

A background network diagram consisting of various sized nodes (circles) connected by thin lines, representing a complex network topology. The nodes are distributed across the page, with a higher density in the upper right and lower right areas.

NOCTION

Intelligent Routing Platform

Lite (free version)

Installation and Configuration Guide

Version 3.11.1

1294 Lawrence Station Rd, Sunnyvale, CA 94089

Tel: 1-650-618-9823 Fax: 1-650-618-8781

Email: info@noction.com

Copyright ©2013 - 2019 Noction Inc., All Rights Reserved.

Noction logos, and trademarks or registered trademarks of Noction Holdings Ltd or its subsidiaries in the United States and other countries.

Other names and brands may be claimed as the property of others. Information regarding third party products is provided solely for educational purposes. Noction Holdings Ltd is not responsible for the performance or support of third party products and does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices or products.

Copyright ©2013 - 2019 Noction Inc., All Rights Reserved.

Contents

1	Introduction	6
1.1	How to contact support	6
1.2	What is IRP	6
1.2.1	IRP Features	6
1.2.2	IRP Components	7
1.2.3	IRP Technical Requirements	8
1.2.4	IRP Operating modes	12
1.2.5	BGP Monitoring	15
1.2.6	Outage detection	18
1.2.7	VIP Improvements	18
1.2.8	Retry Probing	18
1.2.9	Routing Policies (Limited to 3 policies in IRP Lite)	20
1.2.10	Support for Centralized Route Reflectors	23
1.2.11	Support for Internet Exchanges (Not supported in IRP Lite)	23
1.2.12	Optimization for multiple Routing Domains	24
1.2.13	Improvements weight	30
1.2.14	Notifications and events	30
1.2.15	IRP API	33
1.2.16	IRP Failover (Not supported in IRP Lite)	34
1.2.17	Inbound optimization (Not supported in IRP Lite)	38
1.2.18	Flowspec policies	40
1.2.19	Throttling excessive bandwidth use with Flowspec	41
1.2.20	Maintenance windows	41
1.2.21	Optimization of transiting traffic (Not supported in IRP Lite)	42
1.2.22	Circuit issues detection	43
1.2.23	DDOS detection and Blackholing	44
1.3	IRP Optimization modes	44
1.3.1	Performance optimization	44
1.3.2	Cost optimization	45
1.3.3	Commit Control (Not supported in IRP Lite)	45
2	Configuration	50
2.1	Configuration files	50
2.2	Global and Core Configuration	50
2.3	Collector Configuration	52
2.3.1	Irpflowd Configuration	52
2.3.2	Irpspand Configuration	56
2.4	Explorer Configuration	58
2.4.1	Specific PBR configuration scenarios	58
2.4.2	Flowspec PBR	71
2.5	Bgpd Configuration	72
2.5.1	AS-Path behavior in IRP Bgpd	76
2.5.2	Bgpd online reconfiguration	76
2.5.3	BGP Additional paths	77
2.6	Failover configuration (Not supported in IRP Lite)	77
2.6.1	Initial failover configuration	77
2.6.2	Re-purpose operational IRP node into an IRP failover slave	86
2.6.3	Re-purpose operational IRP failover slave into a new master	86

2.6.4	Recover prolonged node downtime or corrupted replication	87
2.7	Frontend Configuration	88
2.8	Administrative Components	88
2.9	IRP Initial Configuration	88
2.10	IRP software management	88
2.11	BMP configuration	90
2.12	Starting, stopping and status of IRP components	90
3	Using IRP	92
3.1	Getting started	92
3.1.1	Accessing the system	92
3.1.2	System dashboard	92
3.2	Frontend structure	94
3.2.1	Sidebar	94
3.2.2	Failover status (Not supported in IRP Lite)	97
3.2.3	Global search	98
3.2.4	Warning bars	99
3.3	Customizing the dashboard	99
3.3.1	Custom dashboard gadgets	100
3.4	Reports	102
3.4.1	Platform overview	102
3.4.2	Platform performance	103
3.4.3	Loss Improvements	103
3.4.4	Latency Improvements	105
3.4.5	Probe conversion rates	107
3.4.6	Current improvements	107
3.4.7	Improvements to Exchanges (Not supported in IRP Lite)	109
3.4.8	Historical records	109
3.4.9	Providers overall	110
3.4.10	Provider performance	110
3.4.11	Cost overview	112
3.4.12	Congestion and outages	112
3.4.13	Top volume prefixes	113
3.4.14	Top volume AS	113
3.4.15	Top problem AS	114
3.4.16	Prefix statistics	114
3.4.17	AS statistics	116
3.4.18	Country statistics	116
3.4.19	Exchanges statistics	117
3.4.20	Current inbound improvements (Not supported in IRP Lite)	118
3.4.21	Inbound Improvements history (Not supported in IRP Lite)	118
3.4.22	Probes today	119
3.4.23	Monitors status	119
3.5	Graphs	121
3.5.1	Improvements by type	121
3.5.2	Performance improvements	121
3.5.3	Probed prefixes and Improvements	122
3.5.4	New and retry improvements	122
3.5.5	Improvements by probing source	123
3.5.6	Prefixes rerouted from provider	123
3.5.7	Prefixes rerouted to provider	124
3.5.8	Improvements by provider	124
3.5.9	Performance improvements by provider	125
3.5.10	Commit improvements by provider(Not supported in IRP Lite)	126
3.5.11	Total and Improved traffic	126
3.5.12	Bandwidth usage by provider	127
3.5.13	Providers bandwidth usage	127
3.5.14	Bandwidth usage and improvements	128
3.5.15	Total bandwidth usage	129

3.5.16	Inbound traffic distribution	129
3.5.17	Provider performance history	129
3.6	Routing Policies	130
3.7	Flowspec policies	133
3.8	Throttling excessive bandwidth use with Flowspec	135
3.9	Maintenance windows	135
3.10	Events	137
3.11	Troubleshooting	139
3.11.1	Traceroute	139
3.11.2	Looking glass	140
3.11.3	Whois	141
3.11.4	Prefix probing	141
3.12	Configuration editor	143
3.12.1	Global configuration	144
3.12.2	BGP and Routers configuration	147
3.12.3	Collector configuration	150
3.12.4	Core configuration	152
3.12.5	Commit Control configuration (Not supported in IRP Lite)	156
3.12.6	Explorer configuration	158
3.12.7	Providers and Peers configuration	159
3.12.8	SNMP hosts configuration	167
3.12.9	Notifications and Events	168
3.12.10	Security Configuration	173
3.12.11	Frontend configuration	179
3.12.12	Inbound configuration (Not supported in IRP Lite)	180
3.12.13	Wizards	187
4	Configuration parameters reference	201
4.1	Complete parameters list	201
4.1.1	Database credentials	201
4.1.2	Global parameters	203
4.1.3	API daemon settings	212
4.1.4	Bgpd settings	214
4.1.5	BGP sessions settings	221
4.1.6	BMP monitoring station settings	230
4.1.7	Collector settings	231
4.1.8	Core settings	238
4.1.9	Explorer settings	253
4.1.10	Notification and events settings	259
4.1.11	Administrative settings	266
4.1.12	Providers settings	268
4.1.13	Inbound settings	280
4.1.14	Routing Policies settings	281
4.1.15	Exchanges settings	283
4.1.16	Global Group settings	285
4.1.17	User Directory settings	286
4.1.18	SNMP Hosts settings	288
4.1.19	Routing domains settings	290
5	Appendixes	291
5.1	Frontend labels for configuration parameters	291
5.2	Configuration parameter index	299

Chapter 1

Introduction

1.1 How to contact support

If you encounter any problems while using or setting up the Intelligent Routing Platform, please contact us.

You may contact us in the following ways:

- Email us at support@noction.com
- Open a support request at <https://helpdesk.noction.com>

1.2 What is IRP

BGP is a fundamental technology for the fault-tolerance of the Internet, which chooses network paths based on the number of hops traffic must traverse before it reaches its destination. However, BGP does not take into account the important factors of network performance. Even if multi-homing does provide some redundancy, multiple network outages have shown that multi-homing alone is not the solution for risk diversity and business continuity. When major blackouts or even congestions happen, multi-homing gives a fallback link for the “first-mile” connection, rather than providing a way to route around the Internet “middle-mile” issues.

Noction Intelligent Routing Platform (IRP) is a product developed by Noction to help businesses to optimize their multi-homed network infrastructure. The platform sits in the network and gets a copy of the traffic from the edge routers. The system passively analyzes it for specific TCP anomalies and actively probes remote destination networks for such metrics as latency, packet loss, throughput, and historical reliability. It computes a performance- or a cost-improvement network traffic engineering policy and applies the new improved route by announcing it to the network’s edge routers via traditional BGP session.

Noction IRP is a complete network monitoring and troubleshooting solution, which facilitates the detection, diagnosis, and automatic resolution performance issues. It delivers real-time views and dashboards that allow to visually track network performance and generate triggers, alerts and notifications when specific problems occur.

1.2.1 IRP Features

The Intelligent Routing Platform is designed to help Service Providers to improve the performance and to reduce the costs of running a multi-homed BGP network. The system makes intelligent routing decisions by analyzing various network performance metrics and selecting the best performing route for the traffic to pass through. As a result, Noction IRP allows you to:

- Improve overall network performance
- Reroute congestion and outages

- Decrease network downtime
- Reduce latency and packet loss
- Get comprehensive network performance analytics
- Facilitate network troubleshooting
- Decrease network operational costs
- Monitor platform performance
- Reduce the risk of human errors during BGP configuration

1.2.1.1 IRP Lite limitations

IRP Lite is designed as a promotional and educational tool that might also be useful to optimize some smaller networks that are not encumbered by IRP Lite limitations. IRP Lite use is governed by Terms of Service that constrain further how IRP Lite can be used.

The high level list of technical limitations in IRP Lite include:

- Support for Internet Exchanges (Not supported in IRP Lite) is missing
- Commit Control (Not supported in IRP Lite) features are missing
- At most 750 improvements, 3 transit providers and 3 routing policies are supported.
- Flowspec policies are limited to maximum of 3 IPv4 and 3 IPv6 policies.

IRP Lite periodically sends event notifications to a designated @noction.com email address. The data is collected for statistics purposes and is used to enhance this and other Noction services like outage detection and confirmation and Internet health reports.

1.2.2 IRP Components

The IRP platform has a few interconnected components (see figure 1.2.1 on the following page) , which are performing together to improve the routing best path selection and to eliminate various common network issues.

A short description of each of the components is given below. The detailed information is available in the next chapters.

To inject the Improvements into the network, the platform needs to ‘tell’ the routers what exactly needs to be changed in the routing table. IRP **BGP daemon** announces the improved prefix with an updated next-hop for the traffic to start flowing through the new path.

Bmpserver receives BMP monitoring sessions to provide route data to IRP components.

Core is the most important part of the system. It runs all the logical operations and interconnects all the components. It handles the performance and cost improvements. It processes, stores and exports data.

Collector **Irpflowd** & **Irpspan** are receiving, analyzing and processing all the traffic passing the network. They have two ways to gather data about the network: by mirroring network traffic or by using NetFlow/sFlow.

Explorer runs all the probes and checks the metrics specified by the platform policies, such as packet loss and latency. This information is sent back to the Core.

Frontend represents a web interface with a comprehensive set of reports, graphs and diagnostic information which can reflect the current and historical network state, as well as the benefits of IRP network optimization.

Globalcc performs communication between different IRP instances to maintain commit levels globally.

Irapid is used to serve requests from Frontend and GMI user interfaces and provides API to external tools.

Irpdetected collects various statistics for DDOS detection and mitigation feature.

Irppushd is responsible for forwarding events to configured notification channels.

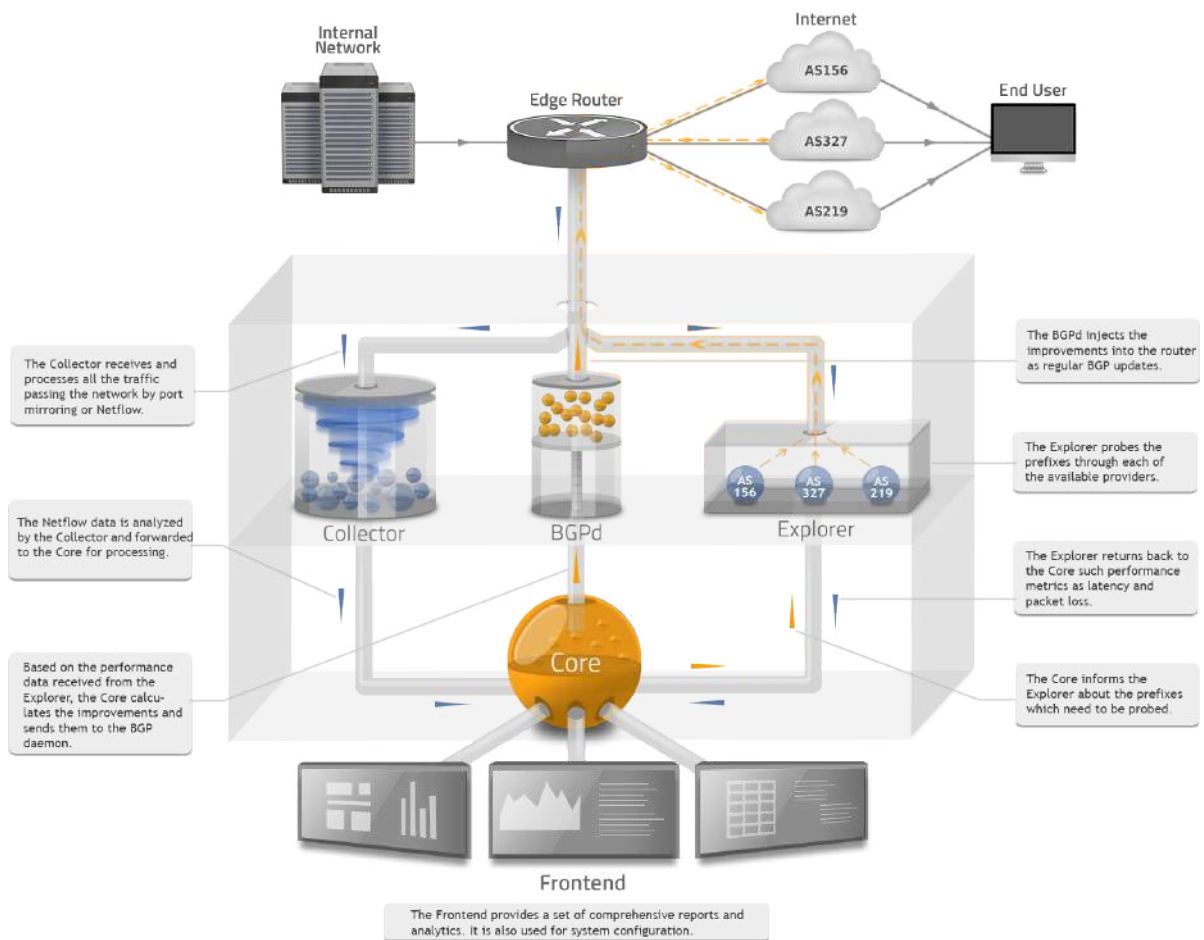


Figure 1.2.1: IRP Components

1.2.3 IRP Technical Requirements

In order to plan the IRP deployment in your network, a series of requirements need to be met and specific information has to be determined to configure IRP.

1.2.3.1 Hardware requirements

⚠ In production, a dedicated server for each IRP instance is strongly recommended. The system can also be deployed on a Virtual Machine with matching specifications, provided that this is hardware- or paravirtualization (Xen, KVM, VMware). Os-level virtualization (OpenVZ/Virtuozzo or similar) is not supported.

1. CPU

- Recommended Intel® Xeon® Processor E3/E5 family, for example:
 - 1x Intel® Xeon® Processor E3 family for up to 20 Gbps traffic;
 - 1x Intel® Xeon® Processor E5 family for 40 Gbps or more traffic.

2. RAM

- if providing sFlow/NetFlow data at least 16 GB, recommended - 32 GB;
- if providing raw traffic data by port mirroring:


- minimum 16 GB for up to 10 Gbps traffic;
- minimum 32 GB for 40 Gbps traffic
- Additional RAM would be required to maintain large amount of BGP & BMP sessions (for example, Bgpd occupies about 10G of RAM for 16 full view BGP sessions; the estimation may change due to growth of world's BGP table and new IRP features).

3. HDD


- At least 160GB of storage;
- SAS disks are recommended (SSDs are required only for 40Gbps+ networks);
- HDD partitioning:
 - LVM is recommended;
 - At least 100GB disk space usable for /var or separate partition;
 - At least 10GB disk space usable for /tmp or separate partition. This is required for big mysql tables manipulation. More disk space might be required under heavy workload.

4. NIC

- if providing sFlow/NetFlow data - at least 1 x 1000Mbps NIC while two NICs are recommended (one will be dedicated to management purposes).
- if providing raw traffic data by port mirroring - additional 10G interfaces are required for each of the configured SPAN ports (Myricom 10G network cards with Sniffer10G license are recommended to be used for high pps networks). When configuring multiple SPAN ports the same number of additional CPU cores are needed to analyze traffic.


 In very large networks carrying hundreds or more of Gbps of traffic and in IRP configurations with very aggressive optimization settings configurations with 2x or 4x CPUs are recommended. The IRP servers in these cases should also allocate double the recommended RAM and use SSD storage.

Noction can size, setup and mail to you an appliance conforming to your needs. The appliance is delivered with OS installed (latest CentOS 7) and IRP software deployed.

 A different supported OS can be installed on customer request.

1.2.3.2 Software requirements

Clean Linux system, with the latest CentOS 7 or Ubuntu Server LTS for x86_64 architecture installed on the server.

 IRP has a dependency on MySQL/MariaDB server and it expects the latest version from official OS repositories. In case the DBMS has been installed from a different repository it is strongly advised that the database instance and its configuration is purged before proceeding with IRP installation.

IRP requires root access to local database instance during first installation. In case the root access can't be given, use the statements below to grant all necessary privileges to the 'irp' user and database.:

```
GRANT ALL ON $dbdbname.* TO '$dbusername'@'$dbourhost' IDENTIFIED BY '$dbpassword' WITH GRANT OPTION
```

```
GRANT SELECT ON $dbdbname_fe.* TO '$dbusername_fe'@'$dbourhost' IDENTIFIED BY '$dbpassword_fe'
```

where \$dbdbname (4.1.1.6), \$dbusername (4.1.1.11), \$dbpassword (4.1.1.9), \$dbourhost (4.1.1.8) are the corresponding parameters from `/etc/noction/db.global.conf` \$dbdbname_fe and \$dbusername_fe from `/etc/noction/db.frontend.conf`

i IRP's Frontend is built on PHP 7.x and Symfony framework. When they are not part of main-stream OS they are retrieved from Noction's repositories.

1.2.3.3 Network-related information and configuration

IRP is designed to help Service Providers (AS) optimize a multi-homed BGP network. This implies the basic prerequisites for using IRP:

- Ownership of the AS for the network where IRP is deployed,
- BGP protocol is used for routing and,
- Network is multi-homed.

Eventually the following needs to be performed in order to deploy and configure IRP:

1. Prepare a network diagram with all the horizontal (own) as well as upstream (providers) and downstream (customers) routers included. Compare if your network topology is logically similar to one or more of the samples listed in section [Collector Configuration](#) for example [Flow export configuration](#).
2. Identify the list of prefixes announced by your AS that must be analyzed and optimized by IRP.
3. Review the output of commands below (or similar) from all Edge Routers:
 - `'sh ip bgp summary'`
 - `'sh ip bgp neighbor [neighbor-address] received-routes'`
 - `'sh ru'` (or similar)

The settings relating to BGP configuration, prefixes announced by your ASN, the route maps, routing policies, access control list, sFlow/NetFlow and related interfaces configurations are used to setup similar IRP settings or to determine what settings do not conflict with existing network policies.

4. Provide traffic data by:
 - (a) **sFlow**, **NetFlow** (v1, 5, 9) or **jFlow** and send it to the main server IP. Make sure the IRP server gets both inbound and outbound traffic info.
Egress flow accounting should be enabled on the provider links, or, if this is not technically possible, ingress flow accounting should be enabled on all the interfaces facing the internal network.
NetFlow is most suitable for high traffic volumes, or in the case of a sophisticated network infrastructure, where port mirroring is not technically possible.
Recommended sampling rates:
 - i. For traffic up to 1Gbps: 1024
 - ii. For traffic up to 10Gbps: 2048
 - (b) Or: configure port mirroring (a partial traffic copy will suffice). In this case, additional network interfaces on the server will be required - one for each mirrored port.

