Volume 11, Issue 6, June – 2026 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

BGP-Route Monitoring and Visualization Tool

Kudzai Chiomba[1]; Macdonald Mukosera[2]

[1,2] Department of Electronics and Telecommunications
University of Zimbabwe, Harare, Zimbabwe

Publication Date: 2026/07/18

Abstract: The Border Gateway Protocol (BGP) forms the foundation of global internet routing, yet it contains no native mechanism to verify that an autonomous system (AS) is authorised to announce a given IP prefix. This fundamental weakness enables two critical attack classes: prefix hijacking, where an unauthorized AS announce ownership of an IP block it does not legitimately control, and route leaking, where routes are propagated with abnormally inflated AS paths. Present monitoring frameworks – including BGPStream, RIPE Stat, and BGPlay – provide data collection and historical visualisation capabilities, though they lack integrated, real-time operator guidance. This paper presents lightweight BGP Route Monitoring and Visualization Tool built on the Detect-Explain-Act (DEA) framework, validated on a six-router EVE-NG topology. The tool polls a monitoring router via Telnet, parses the BGP routing table, applies two rule-based detection algorithms, and enriches each anomaly with a plain-language explanation and prioritized remediation steps displayed on a Flask web dashboard. Experimental testing across six attack cases – including cross-ISP hijacks, provider-level prefix theft, cascade route leaks, and compound attacks – achieved 100% detection rate, 0% false positive rate on a verified clean baseline, and sub-30-second detection latency. The tool outperforms all three comparison frameworks on route leak detection, operator explanation, and effective guidance.

Keywords: *BGP Security; Prefix Hijacking; Route Leaking; Anomaly Detection; Real-Time Monitoring; Detect-Explain-Act; AS Path Analysis; Network Visualization; Internet Routing Security.*

How to Cite: Kudzai Chiomba; Macdonald Mukosera (2026) BGP-Route Monitoring and Visualization Tool. *International Journal of Innovative Science and Research Technology*, 11(6), 3794-3799. https://doi.org/10.38124/ijisrt/26jun1527

I. INTRODUCTION

The Border Gateway Protocol (BGP) is the exterior gateway protocol that exchanges routing information among autonomous systems (ASes) across the global internet [1]. It was originally designed in an environment where network operators trusted each other, BGP has no built-in cryptographic mechanism to verify that a router announcing a prefix is actually authorized to do so [2]. This architectural-level gap has made BGP a persistent target for two categories of attack.

A prefix hijack occurs when a malicious or misconfigured AS announces an IP prefix it does not own, causing traffic to be misdirected to the attacker. A route leak occurs when a router propagates routes with an artificially inflated AS path, causing traffic to traverse unintended networks [12]. Both attack types have caused significant real-world disruption: the 2008 Pakistan Telecom hijack of YouTube's IP prefixes rendered the platform globally unreachable for approximately two hours [3], while the 2019 Cloudflare route leak disrupted multiple cloud providers simultaneously [4].

Existing monitoring tools – BGPStream [5], RIPE Stat [6], and BGPlay [7] – provide data collection and visualization but share a major limitation: they present raw routing data without translating findings into operator-actionable guidance. A network operator receiving a raw alert must independently determine what the anomaly means, its working impact, and the correct remediation steps – a time-consuming process during active incidents.

This paper presents a real-time BGP Route Monitoring and Visualization Tool built on the Detect-Explain-Act (DEA) framework. The system addresses the operator guidance gap by structuring the monitoring pipeline into three stages: anomaly detection using rule-based heuristics, plain-language explanation of each detected event, and prioritized actionable remediation guidance. The system is validated on a six-router BGP topology emulated in EVE-NG.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

II. RELATED WORK

A. BGP Anomaly Detection

Lad et al. [8] established the foundation for origin AS-based prefix hijack detection, demonstrating that comparing the observed origin AS against a known prefix-to-AS registry is an effective detection strategy. Shi et al. [9] extended this approach by showing that AS path length deviation, combined with origin validation, reliably identifies route leak events. The results underpin the two detection rules implemented in this work. Machine learning approaches have also been explored [10], but produce less interpretable outputs – a significant operational limitation since network operators require transparent, auditable alert explanations.

