NEXT-GENERATION FIREWALL: Design, Implementation and Evaluation Using OPNsense

by

Showrov Rahim Ridoy
ID: CSE2301028051

MD. Rakibul Isalm
ID: CSE2301028052

Aisha Akther
ID: CSE2301028096

Aion Chandra Das
ID: CSE2301028082

Tarakul Islam
ID: CSE2301028132

Supervised by
Md. Shamim Hossain

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
September 2026

APPROVAL

The project titled **"NEXT-GENERATION FIREWALL: Design, Implementation and Evaluation Using OPNsense"** submitted by Showrov Rahim Ridoy (CSE2301028051), MD. Rakibul Isalm (CSE2301028052), Aisha Akther (CSE2301028096), Aion Chandra Das (CSE2301028082), and Tarakul Islam (CSE2301028132) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and content.

Board of Examiners

Md. Shamim Hossain — **Supervisor**

Lecturer
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 1**

Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 2**

Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 3**

Department of Computer Science and Engineering
Sonargaon University (SU)

DECLARATION

We hereby declare that the work presented in this thesis is the outcome of our study and laboratory design carried out under the supervision of **Md. Shamim Hossain, Lecturer,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We further declare that no part of this thesis has been submitted elsewhere for the award of any degree or diploma. Where the work of others has been used, it has been acknowledged through citations and references.

Countersigned

(Md. Shamim Hossain)

Supervisor

Signature

Showrov Rahim Ridoy
ID: CSE2301028051

MD. Rakibul Isalm
ID: CSE2301028052

Aisha Akther
ID: CSE2301028096

Aion Chandra Das
ID: CSE2301028082

Tarakul Islam
ID: CSE2301028132

ABSTRACT

Modern networks require controls that go beyond traditional IP address, port, and protocol filtering. This thesis presents the design of an open-source Next-Generation Firewall (NGFW) architecture using OPNsense as the gateway platform, Suricata for intrusion detection and prevention, and Zenarmor for deep packet inspection, application visibility, and web-category control. The proposed architecture separates the network into WAN, protected LAN, and DMZ security zones and applies layered policy enforcement so that stateful firewall rules, Layer-7 inspection, intrusion prevention, virtual private networking, authentication, and logging work as one coordinated security system. The implementation methodology is designed for a virtualized laboratory based on VMware Workstation, Proxmox, or ESXi. It covers interface and VLAN configuration, firewall and NAT policies, Suricata IDS/IPS integration, Zenarmor policies, web filtering, optional TLS inspection, OpenVPN or WireGuard remote access, captive portal and directory-based authentication, and central monitoring through OPNsense logs and optional ELK/Syslog integration. Validation is organized around controlled port scanning, web-application attack simulations, benign and malicious download tests, rule-verification cases, and throughput/latency measurements under increasing security inspection. The work emphasizes a practical balance among security visibility, performance, manageability, privacy, and cost. Because no verified laboratory screenshots or measured benchmark dataset were supplied with the source materials, this manuscript marks screenshot positions and the numerical evaluation table in Chapter 5 as lab-replacement items rather than presenting invented evidence.

ACKNOWLEDGEMENT

First and foremost, we express our sincere gratitude to the Almighty for giving us the strength, patience, and opportunity to complete this work.

We are deeply grateful to our supervisor, **Md. Shamim Hossain, Lecturer,** Department of Computer Science and Engineering, Sonargaon University, for his guidance, technical direction, constructive comments, and encouragement throughout the planning of this Next-Generation Firewall thesis.

We also acknowledge the faculty members of the Department of Computer Science and Engineering for providing the academic foundation required to understand computer networks, operating systems, cybersecurity, and research methodology.

Finally, we thank our families, friends, and classmates for their support. We also acknowledge the OPNsense, Suricata, Zenarmor, FreeBSD, and wider open-source communities whose documentation and tools make low-cost security research and practical experimentation possible.

LIST OF ABBREVIATIONS

AD	Active Directory
APT	Advanced Persistent Threat
CA	Certificate Authority
CPU	Central Processing Unit
DPI	Deep Packet Inspection
DMZ	Demilitarized Zone
DNS	Domain Name System
ELK	Elasticsearch, Logstash, Kibana
FQDN	Fully Qualified Domain Name
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
NGFW	Next-Generation Firewall
NAT	Network Address Translation
OPNsense	Open-source firewall and routing platform
RAM	Random Access Memory
SIEM	Security Information and Event Management
SPI	Stateful Packet Inspection
SQLi	SQL Injection
TLS	Transport Layer Security
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
WAN	Wide Area Network
XSS	Cross-Site Scripting

TABLE OF CONTENTS

Title			Page No.
DECLARATION			iii
ABSTRACT			iv
ACKNOWLEDGEMENT			v
LIST OF ABBREVIATIONS			vi
CHAPTER 1			1 – 7
INTRODUCTION			
1.1	Introduction		1
1.2	Background and Motivation		1
1.3	Problem Statement		2
1.4	Fundamental Concepts: Traditional Firewall vs NGFW		3
1.5	Literature and Technology Review		4
	1.5.1	OPNsense Firewall Platform	4
	1.5.2	Suricata IDS/IPS	4
	1.5.3	Zenarmor DPI & Application / WEB Control	5
	1.5.4	WEB Filtering, Proxy & TLS Inspection	5
	1.5.5	VPN and User Authentication	5
1.6	Research Gap		6
1.7	Objectives		6
1.8	Scope and Limitations		7
CHAPTER 2			8 – 13
METHODOLOGY			
2.1	Introduction		8
2.2	Research and Development Methodology		8

Title Page No.

2.3 Requirements and Threat Model..... 9
2.4 Virtual Laboratory Design..... 10
2.5 Configuration Methodology..... 11
2.6 Testing and Validation Methodology..... 12
2.7 Ethics and Safety..... 12

CHAPTER 3 14 – 17
SYSTEM ARCHITECTURE AND TOPOLOGY DESIGN
3.1 Introduction..... 14
3.2 High-Level Network Topology..... 14
3.3 High-Level System Architecture..... 15
3.4 WAN, LAN, DMZ and VLAN Design..... 15
3.5 Traffic Flow and Security Layer..... 16
3.6 Logging and Management Architecture..... 17
3.7 Design Rationale..... 17

CHAPTER 4 18 – 27
IMPLEMENTATION
4.1 Environment Setup..... 18
4.2 OPNsense Installation and Network Interfaces Configuration..... 18
4.3 Stateful Packet Inspection and Firewall Rules..... 19
4.4 Suricata IDS/IPS..... 20
4.5 Zenarmor DPI and Application Control..... 21
4.6 Web Filtering and TLS Inspection..... 21
4.7 Virtual Private Network (VPN)..... 22
4.8 User Authentication and Captive Portal..... 22
4.8.1 Configuration Backup, Change Control and Recovery..... 23

Title Page No.

4.8.2 Policy Verification and Troubleshooting Workflow……..…….. 23

4.9 Real-Time Monitoring, Logging, and Implementation.…… 24

CHAPTER 5 28 – 34

RESULTS AND DISCUSSION

5.1 Introduction and Test Plan.....…… 28

5.2 Penetration and Attack Simulation.....…… 28

5.3 Performance and Throughput Benchmarking.....……. 29

5.4 Detection and Policy Validation.....…… 29

5.5 Discussion: Security Effectiveness.....……... 30

5.6 Discussion: Performance, Manageability and Cost.....…….. 31

5.6.1 Quantitative Evaluation Metrics……..…… 32

5.6.2 Operational Security Considerations…….……..… 33

5.7 Limitations of the Evaluation……...… 34

CHAPTER 6 35-36

CONCLUSION AND FUTURE WORK

6.1 Conclusion.....…… 35

6.2 Future Work.....…… 36

REFERENCES.....…… 37

LIST OF TABLES

Table No.	Title	Page No.
Table 1.1	Traditional Firewall and NGFW Comparison	3
Table 2.1	Laboratory Software and Tools	11
Table 2.2	Security Test Cases	12
Table 3.1	Interface and Zone Addressing Plan	16
Table 4.1	Baseline Firewall Rule Matrix	19
Table 4.2	Suricata Deployment Parameters	20
Table 4.3	Zenarmor Policy Examples	21
Table 5.1	Attack Simulation Validation Matrix	28
Table 5.2	Security Feature Validation Summary	29

LIST OF FIGURES

Figure No.	Title	Page No.
Fig. 2.1	Controlled Virtual Laboratory Testbed	10
Fig. 3.1	High-Level Network Topology	14
Fig. 3.2	Layered NGFW System Architecture	15
Fig. 3.3	Traffic Flow and Security Processing Pipeline	16
Fig. 4.1	OPNsense System Dashboard	24
Fig. 4.2	Suricata IDS/IPS Rules and Alert Lo	25
Fig. 4.3	Log File	26
Fig. 4.4	Zenarmor Application Control and Web Category Analytics	26
Fig. 4.5	Zenarmor Application Control and Web Category Analytics	27
Fig. 4.6	Traffic / Bandwidth Usage Graph	27

CHAPTER 1
INTRODUCTION

1.1 Introduction

The network perimeter is no longer a simple boundary between a trusted internal network and an untrusted Internet. Organizations now depend on cloud services, remote access, web applications, encrypted traffic, mobile endpoints, and third-party platforms. At the same time, attackers increasingly use legitimate protocols and encrypted channels, which reduces the value of security decisions based only on IP addresses and port numbers. A modern firewall therefore needs to understand not only where traffic is coming from and where it is going, but also what the traffic represents, which application is involved, whether the content is suspicious, and whether the connection is consistent with the organization's security policy [1]–[4].

This thesis investigates a practical open-source Next-Generation Firewall (NGFW) design based on OPNsense. OPNsense provides the routing, stateful packet filtering, network address translation, virtual private networking, captive portal, and management framework. Suricata adds signature- and protocol-aware intrusion detection and prevention, while Zenarmor adds application-aware deep packet inspection, web control, and security analytics. Together, these components allow a small or medium organization to build a layered security gateway without relying entirely on an expensive proprietary firewall appliance [1], [3], [5].

