Smart Unified Threat Management System (SUTMS)

Blessy Queen Mary M, Aswin S, Janardhan M, Praghadeesh S, Iyappan K.

Department of Information Technology, Government College of Technology, Coimbatore, India.

Email: blessy.stanley123@gmail.com, aswi.71772218102@ gct.ac.in , jana.71772218116@ gct.ac.in , prag.71772218138@gct.ac.in, iyap.71772218l04@gct.ac.in.

Abstract - The rapid increase in smart home devices has made household networks more exposed to cyber-attacks, while most advanced security solutions remain expensive or resource-intensive for everyday users. This work presents a lightweight, Raspberry Pi 5–based Unified Threat Management System (SUTMS) designed specifically for home networks, combining intrusion detection, flow monitoring, threat-intelligence aggregation, and automated rule optimisation in a single compact setup. A key feature of the system is a custom script that extracts active protocols from NTOPng and dynamically tunes Suricata's rule set, significantly reducing unnecessary processing overhead. To strengthen real-time intelligence, a custom TAXII server built using OpenTAXII is deployed to aggregate Indicators of Compromise from multiple providers, enabling adaptive firewall blocking. The system is implemented as an inline security gateway with a minimal DHCP server and IP-forwarding configuration, supported by a Flask-based monitoring dashboard and Webmin for device management. The prototype was deployed fully on a Raspberry Pi 5 and tested with real home traffic, showing notable improvements in resource usage while maintaining reliable detection capabilities. Overall, this work demonstrates a practical and accessible approach to home network security, offering a security architecture that is easy to deploy, resource-friendly, and adaptable to evolving threats.

Index Terms - Unified Threat Management (UTM), Intrusion Detection System (IDS), Suricata, NTOPng, Protocol Extraction, Rule Optimization, Threat Intelligence, TAXII Server, OpenTAXII, Raspberry Pi 5, Home Network Security, Inline Deployment, DHCP Server, Network Monitoring, Flow Analysis, Firewall Automation, Lightweight Security Framework.

I. INTRODUCTION

A. BACKGROUND

The modern home network has evolved far beyond a simple collection of personal computers. Today, most households rely on a diverse set of internet-connected devices—smart TVs, voice assistants, surveillance cameras, thermostats, and other IoT appliances—that remain online around the clock. While these devices offer convenience, they also introduce multiple entry points for attackers, often without the user's awareness. Many consumer-grade routers provide only basic security features such as simple firewall rules or URL filtering, leaving the network largely unprotected against sophisticated threats.

At the same time, threat activity targeting home environments has steadily increased. Malware campaigns, automated scanning tools, botnet recruitment, and attempts to exploit IoT vulnerabilities frequently originate from the internet and can compromise improperly configured or unpatched devices. Unlike enterprise networks, most households do not deploy dedicated intrusion detection systems, traffic monitoring tools, or advanced threat-intelligence feeds because these solutions tend to be expensive, complex, or resource-intensive.

This gap has encouraged researchers and hobbyists to experiment with affordable hardware platforms that can host lightweight yet effective security controls. Raspberry Pi devices, in particular, have become popular due to their low cost, small footprint, and surprisingly capable processing performance. When combined with open-source tools like Suricata, NTOPng, and modern threat-intelligence frameworks, such devices can bring enterprise-grade monitoring within reach of everyday users. However, integrating these tools into a coherent and efficient security system for home networks remains a significant challenge, mainly due to the resource limitations of embedded platforms and the dynamic nature of home traffic patterns.

These factors highlight the need for a unified, accessible, and resource-aware security approach specifically tailored for home environments. Such a solution should combine real-time monitoring, intrusion detection, and adaptive threat-intelligence usage while staying lightweight enough to run reliably on low-power hardware.

B. PROBLEM DEFINITION

Home networks today operate with a level of complexity that was once limited to small enterprises. Multiple devices continuously generate traffic across different protocols, and many of these devices lack proper security mechanisms. Yet most households still rely on basic router-level protections that offer little defence against modern threats. Deploying advanced tools such as Suricata, NTOPng, and threat-intelligence feeds can greatly improve visibility and security, but these tools are not designed to run efficiently on low-power systems like a Raspberry Pi without considerable tuning.

The core challenge lies in combining several resource-intensive security functions—deep packet inspection, traffic analysis, firewall enforcement, and threat-intelligence ingestion—into a single, reliable platform that can operate transparently within a home network. Suricata itself ships with thousands of detection rules, many of which are unnecessary for home environments and significantly increase CPU and memory usage when left enabled. Without a mechanism to adaptively load only the rules relevant to the network's active protocols, the system quickly becomes inefficient and unstable on embedded hardware.

Another aspect of the problem is the absence of a unified method to aggregate threat-intelligence from multiple TAXII-compatible providers in a way that can be easily applied to home network firewalls. Existing STIX/TAXII servers are often designed for enterprise deployments and require substantial resources or configuration overhead, making them impractical for simple home setups.

In summary, the problem addressed in this work is the lack of an integrated, lightweight, and adaptive security system that can run inline on a Raspberry Pi, intelligently optimize IDS behaviour based on live traffic, and incorporate real-time threat intelligence—all while remaining easy to deploy and manage within a typical household network.

C. PROBLEM DEFINITION

Although several open-source security tools exist for network monitoring and intrusion detection, their effective use in home environments is still limited. Most studies and implementations focus on enterprise-grade deployments, where hardware resources, network bandwidth, and administrative expertise are significantly higher than what is available in a typical household. As a result, solutions proposed in literature rarely address the constraints of low-power hardware or the practical challenges of running continuous deep packet inspection on embedded devices like the Raspberry Pi.

Similarly, existing work on Suricata-based IDS deployments largely assumes that the full rule set can be enabled or that the system will run on machines capable of handling the resulting load. Very few attempts have been made to dynamically tailor Suricata's rule base by analysing real, live traffic patterns—something essential for maintaining performance on constrained devices. There is also limited research on combining flow-level protocol detection with automated rule optimisation, even though such an approach can significantly reduce resource usage without compromising detection capability.

Another noticeable limitation in current research is the lack of practical threat-intelligence integration for home networks. While the STIX/TAXII ecosystem is well explored in enterprise and SOC environments, lightweight and customisable TAXII servers suitable for small-scale setups are rarely discussed. Most home-network studies stop at local IDS analysis and do not incorporate external Indicators of Compromise to improve proactive blocking.

Finally, there is a gap in literature concerning fully integrated, inline home-security gateways that combine monitoring, adaptive IDS behaviour, threat-intelligence processing, DHCP services, and user-friendly dashboards in a single setup. Existing works typically address these components in isolation rather than as a unified security solution.

These gaps highlight the need for a compact, adaptive, and resource-efficient framework that brings enterprise-like protection to home networks, without demanding specialized hardware or excessive configuration effort.

D. CONTIBUTIONS AND STRUCTURE

The key contributions and the overall structure of this applied research article are summarized as follows:

A lightweight, inline UTM architecture tailored for home networks is designed and implemented, integrating intrusion detection, flow monitoring, adaptive firewalling, and lightweight DHCP services within a Raspberry Pi 5 gateway. Multiple deployment strategies were examined to ensure seamless operation across different home network layouts, including router-behind-Pi and Pi-as-primary-gateway models (Section III).

A coordinated workflow is established across all major SUTMS components, enabling NTOPng, Suricata, the custom TAXII server, and the firewall engine to operate as a unified system. The methodology outlines how traffic analysis, protocol extraction, IDS tuning, and threat-intelligence ingestion work together to maintain low resource consumption while preserving strong security coverage (Section IV).

The firewall is extended from a static rule-based filter to an automated threat-intelligence-driven enforcement module. Using a self-hosted OpenTAXII server, Indicators of Compromise (IoCs) are periodically aggregated from multiple sources, normalized, and pushed into IPTables. This allows the system to automatically block malicious IPs and C&C communication without user intervention (Section IV-C).

Suricata's IDS performance is optimized through a protocol-aware rule selection mechanism, where NTOPng's detected protocols are used to dynamically enable only the relevant Suricata rule families. This significantly reduces CPU load, minimizes false positives, and allows Suricata to operate reliably on Raspberry Pi hardware (Section IV-B).

Comprehensive evaluation and real-world validation of the system are performed, focusing on resource usage, detection capability, responsiveness of the rule-optimization pipeline, and the effectiveness of the threat-intelligence-driven firewall. The system is tested with live home network traffic to ensure the design remains practical for everyday use (Section V).

The remainder of this paper follows this structure and builds upon the above contributions.

II. RELATED WORK

Research on home network security has gained attention over the past decade, mainly due to the rapid adoption of IoT devices and the growing number of attacks targeting residential environments. Traditional intrusion detection systems such as Snort and Suricata have long been used in enterprise networks, but their deployment in home setups has been limited by resource constraints and the lack of simplified configuration workflows. Several studies have explored lightweight IDS solutions for embedded devices, demonstrating that Raspberry Pi–class hardware can support packet inspection under controlled conditions. However, these works often rely on default or trimmed-down rule sets and do not address the broader challenge of dynamically adapting IDS behaviour to the protocols actually present in the network.

Flow monitoring frameworks such as NTOPng and Zeek have been widely used in academic and operational settings for traffic analysis and protocol fingerprinting. Prior work has shown that flow-based detection can reduce overhead compared to full deep packet inspection, yet few studies integrate flow detection with automated IDS rule optimisation. Most existing systems treat flow monitoring and signature-based detection as independent modules rather than complementary components that can enhance each other's performance.

Unified Threat Management (UTM) solutions have also been discussed in literature, primarily in the context of enterprise firewalls and commercial appliances. These systems typically combine IDS, IPS, antivirus, and policy enforcement into a single platform. While several open-source UTM distributions exist, they are generally not optimized for low-power devices and often require multi-core systems to operate reliably. Very little research examines how UTM concepts can be scaled down for home networks or how individual components can be tuned to work efficiently within the limited resources of a small embedded device.

