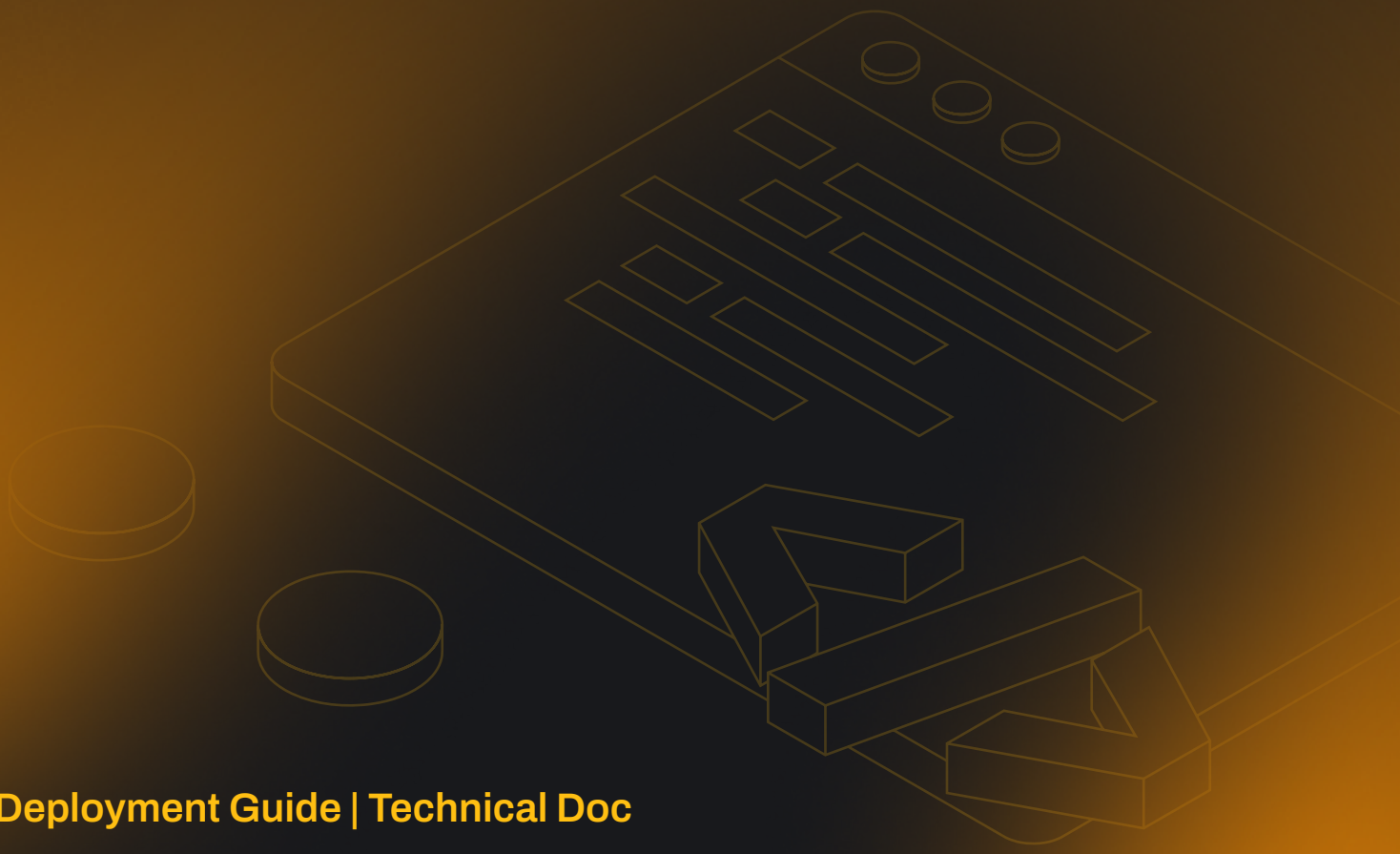
VyOS Networks



Deployment Guide | Technical Doc

BGP ROUTE ORIGIN VALIDATION VIA PUBLIC RPKI CACHE ON VYOS 1.5

2026

Index:

Introduction

This document demonstrates how to configure Border Gateway Protocol (BGP) route origin validation using Resource Public Key Infrastructure (RPKI) on a VyOS edge router. The configuration uses the lightweight RPKI-to-Router (RTR)protocol to connect to a public cache server, which dynamically downloads the global Route Origin Authorization (ROA) certificate database. This architecture effectively offloads the heavy cryptographic validation processing from the local routing engine.