The central idea is defense in depth. The packet filter should reject traffic that is unnecessary at Layer 3/4; application and web controls should regulate permitted traffic at Layer 7; the IDS/IPS should detect malicious patterns and protocol anomalies; and logging should preserve enough evidence to understand what happened. The thesis therefore treats the NGFW not as a single feature, but as an architecture that coordinates multiple inspection and control mechanisms.

1.2 Background and Motivation

Traditional firewalls were designed primarily to permit or deny traffic according to source address, destination address, protocol, and port. Stateful firewalls improved this model by

tracking connection state and allowing return traffic only when it belongs to a legitimate session. These mechanisms remain essential, but they provide limited visibility when an application changes ports, tunnels traffic through HTTPS, or hides malicious activity inside an otherwise permitted protocol. This limitation becomes more important as most user-facing Internet services move to encrypted web traffic.

The NGFW thesis proposal identifies the same practical problem: modern networks are more complex, encrypted traffic can hide malicious content, shadow IT expands the attack surface, and security teams need real-time context about users, devices, and applications. The proposed solution is a modular NGFW that combines firewalling, IDS/IPS, DPI, secure VPN, authentication, logging, and threat-aware policy enforcement. The proposal also emphasizes performance evaluation because deeper inspection consumes CPU, memory, and throughput capacity. This balance between security and performance motivates the design used throughout the thesis.

OPNsense was selected because it is an open-source, FreeBSD-based firewall and routing platform with a web interface, package ecosystem, VLAN support, VPN services, intrusion detection integration, and extensive logging. Its current documentation also supports virtual deployments and recommends allocating sufficient memory and storage when enabling disk-intensive and inspection-heavy features. This makes it well suited to a university laboratory where the same topology can be reproduced on VMware Workstation, Proxmox, or ESXi [1]-[2].

1.3 Problem Statement

Small and medium organizations frequently need NGFW-level controls but face constraints in budget, hardware, specialist staff, and licensing. A basic router or traditional firewall may provide NAT and access-control rules but may not provide enough application visibility, intrusion prevention, encrypted-traffic policy, remote-access security, and central analytics. The specific research problem addressed by this thesis is therefore: how can an open-source OPNsense-based security gateway be designed and configured to deliver practical NGFW capabilities while maintaining understandable policy structure and measurable network performance?

- **Limited application awareness:** ports and protocols do not always identify the actual application or user activity.
- **Encrypted traffic visibility:** HTTPS can conceal malware delivery, command-and-control activity, and policy violations.
- **Intrusion prevention requirements:** perimeter filtering alone cannot detect many exploit signatures, scans, and protocol anomalies.
- **Network segmentation:** a flat network increases the impact of compromise; LAN, DMZ, and VLAN policy boundaries are required.
- **Operational visibility:** without dashboards and logs, administrators cannot verify whether controls are working or investigate incidents.
- **Performance trade-off:** IDS/IPS, DPI, logging, and TLS inspection increase processing cost and can reduce throughput.

1.4 Fundamental Concepts: Traditional Firewall vs NGFW

A traditional stateful firewall and an NGFW share the same basic foundation, including packet filtering, connection-state tracking, routing, and NAT. However, an NGFW extends these functions by providing deeper content and application awareness, integrated intrusion prevention, user- or identity-aware policy enforcement, web security, encrypted traffic inspection, and more detailed threat analytics. The key differences between the two approaches, including IP/port filtering, connection tracking, application identification, IDS/IPS, DPI, web/category control, user/context awareness, encrypted traffic policy, and analytics, are presented in **Table 1.1**. In practice, the most effective architecture combines both layers, where Layer 3/4 rules first remove traffic that has no business purpose before more resource-intensive Layer-7 inspection is applied [3], [5].

Table 1.1: Traditional Firewall and NGFW Comparison

Capability	Traditional Stateful Firewall	NGFW-Oriented Design
IP/port filtering	Core function	Core function
Connection tracking	Yes	Yes

Application identification	Limited / port-based	Layer-7 classification
IDS/IPS	External or optional	Integrated inspection
DPI	Usually limited	Key capability
Web/category control	Basic or proxy-dependent	Policy-based
User/context awareness	Limited	Supported through identity integration
Encrypted traffic policy	Basic pass/block	Metadata and optional TLS inspection
Analytics	Basic logs	Application, threat, user and web analytics

1.5 Literature and Technology Review

This section reviews the main technologies and security mechanisms used in the proposed NGFW architecture. It covers the OPNsense firewall platform, Suricata IDS/IPS, Zenarmor for deep packet inspection and application control, web filtering and TLS inspection, and VPN with user authentication. The purpose is to establish the technical foundation for the system design, configuration, and validation presented in the following chapters.

1.5.1 OPNsense Firewall Platform

OPNsense is an open-source firewall and routing platform built on FreeBSD. It provides a web-based management interface for firewall rules, NAT, interfaces, VLANs, DHCP, DNS, VPN, captive portal, high availability, logging, and plugins. Official hardware guidance distinguishes minimum, reasonable, and recommended profiles and notes that features such as intrusion detection, alert databases, captive portal, and caching proxy can substantially affect resource requirements [1]. Virtual installation is supported, with guidance for VMware and other hypervisors [2].

1.5.2 Suricata IDS/IPS

Suricata is an open-source network threat detection engine capable of intrusion detection, intrusion prevention, network-security monitoring, protocol identification, and rule-based alerting. In an OPNsense deployment, Suricata can inspect traffic on selected interfaces and,

when configured for IPS/inline operation, can drop traffic that matches blocking rules. Signature updates, alert tuning, interface selection, and false-positive management are important operational tasks because an untuned rule set may produce excessive alerts or block legitimate traffic [6]-[7].

1.5.3 Zenarmor DPI and Application/Web Control

Zenarmor extends an OPNsense deployment with Layer-7 visibility, application controls, web controls, security analytics, and, depending on edition, advanced inspection features. The vendor documentation distinguishes OPNsense Layer-4 policy from Zenarmor Layer-7 policy: the administrator should first define appropriate stateful firewall rules and then apply application and web policy to the traffic that is allowed through [3]. Zenarmor web controls are designed for HTTP/HTTPS classification and category enforcement, while application controls classify traffic across protocols using an application-signature database [4]-[5].

1.5.4 Web Filtering, Proxy and TLS Inspection

Web filtering can be implemented through category-based controls, DNS filtering, proxy services, or a combination of these approaches. Full TLS inspection is more powerful because it decrypts, inspects, and re-encrypts traffic, but it also introduces privacy, legal, certificate-management, compatibility, and performance issues. Modern inspection platforms therefore support selective bypass for certificate-pinned or privacy-sensitive services. The architecture in this thesis treats full TLS inspection as an optional controlled-lab feature rather than a default requirement for all networks [8]-[9].

1.5.5 VPN and User Authentication

Remote users must be able to access internal resources securely without exposing management interfaces, administrative services, or other sensitive systems directly to the Internet. OPNsense supports widely used VPN technologies such as OpenVPN and WireGuard, allowing encrypted remote-access connections between authorized users and the internal network. Authentication can be managed through local user accounts or integrated with centralized directory services such as LDAP and Microsoft Active Directory. In addition, the captive portal feature can enforce user login, access restrictions, and network usage policies for guest, student, or managed user

networks. These mechanisms improve access control by allowing security policies to consider user identity, authentication status, assigned privileges, and source network location rather than relying only on IP addresses. As a result, organizations can provide flexible remote connectivity while maintaining stronger control over who can access specific internal resources and services

1.6 Research Gap

Commercial NGFW literature is rich in feature descriptions, but university-level implementation reports often isolate one feature, such as IDS or firewall rules, instead of evaluating how Layer-4 filtering, Layer-7 application control, IDS/IPS, network segmentation, VPN, and monitoring interact within one reproducible open-source security gateway. A second gap is that network performance is often discussed in general terms without applying a structured baseline-to-inspection comparison to measure the effects of additional security controls. Furthermore, limited academic work demonstrates how multiple open-source NGFW components can be integrated, configured, tested, and evaluated under a common network architecture using practical validation scenarios. This creates a need for an implementation-oriented study that considers both security effectiveness and performance impact. This thesis addresses these gaps by defining a complete OPNsense-centered architecture and a repeatable validation plan that compares security behavior and network performance across progressively enabled inspection layers, while also documenting configuration procedures, testing methods, and observable outcomes for future academic replication.

1.7 Objectives

- To study the fundamental difference between traditional stateful firewalls and NGFW architectures.
- To design a modular OPNsense-based NGFW topology for a small-to-medium enterprise environment using WAN, LAN, DMZ, and VLAN segmentation.
- To configure stateful firewall rules, NAT, IDS/IPS with Suricata, DPI/application control with Zenarmor, web filtering, VPN, authentication, and logging.
- To validate policy behavior using controlled benign and malicious traffic, including port scans and web-application attack simulations.

- To measure the effect of security inspection on throughput and latency and analyze the security-performance trade-off.
- To produce a reproducible academic implementation guide, diagrams, tables, screenshots checklist, and recommendations for future deployment.

1.8 Scope and Limitations

The scope is limited to a controlled laboratory or small-enterprise-style deployment. The target design uses one OPNsense gateway and multiple virtual machines representing clients, servers, logging services, and attack/test hosts. The work focuses on perimeter and inter-zone controls rather than endpoint detection, cloud-native firewalls, or a full Security Operations Center. It also focuses on defensive testing; offensive actions are limited to isolated systems owned by the researchers and deliberately configured for security validation.

A critical limitation is the absence of actual implementation screenshots and verified benchmark measurements in the materials supplied for this manuscript. To avoid manufacturing evidence, Chapter 4 includes clearly labelled screenshot insertion points.

CHAPTER 2
METHODOLOGY

2.1 Introduction

This chapter describes the methodology used to translate the approved NGFW proposal into a structured and testable laboratory design. The approach combines design science, incremental implementation, configuration verification, controlled security testing, and network performance measurement. The methodology is organized so that each security function can be implemented, observed, and validated in a controlled environment before being integrated into the complete gateway architecture. Each major capability is therefore tested separately to identify configuration issues and performance effects more clearly. After individual validation, the complete security stack is evaluated as a combined system to determine whether the selected open-source NGFW components can provide effective security while maintaining acceptable network performance.

