

An Intelligent Monitoring System for DMVPN Based Networks

Dana Bassam Abdualлах, and Ibrahim Almerhag^[0000-0003-3759-8889]

Department of Computer Networks, Faculty of Information Technology,
University of Tripoli, Tripoli, Libya

{dana.abdualлах, i.almerhag}@uot.edu.ly

Abstract. This paper presents the design, implementation, and evaluation of a secure, scalable, and intelligent monitoring framework for Dynamic Multipoint Virtual Private Network (DMVPN) deployments. The proposed architecture adopts a layered model in which a Wide Area Network serves as the underlay, while DMVPN Phase 3 with multipoint Generic Routing Encapsulation (mGRE) tunnels protected by IPsec and IKEv2 forms the secure overlay. Enhanced Interior Gateway Routing Protocol (EIGRP) is deployed for dynamic route discovery and optimized spoke-to-spoke communication. A Python-based monitoring engine, built on Netmiko, collects operational telemetry through read-only SSH sessions, normalizes it into JSON, and applies rule-based detection logic against a healthy baseline. The system operates with a ten-second polling interval and is integrated with a centralized dashboard that visualizes tunnel status, routing adjacencies, and security associations. Experimental validation in GNS3 demonstrates that tunnel and routing failures are detected within a single monitoring cycle, achieving an estimated improvement of approximately 95% in fault detection speed compared to traditional manual approaches. The framework consolidates distributed CLI-based verification into a unified, automation-ready interface, supporting proactive enterprise network supervision.

Keywords: DMVPN, Network Automation, Intelligent Monitoring, IPsec, EIGRP.

1 Introduction

Modern enterprise organizations increasingly rely on Wide Area Networks (WANs) to interconnect geographically distributed branch offices, data centers, and remote users. Traditional point-to-point Virtual Private Network (VPN) solutions impose significant configuration and management overhead, and their scalability becomes problematic as the number of sites grows [1]. Dynamic Multipoint Virtual Private Network (DMVPN) addresses these limitations by enabling on-demand encrypted tunnels between sites through a single multipoint interface on the hub, supported by the Next Hop Resolution Protocol (NHRP) and dynamic routing [2].

<https://doi.org/10.5281/zenodo.23049173>

While DMVPN simplifies tunnel provisioning, it does not inherently solve the operational challenge of monitoring the resulting overlay. Tunnel health, NHRP registrations, IPsec security associations, and routing adjacencies must all remain stable for the network to function correctly, yet verifying these elements traditionally requires manual command-line execution across many devices. The lack of consolidated, real-time visibility delays fault identification and increases the burden on network administrators.

This work proposes an intelligent monitoring framework that integrates DMVPN Phase 3, EIGRP routing, and IPsec/IKEv2 security with a Python-based observability layer. The framework continuously collects structured telemetry from network devices, compares it against a healthy-state baseline, classifies anomalies by severity, and visualizes the results through a unified dashboard. The main contributions of this paper are: (i) a layered architecture that cleanly separates underlay transport, secure overlay, dynamic routing, and intelligent management; (ii) a rule-based, JSON-driven detection mechanism with deterministic and auditable behavior; (iii) the design of an operator-oriented dashboard that consolidates per-router protocol state, network-wide health, recent events, and actionable alerts on a single decision-support surface; and (iv) an empirical evaluation in GNS3 that quantifies the reduction in fault detection latency relative to manual troubleshooting.

The remainder of the paper is organized as follows. Section 2 reviews related work on DMVPN performance and network monitoring. Section 3 presents the system architecture. Section 4 describes the implementation in GNS3. Section 5 reports the experimental results. Section 6 concludes the paper and outlines future work.

2 Background and Related Work

2.1 DMVPN Fundamentals

DMVPN is a framework that combines multipoint Generic Routing Encapsulation (mGRE), NHRP, IPsec, and a dynamic routing protocol to create a flexible hub-and-spoke overlay capable of dynamic spoke-to-spoke tunnels [2, 3]. DMVPN operates in three phases that differ in how spoke-to-spoke traffic is handled. Phase 1 forces all inter-spoke traffic through the hub. Phase 2 supports direct spoke-to-spoke tunnels but requires each spoke to maintain a full routing table from the hub. Phase 3 introduces NHRP Redirect and Shortcut messages, enabling the hub to optimize traffic paths while keeping routing summarized, which scales effectively to large enterprise deployments [4]. DMVPN remains a widely deployed WAN overlay and continues to be treated in current multi-domain network design literature [15].

2.2 Routing and Security in DMVPN

Several studies have evaluated dynamic routing protocols over DMVPN. Marah and Ilyas [5] analyzed the impact of encryption techniques and dynamic routing protocols on DMVPN throughput and latency, concluding that EIGRP offers favorable convergence characteristics over multipoint interfaces while AES-based IPsec maintains acceptable overhead. Hairab et al. [6] examined dynamic routing protocol scalability over

DMVPN and confirmed that EIGRP behaves more predictably than OSPF in multipoint scenarios because OSPF requires careful next-hop and network-type tuning to operate correctly across mGRE. More recent work by Shahid et al. [16] confirms, in IPv6 load-sharing and link-failover scenarios, that EIGRP converges faster and loses fewer packets than OSPF after a link failure, which further supports its selection in this work.

Kaur and Gurm [7] simulated DMVPN with OSPF and EIGRP in GNS3 and reported similar findings, recommending EIGRP for enterprise-scale deployments. Bensalem and Boukhalfa [8] proposed a scalable DMVPN configuration validated against varying numbers of sites, observing that split-horizon and next-hop self-adjustments are mandatory on hub tunnel interfaces to enable Phase 3 shortcuts. Regarding security, Cisco's design guide [3] and Stallings [9] establish IPsec with IKEv2 as the standard for protecting GRE traffic over untrusted underlays, providing confidentiality, integrity, and authentication through Encapsulating Security Payload (ESP) and the Internet Key Exchange protocol. Recent independent comparisons of VPN protocols [17, 18] rank IKEv2/IPsec as the preferred option for enterprise deployments because of its authentication flexibility and native support in network operating systems, while showing that the throughput advantage of WireGuard is strongly context-dependent.