BGP route origin validation highly benefits service providers and mission-critical enterprise networks. The validation process efficiently prevents prefix hijacking (BGP Hijacking), mitigates the impact of human configuration errors ("fat fingers"), and automates the penalization of illegitimate routes via route-maps.

This guide focuses on securing IPv4 routes. However, the resulting validation infrastructure also natively protects IPv6 routes. Securing both protocols aligns the network with Mutually Agreed Norms for Routing Security (MANRS).

BGP sustains global internet routing. However, the original protocol design relies on absolute trust. BGP natively assumes that if an Autonomous System (AS) announces a prefix (e.g., 192.0.2.0/24), the AS legitimately owns that prefix. The lack of structural authentication renders the global infrastructure highly vulnerable to human configuration errors, commonly known as "fat fingers," or deliberate malicious attacks. If an AS improperly announces unowned prefixes, the resulting BGP hijack or route leak can redirect, intercept, or discard global traffic, isolate services, and compromise security.

To address BGP vulnerabilities, the internet community developed RPKI, which replaces blind trust with cryptographic validation. RPKI enables legitimate owners to register their prefixes by associating them with AS numbers via digital certificates known as ROAs. Regional Internet Registries (RIRs) such as LACNIC, RIPE, and ARIN cryptographically sign and maintain these records, creating a global, decentralized database that verifies who is authorized to announce which network segment.

RPKI validation relies on a cache server or validator (Relying Party). The validator connects to global repositories, validates certificate trust chains, and compiles a verified list of prefixes and their corresponding ASs. The router connects to the cache server via the RPKI-to-Router (RTR) protocol to obtain data required for real-time routing decisions.

Lab Topology

This guide demonstrates the deployment and validation of the RPKI security architecture on VyOS 1.5. The scenario simulates a network boundary using a primary node, Edge-Router, operating in AS 65001. Edge-Router establishes eBGP peering with an external router, FRROUTING, operating in AS 65002. Edge-Router must validate routes from this peer rather than rely on blind trust.

Edge-Router establishes an RTR session with Cloudflare high-availability public validator servers. Through TCP port 8282, Edge-Router downloads the global table of secure prefixes into its internal Free Range Routing (FRR) engine. Reliable DNS resolution and internet connectivity are essential at this stage; without them, Edge-Router cannot establish the validation tunnel or access RIR databases.