B. Present Frameworks and Gaps

BGPStream [5] provides a software framework for live and historical BGP data but offers no visual representation or operator guidance. RIPE Stat [6] aggregates routing data via a web interface suited for historical analysis, but requires considerable BGP expertise to interpret and provides no real-time alerting. BGPlay [7] provides animated visualization of historical BGP changes for post-incident forensics, but has no real-time capability and generates no alerts. None of the three frameworks provide plain-language anomaly explanations as well as operator action guidance – the gap this project addresses.

C. HCI in Network Operations

Research by Kuforiji [11] found that alerts accompanied by plain-language explanations and structured remediation steps lowered mean time to resolution in network operations centre situations compared to raw technical alerts. This finding directly motivates the design of the DEA framework's Explanation Layer.

III. SYSTEM DESIGN

A. The Detect – Explain – Act Framework

The DEA framework structures the monitoring pipeline into three sequential, independently testable stages (Fig. 1).

BGP Router	show ip bgp output
Data Collection	Telnet socket
BGP Table Parser	Route objects
Explanation Layer	Rule 1 + Rule 2
Action Engine	Remediation steps
Alert Store	Thread-safe list
Flask REST API	/api/alerts JSON
Web Dashboard	Alert cards 3s poll

Fig. 1: DEA Processing Pipeline

- *Detect:*

Two rule-based algorithms evaluate every parsed route objects against the KNOWN_OWNERS and MAX_PATH_LEN registries, raising typed alerts on any violations.

- *Explain:*

A template-based Explanation Layer enriches each alert with a plain-language translation, an operational impact assessment, and a confidence score.

- *Act:*

An Action Engine maps each explained alert to a prioritized, numbered remediation sequence drawn from standardized incident response methods.

B. Network Topology

The system is designed and validated on a six-router BGP hierarchy: one backbone router (R0/AS65000) serving as the monitoring vantage point, two ISPs (R1/AS701, R2/AS702), and four customer routers (R3/AS6501 through R6/AS6504). All sessions use eBGP-4 over point-to-point /30 links. The monitoring router R0 provides visibility into all six prefixes propagating through the network.

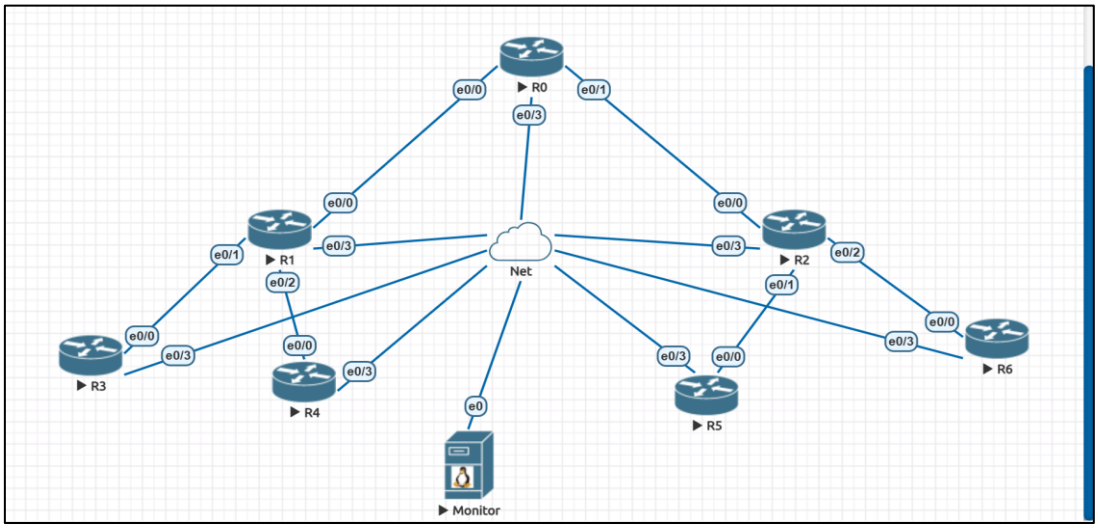

Fig. 2: Six-Router BGP Topology – EVE-NG Lab

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

C. Detection Rules

➢ *Rule 1 – PREFIX_HIJACK:*

For prefix P with detected origin AS O: if O ≠ KNOWN_OWNERS[P], raise PREFIX_HIJACK with confidence 95% and severity HIGH. This rule detects any AS announcing a prefix it does not legitimately own.