2.3 Network Monitoring and Automation

Network automation has shifted from device-centric scripts to programmable management layers built on libraries such as Netmiko and NAPALM [10]. Comprehensive network management practice further requires continuous polling, event correlation, and structured visualization of operational state, principles that have long been established in the network management literature [13]. Al Homsy [11] developed Network Guard, a Python-based monitoring tool that demonstrates how Scapy and structured logging can replace manual diagnostic workflows with continuous, real-time supervision. Azwee, Alkhattali, and Dow [12], in a study conducted at the High Institute of Science and Technology in Libya, surveyed VPN architectures used in mobile networks and emphasized the need for integrated security and monitoring approaches that consolidate fragmented operational data. Recent work confirms the practicality of Netmiko-based automation on production Cisco networks: Osei-Wusu et al. [19] reported a reduction of more than 99% in task completion time for automated configuration management, and Choi [20] documents the SSH-based collection patterns with paramiko and netmiko that the collector in this paper follows.

Despite the growing literature on DMVPN performance and on Python-based monitoring, few studies integrate the two into a single framework. Most existing DMVPN research focuses on routing protocol comparison or on encryption overhead, while monitoring studies typically address generic IP networks rather than overlay-specific signals such as NHRP registration and IPsec security association state. The framework proposed in this paper aims to fill that gap by treating DMVPN-specific telemetry as the primary input to a rule-based intelligent monitoring engine.

3 System Architecture

3.1 Layered Design

The proposed system follows a four-layer architecture that separates concerns and allows each layer to evolve independently. The underlay network provides basic IP connectivity between hub and spokes, simulating an untrusted public transport. The overlay network builds DMVPN Phase 3 multipoint GRE tunnels on top of this underlay. The routing and security layer enforces EIGRP for dynamic route exchange and IPsec/IKEv2 for tunnel protection. The management, automation, and intelligence layer comprises Python modules for data collection, rule-based detection, controlled automation, and dashboard visualization. Fig.1 below shows the DMVPN network layer and the monitoring/automation layer with collector script, detection engine, automation engine, dashboard, and the underlying JSON state files.

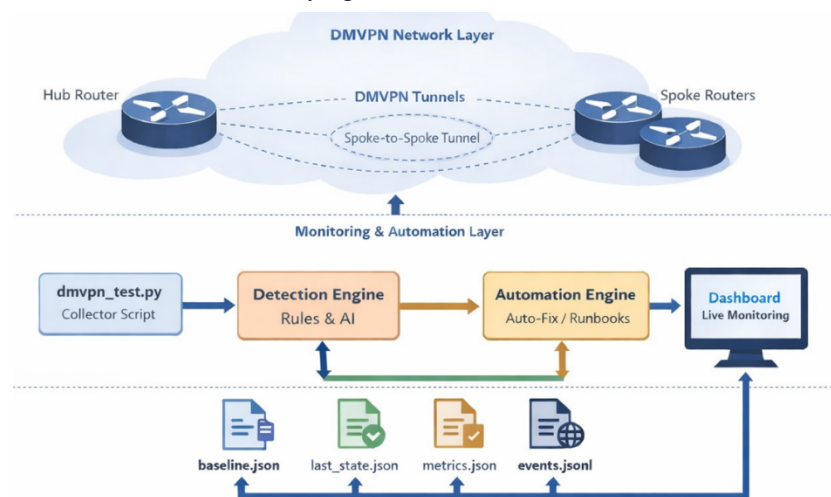


Fig. 1. High-level architecture of the proposed intelligent DMVPN monitoring system.

3.2 Selected Type of Multipoint VPN and DMVPN Phase 3 Design

The multipoint virtual private network adopted in this work is a Cisco Dynamic Multipoint VPN of the single-hub, single-cloud, Phase 3 type. Concretely, it is a site-to-site (router-to-router) overlay rather than a remote-access VPN; it uses a hub-and-spoke topology in which the hub exposes one IPv4 multipoint GRE (mGRE) tunnel interface and every spoke registers with the hub through NHRP, while direct spoke-to-spoke tunnels are created on demand by NHRP Redirect and Shortcut messages; all tunnel traffic is protected by IPsec in transport mode negotiated with IKEv2; EIGRP runs as the overlay routing protocol and OSPF as the underlay routing protocol; and a single DMVPN cloud, identified by one tunnel key and one NHRP network identifier, is used. Multi-hub (dual-cloud or hierarchical) DMVPN variants, which add hub redundancy,

are outside the scope of the present evaluation and are identified as future work in Section 6.

DMVPN Phase 3 was selected because it provides the greatest scalability and the most efficient traffic paths. The hub maintains a single mGRE interface and acts as the NHRP server, while spokes register their tunnel-to-NBMA mappings dynamically. When a spoke initially sends traffic toward another spoke through the hub, the hub responds with an NHRP Redirect, prompting the source spoke to issue an NHRP Resolution Request. Once the destination spoke replies, a direct spoke-to-spoke tunnel is established and subsequent packets bypass the hub. This optimization is reflected in the routing table by the next-hop override flag.

3.3 Routing and Security Design

EIGRP autonomous system 5 is deployed across the tunnel interfaces to advertise branch LAN networks. Two adjustments are required on the hub tunnel: split-horizon must be disabled so that routes received from one spoke can be re-advertised to others, and next-hop-self must be turned off so that spokes learn each other's tunnel addresses as next hops. These two settings are essential for Phase 3 shortcut formation. Security is provided by an IKEv2 profile using pre-shared key authentication and an IPsec transform set with AES-256 encryption and SHA-256 HMAC integrity. Tunnel protection binds the IPsec profile to the mGRE interface, encrypting all GRE-encapsulated traffic before it leaves the device.