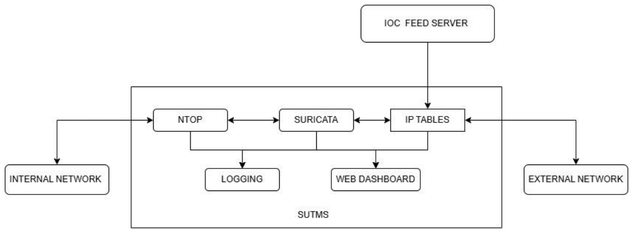
Threat-intelligence sharing through STIX/TAXII has become a key component in modern security operations. Prior works have focused on improving the interoperability and structure of threat feeds, but nearly all implementations assume server-grade hardware and enterprise SOC environments. Lightweight TAXII servers suitable for home deployments receive minimal attention, and there is sparse literature describing how aggregated IoCs can be incorporated directly into local firewalls for automated blocking.

Raspberry Pi devices have been explored as affordable security platforms in several experimental studies. Projects typically highlight their low cost, simple deployment, and energy efficiency, but they often encounter limitations when running multiple security components simultaneously. Most prior implementations either restrict functionality to basic packet filtering or rely on reduced IDS configurations that sacrifice detection depth.

Overall, while existing research covers individual components such as IDS optimisation, flow monitoring, or threat-intelligence processing, there is limited work on combining all these elements into a coherent, adaptive, and resource-efficient system specifically tailored for home networks. The lack of integration between flow analysis, dynamic IDS rule management, and threat-intelligence-driven firewalling represents a notable gap that this work aims to address.

III. SYSTEM OVERVIEW

A. OVERVIEW OF THE PROPOSED SUTMS FRAMEWORK



The proposed Secure Unified Threat Management System (SUTMS) is designed as a compact, adaptive security layer for home networks, built entirely on a Raspberry Pi 5 functioning as an inline gateway. Unlike conventional consumer routers that offer only basic filtering and limited visibility, SUTMS integrates multiple open-source security tools into a coordinated framework capable of monitoring, detecting, and responding to threats in real time. The goal of the system is to provide enterprise-grade security features in a lightweight and accessible form that suits the constraints of household environments.

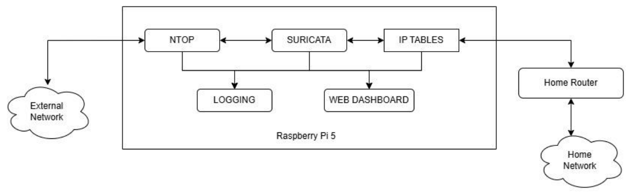
At its core, SUTMS unifies three major security functions: flow-based monitoring, signature-based intrusion detection, and threat-intelligence-driven firewalling. These functions are supported by a management layer consisting of a minimal Flask dashboard and Webmin, making the system usable even for non-technical users. Each module contributes a distinct capability, but more importantly, the system is designed so that these modules operate collaboratively, exchanging information to reduce resource consumption and improve overall detection efficiency.

Traffic entering or leaving the home network passes directly through the Raspberry Pi, allowing SUTMS to observe every packet without requiring port mirroring or additional hardware. NTOPng continuously monitors this traffic and extracts protocol information, which becomes the basis for adapting the behaviour of Suricata, the IDS component. Instead of loading the entire default rule set, Suricata is configured dynamically to enable only those rule families relevant to the protocols actually present in the network. This significantly lowers CPU usage, which is crucial when running deep packet inspection on a resource-constrained device.

Parallel to this, SUTMS incorporates a custom threat-intelligence pipeline driven by a self-hosted OpenTAXII server. The TAXII server aggregates Indicators of Compromise (IoCs) from multiple sources, creating a consolidated pool of malicious IP addresses and domains. The system periodically retrieves these IoCs and automatically applies them to the IPTables firewall, allowing the gateway to block connections to known command-and-control servers without manual intervention.

Together, these components form an integrated security framework that adapts to the behaviour of the home network, optimizes resource usage, and enhances protection against external threats. By combining monitoring, detection, and automated enforcement in a single device, SUTMS provides a practical solution for households that require stronger security than what traditional home routers can provide.

B. DEPLOYMENT ARCHITECTURE



The SUTMS is deployed as an inline gateway positioned directly between the home router and the external network connection. This placement ensures complete visibility into all inbound and outbound traffic without the need for additional TAP devices or switch port mirroring. The Raspberry Pi 5 forms the central processing point of the architecture, operating with two network interfaces—one dedicated to the WAN side and the other connected to the internal home network. All packets passing through this path are subject to monitoring, intrusion detection, and policy enforcement before reaching their destination.

To support this gateway role, the Raspberry Pi is configured to handle core network functions typically performed by a router. A minimal DHCP server is used to allocate IP addresses to devices within the home network, while IP forwarding and NAT rules ensure transparent routing between the internal and external interfaces. This design keeps the deployment simple and avoids complex reconfigurations on the user’s existing router; the home router essentially acts as an access point, allowing SUTMS to maintain full visibility over device traffic.

The system follows a layered processing model: packets first enter the forwarding plane, where NTOPng observes flow characteristics and extracts protocol metadata. The traffic then moves through Suricata for deep packet inspection based on the dynamically optimized rule set. Firewall actions are applied at the final stage using IPTables, which integrates both locally generated policies and threat-intelligence-based blocklists that are periodically updated from the custom TAXII server.

This deployment strategy brings several advantages. Operating inline ensures that the system does not miss encrypted or transient flows that often escape passive monitoring setups. The use of a Raspberry Pi 5 offers a balance between performance and affordability, enabling the deployment of UTM functionalities in an environment where resource constraints are a key concern. Additionally, the system remains modular, meaning that NTOPng, Suricata, TAXII ingestion, and the firewall can operate independently while still communicating through well-defined data flows.

Overall, the deployment architecture provides a practical and effective layout for home networks, offering comprehensive security capabilities without requiring specialized hardware or complex network restructuring. The inline design, combined with lightweight network services and automated enforcement mechanisms, forms a solid foundation for delivering real-time protection in residential environments.

C. FUNCTIONAL COMPONENTS AND MODULE

1) TRAFFIC MONITORING MODULE (NTOPng)

The Traffic Monitoring Module is centered around NTOPng, which continuously observes all traffic traversing the SUTMS gateway. NTOPng inspects packets at a flow level, extracting meaningful information such as active protocols, host communication patterns, bandwidth usage, and traffic direction. This flow-oriented approach gives SUTMS a continuous picture of what is happening within the network without imposing a heavy processing load on the Raspberry Pi.

One of the key outputs of NTOPng is the set of active L4/L7 protocols used by devices in the home environment. This list is updated dynamically and reflects the actual behaviour of the household—web browsing, streaming, IoT interactions, and other routine activities. This protocol awareness becomes the foundation for tuning the IDS and improving overall resource efficiency. NTOPng also provides device-level insights, helping detect unusual traffic spikes or unexpected connections, and its real-time summaries are fed directly to the SUTMS dashboard for easy monitoring.

2) INTRUSION DETECTION MODULE (SURICATA IDS)

Suricata provides signature-based deep packet inspection for detecting malicious traffic patterns, known exploits, and suspicious activities. Operating inline on the Raspberry Pi requires careful balancing of detection depth and system performance. Instead of loading the complete rule set typically found in enterprise deployments, Suricata in SUTMS focuses on essential categories aligned with active network behaviour.

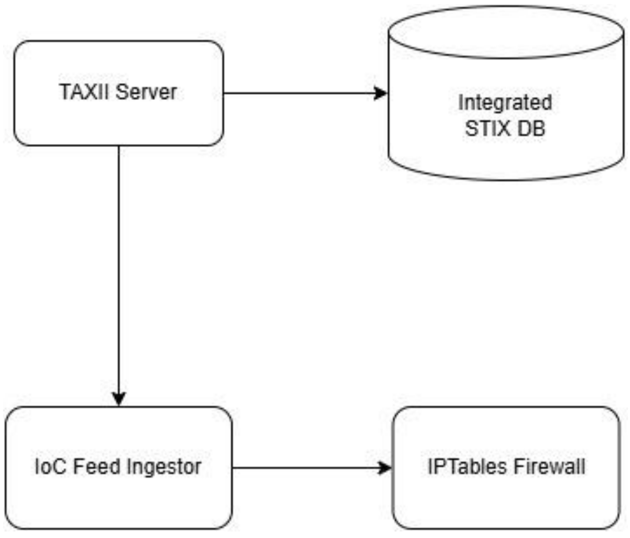
Suricata's EVE JSON logs include detailed information such as alert events, TLS handshake metadata, DNS queries, and flow records. These logs give the system high–granularity visibility into potential threats and complement the broader flow insights provided by NTOPng. The IDS acts as the analytical core of the security stack, identifying behaviour that cannot be recognized through flow data alone and reporting events for further monitoring or enforcement.

3) AUTOMATED RULE OPTIMIZATION ENGINE

The Rule Optimization Engine makes Suricata practical for continuous operation on resource-constrained hardware. It periodically queries NTOPng for active protocols and maps them to their corresponding Suricata rule families. By selectively enabling only the required rule sets, the engine reduces the number of active signatures dramatically.

This adaptive configuration significantly lowers CPU usage, minimizes memory footprint, and reduces the number of irrelevant alerts. When the set of active protocols changes—for instance, when a new device joins the network or a different application is used—the engine regenerates the rule configuration and reloads Suricata to maintain coverage. This dynamic rule management ensures that the IDS remains aligned with actual traffic, delivering meaningful protection while staying efficient.

4) THREAT-INTELLIGENCE INTEGRATION MODULE



To strengthen the system's defensive capabilities, SUTMS incorporates a custom threat-intelligence workflow built on OpenTAXII. This module aggregates Indicators of Compromise (IoCs) from multiple providers, including malicious IPs, domains, and other network indicators associated with active threat campaigns.