2.2 Research and Development Methodology

The project follows an incremental experimental methodology in which the NGFW environment is developed and tested in several controlled stages. First, the researchers establish a stable OPNsense router with working WAN and LAN connectivity to confirm basic routing and Internet access. Second, the network is segmented into protected LAN, DMZ, and optional VLANs to separate systems according to their security requirements. Third, stateful firewall rules and Network Address Translation (NAT) are implemented to control communication between network zones.

Fourth, Suricata is enabled initially in detection mode so that generated alerts can be reviewed and rules can be tuned before IPS blocking is introduced. Fifth, Zenarmor is integrated to provide Layer-7 inspection, application control, and web filtering capabilities. Sixth, VPN access, user authentication, captive portal, and centralized logging are configured to improve secure access and visibility. Finally, controlled security tests and performance measurements are repeated under different combinations of enabled features. This staged approach helps identify the contribution and performance impact of each security layer.

- **Phase 1:** Requirement analysis and threat-model development.
- **Phase 2:** Virtual network topology, interface, and IP-addressing design.
- **Phase 3:** OPNsense installation, routing configuration, stateful firewall rules, and NAT implementation.
- **Phase 4:** Suricata IDS/IPS configuration, signature selection, alert monitoring, and rule tuning.
- **Phase 5:** Zenarmor DPI, application control, web filtering, and policy configuration.
- **Phase 6:** VPN, authentication, captive portal, monitoring, and logging configuration.
- **Phase 7:** Controlled attack simulation, detection testing, segmentation testing, and security-policy validation.
- **Phase 8:** Throughput and latency benchmarking, result analysis, comparison, documentation, and final reporting.

2.3 Requirements and Threat Model

The threat model assumes that the Internet and any untrusted/guest segments may contain hostile hosts. Internal clients are trusted only to the extent required for business use; malware or misconfiguration may exist inside the LAN. DMZ servers are Internet-facing and therefore treated as exposed assets that must not have unrestricted access to the internal LAN. The firewall management interface is considered highly sensitive and should be reachable only from an administrator network or explicit management host.

- External scanning and enumeration of exposed services.
- Malware or exploit traffic entering through permitted web connections.
- Web-application attacks against a DMZ test server, including SQL injection and XSS patterns.
- Unauthorized applications or website categories leaving the LAN.
- Lateral movement from DMZ toward protected LAN.

- Unauthorized remote access attempts against VPN or management services.
- Excessive false positives that reduce the operational value of IDS/IPS.

2.4 Virtual Laboratory Design

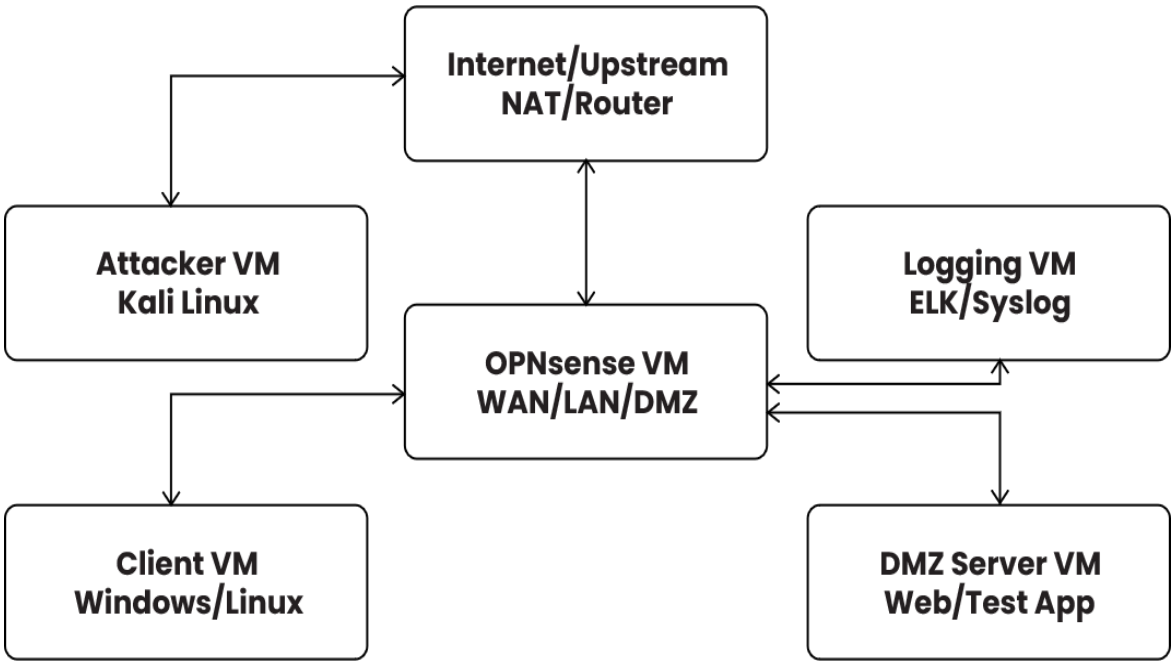


Fig. 2.1: Controlled Virtual Laboratory Testbed

The virtual laboratory can be implemented in VMware Workstation for desktop-based testing or in Proxmox/ESXi for a server-based environment. OPNsense is configured with WAN, LAN, and DMZ interfaces, while additional VLANs may be carried through tagged LAN interfaces where supported. A Kali Linux VM acts as the external attacker, Windows/Linux client VMs represent internal users, a vulnerable web application is placed in the DMZ for controlled testing, and an optional ELK/Syslog VM collects logs for centralized analysis. The overall structure and communication flow of the controlled virtual laboratory are presented in **Fig. 2.1**.

2.5 Configuration Methodology

The configuration methodology is based on a controlled virtual laboratory that combines multiple security and networking tools to implement, test, and validate the NGFW environment.

The selected components include OPNsense as the firewall platform, VMware, Proxmox, or ESXi for virtualization, Suricata for IDS/IPS, Zenarmor for DPI and application control, Unbound DNS for local name resolution and policy support, OpenVPN or WireGuard for secure remote access, local or directory-based authentication for user-aware access, and OPNsense logs, Syslog, or ELK for event monitoring and reporting. Traffic-generation and attack-simulation tools such as iperf3, curl, browsers, Nmap, and controlled web-test tools are also used to evaluate performance and verify defensive rules. These laboratory software components, selected tools, and their respective purposes are summarized in **Table 2.1**.

Table 2.1: Laboratory Software and Tools

Category	Selected Tool / Approach	Purpose
Firewall platform	OPNsense	Routing, NAT, stateful policy, VPN, portal, logging
Virtualization	VMware / Proxmox / ESXi	Repeatable isolated lab
IDS/IPS	Suricata	Threat detection and inline blocking
DPI / App Control	Zenarmor	Layer-7 visibility and policy
DNS	Unbound DNS	Local resolver and policy support
VPN	OpenVPN or WireGuard	Secure remote access
Authentication	Local / LDAP / Active Directory	User-aware access
Logging	OPNsense logs / Syslog / ELK	Event analysis and reporting
Traffic generation	iperf3, curl, browsers, scan tools	Performance and validation
Attack simulation	Nmap, controlled web-test tools	Defensive rule verification

2.6 Testing and Validation Methodology

The testing and validation methodology evaluates whether the configured NGFW functions correctly under different security and performance scenarios. The test cases cover port scanning, application blocking, IDS/IPS detection, DMZ access control, VPN login, web-attack detection,

normal browsing, and throughput measurement. The expected results and required evidence for each test are summarized in **Table 2.2**.

Table 2.2: Security Test Cases

Test ID	Scenario	Expected Security Result	Evidence
T1	External port scan against WAN	Only deliberately exposed services visible	Scan result + firewall log
T2	Blocked LAN category/application	Connection blocked by Zenarmor policy	Zenarmor session/analytics
T3	Suricata signature test	Alert generated; IPS mode blocks when configured	Suricata alert log
T4	DMZ attempts access to protected LAN	Denied except explicit management flows	Firewall live log
T5	VPN remote-access login	Authorized user obtains assigned VPN access	VPN status + log
T6	SQLi/XSS pattern against test web app	Alert/block according to enabled rules	IDS/IPS evidence
T7	Benign download and browsing	Allowed without false blocking	Browser/session log
T8	Throughput baseline vs inspection	Performance delta recorded	iperf3 results

2.7 Ethics and Safety

The attack simulations and security tests described in this thesis are intended only for systems, virtual machines, and network resources that are owned, controlled, or explicitly authorized by the research team. Activities such as port scanning, vulnerability testing, exploit simulation, malware-sample handling, password testing, and TLS interception may create legal, privacy, or operational risks if they are performed against third-party systems or production networks without proper authorization. Therefore, all experiments should be conducted within an isolated and controlled laboratory environment.

TLS decryption and inspection should be limited to test networks or environments operating under a formally approved organizational security policy. Appropriate bypass rules should be applied for privacy-sensitive services, financial or healthcare-related traffic, certificate-pinned applications, and other services where interception may cause privacy or compatibility issues.

The research process must also avoid collecting usernames, passwords, personal information, private communications, or any third-party data that are not necessary for the experiment. All logs, packet captures, test results, and security alerts should be handled responsibly and used only for academic analysis. These precautions ensure that the experiments remain ethical, legally appropriate, and technically safe while still allowing the NGFW security controls to be evaluated effectively.

CHAPTER 3
SYSTEM ARCHITECTURE AND TOPOLOGY DESIGN

3.1 Introduction

The proposed NGFW architecture uses network segmentation and layered inspection to reduce attack surface and improve visibility. OPNsense is positioned as the central security gateway. All traffic crossing from WAN to LAN, WAN to DMZ, LAN to DMZ, or selected VLAN boundaries is evaluated according to stateful rules and, where enabled, deeper security services.