See also: [Collector Configuration](#)

5. Setup Policy Based Routing (**PBR**) for IRP active probing.
 - Apart from the main server IP, please add an additional alias IP for each provider and configure PBR for traffic originating from each of these IPs to be routed over different providers.
 - No route maps should be enforced for the main server IP, traffic originating from it should pass the routers using the default routes.

- Define Provider↔PBR IP routing map

In specific complex scenarios, traffic from the IRP server should pass multiple routers before getting to the provider. If a separate probing Vlan cannot be configured across all routers, GRE tunnels from IRP to the Edge routers should be configured. The tunnels are mainly used to prevent additional overhead from route maps configured on the whole IRP↔Edge routers path.

i If network has Flowspec capabilities then alternatively Flowspec policies can be used instead of PBR. Refer for example [Flowspec policies](#), [global.flowspec.pbr](#).

1. Configure and provide **SNMP** for each provider link, and provide the following information:

- SNMP interface name (or ifIndex)
- SNMP IP (usually the router IP)
- SNMP community

This information is required for the report generation, Commit Control decision-making and prevention of overloading a specific provider with an excessive number of improvements.

i The above is applicable in case of SNMP v2c. If SNMP v3 is used further details will be required depending on security services used.

1. To setup cost related settings as well as the Commit Control mechanism, provide the maximum allowed interface throughput for each provider link as well as the cost per Mbps for each provider.

1.2.4 IRP Operating modes

The IRP platform can operate in two modes, which can be used at different stages of the deployment process. During the initial installation and configuration, it is recommended for the system not to inject any improvements into the network until the configuration is completed.

After running several route propagation tests, the system can be switched to the full Intrusive mode.

1.2.4.1 Non-intrusive mode

While running in this mode, the system will not actually advertise any improvement to the network, and will only reflect the network improvements and events in the platform reports and graphs.

1.2.4.2 Intrusive mode

After the system configuration is completed, and manual route propagation tests were performed in order to ensure that the edge routers behavior is correct, the system can be switched to Intrusive mode. While running in this mode, the system injects all the computed improvements into the edge router(s) routing tables, allowing the traffic to flow through the best performing route.

1.2.4.3 Going Intrusive

While IRP operates in non-intrusive mode it highlights the potential improvements within client's environment. Going Intrusive will realize IRP potential.

The difference between Intrusive and Non-Intrusive operating modes is that IRP advertises improvements to edge routers. In order to switch to Intrusive we follow a controlled process. The highlights of the process are as follows:

1. The optimizing component of IRP (Core) is taken offline and existing improvements are purged. The Core being offline guarantees IRP will not automatically insert new improvements into its Current Improvement table and hinder the Go Intrusive process.

Listing 1.1: Stop IRP Core and purge existing improvements

```
root@server ~ $ service core stop
root@server ~ $ mysql irp -e 'delete_from_improvements;'
```

2. Enable Intrusive Mode and adjust IRP Core and Bgpd parameters as follows:

Listing 1.2: Switch to Intrusive Mode and adjust IRP Core and Bgpd parameters

```
root@server ~ $ nano /etc/noction/irp.conf
global.nonintrusive_bgp = 0
core.improvements.max = 100
bgpd.improvements.remove.next_hop_eq = 0
bgpd.improvements.remove.withdraw = 0
```

3. Improvements towards test networks are introduced manually so that client's traffic is not affected. The improvements are chosen so that they cover all client's providers. Any public networks could be used for test purposes. Just keep in mind that preferably your network shouldn't have traffic with chosen test network in order to do not re-route the real traffic. Use the template below in order to insert the test improvements:

Listing 1.3: Inserting test improvements

```
mysql> insert into improvements
(ni_bgp, prefix, peer_new, ipv6, asn)
```

```

values
(0, '10.10.10.0/24', 1, 0, 48232),
(0, '10.10.11.0/24', 2, 0, 48232),
(0, '10.10.12.0/24', 3, 0, 48232);

```

4. Make sure that 'route-reflector-client' is set for IRP BGP session.
5. Make sure that 'next-hop-self' is not configured for IRP BGP session.
6. On iBGP sessions (between edge routers, route-reflectors; except session with IRP) where 'next-hop-self' is configured, the following route-map should be applied:


Listing 1.4: Remove next-hop-self route-map (RM-NHS) example

```

route-map RM-NHS
set ip next-hop peer-address
neighbor X.X.X.X route-map out RM-NHS

```

where X.X.X.X is the iBGP neighbor

 route-map contents should be integrated into existing route-map in case other route-map already configured on the iBGP session.

7. Use the commands below to restart IRP Bgpd to use actual IRP configuration and establish BGP session(s) and verify if BGP updates are being announced:

Listing 1.5: Restart IRP Bgpd

```

root@server ~ $ service bgpd restart
root@server ~ $ tail -f /var/log/irp/bgpd.conf

```

Wait **for** the following lines **for** each BGP session:

```

NOTICE: Adding peer X
NOTICE: BGP session established X
INFO: N update(s) were sent to peer X

```

where X is the router name and N is the number of the updates sent towards the X router.

8. Verify if IRP BGP announcements are properly propagated across all the network. Run the following commands on each router (the commands vary depends on the router brand):

Listing 1.6: Show BGP information for specified IP address or prefix

```

show ip bgp 10.10.10.1
show ip bgp 10.10.11.1
show ip bgp 10.10.12.1

```

Analyze the output from all the routers. If the IRP BGP announcements are properly propagated, you should see /25 (refer to 4.1.4.33) announcements and the next-hop for each announcement should be the improved provider's next-hop:

```

10.10.10.1 - provider 1 next-hop
10.10.11.1 - provider 2 next-hop
10.10.12.1 - provider 3 next-hop
(refer to 4.1.12.30, 4.1.12.38, 4.1.15.3).

```

Run the following commands in order to check if IRP improvements are announced and applied:

Listing 1.7: Traceroute destination networks

```
root@server ~ $ traceroute -nn 10.10.10.1
root@server ~ $ traceroute -nn 10.10.11.1
root@server ~ $ traceroute -nn 10.10.12.1
```

Again, you should see corresponding providers' next-hops in the traces.

9. If the tests are successful perform the steps below:

(a) Delete test improvements

Listing 1.8: Delete test improvements

```
root@server ~ $ mysql -e "delete_from_improvements_where_prefix_
like_'10.10.1%';"
```

(b) Configure at most 100 improvements and revert Bgpd configuration

Listing 1.9: Configure the maximum improvements limit and revert Bgpd configuration

```
root@server ~ $ nano /etc/noction/irp.conf
core.improvements.max = 100
bgpd.improvements.remove.next_hop_eq = 1
bgpd.improvements.remove.withdraw = 1
```

(c) Restart IRP Core and Bgpd

Listing 1.10: Restart IRP Core and Bgpd

```
root@server ~ $ service core restart
root@server ~ $ service bgpd restart
```

10. If everything goes well, after 1-2 hours the maximum number of improvements announced is increased to 1000 and after 24 hour to 10000.

As a rollback plan we have to revert the changes and switch the system to non-intrusive mode:

1. Delete test improvements

Listing 1.11: Delete test improvements

```
root@server ~ $ mysql -e "delete_from_improvements_where_prefix_like_
'10.10.1%';"
```

2. Switch the system to non-intrusive mode

Listing 1.12: Switch the system to non-intrusive mode

```
root@server ~ $ nano /etc/noction/irp.conf
global.nonintrusive_bgp = 1
```

3. Restart IRP Core and Bgpd

Listing 1.13: Restart IRP Core and Bgpd

```
root@server ~ $ service core restart
root@server ~ $ service bgpd restart
```

1.2.5 BGP Monitoring

IRP uses two types of BGP monitors and a BMP monitoring station to collect data, diagnose and report mainly the state of the BGP session between the edge routers and the providers, as well as the network reachability through a specific provider. The information provided by monitors enables IRP to avoid announcing routing updates that would result in traffic misrouting for example by sending improvements to a failed provider but also to better inform IRP probing and improvement decisions.

1.2.5.1 Internal monitor

Internal BGP Monitor is checking the state of the Edge Router→Provider BGP session by regularly polling the router via SNMP. When queried, the SNMP protocol returns variables describing the session status to be used by the IRP's Internal BGP Monitor. If the session between the edge router and the provider is down, SNMP will return a value, representing session failure and IRP will react as follows:

- the provider will be marked as FAILED,
- all the improvements towards this provider will be withdrawn from the routing tables to avoid creating black holes,
- new improvements towards this providers will not be made.

In some cases (e.g. DDoS attack or various factors causing router CPU over-usage) there may be no response to the SNMP queries at all. In this case a timeout status will be reported to the Internal Monitor and a 30 minutes timer (called longhold timer) (`bgpd.mon.longholdtime`) will be started. During this time the monitor will be sending ICMP/UDP ping requests toward the configured provider's next-hop IP address (`peer.X.ipv4.next_hop` or `peer.X.ipv6.next_hop`). The requests will be sent once in keepalive period (a parameter adjustable in the BGP daemon configuration interface) (`bgpd.mon.keepalive`). If the next-hop stops responding to these requests, another 30 seconds timer (called hold timer) (`bgpd.mon.holdtime`) will be started. If according to the ping response the session is reestablished during this time, the hold timer will be discarded while the longhold timer continues. In case one of the timers expires, the provider is switched to a FAIL state and all the improvements towards this provider will be withdrawn from the routing table. However, if the BGP session with the provider is re-established, the system will start rerouting traffic to this provider.

When the Bgpd is started, the monitors are initialized and one SNMP query is sent towards each router, in order to check the status of the BGP sessions with providers. If there is no reply, the Internal Monitor will send up to two more SNMP requests, separated by a keepalive interval.

i Internal monitors for Internet Exchange peering partners are not initialized until there is an improvement made towards it. When running in non-intrusive mode internal monitors for IX peers are not initialized at all.

If none of the SNMP queries returned a status confirming that the sessions with providers are up, the provider will be assigned a FAIL status and the Internal Monitor will continue the periodical SNMP polling (each 60 seconds), to recheck providers sessions' status.

Then, the BGP session with the edge routers is initialized and Bgpd starts retrieving the routing table from the edge routers. While IRP retrieves the routing table, SNMP request may timeout due to the high CPU usage on the edge routers.

For details refer:

`bgpd.mon.guardtime`

`bgpd.mon.keepalive`

`bgpd.mon.holdtime`

`bgpd.mon.longholdtime`

1.2.5.2 External monitor

External BGP Monitor analyzes the network reachability through a specific provider. It performs ICMP/UDP ping requests towards the configured remote IP address(es) (`peer.X.ipv4.mon` or `peer.X.ipv6.mon`) through the monitored provider. If any of the configured IP addresses are accessible, the monitor is marked as OK. If the monitored remote IP addresses do not reply through the examined provider IRP will react as follows:

- the provider will be marked as FAILED,
- all the improvements towards this provider will be withdrawn from the routing table,
- new improvements towards this providers will not be made.

If for some reason (e.g. when the provider’s interface goes down state), the Next-Hop of the Policy Based Routing rule does not exist in the routing table, then the packets forwarding may return to the default route. In that case, the External BGP Monitor will return a false-positive state. To avoid that by properly configuring PBR, please consult “Specific PBR configuration scenarios” ([Specific PBR configuration scenarios](#)).

The External BGP Monitor status does not depend on the state of the BGP session(s) between the edge router and the provider (which is monitored by the Internal BGP Monitor). Therefore, in the case that the BGP session with one of the providers goes down, the External Monitor still shows an OK state which will remain unchanged as long as the packets are successfully routed towards the monitored destination. We do recommend adding at least two remote IP addresses, in order to prevent false-positive alerts.

When both BGP monitors are enabled (`peer.X.mon.ipv4.internal.state`, `peer.X.mon.ipv4.external.state`), they function in conjunction with each other. If any of them fails, the provider will be declared as FAILED and IRP will react as described above. The BGP monitors’ statuses are displayed on the system dashboard as shown in the screenshot below.

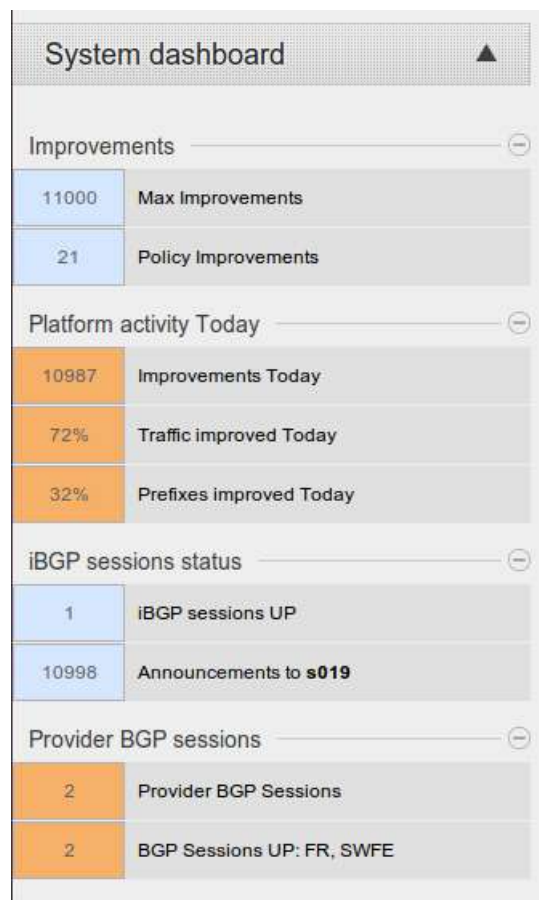



Figure 1.2.2: System Dashboard

 Starting with version 1.8.5, IRP requires at least the Internal Monitor to be configured. Otherwise, the system Frontend will return an error as shown below.

Attention! The Internal BGP Monitor is not configured. This may cause traffic misrouting in case of provider failures! 

Figure 1.2.3: Error: Internal BGP Monitor not configured

For details refer:

peer.X.mon.snmp

peer.X.ipv4.mon

peer.X.ipv6.mon

peer.X.mon.ipv4.bgp_peer


peer.X.mon.ipv6.bgp_peer

1.2.5.3 BMP monitoring station

A BMP monitoring station is included in IRP starting with version 3.9. It implements the monitoring station specified in RFC 7854 BGP Monitoring Protocol (BMP). The BMP monitoring station requires a monitored router to communicate over BMP the detailed routing information received from neighbors. The BMP monitoring station exposes detailed routing data to other IRP components so that better and timelier decisions are made, for example:

- BMP lists both active and inactive routes advertised by peers on an Internet Exchange. The additional information is used by IRP to evaluate and identify the best candidate peers at all times. Without BMP data IRP has knowledge about active routes only which only point to a single peer on the IX while all the alternatives are hidden.
- route changes even for inactive routes are visible via BMP. This allows IRP the opportunity to revisit previously made probes and improvements not only at predefined re-probing intervals but also when route changes are detected for both active and inactive routes.
- prefix monitors for IX improvements consume significant router CPU resources in order to service the SNMP requests traversing the router's relevant OIDs. More so this information is at times inaccurate and vendor dependent. When BMP data is available IRP uses this routing data to determine if IX peers still advertise the routes and no longer makes the SNMP requests for those prefixes thus significantly reducing the CPU overhead especially on routers servicing very large IX.
- IRP reconstructs the AS Path for candidate providers in order to make accurate iBGP announcements of improvements. Unfortunately network configuration practices might cause some errors during reconstruction of AS Paths using traceroute. BMP data makes the reconstruction of AS Path redundant and more accurate as this BGP attribute can be retrieved from actual (inactive) routes received from neighbors.
- improvements can be re-visited on AS Path changes. Both new and old provider AS Path attributes are monitored via BMP for changes. When changes are detected IRP re-probes the prefix to ensure the network uses the best available route. Note that re-probing can be triggered on any AS Path changes or only on major ones - when AS Path traverses a different set of autonomous systems.

The possible benefits of passing BMP data to IRP are many. To benefit from them the monitored router must support BMP too. Configuration is fully performed on monitored router by pointing it to the IRP BMP monitoring station IP address and port. The monitored router establishes the TCP connection and communicates the data while the IRP BMP monitoring station continuously listens and accepts fresh routing data as it comes.

 As per BMP RFC requirements IRP BMP monitoring station never attempts to establish BMP or any other connections with the monitored router leaving the full scope of decisions regarding when and if BMP data is communicated in network's responsibility.

⚠ Any route filtering applied to a BGP session with Partial Routing provider or IX peering partner wouldn't not be taken into consideration by a BMP sender. If filtering is deemed as important, then IRP shouldn't use BMP data to find routes (`peer.X.bmp.check_routes`).

1.2.6 Outage detection

A complete traffic path from source to destination typically passes through multiple networks, with different AS numbers. This is reflected in the traceroute results. If the Outage detection is enabled in the IRP configuration, the system gathers network performance information for all traceroute hops over which the traffic passes to the remote networks. Next, each hop is translated into AS numbers. In case any network anomalies are detected on a specific ASN, then this ASN and the immediate neighbor ASN are declared a problematic AS-pattern. The system then re-probes the prefixes that pass through this AS-pattern. In case the issue is confirmed, all related prefixes are rerouted to the best performing alternate provider.

The Outage detection uses a statistical algorithm for selecting the best routing path. Rerouting will occur for all the prefixes that are routed through the affected as-pattern, despite their current route .

ℹ Several improvements-related reports need to indicate the original route for a specific prefix. This value is taken from the last probing results for this prefix, even if the results are outdated (but not older than 24h). Since the outage-affected prefixes are rerouted in bulk by as-pattern, in some cases the reports can show the same provider for both the old and the new route.

1.2.7 VIP Improvements

The VIP Improvements is a feature that allows manual specification of a list of prefixes or AS numbers that will be periodically probed by IRP and optimized in compliance with the probing results. This allows the system to monitor specific networks or Autonomous Systems, without reference to the data provided by the IRP collector.

Possible usage scenarios include, but are not limited to:

- monitoring and optimizing traffic to commercial partners that should have access to your networks via the best performing routes
- monitoring and optimizing traffic to your remote locations, operating as separate networks
- monitoring and optimizing traffic to AS, which are known for the frequent accessibility issues due to geographical or technical reasons

ℹ If a prefix is being announced from multiple Autonomous Systems, you can see different ASNs in Current improvements report in the prefixes translated from ASN

IRP performs the proactive (more frequent than regular probing) monitoring of the VIP prefixes/ASNs that allows VIPs to be constantly improved.

For future reference, see:

`core.vip.interval.probe`


1.2.8 Retry Probing

Retry Probing is a feature that allows reconfirmation of the initial and already reconfirmed improvements validity. The feature is applicable to all types of improvements made by the system (Performance, Cost and Commit Control improvements). The improvements that were made more than a retry probing period ago (`core.improvements.ttl.retry_probe`) are being sent to retry probing. If the probing results

confirm that the current improvement is still valid, it stays in the system and its description is updated. Otherwise, it will be removed with the log message further described in this section.

During Retry Probing reconfirmation the improvement details will be updated in the following cases:

- Performance and Cost improvements
 - An old provider has been removed from the system configuration.
Example: “Old provider and performance metrics not known. New packet loss 55%, avg rtt 105 ms.”
- Commit Control improvements
 - An old provider’s has been removed from the system configuration.
Example: “Previous provider not known. Rerouted 1 Mbps to Peer5[5] (250 Mbps, 50%)”
 - An old provider’s bandwidth statistics are not available.
Example: “Rerouted 6 Mbps from Peer1[1] to Peer5[5] (250 Mbps, 50%)”
 - A new provider’s bandwidth statistics are not available.
Example: “Rerouted 6 Mbps from Peer1[1] (250 Mbps, 50%) to Peer5[5]”
 - The old and new providers’ bandwidth statistics are not available.
Example: “Rerouted 6 Mbps from Peer1[1] to Peer5[5]”

 Commit control improvements are reconfirmed based on their average bandwidth usage (and not on current bandwidth usage). This way if performance characteristics allow it, even when current bandwidth usage is low but the average is still relevant, the improvement is preserved thus anticipating network usage cycles and reducing number of route changes.