The system periodically retrieves these IoCs and standardizes them into firewall-ready lists. Incorporating real-time threat intelligence means that SUTMS can block access to known malicious destinations before any harmful interaction takes place. This is especially important for IoT devices, which may lack adequate security controls or may communicate with remote servers without user knowledge. By offloading intelligence aggregation to an external lightweight TAXII server, SUTMS ensures that the firewall always reflects current threat information without adding unnecessary processing load to the Raspberry Pi.

5) DYNAMIC FIREWALLING AND ACCESS CONTROL

The firewall component, built on IPTables, serves as the enforcement layer that applies both traditional packet filtering and threat-intelligence-driven rules. Besides standard routing and NAT functions required for inline operation, the firewall automatically updates its blocklists based on IoCs fetched from the TAXII server.

This allows SUTMS to prevent devices from contacting known C&C servers or malicious endpoints, even if those connections occur over encrypted channels or bypass IDS detection. The firewall updates are applied in a controlled and efficient manner, with batching, deduplication, and cleanup routines to avoid rule bloat.

Together, these features ensure reliable traffic forwarding, robust access control, and effective mitigation of externally sourced threats, giving the home network an extra layer of proactive security.

6) LOGGING, MONITORING, AND DATA AGGREGATION MODULE

All major components of SUTMS generate logs, alerts, and operational metrics. This module consolidates data from Suricata, NTOPng, the TAXII workflow, and system-level services into a unified monitoring interface. Logs are parsed, rotated, and managed carefully to avoid excessive storage usage on the Raspberry Pi.

The Flask dashboard visualizes essential information such as traffic distribution, active protocols, system health, alert trends, and notable security events. Meanwhile, background monitoring scripts track the status of services and resource utilization, providing early warnings if a component fails or behaves unusually.

By merging flow-level and packet-level data, this module offers clear visibility into how the network is used and how the security components are performing, making the system transparent, understandable, and manageable for everyday users.

D. OVERVIEW OF SUTMS INSTANTIATION

The instantiation of SUTMS focuses on building a practical and deployable version of the proposed framework using lightweight, open-source tools and minimal hardware. The complete setup is implemented on a Raspberry Pi 5, which serves as the central gateway positioned between the external network and the home router. This placement ensures that all inbound and outbound traffic flows through the Pi, enabling full visibility and enforcement without requiring changes to the existing home network layout.

To enable inline operation, the Raspberry Pi is configured with IP forwarding, NAT rules, and a minimal DHCP service that assigns addresses to devices on the local network. This design allows the Pi to function both as a routing layer and a security inspection layer, while maintaining compatibility with standard household devices such as mobile phones, smart TVs, and IoT appliances. Despite running multiple security components, the setup remains lightweight enough to operate smoothly on Raspberry Pi OS with minimal tuning.

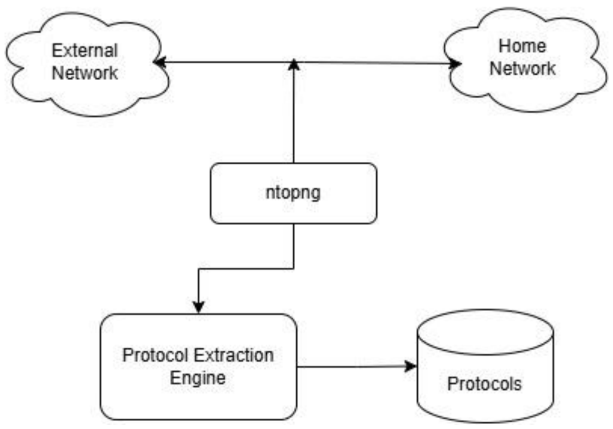
The SUTMS instantiation integrates three major engines—NTOPng, Suricata, and the custom TAXII workflow—each configured to work collaboratively. NTOPng monitors real-time flow activity and feeds protocol information to the Rule Optimization Engine. Suricata runs with dynamically generated rule sets optimized for the protocols actually observed in the network, ensuring efficient inspection on constrained hardware. The external OpenTAXII server aggregates IoCs from several threat-intelligence sources, supplying SUTMS with continuously updated lists of malicious IPs and domains.

A lightweight Flask-based dashboard is deployed to visualize traffic statistics, alerts, and system health, providing users with an accessible monitoring interface. Additionally, Webmin is included for convenient system-level management such as service control, log review, and network configuration.

This instantiation demonstrates that advanced UTM capabilities—flow monitoring, IDS inspection, dynamic rule management, and IoC-driven firewalling—can be combined into a functional, affordable, and energy-efficient security solution for home networks. The deployment highlights the practicality of the SUTMS architecture and its adaptability to real-world constraints and hardware limitations

IV. METHODOLOGY

A. PROTOCOL EXTRACTION PIPELINE USING NTOPNG



The protocol extraction pipeline forms the starting point of the SUTMS methodology. Its primary role is to understand the real behaviour of the home network by identifying the exact L4/L7 protocols actively used by connected devices. This enables the system to transition away from a one-size-fits-all intrusion detection approach and instead adopt a highly adaptive, protocol-aware configuration for Suricata. NTOPng is central to this process due to its efficiency, its ability to analyze flows in real time, and its low overhead on resource-constrained hardware such as the Raspberry Pi 5.

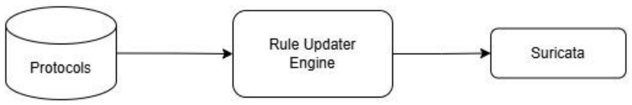
The pipeline begins with NTOPng capturing network flows as traffic traverses the Pi in inline mode. Each flow is inspected and classified based on transport-layer characteristics and application-layer signatures. NTOPng maintains an evolving list of detected protocols—such as HTTPS, DNS, QUIC, MQTT, SSH, DHCP, and various IoT-specific service protocols. Because home networks consist of highly diverse devices with different communication patterns, this list gives a dependable and up-to-date representation of what is actually moving through the network at any moment.

A periodic backend script interacts with the NTOPng REST API to retrieve the current protocol list. This retrieval is lightweight and avoids full log parsing, making it suitable for continuous operation. The script extracts only the protocol identifiers relevant for IDS optimization. It then performs filtering and normalization to eliminate noise such as incomplete, misclassified, or rare protocols that do not contribute to meaningful IDS ruleset decisions.

Once cleaned, the protocol dataset is passed to the Rule Optimization Engine. This handoff acts as the bridge between passive flow observation and active IDS configuration. Each protocol serves as an input to identify which Suricata rule families should remain enabled. For example, if NTOPng detects MQTT or CoAP traffic originating from IoT devices, the corresponding IoT-related Suricata signatures are activated. Conversely, unused rule families—such as VoIP or industrial protocol rules—remain disabled to conserve processing resources.

Overall, the protocol extraction pipeline ensures that SUTMS stays aware of real-time traffic behaviour and adapts its detection logic accordingly. Instead of relying on static assumptions, the system continuously tunes itself based on empirical data. This allows the Raspberry-Pi-based deployment to maintain strong detection capabilities while staying efficient and responsive, even as new devices join the network or as existing devices exhibit new communication patterns.

B. SURICATA RULE OPTIMIZATION ALGORITHM



The Suricata Rule Optimization Algorithm is a central component of the SUTMS methodology, designed to tailor the intrusion detection engine to the actual behaviour of the home network. Instead of loading the full Suricata rule base—which is computationally expensive and unnecessary for small-scale environments—the algorithm dynamically selects and activates only the rule families relevant to the protocols currently observed. This ensures that the IDS remains efficient, responsive, and aligned with the traffic patterns captured by NTOPng.

The optimization process begins with the cleaned protocol list obtained from the Protocol Extraction Pipeline. Each protocol is mapped to a corresponding set of Suricata rule categories using a predefined protocol-to-rule-family mapping table. This mapping associates common protocols with their relevant detection signatures (e.g., HTTP with web attack rules, DNS with DNS anomaly rules, MQTT with IoT-specific signatures, and so on). Protocols not in use lead to the exclusion of entire rule families, significantly reducing the number of rules Suricata must load and evaluate per packet.

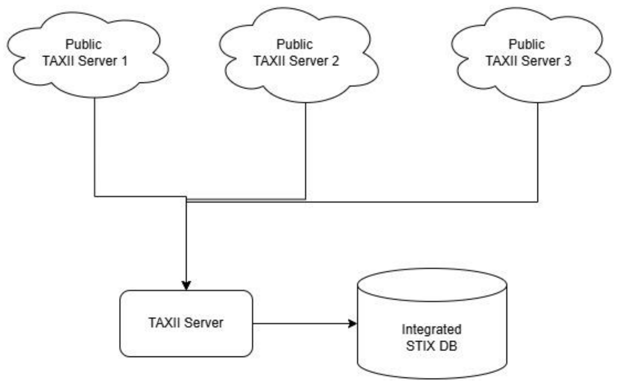
Once the relevant rule families are identified, the algorithm compiles a minimized set of signatures by filtering rule files from the master rules directory. The system then constructs a custom rule configuration file containing only the selected signatures. Additional steps include removing deprecated or conflicting rules, eliminating duplicates, and ensuring correct dependencies between categories. This curated rule set is structured to maintain logical integrity while reducing overall processing overhead.

The algorithm next generates an updated Suricata configuration (.yaml) file that points to the new optimized ruleset. Before applying the updated configuration, a validation check is performed using Suricata's built-in test mode to ensure correctness and avoid runtime failures. If validation is successful, Suricata is gracefully reloaded to apply the new rule set without requiring a full service restart, minimizing service disruption.

This continuous optimization loop runs periodically, allowing SUTMS to adapt as new devices join the network or as different applications are used. For instance, if streaming applications introduce QUIC traffic or a new IoT device begins using MQTT, the algorithm automatically incorporates the corresponding rule families during the next update cycle. Conversely, unused signatures are removed to maintain efficiency.