3.4 Rationale for Selecting IPsec with IKEv2

Several mechanisms could, in principle, protect the overlay traffic: unencrypted GRE, IPsec negotiated with IKEv1, IPsec negotiated with IKEv2, TLS-based VPNs such as OpenVPN, WireGuard, and Cisco Group Encrypted Transport (GET) VPN. IPsec with IKEv2 was selected for four reasons. First, it is the only mechanism that binds natively to a multipoint GRE interface through the tunnel protection command on Cisco IOS, so a single IPsec profile automatically protects every dynamic spoke-to-spoke tunnel that NHRP creates; TLS-based and WireGuard tunnels are point-to-point, are not implemented in the IOS 15.2 image used, and cannot follow NHRP-driven tunnel creation. Second, IKEv2 improves on IKEv1 in ways that matter directly for a monitored overlay: the initial exchange is completed in 4 messages instead of 6 or 9, liveness checking (dead peer detection), NAT traversal and request acknowledgement are part of the base protocol, anti-denial-of-service cookies protect the hub, and EAP or certificate-based authentication can replace pre-shared keys without redesign [22]. Third, the selected suite (ESP with AES-256 encryption and SHA-256 HMAC integrity) matches the algorithms recommended for IPsec VPNs in NIST SP 800-77 Rev. 1 [21], and transport mode avoids a second IP header because GRE already supplies the tunnel encapsulation. Fourth, recent independent comparisons [17, 18] rank IKEv2/IPsec as the most suitable option for enterprise deployments owing to its authentication flexibility and native operating-system support, and show that the throughput advantage of WireGuard depends strongly on the deployment environment.

GET VPN was excluded because it preserves the original IP header and therefore requires a privately routable underlay, which contradicts the assumption of an untrusted public transport. Table 1 summarizes the comparison. In the laboratory a pre-shared key was used for simplicity; in production, the same IKEv2 profile can be switched to certificate (RSA-Sig) or EAP authentication without any change to the DMVPN design.

Table 1. Comparison of candidate mechanisms for protecting the DMVPN overlay.

Mechanism	Binds to mGRE / NHRP on IOS	Key exchange and liveness	Assessment
Unencrypted GRE	Yes	None	No confidentiality or integrity; rejected.
IPsec + IKEv1	Yes (tunnel protection)	6 or 9 messages; DPD and NAT-T as extensions	Legacy protocol; superseded by IKEv2 in current guidance [3, 21].
IPsec + IKEv2 (selected)	Yes (tunnel protection)	4 messages; built-in liveness, NAT-T, DoS cookies, EAP	Standards-based (RFC 7296) [22]; NIST-recommended AES-256/SHA-256 suite; selected.
TLS VPN (OpenVPN)	No (point-to-point, user space)	TLS handshake	Not available on IOS; highest processing overhead [17].
WireGuard	No (point-to-point)	Noise protocol with static keys	Not available on IOS; no dynamic peer discovery; performance gain is context-dependent [17, 18].
GET VPN	Group keying, no tunnels	GDOI key server	Requires privately routable underlay; unsuitable for Internet transport.

3.5 Intelligent Monitoring Layer

The monitoring layer is realized as a set of Python modules that connect to each network device using Netmiko over SSH. Diagnostic commands such as `show dmvpn`, `show crypto ikev2 sa`, `show crypto ipsec sa`, and `show ip eigrp neighbors` are issued in read-only mode and parsed into structured records. The most recent snapshot is written to a `metrics.json` file, while every detected change or alert is appended to an `events.jsonl` log. A rule-based detection engine then evaluates these metrics against a healthy baseline captured during normal operation.

Three severity levels are defined. CRITICAL events represent severe failures such as hub unreachability or complete tunnel loss. WARNING events represent partial degradations such as a single spoke disconnection or a drop in IPsec security associations. INFO events cover normal status confirmations and periodic heartbeats. Heartbeat

events are emitted at every cycle even when no anomaly is detected, which prevents silent monitoring failures from being misinterpreted as a healthy network. The engine runs continuously with a ten-second polling interval, performing six evaluation cycles per minute. Fig. 2 below illustrates the end-to-end flow from network devices through data collection, detection, and alert logging up to the automation and dashboard outputs.

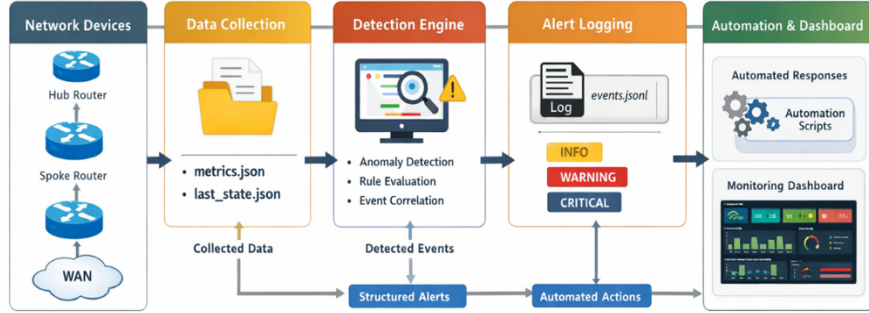


Fig. 2. End-to-end monitoring, detection, and automation pipeline.

4 Implementation

4.1 Topology and Tools

The complete topology was built in GNS3 version 2.2.55 [14] using Cisco 7200 routers running IOS 15.2. The deployment includes one hub router (HQ), two branch routers (BR1 and BR2), and an ISP router that simulates the public underlay. End-host connectivity is represented by virtual PCs and loopback interfaces. OSPF is used as the underlay routing protocol so that the public addresses of all routers can reach each other before any tunnel is built. Python 3.12 hosts the automation and monitoring modules, with Netmiko providing SSH-based device access and JSON/JSONL serving as the structured data layer. Table 2 below summarizes the tools and technologies used in the implementation, together with the role each plays in the project.

4.2 Selection of GNS3 as the Emulation Platform

Candidate platforms for validating the design fall into two classes [23]. Discrete-event simulators such as OMNeT++ (with the INET framework) and ns-3 model protocol behaviour with abstract numerical models and are well suited to reproducible performance studies of large topologies; network emulators such as GNS3, EVE-NG and Cisco Modeling Labs (CML) execute real router software and therefore reproduce the exact control-plane and command-line behaviour of production devices. GNS3 was selected for three reasons that follow directly from the objectives of this work. First, the design depends on vendor-specific protocol implementations, namely NHRP with Phase 3 Redirect and Shortcut, mGRE tunnel protection, IKEv2 and EIGRP; none of

Table 2. Tools and technologies used.