3.2 High-Level Network Topology

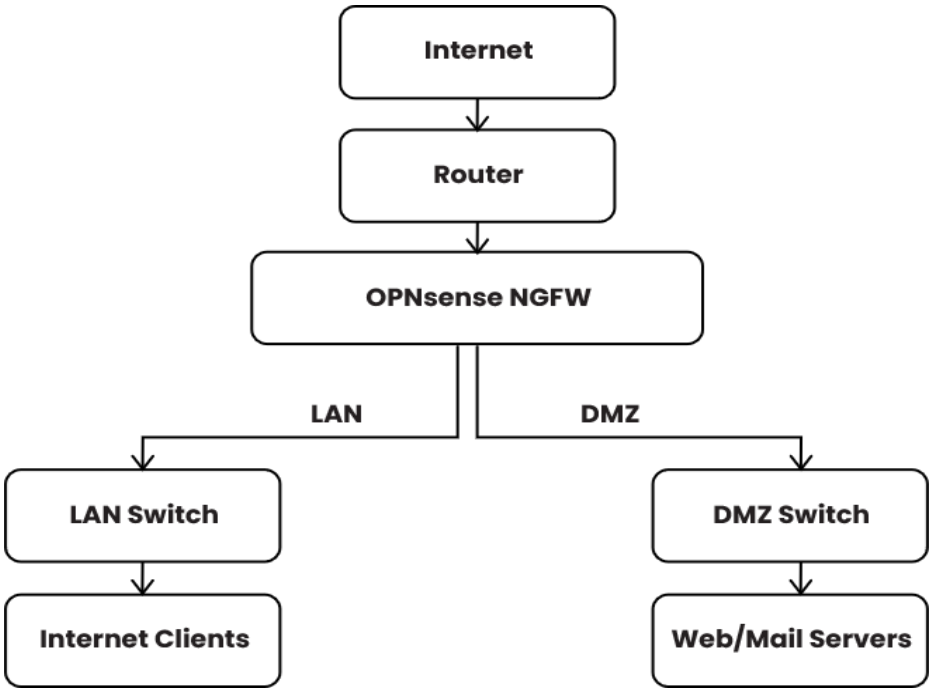


Fig. 3.1: High-Level Network Topology

The high-level network topology separates the environment into WAN, LAN, and DMZ zones, with OPNsense NGFW acting as the central security and policy enforcement point. The LAN protects internal users and services, while the DMZ hosts Internet-facing servers such as web or

mail servers with restricted access to the internal network. Additional VLANs can also be used to separate administration, staff, guest, server, and management traffic. The overall network structure and zone separation are presented in **Fig. 3.1**.

3.3 High-Level System Architecture

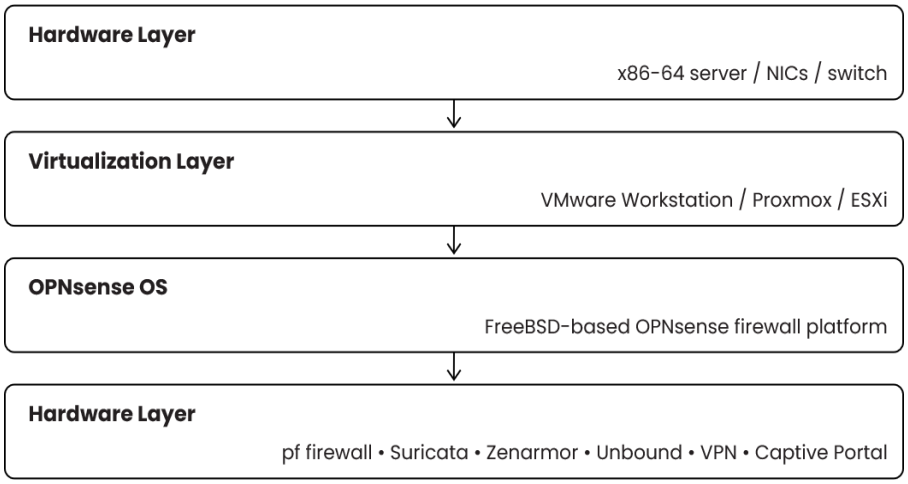


Fig. 3.2: Layered NGFW System Architecture

The high-level system architecture is organized into four layers: hardware, virtualization, OPNsense operating system, and security services. The hardware layer provides the computing and network resources, while the virtualization layer hosts the firewall platform in a controlled environment. OPNsense manages routing, interfaces, and firewall functions, and the security-services layer integrates Suricata, Zenarmor, Unbound DNS, VPN, captive portal, and related security features. The complete layered architecture of the NGFW system is presented in **Fig. 3.2**.

3.4 WAN, LAN, DMZ and VLAN Design

The interface and zone addressing plan defines the IP ranges, trust levels, and main security policies for the WAN, LAN, DMZ, management, user, and guest networks. Each zone is assigned a clear access level to control communication between internal and external network

segments. The example addresses, trust levels, and corresponding policies are summarized in **Table 3.1**.

Table 3.1: Interface and Zone Addressing Plan

Interface / Zone	Example Address	Trust Level	Main Policy
WAN	DHCP or upstream-assigned	Untrusted	Deny unsolicited inbound except explicit published/VPN services
LAN	192.168.10.1/24	Protected	Allow required outbound; inspect selected flows
DMZ	192.168.20.1/24	Semi-trusted	Publish only required services; deny DMZ-to-LAN by default
MGMT VLAN 10	192.168.30.1/24	Highly trusted	Allow firewall administration from authorized hosts
USER VLAN 20	192.168.40.1/24	Protected	Internet/application policy; limited server access
GUEST VLAN 30	192.168.50.1/24	Untrusted internal	Internet only; isolate from LAN/DMZ/MGMT

3.5 Traffic Flow and Security Layer

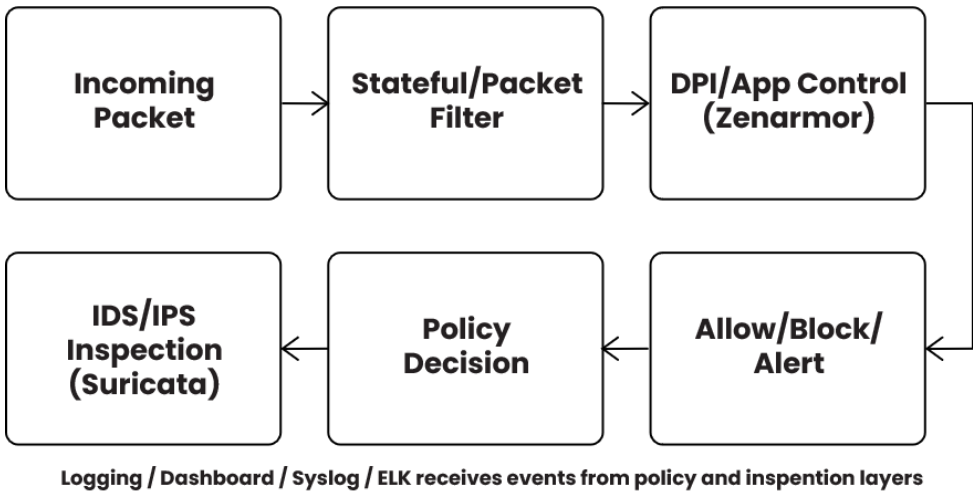


Fig. 3.3: Traffic Flow and Security Processing Pipeline

Traffic is first evaluated through stateful and packet-filtering policies, after which permitted traffic can undergo deeper inspection by Zenarmor for application control and Suricata for IDS/IPS analysis. Based on these checks, the system makes allow, block, or alert decisions while

generating logs for monitoring and analysis. The intended traffic flow and security-processing sequence are presented in **Fig. 3.3**.

3.6 Logging and Management Architecture

Logging is part of the security architecture rather than an afterthought. OPNsense provides firewall live view, system logs, VPN status, DNS information, and plugin-specific dashboards. Suricata produces alerts and event data, while Zenarmor provides application, web, session, and threat-oriented analytics. For a larger lab, these events can be forwarded through Syslog or integrated with an ELK-based platform for searching and correlation. Time synchronization is essential so that events from different sources can be compared accurately.

- Enable rule logging for critical allow and deny policies, but avoid logging every high-volume benign flow.
- Use descriptive rule names so log entries can be mapped back to policy intent.
- Retain Suricata alert fields such as signature, category, source/destination, and timestamp.
- Review Zenarmor application/web analytics for policy verification and unexpected traffic.
- Protect log servers from normal user access and ensure adequate storage for the selected retention period.

3.7 Design Rationale

The design deliberately avoids relying on a single inspection technology. Stateful firewall rules provide deterministic network segmentation and are fast. Suricata focuses on known attack patterns, protocol anomalies, and intrusion prevention. Zenarmor provides application context and web policy. VPN and authentication address secure access. Logging and analytics provide verification. If one layer misses an event, another layer may still detect or contain it. At the same time, the modular approach allows heavy features to be enabled only where their value justifies the performance cost.

CHAPTER 4
IMPLEMENTATION

4.1 Development Environment Setup

A typical laboratory can run on a modern multi-core workstation with at least 16 GB RAM so that the OPNsense VM, client VM, test server, and optional logging VM can operate simultaneously. OPNsense official guidance lists 3 GB RAM as a minimum and 8 GB RAM as a recommended profile for full-featured use, while noting that inspection and disk-intensive services increase resource requirements [1]. For a research build with Suricata and Zenarmor, allocating 4–8 GB RAM to the firewall VM and multiple virtual CPU cores is a practical starting point, subject to available hardware.

- Download the current OPNsense installation image from the official project.
- Create a VM with UEFI/BIOS settings compatible with the selected hypervisor.
- Assign at least three virtual NICs or use VLAN trunking where appropriate.
- Attach WAN to an upstream/NAT network, LAN to an isolated internal switch, and DMZ to a separate isolated switch.
- Install OPNsense and complete the console-based interface assignment before opening the web GUI.
- Take a clean hypervisor snapshot before adding IDS/IPS and DPI plugins.

4.2 OPNsense Installation and Network Interface Configuration

After installation, the administrator assigns physical or virtual NICs to WAN, LAN, and DMZ. The LAN interface becomes the initial management path. DHCP can be enabled for the protected LAN, while the DMZ usually uses static addressing for servers. DNS resolution can be provided by Unbound. Administrative access to the web interface should be restricted to the management network or selected administrator hosts.

VLANs are created on the parent LAN/trunk interface. Each VLAN is then assigned as an OPNsense interface, given an IP subnet, and provided with DHCP settings if needed. Firewall

rules are written on the source interface: for example, a GUEST rule allows DNS and Internet access while private-network destinations are denied; a MGMT rule permits HTTPS/SSH to the firewall and management endpoints; a USER rule allows only necessary internal services before general Internet access.