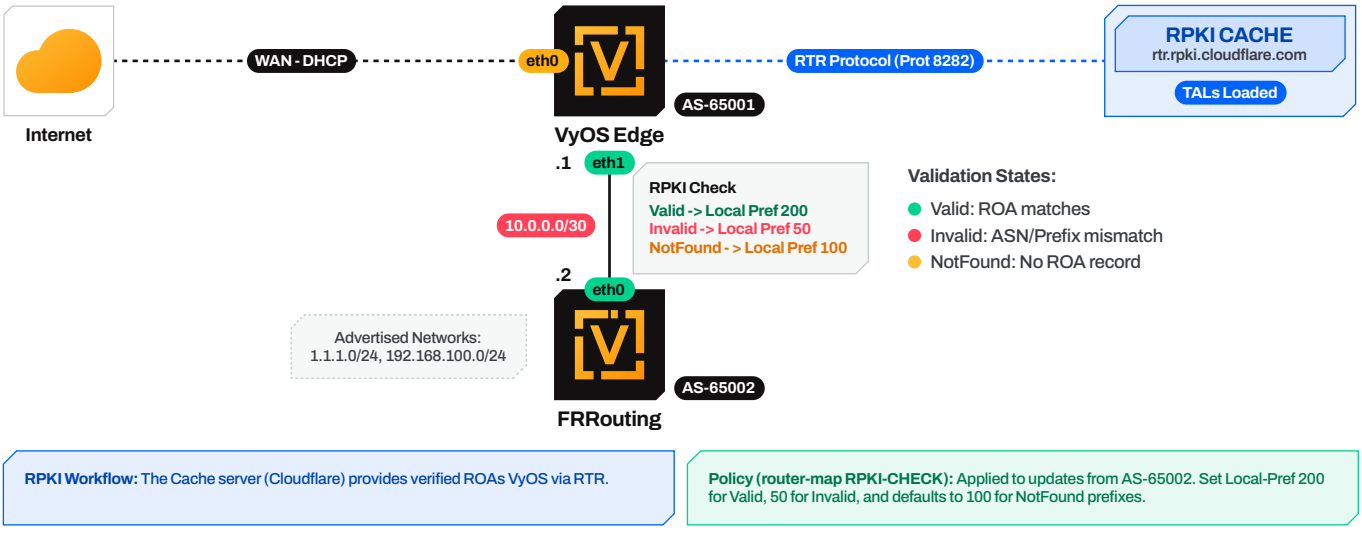
After updating the local database, Edge-Router applies traffic engineering policies. BGP does not automatically discard routes. Instead, route-maps evaluate each incoming prefix and assign one of three RPKI validation states. If a prefix matches the official record, the route-map marks the prefix as Valid, and increases the Local-Preference attribute. This adjustment makes the route the preferred path for outbound traffic.

If the peer announces an unauthorized prefix or a more specific prefix than the ROA allows, the state becomes Invalid. The route-map then decreases the Local-Preference attribute for this route. This adjustment prevents Edge-Router from sending outbound traffic to the unauthorized peer.

Finally, if a prefix lacks an RPKI record, the state becomes NotFound. The route-map applies a standard Local-Preference to these routes. This approach maintains interoperability with legacy networks that have not yet deployed RPKI.

This deployment aligns the edge infrastructure with global routing security standards (MANRS). Network administrators can verify how Edge-Router detects and mitigates unauthorized prefix announcements. Ultimately, implementing RPKI on VyOS 1.5 adds a robust defense layer for modern internet topologies.

BGP Topology with RPKI Validation - Routing security implementation (ROA Verification)



Deployment

PHASE 1: Preparation and deployment of the validator node (Edge-Router)

Edge-Router operates in private AS 65001 and serves two primary functions:

- Download the cryptographic database from RIRs via the internet.
- Enforce security policies on incoming BGP routes.

This phase consists of four sequential steps.

1.1. Base configuration, interfaces, and DNS resolution

Edge-Router requires Layer 3 connectivity and DNS resolution before establishing dynamic routing protocols. Because RPKI global cache servers use Fully Qualified Domain Names (FQDNs) for load balancing, a missing DNS configuration prevents the router from resolving the validator's IP address, causing the RTR session to fail.

This step covers the following foundational configurations:

- Sets the system hostname.
- Configures the internet-facing eth0 interface (WAN) for dynamic addressing.
- Assigns a static IP address to the eth1 interface for the eBGP peering link.

Configures the DNS server to enable RTR resolution.

Apply the following configuration to Edge-Router:

```
# Configure system hostname, WAN/LAN interfaces, and DNS for RTR resolution

configure
set system host-name Edge-Router
set interfaces ethernet eth0 address dhcp
set interfaces ethernet eth0 description 'WAN-INTERNET'
set interfaces ethernet eth1 address 10.0.0.1/30
set interfaces ethernet eth1 description 'LINK-TO-FRROUTING'
set system name-server 8.8.8.8
```

1.2. RTR tunnel establishment

After establishing internet connectivity, Edge-Router must connect to an RPKI cache server (Relying Party).

