  
UNIVERSITY OF HERTFORDSHIRE  
School of Physics, Engineering and Computer Science

MSc Title: MSc Cyber Security  
Module Code and Title

7COM1039 Advanced Computer Science Masters Project  
  
Date:13-07-2025

PROJECT TITLE

Intrusion Detection System Using SNORT

Name: Sandeep Kumar  
Student ID: 23070471  
Supervisor: Vito Graffagnino

Contents

[1. Introduction 3](#_Toc203005116)

[1.1 Problem Statement 3](#_Toc203005117)

[1.2 Research Question 3](#_Toc203005118)

[1.3 Statement of Objectives 3](#_Toc203005119)

[2. Progress To Date 4](#_Toc203005120)

[2.1 Literature Review 4](#_Toc203005121)

[2.2 Proposed Methodology of Research 6](#_Toc203005122)

[2.3 Data Collection 8](#_Toc203005123)

[2.4 Data Preprocessing 9](#_Toc203005124)

[2.5 Supervised Learning Models and Challenges Faced 9](#_Toc203005125)

[3. Planned Work 10](#_Toc203005126)

[3.1 Thesis Timeline 10](#_Toc203005127)

[3.2 Training and Evaluation 11](#_Toc203005128)

[3.3 Deployment and Testing 11](#_Toc203005129)

[3.4 Future Enhancements 11](#_Toc203005130)

[4. Conclusion 12](#_Toc203005131)

[References 12](#_Toc203005132)

# 1. Introduction

The modern world of digital interaction is characterized by unprecedented levels of frequency and aptitude of cyber threats. The organizations and institutions which strongly depend on information technology fall as the first class victims to these dynamic cyber attacks. The reaction to the same is the use of procurrent security measures like the Intrusion Detection Systems (IDS). This initiative looks into the implementation of Snort open-source intrusion detection system to help notify and avert illegal activities taking place within a network within a virtualized scenario. The interim report is used to report on the progress of work, the strategies followed, issues that have been faced and the plan of the final implementation.

Snort, designed by Martin Roesch and supported by Cisco today, is an IDS (Beale et al., 2007) that is rich in features and quite versatile: it is able to log packets in real-time, analyze traffic and attack signatures. Snort has the ability to log the packets and real time effects on it. Its rule-based detection engine provides a large customisation ability and it can be used in educational and professional scenarios and in experimentation. This project seeks to confirm the ability of Snort to detect in as many as possible ways of attacks, and in addition find out its willingness of being employed in real time systems.

## 1.1 Problem Statement

Organizations are becoming heavily dependent on information technology in executing their normal activities, that has made them major targets of cyber attacks. Although conventional security tools like firewalls and antivirus programs are important, they are not adequate to stop every type of attack. The Intrusion Detection Systems (IDS) are significant in identifying the threats that evade other security solutions. Snort is a popular IDS because it is flexible and capable of detection based on the rules. Nevertheless, Snort has been critiqued as to its success in real-world high-throughput-networking environments, especially its capability at identifying encrypted traffic or obfuscated traffic.

## 1.2 Research Question

The primary research question addressed by this project is:

**"How effective is Snort as an intrusion detection system in a virtualized network environment, and what improvements can be made to enhance its performance in detecting novel and encrypted attacks?"**

This question will guide the exploration of Snort’s capabilities, focusing on its detection rates, false positive/negative rates, and overall system resource utilization during different types of attack simulations.

## 1.3 Statement of Objectives

This project will essentially focus on the designing, implementation as well as assessment of a powerful intrusion detection system in Snort. The system will be tested in an isolated simulation of a real environment of cyberattacks. These are particular goals:

* To carry out an in-depth literature review on the IDS technologies, especially Snort.
* To carry on the process of installing and configuring the Snort on Linux type of system with the right parameters of the network.
* In order to crate and to set up custom Snort rules to identify different kinds of attacks such as SQL injection, brute force, XSS, and port scanning.
* To emulate the network attacks of a controlled attacker machine using an instrument such as Nmap, SQLMap, Metasploit and Hydra.
* To gather, capture and decode alert information emanating out of Snort.
* To compare the performance of the IDS according to detection rate, false positives/negatives, along with utilization of system resources.

