lacnic 45

25 - 28 MAYO / CIUDAD DE PANAMÁ, PANAMÁ

Automatización y Telemetría aplicada a Políticas de Enrutamiento BGP

Ernesto Sánchez – Universidad Católica de Salta.
Henri Alves de Godoy – Universidade Estadual de Campinas.
William Sánchez – Celsia Internet.

Contexto

La academia y la comunidad técnica en la región:

- El presente trabajo surge de las sesiones desarrolladas en los eventos LACNIC 43 y 44, promoviendo la colaboración entre la comunidad técnica e investigadores.

Necesidad del operador:

- Celsia, operador de Internet con base en Colombia, requiere la implementación de un sistema de failover inteligente basado en métricas reales de red, reduciendo tiempos de respuesta ante fallos.

Objetivos:

- Automatizar la configuración de políticas BGP.
- Monitorear en tiempo real la calidad de enlaces con providers.
- Responder automáticamente ante degradación de servicios (latencia, jitter, pérdida de paquetes).

Arquitectura de Solución

Objetivos:

- Replicar en laboratorio Containerlab la infraestructura BGP de Celsia Internet
- Implementar un stack open source para automatización y telemetría.
- Demostrar failover automático basado en métricas de calidad de red.

Descripción de plataformas de software utilizadas:

Stack de Telemetría	Framework Automatización
• **MTR:** Diagnóstico de red hop-by-hop con análisis de latencia y pérdida • **Elasticsearch:** Almacenamiento y serialización de eventos de monitoreo • **Grafana:** Visualización en tiempo real de métricas y estado de red	• **NetBox:** Single Source of Truth para configuraciones de red • **Nornir:** Orquestación de tareas de automatización multi-vendor • **GitLab CI/CD:** Pipeline automatizado para despliegue y actualización

lacnic

Stack de Telemetría: Observabilidad de la Red

MTR (Matt's Traceroute):

- Diagnóstico avanzado de red hop-by-hop. Métricas completas: latencia, jitter, pérdida de paquetes.
- Formato de salida JSON para integración automatizada. Análisis proactivo de la calidad de enlaces.
- Fuente: https://github.com/traviscross/mtr

Elasticsearch:

- Motor de búsqueda y análisis en tiempo real. Almacenamiento escalable de eventos de telemetría.
- Indexación automática con rotación diaria. API REST para integración con sistemas externos.
- Fuente: https://www.elastic.co/es/elasticsearch

Grafana:

- Visualización dinámica de métricas de red. Dashboards personalizados para operadores.
- Alertas en tiempo real basadas en umbrales. Soporte nativo para Elasticsearch como datasource.
- Fuente: https://grafana.com/

Framework de Automatización

NetBox:

- Single Source of Truth para la infraestructura de red. Modelado de políticas BGP con custom fields.
- API RESTful para integración con herramientas externas. Webhooks para disparar eventos de automatización.
- Fuente: https://netboxlabs.com/

GitLab CI/CD:

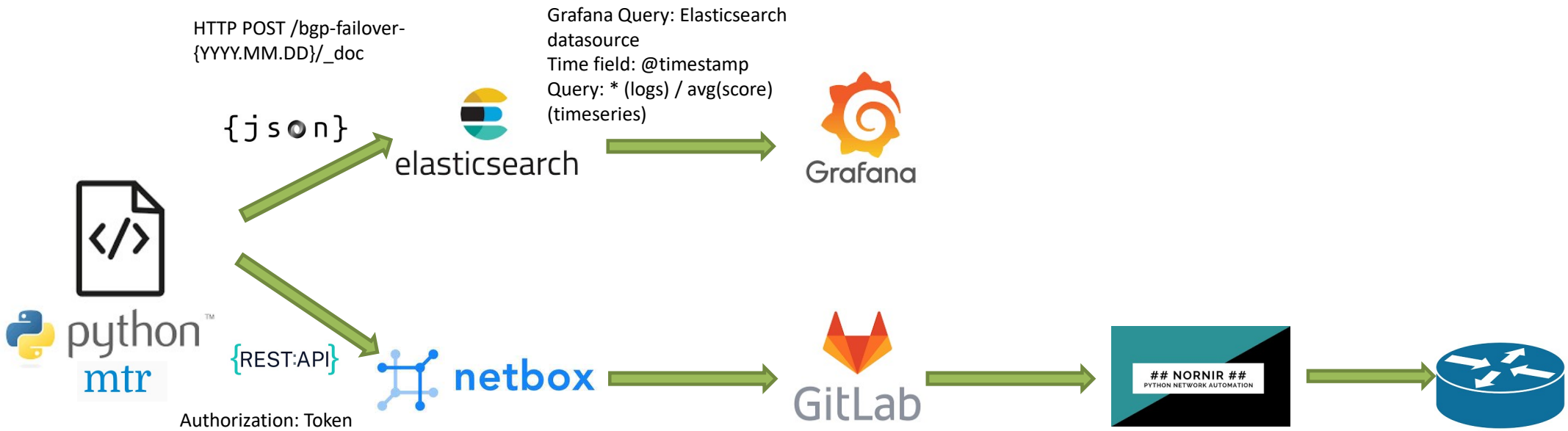
- Pipeline automatizado para cambios de configuración. GitLab Runner como ejecutor de tareas. Control de versiones integrado para configuraciones de red.
- Despliegue seguro con validación previa.
- Fuente: https://gitlab.com/