During Retry Probing reconfirmation the improvements will be removed from the system and the details will be logged into the log file (`core.log`) in the following cases:

- The Commit Control feature has been disabled.
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (Commit Control is disabled)”
- The low prefix traffic volume is less than the configured bandwidth limits (`core.commit_control.agg_bw_min`).
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (low traffic volume, irrelevant for the Commit Control algorithm)”
- The system has been switched from the Cost mode to the Performance mode (applied for cost improvements only).
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (Performance Improvements mode)”
- A prefix has been added to the ignored networks/ASN list.
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (added to ignored networks/ASN)”
- The improvement’s performance metrics are not the best ones anymore.
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (Performance Improvement not actual anymore)”
- The maximum number of improvements limits (`core.improvements.max`, `core.improvements.max_ipv6`) are exceeded.
Example: “Prefix 1.0.2.0/24 withdrawn from Improvements (no more available Improvement slots)”

For future references, see:


`core.eventqueuelimit.retry_probe_pct`


`core.improvements.retry_probe.volume_top_n`

`core.improvements.ttl.retry_probe`


1.2.9 Routing Policies (Limited to 3 policies in IRP Lite)

Routing Policies brings the capability of defining specific routing policies according to business objectives. This feature permits denying or allowing providers to be used for probing and reaching a specific prefix or ASN. It also provides the possibility to set a static route through a particular provider. Within a policy you can choose between VIP or non-VIP (regular) probing mechanisms to be used.

 Policies can be configured for networks (ASN) or aggregate prefixes. These are unpacked into existing specific sub-prefixes from the routing table before being processed by IRP components.

 In version 3.9 IRP added support for policies by Country. Note that IRP maps individual prefixes to a country and does not use a transitive inference based on AS records. These prefix mappings allow IRP to more accurately work with large transcontinental AS.

Starting with version 3.9 IRP introduces a priority attribute that defines what policy to choose in cases when different policies include the same unpacked specific prefix. The policy with the highest priority will apply for such a prefix.


 Note that after upgrade all existing policies are assigned (implicitly) the lowest default priority of 0.


Below you can find some typical scenarios of Routing Policies usage. For instance, there may be a specific prefix that consumes a high volume of traffic and IRP redirects it through the most expensive provider due to performance considerations. At the same time you have another two less costly available providers. In this case, you could deny the expensive provider for the specific prefix and IRP will choose the best-performing provider between the remaining two. This can be achieved by applying a Deny policy to the expensive provider or an Allow policy to the less costly providers.

The table below shows which cases a specific policy can be applied in, depending on the number of providers.

No. of providers \ Policy	Allow	Deny	Static
1 provider	No	Yes	Yes
2 providers or more (maximum: total number of providers - 1)	Yes	Yes	No
All providers (VIP probing disabled)	No	No	No
All providers (VIP probing enabled)	Yes	No	No

If a Static Route policy is applied to a prefix, VIP probing through each of the providers is unnecessary. Regular probing will suffice for detection of a provider failure that would trigger IRP to reroute the traffic to a different provider. Therefore IRP does not allow using VIP probing within a Static Route policy.

 Routing policies are designed to control outgoing paths for destination networks that are outside your infrastructure. This feature should not be used to manipulate your infrastructure network behavior.

 Avoid setting up of policies that point to same prefix. When improvement by aggregate is enabled, multiple prefixes can point to the same aggregate and this can cause unpredictable behaviour.

If a provider is added or removed, suspended or shutdown, the routing policies are adjusted by IRP in the following way:

A new provider is added.

Policy	Result
Allow all providers (VIP probing enabled)	The new provider is automatically included into the configured policy and probed by the VIP probing mechanism.
Allow selected providers only	The new provider is automatically ignored by the configured policy and not probed by the probing mechanism.
Deny	The new provider is automatically included into the configured policy and probed by the selected probing mechanism.
Static Route	The new provider is automatically ignored by the configured policy and probed by the Regular probing mechanism.

The provider under the policy is removed.

Policy	Result
Allow all providers (VIP probing enabled)	The new provider is automatically removed from the configured policy and not probed by the VIP probing mechanism. If there is only one provider left, the policy is automatically deactivated.
Allow selected providers only	The provider is automatically removed from the configured policy and not probed by the probing mechanism. If there is only one provider left, the policy is automatically deactivated.
Deny	The provider is automatically removed from the configured policy and not probed by the probing mechanism. If there is no any other provider under this policy, it is automatically deactivated.
Static Route	The provider is automatically not probed by the probing mechanism. The policy is automatically deactivated.


The provider under the policy is suspended.

Policy	Result
Allow all providers (VIP probing enabled)	The provider is temporarily removed from the configured policy and not probed by the VIP probing mechanism. If there is only one provider left, the policy is temporarily deactivated.
Allow selected providers only	The provider is temporarily removed from the configured policy and not probed by the probing mechanism. If there is only one provider left, the policy is temporarily deactivated.
Deny	The provider is temporarily removed from the configured policy and not probed by the probing mechanism. If there is no any other provider under this policy, it is temporarily deactivated.
Static Route	The provider is temporarily not probed by the probing mechanism. The policy is temporarily deactivated.

The provider under the policy is shutdown.

Policy	Result
Allow all providers (VIP probing enabled)	The provider is temporarily removed from the configured policy and not probed by the VIP probing mechanism. If there is only one provider left, the policy is temporarily deactivated.
Allow selected providers only	The provider is temporarily removed from the configured policy and not probed by the probing mechanism. If there is only one provider left, the policy is temporarily deactivated.
Deny	The provider is temporarily removed from the configured policy and not probed by the probing mechanism. If there is no any other provider under this policy, it is temporarily deactivated.
Static Route	The provider is temporarily not probed by the probing mechanism. The policy is temporarily deactivated.


Policies that target an AS can be cascaded. Cascading applies the same policy to AS that are downstream to target AS, i.e. to AS that are transited by target AS.


 The use case of cascading is to apply a policy to a remote AS that transits a few other AS. Still, a cascading policy can cover a huge number of down-streams. This number is parameterized and can be set to values that best fit customer's needs. Refer for example [4.1.4.17](#).

When multiple routing domains are configured a policy can be configured to prevent global improvements. Refer to [Optimization for multiple Routing Domains](#) for further details about routing domains.

Policies can be assigned a specific community and all policy based improvements will be marked with the designated value to allow their further manipulation on edge routers.

All Routing Policies are stored in `/etc/noction/policies.conf` file, which is automatically generated by the system.

 Do not alter the `/etc/noction/policies.conf` file manually because any modifications will be overwritten by the system. Any changes can only be performed from the Configuration -> Routing Policies section in the system Frontend.

 The rule will be ignored by the platform if the Routing Policy contains a syntax/logical error.

Please check ([Routing Policies settings](#)) for a detailed description of Routing Policies parameters.

1.2.10 Support for Centralized Route Reflectors

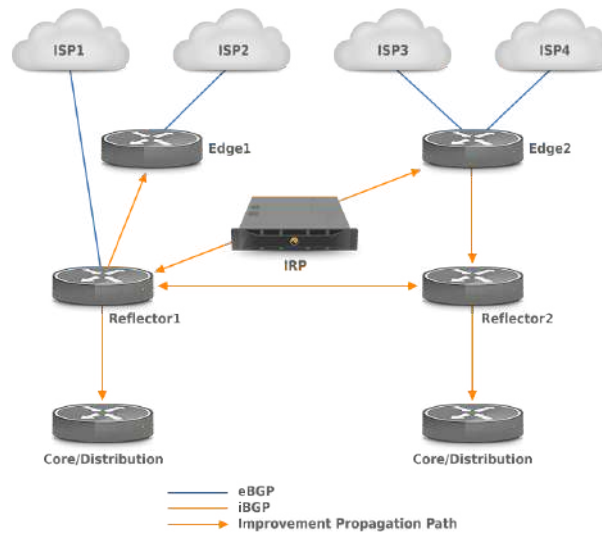


Figure 1.2.4: Support for Centralized Route Reflectors

IRP gives the possibility to advertise routes into one or more route reflectors which subsequently advertise improvements into upper and/or lower routing layers (such as edge, core or distribution).

In case the iBGP sessions can't be established between the IRP appliance and edge routers, a route reflector is used. The following restrictions apply for such a solution:

- The Next-Hop-Self option should not be used. A direct iBGP session is required between IRP and each of the divergence points (Reflector1, Edge2) in case it is enabled. This restriction is not applicable between reflectors and core/distribution layers.
- Next-Hop addresses should be accessible (exist in the routing table) where next-hop-self is not applied (Either static routes or IGP is used).
- An Internal Monitor should be configured to retrieve the eBGP session state from the device where the corresponding eBGP session is terminated. For example, ISP1 should be monitored on Reflector1, ISP2 on Edge1, and ISP3 and ISP4 on Edge2.
- Injecting routes to reflector(s) can cause temporary routing loops.

In order to announce improvements into route reflector, it should be configured as a BGP router in the "Configuration"→"BGP and routers" section and should be assigned to all the related providers in the "Configuration"→"Providers and Peers" section.

1.2.11 Support for Internet Exchanges (Not supported in IRP Lite)

A transit provider can deliver traffic to any destination on the Internet. However, within an Internet Exchange, a peering partner gives access only to the set of prefixes originated or transiting its network. Therefore, when IRP evaluates the Exchange as a best path, it has to know the prefixes announced by each peer, to avoid inefficient probing of paths that cannot lead to the desired destination.

With this purpose, IRP gets the routing table from the edge router containing the list of IPs and the corresponding next-hop; this represents the next router's IP address to which a packet is sent as it traverses a network on its journey to the final destination. IRP matches the prefix with the corresponding next-hop among the configured peers, allowing it to select for probing only those peers that have access to a specific prefix. This process is also performed in the case of a transit provider that gives access only to a limited set of prefixes, rather than the entire Internet.

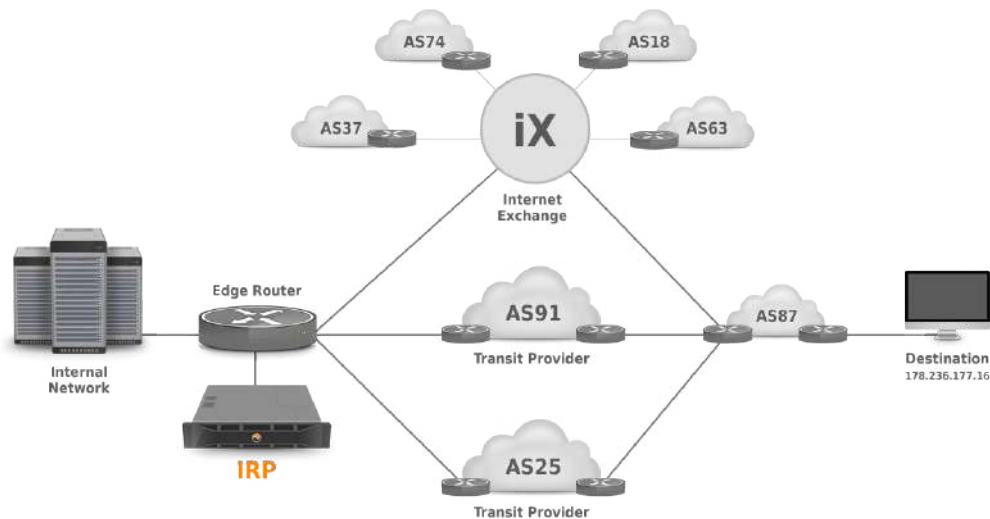


Figure 1.2.5: IRP configuration in a multi-homed network connected to transit providers as well as and Internet Exchange

In the case of multiple transit providers, there is an additional IP alias added on the IRP platform for each provider. The edge router is configured in such a way that traffic originating from each of these IPs is routed over different providers. This is done with the help of Policy Based Routing (PBR) or Flowspec policies.

With PBR, a network engineer has the ability to dictate the routing behavior based on a number of different criteria other than the destination network. These PBR rules are applied to make sure that IRP probes are following the desired paths. However, when it comes to Internet Exchanges, configuring hundreds of IP aliases on the platform would result in inefficient IP address usage and an unmanageable setup.

To avoid this, a set of PBR rules are applied making sure that the probes to be sent through a specific provider are originating from one of the configured IPs with a specific DSCP code assigned. DSCP - Differentiated Services Code Point - is a field in an IP packet that enables different levels of service to be assigned to network traffic. Since DSCP can take up to 64 different values, one configured IP can be associated with up to 64 peers. Although, due to this mechanism, the number of required IP addresses for aliases to be configured has decreased considerably, hard work would still be needed to configure the PBR on the edge router as described above.

To solve this, IRP implemented a built-in PBR config-generator which provides the configuration code to be used for a specific router model. By running this generated set of commands, network administrators can easily configure the required PBR rules on the router.

1.2.12 Optimization for multiple Routing Domains

Overview

Some networks have multiple Points of Presence interconnected both internally via inter-datacenter links and externally via multiple transit providers. The diagram below depicts an example diagram with the available routes to one destination on the Internet.

IRP uses the concept of Routing Domains to separate the locations. A Routing Domain's main characteristic is that its routing tables are mainly built on data received from its locally connected providers and the preferred routes are based on locally defined preferences.

The process of optimizing outbound network traffic in such a configuration is to mainly find better alternative routes locally (within a Routing Domain) and only reroute the traffic to other Routing Domains via inter-datacenter links when local routes are completely underperforming.

It must be noted that a multiple Routing Domain configuration works best if the Points of Presence are not too far away (ex. a network with POPs in San Francisco, Palo Alto and Danville is perfectly suitable under this scenario).

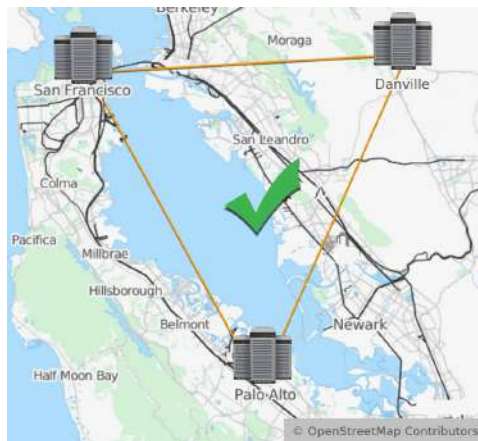


Figure 1.2.6: City wide network

POPs situated at larger distances, for example in Las Vegas and Salt Lake City are still supported by a single IRP instance running in San Francisco.

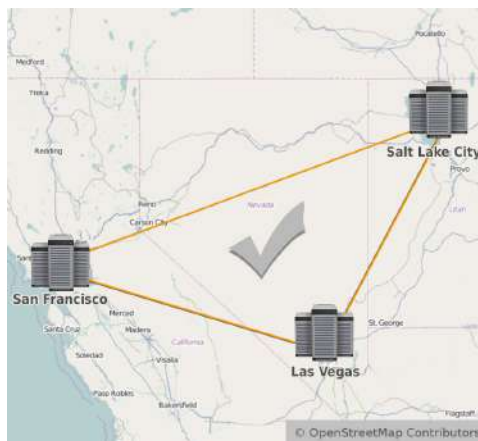


Figure 1.2.7: Regional network

Intercontinental links for POPs in Tokyo and Melbourne are way too far away from the IRP instance in San Francisco and in such a case multiple IRP instances are required.

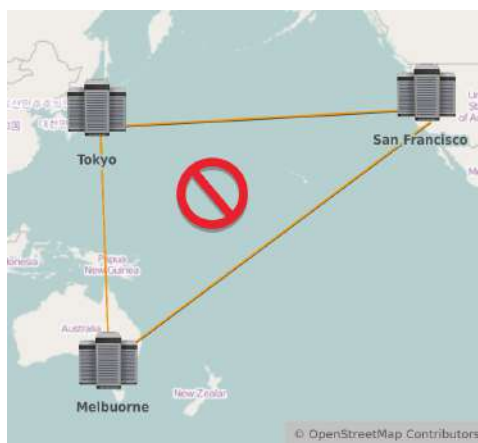


Figure 1.2.8: Intercontinental network

Multiple Routing Domains implementation attributes To further detail the multiple routing

domain attributes the following diagram will be used:

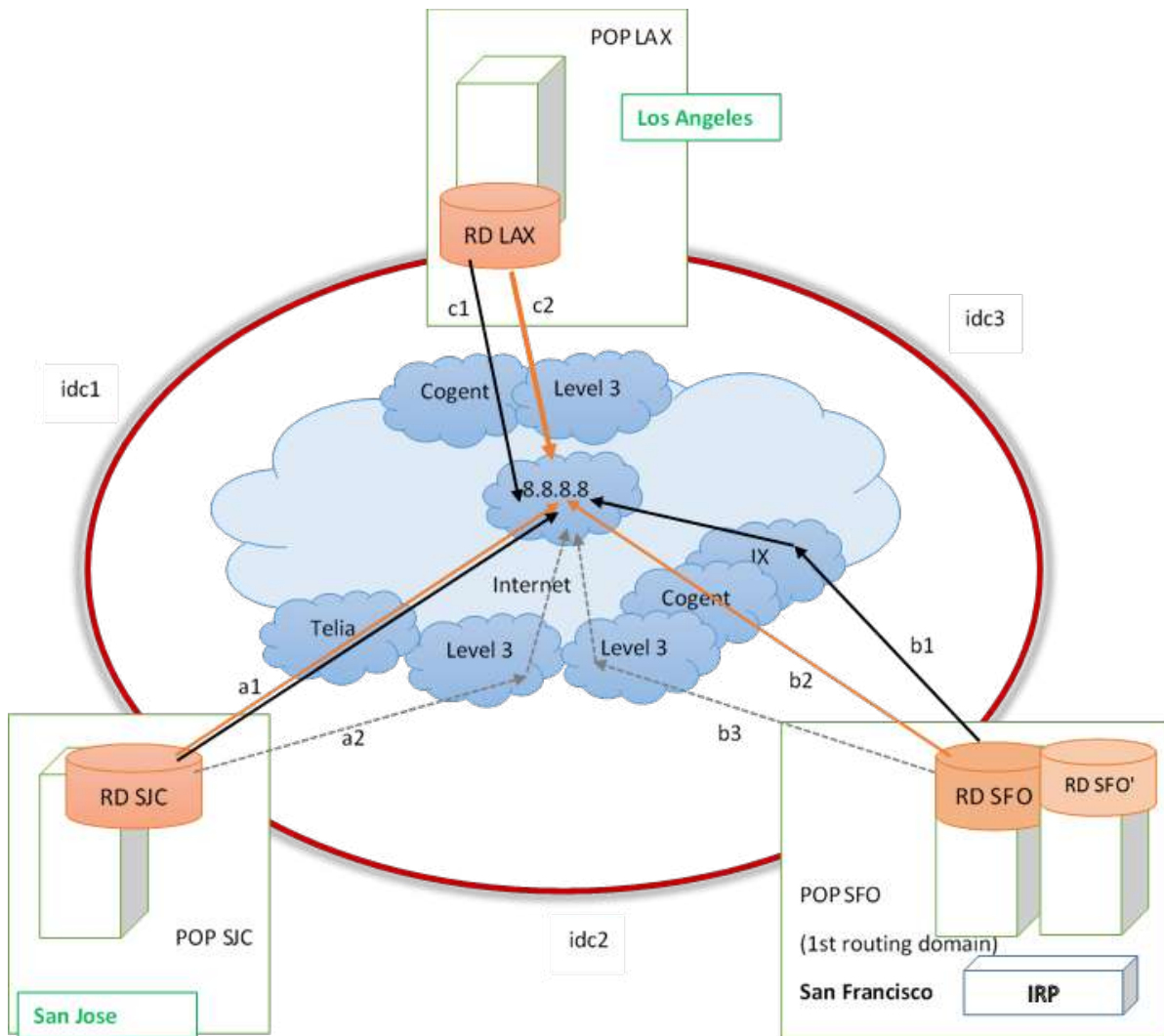


Figure 1.2.9: Multiple routing domains

A multiple Routing Domain configuration has a series of attributes:

- Multiple locations belonging to the same network (AS) represented in the diagram by POP SJC, POP SFO and POP LAX (of course, more than 3 routing domains are supported).
- The locations are distinguished by the different Routing Domains within which they operate (depicted by RD SJC, RD SFO, and RD LAX)
- The Routing Domains are managed by edge routers belonging to different locations
- Nearby locations that process routing data differently should be split into different Routing Domains, even if they have the same upstream providers. In the diagram above RD SFO and RD SFO' are depicted as part of a single Routing Domain. A decision to split or keep in the same routing domain should be made based on exact knowledge on how routing data is processed.
- Inter-datacenter loop interconnects the different locations (depicted by idc1, idc2 and idc3 segments)
- Data flows between locations take only the short path (in the example POP SJC can be reached from POP SFO via idc2 path (short) or idc3 + idc1 path (long))
- Each Routing Domain has different providers and different preferred routes to reach a specific destination (a1, b1, c1)

- A single IRP instance collects statistics about traffic (Irpflowd only), probes available destinations and makes improvements towards specific prefixes/networks on the Internet.
- IRP assumes RTT of zero and unlimited capacity to route traffic within a Routing Domain
- IRP assumes that Sites are not physically too far away. It is ok to have different sites in the same city or region as at this scale inter-datacenter links have predictable characteristics. When taking intercontinental links into consideration this is quite probably not the case.
- Distances between sites (idc1, idc2, idc3 delays) are measured in advance and specified in IRP's configuration.