➢ *Rule 2 – ROUTE_LEAK:*

For prefix P with origin AS O and path π: if O = KNOWN_OWNERS[P] and |π| > MAX_PATH_LEN[P], raise ROUTE_LEAK with confidence 75% and severity MEDIUM. ISP prefixes are expected with path length 1; customer prefixes with length 2.

Both rules evaluate independently on every route object, enabling compound detection where a single route triggers both alert types simultaneously. A critical implementation detail is that the parser deduplicates routes by (prefix, origin_as) pair rather than prefix alone, making sure that competing legitimate and hijacked routes for the same prefix are both passed to the detection engine.

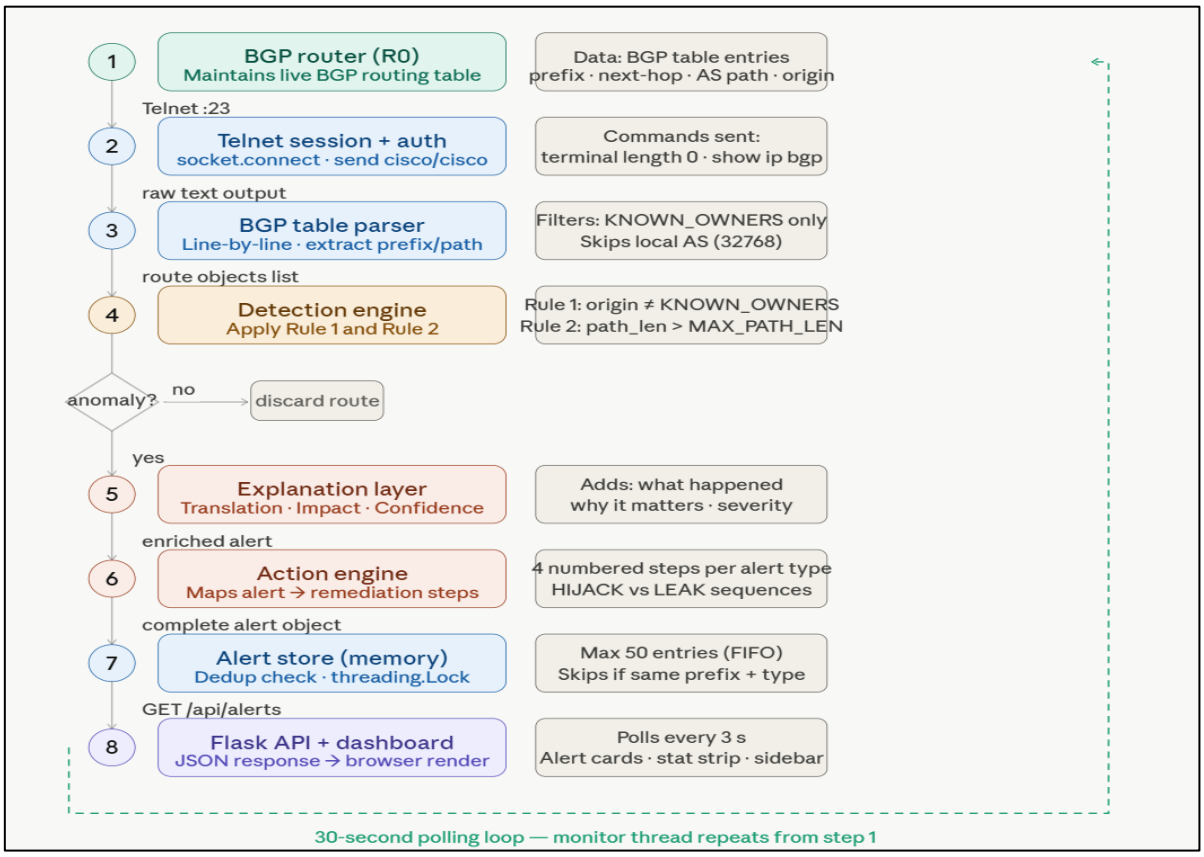

Fig. 3: Data Flow Diagram

D. Architecture Overview

The system is implemented across four layers: (1) Data Source Layer – the EVE-NG BGP topology; (2) Data Ingestion Layer – a Python Telnet client retrieving 'show ip bgp' output from R0; (3) Application Layer – the detection engine, Explanation Layer, Action Engine, and Flask REST API; and (4) Presentation Layer – a browser-based dashboard polling /api/alerts every 3 seconds.

IV. METHODOLOGY

A. Development Approach

The system was developed using a modified Agile-Scrum methodology in four phases. Phase 1 (Sprints 1–2) established the Telnet data collection pipeline and prefix-hijack detection. Phase 2 (Sprints 3 – 4) implemented the Explanation Layer and the initial Flask dashboard. Phase 3 (Sprints 5 – 6) added route leak detection, the Action Engine, and compound detection logic. Phase 4 (Sprints 7 – 8) conducted evaluation, performance measurement, and documentation.

B. BGP Table Parsing

The parser locates the 'Path' column offset from the BGP table header line, then processes each subsequent line containing a monitored prefix. Lines beginning with the standard IOS BGP status characters are processed; locally originated routes (AS 32768) are excluded. Routers are deduplicated by (prefix, origin_as) pair rather than by prefix alone – a critical design decision that makes sure both the legitimate route and any competing hijacked route for the same prefix are passed independently to the detection engine, permitting detection even when the hijacked route loses BGP best-path selection.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