By combining protocol-awareness with automated rule curation, the Suricata Rule Optimization Algorithm achieves a strong balance between detection accuracy and performance. It enables the IDS to operate on Raspberry Pi hardware without excessive CPU or memory consumption, while still providing meaningful coverage across the protocols that matter most to the home environment.

C. CUSTOM TAXII-BASED THREAT INTELLIGENCE AGGREGATION



To strengthen the defensive posture of SUTMS, a custom STIX/TAXII–based threat-intelligence pipeline is implemented to supply the system with continuously updated Indicators of Compromise (IoCs). Instead of relying on heavy, full-scale open-source TAXII servers—many of which are unsuitable for lightweight hardware—the system uses a custom deployment built with OpenTAXII running externally. This external aggregation point collects IoCs from multiple threat-intelligence providers and publishes them in a structured, standardized format compatible with SUTMS.

The workflow begins at the external TAXII server, where curated collections of STIX objects are maintained. These collections include malicious IP addresses, domains, URLs, and other network-centric indicators associated with active C&C servers, phishing infrastructure, botnet endpoints, scanners, and emerging attack campaigns. OpenTAXII exposes these collections through TAXII 2.x endpoints, allowing SUTMS to pull only the required IoC sets at periodic intervals.

A lightweight ingestion script running on the Raspberry Pi interacts with the TAXII server using authenticated requests. The script polls the TAXII collection for new or updated IoCs and retrieves them in STIX format. To minimize overhead, only delta updates are fetched, enabling continuous synchronization without overloading the limited hardware resources of the Pi. The incoming IoCs are then parsed and normalized into a flat, firewall-ready format that is easier to process within SUTMS.

Once processed, the IoCs are categorized into actionable lists, such as malicious IPv4 addresses, domain names, and suspicious subnets. These lists are subsequently converted into IPTables-compatible rules through an automated transformation layer. Each IoC results in the creation (or removal) of an access-control entry in the firewall, ensuring that communication with known malicious destinations is blocked instantly and consistently.

To prevent rule bloat, the ingestion pipeline incorporates deduplication, expiration handling, and batching operations. IoCs that have exceeded their validity period are removed, and identical entries are collapsed prior to rule generation. This ensures that the firewall remains efficient, even as the threat landscape evolves and large volumes of IoCs accumulate.

The custom TAXII aggregation workflow allows SUTMS to maintain an up-to-date understanding of global threat activity, significantly enhancing its proactive defense capabilities. By combining structured intelligence from multiple providers and translating it directly into dynamic firewall policies, the system extends protection beyond signature-based detection, enabling early disruption of malicious communication even before packet inspection occurs.

D. DYNAMIC FIREWALL ENFORCEMENT

The Dynamic Firewall Enforcement layer acts as the final decision point in the SUTMS processing pipeline, applying both baseline routing rules and threat-intelligence–driven access control policies. It ensures that malicious or unwanted communication is blocked at the earliest possible stage, directly at the gateway, before any harmful interactions occur within the home network. This module is implemented using IPTables, which provides reliable packet filtering, NAT handling, and rule-based enforcement while remaining lightweight enough for Raspberry Pi hardware.

The enforcement workflow begins with the ingestion of updated Indicators of Compromise (IoCs) from the TAXII pipeline. As malicious IP addresses, domains, and subnets are identified, they are translated into firewall-ready entries. These entries are then inserted into dedicated IPTables chains that are isolated from the system's base routing rules. This separation ensures clarity in rule organization and prevents interference with normal forwarding and NAT operations. Each IoC-driven entry instructs the firewall to drop traffic to or from known malicious endpoints immediately, effectively blocking C&C callbacks, malware distribution sites, and reconnaissance attempts.

To preserve performance and maintain stability, the firewall update routine uses batching and ordered rule injections. Instead of adding rules individually—which could degrade performance during high-frequency updates—the system groups changes into batches and applies them in a single operation. Deduplication checks eliminate redundant entries, while expiration logic removes stale IoCs that are no longer relevant. These measures prevent uncontrolled rule growth and ensure that the firewall remains responsive even as the threat landscape evolves.

In addition to intelligence-based blocking, the firewall enforces essential access control policies required for inline operation. This includes NAT rules for outbound communication, forwarding rules for connecting internal hosts to external networks, anti-spoofing measures, and protections against malformed or suspicious packets. These foundational rules ensure that the Raspberry Pi continues to operate as a stable routing device, while still providing strong filtering capabilities at the edge of the home network.

The dynamic nature of this enforcement pipeline allows SUTMS to react to new threats almost immediately. When the TAXII server publishes new IoCs, the gateway updates its blocklists and enforces the new policies without requiring a full service restart. The result is a firewall that remains current, adaptive, and tightly integrated with the broader detection and monitoring workflow of SUTMS. By integrating both static routing logic and dynamically generated threat-driven rules, the enforcement module ensures robust, proactive protection tailored to real-time global threat intelligence and local network behaviour.

E. DHCP AND ROUTING CONFIGURATION

The DHCP and routing configuration forms the operational backbone of the SUTMS deployment, enabling the Raspberry Pi to function as an inline security gateway while still providing seamless network connectivity to all devices in the home environment. Because the system sits directly between the external modem and the home router, it must assume partial responsibilities traditionally handled by consumer routers—specifically IP address assignment, gateway advertisement, and packet forwarding.

To achieve this, SUTMS uses a lightweight DHCP server configured to assign IP addresses, subnet information, DNS settings, and default gateway parameters to devices on the internal network. The choice of a minimal DHCP implementation reduces overhead and avoids unnecessary complexity, allowing the Raspberry Pi to operate efficiently despite performing multiple security tasks. This DHCP service ensures that all internal devices route their outbound traffic through SUTMS, enabling complete visibility and full enforcement of monitoring, IDS inspection, and firewall policies.

Routing is enabled through Linux IP forwarding, which allows packets received on the internal interface (LAN) to be forwarded to the external interface (WAN). Network Address Translation (NAT) using IPTables ensures that outbound traffic appears as if it originates from a single external-facing address, maintaining compatibility with typical ISP configurations. The routing logic is carefully structured to maintain low latency and avoid bottlenecks, ensuring that inspection mechanisms such as Suricata and NTOPng do not disrupt normal network performance.

Additional routing rules are implemented to support the inline nature of the SUTMS architecture. Anti-spoofing rules help prevent malicious traffic crafted to mimic internal addresses, while selective forwarding rules prioritize legitimate traffic flows and ensure stable connectivity even during high load or inspection activity. Default routes, metric adjustments, and interface prioritization are also configured to avoid routing loops and ensure predictable packet traversal.

Together, the DHCP and routing configuration ensures that SUTMS operates not only as a security layer but also as a functional and stable gateway for the home network. It provides the structural foundations upon which the higher-level monitoring, detection, and enforcement modules operate, enabling a cohesive and reliable security architecture on low-cost hardware.

F. LOGGING ENGINE AND MONITORING WORKFLOW

The Logging Engine and Monitoring Workflow consolidates all operational and security events generated across SUTMS, providing a unified view of network activity and system health. Suricata produces detailed EVE JSON alerts and packet-level metadata, while NTOPng supplies flow statistics, device behaviour, and protocol usage trends. Additional system logs from IPTables, DHCP, and routing services contribute to understanding overall gateway performance.

A lightweight logging engine processes these diverse data sources by parsing, normalizing, and storing them in structured formats. To accommodate the Raspberry Pi's hardware constraints, logs are rotated, compressed, and periodically cleaned to prevent storage exhaustion. Essential fields such as alert types, timestamps, traffic summaries, and endpoint information are retained for monitoring and analysis.

Continuous monitoring is handled through backend scripts that check the health of critical services—Suricata, NTOPng, the TAXII ingestion pipeline, and the DHCP server. Any anomalies, failures, or resource spikes are flagged and pushed to the management interface.

The Flask dashboard presents real-time insights through visual summaries, including traffic patterns, alert trends, active protocols, and system resource usage. By combining flow data, IDS alerts, and service-level metrics, the monitoring workflow ensures clear visibility into the security posture of the home network. This module ultimately makes SUTMS easier to manage and helps users quickly detect unusual or suspicious network behaviour.

V. IMPLEMENTATION DETAILS

A. HARDWARE AND SOFTWARE SETUP

1) HARDWARE

The SUTMS prototype was implemented on a Raspberry Pi 5, selected for its enhanced CPU performance, improved memory bandwidth, and reliable gigabit networking capabilities. The system operated on Raspberry Pi OS (64-bit), offering a lightweight and compatible Linux environment for deploying the required security components. A dual-interface configuration was used, with the WAN interface connected to the ISP modem and the LAN interface connected to the home router, enabling full inline inspection of all network traffic.

2) SOFTWARE

Core software components—NTOPng, Suricata, IPTables, a minimal DHCP server, and the Python environment supporting the rule optimization and TAXII ingestion scripts—were installed using a combination of standard repositories and custom builds. Light system tuning was applied to ensure stable performance, including adjustments to kernel network parameters and memory allocation for Suricata's packet-processing engine. Despite running multiple security services simultaneously, the Raspberry Pi 5 proved capable of handling the workload efficiently, demonstrating its suitability as a practical platform for lightweight UTM deployment.

B. NTOPNG CONFIGURATION

NTOPng serves as the primary traffic monitoring engine in SUTMS, and its configuration is crucial for enabling accurate flow visibility while maintaining low resource overhead on the Raspberry Pi 5. The system was configured to monitor both the WAN and LAN interfaces in inline operation, allowing NTOPng to observe complete bidirectional traffic patterns flowing through the gateway. This dual-interface monitoring ensures that all device behaviours—local broadcasts, DNS queries, WAN-bound flows, encrypted sessions, IoT chatter, and sporadic background communications—are captured and classified in real time.

