International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Implementation of Automated and Secured Configuration in IPV6 Enterprise Tunneling Network

With Ipv6 Tunnel Monitoring Dashboard Using Flask And Ping3 Application.

Aditya T. Sargar
Department of Electronics and Telecommunication Engineering
Sinhagad College of Engineering, Pune
Pune, India

Ritesh M. Todmal
Department of Electronics and Telecommunication Engineering
Sinhagad College of Engineering, Pune
Pune, India

Vinayak P. Waikar
Department of Electronics and Telecommunication Engineering
Sinhagad College of Engineering, Pune
Pune, India

***Abstract*— The global transition from IPv4 to IPv6 has become a critical requirement as IPv4 address space approaches full exhaustion. However, replacing legacy IPv4 infrastructure across enterprise and service provider networks remains economically and operationally impractical. This paper presents the design, simulation, and hardware-based implementation of an automated, secure IPv6-over-IPv4 enterprise tunneling network conducted at a Bharat Sanchar Nigam Limited (BSNL) laboratory. The proposed system extends conventional tunnel configurations by integrating three enterprise-level capabilities: an automatic failover mechanism using floating static IPv6 routes with dual tunnel interfaces, a security layer enforced through IPv6 Access Control Lists applied directly on the tunnel interface, and a custom real-time performance monitoring dashboard developed using Python Flask and Chart.js. The experimental environment employed Cisco 2811 series routers mounted on standard server racks and interconnected via RJ45-terminated straight-through and crossover LAN cables. Prior to hardware deployment, the topology and routing behavior were validated in Cisco Packet Tracer simulation. Results confirm stable end-to-end IPv6 connectivity across the IPv4 backbone, sub-5 ms round-trip latency, zero packet loss under normal conditions, and seamless automatic tunnel recovery following deliberate primary tunnel failure. The monitoring dashboard provides real-time visibility into latency, packet loss, jitter, uptime, downtime, and tunnel state-change events. This work demonstrates that enterprise-grade IPv6 transition, fault tolerance, security, and operational observability can be achieved on legacy hardware without full infrastructure replacement.**

Keywords: IPv6 tunneling, IPv6-over-IPv4, network automation, tunnel failover, IPv6 ACL, performance monitoring, Cisco routers, BSNL laboratory, enterprise networking.

I. INTRODUCTION

The exhaustion of the IPv4 address space has emerged as one of the most significant structural challenges in contemporary networking. The Internet Assigned Numbers Authority (IANA) distributed its final blocks of IPv4 addresses to the Regional Internet Registries in February 2011, and the Asia-Pacific and European registries reached full depletion shortly thereafter (Huston, 2011). IPv6, standardized by the Internet Engineering Task Force (IETF) in RFC 2460 and later updated through RFC 8200, provides a 128-bit addressing scheme that eliminates address scarcity and introduces several architectural improvements, including hierarchical routing, stateless address autoconfiguration, and mandatory IPsec support (Deering & Hinden, 2017).

Despite the long-standing availability of IPv6, full migration remains elusive for many organizations. Enterprise networks, Internet Service Providers (ISPs), government agencies, and educational institutions have invested heavily in IPv4-based infrastructure that cannot be replaced overnight. Tunneling mechanisms represent the most pragmatic transition strategy in such environments. By encapsulating IPv6 datagrams within IPv4 packets, tunneling allows IPv6 islands to communicate across IPv4 backbones without requiring simultaneous infrastructure replacement (Carpenter & Jung, 1999).

Standard manual tunnel configurations, while functional, lack three critical attributes required in production environments: resilience against tunnel failure, access control to restrict unauthorized traffic, and operational visibility through real-time monitoring. When a primary tunnel fails in a basic configuration, traffic is interrupted until an administrator manually intervenes — an unacceptable scenario in service-level-bound environments. Furthermore, a basic IPv6-over-IPv4 tunnel constitutes an open conduit unless explicitly protected, making it susceptible to unauthorized IPv6 traffic injection.