# 2. Progress To Date

## 2.1 Literature Review

IDS has been embedded in the common defense mechanisms of a network since time immemorial and has played vital roles in furnishing the much needed information on malicious activities and unauthorized access to a network. The current solutions used to protect against infections and attacks using IDS are usually divided into two major categories Host-Based Intrusion Detection Systems (HIDS) and Network-Based Intrusion Detection Systems (NIDS) that have developed over the last 20 years. Whereas Host-Based systems search individual hosts or devices to detect intrusion, Network-Based systems like the Snort detect possible threats by searching the network traffic (Scarfone & Mell, 2007; Garcia-Teodoro et al., 2009).

**Snort: IDS modular network-based**

Snort is a well-known network intrusion detection and prevention system within an open-source network-based IDS tool used to stay safe. Snort uses rule-based detection engine, where both the packet header and payload are used to determine patterns pointing to malicious activity. It is this signature-based detection which forms the base functionality of Snort with pre set rules constantly updated by user community to match known attack signatures. But the strength of Snort is that it is flexible by its modular design having the packet decoder, preprocessors, detection engine, logging facility and output module. This modularity supports massive customization, which gives its users the option to develop specialist rules to suit the specific nature of their networks and threats experienced.

One of the strongest characteristics of Snort is the possibility to set the detection capabilities and fine-tune the revolution tool to various environments adopted by organizations. As an example, we can use distinct preprocessor modules to support the usage of different protocols including HTTP, FTP and DNS, in this way it will make sure that Snort can be able to inspect traffic on a number of network protocols. Also, Snort uses a logging facility that will generate a lot of information on the attack, and its out put modules can be configured to provide a way of informing or reporting on the out come like alerting the security departments as the threats are detected in time.

**Effectiveness and Challenges of Snort**

Snort signature based detection model is very effective in detecting known attack signatures. According to the research conducted by Ahmed and Habib (2021) and Ali and Al-Yasiri (2020), Snort can be used to protect against known exploits and attacks since the program is efficient in identifying existing patterns of attacks. This trustworthiness has made Snort to be a choice of convenience among various organizations that are in need to use an IDS that is very reliable.

Nevertheless, similar to any other signature-based IDS, Snort suffers shortcomings when it comes to identifying something new or zero-day attacks that do not have known signatures. According to Sommer and Paxson (2010) and Zhang and Lee (2000), protection against the encrypted or obfuscated attacks is one of the major drawbacks of Snort. This decreases the effectiveness of Snort to examine the payload of such traffic, and in a world where more and more communications are being secured (HTTPS, SSL/TLS, and/or VPN), the effectiveness of Snort is reduced.

To meet that challenge, the research and practice have proposed to integrate Snort with the algorithms of anomaly detection which are not based on the premade signatures but rather work to sense the abnormalities in the system and network behavior. This set up contributes greatly to the detection abilities of Snort as it now has the capacity to detect the attack methods that may not have been known or elusive in the past. What is more is the possibility of incorporating Snort into Security Information and Event Management (SIEM) systems so as to facilitate centralized logging and more advanced analysis of these Snort alerts and hence the overall effectiveness of analyzing the threats (Sommer & Paxson, 2010; Zhang & Lee, 2000).

**Advancements and Alternatives to Snort**

Different graphical interfaces have been evolved over the decades to provide Snort alert in a more user-friendly format. Snort analyses tools like BASE (Basic Analysis and Security Engine) and Snorby also offer user friendly interfaces to view, search and analyze Snort alerts as well. Such interfaces enhance usage of Snort, particularly in big scale implementation by facilitating the collection and display of alerts in order to recognize possible danger easily.