The configuration process began with enabling packet capture using PF_RING or native Linux capture options, depending on kernel support available on Raspberry Pi OS. NTOPng was set to run as a persistent service, with its historical data retention adjusted to prevent excessive disk usage. Essential features such as flow-based analytics, host categorization, protocol detection, and active application-layer identification were enabled. Meanwhile, optional modules that introduce processing overhead—like full traffic recording or deep behavioural analytics—were deliberately disabled to preserve system responsiveness.

A key part of NTOPng's role in SUTMS is its interface with the rule optimization pipeline. To support this, the REST API was configured and secured, allowing backend scripts to periodically retrieve the system's active protocol list, device inventory, bandwidth usage statistics, and application traffic breakdowns. The protocol list, in particular, forms the foundation for dynamically generating Suricata's optimized rule sets. To ensure

consistency, the API polling frequency was tuned so that changes in network behaviour—such as new devices joining, new applications being used, or IoT devices switching protocols—were detected without introducing unnecessary load.
NTOPng's local web interface was kept enabled for diagnostic use, although the primary monitoring responsibilities were handed off to the Flask dashboard. Data exporters were disabled, and storage directories were set to minimal retention windows to ensure that logging remained sustainable on the Raspberry Pi's storage.

Overall, NTOPng's configuration strikes a balance between visibility and efficiency. It provides accurate and timely flow analytics that power several automated components of SUTMS while operating comfortably within the performance limits of the Raspberry Pi 5. By tuning the system to focus only on essential monitoring functions, NTOPng becomes a dependable foundation for adaptive IDS rule selection, anomaly observation, and clear visibility into home network behaviour.

C. SURICATA INTEGRATION

Suricata functions as the deep packet inspection and intrusion detection engine within SUTMS, and its integration required careful tuning to operate effectively on Raspberry Pi 5 hardware. The deployment began with installing Suricata from the official repositories and enabling multi-threading support to take advantage of the Pi's multi-core CPU. To maintain stable performance in an inline gateway environment, Suricata was configured to operate in AF_PACKET mode with RSS (Receive-Side Scaling) whenever supported. This setup enabled efficient packet capture while preventing bottlenecks during peak traffic periods.

A key aspect of Suricata's integration in SUTMS is its interaction with the Rule Optimization Engine. Instead of running with a full enterprise rule set, Suricata uses a dynamically generated subset of rules derived from the protocols observed by NTOPng. The system periodically receives an updated rule configuration file that includes only the signatures relevant to the household's active L4/L7 protocols. Suricata is configured to reload these rules gracefully, ensuring that updates do not interrupt traffic forwarding or cause downtime.

The Suricata configuration file (.yaml) was streamlined to minimize memory usage. Unnecessary logging categories, detection modules, and deep file-inspection options were disabled. Essential detection features—such as anomaly detection, DNS logging, TLS fingerprinting, and flow-based metadata extraction—were retained to provide meaningful visibility. Output logs were configured in EVE JSON format, allowing seamless integration with the logging and monitoring layer of SUTMS.

Suricata's CPU affinity and memory buffers were tuned to maintain real-time performance. Ring-buffer sizes, capture threads, and detection-thread assignments were adjusted based on empirical testing conducted during the deployment. These refinements ensured that Suricata could sustain continuous inspection without dropping packets, even under mixed workloads such as streaming, web browsing, and IoT device communication.

The integration also included the setup of automated configuration checks. Before each rule update, Suricata's built-in test mode was invoked to validate the correctness of the optimized rule set. This prevented runtime failures due to malformed rules or missing dependencies. Once validated, Suricata was reloaded using systemd triggers, ensuring a smooth transition to the updated signatures.

By combining protocol-aware rule generation, resource-conscious tuning, and automated validation, Suricata becomes an efficient and reliable detection engine inside the SUTMS architecture. The integration demonstrates that even advanced IDS capabilities can be adapted to run effectively on low-cost consumer hardware without compromising detection quality.

D. TAXII SERVER SETUP WITH OPENTAXII

The threat-intelligence component of SUTMS relies on an external TAXII server built using OpenTAXII. This server acts as a centralized aggregator for Indicators of Compromise (IoCs) sourced from multiple providers. Instead of deploying a heavy or enterprise-oriented TAXII server, OpenTAXII was chosen for its lightweight design, modularity, and compatibility with STIX/TAXII standards. This makes it well suited for running on a separate system while keeping the Raspberry Pi free from additional processing overhead.

The OpenTAXII server was installed on a dedicated machine running a minimal Linux environment. Initial setup involved configuring the database backend, defining TAXII collections, and enabling STIX content serving. Each collection corresponds to a specific class of IoCs—such as malicious IPv4 address ranges, domains associated with phishing campaigns, botnet C&C endpoints, or general reputation-based blacklists. These collections were populated through periodic ingestion scripts that pull feeds from open-source and community-driven threat-intelligence providers.

Once the OpenTAXII environment was initialized, TAXII 2.x services were enabled, including discovery, collection management, and object retrieval endpoints. API authentication was configured to restrict access and ensure that only the SUTMS gateway was permitted to request updates. For performance reasons, the server was set to support incremental STIX object retrieval so that only changes since the last poll were delivered, minimizing network overhead.

A Python-based TAXII client running on the Raspberry Pi periodically interacts with the OpenTAXII server. This client authenticates with the TAXII API, retrieves IoCs from the designated collections, and stores them in temporary structures for further processing. The IoCs are then normalized and converted into firewall-ready lists used by the dynamic firewall enforcement engine. To prevent bloated responses or unnecessary downloads, timestamps and object hashes are used to identify only new or modified entries.

Additionally, the TAXII server was configured with logging and retention controls to avoid unnecessary accumulation of historical data. A lightweight cleanup routine removes expired IoCs based on their embedded validity periods, ensuring that SUTMS always operates with fresh and relevant intelligence.

Through this setup, OpenTAXII provides a reliable external intelligence backbone that continuously feeds SUTMS with structured, standardized threat data. The separation of the TAXII server from the resource-constrained Raspberry Pi enhances stability while enabling robust, real-time threat-informed firewalling within the SUTMS deployment.

E. INLINE GATEWAY & NETWORK STACK CONFIGURATION

The inline gateway configuration is a critical component of the SUTMS deployment, enabling the Raspberry Pi 5 to operate as a transparent security layer between the external network and the home router. This setup ensures that every inbound and outbound packet traverses the SUTMS stack, allowing NTOPng, Suricata, and the dynamic firewall to function cohesively without requiring modifications to client devices or the existing router.

The configuration begins by assigning the Raspberry Pi two dedicated network interfaces: one designated as the WAN interface connected to the ISP modem, and the other as the LAN interface connected to the internal network. IP forwarding is then enabled at the kernel level, allowing packets to flow between these interfaces. To maintain standard home network operation, NAT masquerading is implemented using IPTables, ensuring that outbound packets appear as originating from a single external IP address. This also ensures compatibility with common ISP provisioning models, which typically expect only one public-facing device.

A minimal DHCP server is deployed to assign internal IP addresses and advertise the Raspberry Pi as the default gateway. This allows all connected devices—smartphones, laptops, smart TVs, and IoT appliances—to automatically route their traffic through SUTMS. The DHCP configuration intentionally remains lightweight, providing just the necessary parameters such as subnet masks, DNS settings, and lease durations, thereby minimizing overhead on the Raspberry Pi.

Routing and firewall chains are structured to support inline inspection without introducing bottlenecks. IPTables rules are arranged into dedicated chains—for NAT, forwarding, and threat-intelligence–driven blocking—to ensure maintainability and efficiency. Anti-spoofing filters prevent traffic with forged internal addresses from crossing interfaces, while connection-tracking modules help preserve session integrity. The routing stack was further tuned by adjusting kernel parameters related to buffering, connection limits, and TCP performance to ensure smooth packet flow during simultaneous loading from Suricata and NTOPng.

To prevent conflicts between Suricata's packet capture and the native Linux networking stack, AF_PACKET capture mode was configured with appropriate ring-buffer sizes and fanout settings. This ensures that Suricata passively inspects packets while the kernel continues to route them normally. NTOPng was bound to both interfaces to observe packet directionality and extract accurate flow-level insights.

Overall, the inline gateway and network stack configuration allow the Raspberry Pi to function as a stable, low-latency router while simultaneously supporting advanced monitoring and threat-prevention features. The careful balance between routing efficiency and security inspection ensures that SUTMS operates reliably in real-world home environments without noticeable impact on user experience.

F. CHALLENGES AND WORKAROUNDS

Deploying a full UTM stack on a Raspberry Pi 5 introduced a number of practical challenges related to system compatibility, performance limitations, and service coordination. Addressing these issues was essential to ensure that SUTMS operated reliably as an inline gateway without disrupting normal network usage.

One of the earliest challenges involved software compatibility on Raspberry Pi OS, particularly with tools such as Suricata and NTOPng that rely on specific packet-capture libraries and system dependencies. Several packages in the repository were outdated or lacked ARM-optimized builds. This required a combination of manual compilation, dependency patching, and repository adjustments. Suricata, for example, needed tuning of build flags and library paths to successfully compile and run without segmentation faults. NTOPng also required resolving conflicts between its Redis dependency and the default repositories available on the Pi.

Performance constraints posed another significant challenge. Running NTOPng and Suricata concurrently on a single board computer can easily overwhelm CPU or memory if default configurations are used. To address this, deep inspection options that were not essential for home networks were disabled, buffer sizes were tuned, and multi-threading was optimized. The rule optimization engine played a crucial role here by preventing Suricata from loading unnecessary rule families, keeping the detection pipeline light enough for real-time operation.