4.3 Stateful Packet Inspection (SPI) and Firewall Rules

OPNsense uses a stateful firewall model. When an outbound connection is permitted, the firewall creates state so that return traffic can pass without a separate broad inbound rule. This allows policy to remain concise while still controlling unsolicited connections. The rule base should follow least privilege: allow what is required, place more specific rules above general rules, add meaningful descriptions, and use aliases for groups of networks, hosts, and service ports. The baseline firewall rules matrix is shown in **Table 4.1**.

Table 4.1: Baseline Firewall Rule Matrix

Source Zone	Destination	Service	Action	Purpose
WAN	Firewall WAN	OpenVPN/WireGuard port	Pass	Remote access only
WAN	DMZ Web Server	TCP 443	Pass/NAT	Publish HTTPS test service
WAN	LAN networks	Any	Block	Protect internal network
LAN	Internet	HTTP/HTTPS/DNS + required	Pass	User access
LAN	DMZ Server	Required admin/app ports	Pass	Controlled management/use
DMZ	LAN	Any	Block	Contain compromised DMZ server
GUEST	Private networks	Any	Block	Guest isolation
GUEST	Internet	HTTP/HTTPS/DNS	Pass	Internet-only access
MGMT	Firewall interfaces	HTTPS/SSH as required	Pass	Administration

NAT is configured separately from security intent. Outbound NAT translates internal private addresses to the WAN address for Internet access. Port forwarding is used only where a DMZ service must be published. Each port forward should create or reference a matching firewall rule, and administrative services should not be published directly when a VPN can provide safer access.

4.4 Intrusion Detection and Prevention with Suricata

Suricata is enabled first in IDS mode so the team can monitor alerts without immediately blocking traffic. The selected interface must carry the target traffic, while checksum and offloading settings may need adjustment in virtual environments to ensure accurate packet inspection. Rules are downloaded from approved sources and enabled selectively to reduce unnecessary alerts and false positives. The recommended deployment settings, including mode, interfaces, rule sets, home networks, logging, blocking, and update procedures, are summarized in **Table 4.2**.

Table 4.2: Suricata Deployment Parameters

Parameter	Recommended Lab Approach	Reason
Mode	Start IDS, then move to IPS	Reduces risk of accidental outage
Interfaces	LAN/DMZ or selected internal path	Inspect client and server traffic
Rule set	Curated categories first	Avoid excessive false positives
Home networks	Define protected subnets	Improves rule context
Logging	Enable EVE/alert logging	Evidence and SIEM integration
Blocking	Enable only after tuning	Operational stability
Updates	Scheduled rule refresh	Maintain signature currency

For IPS validation, the team chooses known-safe test signatures or controlled attack patterns. A pass condition requires both an alert and the expected block behavior when the signature is configured to drop. If a legitimate connection is blocked, the event is analyzed by signature ID

and context before tuning. Disabling broad categories without investigation should be avoided because it can silently remove protection.

4.5 Deep Packet Inspection and Application Control with Zenarmor

Zenarmor is installed as an OPNsense plugin and applied to the intended interface or protected network to provide Layer-7 visibility and application control. It is introduced gradually so that common applications and web sessions can first be identified correctly before restrictive policies are enforced. Zenarmor documentation recommends maintaining proper Layer-4 OPNsense rules while using Zenarmor separately for Layer-7 application and web controls [3]. Example policies for business browsing, anonymizer blocking, risky category restriction, student/lab controls, and administrative exceptions are summarized in **Table 4.3**.

Table 4.3: Zenarmor Policy Examples

Policy Example	Match	Action	Expected Result
Business browsing	USER VLAN	Allow normal web categories	Standard browsing works
Block anonymizers	USER/GUEST	Block anonymizer/Tor category/application	Session denied and logged
Block risky category	GUEST	Block malware/phishing/risky categories	Request denied
Student/lab restriction	Selected VLAN	Block streaming/social category as policy requires	Category policy enforced
Admin exception	MGMT/admin user	Permit required management applications	Admin work unaffected

4.6 Web Filtering, Proxy, and TLS Inspection

Web filtering can be implemented through Zenarmor web controls and, where required, proxy/DNS controls. Zenarmor distinguishes web controls from general application controls: web controls focus on HTTP/HTTPS destinations and categories, while application controls classify applications more generally [4]. The research team should begin with category policy that does not require TLS decryption and verify that normal HTTPS browsing remains stable.

Full TLS inspection should be treated as an advanced option because it changes the trust relationship between clients and remote sites. The inspection engine becomes an authorized intermediary, which requires clients to trust an internal certificate authority. It can also break applications that use certificate pinning. Zenarmor's documentation advises maintaining bypasses for sensitive or incompatible destinations and notes privacy/legal responsibilities [8]-[9]. Therefore, the laboratory should demonstrate the mechanism only on test clients and test destinations, not on uncontrolled personal traffic.

4.7 Virtual Private Network (VPN)

A VPN provides secure remote access to selected internal resources without opening those resources directly to the Internet. Either OpenVPN or WireGuard can be used. The configuration should define a dedicated VPN tunnel network, user or key authentication, allowed internal routes, DNS behavior, and firewall rules on the VPN interface. Remote users should receive only the access necessary for their role; for example, an administrator VPN may reach management services while a general-user VPN may reach only selected applications.

- Create a dedicated VPN address pool that does not overlap LAN/DMZ/VLAN networks.
- Use strong keys/certificates and protect exported client configuration.
- Restrict the VPN interface with explicit rules rather than an unrestricted any-to-any policy.
- Record successful and failed authentication events.
- Test from an external network path to confirm that the tunnel works without exposing management ports publicly.

4.8 User Authentication and Captive Portal

For networks that require user-level control, OPNsense can authenticate against local users or an external directory such as LDAP/Active Directory. Captive portal is useful for guest, student, or controlled-access networks where users must authenticate before Internet access is granted. Authentication policy should be separated from network segmentation: a successful portal login should not automatically grant access to protected management or server VLANs.

4.8.1 Configuration Backup, Change Control and Recovery

Firewall configuration is security-critical infrastructure and should be managed with the same discipline as server configuration. Before enabling a new inspection engine, changing interface assignments, modifying VLAN trunks, or altering NAT rules, the current OPNsense configuration should be exported and labelled with the date and purpose of the change. A hypervisor snapshot can provide an additional recovery point in a laboratory, but the OPNsense configuration backup remains important because it can be restored to different hardware or a rebuilt virtual machine.

A simple change-control record should contain the rule or component changed, the reason for the change, the person performing it, the expected impact, the validation test, and the rollback method. This practice is especially useful when Suricata or Zenarmor policies appear to block legitimate traffic. Instead of making several unrecorded changes at once, the administrator can change one variable, repeat the test, and return to the previous configuration if the result is worse.

- Export OPNsense configuration before major changes.
- Keep named snapshots for milestone states such as baseline firewall, IDS tuned, IPS enabled, and DPI enabled.
- Document plugin versions and rule-set update dates used during final testing.
- Store VPN private keys and exported configurations securely and separately from the thesis document.
- Test restoration on a disposable lab VM so that backup procedures are verified rather than assumed.

4.8.2 Policy Verification and Troubleshooting Workflow

When a connection fails, troubleshooting should proceed from the simplest layer to the most complex. First verify interface status, addressing, routing, and DNS. Second check the OPNsense live firewall log to determine whether the Layer-4 rule base is blocking the flow. Third inspect Suricata alerts or drop events. Fourth review Zenarmor session and policy logs. Fifth check proxy, TLS, captive portal, or authentication behavior if those services apply. This order

prevents time being wasted inside an application-control engine when the real problem is a missing route or firewall rule.

Packet capture is also valuable because it shows whether traffic reaches an interface, whether a handshake completes, and whether a reset or timeout occurs. Captures should be limited to the necessary interface and filter expression to avoid collecting unrelated data. In a thesis lab, screenshots of the live log and a short packet-capture example can provide strong evidence that the observed allow/block result came from the intended control layer.

4.9 Real-Time Monitoring, Logging, and Implementation

The final implementation chapter contain evidence from the actual running firewall. The demo thesis template includes screenshots [**Fig. 4.1 – Fig. 4.6**] in the implementation section, and the supplied project requirement specifically calls for the OPNsense dashboard, Suricata alerts/rules, Zenarmor application/web analytics, and traffic/bandwidth graphs.

