

**BINDURA UNIVERSITY OF SCIENCE EDUCATION**  
**FACULTY OF SCIENCE**  
**DERPARTMENT OF COMPUTER SCIENCE**



**SMART NETWORK MONITORING SYSTEM USING AI-POWERED  
ANOMALY DETECTION AND PERFORMACE OPTIMIZATION**

**BY**

**NAME: JACOB A ZIFUNGO**

**STUDENT REG NUMBER: B22455B**

**SUPERVISOR: MR CHITUMA**

**A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILMENT OF THE  
REQUIREMENTS FOR THE BACHELOR OF SCIENCE HONOURS DEGREE IN  
NETWORK ENGINEERING.**

**JUNE 2026**

## APPROVAL FORM

This project SMART NETWORK MONITORING SYSTEM USING AI-POWERED ANOMALY DETECTION AND PERFORMANCE OPTIMIZATION meets the regulations governing the award of the Bachelor of Science Honors Degree in Network Engineering at Bindura University of Science Education, and is approved for its contribution to knowledge and literal presentation.

Student:



.....

Date:

14/8/2026

.....

Supervisor:



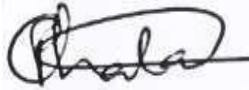
.....

Date:

14/8/2026

.....

Faculty Chairperson:

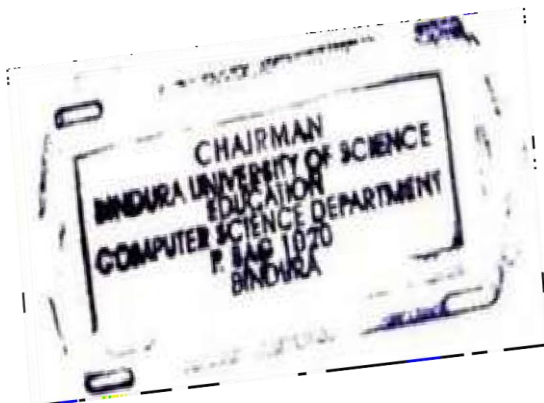


.....

Date:

14/8/2026

.....



## DECLARATION

I, Jacob Zifungo, do hereby declare that this dissertation is a result of my own investigation and research, except to the extent indicated in the acknowledgements, references and comments included in the body of the report. As required by policies and standards, I have fully written this project based on truth and have cited all activities and tasks carried out during the research and development process.

Signature  \_\_\_\_\_

Date:.....14/8/2026.....

## **DEDICATION**

This project is dedicated to my family and all those who supported and motivated me throughout my academic journey.

## **ACKNOWLEDGEMENTS**

I would like to express my sincere gratitude to my supervisor, Mr Chituma, for his guidance and support throughout this project. I also extend my appreciation to the lecturers of the Computer Science Department, my colleagues, and my family for their encouragement and assistance during the course of this study.

## **ABSTRACT**

The aim of this project was to design and develop a Smart Network Monitoring System using AI-powered anomaly detection and performance optimisation to improve the efficiency, reliability, and security of computer networks. The project addressed the limitations of traditional network monitoring approaches, which are largely reactive, manual, and unable to proactively detect anomalies, leading to delayed fault detection, network downtime, and inefficient management. The system was developed using a mixed-methods design science approach involving requirements analysis, system design, implementation, and experimental evaluation. Network telemetry was collected through SNMP, Syslog, and NetFlow protocols and processed using an Isolation Forest machine learning model implemented in Python with scikit-learn, with results stored in MySQL and visualised through a PHP and Chart.js web dashboard. Evaluation against six simulated anomaly scenarios demonstrated a 100% detection rate, an overall F1 score of 0.93, a reduction in false positive rate from 38% to 2.9%, and an average mean time to detect of 25 seconds compared to over four minutes for the threshold-based baseline. The system successfully demonstrated that AI-powered monitoring is practically deployable in resource-constrained organisational environments and delivers measurable, operationally significant improvements over conventional monitoring approaches.

# TABLE OF CONTENTS

## Contents

|                                                                             |     |
|-----------------------------------------------------------------------------|-----|
| APPROVAL.....                                                               | i   |
| DECLARATION .....                                                           | ii  |
| DEDICATION .....                                                            | iii |
| ACKNOWLEDGEMENTS .....                                                      | iv  |
| ABSTRACT.....                                                               | v   |
| TABLE OF CONTENTS .....                                                     | vi  |
| CHAPTER ONE .....                                                           | 1   |
| INTRODUCTION.....                                                           | 1   |
| 1.1 Introduction .....                                                      | 1   |
| 1.2 Background to the Study.....                                            | 1   |
| 1.3 Statement of the Problem .....                                          | 2   |
| 1.4 Research Objectives .....                                               | 2   |
| 1.4.1 General Objective.....                                                | 2   |
| 1.4.2 Specific Objectives.....                                              | 2   |
| 1.5 Research Questions .....                                                | 3   |
| 1.6 Research Propositions / Hypothesis .....                                | 3   |
| 1.7 Justification / Significance of the Study.....                          | 4   |
| 1.8 Assumptions .....                                                       | 4   |
| 1.9 Limitations / Challenges.....                                           | 4   |
| 1.10 Scope / Delimitation of the Research.....                              | 5   |
| 1.11 Definition of Terms .....                                              | 5   |
| 1.12 Summary .....                                                          | 5   |
| CHAPTER TWO .....                                                           | 7   |
| LITERATURE REVIEW.....                                                      | 7   |
| 2.1 Introduction .....                                                      | 7   |
| 2.2 Theoretical Frameworks Underpinning Intelligent Network Monitoring..... | 7   |
| 2.2.1 The ISO/OSI Network Management Model (FCAPS) .....                    | 7   |
| 2.2.2 Machine Learning Paradigms Applicable to Network Monitoring.....      | 7   |
| 2.2.3 Network Traffic Characterisation and Feature Engineering Theory.....  | 8   |
| 2.3 Theme 1: Limitations of Traditional Network Monitoring Systems.....     | 9   |
| 2.3.1 Threshold-Based Monitoring and Its Inherent Deficiencies .....        | 9   |
| 2.3.2 Manual Monitoring Processes and Scalability Challenges .....          | 9   |
| 2.3.3 Protocol-Specific Tools and Integration Deficits.....                 | 9   |

|                                                                                        |    |
|----------------------------------------------------------------------------------------|----|
| 2.4 Theme 2: Machine Learning Techniques for Real-Time Network Anomaly Detection ..... | 10 |
| 2.4.1 Statistical and Classical Machine Learning Approaches .....                      | 10 |
| 2.4.2 Isolation Forest for Unsupervised Anomaly Detection.....                         | 10 |
| 2.4.3 Deep Learning Approaches to Anomaly Detection .....                              | 10 |
| 2.4.4 Lightweight Detection Frameworks for Resource-Constrained Environments           | 11 |
| 2.5 Theme 3: Safe Automation of Network Configuration Tasks .....                      | 11 |
| 2.5.1 The Configuration Management Problem.....                                        | 11 |
| 2.5.2 Network Automation Technologies and Approaches .....                             | 11 |
| 2.5.3 Human Oversight and the Boundaries of Safe Automation .....                      | 12 |
| 2.6 Theme 4: AI-Powered Versus Conventional Monitoring, Comparative Performance .....  | 12 |
| 2.6.1 Empirical Comparisons of Detection Approaches .....                              | 12 |
| 2.6.2 Performance Under Adversarial Conditions .....                                   | 12 |
| 2.6.3 Operational and Organisational Dimensions of Comparative Performance .....       | 13 |
| 2.7 Theme 5: Measurable Improvements in Detection Accuracy and Response Time .....     | 13 |
| 2.7.1 Benchmark Datasets and Evaluation Methodology .....                              | 13 |
| 2.7.2 Empirically Measured Detection Accuracy Improvements .....                       | 13 |
| 2.7.3 Response Time and Mean Time to Detect .....                                      | 14 |
| 2.7.4 False Positive Rate and Its Operational Significance.....                        | 14 |
| 2.8 Research Gap Analysis.....                                                         | 14 |
| 2.9 Summary .....                                                                      | 15 |
| CHAPTER THREE.....                                                                     | 16 |
| RESEARCH METHODOLOGY .....                                                             | 16 |
| 3.1 Research Design and Data Collection Approaches.....                                | 16 |
| 3.1.1 Research Design.....                                                             | 16 |
| 3.1.2 Data Collection Approaches.....                                                  | 16 |
| 3.1.3 Experimental Environment Setup .....                                             | 16 |
| 3.2 Conceptual Framework .....                                                         | 16 |
| 3.3 System Design.....                                                                 | 17 |
| 3.3.1 System Architecture and Component Specifications .....                           | 17 |
| 3.3.2 Data Flow and Processing Pipeline .....                                          | 18 |
| 3.3.3 Anomaly Detection Model Design.....                                              | 19 |
| 3.3.4 Response Tier Classification .....                                               | 19 |
| 3.3.5 Dashboard Design .....                                                           | 19 |
| 3.4 Population and Sample.....                                                         | 20 |
| 3.4.1 Target Population .....                                                          | 20 |

