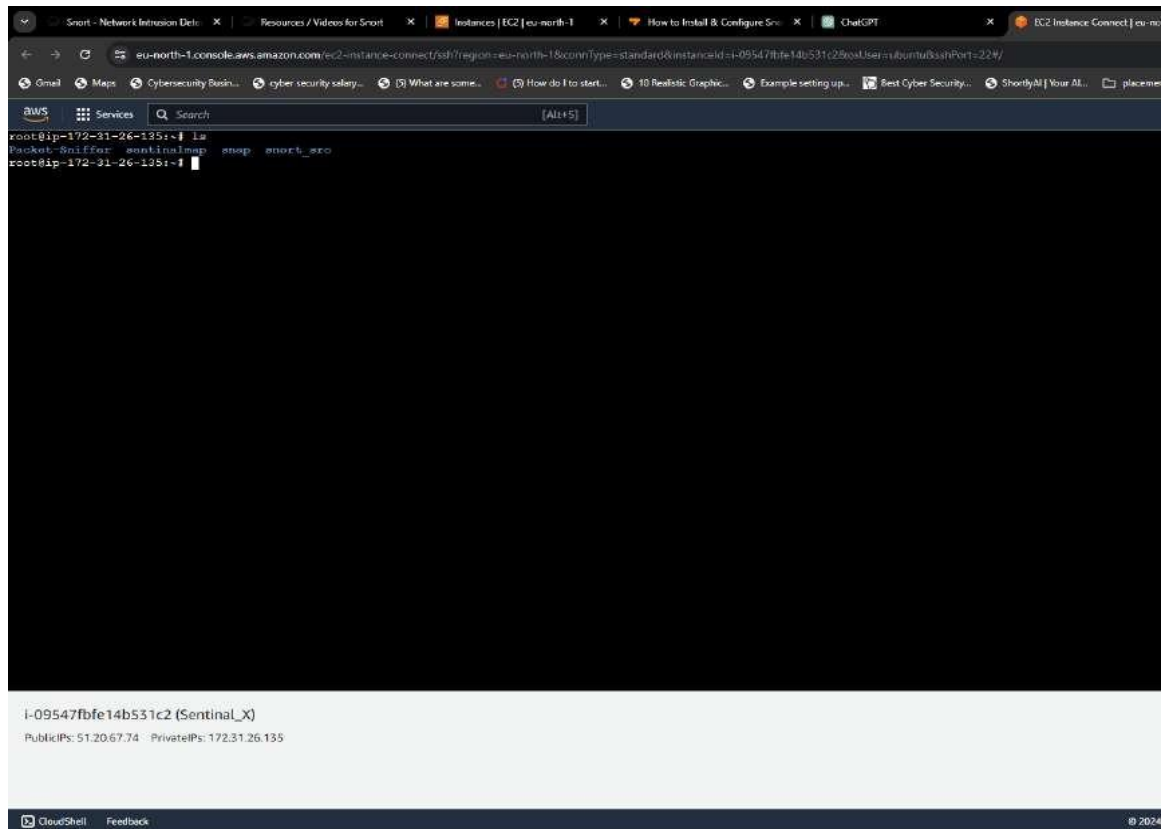
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-26-135:~# sudo systemctl enable firewalld
root@ip-172-31-26-135:~# sudo firewalld-cmd --state
running
root@ip-172-31-26-135:~#

i-09547f14b531c2 (Sentinel_X)
PublicIP: 51.20.67.74 PrivateIP: 172.31.26.135
```

Initialized packet sniffer code , mapping tool code and snort ids



The screenshot shows a terminal window within the AWS CloudShell interface. The terminal prompt is `root@ip-172-31-26-135:~#`. The user has run the command `ls`, which lists the files `Packet-Sniffer`, `snortlinuxmap`, `snop`, and `snort_xo`. The user then runs `snop snort_xo`, which appears to execute a script or command related to Snort configuration. The terminal output is mostly black, suggesting a large block of text or a full-screen application. Below the terminal, the AWS CloudShell interface shows the instance ID `i-09547fbfe14b531c2 (Sentinal_X)` and the public IP address `51.20.67.74`. The bottom of the screen shows the `CloudShell` logo and a `Feedback` link.