Nornir:

- Framework de automatización multi-vendor y multi-plataforma. Enfoque basado en inventario y tareas. Drivers nativos para Huawei, Juniper, Cisco, etc.
- Ejecución paralela para despliegues eficientes.
- Fuente: https://nornir.readthedocs.io/en/latest/#

lacnic

Workflow BGP Failover



lacnic

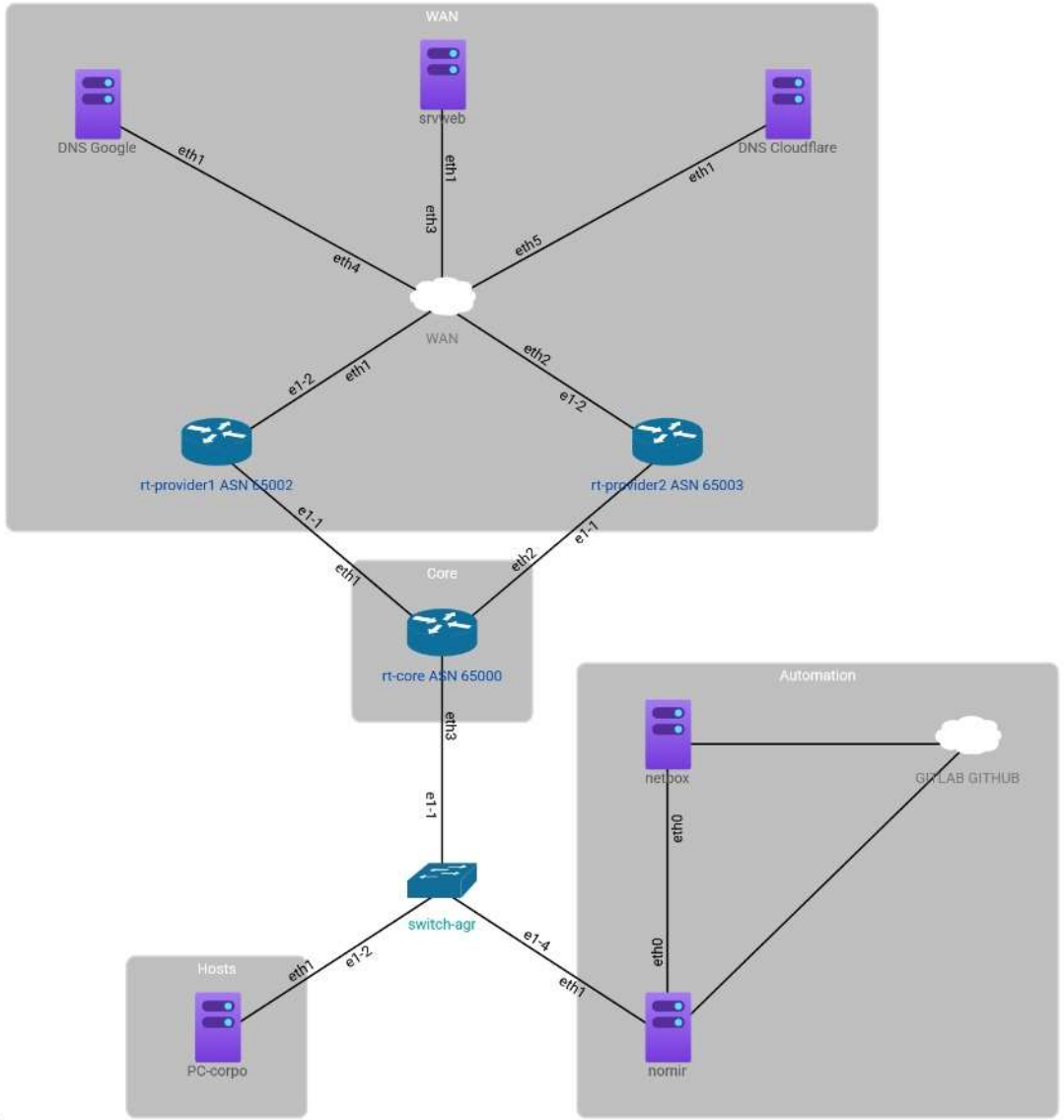
Topología de red virtualizada

Plataforma virtualización: CONTAINERlab

Nodos:

- switch: Cisco IOL L2
- rt-core: Huawei VRP
- rt-provider1: Nokia SR Linux
- rt-provider2: Nokia SR Linux
- Servers: Docker Linux Debian

Topología en repositorio Github:
https://github.com/esanchezv73/isp-automation.git

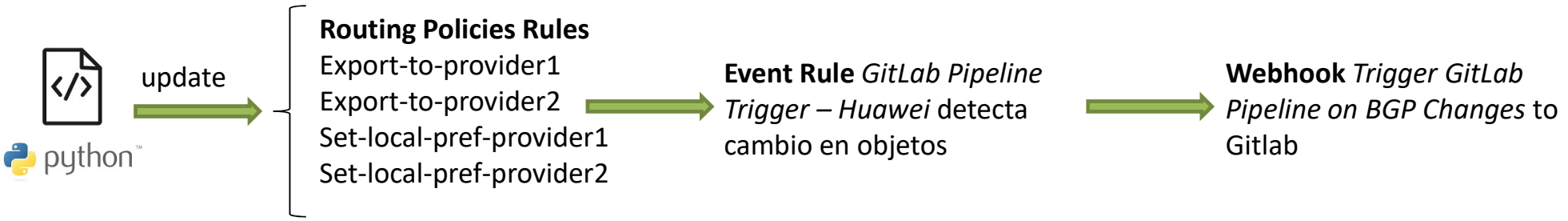


Configuración Framework Automatización

NetBox:

- Instalación Netbox-Docker.
- Fuente: https://github.com/netbox-community/netbox-docker
- Instalación de Plugin BGP.
- Fuente: https://github.com/netbox-community/netbox-bgp
- Integración con topología Containerlab.
- Generación de Token de acceso seguro.
- Configuración de objetos: Site, Plataforms, Manufactures, Devices, Interfaces, etc.
- Configuración de objetos BGP:

  Comunities, Prefix List Rules, Routing Policies Rules, Sessions.
  Custom fields: *local_asn, as_path_prepend_count, local_preference*



lacnic

Configuración Framework Automatización

GitLab CI/CD: Pipeline de Automatización Seguro:

- Creación de repositorio en GitLab.
- Configuración de archivo **.gitlab-ci.yml**, (stages: deploy, Netbox URL, Netbox Token, script: *apply_bgp_policies.py*).
- Configuración de **Inventories Nornir**: defaults.yaml, groups.yaml y hosts.yaml.
- Configuración de archivo **nornir-config.yml**: (rutas de inventario y opciones de conexión).
- Configuración de Script Python apply_bgp_policies.py
- Creación de un **Pipeline trigger token.** Habilita el lanzamiento seguro de pipelines desde NetBox mediante webhooks.
- Creación de un **Proyect Runner**, (tags: nornir, production). Requerido para ejecución de jobs de automatización