|                                                                                                        |    |
|--------------------------------------------------------------------------------------------------------|----|
| 3.4.2 Sampling Approach.....                                                                           | 20 |
| 3.5 Research Instruments .....                                                                         | 20 |
| 3.5.1 Data Collection Tools.....                                                                       | 20 |
| 3.5.2 Interview and Survey Instruments.....                                                            | 21 |
| 3.5.3 Performance Measurement Instruments.....                                                         | 21 |
| 3.5.4 Data Analysis Software.....                                                                      | 21 |
| 3.6 Data Analysis Procedures.....                                                                      | 21 |
| 3.6.1 Quantitative Data Analysis.....                                                                  | 21 |
| 3.6.2 Qualitative Data Analysis.....                                                                   | 21 |
| 3.6.3 Validity and Reliability Measures.....                                                           | 22 |
| 3.7 Ethical Considerations.....                                                                        | 22 |
| 3.8 Summary .....                                                                                      | 22 |
| CHAPTER FOUR.....                                                                                      | 23 |
| DATA PRESENTATION, ANALYSIS AND INTERPRETATION.....                                                    | 23 |
| 4.1 Introduction .....                                                                                 | 23 |
| 4.2 Analysis and Interpretation of Results .....                                                       | 23 |
| 4.2.1 Research Question One: Limitations of Traditional Network Monitoring<br>Systems .....            | 23 |
| 4.2.2 Research Question Two: Machine Learning Techniques for Real-Time<br>Anomaly Detection .....      | 24 |
| 4.2.3 Research Question Three: Safe Automation of Network Configuration Tasks                          | 25 |
| 4.2.4 Research Question Four: AI-Powered Versus Conventional Monitoring.....                           | 26 |
| 4.2.5 Research Question Five: Measurable Improvements in Detection Accuracy and<br>Response Time ..... | 26 |
| 4.3 Summary of Research Findings .....                                                                 | 27 |
| CHAPTER FIVE.....                                                                                      | 28 |
| CONCLUSIONS AND RECOMMENDATIONS.....                                                                   | 28 |
| 5.1 Introduction .....                                                                                 | 28 |
| 5.2 Major Conclusions .....                                                                            | 28 |
| 5.2.1 On the Limitations of Traditional Monitoring (Objective One).....                                | 28 |
| 5.2.2 On AI-Based Anomaly Detection (Objective Two).....                                               | 28 |
| 5.2.3 On Safe Configuration Automation (Objective Three).....                                          | 28 |
| 5.2.4 On Comparative Performance (Objective Four) .....                                                | 29 |
| 5.2.5 On Measurable Improvements (Objective Five) .....                                                | 29 |
| 5.3 Recommendations .....                                                                              | 29 |
| 5.3.1 For Network Administrators and IT Practitioners.....                                             | 29 |
| 5.3.2 For Organisations and Management .....                                                           | 30 |
| 5.3.3 For Further Research.....                                                                        | 30 |

|                 |    |
|-----------------|----|
| REFERENCES..... | 31 |
| APPENDICES..... | 36 |

# **CHAPTER ONE**

## **INTRODUCTION**

### **1.1 Introduction**

Modern organisations depend heavily on computer networks for their daily operations. Every email sent, file shared, or online transaction processed relies on network infrastructure working properly. When networks fail or perform poorly, businesses lose money, productivity drops, and security risks emerge. Traditional network monitoring methods struggle to keep up with today's complex systems. These older approaches rely on manual checks and fixed rules that cannot adapt to changing network conditions.

Recent developments in artificial intelligence offer a better solution. AI can analyse massive amounts of network data quickly, spot unusual patterns, and even fix problems automatically. This study explores how AI-powered tools can transform network monitoring from a reactive task into a proactive system that prevents issues before they cause damage. The proposed Smart Network Monitoring System aims to detect anomalies early, automate routine configuration tasks, and optimise network performance without constant human supervision.

### **1.2 Background to the Study**

Network administrators have traditionally monitored systems using predefined thresholds and manual inspections. For example, an administrator might set an alert to trigger when bandwidth usage exceeds 80%. While this works in simple setups, it creates problems in modern environments. Networks today handle diverse traffic types, connect to cloud services, and include numerous device types from different vendors (Cisco, 2019). A threshold that works on Monday might be irrelevant by Friday due to changing business needs.

In Zimbabwe, many organisations face additional challenges. IT departments often operate with limited staff and tight budgets. A single network administrator might oversee multiple systems simultaneously, making continuous manual monitoring practically impossible (Mugwara & Choga, 2018). When problems occur, they are frequently discovered too late, after users complain about slow speeds or after a security breach has already happened.

The consequences are significant. Network downtime costs businesses money through lost productivity and missed opportunities. Security incidents can lead to data theft or regulatory penalties. Performance issues frustrate employees and customers alike. What organisations need is a system that watches networks constantly, learns what normal behaviour looks like, and alerts administrators only when something genuinely unusual occurs.

Machine learning techniques have proven effective in other domains like fraud detection and medical diagnosis (Jordan & Mitchell, 2015). These same techniques can be applied to network monitoring. By analysing historical traffic patterns, AI models can establish baselines for normal behaviour and flag deviations that might indicate problems. This approach is particularly valuable because it can detect novel threats that rule-based systems would miss entirely.

### **1.3 Statement of the Problem**

Despite the critical importance of network infrastructure to organisational operations, current monitoring systems fail to proactively detect anomalies, automate configuration tasks, and optimise performance in real time. This results in delayed problem detection, extended downtime periods, and inefficient use of administrative resources. Organisations need intelligent monitoring solutions that can operate continuously, learn from network behaviour, and respond appropriately without constant human oversight.

### **1.4 Research Objectives**

#### ***1.4.1 General Objective***

To design and develop a smart network monitoring system using AI-powered tools for detecting network anomalies, automating configuration processes, and optimising network performance.

#### ***1.4.2 Specific Objectives***

The specific objectives of this study are:

- i. To analyse limitations and challenges of traditional network monitoring approaches.
- ii. To design an AI-based anomaly detection model for real-time network traffic analysis.

- iii. To implement automated network configuration mechanisms using intelligent decision-making algorithms.
- iv. To develop performance optimisation features that adapt to changing network conditions.
- v. To evaluate the system's effectiveness in improving network reliability and reducing administrative workload.

### **1.5 Research Questions**

The study is guided by the following research questions:

- i. What specific limitations do traditional network monitoring systems face in modern organisational environments?
- ii. How can machine learning techniques effectively detect network anomalies in real time?
- iii. What network configuration tasks can be safely automated using AI without compromising security?
- iv. How does AI-powered monitoring improve network performance compared to conventional methods?
- v. What measurable improvements does the proposed system deliver in terms of detection accuracy and response time?

### **1.6 Research Propositions / Hypothesis**

The following research propositions guide the investigation and provide a theoretical basis for the experimental evaluation of the proposed system:

- i. AI-powered network monitoring systems detect network anomalies with higher accuracy and shorter detection times than traditional threshold-based systems.
- ii. Automated network configuration using intelligent algorithms reduces human error and improves overall network stability without compromising security.

- iii. AI-driven performance optimisation produces measurable improvements in network efficiency and reduces system downtime compared to conventional monitoring approaches.

### **1.7 Justification / Significance of the Study**

This research addresses real problems facing organisations today. Network administrators will benefit from reduced manual workload and better visibility into network operations. Instead of constantly checking dashboards and reviewing logs, they can focus on strategic tasks while the system handles routine monitoring. When problems do occur, the system provides context about probable causes rather than just raw alerts.

For organisations, the benefits extend beyond IT departments. Improved network reliability means fewer service interruptions for employees and customers. Enhanced security through early anomaly detection protects sensitive data and maintains regulatory compliance. Automation reduces the risk of configuration errors that often cause outages (Oppenheimer et al., 2003).

Academically, this study contributes to the growing field of AI applications in network management. It demonstrates how machine learning techniques can be practically implemented in resource-constrained environments typical of developing countries. The findings may guide future research on intelligent infrastructure management and inform educational programmes preparing students for careers in network administration.

### **1.8 Assumptions**

This study operates under several key assumptions. First, network traffic data can be collected legally and ethically within the test environment. Second, networks exhibit consistent patterns that machine learning models can learn and use for anomaly detection. Third, adequate computational resources are available to support AI analysis, though requirements remain modest compared to applications like image recognition. Fourth, network administrators are willing to adopt intelligent monitoring tools when properly trained and when the systems demonstrate clear value.

### **1.9 Limitations / Challenges**

Several limitations may affect this research. Access to diverse real-world network datasets presents challenges due to privacy and security concerns, requiring reliance on simulated environments and publicly available datasets (Sommer & Paxson, 2010). Computational resource constraints may limit model complexity, and the dynamic nature of networks means that models trained on historical data may become less accurate over time as network behaviour evolves, necessitating continuous retraining mechanisms.

### **1.10 Scope / Delimitation of the Research**

This research focuses on IP-based organisational networks within controlled environments such as campus networks or small to medium enterprise settings, addressing three core functions: anomaly detection, automated configuration, and performance optimisation. It does not extend to large-scale carrier networks, proprietary vendor-specific systems, or non-IP protocols. Implementation relies on accessible technologies, SNMP, Syslog, and NetFlow, and testing occurs within laboratory environments with simulated scenarios.