C. Attack Scenario Design

Six attack scenarios were designed to cover all detectable anomaly types across both ISPs:

- S-01: R5 (AS6503) hijacks 192.168.101.0/24 owned by R3 (AS6501) – cross-ISP hijack
- S-02: R4 (AS6502) hijacks 192.168.104.0/24 owned by R6 (AS6504) – symmetric cross-ISP hijack.
- S-03: R2 (AS702) hijacks 192.168.1.0/24 owned by R1 (AS701) – ISP-level provider hijack.
- S-04: R2 (AS702) applies AS-path prepend x3 outbound – cascade leak affecting three prefixes.
- S-05: R3 (AS6501) prepends own AS twice – customer-level route leak.
- S-06: R6 (AS6504) hijacks 192.168.101.0/24 owned by R3 (AS6501) while R3 simultaneously inflates its own path via AS-path prepend, generating both PREFIX_HIJACK and ROUTE_LEAK alerts on the same prefix. To ensure R6's hijacked route won BGP best-path selection at R0, local-preference policies (localpref) were applied inbound on R2 for routes received from R6, reflecting a realistic scenario where an attacker's upstream provider grants preferential routing treatment.

D. Evaluation Metrics

Four metrics are measured: (1) detection accuracy – proportion of scenarios generating a correct alert; (2) false positive rate – alerts during verified clean baseline operation; (3) detection latency – time from attack execution to dashboard alert; (4) alert completeness – whether all three DEA fields (translation, impact, action) are present in each alert card.

V. IMPLEMENTATION

A. Technology Stack

The monitoring application is implemented in Python 3.9+ using only the standard library for core functionality (socket, threading, time). Flask 2.x serves the REST API and embeds the dashboard HTML. The frontend is implemented in vanilla JavaScript with no framework dependencies. The network emulation environment is EVE-NG Community Edition with Cisco IOL images running IOS 15.x.

B. Key Implementation Decisions

Thread safety: The alert store is protected by a threading. Lock context manager, ensuring atomic access from the concurrent BGP polling thread and Flask HTTP request-handling threads.