Inter-datacenter link characteristics Support for Multiple Routing Domains relies on existence of inter-datacenter links. These links should be independent of upstream providers.

Example of inter-datacenter links that multiple routing domains is designed for are:

- private connections,
- L2 links with guaranteed service,
- MPLS links

⚠ VPNs via public Internet could be used with Multi Routing Domain feature but is a suboptimal choice. Under such conditions IRP MUST be prevented from making Global Improvements. This way IRP will do only local optimizations in each Routing Domain and will operate similarly to multiple IRP instances (while probing excessively because it probes destinations via remote locations too).

Constraints At the moment IRP multiple Routing Domains implementation does not cover the following:

- IRP does not take measurements of inter-datacenter link delays (idc1, idc2 and idc3). This values are configurable.
- IRP does not monitor if inter-datacenter links are operating normally. In case such a link is broken it is expected IRP to loose BGP connectivity with routing domain routers and this will cause IRP improvements to be withdrawn till the link is restored.
- IRP does not try to detect if the traffic is following long or short paths on the inter-datacenter links. In the image above traffic from RD SJC can follow path idc1 (short) or idc2+idc3 (long). IRP always assumes the short path is being followed internally.
- IRP does not take measurements of inter-datacenter link capacity and current bandwidth usage. At this stage IRP assumes there is enough inter-datacenter link capacity to also carry the (few) global improvements. Also, IRP tries to minimize usage of inter-datacenter links.

Routing domains

Routing domain is a generic term used to distinguish a logical location that works with different routing tables. The differences are caused by the fact that a router composes its routing table according to routes received from different providers. It is possible to have multiple routing domains in the same datacenter if routing data is received by different routers (even from same or different sources) and data flows are distributed via different routers by different policies. In the image above RD SFO and RD SFO' can represent a single routing domain or multiple routing domains depending on what routing policies are applied.

Different routing domains are assigned identifiers in the range 1-100. Routing Domain identifier is assigned individually to each provider via parameter `peer.X.rd`. It must be noted the Routing domain that hosts the IRP instance is considered as the first routing domain (RD=1).

Parameter `global.rd_rtt` gives the distances between routing domains. The format of the parameter is

```
rda:rdb:rtt
```

for example if RD SJC has Routing Domain id = 42, RD SFO - 1 (since it hosts IRP), RD LAX - 3 then the idc1, idc2 and idc3 rtt is defined as the collection:

```
global.rd_rtt = 3:42:20 42:1:17 1:3:35
```

This parameter will be validated for correctness and besides the format above it requires that RD SJC and RD SFO values are different and already configured (RD1 is always present).

i Round trip time between one routing domain and another is calculated by executing PING towards edge routers and taking average integer value:

```
$ ping X -c 10 -q
PING X (X) 56(84) bytes of data.
--- X ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9085ms
rtt min/avg/max/mdev = 40.881/41.130/41.308/0.172 ms
```

Flow agents

A very natural constraint for Multiple Routing Domain networks is that IRP can rely only on Flow statistics - NetFlow or sFlow.

- SPAN cannot be used because it does not carry attributes to distinguish traffic between different providers

Flow collector needs to know the exact details of such a configuration in order to correctly determine the overall provider volume and active flows. For this each provider in an MRD setup must be assigned Flow agents to enable IRP to match Flow statistics accordingly. Refer [Flow agents](#) for further details.

Global and local improvements

Local improvements Local improvements represent better alternative routes identified within a routing domain. If in the example image above current routes are represented by black lines then local improvements are depicted by orange lines b2 and c2. Keep in mind that a1 just reconfirmed an existing current route and no improvements are made in such a case.

Local improvements are announced in their routing domains and this has the characteristic that local traffic exits customer's network via local providers. This also means that inter-datacenter interconnects are free from such traffic.

IRP prefers local routes and Improvements to Global improvements.

Parameter `bgpd.rd_local_mark` specifies a community marker that distinguishes local improvements from Global Improvements. A BGP speaker should not advertise these improvements outside its Routing Domain. It must be noted that a single marker is used for all the routing domains and each of them shall be configured to advertise local improvements within the domain and filter it out for inter-domain exchanges.

Local improvements should be stopped from propagating across routing domains. A route map is used to address this. Below are listed sample route maps for Cisco IOS and JUNOS 9.

Cisco IOS

- Refer your router capabilities in order to produce the correct route map. The route map **MUST** be integrated into existing route maps. It is not sufficient to simply append them.

```
neighbor <neighbor-from-another-RD> send-community (should be configured
  for all iBGP sessions)

ip community-list standard CL-IRP permit 65535:1
route-map RM-IRP-RD deny 10
  match community CL-IRP
route-map RM-IRP-RD permit 20

router bgp AS
  neighbor <neighbor-from-another-RD> route-map RM-IRP-RD out
```

Refer [Route-Maps for IP Routing Protocol Redistribution Configuration](#)

JUNOS 9

⊖ Refer your router capabilities in order to produce the correct route map. The route map **MUST** be integrated into existing route maps. It is not sufficient to simply append them.

```
policy-options{
  policy-statement IRP-CL {
    term 0 {
      from {
        protocol bgp;
        community IRP-RD;
      }
      then reject;
    }
    term 1 {
      then accept;
    }
  }
  community IRP-RD members 65535:1;
}
protocols {
  bgp {
    group ebgp {
      type external;
      neighbor 10.0.0.1 {
        export IRP-CL;
      }
    }
  }
}
}
```

Refer [Policy Framework Configuration Guide; Release 9.3](#)

Global improvements Global improvements are made when IRP identifies an alternative route that even after factoring in the latencies incurred by inter-datacenter interconnects are better than all existing alternatives. Such an example can be represented by alternative route c2 in the image above. A global improvement is made when one routing domain alternative is better than the best local alternatives in all other routing domains even considering the latencies incurred by inter-datacenter interconnects. In the image above c2 will become a global improvement if his loss characteristic is best to all alternatives and its latency:

- (c2+idc1 - margin) is better than best local alternative a1 in RD SJC

- $(c2 + idc3 - \text{margin})$ is better than best local alternative b2 in RD SFO

where:

- a1, b2 and c2 represent roundtrip times determined by IRP during probing of a destination.
- idc values are configurable and are set as one entry of `global.rd_rtt` parameter.
- margin is given by `core.global.worst_ms`.

i Global improvements can degrade performance in some routing domains. If performance for some routing domains degrades, IRP announcements for the global improvement also carry designated BGP community attributes set by `rd.X.community_worsening`.

Global improvements move traffic via inter-datacenter interconnects and as such are less desirable to local routes. Global improvements make sense when defined as above and even more sense when packet loss is taken in consideration and routing via a different datacenter reduces packet loss significantly.

1.2.13 Improvements weight

IRP assigns each improvement a weight. The weight takes into consideration many network and environment aspects of the change such as policy or VIP destinations, loss and latency differences, cost or commit control type of the improvement. Based on all of the above the improvement gathers more or less weight as appropriate.

Later on, instead of replacing oldest improvements that might still bring significant benefits with new improvements just because they are fresh, IRP relies on the weights to decide whether the benefit of the new improvement is sufficient to replace an existing one. More so, besides preserving the most relevant improvements this feature reduces route flapping by blocking announcement of new improvements and withdrawal of existing ones if the changes are not offering a good enough return.

1.2.14 Notifications and events

IRP produces a huge number of various events and some of them are critical for customer's awareness. Notifications allow customers to subscribe to any of the available events using the following channels:

- SMS
- Email
- Slack (via Webhook)
- SNMP Traps

IRP service `Irrpushd` provides this feature. In order for Notifications to be delivered correctly the corresponding channel configuration shall be provided. By default only email notifications can be delivered since IRP uses the embedded system email service to send them.

More so, users should subscribe for specific events.

⊖ Only events for valid subscriptions using correctly configured channels will be delivered.

Refer section [Notifications and Events](#) for details about configuring, subscribing and contents of notifications.

Refer section [Notification and events settings](#) for details about individual configuration parameter.

Events

The list of events monitored by IRP that can generate notifications is provided below.

When one of the IRP components detects a transition from normal to abnormal traffic behavior or back it fires these events:

- Abnormal correction: irpflowd
- Abnormal correction: irpspand
- Inbound traffic low: SPAN
- Inbound traffic low: Flow
- Inbound traffic normal: Flow
- Inbound traffic normal: SPAN
- Outbound traffic low: SPAN
- Outbound traffic low: Flow
- Outbound traffic normal: Flow
- Outbound traffic normal: SPAN

When Commit Control limits are exceeded per provider or overall one of the following events fires. Refer section 4.1.10 for configuring the actual limits of the events.

- Commit Control overload by X Mbps
- Commit Control overload by X%
- Commit Control provider X overloaded by Y Mbps
- Commit Control provider X overloaded by Y%

When an IRP component (re)loads the configuration it validates it and depending on results fires one of the following events:

- Configuration Invalid: Bgpd
- Configuration Invalid: Core
- Configuration Invalid: Explorer
- Configuration Invalid: irpapid
- Configuration Invalid: irpflowd
- Configuration Invalid: irpspand
- Configuration Invalid: irpstatd
- Configuration Ok: Bgpd
- Configuration Ok: Core
- Configuration Ok: Explorer
- Configuration Ok: irpapid
- Configuration Ok: irpflowd
- Configuration Ok: irpspand
- Configuration Ok: irpstatd

Outage detection algorithm fires one of the following events when it confirms congestion or outage problems and reroutes traffic around it:

- Congestion or Outage
- Outage: Confirmed and rerouted

Explorer periodically checks the PBRs and its expected probing performance and triggers the following events:

- Failed PBR (IPv6) check for provider
- Failed PBR (IPv4) check for provider
- Successful PBR (IPv4) check for provider
- Successful PBR (IPv6) check for provider
- Explorer performance low
- High number of VIP prefixes degrades IRP performance

IRP BGP Internal and External monitors fire the following events:

- ExternalMonitor (IPv4) Failed status for a provider. All improvements towards the provider will be withdrawn.
- ExternalMonitor (IPv4) OK status for a provider. All improvements towards the provider will be announced.
- ExternalMonitor (IPv6) Failed status for a provider. All improvements towards the provider will be withdrawn.
- ExternalMonitor (IPv6) OK status for a provider. All improvements towards the provider will be announced.
- InternalMonitor (IPv4) Failed status for a provider. All improvements towards the provider will be withdrawn.
- InternalMonitor (IPv4) OK status for a provider. All improvements towards the provider will be announced.
- InternalMonitor (IPv6) Failed status for a provider. All improvements towards the provider will be withdrawn.
- InternalMonitor (IPv6) OK status for a provider. All improvements towards the provider will be announced.

When statistics collection over SNMP is up or down IRP fires the following events:

- Provider SNMP stats down: X
- Provider SNMP stats up: X

Bgpd raises these events when BGP sessions are established/disconnected:

- IRP BGP session disconnected
- IRP BGP session established

When IRP identifies conditions to re-route traffic (make an improvement) and additionally it considers the differences to be excessive it raises these events:

- Excessive packet latency for prefix
- Excessive packet loss for prefix

- Improvements spike
- Low rate of announced IPv4 improvements
- Low rate of announced IPv6 improvements
- New improvement

Once an IRP component is started, stopped or restarted it raises the following events:

- Service started: Bgpd
- Service started: Core
- Service started: Explorer
- Service started: irpapid
- Service started: irpflowd
- Service started: irpspand
- Service started: irpstatd
- Service stopped: Bgpd
- Service stopped: Core
- Service stopped: Explorer
- Service stopped: irpapid
- Service stopped: irpflowd
- Service stopped: irpspand
- Service stopped: irpstatd

SNMP Traps

SNMP traps is a widely used mechanism to alert about and monitor a system's activity.

IRP SNMP traps not only notify about some IRP platform event but also include the list of varbinds which contain detailed information related to the thrown trap. The complete list of traps and varbinds with their descriptions can be found at `/usr/share/doc/irp/NOCTION-IRP-MIB.txt`

1.2.15 IRP API

IRP exposes a web API that uses HTTP verbs and a RESTful endpoint structure. Request and response payloads are formatted as JSON.

The API is running on the IRP instance and is reachable by default over SSL at port 10443. If called directly from the IRP instance server the API can be accessed at `https://localhost:10443` Use `https://hostname/api` in order to access the API from elsewhere on the network.

An IRP user id is required to access most of the API services. Use IRP's Frontend to manage users or configure external User Directories (refer [Security Configuration](#)). IRP API uses an authenticating mechanism based on authentication tokens. The token is passed as a query parameter for all API requests that require authentication.

The API comes with an API Reference available from IRP's Help menu and the API References also includes a sample PHP application with source code to aid in development.

IRP's API is powered by Irpapid service that can be started, stopped, configured like any other IRP service. Refer to the [4.1.3](#) section for Irpapid configuration parameter details.

1.2.16 IRP Failover (Not supported in IRP Lite)

Overview

IRP offers failover capabilities that ensure Improvements are preserved in case of planned or unplanned downtime of IRP server.

IRP's failover feature uses a master-slave configuration. A second instance of IRP needs to be deployed in order to enable failover features. For details about failover configuration and troubleshooting refer [Failover configuration \(Not supported in IRP Lite\)](#).

i A failover license is required for the second node. Check with Noction's sales team for details.

IRP's failover solution relies on:

- slave node running same version of IRP as the master node,
- MySQL Multi-Master replication of 'irp' database,
- announcement of the replicated improvements with different LocalPref and/or communities by both nodes,
- monitoring by slave node of BGP announcements originating from master node based on higher precedence of master's announced prefixes,
- activating/deactivating of slave IRP components in case of failure or resumed work by master,
- syncing master configuration to slave node.

i For exact details about IRP failover solution refer to configuration guides (2.6, 3.12.13.5), template files, and (if available) working IRP configurations. For example, some 'irp' database tables are not replicated, 'mysql' system database is replicated too, some IRP components are stopped.

- IRP versions 3.5 and earlier do not offer failover capabilities for Inbound improvements. It is advised that in these versions only one of the IRP instances is configured to perform inbound optimization in order to avoid contradictory decisions. In case of a failure of this instance inbound improvements are withdrawn.

An overview of the solution is presented in the following figure:

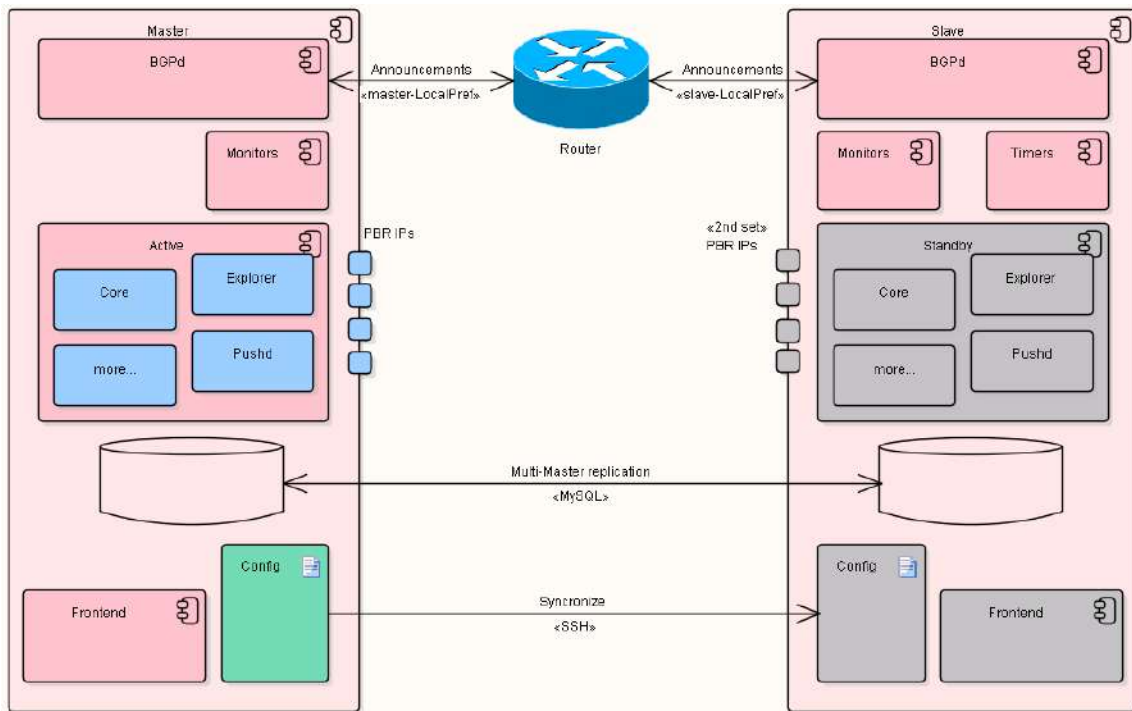


Figure 1.2.10: Failover high level overview

The diagram highlights the:

- two IRP nodes - Master and Slave,
- grayed-out components are in stand-by mode - services are stopped or operating in limited ways. For example, the Frontend detects that it runs on the slave node and prohibits any changes to configuration while still offering access to reports, graphs or dashboards.
- configuration changes are pushed by master to slave during synchronization. SSH is used to connect to the slave.
- MySQL Multi-Master replication is setup for 'irp' database between master and slave nodes. Existing MySQL Multi-Master replication functionality is used.
- master IRP node is fully functional and collects statistics, queues for probing, probes and eventually makes Improvements. All the intermediate and final results are stored in MySQL and due to replication will make it into slave's database as well.
- Bgpd works on both master and slave IRP nodes. They make the same announcements with different LocalPref/communities.
- Bgpd on slave node monitors the number of master announcements from the router (master announcements have higher priority than slave's)
- Timers are used to prevent flapping of failover-failback.

Requirements

The following additional preconditions must be met in order to setup failover:

1. second server to install the slave,
2. MySQL Multi-Master replication for the irp database.

➖ MySQL replication is not configured by default. Configuration of MySQL Multi-Master replication is a mandatory requirement for a failover IRP configuration. Failover setup, and specifically MySQL Multi-Master replication should follow a provided failover script. Only a subset of tables in irp database are replicated. Replication requires extra storage space, depending on the overall traffic and platform activity, for replication logs on both failover nodes.

1. a second set of BGP sessions will be established,
2. a second set of PBR IP addresses are required to assign to the slave node in order to perform probing,
3. a second set of improvements will be announced to the router,
4. a failover license for the slave node,
5. Key-based SSH authentication from master to slave is required. It is used to synchronize IRP configuration from master to slave,
6. MySQL Multi-Master replication of 'irp' database,
7. IRP setup in Intrusive mode on master node.

➖ In case IRP failover is setup in a multiple Routing Domain configuration and IRP instances are hosted by different RDs this must be specified in IRP configuration too. Refer [Optimization for multiple Routing Domains](#), `global.master_rd`, `global.slave_rd`.

Failover

IRP failover relies on the slave node running the same version of IRP to determine if there are issues with the master node and take over if such an incident occurs.

Slave's Bgpd service verifies that announcements are present on a router from master. If announcements from master are withdrawn for some reason the slave node will take over.


⚠ In order for this mechanism to work IRP needs to operate in Intrusive mode and master's node announcements must have higher priority than the slave's.

During normal operation the slave is kept up to date by master so that it is ready to take over in case of an incident. The following operations are performed:


- master synchronizes its configuration to slave. This uses a SSH channel to sync configuration files from master to slave and process necessary services restart.
- MySQL Multi-Master replication is configured on relevant irp database tables so that the data is available immediately in case of emergency,
- components of IRP such as Core, Explorer, Irppushd are stopped or standing by on slave to prevent split-brain or duplicate probing and notifications,
- slave node runs Bgpd and makes exactly the same announcements with a lower BGP LocalPref and/or other communities thus replicating Improvements too.