Because this deployment uses Cloudflare's public infrastructure, Edge-Router establishes the RTR session with the rtr.rpki.cloudflare.com domain on TCP port 8282 (Cloudflare's standard for unencrypted connections). Assigning the preference 1 parameter designates this cache server as the highest-priority source, allowing administrators to configure secondary cache servers in the future without service disruption.

Configure the connection to the Cloudflare public validator server:

```
set protocols rpki cache rtr.rpki.cloudflare.com port 8282
set protocols rpki cache rtr.rpki.cloudflare.com preference 1
```

1.3. Cryptographic mitigation policy design (route-maps)

The BGP process does not drop routes based on RPKI states without explicit instructions. This step establishes the RPKI-CHECK route-map to serve as the core traffic engineering policy. The route-map sequentially evaluates the RPKI state of each incoming prefix and modifies the route's Local-Preference attribute to dictate path selection.

This step configures the RPKI-CHECK route-map with the following sequence rules:

- **Rule 10 (Invalid):** Assigns a Local-Preference of 50 to prefixes with an incorrect origin AS or exceeded subnet mask length, designating them as deprioritized routes to prevent their selection.
- **Rule 20 (Valid):** Assigns a Local-Preference of 200 to prefixes that successfully pass cryptographic verification, designating them as preferred routes.
- **Rule 30 (NotFound):** If a prefix lacks an RPKI record, the route-map permits the route without altering its default priority (100). This maintains connectivity for legacy networks.

Configure the RPKI-CHECK route-map:

```
# Configure RPKI-CHECK route-map to prioritize Valid, permit NotFound, and deprioritize Invalid
prefixes

set policy route-map RPKI-CHECK rule 10 action permit
set policy route-map RPKI-CHECK rule 10 match rpki invalid
set policy route-map RPKI-CHECK rule 10 set local-preference 50
set policy route-map RPKI-CHECK rule 20 action permit
set policy route-map RPKI-CHECK rule 20 match rpki valid
set policy route-map RPKI-CHECK rule 20 set local-preference 200
set policy route-map RPKI-CHECK rule 30 action permit
```

1.4. eBGP session deployment and policy application

The final step of this phase establishes the eBGP session and enforces the RPKI policy (RPKI-CHECK route-map). This involves declaring the local AS (65001) and configuring a peer relationship with the eBGP neighbor (10.0.0.2) in AS 65002.

Because VyOS 1.5 requires explicit hierarchical configuration for route exchange, Edge-Router must activate the IPv4-unicast address family. Applying the RPKI-CHECK route-map in the inbound direction (import) ensures Edge-Router evaluates all routes from the eBGP neighbor (10.0.0.2) against cryptographic validation states before installing them into the routing table.

Configure the BGP process and apply the RPKI-CHECK route-map:

```
# Establish eBGP peering for AS 65001 to AS 65002 and apply the RPKI-CHECK route-map

set protocols bgp system-as 65001
set protocols bgp neighbor 10.0.0.2 remote-as 65002
set protocols bgp neighbor 10.0.0.2 address-family ipv4-unicast route-map import RPKI-CHECK
commit
save
exit
```

PHASE 2: BGP neighbor configuration and attack simulation (FRROUTING)

To demonstrate the architecture's effectiveness, this phase configures the FRROUTING node (AS 65002) to act as an external transit provider. To test Edge-Router's RPKI defense, this neighbor injects two specific prefixes: an unregistered private prefix (harmless) and a legitimate public prefix that does not belong to the neighboring AS 65002 (simulating a BGP hijack).

2.1. Base network configuration and interconnection

This step establishes the foundational Layer 3 connectivity required for the FRROUTING node to communicate with Edge-Router.