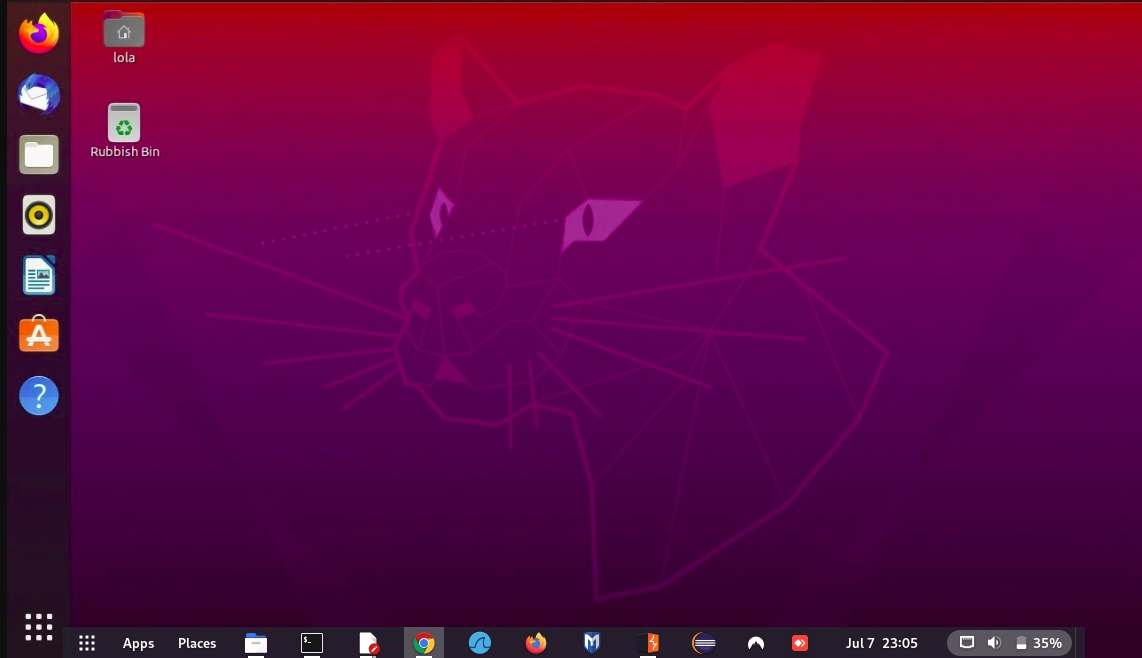
Besides the recent gains in improvements in the interface and functionality of Snort, recent literature has discussed the comparisons between Snort and other solutions of IDS including Suricata and Zeek (formerly Bro). Although Snort is still very popular having the advantages of a simple installation process, extensive documentation and an active user community, others such as Suricata have started providing features like multi-threading which makes all the difference in high throughput systems. The ability of Suricata to handle a large amount of data at the same time and the advantage that Snort has over Suricata is its performance time and scales very well in network settings issues. Nonetheless, the persistence of Snort has been supported by its unswerving ease of use and support offered by thousands of admirers as well as its proficiency (Kumar & Singh, 2022).

## 2.2 Proposed Methodology of Research

This project adopted a **design science research methodology**, focusing on the development, testing, and evaluation of Snort in a controlled lab environment. The project is split into several key stages:

* **Setup of the IDS environment** using three virtual machines (VMs): one for Snort, one for the attacker (Kali Linux), and one for the target (Windows or Ubuntu).

IDS: Hosted by Snort Machine (Ubuntu 20.04).

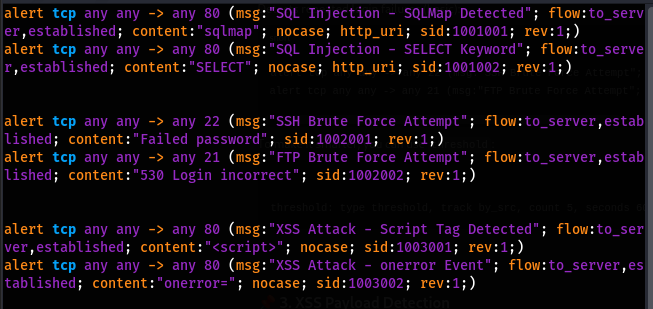


* Attacker Machine (Kali Linux): Runs a number of cyberattacks.