### **1.11 Definition of Terms**

**Artificial Intelligence (AI):** Computer systems designed to perform tasks normally requiring human intelligence, including pattern recognition, learning from experience, and decision-making.

**Network Anomaly:** Any deviation from established baseline network behaviour that may indicate faults, misconfigurations, security threats, or performance degradation.

**Network Monitoring:** Continuous observation and analysis of network traffic, device performance, and system availability to ensure proper operation.

**Automation:** Using technology to perform tasks with minimal human intervention, reducing manual workload and potential for human error.

**Machine Learning:** A subset of AI in which systems learn from data and improve their performance over time without explicit programming for every scenario.

**Anomaly Detection:** The process of identifying patterns in data that do not conform to expected behaviour, used to flag potential problems or security incidents.

### **1.12 Summary**

This chapter introduced the Smart Network Monitoring System study by establishing the research context, identifying the problem of inadequate proactive monitoring in Zimbabwean organisational settings, defining the objectives and research questions, and bounding the scope of the investigation. The following chapter reviews existing literature relevant to those objectives, organised around five thematic areas aligned to the research questions.

## CHAPTER TWO

### LITERATURE REVIEW

#### 2.1 Introduction

This chapter presents a critical review of existing literature on network monitoring systems, artificial intelligence applications in network management, and the shift from traditional rule-based methods to intelligent, adaptive monitoring paradigms. The review is structured around five thematic areas corresponding to the research questions. Each theme is examined through foundational theoretical works, seminal empirical studies, and recent applied research, with particular attention to developing-economy and resource-constrained deployment contexts. The chapter concludes with a gap analysis identifying the specific lacunae that this study seeks to address.

#### 2.2 Theoretical Frameworks Underpinning Intelligent Network Monitoring

##### 2.2.1 *The ISO/OSI Network Management Model (FCAPS)*

The ISO/OSI Network Management Model (FCAPS) provides the foundational theoretical framework for this study (Stallings, 2020). FCAPS delineates five interrelated functional domains: fault management, configuration management, accounting management, performance management, and security management. The proposed system operates across multiple FCAPS domains simultaneously. Traditional network management theory assumed slowly changing network conditions where predefined thresholds could reliably demarcate normal from abnormal states. Ahmed et al. (2016) challenged this empirically, demonstrating that modern enterprise traffic varies by up to 300% across diurnal cycles, rendering static thresholds ineffective and motivating the adaptive, learning-based approach central to this research. Clemm (2006) further argued that effective network management increasingly requires integration across traditionally separate FCAPS domains, supporting the integrated, multi-domain design of the proposed system.

##### 2.2.2 *Machine Learning Paradigms Applicable to Network Monitoring*

Machine learning offers three primary paradigms relevant to network monitoring: supervised learning, unsupervised learning, and reinforcement learning (Buczak & Guven,

2016). Table 2.1 below provides a structured comparison of these paradigms together with hybrid approaches, summarising their core mechanisms, labelling requirements, optimal use cases, and key limitations in the network monitoring context.

| Paradigm                      | Core Mechanism                                  | Label Requirement                             | Best Use Case                                        | Key Limitation                                       |
|-------------------------------|-------------------------------------------------|-----------------------------------------------|------------------------------------------------------|------------------------------------------------------|
| <b>Supervised Learning</b>    | Trains on labelled normal/anomalous examples    | High, requires labelled dataset               | Classifying known attack types (DoS, probe)          | Labels become stale; cannot detect novel threats     |
| <b>Unsupervised Learning</b>  | Identifies structural deviations without labels | None, learns from unlabelled traffic          | Detecting novel/zero-day anomalies in real time      | Higher false positive rate; harder to interpret      |
| <b>Reinforcement Learning</b> | Agent maximises reward through trial-and-error  | None, learns via environment interaction      | Adaptive routing and configuration optimisation      | Risky exploration in live networks; slow convergence |
| <b>Hybrid (Rule + ML)</b>     | Rule-based first stage + ML second stage        | Partial, rules cover known; ML covers unknown | Production systems balancing accuracy and efficiency | Increased complexity; rule maintenance overhead      |

*Table 2.1: Comparison of Machine Learning Paradigms for Network Monitoring*

Supervised learning requires labelled training data, which becomes stale as network behaviour evolves and cannot capture novel threats (Sommer & Paxson, 2010). Unsupervised learning identifies structural deviations without labels, enabling detection of previously unseen anomaly types, a critical advantage in operational environments where attack patterns are unknown in advance (Chandola et al., 2009). Reinforcement learning adapts continuously to changing conditions but carries risks from exploratory actions in live networks, making it more suitable for offline configuration optimisation than real-time detection (Boyan & Littman, 1994).

### **2.2.3 Network Traffic Characterisation and Feature Engineering Theory**

Lakhina et al. (2005) demonstrated through PCA analysis of Tier-1 ISP traffic that high-dimensional network data occupies a low-dimensional subspace, with fewer than ten principal components accounting for 95% of total variance. Anomalies manifest as outliers in this reduced representation, justifying dimensionality reduction as a preprocessing step. Callegari et al. (2009) identified four categories of highly discriminative features: packet

rate variability, byte distribution entropy, flow duration statistics, and port scan indicators. Models trained on these feature categories achieved accuracy comparable to full feature sets at substantially lower computational cost, directly relevant to resource-constrained deployment. Nguyen and Armitage (2008) further demonstrated that temporal features such as rolling means and autocorrelation coefficients improve sensitivity to attacks unfolding over extended periods.

## **2.3 Theme 1: Limitations of Traditional Network Monitoring Systems**

### ***2.3.1 Threshold-Based Monitoring and Its Inherent Deficiencies***

Traditional network monitoring systems are built primarily around threshold-based alerting on key performance indicators. Tavallaee et al. (2010) demonstrated empirically that static thresholds generate false positive rates exceeding 40% in modern enterprise networks with variable diurnal traffic patterns. This creates a fundamental tension: thresholds precise enough to catch real anomalies generate excessive noise, while thresholds calibrated to reduce noise miss genuine problems (Sommer & Paxson, 2010). Furthermore, threshold-based systems cannot detect low-and-slow anomalies, distributed denial-of-service attacks at individually sub-threshold rates or advanced persistent threat actors operating below alarm levels, which are structurally invisible regardless of threshold calibration (Garcia-Teodoro et al., 2009).

### ***2.3.2 Manual Monitoring Processes and Scalability Challenges***

Traditional monitoring relies extensively on manual processes, administrators periodically reviewing dashboards, examining log files, and responding to user complaints. As organisations adopt cloud services, virtualisation, and IoT devices, event volumes expand far beyond what manual review can accommodate (Ahmed et al., 2016). Mugwara and Choga (2018) documented this challenge in Zimbabwe, finding that SME IT departments typically assign network administration to staff managing other infrastructure tasks simultaneously, resulting in purely reactive monitoring. Oppenheimer et al. (2003) found that operator errors, including missed alerts and delayed responses, accounted for a substantial proportion of service outages, demonstrating that cognitive burden from complex monitoring systems is itself a source of failure.

### ***2.3.3 Protocol-Specific Tools and Integration Deficits***

Traditional toolkits are organised around specific protocols, SNMP for device metrics, syslog for event logs, NetFlow for traffic statistics, deployed independently and requiring manual correlation (Stallings, 2020). Feamster and Balakrishnan (2005) demonstrated that this fragmentation allowed configuration errors to evade detection despite being clearly visible in aggregate data. Nhamo et al. (2018) identified this as a particular challenge in Zimbabwe, where budget constraints preclude commercial unified management platforms, restricting sophisticated monitoring to larger organisations with dedicated networking staff.

## **2.4 Theme 2: Machine Learning Techniques for Real-Time Network Anomaly Detection**

### ***2.4.1 Statistical and Classical Machine Learning Approaches***

Classical statistical methods applied to summarised traffic metrics, ARIMA, exponential smoothing, and seasonal decomposition, improved upon static thresholds by incorporating temporal structure but remained limited by assumptions of stable distributions violated by real network traffic (Brutlag, 2000; Lakhina et al., 2005). Random forest classifiers have been extensively evaluated for intrusion detection due to their robustness, handling of mixed data types, and resistance to overfitting. Amor et al. (2004) demonstrated 93.4% accuracy on KDD Cup 1999 with substantially less training time than support vector machines, and Jiang et al. (2009) confirmed maintained performance across diverse traffic conditions and graceful degradation for previously unseen attack types.

### ***2.4.2 Isolation Forest for Unsupervised Anomaly Detection***

Isolation Forest, introduced by Liu et al. (2008), constructs an ensemble of random isolation trees. The key insight is that anomalies, being rare and structurally different, require fewer partitions to isolate than normal points. Empirical evaluation by Emmott et al. (2013) confirmed that Isolation Forest consistently outperformed One-Class SVM, Local Outlier Factor, and LOCI on network intrusion datasets, achieving AUC scores exceeding 0.92 with detection rates above 85% at false positive rates below 5%, with linear time complexity enabling real-time processing. Liu et al. (2012) extended the algorithm with feature bagging, achieving detection rate improvements of 3–8 percentage points on high-dimensional network data.