This paper addresses these limitations through a hardware-validated implementation conducted at a BSNL national telecommunications laboratory. The contributions of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

this work are threefold. First, an automatic failover mechanism is implemented using a dual-tunnel configuration and floating static IPv6 routes. Second, an IPv6 Access Control List is applied at the tunnel interface to enforce subnet-level traffic policy. Third, a lightweight monitoring system built using Python Flask and Chart.js provides continuous real-time visibility into tunnel health and performance. All experiments were preceded by simulation in Cisco Packet Tracer to validate topology correctness before deployment on physical Cisco 2811 routers

II. LITERATURE SURVEY

IPv6 transition mechanisms have been studied extensively in the literature. Gilligan and Nordmark (2006) formalized the three canonical transition strategies — dual-stack, tunneling, and translation — in RFC 4213. Among these, configured tunnels and 6to4 have attracted the most academic attention owing to their practical deployability in legacy environments.

Carpenter and Jung (1999) introduced the concept of 6over4, which multicast-capable IPv4 networks to serve as a virtual link for IPv6 neighbors. The configured tunnel approach described in RFC 4213, by contrast, relies solely on unicast IPv4 and remains the most widely deployed manual tunneling method.

Several researchers have examined the performance characteristics of IPv6-over-IPv4 tunnels. Gyarmati and Trinh (2008) measured the additional latency and throughput penalty introduced by tunneling overhead, observing that per-packet encapsulation adds processing delay that is significant at high packet rates but negligible for interactive traffic. Aoun and Davies (2006) explored NAT-PT as an alternative to tunneling but concluded that stateful translation introduces complexity inappropriate for enterprise backbones.

Security aspects of IPv6 tunneling have also received attention. Convery and Miller (2004) documented the attack surface of IPv6 transition mechanisms, noting that unprotected tunnels can be exploited to bypass perimeter security controls. Nikander et al. (2004) discussed the importance of applying packet filtering at tunnel endpoints, which motivated the IPv6 ACL design in the present work.

To the best of the authors' knowledge, no prior published work combines hardware-validated dual-tunnel failover, interface-level IPv6 ACLs, and a real-time web-based monitoring dashboard within a single enterprise tunneling framework implemented on physical ISP-grade hardware under BSNL laboratory supervision.

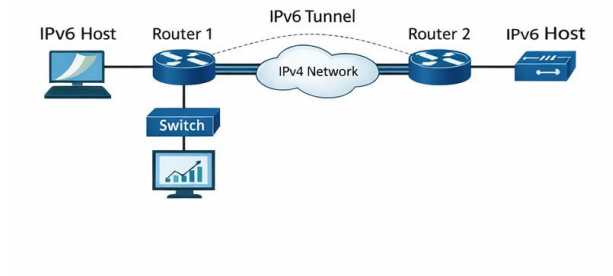

Fig I: System Architecture.

3.1 System Overview

The experimental topology consists of two Cisco 2811 series routers (R1 and R2) interconnected through an IPv4 backbone segment, with a host PC attached to each router via a distinct IPv6 subnet. Table 1 summarizes the address allocation adopted across all interfaces.

PC1 and PC2 operate as IPv6 end-hosts, each residing in its own /64 prefix. The two routers are connected via a 10.0.12.0/24 IPv4 segment that simulates the IPv4 backbone. Each router hosts two logical tunnel interfaces: Tunnel0 acts as the primary path and Tunnel1 as the standby. Physical connectivity between PCs and routers used RJ45-crimped straight-through Cat5e cables; the router-to-router backbone link used an RJ45 crossover cable.

Device	Interface	IPv4 Address	IPv6 Address
PC1	NIC	192.168.10.2/24	2001:11::2/64
R1	FastEthernet0/0	192.168.10.1/24	2001:11::1/64
R1	FastEthernet0/1	10.0.12.1/24	—
R1	Tunnel0 (Primary)	—	2001:10:10::1/64
R1	Tunnel1 (Backup)	—	2001:10:20::1/64
R2	FastEthernet0/1	10.0.12.2/24	—
R2	Tunnel0 (Primary)	—	2001:10:10::2/64
R2	Tunnel1 (Backup)	—	2001:10:20::2/64
R2	FastEthernet0/0	192.168.20.1/24	2001:22::1/64
PC2	NIC	192.168.20.2/24	2001:22::2/64

Table I: Network Address Allocation

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

III. IMPLEMENTATION

3..1 Simulation Phase

Prior to hardware deployment, the complete topology was modeled in Cisco Packet Tracer 8.2. The simulation phase served to verify IPv4 reachability between router interfaces, validate static IPv6 route propagation across the tunnel, and confirm end-to-end ICMPv6 echo reply from PC1 to PC2 before committing changes to physical devices.