➖ It is imperative that master's LocalPref value is greater than slave's value. This ensures that master's announcements are preferred and enables slave to also observe them as part of monitoring.

In case of master failure its BGP session(s) goes down and its announcements are withdrawn.

 Slave node only considers that master is down and takes over only if master's Improvements are withdrawn from all edge routers in case of networks with multiple edge routers.


The same announcements are already in router's local RIB from slave and the router chooses them as best.

 This is true only if LocalPref and/or communities assigned to slave node are preferred. If other most preferable announcements are sent by other network elements, no longer announcements from slave node will be best. This defeats the purpose of using IRP failover.

At the same time, Failover logic runs a set of timers after master routes are withdrawn (refer `global.failover_timer_fail`). When the timers expire IRP activates its standby components and resumes optimization.


Failback

IRP includes failback feature too. Failback happens when master comes back online. Once Bgpd on the slave detects announcements from master it starts its failback timer (refer `global.failover_timer_failback`). Slave node will continue running all IRP components for the duration of the failback period. Once the failback timer expires redundant slave components are switched to standby mode and the entire setup becomes normal again. This timer is intended to prevent cases when master is unstable after being restored and there is a significant risk it will fail again.

 During failback it is recommended that both IRP nodes are monitored by network administrators to confirm the system is stable.

Recovery of failed node


IRP failover configuration is capable to automatically restore its entire failover environment if downtime of failed node is less than 24 hours.

 Recovery speed is constrained by restoring replication of MySQL databases. On 1Gbps non-congested links replication for a full day of downtime takes approximately 30-45 minutes with 200-250Mbps network bandwidth utilization between the two IRP nodes. During this time the operational node continues running IRP services too.

If downtime was longer than 24 hours MySQL Multi-Master replication is no longer able to synchronize the databases on the two IRP nodes and manual MySQL replication recovery is required.

Upgrades

Failover configurations of IRP require careful upgrade procedures especially for major versions.

 It is imperative that master and slave nodes are not upgraded at the same time. Update one node first, give the system some time to stabilize and only after that update the second node.

1.2.17 Inbound optimization (Not supported in IRP Lite)

Starting with version 3.4 IRP introduced optimization of Inbound traffic. Inbound bandwidth control reshapes the traffic from different providers targeting your sub-prefixes .

IRP uses well known and proven BGP mechanisms to instruct your routers to adjust their advertisements of your network segments to upstream providers and subsequently to the World. The adjusted advertisements take advantage of existing BGP policies implemented by edge routers worldwide in order to increase or decrease the preference of your internal network segments as advertised by one or another AS to the world. This allows more traffic to lean towards some of your upstream providers and less towards others. In case of an incident that your multihomed configuration is designed to be resilient against, the entire world still knows of the alternative routes towards your network and will be able to adjust accordingly.

Noction's IRP Inbound feature:

- advises your edge routers to advertise different prefixes of your network with different counts of BGP prepends to each of your upstream providers;
- monitors inbound and outbound traffic on your network interfaces using standard protocols (SNMP, NetFlow, sFlow) to determine if there is need for action;
- continuously assesses network health and performance to ensure that when overloads are possible it knows good and reliable alternative routes to use;
- uses a proprietary inferring mechanism that takes into account the inertial nature of the Internet to react to inbound preference changes and thus dampens the urge to “act now” and destabilize the network;
- provides you with configuration, monitoring and visualization tools that allow you to cross-check and validate its actions.

The entire Inbound optimization solution covers:

1. Segmenting your network into prefixes that are controlled by IRP
2. Coordinating communication over BGP between IRP and routers
3. Setting network conditions for making Inbound Improvements
4. Reviewing Inbound optimization results

i Starting with version 3.7 IRP has the capability to also monitor and improve transit routes. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#) for details.

The use cases below highlight typical scenarios when IRP's inbound bandwidth control capabilities might come handy:

Lets start with the case of unbalanced inbound and outbound peering You have a peering agreement with one of your neighbors to exchange traffic. Unfortunately the rest of the neighboring network configuration significantly unbalances the volumes of inbound and outbound traffic.

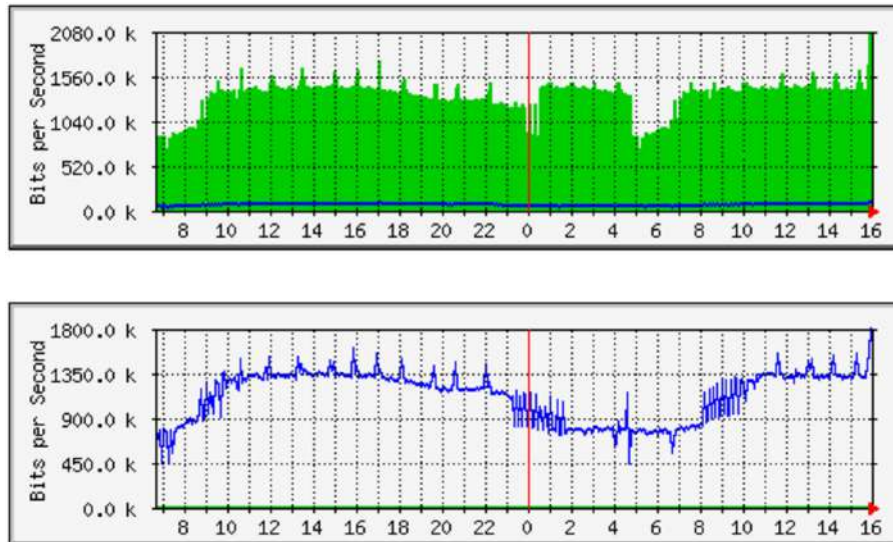


Figure 1.2.11: Unbalanced inbound/outbound peering

You rely on manipulating prefix announcements towards neighbors in order to shape the traffic. Unfortunately this is a reactive solution and consumes a lot of time while at the same time pushing the balance either one way or another. Often this pushes the network into the second typical scenario.

Fluctuating traffic shape A multihomed configuration overwhelms some links while other links remain barely used. Your network administrators frequently get alerts during peak network use and they manually add or remove prepends or altogether remove some inbound prefixes from being announced to affected neighboring links. These changes are sometime forgotten or other times just push the bulk of traffic towards other links and the problem re-appears.

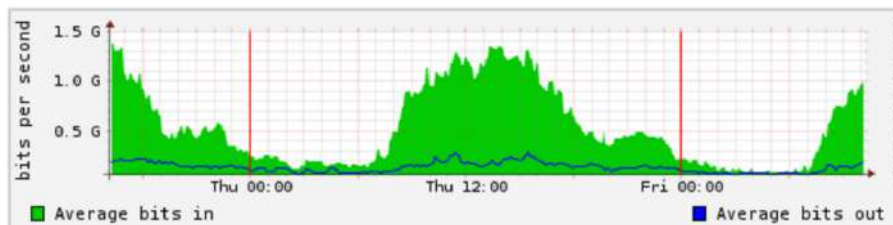



Figure 1.2.12: Fluctuating inbound traffic


For both above scenarios Inbound commit control in IRP can automate most of the inbound traffic shaping operations. It uses a typical solution for traffic shaping and specifically manipulates prepend counts announced to different providers and this eventually propagates across the Internet thus diverting traffic as desired. IRP automates your routine day to day traffic shaping actions and probably makes significantly more adjustments than you can afford to. At the same time it offers you reliable reviewing and fine-tuning options that will allow you to adjust IRP's work in a changing and evolving network.

Refer [Inbound configuration \(Not supported in IRP Lite\)](#), [Current inbound improvements \(Not supported in IRP Lite\)](#), [Inbound traffic distribution](#), [Inbound settings](#) for details.

1.2.18 Flowspec policies

 Flowspec policies can be used only in conjunction with Flowspec capable routers.

Starting with version 3.5 IRP has support of Flowspec policies. This means that Flowspec capability is recognized and can be used accordingly for BGP sessions established by IRP. In short Flowspec defines matching rules that routers implement and enforce in order to ensure that only desirable traffic reaches a network. Flowspec by relying on BGP to propagate the policies uses a well understood and reliable protocol.

 As specified by BGP, when a session disconnects all the announcements received from that session are discarded. The same is generally true for Flowspec policies. Still, since some vendors recognize Flowspec policies but implement them using capabilities other than BGP a confirmation is needed whether on session disconnect the specific router model indeed removes all the underlying constructs and reverts to a known state.


IRP not being involved in direct packet forwarding expects that Flowspec policies are implemented at least by your edge routers. If upstream providers also offer Flowspec support these policies can be communicated upstream where their implementation is even more effective.

Eventually Flowspec policies help ensure that traffic entering or exiting your network conforms to your plans and expectations. The main use cases that can be accomplished with Flowspec policies in IRP allows:

- controlling bandwidth usage of your low priority traffic towards external services, for example throttling bandwidth usage originating on your backup systems towards off-premises services.
- anticipating inbound traffic towards your services and shaping bandwidth use in advance, for example anticipating low numbers of legitimate customers from Russia, China or India on your e-commerce services and setting high but controllable rate limits on packets originating in those networks.
- reacting on a packet flooding incident by dropping specific packets, for example dropping all packets targeting port 53.
- redirecting some traffic for scrutiny or cleansing, for example forwarding port 80 packets through an intelligent device capable of detecting RUDY, slow read or other low-bandwidth/amplification attacks.

IRP Flowspec policies rely on a minimal set matching rules and actions that offer most of the capabilities while keeping the learning curve low and integration simple:

- Source or destination IP address specified as either CIDR format prefix or direct IP address
- Traffic protocols, for example TCP, UDP or ICMP
- Source or destination TCP/UDP ports
- Throttle, drop and redirect actions.

 It is important to note that IRP does not cross-validate Flowspec policies with improvements. While it is possible that for example a Flowspec redirect action pushes some traffic a different way to what an improvement advises, usually improvements cover many prefixes and while there will be a contradiction for one prefix there will be many other prefixes that IRP improves to compensate for these unitary abnormalities. It is recommended that Flowspec policies take precedence over improvements in order to benefit from this compensating nature of improvements.

Consider that depending on whether source or destination prefix belongs to your network the policy applies to either inbound or outbound traffic while the choice of ports allows targeting different traffic types.

Compare Flowspec policies to the already well known [Routing Policies \(Limited to 3 policies in IRP Lite\)](#). For further details regarding Flowspec configuration refer [Flowspec policies](#).

1.2.19 Throttling excessive bandwidth use with Flowspec

IRP can be configured to automatically add throttling Flowspec policies for prefixes that started using abnormal volumes of traffic. This feature can be used only if your network has Flowspec capabilities. Refer [Flowspec policies](#) for details about Flowspec.

⊖ In case thresholds for excessive bandwidth use are set to very aggressive levels IRP can create large numbers of Flowspec policies.

This feature can be described as:

- configure excess threshold and throttling multipliers
- periodically determine current and average prefix bandwidth usage for this hour of the day
- verify if current usage exceeds the average by a larger factor than the threshold multiplier
- rate-limit excessive bandwidth usage prefixes at their average use times throttling multiplier.

Past throttling rules are revised if/when prefix abnormal usage pattern ends.

For example, when a prefix usually consumes 1-2Mbps of traffic and its current bandwidth spikes tenfold and the spike is sustained for a significant period of time a throttling rule limiting usage by this prefix at 5-6Mbps will still offer ample bandwidth for normal service use and will also protect other services from network capacity starvation.

Refer [Throttling excessive bandwidth use with Flowspec, Core configuration](#) for details.

1.2.20 Maintenance windows

Maintenance works are on everybody's agenda in current fast paced and continuously evolving networks. During maintenance network engineers are very busy and will welcome any help their systems can offer in carrying out those works with the least amount of headaches. IRP is clearly not in the top of network engineer's priorities and asking to suspend or shutdown a provider immediately before a maintenance window starts and restart the provider back once the maintenance works end is not very helpful if not even annoying.

Instead IRP offers the facility to plan maintenance windows in advance. Knowing when a maintenance window starts and ends, IRP excludes that specific provider link from either performance optimization or bandwidth control. More so, IRP has the capability to reshape the traffic flowing in and out of a network to anticipate any downtime on a link.

ⓘ Setting a maintenance window by router sets each of the providers on the router with a maintenance window of their own.

Properly configured maintenance windows allows IRP time to move most of the outbound traffic and deflect most of inbound traffic away from the provider link that is scheduled for maintenance. Having only a small fraction of traffic or none at all on the maintenance link before the downtime starts avoids any (shall we say, catastrophic) spikes, possible overloads and consequently unpredictable behavior of the remaining live network equipment.

Specifically the following applies:

- a maintenance window is configured in advance and can be removed/revised at any time,
- a maintenance window sets details for single provider. If needed multiple maintenance windows can be setup and even overlapping maintenance windows are OK.
- IRP highlights maintenance windows in IRP's Frontend sidebar so that it is easy to spot current maintenance window status.
- optionally IRP can preserve existing improvements so that once the maintenance window ends improvements are reimplemented. It is advised that this feature is used only when the maintenance window is very short (a few minutes).
- an unloading period can be setup. During unloading IRP actively re-routes outbound prefixes through other available providers. While IRP is able to make most of the unloading improvements fast consideration shall be given to the announcement rate limitations setup in Bgpd in order for all the improvements to reach network routers in time for maintenance window starting time.
- a prepend time can be setup. This is only applicable if Inbound optimization is operational. If this time is setup then IRP will prepend configured inbound prefixes with the maximum allowed number of prepends through the provider link under maintenance in order to deflect inbound traffic towards other providers. Refer to [Inbound optimization \(Not supported in IRP Lite\)](#) for more details about Inbound optimization.

Refer [Maintenance windows](#) for details how to configure, review, create, edit, delete maintenance windows.

1.2.21 Optimization of transiting traffic (Not supported in IRP Lite)

Starting with IRP 3.7 IRP introduces optimization of transiting traffic capabilities. Optimization of transiting traffic is an enhancement of [Inbound optimization \(Not supported in IRP Lite\)](#).

Optimization of transiting traffic relies on the same method of influencing Internet-wide routing - manipulating best path selection by increasing AS Path length for a prefix carrying traffic on an undesirable interface. The most secure and effective way of AS Path manipulation is to apply these changes on a router facing a provider. Edge routers prepend routes according to designated communities.

Besides being an enhancement of Inbound optimization, transit prefixes are governed by a different set of constraints:

- The number of potential routes is very large and only some will/should be targeted for optimization. For this IRP uses filters by ASN/prefix allowing network administrators to configure what segments of the Internet to focus IRP's attention to.
- The improvements are visible on the Internet and excessive route changes can be flagged by external monitoring services as flapping or route instability. IRP protects against this by rate limiting number of route changes through a specific provider.
- Once a route has been improved all its traffic might be diverted away from this network and no new statistics will be available to make further inferences. In such cases IRP reverts old transit improvements during network's off-peak hour by either decreasing the number of advertised prepends or withdrawing the improvement altogether (configurable).
- Transit improvements apply to the same prefixes as do outbound improvements. The outbound improvements are withdrawn in order to avoid the risk of contradictory routing decisions.

Implementing optimization of transiting traffic introduces a series of risks and some inherent drawbacks, for example:

- Potential blackholing of traffic when all alternative routes are withdrawn. IRP implements a protection against this by traversing all RIB-in entries for improved transit prefixes and confirming that the route is still being announced by other providers besides IRP. Still there can be a short time period between confirmations when all alternative routes have been withdrawn and IRP did not yet get a chance to re-confirm this. Attempting to reduce this time period by setting up a

higher frequency of confirmations leads to increased load on the router and a tradeoff needs to be made for this. For example when the number of providers is quite large the probability that a route will be withdrawn through all of them is quite small and thus the frequency of confirmations can be reduced too.

- Additional CPU load on edge router(s) for servicing mandatory SNMP requests that check alternative route presence and BGP Best Path selection inside IRP.
- Working with missing BGP attributes that is not available over SNMP.
- Strict upper limits on the number of possible Inbound improvements are imposed by the trade-off required to reduce excessive router's CPU load.

Refer [Optimization of transiting traffic](#) for details.

1.2.22 Circuit issues detection

Starting with IRP 3.8 IRP adds excessive loss circuit issues detection features.

i Circuit issues detection feature is available for transit providers only.

When this feature is enabled for a provider IRP uses past probing data to detect when it suffers from excessive levels of packet loss. To determine excessive loss IRP compares a provider's average loss over an immediate past time horizon, number of probes and average loss difference from other providers. Depending on packet loss thresholds IRP can attempt different actions on the network.

Every time a circuit issue is detected IRP will raise corresponding alerts that can be subscribed to. Network engineers or external network management systems can act on them.

Additionally IRP even though it is constrained on how much can do, ensures that the following will take place:

- provider is marked with an issue badge and excluded from being considered a candidate for performance improvements,
- reprobing is performed for destination prefixes routed through affected provider,
- outbound improvements through affected provider are withdrawn,
- max prepends are announced for inbound and transit prefixes through affected provider,
- FlowSpec rules to induce drop of BGP session towards affected provider are added,

i Note that this is only possible if FlowSpec is enabled and the network is capable of processing FlowSpec rules. Dropping the BGP session with the affected provider causes all outbound and inbound traffic through this provider to be re-routed through known good providers.

- affected provider is monitored to detect if the issue was temporary and if loss averages return to normal restore it to a good state.

Enable this feature for each of the designated provider as detailed in [Configuration editor: Provider name](#) and review circuit issue detection thresholds as detailed in [Core configuration](#).

1.2.23 DDOS detection and Blackholing

i Starting with IRP 3.11.0, IRP introduces statistics collection and manual blackholing. Automated blackholing will be available in a future version.

Blackholing user interface is only available in Global Management Interface (GMI).

Blackholing feature allows redirection of traffic to a non-existent resource (a so-called black hole), or the blocking of the unwanted traffic in a provider's network to prevent such traffic from entering the user's network. The feature can be specially used to better understand and mitigate the effects of the Distributed Denial of Service (DDoS) attacks.

1.2.23.1 Configuring Blackholing

A Provider in IRP should be configured before it could be used for blackholing.

IRP should know next-hop (`bgpd.peer.X.blackholing.ipv4.next_hop`, `bgpd.peer.X.blackholing.ipv6.next_hop`), localpref (`bgpd.peer.X.blackholing.localpref`) and community (`peer.X.blackholing.community`) values to be able to send a route to a user's network.

A user's router is responsible to distinguish communities sent by an IRP instance and advertise blackholing routes to a provider's router used to receive such routes.

1.3 IRP Optimization modes

1.3.1 Performance optimization

Performance optimization mode makes sure that traffic flows through the best performing routes by reducing packet loss and latency, while ignoring other characteristics such as provider bandwidth usage or transit cost. The system analyzes each of the connected providers and compares only their performance metrics in order to choose the best candidate and make an improvement.

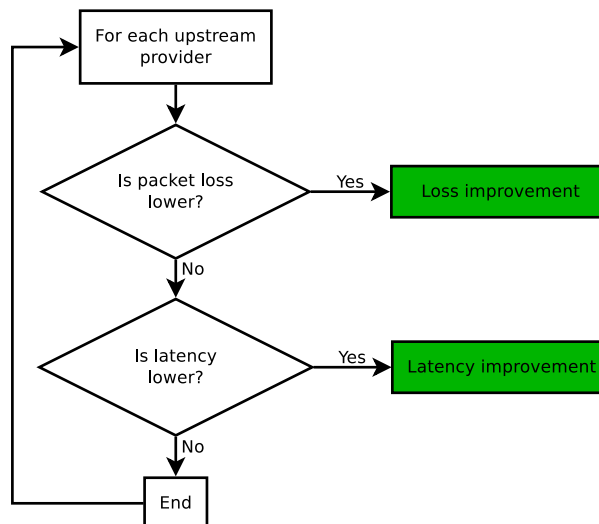


Figure 1.3.1: Performance optimization algorithm

First of all, IRP considers packet loss. If loss is lower and the difference is greater than a predefined value, then the system checks if sending the traffic through this provider will not cause any network congestion. If it confirms that the traffic can flow freely, the system declares the provider as the best one to route through.

However, if loss values are equal or the difference between providers is smaller than predefined, the system continues by comparing latency values. If latency is lower and the difference in latency between providers is greater than predefined, then the system declares a latency-based improvement.

1.3.2 Cost optimization

Cost optimization mode decreases packet loss and improves the cost while not worsening latency more than allowed. If IRP cannot improve costs it tries to reduce latency. The platform runs the same algorithm for loss comparison as in performance optimization mode.