Tool/Technology	Purpose	Role
GNS3 v2.2.55	Network simulation	Emulates the full DMVPN topology with virtual Cisco IOS images.
Cisco IOS 15.2	Router OS	Provides NHRP, IPsec, EIGRP, and mGRE on hub and spokes.
DMVPN Phase 3	VPN framework	Enables scalable multipoint connectivity with dynamic shortcuts.
IPsec / IKEv2	Security	Encrypts GRE traffic using AES-256 and SHA-256.
EIGRP (AS 5)	Dynamic routing	Advertises branch LANs and supports Phase 3 shortcuts.
Python 3 + Netmiko	Automation & monitoring	Connects via SSH to collect structured telemetry.
JSON / JSONL	Data layer	Stores latest metrics and chronological event log.
Web Dashboard	Visualization	Renders tunnel, routing, and security state in real time.

Phase 3 Redirect and Shortcut, mGRE tunnel protection, IKEv2 and EIGRP; none of these is available as a ready model in OMNeT++/INET or ns-3, so a simulator would have required re-implementing and validating the protocols themselves instead of evaluating the monitoring framework. Second, the monitoring engine is built on Netmiko and consumes the textual output of show commands over SSH; a simulator exposes no such management plane, whereas GNS3 routers run genuine Cisco IOS and are reachable over SSH exactly as physical routers are, so the collector, detection rules and dashboard validated in GNS3 can be pointed at real hardware without modification. Third, among the emulators, GNS3 is open source and free, runs the Cisco 7200 image with the required feature set, supports Docker containers and virtual machines for the management node, offers per-link impairment filters (Section 4.3), and was used in the closest prior DMVPN studies [5, 7, 8], which makes the results directly comparable.

Cisco Packet Tracer was excluded because it does not implement mGRE/NHRP or IKEv2 and offers no external SSH access, while EVE-NG and CML were considered functionally equivalent but respectively require a dedicated server deployment or a commercial license. Table 3 summarizes the comparison. The trade-off accepted is the one common to all emulators: topology size is bounded by the CPU and memory of the host, and absolute timing figures reflect the host rather than router hardware.

Table 3. Comparison of candidate simulation and emulation platforms.

Platform	Class	DMVPN (mGRE/NHRP), IKEv2, EIGRP	Real CLI over SSH (Netmiko)	Licence
GNS3 (selected)	Emulator (real Cisco IOS)	Yes, native	Yes	Open source, free
EVE-NG	Emulator	Yes, native	Yes	Community / Pro; server-based
Cisco CML	Emulator	Yes, native	Yes	Commercial licence
Cisco Packet Tracer	Educational simulator	No mGRE/NHRP, no IKEv2	No external SSH	Free
OMNeT++ / INET	Discrete-event simulator	No models; must be implemented	No	Open source (academic)
ns-3	Discrete-event simulator	No models; must be implemented	No	Open source

4.3 Extensibility of the Emulated Environment and Handling of Lost Packets

The emulated environment is not closed: GNS3 allows additional elements to be attached to the same topology without altering the DMVPN configuration, and two classes of additions are relevant to packet loss. The first is loss injection. Every GNS3 link accepts packet filters (Frequency Drop, Packet Loss, Delay, Corrupt and Berkeley Packet Filter expressions), so a defined percentage of loss, latency or corruption can be applied to the underlay links to reproduce a degraded WAN [14]; alternatively, a Linux or Docker node running `tc/netem` can be inserted as a WAN emulator between the ISP router and a spoke. The second is loss processing and recovery. DMVPN routers, like any IP routers, do not retransmit lost data packets; recovery is provided end-to-end by the transport layer of the hosts (TCP), while the control plane already contains its own retransmission mechanisms: the EIGRP Reliable Transport Protocol retransmits unacknowledged updates and queries, NHRP re-sends registration and resolution requests until they are answered, IKEv2 retransmits requests with exponential back-off and detects a dead peer, and the IPsec anti-replay window absorbs reordering. Elements that can be added on the routers to act on loss include IP SLA probes with object tracking to measure loss and delay across each tunnel, QoS policies (LLQ, WRED) on the tunnel interface to protect critical traffic, and a second hub for failover.

From the monitoring perspective, the collector can be extended in the same JSON-driven way to poll tunnel interface input errors and drops, IPsec decrypt and replay failure counters, EIGRP retransmission counters and IP SLA statistics, and the detection engine can raise a WARNING when the measured loss exceeds a configurable threshold. The automation module can then trigger a controlled re-registration (`clear ip nhrp`) or re-keying (`clear crypto session`) as a verified action. These extensions were not part of the present evaluation, which focused on fault detection latency, and are listed in Section 6 as future work.

4.4 Number of Participating Stations and Scalability

The evaluated topology contains a fixed set of 7 enrolled nodes: 4 routers (the hub HQ, the spokes BR1 and BR2, and the ISP underlay router) and 3 end hosts (PC1, PC2 and PC3). Of these, the 3 routers that take part in the overlay (HQ, BR1 and BR2) send and receive tunnel traffic, and the 4 routers are polled over SSH in every cycle. This number was chosen as the minimum that exercises every DMVPN Phase 3 mechanism, namely hub-to-spoke registration, spoke-to-spoke shortcut formation and hub bypass, while keeping the fault-injection experiments reproducible.

The maximum number of stations is bounded by three independent limits. At the protocol level, a single Phase 3 hub with one mGRE interface can serve hundreds of spokes, the practical ceiling being set by the IPsec security-association capacity and NHRP cache of the hub platform rather than by the design, and multi-hub or hierarchical hub deployments extend this to thousands of sites [1, 15]. At the emulation level, each Cisco 7200 IOS instance consumes dedicated memory and CPU on the GNS3 host, so the number of routers that can run concurrently on a single workstation is on the order of tens rather than hundreds; larger topologies require a dedicated GNS3

server. At the monitoring level, the collector polls devices sequentially within the 10-second cycle; the 4 routers of the present topology are polled well within one cycle, and because the cycle time grows approximately linearly with the number of devices, larger deployments are handled by polling devices concurrently through a pool of SSH sessions, by lengthening the interval through the refresh control of the dashboard, or by deploying distributed collectors per hub.