Duplicate suppression: Before inserting a new alert, the system checks for existing entries matching both prefix and alert type. This prevents alert flooding across polling cycles while still storing both PREFIX_HIJACK and ROUTE_LEAK alerts for the same prefix (compound detection).

Explanation templates: All three DEA text fields are generated from parameterized string templates populated with the specific AS numbers and prefix values of each incident, ensuring contextually precise rather than generic explanations.

VI. RESULTS AND DISCUSSION

A. Baseline Verification

The monitoring system was run for five consecutive polling cycles (150 seconds) against the clean baseline topology. All six expected routes were detected with correct origin ASes and path lengths on every cycle. Zero alerts were generated, confirming a 0% false-positive rate for accurately configured BGP routing.

B. Detection Results

Table 1 presents detection results across all six attack cases.

Table 1: Attack Detection Results

S#	Router	Attack Type	Prefix Attacked	Expected Alert	Result
S-01	R5 AS6503	Cross-ISP Hijack	192.168.101.0/24	PREFIX_HIJACK	PASS
S-02	R4 AS6502	Cross-ISP Hijack	192.168.104.0/24	PREFIX_HIJACK	PASS
S-03	R2 AS702	ISP Provider Hijack	192.168.1.0/24	PREFIX_HIJACK	PASS
S-04	R2 AS702	Cascade Leak x3	2.0+103.0+104.0	3x ROUTE_LEAK	PASS
S-05	R3 AS6501	Customer Leak	192.168.101.0/24	ROUTE_LEAK	PASS
S-06	R6+R3	Compound Attack	192.168.101.0/24	HIJACK+LEAK	PASS

All six scenarios were detected correctly. Detection rate = 100% (8 alerts expected, 8 generated). Scenario S-04 produced three simultaneous alerts from one misconfigured route-map, validating cascade detection. Scenario S-06 produced two simultaneous alert types on the same prefix, validating compound detection.

C. Traffic Interception Verification

A key feature of this evaluation is live traffic interception verification. All prefix hijack attacks (S-01, S-02, S-03, S-06) used loopback10 interfaces on attacker routers configured with secondary IP addresses corresponding to the hijacked prefix. Table 2 confirms that traffic was actively redirected to attack routers, not just misdirected at the routing level

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

Table 2: Traffic Interception Verification

Attack	Hijacked Prefix	CEF Next Hop	Ping Result	Interception Verified
S-01 R5	192.168.101.0/24	10.0.23.2 (ISP-B)	5/5 success	✓
S-02 R4	192.168.104.0/24	10.0.12.2 (ISP-A)	5/5 success	✓
S-03 R2	192.168.1.0/24	10.0.23.2 (ISP-B)	5/5 success	✓
S-06 R6	192.168.101.0/24	10.0.23.2 (ISP-B)	5/5 success	✓

D. Detection Latency

All attacks were detected within one polling cycle (≤30 seconds) of BGP table update propagation to R0. Parsing and anomaly-checking of six route objects completed in under 100 milliseconds, confirming that polling interval – not processing time – is the dominant latency component.

E. Comparative Study

Table 3 compares the system against three present frameworks.

Table III: Feature Comparison with Related Frameworks

Capability	This system	BGPStream	Ripe Stat	BGPlay
Real-time detection	YES	YES	Partial	NO
Prefix hijack detect	YES	YES	YES	NO
Route leak detection	YES	NO	NO	NO
Plain-language alerts	YES	NO	NO	NO
Operator action steps	YES	NO	NO	NO
Compound alert detect	YES	NO	NO	NO
No external needed	YES	NO	NO	NO
Web dashboard	YES	NO	YES	YES

The proposed system is the only framework that provides all three DEA pipeline components and detects route leaks and compound anomalies.

F. Limitations

Three limitations are acknowledged. First, the KNOWN_OWNERS and MAX_PATH_LEN registries require manual maintenance; in production, they would need to be populated automatically from RPKI ROA data. Second, single-vantage-point monitoring from R0 may miss anomalies that do not propagate to the backbone, underscoring the need for a multi-vantage-point architecture. Third, Telnet transport is appropriate only for isolated lab environments; SSH/NETCONF is required for production.