Apply the following configuration to the FRROUTING node:

```
# Configure FRROUTING hostname and interface for direct interconnection to Edge-Router

configure
set system host-name FRROUTING
set interfaces ethernet eth0 address 10.0.0.2/30
set interfaces ethernet eth0 description 'LINK-TO-EDGE'
```

2.2. BGP process initialization and peering relationship

This step initializes the local BGP process for AS 65002 and establishes the eBGP session with Edge-Router.

Apply the following configuration to the FRROUTING node:

```
# Initialize BGP for AS 65002, establish an eBGP session with Edge-Router, and activate the IPv4-
unicast address family

set protocols bgp system-as 65002
set protocols bgp neighbor 10.0.0.1 remote-as 65001
set protocols bgp neighbor 10.0.0.1 address-family ipv4-unicast
```

2.3. Prefix injection and attack simulation

This step injects two specific prefixes directly into the global BGP table:

- **Prefix 192.168.100.0/24:** An unregistered private IP address block. Without an associated Route Origin Authorization (ROA), this announcement returns a NotFound state on Edge-Router.

- **Prefix 1.1.1.0/24:** A legitimate public IP address block belonging to Cloudflare (AS 13335). Announcing this prefix from the FRROUTING node (AS 65002) simulates an intentional BGP hijack. Due to the unauthorized AS origin, this announcement returns an Invalid state on Edge-Router.

Apply the following configuration to the FRROUTING node:

```
# Inject a private prefix (NotFound) and hijack a Cloudflare public prefix (Invalid) for simulation

set protocols bgp address-family ipv4-unicast network 192.168.100.0/24
set protocols bgp address-family ipv4-unicast network 1.1.1.0/24
commit
save
exit
```

PHASE 3: Validation, verification, and result analysis

This final phase operates exclusively within Edge-Router's operational mode to audit the control plane and ensure that the mitigations perform exactly as designed against the simulated attack.

3.1. Cryptographic tunnel verification (RTR)

Before BGP can make any routing decisions, confirm that the certificate database has been successfully downloaded to Edge-Router.

```
vyos@Edge-Router:~$ show rpki cache-connection
Connected to group 1
rpki tcp cache rtr.rpki.cloudflare.com 8282 pref 1 (connected)
vyos@Edge-Router:~$
```

3.2. eBGP session confirmation and prefix reception

Verify the established eBGP peering relationship between Edge-Router and the FRROUTING node.

```
vyos@Edge-Router:~$ show ip bgp summary

IPv4 Unicast Summary:
BGP router identifier 10.217.32.100, local AS number 65001 VRF default vrf-id 0
BGP table version 2
RIB entries 3, using 384 bytes of memory
Peers 1, using 24 KiB of memory

Neighbor  V      AS   MsgRcvd   MsgSent   TblVer   InQ OutQ   Up/Down  State/PfxRcd   PfxSnt   Desc
10.0.0.2  4   65002       134       134        2     0    0  02:07:04            2        2   N/A

Total number of neighbors 1
```

Result analysis: In the row corresponding to neighbor 10.0.0.2 (AS 65002), the state and prefixes received column (State/PfxRcd) displays a numerical value of 2. This result confirms two key milestones: the eBGP session is fully established after completing all negotiation states, and Edge-Router has received and processed both network advertisements injected by the simulator node.

3.3. BGP table analysis

Review the Global Routing Information Base (Global RIB) to assess the effectiveness of the rules configured within the RPKI-CHECK route-map.

3.3.1. Private prefix evaluation (safe behavior):

First, audit the harmless prefix injected by the FRROUTING node.

```
vyos@Edge-Router:~$ show ip bgp ipv4 unicast 192.168.100.0/24
BGP routing table entry for 192.168.100.0/24, version 2
Paths: (1 available, best #1, table default)
 Advertised to peers:
 10.0.0.2
 65002
  10.0.0.2 from 10.0.0.2 (10.0.0.2)
   Origin IGP, metric 0, valid, external, best (First path received), rpki validation-state: not found
   Last update: Thu Apr 23 10:48:46 2026
vyos@Edge-Router:~$
```

Result analysis: Inspecting the output for the private prefix (RFC 1918) reveals the attribute rpki validation-state: not found. Since RIRs do not issue cryptographic certificates for non-publicly routable prefixes, Edge-Router correctly categorizes this announcement. As the prefix matches Rule 30 of the RPKI-CHECK route-map, Edge-Router preserves the default local preference (typically 100).