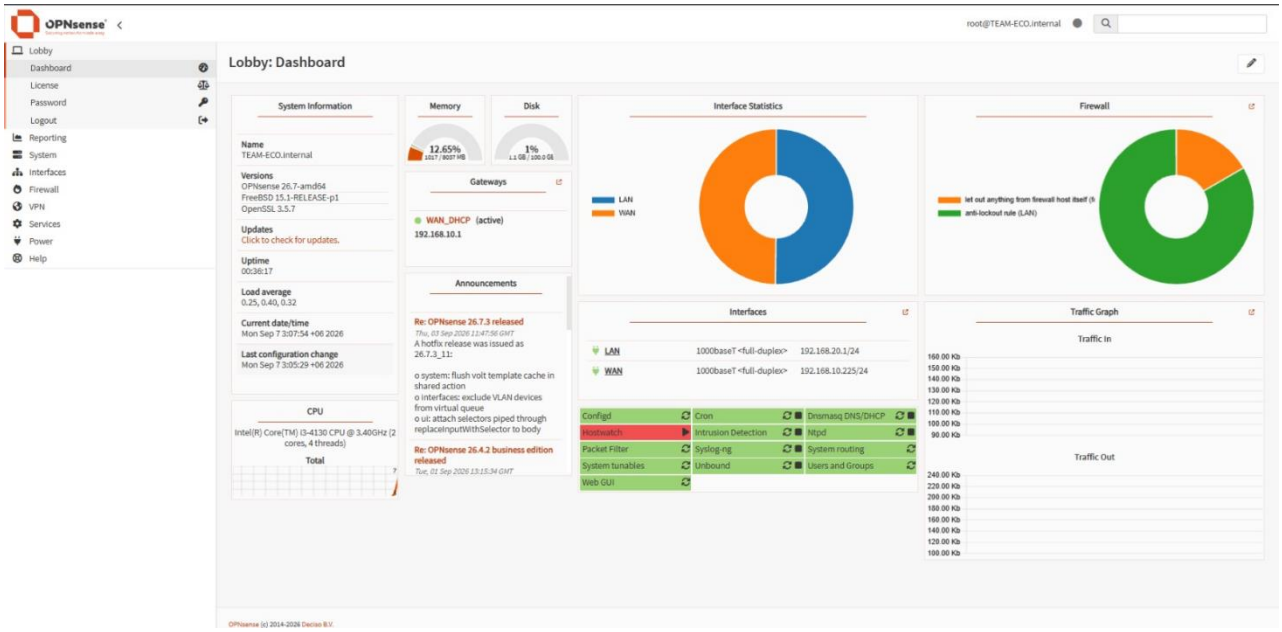


Fig. 4.1: OPNsense System Dashboard

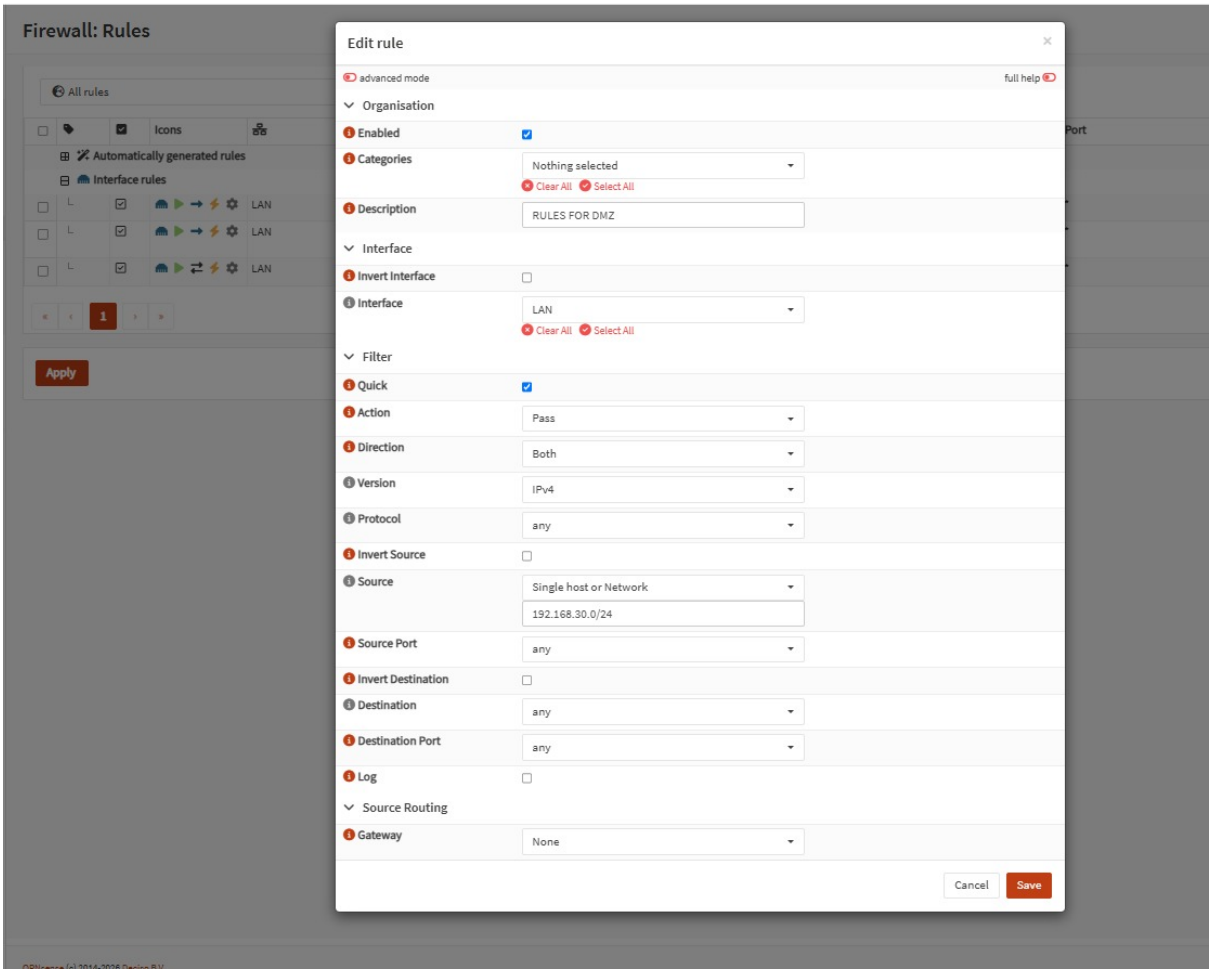


Fig. 4.2: Suricata IDS/IPS Rules and Alert Log

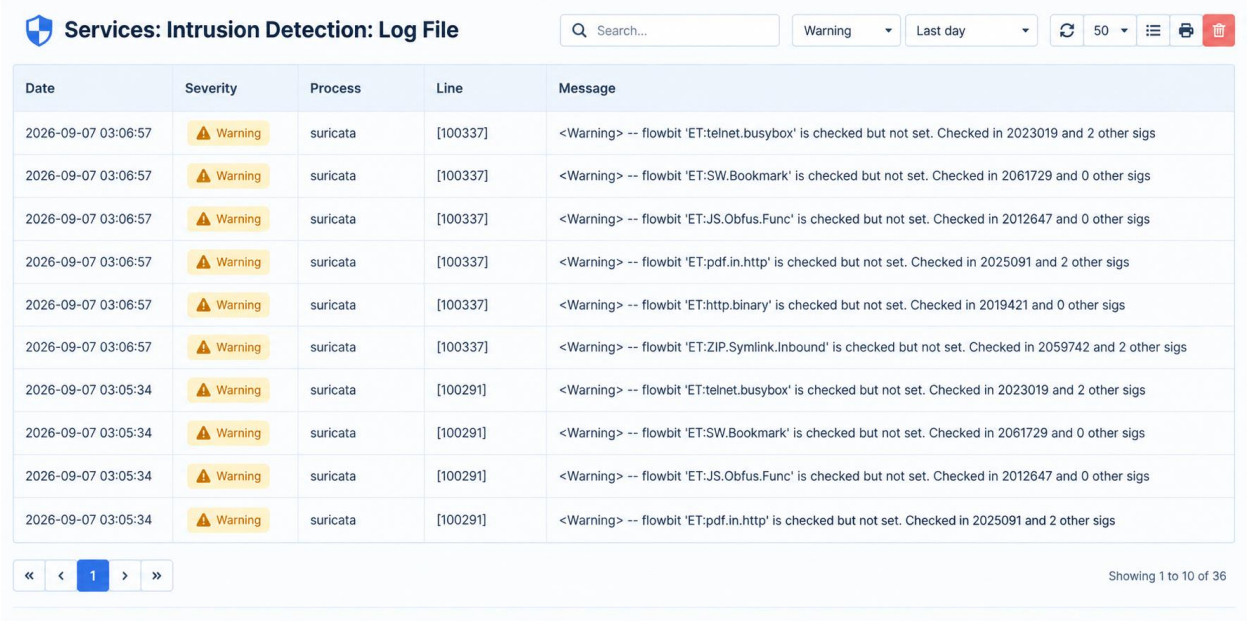


Fig. 4.3: Log File

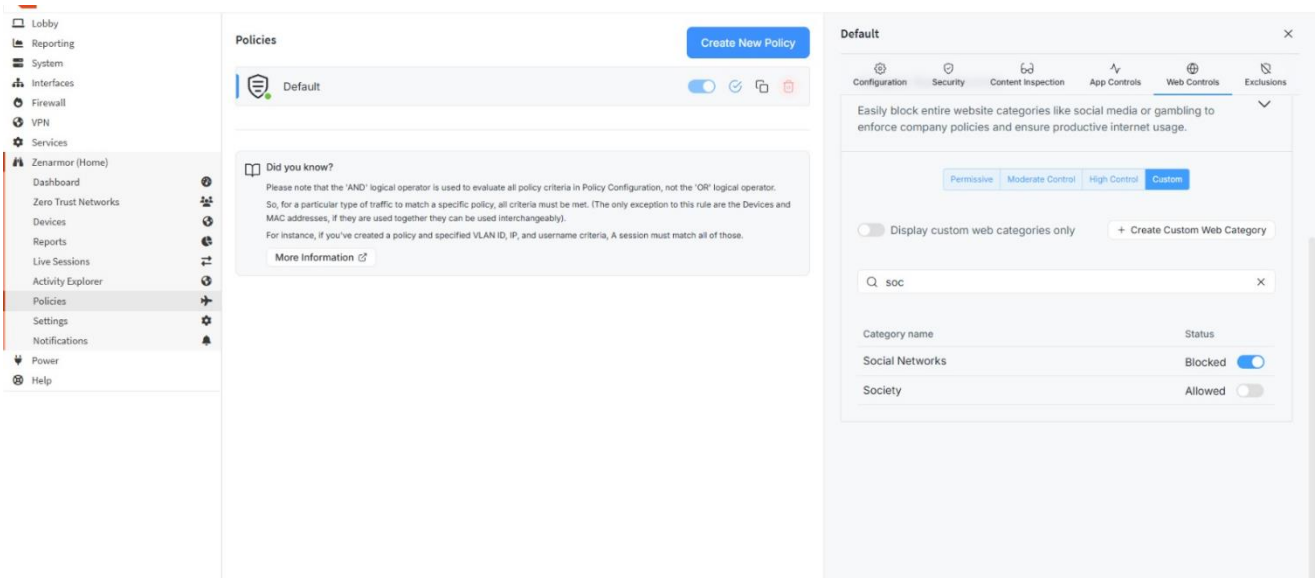


Fig. 4.4: Zenarmor Application Control and Web Category Analytics

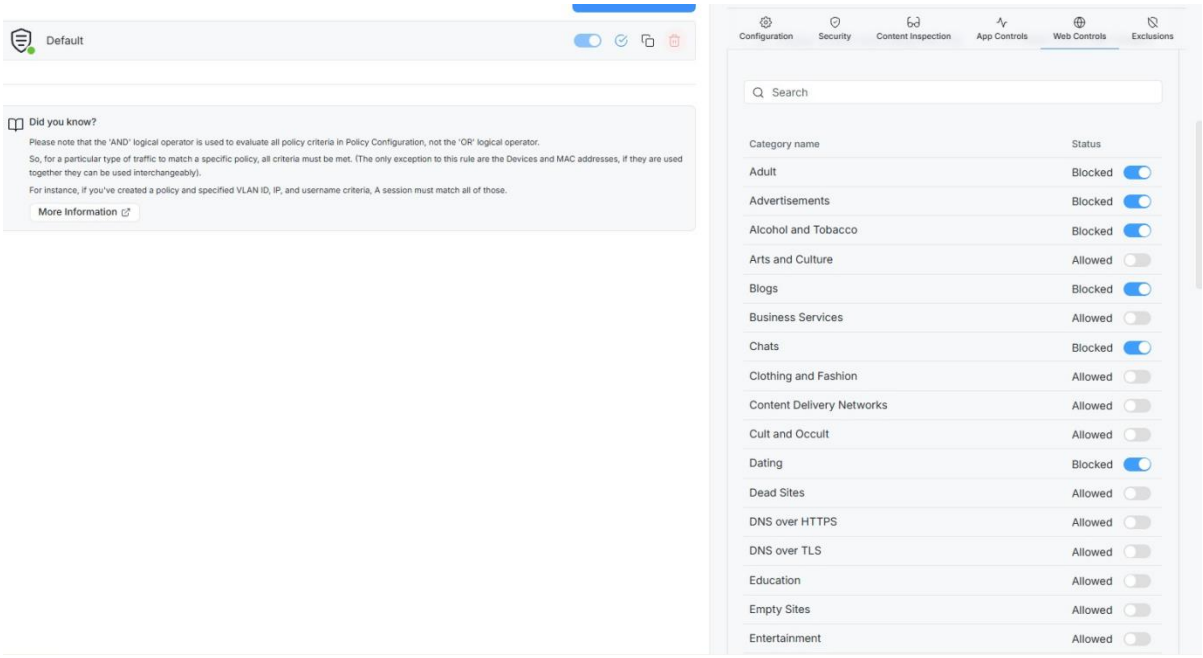


Fig. 4.5: Zenarmor Application Control and Web Category Analytics

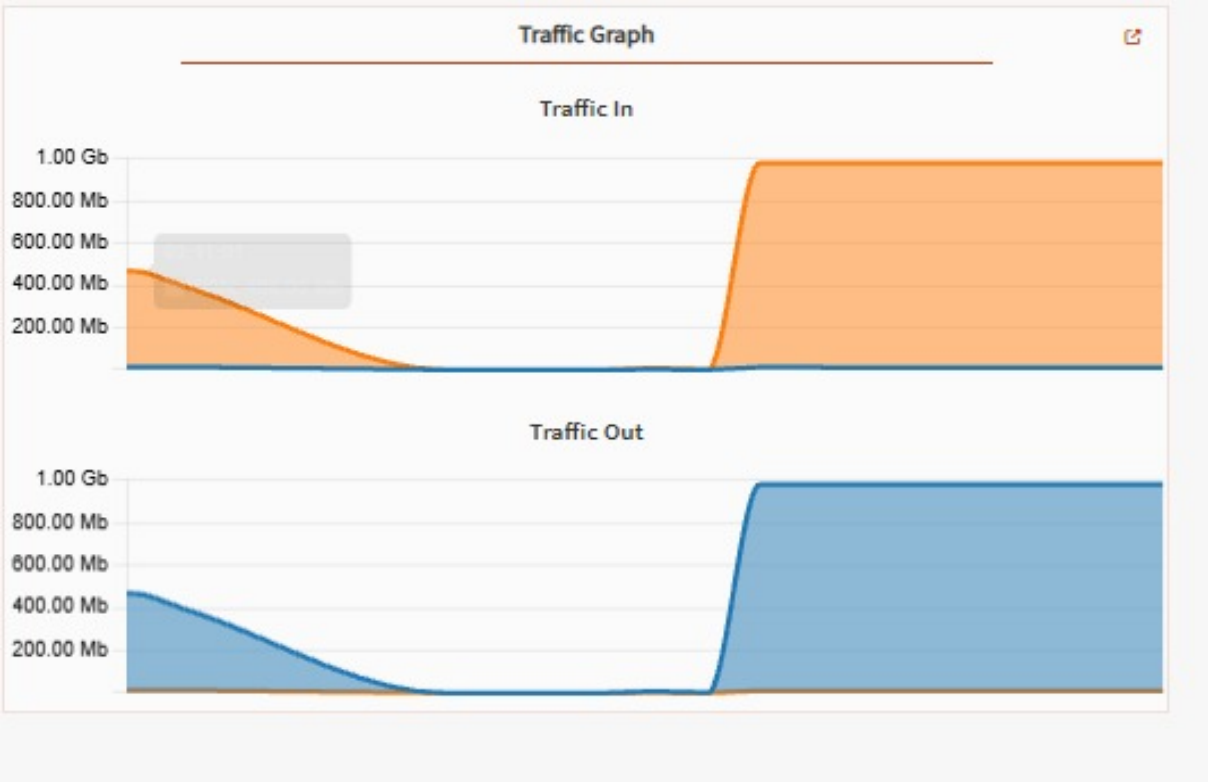


Fig. 4.6: Traffic / Bandwidth Usage Graph

CHAPTER 5
RESULTS AND DISCUSSION

5.1 Introduction and Test Plan

This chapter defines the evidence needed to determine whether the NGFW design meets the objectives stated in Chapter 1. A valid evaluation must show two things: first, that security policies behave correctly under benign and malicious traffic; second, that the performance impact of inspection is measured rather than assumed. The test plan therefore combines connectivity checks, attack simulations, logging verification, and throughput/latency measurement.

5.2 Penetration and Attack Simulation

Attack simulations are conducted only within the isolated laboratory to verify whether the configured NGFW controls can detect, block, and log defined security events. The validation covers WAN port scanning, SQL injection, XSS, blocked web categories and applications, DMZ-to-LAN access, and valid or invalid VPN login attempts. The test conditions, expected observations, and final laboratory results are summarized in **Table 5.1**.

Table 5.1: Attack Simulation Validation Matrix

Test	Precondition	Expected Observation	Final Lab Result
WAN port scan	Only VPN/HTTPS test service published	Unapproved ports filtered/closed	REPLACE WITH OBSERVED RESULT
DMZ SQLi test	IDS/IPS rules enabled	Suricata alert/block	REPLACE WITH OBSERVED RESULT
DMZ XSS test	Web test app available	Alert/block depending on rule	REPLACE WITH OBSERVED RESULT
Blocked web category	Zenarmor policy active	Block page/session event	REPLACE WITH OBSERVED RESULT
Blocked application	Application control active	Session denied and classified	REPLACE WITH OBSERVED RESULT
DMZ-to-LAN attempt	Default deny DMZ→LAN	Firewall log shows deny	REPLACE WITH OBSERVED RESULT

VPN login	Valid user/key	Tunnel established and limited routes	REPLACE WITH OBSERVED RESULT
Invalid VPN login	Invalid credentials	Authentication rejected/logged	REPLACE WITH OBSERVED RESULT

5.3 Performance and Throughput Benchmarking

Performance benchmarking should use the same client/server path for every configuration state. A baseline measurement is taken with OPNsense routing/firewalling only. The test is repeated with Suricata IDS enabled, Suricata IPS enabled, Zenarmor DPI enabled, and finally the combined inspection stack. Each state should be tested multiple times with the same packet size, number of parallel streams, duration, and virtual CPU allocation. Latency can be recorded with ping or a similar tool, while throughput can be measured using iperf3.

5.4 Detection Accuracy and Policy Validation

Detection accuracy and policy validation are assessed by examining true positives, false positives, and false negatives across the defined security tests. The objective is to confirm that each NGFW capability operates according to the expected policy rather than to claim industry-scale statistical accuracy. The validation methods, pass criteria, and verification status for the firewall, network segmentation, Suricata, Zenarmor, web filtering, VPN, captive portal, and logging are summarized in **Table 5.2**.