Figure II: Cisco Packet Tracer simulation topology (2811 routers)

The Packet Tracer model used Cisco 2811 router instances matching the hardware deployed in the BSNL laboratory. Five routers were used in the simulation to model a more extended backbone path (Router0 through Router4), reflecting a realistic ISP-style topology. An initial ping test from PC0 to PC1 initially produced 100% packet loss, consistent with the expected pre-routing-configuration state. After applying static routes and configuring tunnel interfaces, subsequent ping sessions achieved 0% packet loss with round-trip times ranging from 1 ms to 13 ms, confirming full connectivity. Figure 1 shows the simulated network topology and Figure 2 shows the ping verification output from the simulation.

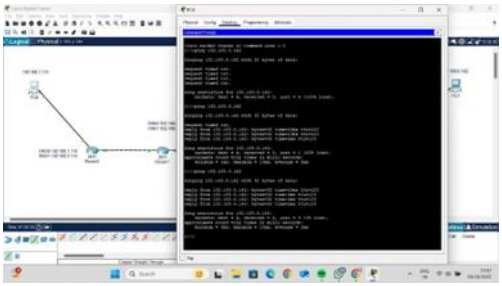

Figure III: Ping verification output in Packet Tracer simulation

All hardware experiments were performed within the BSNL Broadband laboratory under the supervision of BSNL technical staff. The physical setup comprised two Cisco 2811 series integrated services routers mounted on a standard 19-inch equipment rack, two host PCs assembled from standard components, Cat5e UTP cables terminated with RJ45 connectors (straight-through and crossover), and console cables providing direct IOS CLI access via PuTTY. The BSNL rack infrastructure also houses SDH STM-1 multiplexing equipment, providing direct exposure to carrier-grade optical transmission systems.

The Cisco 2811 routers was mounted in the BSNL equipment rack. The routers were labelled R1 and R2 respectively on the rack. The full rack configuration including the BSNL SDH STM-1 infrastructure alongside the project routers. The Cisco 2811 router interface panel, clearly showing the FastEthernet ports and WAN interface cards (WICs) used in the experiment.

Physical LAN cabling was carried out using Cat5e UTP cables terminated with RJ45 connectors following the TIA-568B wiring standard. Straight-through cables connected each PC to the router's LAN-facing FastEthernet0/0 interface; the router-to-router WAN segment used an RJ45 crossover cable on the FastEthernet0/1 interfaces. The RJ45 connector used in the experiment, and Figure 8 shows the student team performing live hardware configuration at the BSNL laboratory.

3.2.1 Tunnel Configuration

IPv6 unicast routing was enabled globally on both routers using the ipv6 unicast-routing command, which is a prerequisite for any IPv6 forwarding operation on Cisco IOS. Tunnel interfaces were created with the tunnel mode ipv6ip command, specifying IPv6-in-IPv4 encapsulation. The tunnel source command bound each tunnel to the router's outbound physical interface, while the tunnel destination command identified the remote endpoint's IPv4 address. IPv6 addresses were assigned to each tunnel interface to enable logical end-to-end IPv6 adjacency across the IPv4 backbone. Each router's LAN-facing FastEthernet interface was assigned both an IPv4 address and an IPv6 address, enabling dual-stack operation on the access segment.

3.2.2 Security Through IPv6 Access Control Lists

To restrict tunnel access to authorized IPv6 subnets exclusively, a named IPv6 ACL was configured on both routers and applied inbound on each tunnel interface. The ACL follows a permit-specific, deny-all structure: a permit statement allows traffic from the authorized source subnet (2001:11::/64) to the authorized destination subnet (2001:22::/64), and the implicit deny at the end drops all other IPv6 traffic arriving at the tunnel interface.