Webhook HTTPs POST

Envía la configuración del job vía API

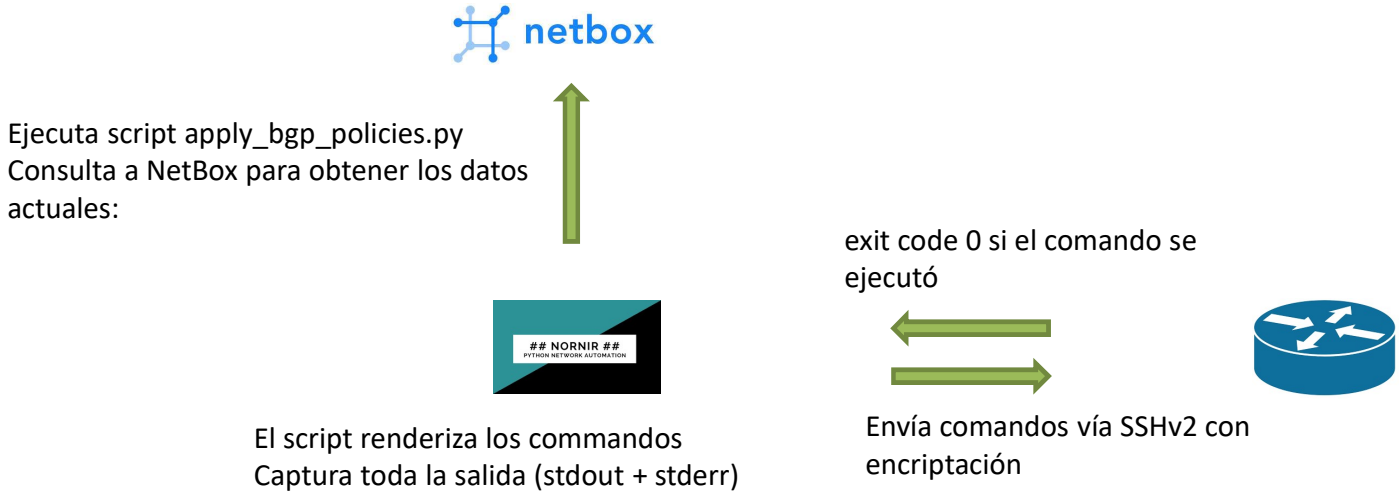
Envía logs y resultados de vuelta a GitLab

Clona el repositorio usando ci_job_token.
Ejecuta los comandos definidos en .gitlab-ci.yml

Configuración Framework Automatización

Nornir:

- Instalación de Gitlab runner.
- Registración de runner, (URL Gitlab, registration-token, tag-list y executor)
- "Automatización del plano de control mediante gestión remota, permitiendo que los cambios de enrutamiento se propaguen al plano de datos de forma automática."



Script BGP Failover Telemetry

- Script generado mediante IA Generativa. Se adopta el enfoque “IA centrada en el humano”. Se mantiene el control sobre decisiones críticas al momento del diseño y validación final de la solución propuesta.

- Diseño de prompts iterativos para cada una de las funcionalidades del scripts y refinamiento de configuraciones.

- Software de monitoreo: MTR (My Traceroute). Combinación de ping y traceroute.

- Protocolo: IPv4/IPv6 (detección automática por Provider).

- Formato: JSON para parsing automático.

lacnic

BGP Failover: Mediciones y Scoring

Puntos de Medición:	
Peer BGP	Latencia directa al router
DNS Público	Latencia hop by hop

Métricas Recolectadas:	
Latencia Promedio	(ms)
Jitter / Variabilidad	(StDev)
Pérdida de Paquetes	(%)

Parámetros de Medición:	
Intervalo de ciclo	30 segundos
Paquetes por ciclo	5 paquetes
Tamaño de paquete	64 bytes
Intervalo entre paquetes	0.5 segundos
Timeout	30 segundos

Sistema de Scoring Ponderado

Score = Latencia_Ponderada + Penalización_Pérdida + Penalización_Jitter
Donde:
- Latencia_Ponderada = (Peer × 70%) + (DNS × 30%)
- Penalización_Pérdida = (Peer_Loss% + DNS_Loss%) × 100
- Penalización_Jitter = (Peer_StDev + DNS_StDev) × 0.5

lacnic

BGP Failover: Flujo de decisión

Score menor = Mejor calidad
Thresholds Configurables:

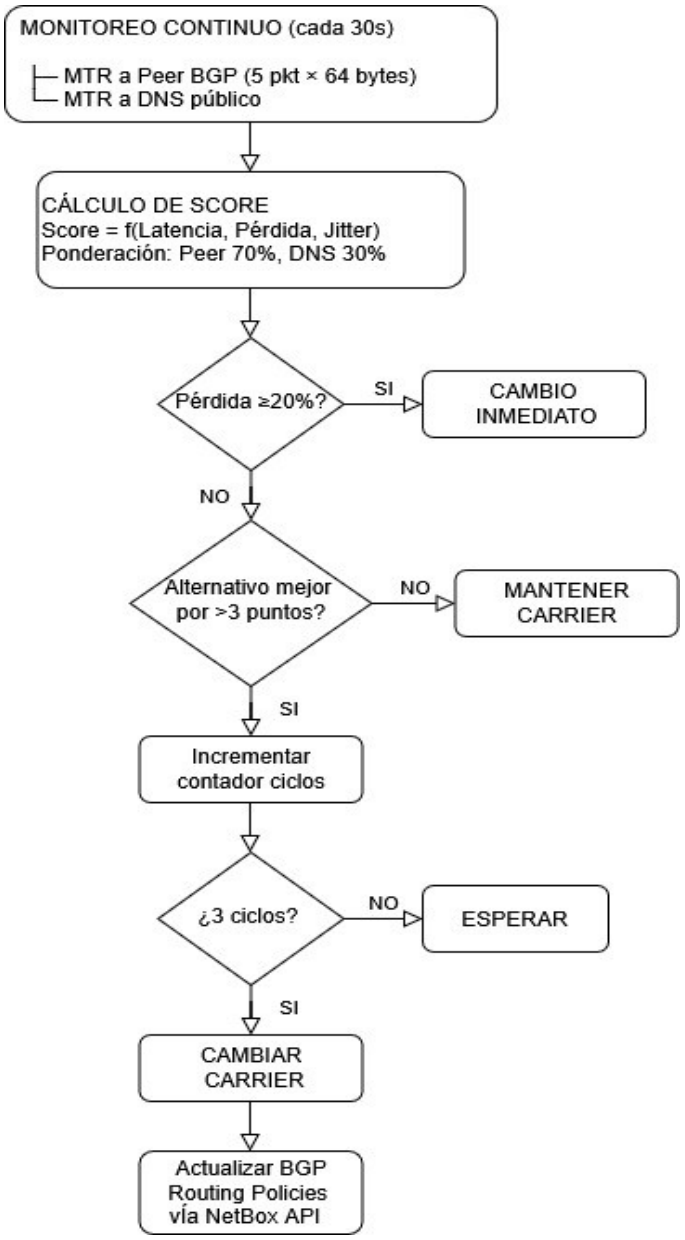
Métrica	Warning	Crítico
Latencia Peer	12 ms	25 ms
Latencia DNS	20 ms	50 ms
Pérdida Paquetes	0%	20% (cambio inmed.)
Margen de Cambio	-	3 puntos

Acción de Failover al detectar necesidad de cambio:

Provider Primario:
- AS Path Prepend: 0 (ruta preferida)
- Local Preference: 200 (prioridad alta)

Provider Backup:
- AS Path Prepend: 3 (ruta menos preferida)
- Local Preference: 100 (prioridad baja)

Actualización: Via NetBox API (routing policies)
Ejecución: Automática sin intervención manual



BGP Failover: Ejemplo mediciones

Inicio ciclo mediciones

```
# python3 bgp_failover_telemetry_u.py
⚠️Usando configuración por defecto
2026-03-31 01:51:12,787 - INFO - ✅ Conectado a Elasticsearch: http://172.90.90.9:9200
2026-03-31 01:51:12,788 - INFO -    Cluster: docker-cluster, Status: green
2026-03-31 01:51:12,788 - INFO - 🚀 BGP Failover Engine con Telemetría Unificada
2026-03-31 01:51:12,789 - INFO - 📍 Thresholds: {'peer_warning': 12, 'peer_critical': 25, 'dns_warning': 15, 'dns_critical': 30,
switch_margin': 5}
2026-03-31 01:51:12,789 - INFO - 📡 Providers: PROVIDER1, PROVIDER2
2026-03-31 01:51:12,789 - INFO - ⏱️Ciclo: 30s
2026-03-31 01:51:12,789 - INFO - 📊 Elasticsearch: http://172.90.90.9:9200/bgp-failover-YYYY.MM.DD
2026-03-31 01:51:12,790 - INFO -    Formato: UN documento por ciclo (sin anidaciones)
2026-03-31 01:51:12,790 - INFO - 🔧 Versiones IP:
2026-03-31 01:51:12,790 - INFO -    • PROVIDER1: IPv6
2026-03-31 01:51:12,790 - INFO -    • PROVIDER2: IPv6
2026-03-31 01:51:12,790 - INFO -
================================================================================
🔄 Ciclo #1
================================================================================
```