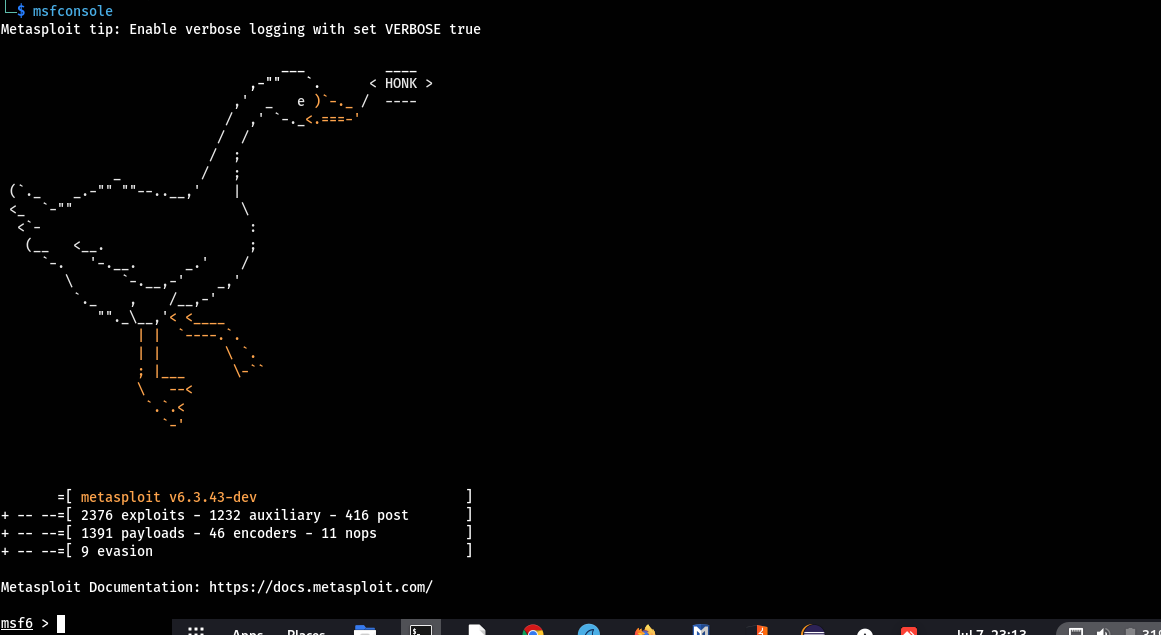
**Wiring up** the Snort configuration files (snort.conf, local.rules) to ensure proper packet capture and rule activation.

* **Development of custom Snort rules** to detect specific attacks such as SQL injection, brute-force login attempts, and port scanning.



Every rule is checked in terms of syntax and usefulness. SID and revision numbering is done and test attacks are used to test alert generation.

* **Simulation of network attacks** using tools like Nmap, SQLMap, and Hydra to test Snort’s detection capabilities.



All the sessions of the attacks are recorded and monitored through output and packet capture on Snort and through Wireshark.

* **Collection of data** from Snort logs to analyze detection rates and false positives/negatives.

## 2.3 Data Collection

Data collection focused on capturing Snort alerts and logs during attack simulations. Logs were saved to the /var/log/snort/alert file and processed using Python scripts to extract useful data, such as timestamps, source/destination IPs, ports, and attack types. This data is essential for evaluating Snort’s performance in real-time attack scenarios and determining areas for improvement.

| **Data Element** | **Description** | **File Location** | **Purpose** |
| --- | --- | --- | --- |
| **Timestamps** | The time at which each alert was generated | /var/log/snort/alert | To track when an attack occurred and correlate it with other data |
| **Source IP** | The IP address from which the attack originated | /var/log/snort/alert | To identify the attacking machine or source of malicious activity |
| **Destination IP** | The IP address targeted by the attack | /var/log/snort/alert | To determine which machine or system was targeted in the attack |
| **Ports** | The ports involved in the attack (source/destination) | /var/log/snort/alert | To understand which services or protocols were targeted |
| **Attack Types** | The type of attack detected (e.g., SQL injection, port scan, etc.) | /var/log/snort/alert | To classify the attack and identify recurring patterns or threats |
| **Alert Message** | The detailed alert message generated by Snort | /var/log/snort/alert | To provide context and explanation of the detected attack |

## 2.4 Data Preprocessing

After data collection, logs were cleaned and organized to ensure the removal of noise from irrelevant alerts. Alerts were categorized based on attack types, and statistical analysis was performed to calculate the detection rate, false positive rate, and false negative rate. Preprocessing also involved the analysis of system resource usage during Snort’s operation, which provided insights into the resource efficiency of the system under different load conditions.

## 2.5 Supervised Learning Models

The even though Snort is mainly capable of detection as facilitated by the use of rule-based detection elements, it will additionally be improved in future whereby anomaly detection models will be incorporated. This will deal with the training of supervised learning models based on training data to achieve a classification then train the model to classify network traffic, benign or malicious. The decision tree and support machine techniques will be investigated to identify attack patterns that have not occurred before in order to make Snort step up its detection activities.