Adding stations requires no change on the hub. A new spoke is configured with the same tunnel key, NHRP network identifier and next-hop-server address and registers itself dynamically, after which EIGRP advertises its LAN and Phase 3 shortcuts form on demand. On the monitoring side, the device inventory is a JSON list with no hard-coded maximum: adding a device means adding an entry with its management address and credentials and re-capturing the healthy baseline, and the search bar and per-device cards of the dashboard were designed for exactly this growth. Additional end hosts are attached to the branch LANs without any change to the overlay or to the monitoring engine.

4.5 DMVPN, Routing, and Security Configuration

Phase 3 is enabled by configuring `ip nhrp redirect` on the hub and `ip nhrp shortcut` on the spokes, with a tunnel key and network identifier of 10. EIGRP is bound to the tunnel network 10.10.10.0/24 and the LAN networks 200.x.x.0/24, with the WAN interface set as passive. The two Phase 3 EIGRP adjustments — `no ip split-horizon eigrp 5` and `no ip next-hop-self eigrp 5` — are applied on the hub tunnel interface. Security is enforced through an IKEv2 keyring with a pre-shared key, an IKEv2 profile that matches all remote identities, and an IPsec transform set using `esp-aes 256` and `esp-sha256-hmac` in transport mode. Tunnel protection binds this IPsec profile to the mGRE interface so that all GRE packets are encrypted before transmission.

4.6 Monitoring and Automation Modules

The monitoring engine is deployed on a dedicated management node that does not participate in routing or forwarding. Each cycle, the engine connects to the hub and the spokes, executes a fixed list of `show` commands, and parses the textual output into structured fields such as `dmvpn_peers`, `ipsec_sas`, `eigrp_neighbors`, and `device_status`. These fields are written to `metrics.json`. The detection module then loads the latest snapshot and evaluates a set of threshold rules. For example, a peer count of zero on a reachable device implies a tunnel failure, a sudden drop in IPsec security associations indicates a key negotiation problem, and a missing EIGRP neighbor signals control-plane instability. Each detected condition is logged to `events.jsonl` together with a timestamp, severity, device identifier, and the metric values that triggered it.

Automation in the current implementation is intentionally constrained to verification-oriented actions. When a CRITICAL or WARNING event is logged, the automation module reads the event, executes additional read-only verification commands on the affected device, and records the post-check state. No configuration changes are applied automatically. This conservative design ensures operational safety and aligns with

academic best practice, while leaving room for future extensions toward genuine self-healing. The interaction between these three components and the structured files they consume and produce is summarized in Table 4.

Table 4. Mapping between monitoring, detection, and automation components.

Component	Input Data	Output	Related File
Monitoring	Router CLI outputs	Metrics	metrics.json
Detection	Metrics	Alerts	events.jsonl
Automation	Alerts	Verifications	events.jsonl

5 Experimental Evaluation

5.1 Test Methodology

The evaluation focuses on five aspects: end-to-end LAN connectivity, EIGRP convergence, DMVPN Phase 3 shortcut formation, IPsec/IKEv2 stability, and the latency of the monitoring system in detecting injected faults. All tests were executed in the GNS3 environment described in Section 4. Faults were introduced by administratively shutting down tunnel interfaces, clearing IPsec security associations, and disconnecting the simulated underlay link of a spoke.

5.2 Connectivity, Routing, and Security

ICMP echo tests confirmed bidirectional connectivity between all branch LANs. EIGRP adjacencies formed correctly between the hub and each spoke, and remote LAN routes were learned without static configuration. Traffic between spokes initially traversed the hub; after the first NHRP Redirect, a direct spoke-to-spoke tunnel was established, indicated by the H (next-hop override) flag in the routing table. IPsec security associations were established in both directions with active inbound and outbound counters, and IKEv2 sessions remained stable throughout the test period. Encrypted packet counters increased consistently with traffic flow, confirming that all GRE-encapsulated packets were protected. Table 5 summarizes the connectivity, routing, and security results.

Table 5. Connectivity, routing, and security verification results.

Test Category	Description	Expected	Observed
LAN-LAN connectivity	Ping between branch LANs over the overlay.	Success	Success
EIGRP adjacency	Hub-spoke neighbor formation over mGRE.	Established	Established
Phase 3 shortcut	Spoke-to-spoke direct tunnel via NHRP Redirect.	Direct tunnel	Confirmed
IPsec/IKEv2	Active SAs and stable key negotiation.	Active	Verified
Traffic encryption	Encrypted GRE packet counters increasing.	Increasing	Confirmed

5.3 Fault Detection Latency

Four failure scenarios were injected to evaluate the monitoring system: loss of a DMVPN peer, tunnel interface shutdown, IPsec security association drop, and EIGRP neighbor loss. In each case the corresponding event appeared in `events.jsonl` and on the dashboard within a single ten-second monitoring cycle. The detection logic correctly classified each fault by severity, and the verification action recorded the post-fault device state without modifying the configuration.

Compared with manual troubleshooting, where an administrator typically needs several minutes to log into the relevant devices, run diagnostic commands, and correlate the output, the automated framework reduces the detection time to at most ten seconds. Taking a conservative manual baseline of three minutes, this represents an estimated improvement of approximately 95% in fault detection speed. More importantly, the framework consolidates information that would otherwise require separate sessions on multiple devices into a single dashboard view, lowering the cognitive load on the operator.

5.4 Dashboard Design and Operational Workflow

Beyond rapid fault detection, a key objective of the proposed framework is to translate raw telemetry into a presentation layer that supports observation, decision-making, and corrective action in a single workspace. The dashboard, named the Intelli Dashboard, was developed to give the operator a complete operational picture of the DMVPN deployment without the need to log into individual routers. It refreshes every ten seconds in synchronization with the monitoring loop and reads exclusively from the structured artifacts produced by the collector and detection modules, namely `metrics.json`, `last_state.json`, `anomalies.json`, and `events.jsonl`. This decoupling between data acquisition and visualization allows the dashboard to be replaced or extended without affecting the underlying monitoring logic.