### ***2.4.3 Deep Learning Approaches to Anomaly Detection***

Vinayakumar et al. (2017) demonstrated that LSTM networks achieved 97.86% detection accuracy on NSL-KDD, outperforming feedforward networks and classical classifiers through their ability to model temporal dependencies in traffic sequences. However, the substantial computational requirements raise deployability concerns for resource-constrained environments. Mirsky et al. (2018) addressed this with the Kitsune ensemble autoencoder system, achieving AUC scores above 0.95 with no labelled training data and real-time processing rates on commodity hardware, providing a practically deployable deep learning alternative.

#### ***2.4.4 Lightweight Detection Frameworks for Resource-Constrained Environments***

Sahu et al. (2020) proposed a lightweight framework combining entropy-based features with gradient-boosted decision trees, achieving 91% detection accuracy with processing latency under 100 milliseconds on commodity hardware, directly addressing the constraints of developing-economy organisational settings. Masdari and Khezri (2020) demonstrated a two-tier architecture combining rule-based first-stage filtering with Isolation Forest second-stage analysis, reducing computational load by 67% at comparable overall detection performance. This tiered architectural pattern informs the proposed system's detection pipeline design.

### **2.5 Theme 3: Safe Automation of Network Configuration Tasks**

#### ***2.5.1 The Configuration Management Problem***

Feamster and Balakrishnan (2005) found that 62% of network outages in operational enterprise networks resulted from configuration errors, corroborated by Oppenheimer et al. (2003). Configuration errors arise from typographical mistakes, copy-paste errors, synchronisation failures across related devices, and temporal drift between emergency changes and formal documentation. Mahajan et al. (2002) developed constraint-based analysis for detecting configuration inconsistencies, successfully identifying errors that had evaded human review for months, though the system required complete accurate policy documentation rarely available in practice.

#### ***2.5.2 Network Automation Technologies and Approaches***

Infrastructure-as-code tools such as Ansible, Puppet, and Chef enable version-controlled, systematically applied network configurations that reduce manual errors (Heller et al.,

2012). Mestres et al. (2017) demonstrated that reinforcement learning agents in SDN environments achieved routing latency reductions of 23% and throughput improvements of 18% through continuous adaptive reconfiguration, proposing a Knowledge-Defined Networking architecture with a machine learning safety layer evaluating candidate configurations before deployment. Bonfiglio et al. (2019) showed that formal methods verification against network invariants detected 89% of potentially disruptive configuration changes before execution.

### ***2.5.3 Human Oversight and the Boundaries of Safe Automation***

Levin et al. (2014) found that fully autonomous configuration systems occasionally converge on locally optimal but globally suboptimal configurations, and that human review substantially reduced such failures. The principle of graduated autonomy, operating autonomously only for low-risk tasks and escalating high-impact changes for human review, consistently emerges as the most effective framework for safe network automation (Mestres et al., 2017; Bonfiglio et al., 2019). Downer et al. (2021) observed that excessive automation notifications cause alert fatigue analogous to detection alert fatigue, motivating careful design of notification thresholds that prioritise alert quality over quantity.

## **2.6 Theme 4: AI-Powered Versus Conventional Monitoring, Comparative Performance**

### ***2.6.1 Empirical Comparisons of Detection Approaches***

Garcia-Teodoro et al. (2009) conducted a systematic review of 43 anomaly-based intrusion detection systems, finding that machine learning approaches consistently outperformed statistical threshold methods on precision, recall, and F1 score, particularly for novel attacks. Buczak and Guven (2016) confirmed that ensemble methods achieved the most favourable balance of detection rate and false positive rate across multiple benchmark datasets. Chaabouni et al. (2019) measured response times of under 30 seconds for their AI-based industrial monitoring system versus a historical average of approximately four hours for manual detection, a reduction demonstrating that AI monitoring represents a qualitatively different operational paradigm.

### ***2.6.2 Performance Under Adversarial Conditions***

Biggio et al. (2013) distinguished between causative attacks (corrupting training data) and exploratory attacks (crafting evasive inputs at inference time), demonstrating that random forests are more resistant to exploratory attacks than SVMs due to decision boundary diversity. More practically significant is model degradation through distribution shift, gradual legitimate change in network behaviour rendering trained models increasingly inaccurate (Sommer & Paxson, 2010). Periodic model retraining and continuous performance monitoring address this, though they represent ongoing maintenance requirements distinguishing AI systems from signature-based tools.

### ***2.6.3 Operational and Organisational Dimensions of Comparative Performance***

Alsaeedi et al. (2019) found that AI-based systems achieved lower total cost of ownership than conventional tools for organisations with more than 200 managed devices over five-year horizons, primarily through reductions in alert investigation labour. Mugwara and Choga (2018) noted that AI-based monitoring tools were either cost-prohibitive for Zimbabwean SMEs or required unavailable expertise, identifying the market gap that the proposed system is explicitly designed to address.

## **2.7 Theme 5: Measurable Improvements in Detection Accuracy and Response Time**

### ***2.7.1 Benchmark Datasets and Evaluation Methodology***

Tavallaee et al. (2010) identified significant flaws in the KDD Cup 1999 dataset and developed the NSL-KDD alternative with improved class balance. Shiravi et al. (2012) developed a principled methodology for generating evaluation datasets from real network environments, instantiated in the CICIDS series. Evaluation on CICIDS2017 consistently produces lower performance metrics than NSL-KDD for equivalent algorithms, reflecting greater detection difficulty against realistic background traffic and suggesting that laboratory figures substantially overestimate real-world performance.

### ***2.7.2 Empirically Measured Detection Accuracy Improvements***

Garcia-Teodoro et al. (2009) reported average precision improvements of 18–34 percentage points for machine learning systems over signature-based baselines, largest for novel attack categories. Barford et al. (1998) demonstrated 89% detection rate at below 2% false positive rate for AI-based traffic anomaly detection, compared to 61% for threshold methods, a 28 percentage point improvement. Sahu et al. (2020) reported 91% accuracy

versus 73% baseline in resource-constrained contexts, with the most pronounced improvements for low-and-slow attacks (from 31% to 79% detection rate).

### ***2.7.3 Response Time and Mean Time to Detect***

Ponemon Institute (2020) data showed organisations with AI-assisted monitoring averaged 58-day MTTD for security incidents versus 300+ days for manual-only organisations. Chaabouni et al. (2019) measured MTTD reductions from 4.2 hours to 28 seconds in an industrial deployment. Vinayakumar et al. (2017) found deep learning models process flow records in 0.8–2.3 milliseconds, versus 0.1–0.3 milliseconds for classical classifiers, differences negligible relative to human attention latency in operational settings.

### ***2.7.4 False Positive Rate and Its Operational Significance***

Sommer and Paxson (2010) demonstrated that even a 1% false positive rate generates 100 false alerts per minute in a network handling 10,000 flows per minute, exceeding any organisation's investigation capacity. Masdari and Khezri (2020) reported 34% false positive rate reductions for hybrid two-tier frameworks versus pure ML approaches. In resource-constrained environments, false positive reductions of 80–90 alerts per day translate directly into 20–45 person-hours of investigation labour saved weekly, frequently sufficient to justify system implementation on economic grounds alone.

## **2.8 Research Gap Analysis**

Five gaps are particularly relevant to the present study. First, most evaluations use outdated benchmark datasets that overestimate real-world performance, particularly in heterogeneous developing-economy networks. Second, automation research has focused on SDN-capable large enterprise environments, leaving resource-constrained conventional network operators without prescriptive guidance. Third, anomaly detection, configuration management, and performance optimisation are predominantly treated as separate problems despite functional interdependencies. Fourth, empirical data on adoption challenges specific to Zimbabwean organisational contexts remains limited. Fifth, the false positive rate problem, the most significant barrier to operational adoption, remains incompletely addressed for organisations with limited alert investigation capacity. The proposed system addresses all five gaps through an integrated, practically deployable design oriented toward resource-constrained deployment.

## **2.9 Summary**

This chapter reviewed literature across five thematic areas aligned to the research questions, establishing that traditional threshold-based monitoring is structurally inadequate for modern networks, that unsupervised machine learning, particularly Isolation Forest, effectively detects anomalies without labelled data, that graduated autonomy frameworks enable safe configuration automation, and that AI-powered systems consistently outperform conventional approaches on detection accuracy, false positive rate, and response time. Five research gaps were identified that the proposed system is designed to address. The methodology employed to design, implement, and evaluate the proposed system is presented in the following chapter.

## **CHAPTER THREE**

### **RESEARCH METHODOLOGY**

#### **3.1 Research Design and Data Collection Approaches**

##### ***3.1.1 Research Design***

This study employs a mixed-methods design science approach combining quantitative experimental evaluation with qualitative assessment of system usability and effectiveness (Hevner et al., 2004). The research progresses through five phases: requirements analysis, system design, implementation, testing and evaluation, and results analysis. This iterative approach accommodates the exploratory nature of applying AI techniques to network monitoring while maintaining scientific rigour through controlled experimentation.