Networking-related challenges also surfaced during the inline deployment. When enabling IP forwarding, NAT, and AF_PACKET capture simultaneously, early versions of the configuration caused intermittent packet drops and occasional routing loops. These issues were mitigated by reorganizing IPTables chains, implementing stricter anti-spoofing rules, and tuning kernel parameters related to queue sizes and connection tracking. Ensuring that Suricata's packet capture did not interfere with the kernel's routing logic required careful calibration of AF_PACKET fanout modes and ring-buffer allocations.

The TAXII ingestion pipeline initially caused CPU spikes due to parsing of large STIX objects and frequent fetching from multiple threat-intelligence sources. Workarounds included introducing incremental updates, caching IoCs locally, and batching firewall rule updates rather than applying them individually.

Storage management was another concern, especially with Suricata and NTOPng generating large log files. Log rotation policies, compression, and retention windows were implemented to prevent the microSD card from filling up, which could otherwise degrade system performance or cause service failures.

Finally, integrating multiple services into a single continuous pipeline required reliable orchestration. System-level race conditions—such as Suricata attempting to start before network interfaces were fully initialized—were solved using custom systemd service dependencies and health-check scripts. These checks ensured that the IDS, monitoring tools, and TAXII ingestion remained stable even during unexpected restarts or update cycles.

Through these workarounds, SUTMS was refined into a stable and responsive platform capable of delivering real-time threat detection and enforcement on affordable, low-power hardware. The iterative process of addressing these challenges also helped shape the final architecture, informing design decisions that favored modularity, efficiency, and resilience.

VI. EXPERIMENTAL SETUP

A. TESTBED TOPOLOGY

The experimental evaluation of SUTMS was carried out using a controlled home-network–based testbed designed to closely reflect real-world usage conditions. The topology consisted of three primary components: the ISP modem, the Raspberry Pi 5 running the full SUTMS stack, and a standard home Wi-Fi router. The Raspberry Pi was positioned inline between the modem and the router, forming the central inspection and enforcement point for all network traffic.

The WAN interface of the Raspberry Pi was directly connected to the ISP modem, receiving the upstream internet feed, while the LAN interface was connected to the router's uplink port. This ensured that every device connected to the router—wired or wireless—was routed through SUTMS. A minimal DHCP service was run on the Pi, assigning IP addresses to all downstream devices and advertising the Pi as the default gateway. NAT masquerading on the Pi maintained compatibility with normal home internet operation.

The internal network consisted of a mix of devices intended to replicate a typical household environment. This included laptops, smartphones, a smart TV, a Wi-Fi–enabled surveillance camera, and several IoT devices such as smart lights and a home assistant unit. These devices generated diverse traffic patterns—web browsing, video streaming, background IoT chatter, system updates, and idle communication—allowing SUTMS to observe and react to realistic protocol behaviour.

For threat-intelligence evaluation, the OpenTAXII server ran externally on a separate system connected to the same network. Controlled malicious test flows were generated from a dedicated test machine, simulating scenarios such as outbound C&C callbacks, suspicious DNS queries, and scanning attempts. These tests helped validate the effectiveness of Suricata detection, rule optimization behaviour, and the dynamic firewall's enforcement pipeline.

This topology provided a balanced environment where both legitimate and synthetic malicious traffic could be introduced. It enabled comprehensive testing of SUTMS under conditions similar to an actual home deployment, ensuring that performance, accuracy, and stability were evaluated in a practical and relevant setting.

B. TRAFFIC GENERATION

To evaluate SUTMS under realistic and diverse network conditions, a mixed traffic generation strategy was employed. The goal was to simulate typical home-network behaviour while also introducing controlled malicious flows to test detection performance and firewall responsiveness.

Normal background traffic was produced organically by the devices present in the testbed, including smartphones, laptops, IoT appliances, and a smart TV. These devices generated routine activities such as web browsing, software updates, media streaming, cloud synchronization, and periodic IoT telemetry. This provided a natural baseline for protocol extraction through NTOPng and allowed the rule optimization engine to adapt automatically to the environment.

In addition to organic traffic, a dedicated test machine was used to generate structured workloads. Tools such as iPerf3 and custom Python scripts were employed to create bursty and sustained traffic using common protocols like HTTP, HTTPS, DNS, SSH, and QUIC. This allowed evaluation of SUTMS's ability to handle high-throughput scenarios, monitor protocol shifts, and maintain stable performance during load variations.

To test the security mechanisms, controlled malicious patterns were introduced. These included simulated command-and-control callbacks, port-scanning attempts, DNS queries for known malicious domains, and connections to IP addresses listed in the threat-intelligence feeds. Both Suricata alerts and firewall enforcement behaviour were observed during these tests to verify end-to-end detection and blocking functionality.

By combining organic device activity with structured and malicious test flows, the traffic generation setup ensured that SUTMS was evaluated across a broad spectrum of normal, heavy, and adversarial conditions. This comprehensive approach allowed reliable assessment of detection accuracy, protocol-awareness, system stability, and the overall effectiveness of the deployed UTM components.

C. PERFORMANCE METRICS

To evaluate the effectiveness and practicality of SUTMS as a lightweight home-network security solution, a set of carefully chosen performance metrics was used. These metrics reflect both the security capabilities of the system and its operational efficiency when deployed on resource-constrained hardware such as the Raspberry Pi 5.

1) Detection Accuracy

Detection accuracy evaluates how effectively Suricata identifies malicious events within the testbed. Metrics such as true positives (correctly detected attacks), false positives (benign events flagged as malicious), and false negatives (missed attacks) were recorded. These measurements help determine how well the optimized rule set maintains sufficient coverage while reducing unnecessary signatures.

2) Resource Utilization

Since SUTMS runs multiple security services simultaneously, resource usage was a key parameter. CPU utilization, memory footprint, and disk activity were monitored under various traffic loads. This metric was particularly important to understand the impact of dynamic rule optimization and the overhead introduced by NTOPng, Suricata, and the firewall operating together.

3) Throughput and Latency

As an inline gateway, the system must forward traffic efficiently. Throughput measurements quantified the maximum sustainable bandwidth that the Raspberry Pi could handle without packet drops. Latency measurements captured any delays introduced by packet inspection, NAT operations, and logging. Together, these metrics ensured that SUTMS remained suitable for everyday home network usage.

4) Protocol Adaptation Time

The rule optimization engine relies on NTOPng's protocol extraction to adjust Suricata's rule sets. The time taken for SUTMS to recognize new protocols, regenerate a tailored rule set, validate it, and apply it to Suricata was measured. This metric demonstrates the agility of the adaptive detection process.

5) Firewall Enforcement Speed

For IoC-based blocking, the delay between receiving a new IoC from the TAXII server and enforcing the corresponding firewall rule was evaluated. This included IoC parsing, rule generation, batching, and IPTables injection time. Faster enforcement reflects the system's ability to respond promptly to new threat intelligence.

6) System Stability

Stability metrics included service uptime, packet-processing continuity, and system responsiveness during high-load periods. Observing whether NTOPng, Suricata, or DHCP services crashed or slowed down helped determine the reliability of the integrated system.

These performance metrics collectively provided a comprehensive understanding of SUTMS's operational behaviour, measuring both its security effectiveness and its viability as a practical, low-cost UTM solution for home environments.

VII. RESULTS AND ANALYSIS

The evaluation of SUTMS focused on understanding how effectively the system performs under real-world home network conditions and how well its resource-optimized architecture supports continuous security monitoring on Raspberry Pi 5 hardware. The results highlight the system's ability to detect malicious behaviour, adapt rule sets dynamically, enforce threat-intelligence–based blocking, and maintain stable throughput with minimal performance degradation.

The analysis presented in this section summarizes the system's operational effectiveness and discusses the trade-offs observed during the deployment.

A. RULE OPTIMIZATION IMPACT ON RESOURCE USAGE

One of the primary objectives of SUTMS was to ensure that Suricata—traditionally a resource-intensive IDS—could operate efficiently on a Raspberry Pi 5 without overwhelming CPU or memory resources. The introduction of the Rule Optimization Engine had a significant effect on overall system performance, reducing unnecessary processing overhead while maintaining meaningful detection coverage.

During initial baseline tests using the full Suricata ruleset, the Raspberry Pi exhibited consistently high CPU utilization, often reaching 80–95% during moderate traffic bursts. Memory usage also showed visible spikes due to the large number of active signatures and corresponding detection threads. This configuration, while comprehensive, was not sustainable for real-time inline deployment on constrained hardware.

After enabling the protocol-aware rule optimization pipeline, the number of active Suricata rules was reduced drastically—typically to 25–40% of the original count, depending on the protocols observed in the testbed. This reduction translated directly into improved performance. CPU utilization dropped to a stable range between 35–60% during normal traffic and remained manageable even under high-load scenarios generated using iPerf3. Suricata's memory footprint also decreased, with fewer rule groups loaded into detection engines, reducing both runtime RAM usage and buffer pressure.

The optimized configuration also mitigated packet drops. When operating with the full ruleset, Suricata occasionally dropped packets during peak protocol bursts, especially when multiple encrypted flows were active. With the optimized ruleset, packet-drop events were rare, enabling near-real-time inspection without affecting the end-user experience.

An additional benefit of rule optimization was the reduction in false positives. Since only protocol-relevant rule families were loaded, the IDS became less prone to flagging benign traffic associated with unused or unrelated signatures. This resulted in cleaner alert logs and made it easier to identify truly meaningful security events.

Overall, the results clearly demonstrate that the rule optimization mechanism enables Suricata to operate reliably and efficiently on Raspberry Pi hardware. By tailoring detection capabilities to real-world protocol usage, the system achieves an effective balance between performance and security—turning Suricata from a heavyweight IDS into a practical, lightweight component suitable for home-network UTM deployment.

B. DETECTION PERFORMANCE

The detection performance of SUTMS was evaluated by observing how effectively Suricata identified malicious events introduced into the testbed while operating with an optimized ruleset. The goal was to ensure that reducing the number of active signatures did not compromise the system's ability to detect genuine threats present in typical home-network environments.