Actualización de BGP Routing Policies

```
🔄 Ciclo #5
================================================================================
2026-03-31 01:54:12,742 - INFO - ================================================================================
2026-03-31 01:54:12,742 - INFO - 🔍 Ciclo #5 - Primary: PROVIDER1
2026-03-31 01:54:20,215 - INFO - 📊 PROVIDER1 - Peer: 26.76ms (±4.45ms, loss 0.0%) | DNS: 29.31ms (±8.68ms, loss 0.0%)
2026-03-31 01:54:20,216 - INFO -    └─ Score: 34.10
2026-03-31 01:54:27,668 - INFO - 📊 PROVIDER2 - Peer: 4.42ms (±2.77ms, loss 0.0%) | DNS: 3.71ms (±1.97ms, loss 0.0%)
2026-03-31 01:54:27,669 - INFO -    └─ Score: 6.57
2026-03-31 01:54:27,669 - INFO - 📈 Scores promedio:
2026-03-31 01:54:27,670 - INFO -    ⭐ PROVIDER1: 26.65 ✅
2026-03-31 01:54:27,670 - INFO -       PROVIDER2: 9.36 ✅
2026-03-31 01:54:27,675 - INFO - ⏱️Degradación sostenida: 3/3 ciclos (PROVIDER2 mejor que PROVIDER1 por 17.29 puntos)
2026-03-31 01:54:27,941 - INFO - ✅ Métricas del ciclo #5 enviadas a bgp-failover-2026.03.31
2026-03-31 01:54:27,941 - INFO - 🔄 Cambiando de PROVIDER1 a PROVIDER2
2026-03-31 01:54:27,941 - INFO - 📝 Razón: PROVIDER2 mejor por 17.29 puntos (3 ciclos)
2026-03-31 01:54:27,941 - INFO - 🧪 DRY RUN - Actualizaría regla 1: {'as_path_prepend_count': 3}
2026-03-31 01:54:27,941 - INFO - 🧪 DRY RUN - Actualizaría regla 3: {'local_preference': '100'}
2026-03-31 01:54:27,941 - INFO - 🧪 DRY RUN - Actualizaría regla 2: {'as_path_prepend_count': 0}
2026-03-31 01:54:27,942 - INFO - 🧪 DRY RUN - Actualizaría regla 4: {'local_preference': '200'}
2026-03-31 01:54:27,942 - INFO - ✅ Cambio completado. Primary: PROVIDER2
```

BGP Failover: Dashboards Grafana

Video demo

BGP Failover: Resultados obtenidos

- Diseño, configuración y deploy completo en ambiente de pruebas. Automatización de routing policies BGP vía NetBox API. Reducción del tiempo de failover. Eliminación de intervención manual.

- Se integró al Sistema anterior un Stack de Telemetría para el monitoreo en tiempo real de calidad de enlaces con peers BGP. Métricas cada 30 segundos y visualización en Grafana con dashboards personalizados.

- Solución basada en arquitectura Open Source escalable y segura que combina control de versiones de configuraciones BGP, pipeline automatizado con validación y despliegue seguro y multivendor nativo.

Conclusiones

- A partir de una demostración práctica de observabilidad y automatización de políticas de ruteo BGP se prueba que es posible la transformación de procesos reactivos a proactivos, permitiendo a las redes adaptarse automáticamente a condiciones cambiantes.

- La automatización de procesos en la gestión de tráfico muestra una reducción significativa de errores en configuraciones manuales, permitiendo controlar y auditar los cambios con trazabilidad en plataformas como Gitlab.

- La solución propuesta, basada en herramientas Open Source, amplia el acceso a capacidades de automatización sin requerir inversiones significativas en soluciones propietarias.

- La automatización no es el futuro, es el presente necesario para operadores comprometidos con calidad de servicio y eficiencia de Internet en la región.

lacnic

Gracias!

Preguntas?