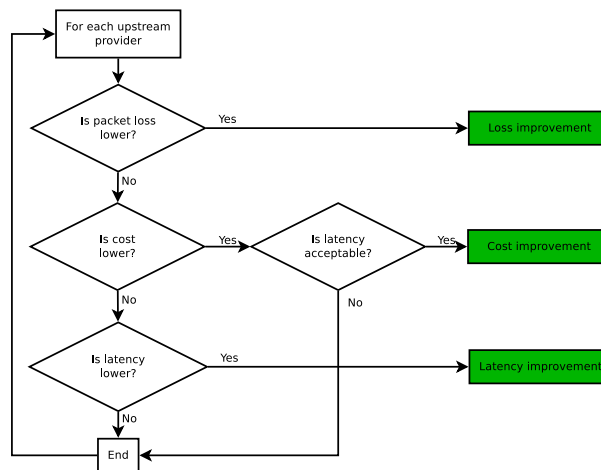


Figure 1.3.2: Cost optimization algorithm

i Note that the diagram does not highlight higher cost cases. IRP operating in cost optimization mode can make improvements towards higher cost providers only when current routes suffer from loss.

Before comparing latency values IRP compares the transit cost for each provider. If the cost of the new provider is better, the system goes further by checking the latency and validates the cost improvement only if the latency is not worsening more than predefined. However, if the cost cannot be improved, IRP tries to make a latency improvement using the same algorithm as in performance mode. Thus, if there is no way to make a cost improvement, system reroutes the traffic to the best performing provider.

1.3.3 Commit Control (Not supported in IRP Lite)

Commit Control, allows to keep the commit levels for each provider at a pre-configured level. It includes bandwidth control algorithms for each provider as well as the active traffic rerouting, in case bandwidth for a specific provider exceeds the configured limit. Commit Control also includes passive load adjustments inside each provider group.

A parameter called “precedence” (see `peer.X.precedence`) is used to set the traffic unloading priorities, depending on the configured bandwidth cost and providers throughput. The higher is the precedence, the lower is the probability for the traffic to be sent to a provider, when its pre-configured 95th percentile usage is higher. The platform will reroute excessive bandwidth to providers, whose current load is less than their 95th percentile. If all providers are overloaded, traffic is rerouted to the provider with the smallest precedence - usually this provider has either the highest available bandwidth throughput, or the lowest cost.

i IRP usually allows CC improvements when the candidate providers have better or equal loss to current route. This can be configured under `core.commit_control.loss_override`.

See also: `core.commit_control`.

1.3.3.1 Flexible aggressiveness of Commit algorithm based on past overloads

Metering bandwidth usage by the 95th presents the following alternative interpretation - the customer is allowed to exceed his limits 5% of times. As such, IRP assumes there's a schedule of overloads based on the current time within a month and an actual number of overloads already made.

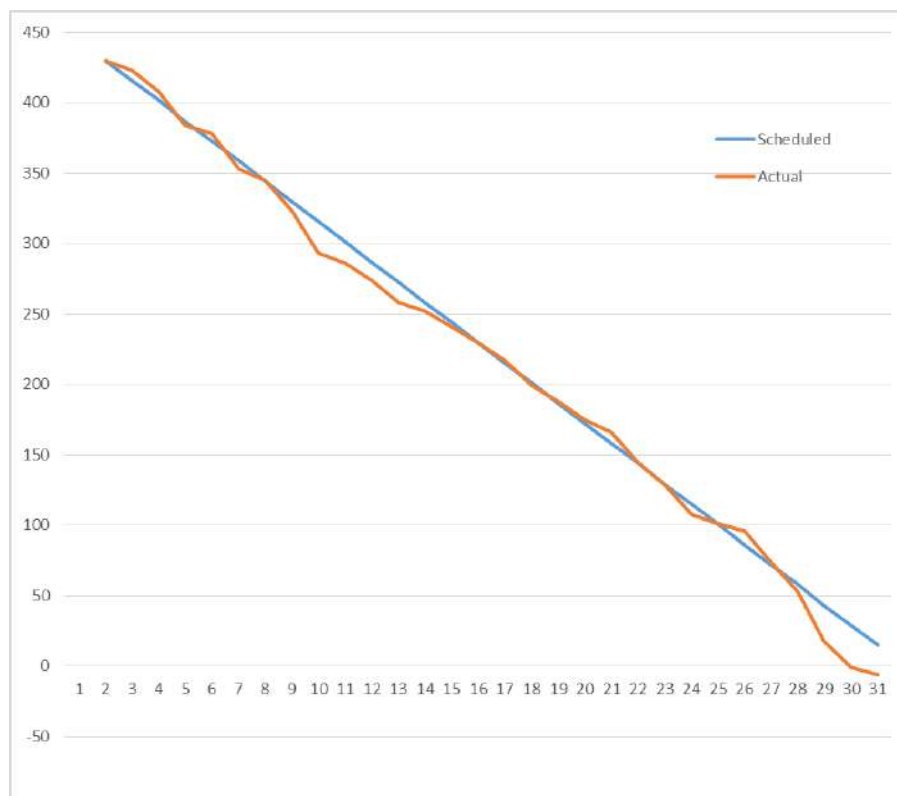


Figure 1.3.3: More aggressive when actual exceeds schedule

The sample image above highlights the remaining scheduled and the actual amount of allowed overloads for the month decreasing. Whenever IRP depicts that the actual line goes below schedule (meaning the number of actual overloads exceeds what is planned) it increases its aggressiveness by starting unloading traffic earlier than usual. For example, if the commit level is set at 1Gbps and IRP will start unloading traffic at possibly 90% or 80% depending on past overloads count.

The least aggressive level is set at 99% and the most aggressive level is constrained by configuration parameter `core.commit_control.rate.low + 1%`.

This is a permanent feature of IRP Commit Control algorithm.

1.3.3.2 Trigger commit improvements by collector

Commit control improvements are made after the flows carrying traffic are probed and IRP has fresh and relevant probe results. Subsequent decisions are made based on these results and this makes them more relevant. Still, probing takes some time and in case of fluctuating traffic patterns the improvements made will have reduced impact due to flows ending soon and being replaced by other flows that have not been probed or optimized.

For networks with very short flows (average flow duration under 5 minutes) probing represents a significant delay. In order to reduce the time to react to possible overload events IRP added the feature to trigger commit control improvements on collector events. When Flow Collector detects possible overload events for some providers, IRP will use data about past probed destinations in order to start unloading overloaded providers early. This data is incomplete and a bit outdated but still gives IRP the opportunity to reduce the wait time and prevent possible overloads. Later on, when probes are finished another round of improvements will be made if needed.

Due to the fact that the first round of improvements is based on older data, some of the improvements might become irrelevant very soon. This means that routes fluctuate more than necessary while on average getting a reduced benefit. This is the reason this feature is disabled by default. Enabling this feature represents a tradeoff that should be taken into consideration when weighing the benefits of a faster react time of Commit Control algorithm.

This feature is configurable via parameter: `core.commit_control.react_on_collector`. After enabling/disabling this feature IRP Core service requires restart.

1.3.3.3 Commit Control improvements on disable and re-enable

IRP up to version 2.2 preserved Commit Control improvements when the function was disabled globally or for a specific provider. The intent of this behavior was to reduce route fluctuation. In time these improvements are overwritten by new improvements.

We found out that the behavior described above ran contrary to customer's expectations and needs. Usually, when this feature is disabled it is done in order to address a more urgent and important need. Past Commit Control improvements were getting in the way of addressing this need and was causing confusion. IRP versions starting with 2.2 aligns this behavior with customer expectations:

- when Commit Control is disabled for a provider (`peer.X.cc_disable = 1`), this Provider's Commit Control improvements are deleted;
- when Commit Control is disabled globally (`core.commit_control = 0`), ALL Commit Control improvements are deleted.

It must be noted that the improvements are actually moved into a 'recycle bin' of sorts and if need be they can be restored after a short period of time. When Commit Control is enabled overall or for a Provider IRP will need some time to make a series of improvements that level traffic flows. This is only natural because IRP does not have statistics and has no knowledge of flow behaviors in the network.

Still, if Commit Control feature is re-enabled there's the possibility to restore old improvements based on past data and measurements. Unfortunately network characteristics fluctuate and IRP cannot restore any past improvements since they might be no longer relevant or even altogether wrong. As such, IRP will preserve and restore only improvements that are not older than the retry time interval which is configurable via parameter: `core.improvements.ttl.retry_probe`

1.3.3.4 Provider load balancing

Provider load balancing is a Commit Control related algorithm, that allows a network operator to evenly balance the traffic over multiple providers, or multiple links with the same provider.

For example, a specific network having an average bandwidth usage of 6Gbps has two separate ISPs. The network operator wants (for performance and/or cost reasons) to evenly push 3Gbps over each provider. In this case, both upstreams are grouped together (see `peer.X.precedence`), and the IRP system passively routes traffic for an even traffic distribution. Provider load balancing is enabled by default via parameter `peer.X.group_loadbalance`.

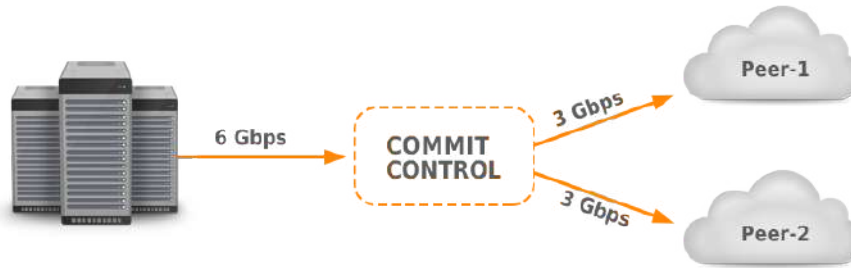


Figure 1.3.4: Provider load balancing

1.3.3.5 Commit control of aggregated groups

Customers can deploy network configurations with many actual links going to a single ISP. The additional links can serve various purposes such as to provision sufficient capacity in case of very large capacity requirements that cannot be fulfilled over a single link, to interconnect different points of presence on either customer (in a multiple routing domain configuration) or provider sides, or for redundancy purposes. Individually all these links are configured in IRP as separate providers. When the customer has an agreement with the ISP that imposes an overall limitation on bandwidth usage, these providers will be grouped together in IRP so that it can optimize the whole group.

The rationale of this feature as illustrated in the figure below is that if in the group overusages on one provider are compensated by underusages on another provider there is no need to take any action since overall the commitments made by the customer to the ISP have not been violated. Commit control algorithm will take action only when the sum of bandwidth usage on all providers in the group exceed the sum of bandwidth limits for the same group of providers.

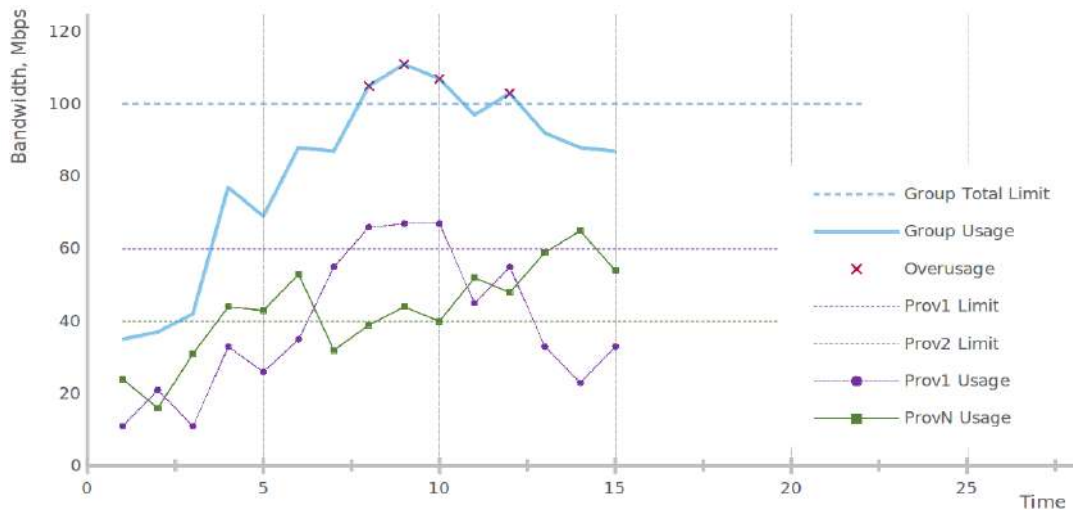


Figure 1.3.5: Commit control of aggregated groups

The image above highlights that many overusages on the green line are compensated by purple line underusages so that the group usage is below group total limits. Only when the traffic on the purple line increases significantly and there are no sufficient underusages on the other providers in the group to compensate the overusages, Commit Control identifies overusages (highlighted with a red x on the drawing above) and takes action by rerouting some traffic towards providers outside the group.

⚠ It is important to note that in order for this feature to be effective there must be providers configured in IRP that are not part of this group. This way when candidate improvements are

considered there are alternative routes via those providers that the traffic can be rerouted to.

In order to configure providers that are optimized as an aggregated group 1) first the providers will be configured with the same precedence in order to form a group; 2) the overall 95th limitation will be distributed across providers in the group as appropriate; 3) and finally load balancing for the group will be Disabled by parameter `peer.X.group_loadbalance`.

1.3.3.6 95th calculation modes

Commit control uses the 95th centile to determine whether bandwidth is below or above commitments. There are different ways to account for Outbound and Inbound traffic when determining the 95th value. IRP supports the following 95th calculation modes:

- Separate 95th for in/out: The 95th value for inbound and outbound traffic are independent and consequently bandwidth control for each is performed independently of each other. For this 95th calculation modes IRP monitors two different 95th for each inbound and outbound traffic levels.
- 95th from greater of in, out: At each time-point the greater of inbound or outbound bandwidth usage value is used to determine 95th.
- Greater of separate in/out 95th: 95th are determined separately for inbound and outbound traffic and the larger value is used to verify if commitments have been met.

Refer 4.1.12.5.

1.3.3.7 Other commit control features

Current improvements report has been improved to keep CC improvements applied during an algorithm cycle in a single sorted batch. This way the report no longer confuses by intermingling the improvements.

30	08:47:04	91.237.164.0/22	198640	B	A				Rerouted 9.11 Mbps from B[3] (44.61 Mbps, 109%) to A[2] (6.74 Mbps, 11%)
31	08:47:04	85.177.192.0/10	13184	B	A				Rerouted 6.09 Mbps from B[3] (35.49 Mbps, 87%) to A[2] (15.86 Mbps, 26%)
32	08:47:04	146.0.80.0/21	35382	B	A				Rerouted 5.70 Mbps from B[3] (29.40 Mbps, 72%) to A[2] (21.95 Mbps, 37%)
33	08:47:04	128.73.88.0/24	8402	B	A				Rerouted 4.26 Mbps from B[3] (23.71 Mbps, 58%) to A[2] (27.65 Mbps, 46%)

Figure 1.3.6: Sorted Commit Control improvements

The excerpt above highlights the new commit control improvements done at the same time. They unload 9.11, 6.09, 5.70, 4.26 Mbps from provider B to A thus unloading B from 109% to 58% and increasing the load on A from 11% to 46%. The data would have been more confusing if the improvements in that single batch were intermingled.

During retry probing IRP will delete Commit Control improvements with current bandwidth less than the configured threshold given by `core.commit_control_agg_bw_min`.

Chapter 2

Configuration

2.1 Configuration files

IRP is currently configured using a set of textual Unix-style configuration files, located under the `/etc/noction/` directory. In-line comments can be added to any line, using the “#” symbol.

Several separate configuration files are presented as follows:

- `/etc/noction/irp.conf` - the main IRP configuration file contains configuration parameters for all IRP components, including algorithm parameters, optimization modes definitions and providers settings.
- `/etc/noction/db.global.conf` - database configuration file for all IRP components, except the Frontend component.
- `/etc/noction/db.frontend.conf` - database configuration file for the Frontend component.
- `/etc/noction/exchanges.conf` - Exchanges configuration file (3.12.7.2).
- `/etc/noction/inbound.conf` - Inbound prefixes configuration file (3.12.12).
- `/etc/noction/policies.conf` - Routing Policies configuration file (1.2.9).
- `/etc/noction/user_directories.conf` - User Directories configuration file (3.12.10.2).

Additional configuration files can be used for several core, global and explorer preferences, as described in sections 4.1.2.25, 4.1.7.18 and 4.1.9.4.

A comprehensive list of all the IRP parameters along with their description can be found in the [Configuration parameters reference](#) chapter.

2.2 Global and Core Configuration

The default values, specified for the Core service are sufficient for a proper system start-up. Several parameters can be adjusted during the initial system deployment. For a comprehensive list please see the [Global parameters](#) and [Core settings](#) sections.

During the initial setup and configuration stage, one must pay attention to the following configuration parameters:

- `global.nonintrusive_bgp` - must be set to “1” until the configuration and route propagation tests are completed.
- `global.improve_mode` - must be configured according to the specific network operator policies. See also: [IRP Optimization modes](#)

- `global.aggregate` - in most cases, it is recommended to enable the aggregates, in order to reduce the number of prefixes advertised by the IRP. Please consult the network infrastructure and configuration.
- `core.commit_control` - should be configured according to the specific network operator policies, see also: [Commit Control \(Not supported in IRP Lite\)](#)
- `core.outage_detection` - in most cases it must be enabled. For more details see [Outage detection](#)

2.3 Collector Configuration

Depending on the preferred method of traffic data collection, one or both collector components should be configured. As specified in the [IRP Components](#) section, IRP can gather traffic data using the Flow (from now on: `irpflowd`) and Span collector (`irpspand`).

First of all, specific configuration to each collector will be described, along with the required router configuration changes.

2.3.1 Irpflowd Configuration

`Irpflowd` is a NetFlow/sFlow collector that receives and analyzes network traffic information, generated by your router(s).

NetFlow - an IP network statistics protocol developed by Cisco Systems, Inc. for collecting and transferring statistics regarding IP traffic information from network devices such as switches/routers to network analysis applications. `Irpflowd` currently supports the following **NetFlow** versions: v1, v5, v9.

sFlow - is a protocol designed for monitoring network, wireless and host devices. Developed by the [sFlow.org Consortium](#), it is supported by a wide range of network devices, as well as software routing and network solutions.

i Flow collector use is mandatory for Multiple Routing Domain networks since SPAN does not carry attributes to distinguish traffic between different providers.

2.3.1.1 Flow agents

Multi-router networks usually simultaneously carry traffic to a prefix over multiple providers. Flow collector needs to know the exact details of such a configuration in order to correctly determine the overall provider volume and active flows. Each provider configured in IRP can be explicitly set to match Flow statistics to specific Flow agents and help IRP Flow collector assign accurate statistics for each provider.

Flow agents have been added in order to support [Optimization for multiple Routing Domains](#) but when available they are used to enhance IRP capabilities in other areas too. For example a correct set of Flow agents for all providers enables IRP to accurately determine a prefix's current route. IRP components, especially Core during decision making can spot the latest prefix statistics matching a given Flow agent(s) and infer amount of traffic is sent over individual Providers. Collected data is later used to make Performance and Bandwidth decisions with knowing about multiple best routes.

i In case Flow agent data is missing IRP relies on past probing data to determine the current route of a prefix.

Flow agents are specified in the form of:

```
IPv4/interfaceID
```

where IPv4 is the source IP address of the packets coming from the agent and interfaceID is a numerical identifier of the interface in the range 1- 4294967295. The interface ID is usually its SNMP Interface ID on the router.

A collection of such values is assigned when multiple physical interfaces are used. For example:

```
peer.X.flow_agents = 8.8.8.8/1 8.8.8.8/2 8.8.8.8/3 8.8.8.8/4
```

The value is set via parameter `peer.X.flow_agents` or under Providers and Peers configuration in Frontend. The Frontend will also retrieve and suggest a list of available values that can be matched with the provider.