```
root@ip-172-31-26-135:~# ls
Packet-Sniffer  snortlinuxmap  snop  snort_xo
root@ip-172-31-26-135:~# snop snort_xo
```

i-09547fbfe14b531c2 (Sentinal_X)
PublicIPs: 51.20.67.74 · PrivateIPs: 172.31.26.135

CloudShell Feedback © 2024

Snort(Output)

Services

Search

Alt+S

Stockholm

SentinelX

PCP TTL:64 TOS:0x0 ID:34797 Iplen:20 DgLen:44 DF
A Seq: 0xC2B35745 Ack: 0xA0403D71 Win: 0x1E8 TopLen: 20
=====

04/17-11:45:45.470653 172.31.26.135:53870 -> 52.46.200.112:443
PCP TTL:64 TOS:0x0 ID:34798 Iplen:20 DgLen:361 DF
A Seq: 0xC2B35746 Ack: 0xA0403D71 Win: 0x1E8 TopLen: 20
=====

WARNING: No preprocessors configured for policy 0.
04/17-11:45:45.474816 52.46.200.112:443 -> 172.31.26.135:53870
PCP TTL:243 TOS:0x0 ID:14888 Iplen:20 DgLen:40 DF
A Seq: 0xA0403D71 Ack: 0xC2B26748 Win: 0x3D0 TopLen: 20
=====

WARNING: No preprocessors configured for policy 0.
04/17-11:45:45.474928 52.46.200.112:443 -> 172.31.26.135:53870
PCP TTL:243 TOS:0x0 ID:14889 Iplen:20 DgLen:40 DF
A Seq: 0xA0403D71 Ack: 0xC2B26859 Win: 0x3D0 TopLen: 20
=====

WARNING: No preprocessors configured for policy 0.
04/17-11:45:45.475054 52.46.200.112:443 -> 172.31.26.135:53870
PCP TTL:243 TOS:0x0 ID:14890 Iplen:20 DgLen:200 DF
A Seq: 0xA0403D71 Ack: 0xC2B26859 Win: 0x3D0 TopLen: 20
=====

WARNING: No preprocessors configured for policy 0.
04/17-11:45:45.475222 172.31.26.135:53870 -> 52.46.200.112:443
PCP TTL:64 TOS:0x0 ID:34799 Iplen:20 DgLen:40 DF
A Seq: 0xC2B26859 Ack: 0xA0403E31 Win: 0x1EA TopLen: 20
=====

WARNING: No preprocessors configured for policy 0.
04/17-11:45:45.475358 172.31.26.135:53870 -> 52.46.200.112:443
PCP TTL:64 TOS:0x0 ID:34800 Iplen:20 DgLen:40 DF
A Seq: 0xC2B26859 Ack: 0xA0403E31 Win: 0x1C3 TopLen: 20
=====

i-09547fbfe14b531c2 (Sentinel_X)

PublicIP: 51.20.67.74 PrivateIP: 172.31.26.135

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
Run time for packet processing was 16.391508 seconds
Snort processed 343 packets.
Snort ran for 0 days 0 hours 0 minutes 16 seconds
Pkt/s/sec: 21

Memory usage summary:
Total non-mapped bytes (arena): 786432
Bytes in mapped regions (hblkhd): 21590016
Total allocated space (usrblk): 682072
Total free space (fndblk): 103360
Topmost releasable block (keepcount): 100960

Packet I/O Totals:
Received: 343
Analyzed: 343 ( 88.860%)
Dropped: 0 ( 0.000%)
Filtered: 0 ( 0.000%)
Outstanding: 43 ( 11.140%)
Injected: 0

Breakdown by protocol (includes rebuilt packets):
Eth: 343 (100.000%)
VLAN: 0 ( 0.000%)
IP4: 343 (100.000%)
Frag: 0 ( 0.000%)
ICMP: 0 ( 0.000%)
UDP: 4 ( 1.164%)
TCP: 338 (98.836%)
IP6: 0 ( 0.000%)
IP6 Ext: 0 ( 0.000%)
IP6 Opt: 0 ( 0.000%)
Frag6: 0 ( 0.000%)
ICMP6: 0 ( 0.000%)
UDP6: 0 ( 0.000%)
TCP6: 0 ( 0.000%)
Teardo: 0 ( 0.000%)
ICMP-IP: 0 ( 0.000%)
IP4/IP4: 0 ( 0.000%)
IP4/IP6: 0 ( 0.000%)
IP6/IP4: 0 ( 0.000%)
IP6/IP6: 0 ( 0.000%)
GRE: 0 ( 0.000%)
```

i-09547fbfe14b531c2 (Sentinal_X)
PublicIPs: 51.20.67.74 PrivateIPs: 172.31.26.135

```
TCP: 338 ( 98.836%)
IP6: 0 ( 0.000%)
IP6 Ext: 0 ( 0.000%)
IP6 Opt: 0 ( 0.000%)
Frag: 0 ( 0.000%)
ICMP6: 0 ( 0.000%)
UDP6: 0 ( 0.000%)
TCP6: 0 ( 0.000%)
Teardo: 0 ( 0.000%)
ICMP-IP: 0 ( 0.000%)
IP4/IP4: 0 ( 0.000%)
IP4/IP6: 0 ( 0.000%)
IP6/IP4: 0 ( 0.000%)
IP6/IP6: 0 ( 0.000%)
GRE: 0 ( 0.000%)
GRE Eth: 0 ( 0.000%)
GRE VLAN: 0 ( 0.000%)
GRE IP4: 0 ( 0.000%)
GRE IP6: 0 ( 0.000%)
GRE IP6 Ext: 0 ( 0.000%)
GRE PPTP: 0 ( 0.000%)
GRE SARP: 0 ( 0.000%)
GRE IPX: 0 ( 0.000%)
GRE Loop: 0 ( 0.000%)
GRE LSP: 0 ( 0.000%)
ARP: 0 ( 0.000%)
IPX: 0 ( 0.000%)
Eth Loop: 0 ( 0.000%)
Eth Dlac: 0 ( 0.000%)
IP4 Dlac: 1 ( 0.292%)
IP6 Dlac: 0 ( 0.000%)
TCP Dlac: 0 ( 0.000%)
UDP Dlac: 0 ( 0.000%)
ICMP Dlac: 0 ( 0.000%)
All Discard: 1 ( 0.292%)
Other: 0 ( 0.000%)
Bad Chk Sum: 183 ( 53.353%)
Bad CRC: 0 ( 0.000%)
SS G 1: 0 ( 0.000%)
SS G 2: 0 ( 0.000%)
Total: 343

Snort exiting
```