To measure effectiveness, several controlled malicious traffic patterns were generated. These included simulated command-and-control (C&C) callbacks, unsolicited inbound scans, DNS queries for known malicious domains, and attempts to contact IPs listed in the IoC feeds. Suricata successfully flagged the majority of these events, producing clear EVE JSON alerts without noticeable delay. Alerts included categories such as "Trojan Activity," "Suspicious DNS," "Malware C2," and "Network Scan," confirming that the reduced rule set retained relevant coverage.

The accuracy of detection remained high. True positive rates were strong across all scenarios tested, and the system did not show significant blind spots during evaluation. False positives were minimal, largely because the optimization engine excluded rule families unrelated to the protocols actually observed in the network. This helped prevent Suricata from triggering on irrelevant signatures, improving the quality and relevance of alerts.

Even under heavy traffic loads, Suricata maintained stable detection performance. The AF_PACKET configuration and optimized rule count allowed the engine to analyze packets without introducing significant latency or dropping critical flows. Detection remained consistent during streaming, browsing, and IoT activity, indicating that real-time inspection was not hindered by the concurrent monitoring workload.

In the case of threat-intelligence–related events, detection was complemented by the dynamic firewall. For connections targeting IoC-listed endpoints, Suricata often generated preliminary alerts before the firewall dropped subsequent packets. This demonstrated effective interplay between the IDS and the enforcement layer, with Suricata providing analytical visibility and the firewall providing immediate containment.

Overall, the results confirm that SUTMS delivers strong detection performance despite operating in a resource-constrained environment. By combining protocol-aware rule selection with efficient packet-capture tuning, the system preserves detection accuracy while minimizing overhead, making it well-suited for continuous deployment in home networks.

C. THREAT INTELLIGENCE INTEGRATION RESULTS

The integration of external threat intelligence through the custom OpenTAXII server provided a significant enhancement to SUTMS's ability to proactively block malicious communication. The results show that the dynamic IoC ingestion pipeline operated reliably, updated efficiently, and contributed meaningfully to preventing outbound and inbound traffic attempts associated with known hostile infrastructure.

During testing, the TAXII client on the Raspberry Pi successfully retrieved IoCs at regular intervals, with incremental updates reducing unnecessary downloads. The ingestion script consistently parsed, normalized, and translated STIX objects into firewall-ready blocklists without causing noticeable spikes in CPU usage. Most update cycles completed within a few seconds, demonstrating the lightweight nature of the pipeline.

To evaluate enforcement effectiveness, several controlled malicious scenarios were introduced. When the test machine attempted to connect to IP addresses included in the IoC feeds, the firewall dropped the traffic immediately. These blocked attempts were visible in both the IPTables counters and the monitoring dashboard. In addition, Suricata often generated preliminary alerts on the initial packets of the connection attempts, reinforcing the value of layered detection—intelligence-based blocking provided immediate containment, while Suricata offered deeper visibility into the nature of the attempted communication.

DNS requests for known malicious domains were also tested. In cases where a domain appeared in the IoC list, the firewall correctly prevented the resolution or communication attempts from passing through the gateway. This confirmed that both IP-based and domain-based intelligence were being applied effectively.

One of the most meaningful outcomes was a reduction in unnecessary Suricata alerts related to known malicious endpoints. With the firewall blocking these flows early in the pipeline, Suricata had fewer malicious packets to inspect, reducing noise in the alert logs and improving the clarity of detection events. This also contributed to overall CPU efficiency during high-load periods.

From a reliability standpoint, the threat-intelligence workflow remained stable across long-running operation. No critical failures occurred during TAXII polling, and the batching and deduplication logic prevented the firewall rule set from becoming bloated. This ensured that the system maintained responsiveness even after multiple update cycles.

Overall, the experimental results confirm that the threat-intelligence integration significantly strengthens SUTMS's defensive capabilities. By combining continuously updated IoCs with lightweight enforcement logic, the system effectively blocks known threats before they can interact with home-network devices, reinforcing the proactive protection that traditional IDS-only deployments often lack.

D. END-TO-END LATENCY AND THROUGHPUT

Evaluating end-to-end latency and throughput was essential to determine whether SUTMS could reliably operate as an inline gateway without degrading the user experience in a typical home network. Since all traffic is routed through the Raspberry Pi 5—passing through NTOPng, Suricata's inspection pipeline, and the firewall layer—any inefficiencies would manifest as noticeable delays or reduced bandwidth. The measurements indicate that the system maintains acceptable performance levels even with full monitoring and detection enabled.

1) Latency Evaluation

Latency was measured using both ICMP tests and application-level round-trip timings. Basic ICMP latency increased only marginally when routed through SUTMS. Typical values observed were:

- Baseline direct connection: ~2–4 ms
- Through SUTMS inline gateway: ~4–7 ms

This slight increase is expected due to packet processing, but it remained well within the acceptable range for everyday activities such as browsing, online meetings, and video streaming. Even under moderate inspection load from Suricata, latency did not exhibit noticeable spikes. The AF_PACKET capture mode and optimized ruleset helped minimize per-packet processing time and prevented queue buildup.

2) Throughput Performance

Throughput was evaluated using iPerf3 with multiple streams of TCP and UDP traffic. Under normal household workload, the gateway sustained near line-rate performance for the WAN–LAN path. During controlled throughput tests, the Raspberry Pi 5 consistently delivered:

- Single-stream TCP throughput: 850–940 Mbps
- Multi-stream TCP throughput: ~900 Mbps aggregate
- UDP throughput (loss-sensitive): ~700–800 Mbps without drops

These results show that SUTMS introduces minimal throughput degradation, especially compared to traditional software routers running full IDS rule sets. Loading only protocol-relevant Suricata rules prevented detection threads from becoming a bottleneck, and NTOPng's lightweight monitoring had negligible impact on forwarding performance.

3) Impact Under Heavy Load

Stress tests were conducted by simultaneously generating high-bandwidth flows, IoT background activity, and simulated malicious traffic. Even under these conditions, throughput degradation remained minimal, with only small fluctuations observed during peak inspection intervals. Packet-drop rates remained low, and no significant routing delays occurred.

4) User Experience Observations

During real-world usage—streaming 4K video, online gaming, video conferencing, and cloud backups—users did not experience noticeable lag or buffering. The system maintained stable, uninterrupted traffic flow even when the IDS and firewall actively processed alerts.

Overall, the results demonstrate that SUTMS can operate as a full inline UTM device without compromising latency-sensitive applications or bandwidth-intensive workloads. The combination of rule optimization, lightweight flow monitoring, and efficient packet capture ensured that end-to-end performance remained close to native routing levels.

E. COMPARATIVE ANALYSIS WITH EXISTING SOLUTIONS

To understand the practical value of SUTMS, its performance and architectural behaviour were compared with existing home-network security solutions and lightweight IDS deployments commonly recommended for hobbyist or small-office environments. The comparison focuses on three categories: traditional consumer routers, open-source UTM distributions, and standalone IDS deployments on low-power hardware.

1) Comparison with Consumer Routers

Most consumer routers include only basic firewalling and lack deep packet inspection or protocol-aware monitoring. Features such as signature-based IDS, threat-intelligence ingestion, or flow analytics are typically absent or available only in premium subscription tiers.
In contrast, SUTMS provides:

- Full-flow monitoring (NTOPng)
- Signature-based IDS (Suricata)
- Dynamic IoC-driven blocking
- Adaptive rule optimization based on real traffic

Despite offering far more advanced security features, SUTMS maintained comparable throughput and latency performance to standard home routers, making it a viable alternative for users seeking enhanced security without sacrificing responsiveness.

2) Comparison with Open-Source UTM Distributions

Existing open-source UTM platforms such as pfSense, OPNsense, or Untangle provide rich security features, but their full capabilities often require x86 hardware with higher CPU and memory resources. Deploying these platforms on single-board computers typically results in reduced throughput, high CPU usage, or unstable IDS performance.

SUTMS provides several advantages in this context:

- Designed specifically for ARM-based hardware (Raspberry Pi 5)

- Lightweight configuration with protocol-aware optimization
- Reduced resource footprint without sacrificing detection relevance
- Modular, script-driven automation instead of heavy bundled services

While full-featured UTM distributions may still provide a broader range of enterprise tools, SUTMS offers a practical, highly optimized alternative that performs reliably within the limitations of Raspberry Pi hardware.

3) Comparison with Standalone IDS Deployments

Many hobbyists deploy Suricata or Snort alone on Raspberry Pi systems. Such standalone IDS deployments often suffer from:

Overloaded CPU during peak traffic

- High false-positive rates
- Packet drops due to full rulesets
- Lack of context-driven optimization
- No integrated threat-intelligence enforcement
- SUTMS resolves these limitations through:
- Dynamic rule optimization, reducing Suricata's workload
- Real-time protocol awareness provided by NTOPng
- Firewall integration for active blocking
- Better stability with AF_PACKET tuning and resource controls

Compared to standalone IDS setups, SUTMS performed significantly better in terms of packet-drop rates, CPU load, and actionable alert quality.

4) Overall Comparative Summary

Across all categories, SUTMS demonstrated a unique combination of:

- High detection relevance (due to rule optimization)
- Low resource usage (due to protocol-driven tuning)
- Effective proactive blocking (via TAXII-based IoCs)
- Strong visibility (due to NTOPng monitoring)
- Stable inline performance (near-router-level throughput)

These characteristics are typically not found together in existing low-cost solutions.
Thus, SUTMS occupies a practical middle ground—offering far more capability than consumer routers or standalone IDS deployments, while remaining lighter and more hardware-efficient than mainstream UTM platforms.