Table 5.2: Security Feature Validation Summary

Capability	Validation Method	Pass Criterion	Status
Stateful firewall	Allowed/denied connection matrix	All required paths match policy	TO VERIFY
Network segmentation	Cross-zone access tests	No unauthorized LAN/DMZ/VLAN access	TO VERIFY
Suricata IDS	Controlled signature triggers	Expected alerts generated	TO VERIFY
Suricata IPS	Same tests in inline mode	Configured malicious flow blocked	TO VERIFY
Zenarmor DPI	Known applications	Correct classification and policy action	TO VERIFY

Web filtering	Allowed + blocked categories	Policy applied with evidence	TO VERIFY
VPN	Valid/invalid remote users	Authorized access only	TO VERIFY
Captive portal/auth	Authenticated vs unauthenticated client	Access follows policy	TO VERIFY
Logging	Review firewall/Suricata/Zenarmor events	Evidence available with timestamps	TO VERIFY

5.5 Discussion: Security Effectiveness

The architecture is strongest when each layer has a narrow, clear responsibility. Stateful rules and segmentation reduce the volume of traffic reaching higher-cost inspection. Suricata provides threat-oriented signatures and anomaly visibility. Zenarmor adds application and web context that ordinary port-based rules cannot provide. VPN and authentication reduce the need to expose management services. Logging connects these layers by showing whether the expected policy actually occurred.

A major operational issue is false-positive control. Enabling every available IDS/IPS rule can create alert fatigue or accidental blocking, especially in a university lab with varied tools and protocols. Tuning should therefore be evidence-based: identify the exact signature, confirm whether the event is benign, narrow the rule scope if possible, and document the exception. This preserves security value better than disabling entire categories.

TLS inspection illustrates the difference between technical capability and good policy. Decryption can improve visibility into encrypted sessions, but it increases CPU cost, changes the trust model, and can conflict with privacy expectations and certificate pinning. A mature NGFW design therefore applies selective inspection based on organizational need and legal authorization rather than assuming that maximum inspection is always best.

5.6 Discussion: Performance, Manageability and Cost

The expected performance trend is that routing and basic stateful filtering provide the highest throughput because these functions require comparatively less processing. When IPS, DPI, application control, logging, and other inspection features are enabled, available throughput is likely to decrease as the firewall performs additional packet analysis and policy checks. The

important result is not simply whether throughput decreases—it almost certainly will—but whether the measured performance remains acceptable for the intended number of users, traffic volume, and Internet link speed. Latency and CPU utilization should also be considered alongside throughput because they provide a more complete picture of system performance. For this reason, hardware sizing should be based on the security features that will actually be enabled and the expected real-world workload rather than relying only on vendor minimum specifications.