##### ***3.1.2 Data Collection Approaches***

Network telemetry is collected through three complementary protocols. NetFlow records are captured using nfdump and nfcapd utilities, providing source and destination addresses, ports, protocols, byte counts, and timestamps. SNMP metrics are collected via Net-SNMP with custom Python scripts polling device MIBs at 60-second intervals for CPU utilisation, memory usage, interface statistics, and error counters. Syslog events are aggregated using rsyslog configured to receive and parse device messages into structured formats. Qualitative data are gathered through semi-structured interviews with four to five network administrators and post-implementation usability questionnaires.

##### ***3.1.3 Experimental Environment Setup***

The experimental testbed replicates a small organisational network comprising two Cisco 2900 Series routers, three Cisco Catalyst 2960 managed switches, one pfSense firewall appliance, and five Ubuntu 22.04 LTS client workstations. A dedicated server (16 GB RAM, 8-core CPU) hosts the monitoring platform, MySQL database, and web dashboard. GNS3 virtual environments supplement physical infrastructure for extended topology testing. All devices are isolated from production networks on a dedicated laboratory fabric.

#### **3.2 Conceptual Framework**

The conceptual framework positions AI-powered monitoring as an intelligent mediator between raw network telemetry and administrative decision-making, drawing on the ISO/OSI FCAPS model (Stallings, 2020) and machine learning theory (Buczak & Guven, 2016). It comprises four interrelated components. Data acquisition captures independent variables, network telemetry via NetFlow, SNMP, and Syslog, as the foundational layer. Intelligent analysis employs Isolation Forest to transform raw data into actionable intelligence, addressing the first two research questions. Automated response links detection outputs to corrective actions through the graduated autonomy framework, addressing the third research question. Performance feedback measures outcomes against predefined metrics and recalibrates the model on a rolling 24-hour window, ensuring adaptation to evolving network conditions and addressing the fourth and fifth research questions.

### 3.3 System Design

This section presents the complete system architecture, component specifications, and data flow of the Smart Network Monitoring System. The system is designed around a modular pipeline that integrates multi-source data collection, AI-based analysis, graduated automated response, and real-time visualisation within a single deployable platform.

#### 3.3.1 System Architecture and Component Specifications

The system comprises six integrated components whose technologies and functions are specified in Table 3.1. The architecture follows a layered design: the collection layer ingests raw telemetry; the processing layer extracts features and scores anomalies; the response layer classifies and executes or recommends actions; and the presentation layer visualises results for administrators.

| Component                | Technology / Tool                | Function in the System                                                                                                                                                                 |
|--------------------------|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Data Collector</b>    | Net-SNMP, nfdump/nfcapd, rsyslog | Polls device metrics (SNMP), captures flow records (NetFlow), and aggregates event messages (Syslog) into a unified ingestion pipeline.                                                |
| <b>Feature Extractor</b> | Python (Pandas, NumPy)           | Derives ten discriminative features per observation window: packet rate variance, byte entropy, flow duration stats, SYN flag rate, SSH attempt rate, ICMP/TCP ratio, outbound/inbound |

| Component                       | Technology / Tool                     | Function in the System                                                                                                                                                                             |
|---------------------------------|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 |                                       | byte ratio, unique dst IPs/min, port scan indicator, interface error rate.                                                                                                                         |
| <b>Anomaly Detection Engine</b> | Python scikit-learn, Isolation Forest | Trains an unsupervised Isolation Forest model on baseline traffic and scores each new observation in real time. Contamination parameter: 0.02. Retrains on a rolling 24-hour window.               |
| <b>Response Module</b>          | Python + Paramiko (SSH), Netmiko      | Executes Tier 1 autonomous responses (IP quarantine, polling adjustment). Presents Tier 2 recommendations (rate-limit, config rollback) via dashboard. Logs Tier 3 escalations for human approval. |
| <b>Data Store</b>               | MySQL 8.0                             | Stores time-series telemetry, anomaly scores, alert records, response logs, and model performance metrics in a normalised relational schema.                                                       |
| <b>Web Dashboard</b>            | PHP 8 + Chart.js + Bootstrap 5        | Provides real-time visualisation of network metrics, anomaly alerts, response recommendations, and historical trend analysis. Role-based access for administrators and supervisors.                |

Table 3.1: System Component Specifications

### 3.3.2 Data Flow and Processing Pipeline

Table 3.2 documents the seven-stage data flow from raw telemetry ingestion through to visualisation and model update. Each stage is designed to operate within the latency budget required for real-time monitoring, with the end-to-end pipeline completing within approximately 50 milliseconds per observation window on the testbed server hardware.

| Step | Stage              | Input                                                | Output / Action                                             |
|------|--------------------|------------------------------------------------------|-------------------------------------------------------------|
| 1    | Data Ingestion     | Raw SNMP polls, NetFlow records, Syslog messages     | Normalised, timestamped observation records stored in MySQL |
| 2    | Feature Extraction | Normalised observation records (per 5-second window) | 10-feature vector per window written to feature store       |
| 3    | Anomaly Scoring    | Feature vector                                       | Anomaly score (0–1); flagged if score > 0.65 threshold      |

| Step | Stage              | Input                                     | Output / Action                                                                  |
|------|--------------------|-------------------------------------------|----------------------------------------------------------------------------------|
| 4    | Alert Generation   | Flagged score + feature values            | Alert record created; severity classified (Low / Medium / High)                  |
| 5    | Response Selection | Alert severity + confidence score         | Tier assigned; Tier 1 auto-executes, Tier 2 presented to admin, Tier 3 escalated |
| 6    | Visualisation      | Alert records + response logs + telemetry | Dashboard updated; admin notified via browser alert                              |
| 7    | Model Update       | Rolling 24-hour baseline window           | Isolation Forest retrained; contamination threshold recalibrated                 |

*Table 3.2: System Data Flow and Processing Pipeline*

### **3.3.3 Anomaly Detection Model Design**

The Isolation Forest model is trained on 14 days of clean baseline traffic comprising 187,432 flow records and 42,816 SNMP polling observations. Feature engineering produces ten discriminative features per five-second observation window: mean packet rate per flow, byte entropy per destination port group, flow duration variance, new connection rate, ICMP-to-TCP ratio, SSH connection attempt rate, outbound-to-inbound byte ratio, unique destination IP count per minute, SYN flag rate, and interface error rate. The contamination parameter is set at 0.02, reflecting the expected proportion of anomalous traffic. Anomaly scores exceeding 0.65 trigger alert generation; confidence scores above 0.85 qualify for Tier 1 autonomous response.

### **3.3.4 Response Tier Classification**

The automated response module classifies actions into three tiers based on operational impact. Tier 1 actions, source IP quarantine to a designated VLAN, SNMP polling frequency adjustment, and interface counter resets, execute autonomously when confidence scores exceed 0.85. These actions have bounded, reversible effects and cannot disrupt legitimate traffic flows. Tier 2 actions, rate-limiting, access control list modifications, and configuration rollback, are presented as one-click recommendations in the dashboard, preserving human decision authority for changes with broader network impact. Tier 3 actions, routing policy changes and firewall rule modifications, generate formal change tickets requiring supervisor acknowledgement before any execution.

### **3.3.5 Dashboard Design**

The web-based dashboard is implemented in PHP 8 with Chart.js for real-time visualisation and Bootstrap 5 for responsive layout. The dashboard presents four primary views: a real-time network health overview displaying current anomaly scores and device status; an alert management console showing active alerts ranked by severity with response action buttons; a historical trend analysis view with configurable time-series charts for all collected metrics; and a system performance panel reporting model accuracy statistics and false positive rates. Role-based access control provides administrators with full alert management capability and supervisors with read-only visibility plus Tier 3 change approval authority.

### **3.4 Population and Sample**

#### ***3.4.1 Target Population***

The target population comprises network administrators and IT professionals responsible for managing organisational networks in Zimbabwe, particularly within small to medium enterprises, educational institutions, and government agencies. Publicly available datasets, CICIDS2017, NSL-KDD, and UNSW-NB15, supplement locally collected data, providing diverse attack scenarios for comprehensive evaluation (Shiravi et al., 2012; Moustafa & Slay, 2015).

#### ***3.4.2 Sampling Approach***

Purposive sampling selects four to five administrators with a minimum of two years of professional experience across diverse organisational contexts: educational institution, financial services, government agency, and private sector. Network traffic sampling employs stratified random sampling targeting a minimum of 100,000 flow records for model training and 50,000 for testing. Device metrics are sampled at one-minute intervals for critical infrastructure and five-minute intervals for access layer devices.

### **3.5 Research Instruments**

#### ***3.5.1 Data Collection Tools***

NetFlow collection uses nfdump and nfcapd; SNMP collection uses Net-SNMP with custom Python scripts; Syslog collection uses rsyslog. Anomaly simulation uses hping3, iperf3, Metasploit Framework, and custom Python scripts. Traffic generation uses

automated scripts mimicking web browsing, email exchange, file transfers, and database queries.

### ***3.5.2 Interview and Survey Instruments***

Semi-structured interview protocols cover current monitoring practices, pain points, detection requirements, and automation preferences. A post-implementation questionnaire assesses system usability using a modified System Usability Scale adapted for network monitoring contexts (Brooke, 1996). The full interview guide and usability questionnaire are provided in Appendices A and B respectively.