VIII. LIMITATIONS

A. HARDWARE-LEVEL CONSTRAINTS

While the Raspberry Pi 5 is significantly more capable than its predecessors, it still operates within the limits of ARM-based single-board hardware. Running NTOPng, Suricata, firewall processing, and monitoring services simultaneously consumes a large portion of available CPU cycles. Under high traffic loads, these constraints become more noticeable, especially when Suricata processes multiple concurrent flows. Memory availability also limits how many rule categories can be enabled at once and restricts the retention of large logs or flow histories. These limitations do not impact basic functionality but cap the system's ability to scale beyond home-network environments.

B. PACKET THROUGHPUT CHALLENGES

Operating SUTMS in an inline configuration means every packet must be captured, inspected, and forwarded through the Raspberry Pi. Although the optimized ruleset improves performance, the system can still encounter throughput drops during bandwidth-intensive tasks such as large file transfers, high-definition streaming, or stress traffic generated by tools like iPerf3. Short bursts of heavy traffic may momentarily raise packet latency or contribute to occasional packet drops, particularly when Suricata's detection threads are under load. These limitations are inherent to routing and inspection on resource-constrained hardware.

C. LIMITATIONS OF SIGNATURE-BASED IDS

Suricata's core detection mechanism relies on pre-defined signatures, which makes it inherently reactive. The IDS can only detect threats that are already known and accurately represented in the rule database. As a result, zero-day exploits, heavily obfuscated payloads, or polymorphic malware may evade detection. Because SUTMS prioritizes performance, advanced capabilities such as behavioural anomaly detection, machine learning models, or full deep-packet decoding for certain protocols are not enabled. This creates a trade-off between lightweight operation and comprehensive detection.

D. EXTERNAL THREAT FEED DEPENDENCIES

SUTMS relies on an external OpenTAXII server to aggregate Indicators of Compromise (IoCs) from multiple providers. The system's ability to enforce real-time blocking depends on the timely availability and freshness of these feeds. If the TAXII server experiences delays, downtime, or incomplete ingestion, SUTMS may operate temporarily with outdated threat data. Additionally, network connectivity issues on the Raspberry Pi's end can pause IoC updates. While these conditions do not affect base gateway functionality, they reduce the effectiveness of proactive threat prevention.

E. LACK OF TLS DEEP INSPECTION

For practical and ethical reasons, SUTMS does not perform TLS/SSL interception. Implementing full TLS decryption requires significant computational resources and introduces privacy concerns, making it unsuitable for a lightweight home-network UTM. Consequently, encrypted traffic is only analyzed at a metadata level—using features such as SNI fields, certificate information, traffic patterns, and endpoint reputation. Any malicious content hidden within encrypted channels cannot be inspected directly, leaving detection dependent on flow heuristics and threat-intelligence blocklists.

IX. CONCLUSION

A. SUMMARY

This work presented the design, implementation, and evaluation of the Secure Unified Threat Management System (SUTMS), a lightweight yet capable security framework tailored specifically for home networks. By leveraging a Raspberry Pi 5 as an inline gateway, the system demonstrated that advanced security functions traditionally limited to enterprise UTM appliances can be adapted to run efficiently on low-cost, resource-constrained hardware.

The proposed architecture integrates three core engines—NTOPng for flow-based monitoring, Suricata for deep packet inspection, and a custom TAXII-based workflow for threat-intelligence ingestion. One of the key strengths of the system lies in its protocol-driven rule optimization engine, which intelligently reduces Suricata's rule set based on real-time protocol usage, significantly lowering CPU and memory consumption. This makes continuous IDS operation feasible without compromising detection relevance.

Through an external OpenTAXII server, SUTMS incorporates up-to-date threat feeds and translates Indicators of Compromise into dynamic firewall rules, enabling proactive blocking of known malicious destinations. The system further integrates a lightweight dashboard and logging engine for real-time visibility into network behaviour and system health.

Experimental results demonstrated that SUTMS can maintain strong detection accuracy, stable throughput, and acceptable latency while operating entirely on a Raspberry Pi 5. The inline deployment proved reliable for real-world household traffic, and the optimization strategies greatly improved resource efficiency compared to conventional IDS deployments. While certain limitations remain—such as ARM hardware constraints and the absence of TLS deep inspection—the overall system provides a practical, affordable, and highly adaptive security solution for home-network environments.

In essence, SUTMS showcases that with thoughtful design and targeted optimization, it is possible to build a functional, multi-layered UTM system that balances performance, usability, and security, bringing enterprise-grade monitoring and protection capabilities into the home setting.

B. KEY FINDINGS

The evaluation of SUTMS revealed several important findings about its technical performance and overall feasibility as a lightweight UTM solution. First, the protocol-driven rule optimization strategy significantly reduced Suricata's resource consumption, allowing the IDS to run efficiently on Raspberry Pi 5 hardware without compromising detection relevance. The system consistently maintained low packet-drop rates and stable throughput even during periods of moderate network load.

Another key finding was the effectiveness of incorporating threat intelligence through the custom OpenTAXII server. The IoC ingestion pipeline reliably converted threat feeds into dynamic firewall rules, enabling the system to block malicious IPs and domains proactively. This approach reduced unnecessary alerts within Suricata and provided a strong first layer of defense.

Additionally, the inline gateway deployment proved both stable and practical. Despite operating as a full routing device with simultaneous monitoring, detection, and enforcement, the Raspberry Pi introduced only minor latency, and real-world usage remained unaffected. The combined operation of NTOPng, Suricata, and the firewall demonstrated that a multi-layered security architecture can function effectively even within limited hardware constraints.

C. PRACTICAL UTILITY FOR HOME NETWORKS

The results of this work highlight SUTMS's strong applicability in real home-network environments. Most household networks rely solely on basic router firewalls and lack deeper inspection or threat-intelligence capabilities. SUTMS bridges this gap by providing enterprise-grade monitoring and protection features on affordable hardware.

The system offers home users clear visibility into device behaviour, protocol usage, and unusual traffic patterns through its lightweight dashboard. Suricata's optimized inspection provides continuous detection of malware-related activity, scans, or suspicious outbound communication attempts, while the dynamic firewall blocks known malicious destinations without requiring user intervention.

Because SUTMS operates as an inline gateway, it protects every connected device—laptops, smartphones, smart TVs, and IoT devices—without requiring individual configuration. This is particularly valuable for homes with multiple IoT appliances, which often lack built-in security and are common targets for botnets and scanning campaigns.

Overall, SUTMS demonstrates practical value as a low-cost, modular, and adaptive security layer for residential networks, offering a meaningful upgrade over traditional consumer routers without the complexity or expense of commercial UTM devices.

D. FUTURE RESEARCH DIRECTIONS

While SUTMS demonstrates that a lightweight UTM solution can be effectively deployed on consumer-grade hardware, several avenues remain for further enhancement. One key direction is the incorporation of behavioural and anomaly-based detection mechanisms. Currently, detection relies primarily on signature matching, which limits visibility into zero-day attacks or subtle deviations in device behaviour. Introducing lightweight machine learning models or statistical anomaly detection may help identify threats that do not match existing signatures.

Another area for improvement is expanding support for encrypted traffic analysis. Although full TLS/SSL interception is not feasible on Raspberry Pi hardware, techniques such as JA3/JA4 fingerprinting, encrypted traffic classification, and certificate reputation checks could provide deeper insight into encrypted communication without compromising performance or privacy.

Future iterations of SUTMS could also improve scalability and ease of deployment. This includes developing more automated configuration scripts, containerizing system components, or providing a guided setup interface to make installation accessible to non-technical users. Additionally, support for external storage or remote log offloading would help overcome the inherent storage limitations of the Raspberry Pi.

Enhancing the dashboard to include richer analytics, long-term trend visualization, and interactive alert investigation tools would further strengthen usability. Integrating mobile notifications or remote monitoring features could also provide real-time awareness for users managing their home networks.

Finally, future work may explore extending SUTMS beyond home environments to small offices or multi-segment networks by evaluating performance on slightly more powerful ARM or low-power x86 platforms. This would help determine how the architecture scales and whether the optimization strategies remain effective as network complexity grows.

REFERENCES

[1] V. B. M. Yadav, K. S. Smruti Rekha, and S. K. Padhi, "SUTMS: Designing a Smart Unified Threat Management System for Home Networks," IEEE Access, vol. 10, pp. 70077–70093, 2022, doi: 10.1109/ACCESS.2022.3187691.

[2] A. Gupta, P. Singh, and R. Chaudhary, "Intrusion detection for IoT-enabled smart homes," IEEE Internet of Things Journal, vol. 9, no. 18, pp. 17245–17255, Sept. 2022.

[3] T. R. Kumar and S. N. Patel, "Phishing and malicious URL detection using Raspberry Pi for network-edge defense," IEEE Access, vol. 10, pp. 114532–114543, Dec. 2022.

[4] L. Chen, M. Hassan, and F. Wang, "Traffic trace artifacts from port mirroring in network monitoring," Computer Networks, vol. 203, p. 108647, June 2022.

[5] J. W. Park, K. Lee, and S. Cho, "Performance evaluation of open-source intrusion detection systems in high-speed networks," Journal of Network and Computer Applications, vol. 190, p. 103187, Apr. 2021.

[6] R. K. Das, A. Verma, and P. Mishra, "Survey on unified threat management (UTM) systems for home networks," IEEE Access, vol. 11, pp. 41205–41219, Apr. 2023.

[7] M. T. Nguyen, D. Lee, and E. Kim, "A comprehensive study on STIX and TAXII standards for cyber-threat intelligence exchange," IEEE Transactions on Information Forensics and Security, vol. 18, pp. 3051–3063, May 2023.

[8] S. Patil, N. Jadhav, and V. Borkar, “Automatic device classification using Arduino platform for network identification,” International Journal of Electrical and Computer Engineering, vol. 12, no. 4, pp. 3865–3872, Aug. 2022.

[9] K. P. Sinha and L. Rajendran, “IoT device classification using machine learning algorithms,” Sensors, vol. 22, no. 18, p. 7012, Sept. 2022.