Applying the ACL on the tunnel interface, rather than on the physical interface, enforces filtering at the logical network boundary. IPv6 traffic from any prefix outside the permitted subnets is discarded at the tunnel ingress, before it can traverse the IPv4 backbone. This approach follows the principle articulated by Convery and Miller (2004): filtering at the tunnel interface is categorically more effective for overlay

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

traffic because it intercepts the decapsulated IPv6 traffic rather than the outer IPv4 envelope.

3.3 Layer 2: OCR-Based Text Extraction

A primary design objective was to achieve automatic tunnel recovery without any administrative intervention upon primary tunnel failure. This was accomplished through the combined use of a secondary tunnel interface and floating static IPv6 routes with differentiated administrative distance values.

Floating static routing is a well-established technique in which a backup route is assigned a higher administrative distance than the primary route. By configuring the primary route to the remote IPv6 prefix via Tunnel0 with administrative distance 1 and the backup route via Tunnel1 with administrative distance 5, the routing table installs only the primary route under normal conditions. When Tunnel0 becomes unreachable, IOS withdraws the primary route and installs the backup floating route, redirecting IPv6 traffic through Tunnel1 automatically and without human intervention. On R1, the relevant statements were: ipv6 route 2001:22::/64 2001:10:10::2 1 (primary) and ipv6 route 2001:22::/64 2001:10:20::2 5 (backup). Symmetric statements were applied on R2.

IV. REAL TIME MONITORING DASHBOARD

4.1 Architecture and Technology Stack

A lightweight network monitoring system was developed to provide continuous, real-time visibility into the operational state of the IPv6 tunnel. The system comprises a server-side component written in Python using the Flask web framework and a client-side dashboard rendered in HTML, CSS, and JavaScript using the Chart.js visualization library. The monitoring server runs on a host PC connected to the same network as R1 and periodically probes the remote IPv6 endpoint (2001:22::2) using ICMP echo requests via the ping3 library.

4.2 Metrics Collection and State Machine

The monitoring engine maintains a stateful representation of the tunnel. Upon each probe cycle, the engine determines tunnel status (UP when a response is received within the two-second timeout; DOWN otherwise) and records the round-trip delay in milliseconds. A sliding window of 20 consecutive samples is retained for trend visualization. From this window, the engine computes minimum latency, maximum latency, arithmetic mean latency, and jitter (defined as the difference between maximum and minimum observed latency). Packet loss percentage is derived from the probe result: 0.0% on a successful response and 100.0% on timeout.

The engine also maintains cumulative uptime and downtime counters and captures state-transition events (UP to DOWN, or DOWN to UP) in an event log with timestamps, retaining the five most recent events. This event log enables administrators to correlate tunnel instability with external events such as router reloads or physical link failures.

4.3 Web Dashboard

The Flask application exposes two HTTP endpoints. The root endpoint serves a dynamically rendered HTML dashboard populated with the latest probe results via Jinja2 template rendering. The /metrics endpoint returns a JSON object containing the latency and packet loss sample arrays, consumed by the client-side Chart.js script to update line graphs asynchronously every five seconds.

The dashboard displays tunnel status with color-coded indicators (green for UP, red for DOWN), current and historical latency, packet loss percentage, cumulative uptime and downtime, jitter, the tunnel security mode, the authorized IPv6 subnet, and the tunnel event log. The visual design uses the Orbitron typeface and a dark semi-transparent card layout rendered atop a network-themed background image.