2.3.1.2 Configuration

To use the `irpflowd` collector, the following steps must be completed:

1. NetFlow/sFlow/jFlow must be configured on the router(s), which must send traffic flow information to the main IRP server IP (Figure 2.3.1). See (2.3.1.3) for specific network device configuration instructions

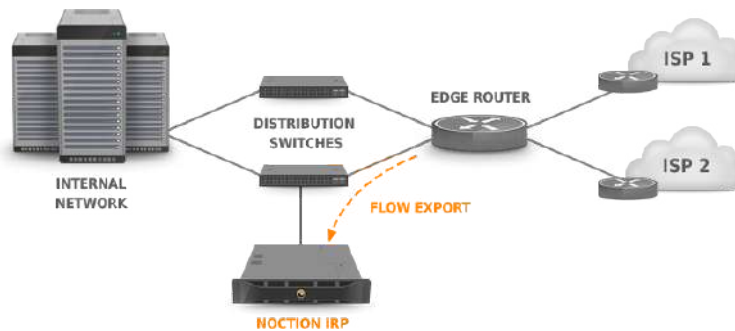


Figure 2.3.1: Flow export configuration

2. `irpflowd` must be enabled (by setting the `collector.flow.enabled` parameter):

```
collector.flow.enabled = 1
```

3. A list of all the networks, advertised by the edge routers that IRP will optimize, should be added to the configuration. This information should be specified in the `collector.ournets` parameter.
4. For security reasons, the list of valid Flow sending IP addresses must be configured in the `collector.flow.sources`, to protect `irpflowd` from unauthorized devices sending Flow data.

Example:

```
collector.flow.sources = 10.0.0.0/29
```

5. In case the Flow exporters are configured to use non-standard port numbers (2055 for NetFlow/jFlow and 6343 for sFlow), then `collector.flow.listen.nf` and `collector.flow.listen.sf` must be adjusted accordingly:

```
collector.flow.listen.nf = 2055
collector.flow.listen.sf = 6343
```

2.3.1.3 Vendor-specific NetFlow configuration examples

The following sections contain vendor-specific configuration examples, along with possible pitfalls and version-specific issues

NetFlow configuration on Cisco 7600/6500 series routers

 Refer also to the [Cisco NetFlow Software Configuration Guide](#)

Listing 2.1: Global MLS settings configuration

```
(config)# mls netflow
(config)# mls flow ip interface-full
(config)# mls flow ipv6 interface-full
(config)# mls sampling packet-based 512 8192
(config)# mls nde sender version 7
```

i Please replace the IP address and port with the actual IRP host IP and the collector UDP port (2055 by default)

Listing 2.2: Global NetFlow settings and export configuration

```
(config)# ip flow-cache entries 524288
(config)# ip flow-cache timeout inactive 60
(config)# ip flow-cache timeout active 1
(config)# ip flow-export version 9
(config)# ip flow-export destination 10.11.12.14 2055
```

i Ingress flow collection must be enabled on all interfaces facing the internal network. MLS NetFlow sampling must be enabled to preserve router resources.

Listing 2.3: Per-interface NetFlow settings configuration

```
(config)# int GigabitEthernet 3/6
(config-if)# mls netflow sampling
(config-if)# ip flow ingress
```

Flexible NetFlow configuration on Cisco 6500 series routers IOS 15.0SY series

Listing 2.4: Flexible NetFlow monitor configuration

```
(config)# flow monitor IRP-FLOW-MONITOR
(config-flow-monitor)# record platform-original ipv4 full
(config-flow-monitor)# exporter IRP-FLOW-EXPORTER
(config-flow-monitor)# cache timeout inactive 60
(config-flow-monitor)# cache timeout active 60
(config-flow-monitor)# cache entries 1048576
```

Listing 2.5: Flexible NetFlow exporter configuration

```
(config)# flow exporter IRP-FLOW-EXPORTER
(config-flow-exporter)# destination 10.11.12.14
(config-flow-exporter)# source Loopback0
(config-flow-exporter)# transport udp 2055
(config-flow-exporter)# template data timeout 120
```

i Please replace the IP address and port with the actual IRP host IP and the collector UDP port (2055 by default). Also replace the source interface with the actual one.

Listing 2.6: Flexible NetFlow sampler configuration

```
(config)# sampler flow-sampler
(config-sampler)# mode random 1 out-of 1024
```

Listing 2.7: Per-interface Flexible NetFlow settings configuration

```
(config)# interface FastEthernet0/0
(config-if)# ip flow monitor IRP-FLOW-MONITOR sampler flow-sampler input
(config-if)# ip flow monitor IRP-FLOW-MONITOR sampler flow-sampler output
```

NetFlow configuration on Cisco 7200/3600 series routers

i Please replace the IP address and port with the actual IRP host IP and the collector UDP port (2055 by default)

w Do not attempt to configure 7600/6500 series routers according to 7200/3600 router's configuration guide.

Listing 2.8: NetFlow configuration on Cisco 7200/3600 series routers

```
Router(config)# ip flow-cache entries 524288
Router(config)# ip flow-cache timeout inactive 60
Router(config)# ip flow-cache timeout active 1
Router(config)# ip flow-export version 9
Router(config)# ip flow-export destination 10.11.12.14 2055
```

Ingress/egress flow export configuration on peering interfaces

i According to Cisco IOS NetFlow Command Reference regarding the "ip flow" command history in IOS releases, this feature was introduced in IOS 12.3(11)T, 12.2(31)SB2, 12.2(18)SXE, 12.2(33)SRA.

NetFlow exporting must be configured on each peering interface which is used to send and/or receive traffic:

Listing 2.9: Ingress/egress flow export configuration on peering interfaces

```
Router(config)# interface FastEthernet 1/0
Router(config-if)# ip flow ingress
Router(config-if)# ip flow egress
```

Ingress flow export configuration (earlier IOS releases)

i Ingress flow export must be enabled on all interfaces facing the internal network. Prior to IOS 12.3(11)T, 12.2(31)SB2, 12.2(18)SXE, 12.2(33)SRA, flow export can be enabled only for ingress traffic, therefore it must be enabled on each interface that transmits/receives traffic from/to networks that must be improved

Listing 2.10: Ingress flow export configuration

```
Router(config)# interface FastEthernet 1/0
Router(config-if)# ip route-cache flow
```

NetFlow/sFlow configuration examples for Vyatta routers (VC 6.3)

w Do not configure both NetFlow and sFlow export for the same router interface. It will lead to traffic statistics distortion.

i Sampling interval must be set to at least 2000 for 10G links and 1000 for 1G links in order to save resources.

Listing 2.11: Configuring NetFlow export on Vyatta


```
vyatta@vyatta# set system flow-accounting netflow server 10.11.12.14 port
2055
vyatta@vyatta# set system flow-accounting netflow version 5
```

Listing 2.12: Configuration of an interface for the flow accounting

```
vyatta@vyatta# set system flow-accounting interface eth0
vyatta@vyatta# commit
```

jFlow export configuration for Juniper routers

Routing Engine-based sampling supports up to eight flow servers for both version 5 and version 8 configurations. The total number of collectors is limited to eight, regardless of how many are configured for version 5 or version 8. During the sampling configuration, the export packets are replicated to all the collectors, configured to receive them. If two collectors are configured to receive version 5 records, then both collectors will receive records for a specified flow.

 Default export interval for active/inactive flows on some Juniper routers is 1800 seconds. IRP requires significantly more frequent updates. The export interval recommended by IRP is 60 seconds. Refer your JunOS documentation on how to set export intervals. These parameters are named `flow-active-timeout` and `flow-inactive-timeout`.

Listing 2.13: Juniper flow export configuration

```
forwarding-options {
  sampling {
    input {
      family inet {
        rate 1000;
      }
    }
  }
  family inet {
    output {
      flow-server 10.10.3.2 {
        port 2055;
        version 5;
        source-address 10.255.255.1;
      }
    }
  }
}
```

NetFlow export must be configured on all the interfaces facing the providers. In some cases, it may also be necessary to enable NetFlow forwarding on the interfaces facing the internal network.

Listing 2.14: Per-interface NetFlow sampling configuration

```
interfaces {
  xe-0/0/0 {
    unit 0 {
      family inet {
        sampling {
          input output;
        }
      }
    }
  }
}
```

2.3.2 Irpspand Configuration

Irpspand acts like a passive network sniffer, analyzing the traffic that is provided to one or more dedicated network interfaces on the IRP server (defined in `collector.span.interfaces`) from a mirrored

port on your router or switch. Irspsand looks up the IP header in the mirrored traffic. When the link level traffic is VLAN tagged as for example in IEEE 802.1Q or 802.1ad for Q-in-Q datagrams, Irspsand will advance its IP packet sniffer past VLAN tags. For better results (higher performance and more analyzed traffic), as specified in the [IRP Technical Requirements](#) section, we recommend using Myricom 10Gbps NICs, with Sniffer10G license enabled.

⚠ Enable `collector.span.size_from_ip_header` configuration parameter if packets are stripped before forwarding to IRP's SPAN port.

To use the `irspand` collector, the following steps must be completed:

1. Configure port mirroring on your router or switch, as shown in figures (2.3.2) and (2.3.3).

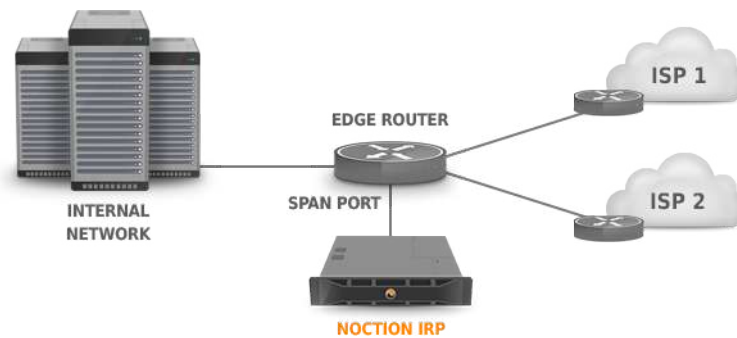


Figure 2.3.2: Span port configuration (on the router)

or:

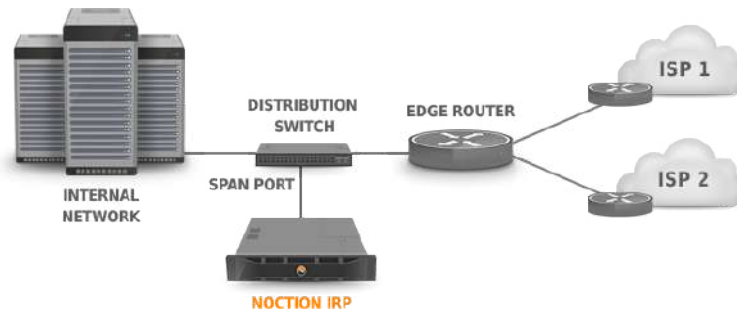


Figure 2.3.3: Span port configuration (on the switch)

2. Enable the span collector by setting the `collector.span.enabled` parameter in the configuration file:

```
collector.span.enabled = 1
```

3. Define the list of network interfaces that receive mirrored traffic, by setting the `collector.span.interfaces` parameter (multiple interfaces separated by space can be specified):

```
collector.span.interfaces = eth1 eth2 eth3
```

4. A list of all the networks advertised by the edge routers that IRP will optimize must be added to the configuration. This information should be specified in the `collector.ournets` parameter.

5. In case blackouts, congestions and excessive delays are to be analyzed by the system, the `collector.span.min_delay` must be turned on as well

```
collector.span.min_delay = 1
```

2.4 Explorer Configuration

As described in the [IRP Components](#) section, the explorer is the actual service that performs active remote network probing and tracing.

Explorer runs active and passive probings through all the available providers in order to determine the best one for the particular prefix. Such metrics as packet loss, latency, link capacity and link load are taken into consideration. In order to run a probe through a particular provider, Explorer needs the following to be configured:

1. An additional IP alias for each provider should be assigned and configured on the IRP server. This IP will be used as a source address during the probing process.

i It is recommended to configure reverse DNS records for each IP using the following template:
 performance-check-via-<PROVIDER-NAME> .
 HARMLESS-NOCTION-IRP-PROBING . <YOUR-DOMAIN-NAME> .

2. Policy-based routing (PBR) has to be configured on the edge router(s), so that traffic originating from each of these probing IP addresses will exit the network via specific provider. See specific PBR configuration in the [Specific PBR configuration scenarios](#) section.

i If network has Flowspec capabilities then alternatively Flowspec policies can be used instead of PBR. Refer for example [Flowspec policies](#), [global.flowspec.pbr](#).

1. Policy-based routing has to be configured to drop packets rather than routing them through the default route in case that the corresponding Next-Hop does not exist in the routing table.

2.4.1 Specific PBR configuration scenarios

PBRs can be setup in multiple ways, depending on the existing network infrastructure.

We will assume the following IP addresses/networks are used:

10.0.0.0/24 - used on the IRP server as well as the probing VLANs

10.0.0.2/32 - main IRP server IP address

10.0.0.3-10.0.0.5 - probing IP addresses

10.0.0.250-10.0.0.254 - router-side IP addresses for the probing VLANs

10.0.1.0/24 - used for GRE tunnel interfaces, if needed

10.10.0.0/24 - real edge routers IP addresses

10.11.0.0/30 - BGP session with the 1st provider, **10.11.0.1** being the ISP BGP neighbor IP

10.12.0.0/30 - BGP session with the 2nd provider, **10.12.0.1** being the ISP BGP neighbor IP

10.13.0.0/30 - BGP session with the 3rd provider, **10.13.0.1** being the ISP BGP neighbor IP

Vlan 3 - the probing Vlan

eth0 - the probing network interface on the IRP server

w In a production network, please change the IP addresses used in these examples to the actual addresses assigned and used in the network. Same goes for the Vlan interface.

i Brocade routers use Cisco compliant configuration commands. Unless otherwise noted, the Cisco configuration examples will also work on Brocade devices.

Case 1: Single router, two providers, and separate probing Vlan.

i 10.0.0.1/32 is configured on the edge router, on the probing vlan interface (ve3).

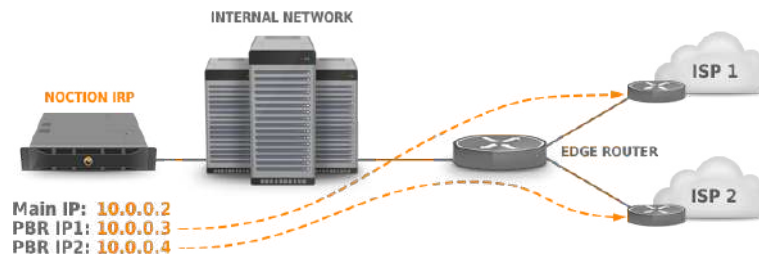


Figure 2.4.1: PBR configuration: single router and separate probing Vlan

In this case, a single PBR rule should be enforced on the Edge router for each of the probing IP addresses.

Listing 2.15: Cisco IPv4 PBR configuration: 1 router, 2 providers

```
access-list 1 permit ip host 10.0.0.3
access-list 2 permit ip host 10.0.0.4
!
route-map irp-peer permit 10
 match ip address 1
  set ip next-hop 10.11.0.1
  set interface Null0
!
route-map irp-peer permit 20
 match ip address 2
  set ip next-hop 10.12.0.1
  set interface Null0
!
interface ve 3
ip policy route-map irp-peer
```

- Route-map entries with the destination set to Null0 interface are used for preventing packets to flow through the default route in the case that the corresponding next-hop does not exist in the routing table (typically when a physical interface administrative/operational status is down).

Cisco ASR9000 routers running IOS XR use a different PBR syntax.

Listing 2.16: Cisco IPv4 PBR configuration for IOS XR: 1 router, 2 providers

```
configure
ipv4 access-list irp-peer
 10 permit ipv4 host 10.0.0.3 any nexthop1 ipv4 10.11.0.1 nexthop2 ipv4
 169.254.0.254
 11 permit ipv4 host 10.0.0.4 any nexthop1 ipv4 10.12.0.1 nexthop2 ipv4
 169.254.0.254
end
```

```

router static
  address-family ipv4 unicast
  169.254.0.254 Null0
end

interface FastEthernet1/1
  ipv4 access-group irp-peer ingress
end

```

For Juniper routers, a more complex configuration is required:

Listing 2.17: Juniper IPv4 PBR configuration: 1 router, 2 providers

```

[edit interfaces]
xe-0/0/0 {
  unit 3 {
    family inet {
      filter {
        input IRP-policy;
      }
    }
  }
}
[edit firewall]
family inet {
  filter IRP-policy {
    term irp-peer1 {
      from {
        source-address 10.0.0.3/32;
      }
      then {
        routing-instance irp-isp1-route;
      }
    }
    term irp-peer2 {
      from {
        source-address 10.0.0.4/32;
      }
      then {
        routing-instance irp-isp2-route;
      }
    }
    term default {
      then {
        accept;
      }
    }
  }
}
[edit]
routing-instances {
  irp-isp1-route {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 10.11.0.1;
      }
    }
  }
}

```

```

    irp-isp2-route {
        instance-type forwarding;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.12.0.1;
            }
        }
    }
}
routing-options {
    interface-routes {
        rib-group inet irp-policies;
    }
    rib-groups {
        irp-policies {
            import-rib [ inet.0 irp-isp1-route.inet.0 irp-isp2-route.inet.0 ];
        }
    }
}

```

PBR configuration on Vyatta routers.

Unfortunately, prior to and including VC6.4, Vyatta does not natively support policy-based routing. Thus, the PBR rules should be configured using the standard Linux `ip` toolset. To make these rules persistent, they should be also added to `/etc/rc.local` on the Vyatta system.

Listing 2.18: Vyatta IPv4 PBR configuration example

```

ip route add default via 10.11.0.1 table 101
ip route add default via 10.12.0.1 table 102
ip rule add from 10.0.0.3 table 101 pref 32001
ip rule add from 10.0.0.4 table 102 pref 32002

```

Vyatta versions VC6.5 and up, natively support source-based routing. The following example can be used:

Listing 2.19: Vyatta (VC6.5 and up) IPv4 PBR configuration example

```

# Setup the routing policy:
set policy route IRP-ROUTE
set policy route IRP-ROUTE rule 10 destination address 0.0.0.0/0
set policy route IRP-ROUTE rule 10 source address 10.0.0.3/32
set policy route IRP-ROUTE rule 10 set table 103
set policy route IRP-ROUTE rule 20 destination address 0.0.0.0/0
set policy route IRP-ROUTE rule 20 source address 10.0.0.4/32
set policy route IRP-ROUTE rule 20 set table 104
set policy route IRP-ROUTE rule 30 destination address 0.0.0.0/0
set policy route IRP-ROUTE rule 30 source address 0.0.0.0/0
set policy route IRP-ROUTE rule 30 set table main
commit

# Create static route tables:
set protocols static table 103 route 0.0.0.0/0 nexthop 10.11.0.1
set protocols static table 104 route 0.0.0.0/0 nexthop 10.12.0.1
commit

# Assign policies to specific interfaces, Vlan 3 on eth1 in this example:
set interfaces ethernet eth1.3 policy route IRP-ROUTE

# Verify the configuration:
show policy route IRP-ROUTE

```

```
show protocols static
show interfaces ethernet eth1.3
```

Case 2: Two edge routers, two providers, and a separate probing Vlan.

i Following IP addresses are configured on the routers:

- 10.0.0.251 is configured on **R1, VE3**
- 10.0.0.252 is configured on **R2, VE3**

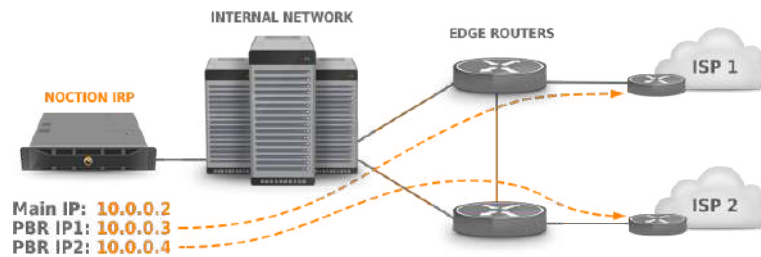


Figure 2.4.2: PBR configuration: two routers and separate probing Vlan

To reduce the number of PBR rules to one per each router, additional source based routing rules must be configured on the IRP server. This can be done either by using the `ip` command, or by adjusting the standardized `/etc/sysconfig/network-scripts/route-eth*` and `/etc/sysconfig/network-scripts/rule-eth*` configuration files. Both ways will be presented.

Listing 2.20: IRP server source-based routing using the ‘ip’ command

```
ip route add default via 10.0.0.251 table 201
ip route add default via 10.0.0.252 table 202
ip rule add from 10.0.0.3 table 201 pref 32101
ip rule add from 10.0.0.4 table 202 pref 32102
```

Listing 2.21: IRP server source-based routing using standard CentOS configuration files

```
#/etc/sysconfig/network-scripts/route-eth0:
default via 10.0.0.251 table 201
default via 10.0.0.252 table 202
#/etc/sysconfig/network-scripts/rule-eth0:
from 10.0.0.3 table 201 pref 32101
from 10.0.0.4 table 202 pref 32102
```

i Refer to OS configuration manual for configuration guidelines.