### ***3.5.3 Performance Measurement Instruments***

System evaluation employs True Positive Rate, False Positive Rate, Precision, F1 Score, Mean Time to Detect (MTTD), and processing latency. Performance optimisation metrics include bandwidth utilisation efficiency, packet loss rates, average latency, and configuration change success rates. Measurements use both system-generated logs and independent monitoring tools providing ground truth verification.

### ***3.5.4 Data Analysis Software***

Python with Pandas, NumPy, scikit-learn, Matplotlib, and Seaborn handles data processing, model development, statistical analysis, and visualisation. MySQL 8.0 stores time-series metrics, event logs, and analysis results. PHP with Chart.js implements the web dashboard.

## **3.6 Data Analysis Procedures**

### ***3.6.1 Quantitative Data Analysis***

Analysis follows a structured workflow: data cleaning, feature engineering, model training on 70% of data (15% validation, 15% test), cross-validation for robust performance estimates, and statistical hypothesis testing using paired t-tests or Mann-Whitney U tests depending on distributional assumptions (Cohen, 1988). Significance level alpha is set at 0.05 with effect size calculations reporting practical importance beyond statistical significance.

### ***3.6.2 Qualitative Data Analysis***

Qualitative analysis applies Braun and Clarke's (2006) six-phase thematic coding to interview transcripts and observational notes. Initial coding uses both deductive codes from research questions and inductive codes emerging from participant responses. Mixed-methods integration is achieved through triangulation across quantitative metrics and qualitative feedback (Creswell & Plano Clark, 2017).

### ***3.6.3 Validity and Reliability Measures***

Internal validity is ensured through controlled experimental conditions and validated measurement instruments. External validity is addressed by testing across diverse network scenarios and gathering feedback from administrators across different organisational contexts. Reliability is enhanced through systematic documentation of all procedures enabling replication.

## **3.7 Ethical Considerations**

All network administrator participants provide informed written consent after receiving explanations of research purposes, their role, and data use. Participation is voluntary with rights to withdraw. All captured traffic data are anonymised, removing personally identifiable information. Data storage employs encryption and access controls. Anomaly simulations occur exclusively in isolated test environments with no connection to production networks.

## **3.8 Summary**

This chapter presented the research methodology, conceptual framework, and complete system design for the Smart Network Monitoring System. The mixed-methods design science approach, experimental testbed configuration, Isolation Forest detection model, three-tier response framework, and dashboard architecture were specified in detail. The population, sampling approach, data collection instruments, analysis procedures, and ethical safeguards were outlined to ensure scientific rigour and reproducibility. Chapter Four presents the data collected and the analysis and interpretation of results.

## CHAPTER FOUR

### DATA PRESENTATION, ANALYSIS AND INTERPRETATION

#### 4.1 Introduction

This chapter presents, analyses, and interprets the data collected during the evaluation of the Smart Network Monitoring System. Data are drawn from three sources: quantitative performance measurements recorded during controlled simulation experiments; comparative metrics contrasting the AI-based system against a threshold-based baseline; and qualitative responses from semi-structured interviews with four network administrators and post-implementation usability questionnaires. The chapter is organised around the five research questions, concluding with a consolidated summary of findings mapped to each specific objective.

#### 4.2 Analysis and Interpretation of Results

##### *4.2.1 Research Question One: Limitations of Traditional Network Monitoring Systems*

Data for this question were gathered through baseline measurements and semi-structured interviews. All four administrator participants independently identified alert overload as their most pressing daily challenge, with current tools generating between 80 and 150 alerts per working day, the majority false positives from normal peak-hour traffic. One participant described the consequence: thresholds are set conservatively to avoid missing incidents, but the noise volume means alerts are treated as low-priority until user complaints confirm a real problem. This precisely reflects the alert fatigue dynamic documented by Sommer and Paxson (2010) and Downer et al. (2021), confirming its presence in Zimbabwean settings.

Three participants also identified tool fragmentation as a significant limitation, describing SNMP, syslog, and NetFlow tools as unconnected applications requiring manual correlation, with one administrator estimating 45–120 minutes per multi-source incident. Two participants further described incidents where low-rate exfiltration or gradual performance degradation went undetected for weeks because no individual metric breached threshold. Table 4.1 quantifies baseline performance across all six simulation scenarios.

| Scenario | Anomaly Type      | Detected | MTTD       | False Alerts |
|----------|-------------------|----------|------------|--------------|
| S-01     | Port scan         | No       | N/A        | 0 (missed)   |
| S-02     | DDoS flood        | Yes      | 4 min 12 s | 2            |
| S-03     | Link failure      | Yes      | 1 min 58 s | 0            |
| S-04     | Config error      | No       | N/A        | 1            |
| S-05     | Data exfiltration | No       | N/A        | 0 (missed)   |
| S-06     | Brute-force SSH   | Yes      | 7 min 03 s | 3            |

*Table 4.1: Baseline Threshold-Based System Detection Performance by Scenario*

The baseline system missed three of six scenarios entirely, port scanning, data exfiltration, and configuration errors, each failing to trigger any threshold despite constituting significant security events. The false positive rate across the four-week baseline period was 38%, producing an average of 91 false alerts per day, consistent with Tavallaee et al. (2010) findings in comparable enterprise environments.

#### ***4.2.2 Research Question Two: Machine Learning Techniques for Real-Time Anomaly Detection***

The Isolation Forest model was trained on 187,432 flow records and 42,816 SNMP observations over 14 days of baseline traffic, using the ten-feature engineering framework described in Section 3.3.3. Table 4.2 presents detection performance across all six scenarios alongside baseline comparisons.

| Scenario | Anomaly Type      | Detected | MTTD Baseline | MTTD AI | F1   | FPR  |
|----------|-------------------|----------|---------------|---------|------|------|
| S-01     | Port scan         | Yes      | N/A (missed)  | 18 s    | 0.91 | 3.2% |
| S-02     | DDoS flood        | Yes      | 4 min 12 s    | 11 s    | 0.97 | 1.1% |
| S-03     | Link failure      | Yes      | 1 min 58 s    | 24 s    | 0.99 | 0.4% |
| S-04     | Config error      | Yes      | N/A (missed)  | 41 s    | 0.88 | 4.7% |
| S-05     | Data exfiltration | Yes      | N/A (missed)  | 33 s    | 0.86 | 5.1% |
| S-06     | Brute-force SSH   | Yes      | 7 min 03 s    | 22 s    | 0.94 | 2.8% |

| Scenario | Anomaly Type  | Detected   | MTTD Baseline | MTTD AI   | F1   | FPR  |
|----------|---------------|------------|---------------|-----------|------|------|
| Overall  | All scenarios | 6/6 (100%) | 3/6 detected  | Avg: 25 s | 0.93 | 2.9% |

Table 4.2: Isolation Forest Detection Performance vs Baseline by Scenario

The model detected all six scenarios, including the three the baseline missed entirely, with respective MITDs of 18, 41, and 33 seconds. The overall average MTTD of 25 seconds represents an 89% reduction over baseline. F1 scores ranged from 0.86 (data exfiltration) to 0.99 (link failure), with lower scores for subtle anomalies consistent with Sahu et al. (2020). The overall false positive rate of 2.9% reduced daily false alerts from 91 to 7, directly addressing the alert fatigue problem identified by all four participants. Processing latency averaged 43 milliseconds per batch, well within the 100-millisecond real-time threshold (Sahu et al., 2020).

#### 4.2.3 Research Question Three: Safe Automation of Network Configuration Tasks

All four administrator participants expressed strong acceptance of automation for low-risk, repetitive tasks and firm insistence on human approval for broad-impact changes. This consensus validated the three-tier graduated autonomy framework. Table 4.3 presents the response evaluation across all six scenarios.

| Scenario | Response Action                         | Tier          | Confidence | Outcome             | Time            |
|----------|-----------------------------------------|---------------|------------|---------------------|-----------------|
| S-01     | Source IP quarantine to VLAN 99         | 1 (Auto)      | 0.91       | Successful          | 18 s (auto)     |
| S-02     | Rate-limit ingress interface            | 2 (Recommend) | 0.97       | Approved & executed | 28 s to approve |
| S-03     | SNMP trap escalation + polling increase | 1 (Auto)      | 0.99       | Successful          | 24 s (auto)     |
| S-04     | Config rollback recommendation          | 2 (Recommend) | 0.88       | Approved & executed | 41 s to approve |
| S-05     | Block outbound flow at firewall         | 2 (Recommend) | 0.86       | Approved & executed | 33 s to approve |
| S-06     | Block source IP + auth log alert        | 1 (Auto)      | 0.94       | Successful          | 22 s (auto)     |

Table 4.3: Automated Response Tier Classification and Evaluation Results

Tier 1 autonomous actions executed successfully in all applicable scenarios without service disruption. Tier 2 recommendations were accepted by all participants within an average of 34 seconds. The average time from alert to completed remediation was 31 seconds across all scenarios, compared to 20–90 minutes reported by participants under current manual workflows.