The top-level header of the dashboard exposes three global controls that frame every operational session. A search bar allows the operator to filter the device cards by name, which is useful as the network grows. A refresh interval input lets the cycle period be tuned for different lab or production conditions, with a default of ten seconds. A start/stop control allows the loop to be paused for maintenance windows. Adjacent to these controls, a date and time panel anchored to the Africa/Tripoli timezone provides session context and confirms that the dashboard is rendering live data. These elements together ensure that every screenshot or shared view is unambiguous in time and scope. Fig. 3 shows the main view of the Intelli Dashboard: the Device Baseline panel enumerates all seven monitored nodes, the per-router cards expose Interfaces, DMVPN, NHRP Phase 3, EIGRP, OSPF, and IPsec/IKE state, and the right-hand column reports the live timestamp, the active device count, and the alerts summary.

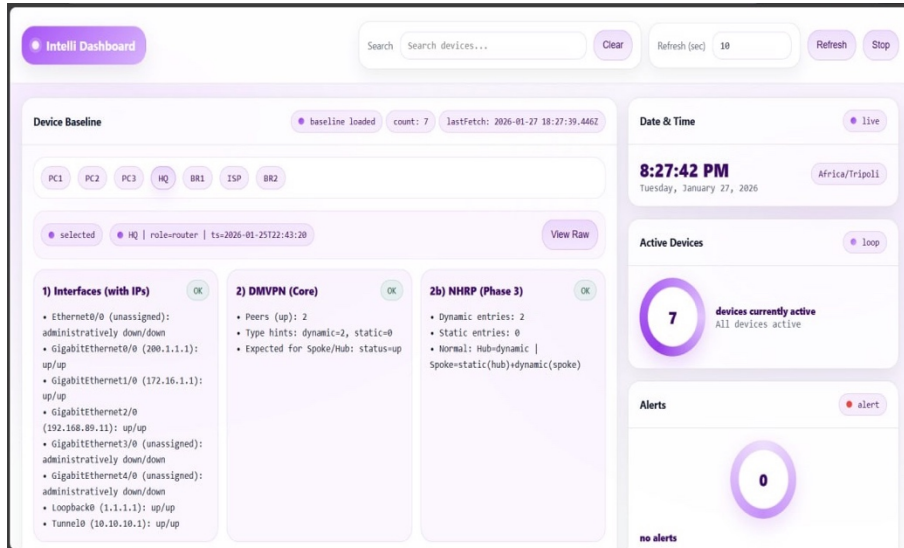


Fig. 3. Intelli Dashboard — main view.

The first major area of the dashboard is the Device Baseline panel, which lists every node discovered during the initial enrollment phase. In the configuration evaluated here it shows seven entries: three end-host PCs (PC1, PC2, PC3), the hub router (HQ), the two branches (BR1 and BR2), and the ISP router that simulates the public underlay. Selecting a device exposes its full live record and a View Raw control that opens the underlying JSON snapshot. The baseline also reports its own health — a baseline-loaded badge and the timestamp of the last successful fetch — so that an operator can distinguish a network problem from a stale or failed collection.

Per-router information is organized into six functional cards that mirror the layered architecture of Section 3 and explicitly identify the operational role of each protocol. The Interfaces card reports administrative and operational state for every physical port, the loopback, and the mGRE tunnel, including the assigned IPv4 address. The DMVPN Core card summarizes peer count, dynamic versus static type hints, and the expected baseline status for spoke or hub routers. The NHRP Phase 3 card breaks the entries down into dynamic and static counts and indicates whether the device is acting as a hub or as a spoke, which is the key signal that Phase 3 is forming correctly. The EIGRP Overlay and OSPF Underlay cards show neighbor counts against expected baselines, separating control-plane health on the tunnel network from health on the underlay. Finally, the Security card reports IKEv2 readiness, IPsec activity, and current encap/decap counters, allowing the operator to verify that traffic is actually being encrypted rather than merely that a tunnel is up. Each card carries a status badge (OK, WARN, or CRIT) so that protocol-level problems are visible at a glance even without opening the card.

To support rapid decision-making, the right-hand column condenses the global state into two indicators. The Active Devices card displays the number of devices currently reachable against the baseline total, accompanied by a textual summary such as “All

devices active” or “N of M reachable”. The Alerts card displays the count of active alerts; when alerts are present, the operator can expand them to see each alert’s severity, the originating device, the failing metric, and a short suggested action drawn from a small remediation knowledge base. The intent is that the operator should be able to move from “something is wrong” to “here is what to check” without leaving the dashboard. Fig.4 The Loop Status card reports liveness and a diff summary against the baseline, the Latest Events card shows the most recent severity-tagged events parsed from events.jsonl, and the Alerts card aggregates active issues with an actionable trigger.

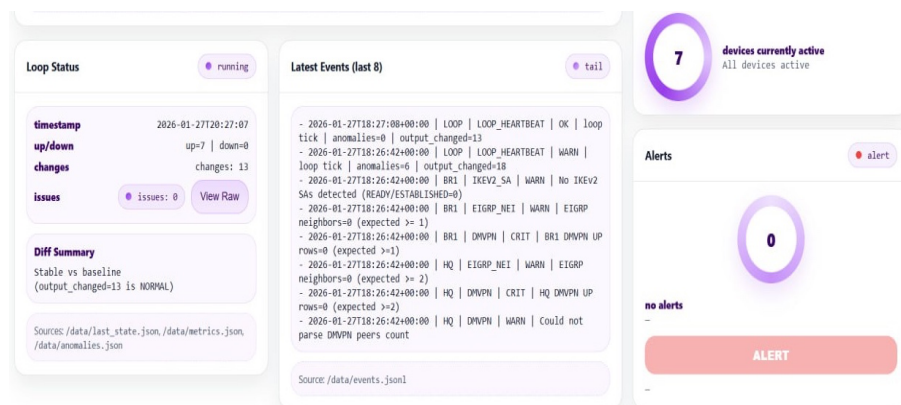


Fig. 4. Loop status, latest events, and alerts panel.