# Challenges Faced

Some of the issues have been presented at the stage of implementation:

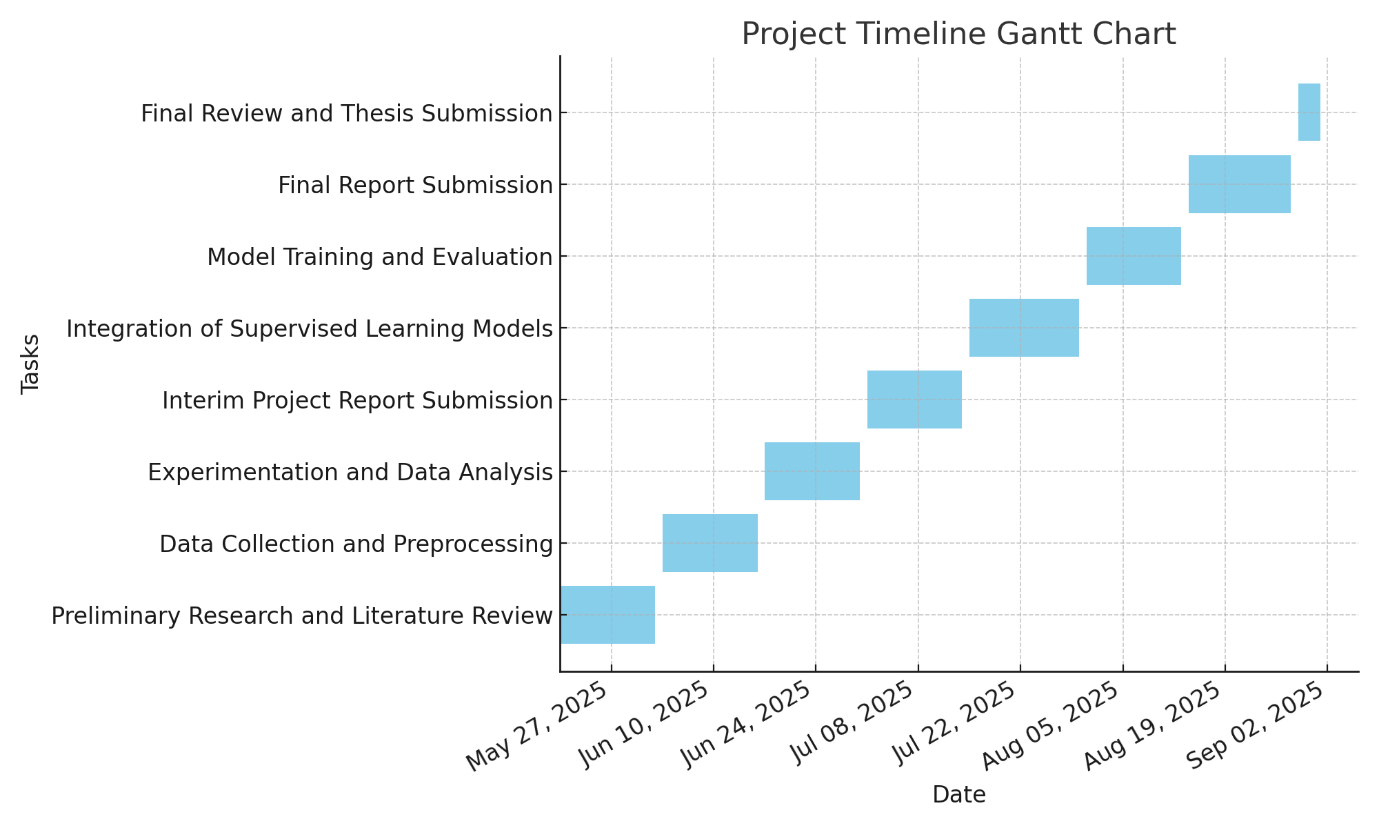
* To setup Snort, there was the need to resolve dependency problems and configure various configuration files.
* Custom rule development demanded trial and error to reduce false positives (Ali and Al-Yasiri, 2020; Axelsson, 2000).
* Detection of encrypted traffic (HTTPS) remains a limitation (Iqbal et al., 2019) without implementing SSL inspection.
* Snort is single-threaded and its performance suffers a little when subjected to high traffic loads.
* Visualization of Snorby has to be set up and troubleshooting is needed to have GUI appearance.