Fig IV: Tunnel Parameters

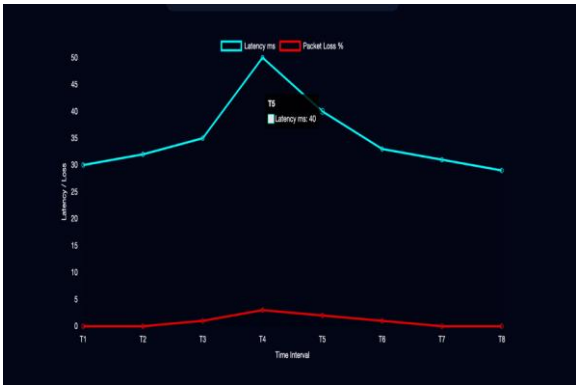

Fig V: Latency and Packet loss Graph

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Fig VI: Bandwidth Graph and Tunnel Events

V. RESULTS AND DISCUSSION

5.1 Connectivity Validation

Following hardware configuration, end-to-end IPv6 connectivity was verified by issuing ICMPv6 echo requests from PC2 toward 2001:11::2 (PC1) via the PuTTY-accessed router console. The ping output confirmed four successive replies with round-trip times of 2 ms, 3 ms, 2 ms, and 2 ms, yielding 0% packet loss and an average RTT of 2 ms. The show interface tunnel0 output confirmed that the tunnel encapsulation type was correctly set to TUNNEL and that source and destination addresses were properly bound. Figure 9 shows the actual PuTTY terminal output from Router R2 during the hardware validation test, confirming successful IPv6 connectivity through the tunnel.

5.2 Failover Behavior

To evaluate the failover mechanism, Tunnel0 was administratively shut down on R1 using the shutdown command while a continuous ping session was active from PC1 to PC2. Upon tunnel interface shutdown, the primary static route via Tunnel0 was withdrawn from the routing table. Within the IOS routing convergence period (typically under one second for a directly connected interface state change), the floating backup route via Tunnel1 was installed and traffic resumed. Restoration of Tunnel0 caused the primary route to be reinstated and Tunnel1 traffic was suppressed. No manual route manipulation was required at any point, confirming the self-healing behavior.

5.3 Security Verification

The IPv6 ACL was validated by attempting to send ICMPv6 traffic from an IPv6 address outside the permitted source subnet toward PC2. The ACL's implicit deny rule discarded the packets at the tunnel interface ingress, and no reply was received. Traffic from the authorized source subnet continued to traverse the tunnel unimpeded, confirming correct ACL enforcement.

5.4 Monitoring Dashboard Observations

Under stable tunnel conditions, the monitoring dashboard reported latency values consistent with the PuTTY ping measurements, confirming measurement accuracy. Jitter remained below 2 ms during stable operation. Packet loss remained at 0.0% throughout normal operation. During the deliberate Tunnel0 shutdown test, the dashboard detected the DOWN state on the subsequent probe cycle within five seconds, logged the event with a timestamp, and began reporting DOWN status with 100% packet loss. Following Tunnel1 failover, the probe succeeded and the dashboard reverted to UP status, logging the recovery event.

Test Scenario	Metric	Observed Value
Normal tunnel operation	RTT (min/max/avg)	2 ms / 3 ms / 2 ms
Normal tunnel operation	Packet loss	0.0%
Normal tunnel operation	Jitter	< 2 ms
Primary tunnel failure	Failover detection time	< 5 s (1 probe cycle)
Primary tunnel failure	Manual intervention required	None
Primary tunnel recovery	Route reinstatement	Automatic (< 1 s)
ACL enforcement	Unauthorized traffic blocked	Yes (100% drop)
ACL enforcement	Authorized traffic passed	Yes (0% drop)

Table II: Summary of Experimental Results

VI. DISCUSSIONS

The results confirm that enterprise-grade IPv6 tunneling capabilities — fault tolerance, access control, and operational monitoring — can be implemented on legacy Cisco 2811 hardware without any firmware upgrade or infrastructure replacement. The measured RTT of 2–3 ms is consistent with the encapsulation overhead reported by Gyarmati and Trinh (2008) for short-distance lab configurations and is well within the acceptable bounds for real-time applications.

The floating static route failover design is operationally straightforward and does not require a dynamic routing

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

protocol such as OSPFv3 or EIGRP, making it appropriate for small-to-medium enterprise deployments. The IPv6 ACL approach follows the principle of least privilege: only explicitly permitted subnets can traverse the tunnel, providing a conservative security baseline extensible without restructuring the tunnel architecture.