i-09547fbfe14b531c2 (Sentinal_X)
PublicIPs: 51.20.67.74 PrivateIPs: 172.31.26.135

Chapter 6

6.1 Research findings

6.2 Result Analysis and evaluation metrics

Conclusion & Future Work

In conclusion, the Image Forgery Detection project presents a comprehensive and adaptive solution to the escalating challenges posed by sophisticated image forgeries in today's digitally driven society. The multi-model approach, coupled with specialized machine learning models for distinct forgery types, ensures a robust and accurate detection framework. By training these models on diverse datasets that emulate real-world scenarios, the system gains a deep understanding of intricate patterns associated

with various manipulations. The emphasis on adaptability and resilience against evolving forgery techniques further enhances the system's effectiveness. The incorporation of ensemble learning techniques consolidates the strengths of individual models, providing a high level of detection accuracy and reliability. The project's scope covers a wide range of forgery types, including Deepfakes, Face Swapping, Object Insertion, Photo Manipulation, Style Transfer, Generative Adversarial Network Manipulation, Text Addition / Removal, Morphing, Image Splicing, Copy Move Forgery and the experimental validation demonstrates superior performance. Ultimately, this project contributes significantly to countering the proliferation of deceptive visual content, offering a promising solution to safeguard the integrity of digital media in our technology-driven era.

Future scope of the work for improvement may also be included

Appendices are provided to give supplementary information, which is not included in the main text may serve as a separate part contributing to main theme.

- Appendices should be numbered using Arabic numerals, e.g. Appendix 1, Appendix 2 etc.
- Appendices, tables and references appearing in appendices should be numbered and referred to at appropriate places just as in the case of chapters.
- Appendices shall carry the title of the work reported in it and the same title shall be used in the contents page also.

Appendices A

Items listed below are for example only	Total Cost	In kind or Match (what you already have)	Requested
Facility Expenses			
Utilities			
maintenance (cleaning)			
Internet connection			
Total Facility Expenses			
Supplies			
office supplies (provide details such as \$20 per month x 12 months)			
Workbooks			
arts & crafts supplies			
Software			
classroom supplies (for students and teachers)			

Total Supplies			
Equipment			
desktop computers			
laptop computers			
Printer			
Scanner			
Chairs			
digital camera			
Total Equipment			
Contractual			
Outside evaluator for program			
Experts we hire to come train our personnel			
Total Contractual			
Communications			
Telephone			
long distance			
cellular phones			
Postage			

Internet			
Total Communications			
Other Expenses			
Total Non-Personnel Expenses			
Total Direct Costs (personnel and non-personnel expenses)			
Total Project Costs/Total Request			

•

REFERENCES

1. WeiZhe Zhang , Bin ZhangYing Zhou , Hui HeZeyu Ding, “An IOT Honeynet Basedon multiport Honeypots for capturing IOT Attacks”. [IEEE Internet of Things Journal](#) (Volume: 7, [Issue: 5](#), May 2020). **Date of Publication:** 27 November 2019, DOI: [10.1109/JIOT.2019.2956173](#)
2. Yao Xu , Hiroshi Koide Danillo Vasconcellos , Kouichi Sakurai. “Tracing MIRAIMalware in Networked System”. **Published in:** [2018 Sixth International Symposium on Computing and Networking Workshops \(CANDARW\)](#).- **DOI:** [10.1109/CANDARW.2018.00104](#)
3. Sibi Chakkaravarthy , D.Sangeetha , Meenalosini Vimal Cruz, V.Vaidehi Balasubramanian Raman. “Design of Intrusion Detection Honeypot Using Social Leopard Algorithm to Detect IOT Ransomware Attacks”. **Date of Publication:** 14September 2020. DOI: [10.1109/ACCESS.2020.30237644](#)
4. Sebastian Poeplau Jan Gassen, “A Honypot for arbitrary malware on USB Storagedevices”. **Published in:** [2012 7th International Conference on Risks and Security of Internet and Systems \(CRiSIS\)](#). DOI : [10.1109/CRISIS.2012.6378948](#)

5. Seyed Mohammad Mousavi , Marc St-Hilarie. “Early Detection of DDoS attacks against SDN controllers”.
Published in: [2015 International Conference on Computing, Networking and Communications \(ICNC\)](#).
DOI: [10.1109/ICCNC.2015.7069319](#)