# 3. Planned Work

## 3.1 Thesis Timeline

The phases of the project will follow the timeline outlined below:

| **Task No.** | **Task Name** | **Start Date** | **Expected End Date** | **Duration (Weeks)** |
| --- | --- | --- | --- | --- |
| 1 | Preliminary Research and Detailed Project Plan | May 20, 2025 | June 2, 2025 | 2 |
| 2 | Literature Review ,Data Collection and Preprocessing | June 3, 2025 | June 16, 2025 | 2 |
| 3 | Experimentation and Data Analysis | June 17, 2025 | June 30, 2025 | 2 |
| 4 | Interim Project Report Submission | July 1, 2025 | July 14, 2025 | 2 |
| 5 | Integration of Supervised Learning Models | July 15, 2025 | July 30, 2025 | 2 |
| 6 | Model Training and Evaluation | July 31, 2025 | Aug 13, 2025 | 2 |
| 7 | Final Report Preparation | Aug 14, 2025 | Aug 28, 2025 | 2 |
| 8 | Final Review and Thesis Submission | Aug 29, 2025 | Sept 8 2025 | 1.5 |



## 3.2 Training and Evaluation

Training supervised learning models will require careful feature extraction from network traffic data. Once trained, these models will be evaluated for accuracy using cross-validation techniques and compared against Snort’s rule-based detection.

## 3.3 Deployment and Testing

After integrating the anomaly detection models, Snort will be redeployed in the lab environment to test its combined detection approach. The system will undergo rigorous testing against a variety of attack types, including encrypted and zero-day attacks.

## 3.4 Future Enhancements

Future enhancements will include:

* **Integration with SIEM systems**: To enable centralized logging and more advanced analysis.
* **Rule optimization**: To reduce false positives and enhance detection rates.
* **Machine learning-based rule creation**: To automate the generation of new Snort rules.

# 4. Conclusion

This project has been able to come up with the foundation on assessing Snort as an IDS in a virtualized network system. The early experiments demonstrated the effectiveness of Snort in identifying known attacks and identified the weakness of Snort when processing encrypted traffic. The next steps will be orientation towards integrating the machine learning models to improve the detection capabilities of Snort in a way that it will be able to better mitigate novel and encrypted attacks. The study will also yield useful information on how Snort can be used in practice in terms of cybersecurity implications.

# References

1. Scarfone, K., & Mell, P. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS)*. NIST Special Publication 800-94. National Institute of Standards and Technology.
2. Beale, J., Baker, A., Posluns, J., Northcutt, S., & Poor, M. (2007). *Snort Intrusion Detection and Prevention Toolkit*. Syngress Publishing.
3. Sommer, R., & Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *2010 IEEE Symposium on Security and Privacy* (pp. 305–316). IEEE.
4. Kumar, A., & Singh, H. (2022). Comparative Study of Snort, Suricata and Zeek Intrusion Detection Systems. *International Journal of Network Security & Its Applications*, 14(1).
5. Ahmed, H., & Habib, A. (2021). Lightweight IDS for IoT Networks Using Snort. *Procedia Computer Science*, 184, 674–681.
6. Ali, M., & Al-Yasiri, A. (2020). Custom Rule Optimization for Snort to Reduce False Positives. *Journal of Cyber Security Technology*, 4(3), 145–160.
7. Garcia-Teodoro, A., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based Network Intrusion Detection: Techniques, Systems and Challenges. *Computers & Security*, 28(1-2), 18–28.
8. Iqbal, M., Shah, M. A., & Khan, M. A. (2019). Performance Evaluation of Open Source Intrusion Detection Systems on Encrypted Network Traffic. *IEEE Access*, 7, 115104–115116.
9. Axelsson, S. (2000). The Base-Rate Fallacy and the Difficulty of Intrusion Detection. *ACM Transactions on Information and System Security*, 3(3), 186–205.
10. Zhang, Y., & Lee, W. (2000). Intrusion Detection in Wireless Ad-Hoc Networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking* (pp. 275–283). ACM.