This confirms that the RPKI implementation maintains full backward compatibility and does not disrupt traffic flow for internal or legacy prefixes without an RPKI record.

3.3.2. BGP hijack evaluation (active mitigation):

Next, audit the public route owned by Cloudflare that the neighbor attempts to hijack.

```
vyos@Edge-Router:~$ show ip bgp ipv4 unicast 1.1.1.0/24
BGP routing table entry for 1.1.1.0/24, version 1
Paths: (1 available, best #1, table default)
 Advertised to peers:
 10.0.0.2
 65002
  10.0.0.2 from 10.0.0.2 (10.0.0.2)
   Origin IGP, metric 0, valid, external, best (First path received), rpki validation-state: invalid
   Last update: Thu Apr 23 10:48:46 2026
vyos@Edge-Router:~$
```

Result analysis: This output demonstrates that the RPKI security mitigation operates successfully. The rpki validation-state: invalid attribute confirms that Edge-Router detected the anomaly by cross-referencing the fraudulent origin AS (65002) against the official ROA record (AS 13335) stored in the cache.

This discrepancy triggers Rule 10 of the RPKI-CHECK route-map, which penalizes the prefix with a local preference of 50. This result confirms that the hijacked prefix is effectively suppressed within the BGP path selection algorithm, ensuring user traffic is never redirected to the attacker.

Conclusion

The successful deployment and validation of this lab mark a fundamental turning point in modern network architecture.

Conceived in the early days of the internet, BGP routing has historically relied on an implicit trust model that assumes the legitimacy of all actors. It permits human error or malicious actors to execute BGP prefix hijacks, which can intercept traffic, disrupt service availability, and compromise information integrity on a global scale.

Deploying RPKI on VyOS 1.5 resolves this foundational vulnerability at its source. Moving beyond basic static filtering, this architecture elevates the VyOS edge router into a cryptographic routing firewall.

This deployment's technical excellence is based on three empirically verified pillars:

- **Decentralized intelligence and efficiency (RTR tunnel):** The router gains cryptographic intelligence without overloading the control plane. Offloading digital signature verification to a cache server (Relying Party) and utilizing the lightweight RTR protocol ensures optimal CPU and memory performance.
- **Autonomous and deterministic mitigation (route-maps):** The configured route-map acts as a highly deterministic decision engine. As demonstrated, the router autonomously evaluates the 1.1.1.0/24 hijack attempt against the official database. By categorizing the prefix as Invalid and penalizing its local preference to 50, the system immediately suppresses the malicious announcement. This confirms the network can defend itself without human intervention and maintenance windows.
- **Guaranteed backward compatibility (NotFound state):** One of the greatest challenges in security engineering is protecting the network without breaking existing services. By allowing private and legacy prefixes to transit unaltered in the NotFound state, this lab proves that RPKI adoption is a safe transition that secures routing without penalizing unsigned ASs.

Strategic Impact

Implementing RPKI is not simply enabling an additional protocol; it is assuming the operational responsibility demanded by the modern internet. This architecture strictly aligns with the MANRS global security standards, shifting the network from a vulnerable routing participant to an actively defended infrastructure. Consequently, the organization is guaranteed that its traffic will never be diverted toward illegitimate destinations.

The edge router no longer assumes; it now verifies. This deployment guide lays the foundation for a robust network environment, preparing the infrastructure for the next evolutionary leap toward cloud-native architectures and local validation, where cryptographic control and data sovereignty will be absolute. The future of routing is secure, automated, and cryptographically validated—and with this deployment, that architecture is already a reality within the infrastructure.