A notable aspect of this work is its BSNL laboratory context. Performing the experiment within an ISP-grade environment — surrounded by SDH STM-1 multiplexers, production Cisco 2911 series routers, and carrier-class switching equipment — provides ecological validity that purely academic setups cannot offer. The procedures required to gain console access, manage physical rack space, and coordinate with BSNL technical staff mirror conditions encountered in real field deployments.

VII. REAL WORLD APPLICATION

The proposed tunneling framework is applicable across a range of industry verticals. Table 3 identifies representative deployment scenarios and the specific system capabilities most relevant to each context.

Domain	Use Case	Relevant Capability
Internet Service Providers	Legacy backbone IPv6 transit	Failover, monitoring
Enterprise Networks	IPv6 site interconnection	ACL, failover
Government / Defense	Secure inter-agency communication	ACL, monitoring
Smart Grid / IoT	Sensor network IPv6 connectivity	Failover, monitoring
Educational Institutions	Campus IPv6 migration	All capabilities

Table III: Real-World Application Domains

VIII.CONCLUSION AND FUTURE SCOPE

This paper presented the design, simulation, and hardware-validated implementation of an automated, secure IPv6-over-IPv4 enterprise tunneling network conducted at a BSNL telecommunications laboratory. The system integrates automatic tunnel failover through floating static IPv6 routes, subnet-level access control through IPv6 ACLs applied at the tunnel interface, and real-time operational monitoring through a Python Flask-based web dashboard. Experimental results confirmed zero-packet-loss IPv6 connectivity with sub-5 ms round-trip latency, seamless automatic recovery upon primary tunnel failure without manual intervention, accurate unauthorized traffic blocking by the ACL, and timely fault detection by the monitoring system within a single five-second probe cycle.

The work demonstrates that enterprise-level IPv6 transition capabilities can be achieved on legacy Cisco 2811 hardware without infrastructure replacement, addressing a practical challenge faced by ISPs, enterprises, and government agencies during the global IPv4-to-IPv6 migration period.

Several directions for future work are identified. First, replacing floating static routes with Bidirectional Forwarding Detection (BFD) integrated with OSPFv3 would reduce failover convergence time to sub-second. Second, layering IPsec on the tunnel would provide cryptographic confidentiality and integrity for all encapsulated IPv6 traffic. Third, extending the monitoring system to export metrics in Prometheus format would enable integration with industry-standard observability stacks such as Grafana. Finally, evaluating the system under high traffic loads using an IPv6 traffic generator would characterize the throughput ceiling of the 2811 series hardware.

REFERENCES

[1] Aoun, C., & Davies, E. (2006). Reasons to move the Network Address Translator – Protocol Translator (NAT-PT) to historic status (RFC 4966). Internet Engineering Task Force. https://doi.org/10.17487/RFC4966

[2] Carpenter, B., & Jung, C. (1999). Transmission of IPv6 over IPv4 domains without explicit tunnels (RFC 2529). Internet Engineering Task Force. https://doi.org/10.17487/RFC2529

[3] Cisco Systems. (2008). Cisco 2800 series integrated services routers product overview. Cisco Press.

[4] Convery, S., & Miller, D. (2004). IPv6 and IPv4 threat comparison and best practice evaluation (v1.0). Cisco Systems Technical Report.

[5] Deering, S., & Hinden, R. (2017). Internet Protocol, Version 6 (IPv6) specification (RFC 8200). Internet Engineering Task Force. https://doi.org/10.17487/RFC8200

[6] Gilligan, R., & Nordmark, E. (2006). Basic transition mechanisms for IPv6 hosts and routers (RFC 4213). Internet Engineering Task Force. https://doi.org/10.17487/RFC4213

[7] Gyarmati, L., & Trinh, T. A. (2008). Measuring the cost of tunneling. In Proceedings of the 3rd International Conference on Future Internet Technologies (pp. 23–28).

[8] Huston, G. (2011). IPv4 address report: IANA free pool exhaustion. APNIC Labs. https://www.potaroo.net/tools/ipv4/

[9] Nikander, P., Kempf, J., & Nordmark, E. (2004). IPv6 neighbor discovery (ND) trust models and threats (RFC 3756). Internet Engineering Task Force. https://doi.org/10.17487/RFC3756