Below the per-device cards, the dashboard exposes three integrated panels that turn the system into a continuous operational record. The Loop Status panel reports the timestamp of the last cycle, the up/down counters across all devices, the number of changes observed against the baseline, and an issues counter; a small Diff Summary indicates whether the deviation from the baseline is normal (for example, a fixed number of expected output changes per cycle) or abnormal. The Latest Events panel renders the last several entries from events.jsonl in their canonical timestamped form, including the device, the metric category (such as DMVPN, EIGRP_NEI, IKEV2_SA), and the severity tag. Read together with the device cards, this panel allows the operator to correlate a CRIT or WARN event directly with the device card that triggered it. The Alerts trigger at the right collects unresolved issues and exposes a single ALERT control for downstream notification, which is the integration point reserved for future SIEM or messaging hooks.

Table 6 maps each dashboard area to the operational decision it supports. The intent is to demonstrate that the dashboard is not merely a visual aggregation of data but a decision-support surface, in which every panel contributes to a specific class of operator action.

Table 6. Mapping between dashboard areas and the operational decisions they support.

Dashboard Area	Information Presented	Operational Decision Supported
Header controls	Search, refresh interval, start/stop, live timestamp.	Scope and pace the monitoring session.
Device Baseline	Enrolled nodes and last fetch timestamp.	Confirm topology integrity and freshness of data.
Per-router cards	Interfaces, DMVPN, NHRP, EIGRP, OSPF, IPsec/IKE.	Identify which protocol or layer is failing.
Active Devices	Reachable count vs. baseline total.	Detect partial outages at a glance.
Loop Status	Liveness, up/down, changes, diff vs. baseline.	Verify monitoring health and rule out false alarms.
Latest Events	Recent timestamped, severity-tagged events.	Trace the sequence and root cause of an incident.
Alerts card	Active alerts with cause and suggested action.	Drive the immediate remediation step.

5.5 Discussion

Two observations follow from the experiments. First, the rule-based detection layer, despite its simplicity, is well suited to DMVPN because the relevant operational signals peer count, security association state, and neighbor count — are discrete and have clear thresholds. A trained machine learning model is not required to deliver useful intelligence at this scope. Second, the strict separation between monitoring and automation, with automation limited to verification rather than reconfiguration, made the system safe to run in an academic environment without risk of cascading failures. The collected JSON dataset can serve as training data for predictive models in future work. The dashboard, in turn, demonstrates that consolidating fragmented CLI outputs into a single decision-oriented surface materially changes how an operator perceives the network: events are no longer found by typing commands but by reading a layout that has been designed around the questions an operator actually asks.

6 Conclusion and Future Work

This paper presented an intelligent monitoring framework for DMVPN-based networks that combines a Phase 3 mGRE overlay, IPsec/IKEv2 security, EIGRP routing, and a Python-based observability layer. Experimental validation in GNS3 confirmed that the framework reliably detects tunnel, routing, and security failures within a ten-second polling cycle, providing an estimated 95% reduction in fault detection latency relative to manual workflows. The structured JSON telemetry, severity-based alert classification, and centralized dashboard together transform the operational view of a distributed DMVPN deployment from a set of fragmented CLI outputs into a unified, automation-ready interface.

Two main limitations should be acknowledged. The system was validated in a simulated environment, and real production traffic loads, hardware constraints, and external instability are not fully represented. The detection logic is rule-based and cannot identify previously unseen failure patterns. In addition, the evaluation covered a fixed topology of 7 nodes with 3 overlay routers and did not include controlled packet-loss

injection. Future work will extend the framework along four directions: integrating machine learning techniques for predictive fault analysis on the existing JSONL dataset; supporting multi-hub DMVPN deployments with a larger number of spokes, concurrent polling and distributed collectors; extending the detection rules to packet-loss and delay indicators (interface drops, IPsec replay failures and IP SLA statistics) under link impairment injected through GNS3 packet filters; and connecting the event log to enterprise platforms such as SIEM and IT service management systems for end-to-end operational integration.

References

1. Cisco Systems: Dynamic Multipoint VPN (DMVPN) Design Guide, Version 1.1. Cisco Systems, San Jose (2007).
2. Cisco Systems: Dynamic Multipoint VPN Configuration Guide, Cisco IOS XE 17. Cisco Systems, San Jose (2023).
3. Cisco Systems: Security for VPNs with IPsec Configuration Guide, Cisco IOS XE 17. Cisco Systems, San Jose (2023).
4. Edgeworth, B., Lacoste, R.: CCNP Enterprise Advanced Routing ENARSI 300-410 Official Cert Guide, 2nd edn. Cisco Press, Hoboken (2023).
5. Prajapati, D., Bowman, J., Suvarna, N.: Designing Real-World Multi-domain Networks. Cisco Press, Hoboken (2024).
6. Marah, H.M., Ilyas, M.: The Influence of Encryption Techniques and Dynamic Routing Protocols on DMVPN Network Performance. In: Proc. Int. Conf. on Engineering and Emerging Technologies (ICEET), pp. 1–6. IEEE (2021). <https://doi.org/10.1109/ICEET53442.2021.9659768>
7. Tizazu, G.A., Kim, K.-H., Berhe, A.B.: Dynamic Routing Influence on Secure Enterprise Network Based on DMVPN. In: Proc. 9th Int. Conf. on Ubiquitous and Future Networks (ICUFN), pp. 756–759. IEEE (2017). <https://doi.org/10.1109/ICUFN.2017.7993894>
8. Shahid, K., Ahmad, S.N., Rizvi, S.T.H.: Optimizing Network Performance: A Comparative Analysis of EIGRP, OSPF, and BGP in IPv6-Based Load-Sharing and Link-Failover Systems. *Future Internet* 16(9), 339 (2024). <https://doi.org/10.3390/fi16090339>
9. Kaur, H., Gurm, R.K.: Comparative Analysis of WAN Technologies. *International Journal of Computer Science Trends and Technology* 3(5), 9–20 (2015).
10. Masruroh, S.U., Widya, K.H.P., Fiade, A., Julia, I.R.: Performance Evaluation DMVPN Using Routing Protocol RIP, OSPF, and EIGRP. In: Proc. 6th Int. Conf. on Cyber and IT Service Management (CITSM), pp. 1–6. IEEE (2018). <https://doi.org/10.1109/CITSM.2018.8674051>
11. Stallings, W.: Network Security Essentials: Applications and Standards, 6th edn. Pearson, Harlow (2017).
12. Ahmadi, A.T., Aslamzai, S.: A Comparative Analysis of OpenVPN, WireGuard, and IKEv2/IPSec Using Ubuntu 24.04 LTS. *Journal of Advanced Computer Knowledge and Algorithms* 3(3), 96–102 (2026). <https://doi.org/10.29103/jacka.v3i3.27573>
13. Anyam, J., Singh, R.R., Larijani, H., Philip, A.: Empirical Performance Analysis of WireGuard vs. OpenVPN in Cloud and Virtualised Environments Under Simulated Network Conditions. *Computers* 14(8), 326 (2025). <https://doi.org/10.3390/computers14080326>
14. Byers, K.: Netmiko — Python Library for Network Automation (2023). <https://github.com/ktbyers/netmiko>, accessed 2025/11/15.