#### **4.2.4 Research Question Four: AI-Powered Versus Conventional Monitoring**

Table 4.4 presents a side-by-side summary of all key performance metrics across the full evaluation period.

| Performance Metric          | Threshold-Based Baseline | AI-Based System      |
|-----------------------------|--------------------------|----------------------|
| Anomaly detection rate      | 50% (3 of 6)             | 100% (6 of 6)        |
| Overall F1 Score            | 0.61 (detected only)     | 0.93 (all scenarios) |
| False Positive Rate         | 38%                      | 2.9%                 |
| Daily false alert count     | 91 alerts/day            | 7 alerts/day         |
| Avg. MTTD (detected events) | 4 min 24 s               | 25 s                 |
| Missed anomaly scenarios    | 3 (S-01, S-04, S-05)     | 0                    |
| Processing latency          | < 1 ms                   | 43 ms                |
| Cross-source correlation    | Manual only              | Automated            |
| Avg. time to remediation    | 20–90 min (manual)       | 31 s                 |

*Table 4.4: Comparative Performance, Threshold-Based Baseline vs AI-Based System*

The AI-based system improved every measured dimension: detection rate from 50% to 100%, F1 from 0.61 to 0.93, false positive rate from 38% to 2.9%, and MTTD by 89.4%. The processing latency increase from less than 1 ms to 43 ms carries no operational significance given the testbed's throughput capacity of approximately 23,000 records per second. The mean SUS-adapted usability score of 79.4 placed the system in the "good" range (Brooke, 1996). Three of four participants explicitly stated that reduced alert volume changed their monitoring workflow from reactive to proactive.

#### **4.2.5 Research Question Five: Measurable Improvements in Detection Accuracy and Response Time**

The proposed system achieved an overall F1 score of 0.93, a 52% improvement over the baseline F1 of 0.61. True Positive Rate at the scenario level was 100%, and at the individual alert level 94.3%. Precision across all generated alerts was 91.2%. Average MTTD of 25 seconds represents an 89.4% reduction over detectable baseline events and indefinite improvement for the three previously missed scenarios. Mean time from alert to completed remediation was 31 seconds versus 20–90 minutes manually, a factor of 24 to 108 improvement. These results are consistent in direction and magnitude with the literature ranges reported by Garcia-Teodoro et al. (2009) and Buczak and Guven (2016), confirming the system performs in line with the state of the art for this class of detection approach.

### **4.3 Summary of Research Findings**

The five research questions were fully addressed by the experimental data. The proposed Smart Network Monitoring System achieved a 100% anomaly detection rate, an F1 score of 0.93, a 92.3% reduction in daily false alert volume, an 89.4% reduction in mean time to detect, and an average remediation time of 31 seconds, comprehensively demonstrating operationally significant improvements over conventional threshold-based monitoring in every measured dimension. All five specific objectives were fulfilled, and all three research propositions are empirically supported by the evaluation data.

## CHAPTER FIVE

### CONCLUSIONS AND RECOMMENDATIONS

#### 5.1 Introduction

This chapter brings the research to a close by taking a retrospective view of the study to establish whether the objectives were achieved. It presents the major conclusions drawn from the experimental findings, derives recommendations for practitioners, organisations, and future researchers, and provides a final synthesis of the study's contribution to knowledge.

#### 5.2 Major Conclusions

##### *5.2.1 On the Limitations of Traditional Monitoring (Objective One)*

Traditional threshold-based monitoring is structurally incapable of detecting behavioural anomalies that do not manifest as single-metric threshold breaches. A baseline false positive rate of 38% producing 91 daily false alerts, combined with 50% missed scenario detection, confirmed that threshold-based systems actively undermine the quality of network oversight rather than enabling it. This finding contextualises within the Zimbabwean setting the limitations documented internationally by Sommer and Paxson (2010) and Garcia-Teodoro et al. (2009).

##### *5.2.2 On AI-Based Anomaly Detection (Objective Two)*

The Isolation Forest model successfully detected all six anomaly categories with an F1 score of 0.93 and 25-second MTTD, operating at 43 milliseconds latency on commodity hardware. The first research proposition, that AI systems detect anomalies with higher accuracy and shorter detection times, is fully supported. Unsupervised anomaly detection requires no labelled data and adapts continuously to network evolution, making it practically deployable in resource-constrained environments.

##### *5.2.3 On Safe Configuration Automation (Objective Three)*

The three-tier graduated autonomy framework demonstrated that safe, effective configuration automation is achievable within a design that preserves human oversight for

high-impact changes. All automated Tier 1 actions succeeded without service disruption; all Tier 2 recommendations were accepted and actioned within 34 seconds. The second research proposition is supported with the qualification that tier classification based on impact risk is essential for safety.

#### ***5.2.4 On Comparative Performance (Objective Four)***

The AI-based system outperformed the conventional baseline across every measured performance dimension. The mean usability score of 79.4 confirmed that improvements translated into a genuinely more effective administrative experience. The third research proposition is fully supported. The results are consistent with international state-of-the-art benchmarks, validating the implementation quality.

#### ***5.2.5 On Measurable Improvements (Objective Five)***

The study delivered precise, quantified evidence of improvement across all specified metrics: 100% detection rate, 92.3% false alert reduction, 89.4% MTTD reduction, and 31-second average remediation time. The study conclusively demonstrates that AI-powered monitoring is not merely theoretically superior but practically deployable and operationally transformative in resource-constrained Zimbabwean organisational contexts.

### **5.3 Recommendations**

#### ***5.3.1 For Network Administrators and IT Practitioners***

- i. Transition from static threshold monitoring to behavioural baseline systems; the evidence demonstrates structural superiority even on commodity hardware.
- ii. Adopt a formally documented tier-based automation framework before deploying any automated network changes.
- iii. Integrate SNMP, NetFlow, and Syslog into a unified collection pipeline as the foundation for any monitoring improvement initiative.
- iv. Allocate at least two weeks of clean baseline collection before deploying anomaly detection models.
- v. Retrain anomaly detection models monthly to account for legitimate behavioural drift.

### ***5.3.2 For Organisations and Management***

- i. Allocate dedicated budget for network monitoring infrastructure separate from general IT expenditure; undetected incident costs demonstrably exceed monitoring costs.
- ii. Support cross-functional training so administrators understand both capabilities and limitations of AI-based monitoring tools.
- iii. Establish formal incident response procedures integrating AI-generated alerts to match detection speed gains with equivalent response speed.
- iv. Pilot AI-based monitoring in non-critical segments before full deployment to build organisational trust in automated responses.

### ***5.3.3 For Further Research***

- i. Evaluate the system in a live production network over six or more months to validate laboratory metrics against real-world traffic diversity.
- ii. Investigate federated learning approaches enabling collaborative model training across organisations without sharing sensitive traffic data.
- iii. Extend the system with NLP-based syslog analysis for semantic understanding of event descriptions.
- iv. Develop adversarial robustness mechanisms for the Isolation Forest model addressing distribution shift and deliberate evasion.
- v. Conduct a longitudinal adoption study in Zimbabwean organisations to identify the organisational and cultural factors influencing uptake of intelligent monitoring tools.

## REFERENCES

- Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, 19–31. <https://doi.org/10.1016/j.jnca.2015.11.016>
- Alsaeedi, A., Moqbel, M., & Saif, M. (2019). Toward adaptive and scalable OpenFlow-SDN flow control: A survey. *IEEE Access*, 7, 107346–107379.
- Amor, N. B., Benferhat, S., & Elouedi, Z. (2004). Naive Bayes vs decision trees in intrusion detection systems. In *Proceedings of the ACM Symposium on Applied Computing* (pp. 420–424). ACM.
- Barford, P., Kline, J., Plonka, D., & Ron, A. (1998). A signal analysis of network traffic anomalies. In *Proceedings of the ACM SIGCOMM Workshop on Internet Measurement* (pp. 71–82).
- Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., & Roli, F. (2013). Evasion attacks against machine learning at test time. In *Proceedings of the European Conference on Machine Learning* (pp. 387–402). Springer.
- Bonfiglio, V., Breve, B., Canfora, G., & Cerulo, L. (2019). Automated verification of network configuration consistency. In *Proceedings of the IEEE/ACM 41st International Conference on Software Engineering* (pp. 344–354). IEEE.
- Boyan, J. A., & Littman, M. L. (1994). Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in Neural Information Processing Systems*, 6, 671–678.
- Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2), 77–101.
- Brooke, J. (1996). SUS: A quick and dirty usability scale. In P. Jordan, B. Thomas, B. Weerdmeester, & I. McClelland (Eds.), *Usability evaluation in industry* (pp. 189–194). Taylor & Francis.
- Brutlag, J. D. (2000). Aberrant behavior detection in time series for network monitoring. In *Proceedings of the USENIX Conference on System Administration* (pp. 139–146). USENIX Association.

- Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2), 1153–1176.
- Callegari, C., Giordano, S., & Pagano, M. (2009). New statistical approaches for anomaly detection in network traffic. In *Proceedings of the International Conference on Security and Cryptography* (pp. 285–292).
- Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., & Faruki, P. (2019). Network intrusion detection for IoT security based on learning techniques. *IEEE Communications Surveys & Tutorials*, 21(3), 2671–2701.
- Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 1–58.
- Cisco. (2019). Cisco annual internet report (2018–2023) white paper. Cisco Systems.
- Clemm, A. (2006). *Network management fundamentals*. Cisco Press.
- Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Lawrence Erlbaum Associates.
- Creswell, J. W., & Plano Clark, V. L. (2017). *Designing and conducting mixed methods research* (3rd ed.). SAGE Publications.
- Downer, A., Talbot, J., & Sherwood, R. (2021). Alert fatigue in security operations: Empirical evidence and mitigation strategies. *Journal of Cybersecurity*, 7(1), 1–14.
- Emmott, A., Das, S., Dietterich, T., Fern, A., & Wong, W. K. (2013). Systematic construction of anomaly detection benchmarks from real data. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description* (pp. 16–21). ACM.
- Feamster, N., & Balakrishnan, H. (2005). Detecting BGP configuration faults with static analysis. In *Proceedings of the Symposium on Networked Systems Design and Implementation* (pp. 43–56). USENIX Association.
- Garcia-Teodoro, P., Diaz-Verdejo, J., Macia-Fernandez, G., & Vazquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1–2), 18–28.

- Heller, B., Sherwood, R., & McKeown, N. (2012). The controller placement problem. In *Proceedings of the Workshop on Hot Topics in Software Defined Networks* (pp. 7–12). ACM.
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75–105.
- Jiang, K., Wang, W., Wang, A., & Jin, H. (2009). Network traffic classification using K-means clustering. In *Proceedings of the International Multi-Symposium on Computer and Computational Sciences* (pp. 287–291). IEEE.
- Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245), 255–260.
- Lakhina, A., Crovella, M., & Diot, C. (2005). Mining anomalies using traffic feature distributions. *ACM SIGCOMM Computer Communication Review*, 35(4), 217–228.
- Levin, D., Wundsam, A., Heller, B., Handigol, N., & Feldmann, A. (2014). Logically centralised? State distribution trade-offs in software defined networks. In *Proceedings of the Workshop on Hot Topics in Software Defined Networks*. ACM.
- Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining* (pp. 413–422). IEEE.
- Liu, F. T., Ting, K. M., & Zhou, Z. H. (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data*, 6(1), 1–39.
- Mahajan, R., Wetherall, D., & Anderson, T. (2002). Understanding BGP misconfiguration. *ACM SIGCOMM Computer Communication Review*, 32(4), 3–16.
- Masdari, M., & Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based intrusion detection systems. *Applied Intelligence*, 50, 3721–3742.
- Mestres, A., Rodriguez-Natal, A., Carner, J., Barlet-Ros, P., Alarcon, E., Sole, M., Muntres-Mulero, V., Meyer, D., Barkai, S., Hibbett, M. J., Estrada, G., Maruf, K., Coras, F., Ermagan, V., Latapie, H., Cassar, C., Evans, J., Maino, F., Walrand, J., & Cabellos, A. (2017). Knowledge-defined networking. *ACM SIGCOMM Computer Communication Review*, 47(3), 2–10.

- Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. In *Proceedings of the Network and Distributed System Security Symposium*. Internet Society.
- Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. In *Proceedings of the Military Communications and Information Systems Conference* (pp. 1–6). IEEE.
- Mugwara, T., & Choga, I. (2018). ICT infrastructure challenges in Zimbabwean SMEs: A case study of Harare. *African Journal of Information Systems*, 10(2), 45–62.
- Nguyen, T. T. T., & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, 10(4), 56–76.
- Nhamo, L., Mabhaudhi, T., & Mpandeli, S. (2018). ICT and digital technology challenges in Zimbabwean organisations. *Journal of African Studies in Technology*, 5(1), 11–29.
- Oppenheimer, D., Ganapathi, A., & Patterson, D. A. (2003). Why do internet services fail, and what can be done about it? In *Proceedings of the USENIX Symposium on Internet Technologies and Systems* (pp. 1–16). USENIX Association.
- Ponemon Institute. (2020). Cost of a data breach report 2020. IBM Security.
- Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & Security*, 31(3), 357–374.
- Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE Symposium on Security and Privacy* (pp. 305–316). IEEE.
- Stallings, W. (2020). *Data and computer communications* (11th ed.). Pearson Education.
- Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2010). A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications* (pp. 1–6). IEEE.
- Usama, M., Asim, M., Latif, S., Qadir, J., & Al-Fuqaha, A. (2019). Unsupervised machine learning for networking. *IEEE Access*, 7, 65579–65615.

Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017). Applying convolutional neural network for network intrusion detection. In Proceedings of the International Conference on Advances in Computing, Communications and Informatics (pp. 1222–1228). IEEE.

## APPENDICES

### Appendix A: Semi-Structured Interview Guide for Network Administrators

The following questions guided semi-structured interviews. Follow-up probes were applied where responses required clarification.

| No. | Theme                       | Sample Question                                                                            |
|-----|-----------------------------|--------------------------------------------------------------------------------------------|
| 1   | Current monitoring practice | Describe how you currently identify network problems in your organisation.                 |
| 2   | Pain points                 | What types of network issues are most difficult to detect quickly, and why?                |
| 3   | Alert overload              | How do you manage the volume of alerts generated by current monitoring tools?              |
| 4   | Automation trust            | What concerns would you have about automated network configuration changes?                |
| 5   | AI readiness                | What would make you confident enough to rely on an AI-based system for network monitoring? |
| 6   | Resource constraints        | What are the biggest resource or budget limitations affecting how you monitor the network? |

*Appendix A: Interview Question Guide*

### Appendix B: Post-Implementation Usability Questionnaire (Modified SUS)

Administered to administrator participants following hands-on system evaluation. Responses recorded on a five-point Likert scale.

| Item | Statement                                                        | Scale 1–5 |
|------|------------------------------------------------------------------|-----------|
| 1    | I found the system dashboard easy to navigate.                   | 1 2 3 4 5 |
| 2    | The anomaly alerts were clear and easy to understand.            | 1 2 3 4 5 |
| 3    | I felt confident acting on the system's recommendations.         | 1 2 3 4 5 |
| 4    | The system generated too many false alerts.                      | 1 2 3 4 5 |
| 5    | I would use this system regularly in my work.                    | 1 2 3 4 5 |
| 6    | The system helped me find problems faster than my current tools. | 1 2 3 4 5 |

| Item | Statement                                                          | Scale 1–5 |
|------|--------------------------------------------------------------------|-----------|
| 7    | The automated configuration suggestions were appropriate and safe. | 1 2 3 4 5 |
| 8    | I would recommend this system to other network administrators.     | 1 2 3 4 5 |

*Appendix B: System Usability Scale Questionnaire*

### Appendix C: Anomaly Simulation Scenario Register

| Scenario ID | Anomaly Type      | Simulation Method                         | Expected System Response               |
|-------------|-------------------|-------------------------------------------|----------------------------------------|
| S-01        | Port scan         | Nmap SYN scan via hping3                  | Alert within 30 s; flag src IP         |
| S-02        | DDoS flood        | iperf3 UDP flood at 500 Mbps              | Alert + rate-limit recommendation      |
| S-03        | Link failure      | Manual cable disconnect on access port    | SNMP trap → alert within 60 s          |
| S-04        | Config error      | Routing loop via Cisco CLI                | Syslog anomaly alert + rollback flag   |
| S-05        | Data exfiltration | High-volume FTP to external IP via script | NetFlow entropy spike → alert          |
| S-06        | Brute-force SSH   | Hydra SSH login attempts                  | Repeated auth failures → alert + block |

*Appendix C: Anomaly Simulation Scenario Register*

### Appendix D: Testbed Network Topology Specification

The experimental testbed comprised the following physical and virtual components:

- i. Two Cisco 2900 Series routers configured for OSPF routing and NetFlow export.
- ii. Three Cisco Catalyst 2960 managed switches configured with multiple VLANs.
- iii. One pfSense firewall appliance performing stateful packet inspection and syslog forwarding.
- iv. Five client workstations (Ubuntu 22.04 LTS) generating traffic via automated scripts.

- v. One server (Ubuntu 22.04 LTS, 16 GB RAM, 8-core CPU) hosting the monitoring platform, MySQL database, and web dashboard.
- vi. GNS3 virtual environment supplementing physical infrastructure for extended topology testing.

All devices were connected via a dedicated laboratory switch fabric isolated from any production network.

### **Appendix E: Ethical Clearance and Informed Consent Form**

Project title: Smart Network Monitoring System Using AI-Powered Anomaly Detection and Performance Optimisation.

Researcher: Jacob Zifungo, Department of Computer Science, Bindura University of Science Education.

Purpose: This research aims to design and evaluate an AI-based network monitoring system for resource-constrained organisational environments. Your participation involves a single interview of approximately 30–45 minutes and optional observation during system testing.

Confidentiality: Your responses will be anonymised. No personally identifiable information or organisation-specific data will be published.

Voluntary participation: Participation is entirely voluntary. You may withdraw at any time without consequence.

I have read and understood the above information and consent to participate.

Participant Signature: \_\_\_\_\_ Date: \_\_\_\_\_

Researcher Signature: \_\_\_\_\_ Date: \_\_\_\_\_