Manageability is improved by using aliases, descriptive firewall rules, clearly defined security zones, consistent naming conventions, and staged feature deployment. These practices make the configuration easier to understand, maintain, and troubleshoot. When multiple security engines are enabled at the same time, troubleshooting becomes more difficult because a blocked connection may originate from the firewall rule set, Suricata, Zenarmor, DNS filtering, authentication controls, or another policy layer. The implementation procedure therefore enables and validates one layer at a time so that problems can be isolated more accurately. Centralized logging and regular review of alerts can further improve operational visibility and help administrators identify which component is responsible for a particular event.

From a cost perspective, OPNsense and Suricata provide a strong open-source foundation that can significantly reduce software licensing expenses. Zenarmor adds important Layer-7 and application-aware capabilities, although some advanced features depend on the selected subscription tier. However, an open-source approach does not mean that the overall system is free of cost. Suitable hardware, storage, administration time, monitoring, backups, upgrades, testing, staff training, and possible commercial subscriptions may still be required. Additional costs may also arise when higher network speeds require more powerful processors, memory, or dedicated hardware.

The main advantage of this approach is flexibility, transparency, and the ability to build a research-grade NGFW architecture using commodity or virtual infrastructure. It also allows individual security components to be tested, modified, and scaled according to organizational requirements. Therefore, the final evaluation should consider not only security effectiveness, but also whether the system provides an acceptable balance between performance, manageability, and total operational cost.

5.6.1 Quantitative Evaluation Metrics

The performance analysis can be expressed using a set of simple quantitative metrics so that the effect of each enabled security layer can be compared consistently. Throughput impact may be reported as the percentage reduction from the firewall-only baseline:

Throughput Reduction (%) = ((Baseline Throughput − Inspected Throughput) / Baseline Throughput) × 100.

This metric shows how much network capacity is lost after enabling additional inspection features such as IDS/IPS or DPI. Latency overhead may similarly be calculated as:

Latency Overhead = Inspected Latency − Baseline Latency

CPU and RAM utilization should be sampled during the same testing period so that the research team can determine whether a reduction in throughput is associated with processor or memory saturation, or whether it is caused by another bottleneck in the virtualized network environment. Where possible, repeated measurements should be taken and average values reported to reduce the influence of temporary fluctuations.

For security detection tests, the team can report ***Precision = TP / (TP + FP)*** and Detection Rate ***Recall = TP / (TP + FN),*** where TP represents a malicious test correctly detected by the security system, FP represents benign traffic incorrectly identified as malicious, and FN represents a malicious test that was not detected. These metrics provide a more objective way to evaluate detection quality and false-positive behavior.

In a small and controlled experimental dataset, these values should be interpreted cautiously because a limited number of tests may not represent all real-world attack conditions. However, they still provide a clearer and more reproducible summary than general statements such as “the firewall detected most attacks.” Therefore, the exact number, type, and conditions of each security test should be listed alongside the calculated metrics so that the results can be understood and reproduced accurately.

5.6.2 Operational Security Considerations

A functioning NGFW requires continuous operational attention even after the initial installation and configuration are completed. Security signatures, threat intelligence feeds, and application databases should be updated regularly so that the system can recognize newly emerging threats and applications. Firewall rules should also be reviewed periodically to identify obsolete entries,

temporary exceptions, duplicate policies, or unnecessary access permissions. VPN accounts and credentials must be disabled or revoked when users no longer require remote access, while log storage should be monitored to prevent excessive disk usage. Administrators should also regularly evaluate whether blocked website categories, applications, or access restrictions are still relevant to organizational requirements. Security policies that are never reviewed often accumulate exceptions over time and eventually become difficult to manage, audit, and understand.

The NGFW platform itself must also be treated as a critical security asset. The management interface should use HTTPS and should preferably be accessible only from authorized administrative hosts or management networks. Administrator accounts should use strong, unique passwords and multi-factor authentication where supported. Default credentials should never remain active, and unnecessary administrative accounts should be removed. Configuration backups should be created regularly, stored securely, and protected against unauthorized modification or disclosure. Similarly, plugins, packages, and updates should only be obtained from trusted and verified sources to reduce the risk of introducing malicious or unstable software.

System health and resource utilization also require continuous monitoring. CPU usage, memory consumption, storage capacity, interface statistics, and firewall resource graphs should be reviewed regularly, particularly after enabling resource-intensive features such as IPS, DPI, or extensive logging. An overloaded security gateway can become a network availability problem even when its firewall policies are technically correct. Therefore, effective NGFW operation depends not only on strong initial configuration but also on regular maintenance, monitoring, policy review, secure administration, timely updates, and proper capacity management.

5.7 Limitations of the Evaluation

A virtual laboratory cannot reproduce every technical, operational, and traffic condition that may exist in a real production network. Factors such as hypervisor scheduling, virtual NIC behavior, host CPU contention, shared memory resources, limited traffic diversity, and relatively small test datasets may influence the measured results. Performance findings from the laboratory therefore may not directly represent the throughput, latency, or resource utilization that would be observed on dedicated physical hardware or in a larger enterprise environment.

The security evaluation also has important limitations. Signature-based detection tests can demonstrate whether known attack patterns are identified, but they do not prove protection against unknown attacks, zero-day vulnerabilities, advanced evasion techniques, or highly customized malware. Similarly, a limited number of simulated attacks cannot represent the full range of threats that may appear in real-world network traffic.

For these reasons, this thesis treats the laboratory environment as a controlled proof of design, configuration behavior, policy enforcement, and relative performance rather than as a formal certification that the system can prevent every possible cyber threat. The findings are intended to show how the selected NGFW components behave under defined and repeatable test conditions.

A production deployment would require additional activities such as longer-term monitoring, formal change management, broader user testing, backup and restore exercises, high-availability planning, hardware capacity assessment, incident-response procedures, periodic security audits, and continuous rule and signature updates. These measures would be necessary to validate the architecture under real operational conditions and ensure that the system remains secure, stable, and manageable over time.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

This thesis presents a complete design framework for an open-source Next-Generation Firewall using OPNsense as the central security gateway. The design begins with the functions that remain fundamental to every firewall—routing, stateful packet inspection, NAT, and segmentation—and then extends them with Suricata IDS/IPS, Zenarmor deep packet inspection and application/web control, VPN, authentication, and structured logging. The WAN/LAN/DMZ/VLAN topology provides clear security boundaries, while the layered architecture allows the most expensive inspection features to be applied where they provide the greatest value.

The proposed implementation process is intentionally incremental. Basic connectivity is established first; segmentation and stateful rules are validated before IDS/IPS; IDS is tuned before IPS blocking; DPI is introduced after the Layer-4 policy is stable; and VPN, authentication, logging, and optional TLS inspection are added only after the core gateway is reliable. This sequence reduces troubleshooting complexity and makes the laboratory reproducible.

The evaluation framework connects configuration to evidence. Port scans, DMZ web-attack simulations, application/web-policy tests, VPN authentication tests, and cross-zone access attempts verify security behavior, while iperf3 and latency measurements quantify the performance cost of inspection. The thesis also explicitly avoids inventing implementation evidence: the final screenshots and benchmark values must come from the researchers' actual lab. Once those marked items are replaced, the manuscript will provide a complete record of the design, implementation, validation, limitations, and practical recommendations of the NGFW prototype.

6.2 Future Work

- High availability using two OPNsense nodes with state/configuration synchronization.
- Automated threat-intelligence feeds and dynamic block lists with careful false-positive controls.
- Extended ELK/SIEM integration for cross-source correlation and dashboards.
- Long-duration performance testing with mixed traffic profiles and higher-speed NICs.
- Comparison against a commercial NGFW using the same test topology and traffic set.
- Integration with 802.1X, RADIUS, and identity-aware access policies.
- Automated configuration compliance checks and infrastructure-as-code style backup/versioning.
- Advanced encrypted-traffic analysis that minimizes content decryption while preserving useful security metadata.
- Use of additional network-monitoring tools such as Zeek for behavioral visibility and forensic enrichment.

REFERENCES

[1] OPNsense, “Hardware sizing & setup,” OPNsense Documentation, accessed Sep. 2026.

[2] OPNsense, “Virtual & Cloud-Based Installation,” OPNsense Documentation, accessed Sep. 2026.

[3] Zenarmor, “How does the engine work? Relationship between Zenarmor and OPNsense/L4 firewalls,” Zenarmor Documentation, accessed Sep. 2026.

[4] Zenarmor, “Application Control Rules on OPNsense,” Zenarmor Documentation, accessed Sep. 2026.

[5] Zenarmor, “Web Control Rules on OPNsense,” Zenarmor Documentation, accessed Sep. 2026.

[6] Open Information Security Foundation, “Suricata User Guide,” Suricata Documentation, accessed Sep. 2026.

[7] OPNsense, “Intrusion Detection,” OPNsense Documentation, accessed Sep. 2026.

[8] Zenarmor, “Managing TLS Inspection on OPNsense,” Zenarmor Documentation, Apr. 2024, accessed Sep. 2026.

[9] Zenarmor, “TLS Inspection Rules on OPNsense,” Zenarmor Documentation, Dec. 2025, accessed Sep. 2026.

[10] M. Roesch, “Snort: Lightweight Intrusion Detection for Networks,” in Proc. LISA, 1999.

[11] Canadian Institute for Cybersecurity, “CIC-IDS2017 Dataset,” University of New Brunswick, 2017.

[12] NIST, Guidelines on Firewalls and Firewall Policy, Special Publication 800-41 Rev. 1, National Institute of Standards and Technology, 2009.

[13] NIST, Zero Trust Architecture, Special Publication 800-207, National Institute of Standards and Technology, 2020.

[14] The FreeBSD Project, “FreeBSD Documentation Portal,” accessed Sep. 2026.

[15] WireGuard, “WireGuard: fast, modern, secure VPN tunnel,” official documentation, accessed Sep. 2026.

[16] OpenVPN Inc., “OpenVPN Community Documentation,” accessed Sep. 2026.