15. Clemm, A.: Network Management Fundamentals. Cisco Press, Indianapolis (2006).
16. Al Homsy, M.: Network Guard — A Python-Powered Network Monitoring Solution. M.Sc. Thesis, KTH Royal Institute of Technology (2024).
17. Azwee, K., Alkhattali, M., Dow, M.: Exploring the Effectiveness of VPN Architecture in Enhancing Network Security for Mobile Networks: An Investigation Study. *International Journal of Network Security & Its Applications* 15(5), 33–46 (2023).
18. Osei-Wusu, F., Asiedu, W., Frimpong, K.A., Yeboah, D., Sarfo, E.A., Muntaka, S.A.: Automating Network Programmability and Backup on Cisco Devices Using Python and Netmiko Library: A Case Study of Komfo Anokye Teaching Hospital LAN. *Journal of Computing Research and Innovation* 10(1), 227–242 (2025). <https://doi.org/10.24191/jcrinn.v10i1.488>
19. Choi, B.: Python Network Automation Labs: SSH in Action, paramiko and netmiko Labs. In: *Introduction to Python Network Automation Volume II*, pp. 121–227. Apress, Berkeley (2024). https://doi.org/10.1007/979-8-8688-0391-8_3
20. Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., Kivinen, T.: Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, IETF (2014). <https://doi.org/10.17487/RFC7296>
21. Barker, E., Dang, Q., Frankel, S., Scarfone, K., Wouters, P.: Guide to IPsec VPNs. NIST Special Publication 800-77 Rev. 1. National Institute of Standards and Technology, Gaithersburg (2020). <https://doi.org/10.6028/NIST.SP.800-77r1>
22. GNS3 Technologies: GNS3 Documentation and User Guide (2023). <https://docs.gns3.com>, accessed 2025/11/15.
23. Gomez, J., Kfoury, E.F., Crichigno, J., Srivastava, G.: A Survey on Network Simulators, Emulators, and Testbeds Used for Research and Education. *Computer Networks* 237, 110054 (2023). <https://doi.org/10.1016/j.comnet.2023.110054>

نظام مراقبة ذكي للشبكات المبنية على تقنية DMVPN

دانا بسام عبدالله و إبراهيم علي المرهق [0000-0003-3759-8889]

كلية تقنية المعلومات، جامعة طرابلس، طرابلس، ليبيا

{dana.abduallah, i.almerhag}@uot.edu.ly

المستخلص. يقدم هذا البحث تصميم وتنفيذ بنية تحتية شبكية آمنة وقابلة للتوسع وذات طابع ذكي لربط الفروع الجغرافية المتباعدة باستخدام تقنية الشبكات الافتراضية متعددة النقاط الديناميكية (DMVPN). تعتمد البنية المقترحة على تصميم طبقي، حيث تُستخدم شبكة واسعة النطاق (WAN) كطبقة نقل أساسية (Underlay)، بينما يتم إنشاء شبكة افتراضية آمنة (Overlay VPN) لربط الفروع عبر أنفاق GRE مؤمنة باستخدام بروتوكولي IPsec وIKEv2. كما تم تطبيق بروتوكولات توجيه ديناميكية لضمان كفاءة توجيه البيانات والاكتشاف التلقائي للمسارات وسرعة التقارب بين المواقع.

إلى جانب توفير الاتصال الآمن، يتضمن النظام إطارًا ذكيًا قائمًا على القواعد للمراقبة والأتمتة بهدف تحسين الرؤية التشغيلية وإدارة الأعطال. يعمل نظام المراقبة بدورة فحص زمنية مقدارها 10 ثوانٍ، أي بمعدل ست دورات تقييم آلية في الدقيقة الواحدة. وقد أظهرت التجارب العملية في بيئة محاكاة باستخدام GNS3 أن أعطال الأنفاق والتوجيه يتم اكتشافها خلال دورة مراقبة واحدة (بحد أقصى 10 ثوانٍ). وبالمقارنة مع أساليب المراقبة اليدوية التقليدية التي قد تستغرق عدة دقائق لاكتشاف العطل، يحقق النظام المقترح تحسنًا يُقدَّر بحوالي 95% في سرعة اكتشاف الأعطال.

علاوة على ذلك، يقوم الإطار المقترح بدمج إجراءات التحقق اليدوية المعتمدة على أوامر CLI عبر عدة أجهزة ضمن لوحة تحكم مركزية مؤتمتة، مما يلغي الحاجة إلى عمليات الفحص اليدوي الروتيني ويعزز الوعي التشغيلي بشكل ملحوظ. توفر لوحة التحكم عرضًا لحظيًا لحالة الأنفاق وعلاقات التوجيه والارتباطات الأمنية وسجل الأحداث، مما يدعم الإشراف الاستباقي على الشبكة.

بشكل عام، تبرهن النتائج على إمكانية بناء حل شبكي حديث قائم على DMVPN يدمج الاتصال الآمن، والتوجيه الديناميكي، والأتمتة، والمراقبة التشغيلية الذكية ضمن معمارية موحدة ومتينة.

الكلمات الدلالية: الشبكات الافتراضية الديناميكية متعددة النقاط ، أتمتة الشبكات ، المراقبة الذكية ، EIGRP ، IPsec