VII. FUTURE WORK

Five extensions are identified for future development: (1) RPKI integration for automated registry construction from Route Origin Authorizations; (2) multi-vantage-point monitoring aggregating BGP tables from multiple routers to eliminate blind spots; (3) SSH/NETCONF transport replacing Telnet for production-grade security; (4) persistent time-series database storage for alert history and trend analysis; and (5) IPv6 prefix support for monitoring of dual-stack networks.

VIII. CONCLUSION

This paper illustrated the design, implementation, and evaluation of a real-time BGP Route Monitoring and Visualization Tool grounded in the Detect-Explain-Act framework. The system addresses the main limitation of existing BGP monitoring tools – the absence of integrated operator guidance at the point of detection, by generating contextually precise explanations and prioritised remediation steps alongside each alert.

Initial testing on a six-router EVE-NG topology achieved 100% detection accuracy across six attack cases, spanning prefix hijacking and route leaking, cross-ISP attacks, provider-level prefix theft, cascade leaks from single misconfigurations, and compound attacks that trigger both alert types on the same prefix simultaneously. The false positive rate was 0% against a verified clean baseline, and all detections occurred within 30 seconds. All five research objectives were met. The system's lightweight, dependency-free architecture makes it immediately deployable in ISP operations and academic network security education.

REFERENCES

[1]. Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol (BGP-4)," RFC 4271, IETF, Jan. 2006.
[2]. S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway Protocol (S-BGP)," IEEE J. Sel. Areas Commun., vol. 18, no. 4, pp. 582–592, Apr. 2000.
[3]. P. Bangera, "Impact of Prefix Hijacking on Payments of Providers," M.S. thesis, Dept. Telecommunications, Aalto University, Espoo, Finland, 2010.
[4]. R. Owen, A. Bryant, L. Finch, D. Franklin, M. Abdollahi, and M. Abolhasan, "Failures and Resilience in the IP Era: Navigating the Fragility of Modern Telecommunications Networks: The

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26jun1527

Sovereign Functions," IEEE Access, vol. 13, pp. 1–20, 2025.

[5]. A. Aris, S. F. Oktug, S. Bingol, and A. Lakhina, "BGPStream: A Software Framework for Live and Historical BGP Data Analysis," in Proc. ACM Internet Measurement Conf. (IMC), Tokyo, Japan, 2016, pp. 429–444.

[6]. C. C. Gray, P. D. Ritsos, and J. C. Roberts, "Contextual Network Navigation to Provide Situational Awareness for Network Administrators," in Proc. IEEE Symp. Visualization for Cyber Security (VizSec), Chicago, IL, USA, Oct. 2015, pp. 1–8.

[7]. L. Colitti, G. Di Battista, F. Mariani, M. Patrignani, and M. Pizzonia, "Visualizing Interdomain Routing with BGPlay," J. Graph Algorithms Appl., vol. 9, no. 1, pp. 117–148, 2005.

[8]. M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang, "PHAS: A Prefix Hijack Alert System," in Proc. USENIX Security Symp., Vancouver, BC, Canada, 2006, pp. 153–166.

[9]. X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu, "Detecting Prefix Hijackings in the Internet with Argus," in Proc. ACM Internet Measurement Conf. (IMC), Berlin, Germany, 2012, pp. 15–28.

[10]. C. Shen, R. Wang, X. Li, P. Zhang, K. Liu, and L. Tan, "Border Gateway Protocol Route Leak Detection Technique Based on Graph Features and Machine Learning," Electronics, vol. 13, no. 20, p. 4072, Oct. 2024.

[11]. J. Kuforiji, "Digital Forensics and Incident Response (DFIR) Automation: Leveraging AI to Accelerate Breach Investigation, Evidence Collection, and Cyberattack Mitigation," J. Data Analysis Critical Manage., vol. 1, no. 4, pp. 1–19, 2025.

[12]. J. Mauch, J. Snijders, and G. Hankins, "Default EBGP Route Propagation Behaviour Without Policies," RFC 8212, IETF, Jul. 2017.