Router configuration looks similar to the previous case. A Cisco/Brocade example will be provided.

! Some Brocade routers/switches have PBR configuration limitations. Please refer to the “Policy-Based Routing” → “Configuration considerations” section in the Brocade documentation for your router/switch model.

For example, BigIron RX Series of switches do not support more than 6 instances of a route map, more than 6 ACLs in a matching policy of each route map instance, and more than 6 next hops in a set policy of each route map instance.

On the other hand, some Brocade CER/CES routers/switches have these limits raised up to 200 instances (depending on package version).

Listing 2.22: Cisco IPv4 PBR configuration: 2 routers, 2 providers, separate Vlan

```

#Router R1
access-list 1 permit ip host 10.0.0.3
!
route-map irp-peer permit 10
match ip address 1
set ip next-hop 10.11.0.1
set interface Null0
!
interface ve 3
ip policy route-map irp-peer

#Router R2
access-list 1 permit ip host 10.0.0.4
!
route-map irp-peer permit 10
match ip address 1
set ip next-hop 10.12.0.1
set interface Null0
!
interface ve 3
ip policy route-map irp-peer

```

Case 3: Complex network infrastructure, multiple routers, no probing VLAN

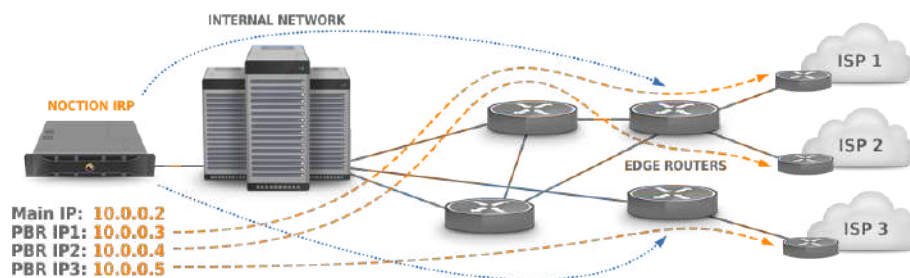


Figure 2.4.3: PBR configuration via GRE tunnels

In specific complex scenarios, traffic from the IRP server should pass multiple routers before getting to the provider. If a separate probing Vlan cannot be configured across all routers, GRE tunnels from IRP to the Edge routers should be configured.

i It is also possible to configure the PBRs without GRE tunnels, by setting PBR rules on each transit router on the IRP↔ provider path.

w Brocade routers do not support PBR set up on GRE tunnel interfaces. In this case the workaround is to configure PBR on each transit interface towards the exit edge router(s) interface(s).

GRE Tunnels configuration

Listing 2.23: IRP-side IPv4 GRE tunnel CLI configuration

```

modprobe ip_gre
ip tunnel add tun0 mode gre remote 10.10.0.1 local 10.0.0.2 ttl 64 dev eth0
ip addr add dev tun0 10.0.1.2/32 peer 10.0.1.1/32
ip link set dev tun0 up

```

Listing 2.24: IRP-side IPv4 GRE tunnel configuration using standard CentOS configs

```

#/etc/sysconfig/network-scripts/ifcfg-tun0
DEVICE=tun0
TYPE=GRE
ONBOOT=yes
MY_INNER_IPADDR=10.0.1.2
MY_OUTER_IPADDR=10.0.0.2
PEER_INNER_IPADDR=10.0.1.1
PEER_OUTER_IPADDR=10.10.0.1
TTL=64

```

i Refer to OS configuration manual for configuration guidelines.

Listing 2.25: Router-side IPv4 GRE tunnel configuration (Vyatta equipment)

```

set interfaces tunnel tun0
set interfaces tunnel tun0 address 10.0.1.1/30
set interfaces tunnel tun0 description "IRP_Tunnel_1"
set interfaces tunnel tun0 encapsulation gre
set interfaces tunnel tun0 local-ip 10.10.0.1
set interfaces tunnel tun0 remote-ip 10.0.0.2

```

Listing 2.26: Router-side IPv4 GRE tunnel configuration (Cisco equipment)

```

interface Tunnel0 routers
ip address 10.0.1.1 255.255.255.252
tunnel mode gre ip
tunnel source Loopback1
tunnel destination 10.0.0.2

```

Listing 2.27: Router-side IPv4 GRE tunnel configuration (Juniper equipment)

```

interfaces {
  gr-0/0/0 {
    unit 0 {
      tunnel {
        source 10.0.0.2;
        destination 10.10.0.1;
      }
      family inet {
        address 10.0.1.1/32;
      }
    }
  }
}

```

The above configuration is for the first edge router (R1). One more GRE tunnel should be configured on the 2nd router (R2).

As soon as the GRE tunnels are configured, the source-based routing and the PBR rules should be configured, similar to the previous section.

Listing 2.28: IRP server IPv4 source-based routing via GRE using the 'ip' command

```

ip route add default dev tun0 table 201
ip route add default dev tun1 table 202
ip route add default dev tun2 table 203
ip rule add from 10.0.1.2 table 201 pref 32101

```

```
ip rule add from 10.0.1.6 table 202 pref 32102
ip rule add from 10.0.1.10 table 202 pref 32103
```

Listing 2.29: IRP server IPv4 source-based routing via GRE using standard CentOS configuration files

```
#/etc/sysconfig/network-scripts/route-tun0:
default dev tun0 table 201
default dev tun1 table 202
default dev tun2 table 203
#/etc/sysconfig/network-scripts/rule-tun0:
from 10.0.1.2 table 201 pref 32101
from 10.0.1.6 table 202 pref 32102
from 10.0.1.10 table 203 pref 32103
```

i Refer to OS configuration manual for configuration guidelines.

Router configuration looks similar to the previous cases. A Cisco/Brocade example will be provided.

Listing 2.30: Cisco IPv4 PBR configuration: 2 routers, 2 providers, separate Vlan

```
#Router R1
access-list 1 permit ip host 10.0.1.2
access-list 2 permit ip host 10.0.1.6
!
route-map irp-peer permit 10
match ip address 1
set ip next-hop 10.11.0.1
set interface Null0
!
route-map irp-peer permit 20
match ip address 2
set ip next-hop 10.12.0.1
set interface Null0
!

interface Tunnel0
ip policy route-map irp-peer
interface Tunnell
ip policy route-map irp-peer

#Router R2
access-list 1 permit ip host 10.0.1.10
!
route-map irp-peer permit 10
match ip address 1
set ip next-hop 10.13.0.1
set interface Null0
!
interface Tunnel0
ip policy route-map irp-peer
```

Case 4: Internet Exchanges configuration examples

The PBR rules for Cisco routers/switches are generated by IRP, following the template below.

Listing 2.31: Cisco IPv4 PBR configuration template

```
!--- repeated block for each peering partner
no route-map <ROUTEMAP> permit <ACL>
```

```

no ip access-list extended <ROUITEMAP>--<ACL>

ip access-list extended <ROUITEMAP>--<ACL>
  permit ip host <PROBING_IP> any dscp <PROBING_DSCP>

route-map <ROUITEMAP> permit <ACL>
  match ip address <ROUITEMAP>--<ACL>
  set ip next-hop <NEXT_HOP>
  set interface Null0

!--- block at the end of PBR file
interface <INTERFACE>
  ip policy route-map <ROUITEMAP>

```

The “<>” elements represent variables with the following meaning:

- <ROUITEMAP> represents the name assigned by IRP and equals the value of the Route Map parameter in PBR Generator ("irp-ix" in Figure 4)
- <ACL> represents a counter that identifies individual ACL rules. This variable's initial value is taken from ACL name start field of PBR Generator and is subsequently incremented for each ACL
- <PROBING_IP> one of the configured probing IPs that IRP uses to probe link characteristics via different peering partners. One probing IP is sufficient to cover up to 64 peering partners
- <PROBING_DSCP> an incremented DSCP value assigned by IRP for probing a specific peering partner. This is used in combination with the probing IP
- <NEXT_HOP> represents the IP address identifying the peering partner on the exchange. This parameter is retrieved during autoconfiguration and preserved in Exchange configuration
- <INTERFACE> represents the interface where traffic conforming to the rule will exist the Exchange router. This is populated with the Interface value of PBR Generator

The PBR rules for Brocade non-XMR router/switches are generated by IRP, following the template below.

Listing 2.32: Cisco IPv4 PBR configuration template

```

!--- repeated block for each peering partner
no route-map <ROUITEMAP> permit <ACL>
no ip access-list extended <ROUITEMAP>--<ACL>


ip access-list extended <ROUITEMAP>--<ACL>
  permit ip host <PROBING_IP> any dscp-matching <PROBING_DSCP>

route-map <ROUITEMAP> permit <ACL>
  match ip address <ROUITEMAP>--<ACL>
  set ip next-hop <NEXT_HOP>
  set interface Null0

!--- block at the end of PBR file
interface <INTERFACE>
  ip policy route-map <ROUITEMAP>

```

The “<>” elements represent variables with the same meaning as per Cisco example.

 Brocade XMR routers use keyword “dscp-mapping” instead of “dscp-matching”.

The PBR rules for Juniper routers/switches are generated by IRP, following the template below.

i It is important to note that the different sections of the PBR rules (load replace/merge relative terminal) should be entered independently and not as a single file that is output by IRP. Also take note that the last group includes a 'load merge' combo and not a 'load replace' as the first three groups.

Listing 2.33: Juniper IPv4 PBR configuration template

```
load replace relative terminal
[Type ^D at a new line to end input]

interfaces {
  <INTERFACE> {
    unit <INTERFACE_UNIT> {
      family inet {
        filter {
          replace:
            input <ROUТЕMAP>;
        }
      }
    }
  }
}

load replace relative terminal
[Type ^D at a new line to end input]

firewall {
  family inet {
    filter <ROUТЕMAP> {
      replace:
        term <ROUТЕMAP><ACL> {
          from {
            source-address <PROBING_IP>;
            dscp <PROBING_DSCP>;
          }
          then {
            routing-instance <ROUТЕMAP><ACL>-route;
          }
        }
      ...
    }
    replace:
      term default {
        then {
          accept;
        }
      }
  }
}

load replace relative terminal
[Type ^D at a new line to end input]

routing-instances {
  replace:
```

```

<ROUТЕMAP><ACL>-route {
    instance-type forwarding;
    routing-options {
        static {
            route 0.0.0.0/0 next-hop <NEXT_HOP>;
        }
    }
}
...
}

load merge relative terminal
[Type ^D at a new line to end input]

routing-options {
    interface-routes {
        replace:
        rib-group inet <ROUТЕMAP>rib;
    }
    rib-groups {
        replace:
        <ROUТЕMAP>rib {
            import-rib [ inet.0 <ROUТЕMAP><ACL>-route.inet.0 ... ];
        }
    }
}

```

The “<>” elements represent variables with the following meaning:

- <INTERFACE> represents the interface where traffic conforming to the rule will exist the Exchange router. This is populated with the Interface value of PBR Generator
- <INTERFACE_UNIT> is the value of the Interface Unit parameter in PBR Generator
- <ROUТЕMAP> represents the name assigned by IRP and equals the value of the Route Map parameter in PBR Generator
- <ACL> represents a combined counter like "00009" that identifies individual ACL rules. This variable's initial value is taken from ACL name start field of PBR Generator and is subsequently incremented for each ACL
- <PROBING_IP> one of the configured probing IPs that IRP uses to probe link characteristics via different peering partners. One probing IP is sufficient to cover up to 64 peering partners
- <PROBING_DSCP> an incremented DSCP value assigned by IRP for probing a specific peering partner. This is used in combination with the probing IP
- <NEXT_HOP> represents the IP address identifying the peering partner on the exchange. This parameter is retrieved during autoconfiguration and preserved in Exchange configuration

i Note that Juniper routers/switches need an additional parameter in order to correctly configure PBRs - Interface unit.

Verifying PBR configuration

To ensure that the PBRs are properly configured on the router(s), the standard *NIX traceroute command can be used, by specifying each probing IP as the source IP address for tracing the route. The route should pass through a different provider (note the ISP BGP neighbor IP).

Listing 2.34: PBR validation using ‘traceroute’

```

root@server ~ $ traceroute -m 5 8.8.8.8 -nns 10.0.0.3
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 10.0.0.1 0.696 ms  0.716 ms  0.783 ms
 2 10.11.0.1  0.689 ms  0.695 ms  0.714 ms
 3 84.116.132.146 14.384 ms 13.882 ms 13.891 ms
 4 72.14.219.9 13.926 ms 14.477 ms 14.473 ms
 5 209.85.240.64 14.397 ms 13.989 ms 14.462 ms

root@server ~ $ traceroute -m 5 8.8.8.8 -nns 10.0.0.4
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 10.0.0.1 0.696 ms  0.516 ms  0.723 ms
 2 10.12.0.1  0.619 ms  0.625 ms  0.864 ms
 3 83.16.126.26 13.324 ms 13.812 ms 13.983 ms
 4 72.14.219.9 15.262 ms 15.347 ms 15.431 ms
 5 209.85.240.64 16.371 ms 16.991 ms 16.162 ms

```

Another useful method for checking that the PBRs are properly configured is to use Explorer self check option. Make sure that the providers are added to IRP configuration before executing Explorer self check.

Listing 2.35: PBR validation using Explorer self check

```

root@server ~ $ /usr/sbin/explorer -s
Starting PBR check

PBR check failed for provider A[2]. Diagnostic hop information: IP
=10.11.0.12 TTL=3
PBR check succeeded for provider B[3]. Diagnostic hop information: IP
=10.12.0.1 TTL=3

```

In order to ensure that the PBRs are properly configured for Internet Exchanges, the method below can be used.

Listing 2.36: PBR validation using ‘iptables’ and ‘traceroute’

```

root@server ~ $ iptables -t mangle -I OUTPUT -d 8.8.8.8 -j DSCP --set-dscp
<PROBING_DSCP>


root@server ~ $ traceroute -m 5 -nns <PROBING_IP> 8.8.8.8
traceroute to 8.8.8.8, 30 hops max, 60 byte packets
 1 ...
 2 ...
 3 <NEXT_HOP> 126.475 ms !X^C

```

where

- <NEXT_HOP> is a Peering Partner’s next-hop IP address in IRP configuration
- <PROBING_DSCP> is a Peering Partner’s DSCP value in IRP configuration
- <PROBING_IP> is a Peering Partner’s probing IP address in IRP configuration

The first IP of the trace leaving client infrastructure should be on the Exchange and the next-hop should belong to the correct Peering Partner.

 iptables rules should be deleted after all tests are done.

2.4.1.1 Current route detection

Current route detection algorithm requires the `explorer.infra_ips` parameter to be configured. All the IP addresses and subnets belonging to the infrastructure should be added here. These IP addresses can be easily determined by using the `traceroute` command.

2.4.1.2 Providers configuration

Before Explorer starts probing the prefixes detected by the Collector, all providers should be configured.

i For the complete list of provider settings please see the [Providers settings](#) section. Before configuring providers in IRP, the BGP sessions need to be defined, see [Bgp Configuration](#)

- Please note that some Brocade router models do not support SNMP counters per Vlan, therefore physical interfaces should be used for gathering traffic statistics.

For the sample configuration below, let's assume the following:

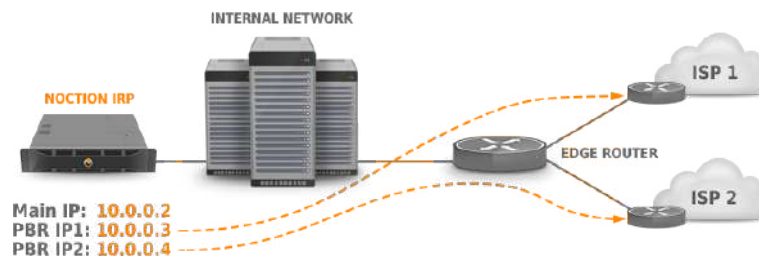


Figure 2.4.4: PBR configuration: single router and separate probing Vlan

ISP1 - the provider's name

10.0.0.1 - Router IP configured on the probing Vlan

10.0.0.3 - Probing IP for ISP1, configured on the IRP server

10.11.0.1, 10.11.0.2 - IP addresses used for the EBGP session with the ISP, 10.11.0.2 being configured on the router

400Mbps - the agreed bandwidth

1Gbps - the physical interface throughput

'public' - read-only SNMP community configured on R1

GigabitEthernet2/1 - the physical interface that connects R1 to ISP1

As presented in Listing 2.37, all the parameters are pretty self-explanatory. Please check the [BGP Monitoring](#) section as well.

Listing 2.37: Sample provider configuration

```
peer.1.95th = 400
peer.1.95th.bill_day = 1
peer.1.bgp_peer = R1
peer.1.cost = 6
peer.1.description = ISP1
peer.1.ipv4.next_hop = 10.11.0.1
peer.1.ipv4.probing = 10.0.0.3
```

```
peer.1.ipv4.diag_hop = 10.11.0.1
peer.1.ipv4.mon = 10.11.0.1 10.11.0.2
peer.1.limit_load = 1000
peer.1.shortname = ISP1
peer.1.snmp.interfaces = 1:GigabitEthernet2/1
peer.1.mon.ipv4.bgp_peer = 10.11.0.1

snmp.1.name = Host1
snmp.1.ip = 10.0.0.1
snmp.1.community = public
```

SNMP parameters validation

To make sure that the SNMP parameters are correct, the ‘snmpwalk’ tool can be run on the IRP server:

Listing 2.38: SNMP parameters validation

```
root@server ~ $ snmpwalk -v2c -c irp-public 10.0.0.1 ifDescr
IF-MIB::ifDescr.1 = STRING: GigabitEthernet1/1
IF-MIB::ifDescr.2 = STRING: GigabitEthernet1/2
IF-MIB::ifDescr.3 = STRING: GigabitEthernet2/1
IF-MIB::ifDescr.4 = STRING: GigabitEthernet2/2
IF-MIB::ifDescr.5 = STRING: GigabitEthernet2/3
IF-MIB::ifDescr.6 = STRING: GigabitEthernet2/4


root@server ~ $ snmpwalk -v2c -c irp-public 10.0.0.1 ifIndex
IF-MIB::ifIndex.1 = INTEGER: 1
IF-MIB::ifIndex.2 = INTEGER: 2
IF-MIB::ifIndex.3 = INTEGER: 3
IF-MIB::ifIndex.4 = INTEGER: 4
IF-MIB::ifIndex.5 = INTEGER: 5
IF-MIB::ifIndex.6 = INTEGER: 6
```

2.4.2 Flowspec PBR

In networks that have Flowspec and Redirect to IP capabilities, PBR can be implemented by means of Flowspec policies. Refer details in [Flowspec policies](#).

In order to use assigned providers or peers on Internet Exchanges in IRP the same set of source IP addresses and DSCP values (for Internet Exchanges) are assigned to individual providers/peers. These values are used to automatically generate Flowspec policies that redirect IRP probes to designated next-hops.

In order to use this feature `global.flowspec`, `global.flowspec.pbr` and `bgpd.peer.X.flowspec` must be enabled.

 Flowspec PBR cannot be used during non-intrusive (`global.nonintrusive_bgp`) mode.

2.5 Bgpd Configuration

For IRP to inject the improved prefixes to the routing tables, proper iBGP sessions have to be configured between the edge routers and the IRP. The IRP BGP daemon acts similarly to a *route-reflector-client*. Following criteria need to be met:

1. An internal BGP session using the same autonomous system number (ASN) must be configured between each edge router and the IRP. BGP sessions must not be configured with *next-hop-self* (route reflectors can't be used to inject routes with modified *next-hop*) - the *next-hop* parameter advertised by IRP Bgpd should be distributed to other iBGP neighbors.
2. *route-reflector-client* must be enabled for the routes advertised by IRP Bgpd to be distributed to all non-client neighbors.
3. Routes advertised by IRP Bgpd must have a higher preference over routes received from external BGP neighbors.

This can be done by different means, on the IRP or on the router side:

- *Local-pref* can be set to a reasonably high value in the Bgpd configuration
- *Communities* can be appended to prefixes advertised by Bgpd

❌ Avoid collisions of localpref or communities values assigned to IRP within both its configuration and/or on customer's network.

- Multi-exit-discriminator (*MED*) can be changed to affect the best-path selection algorithm
- *Origin* of the advertised route can be left unchanged or overridden to a specific value (*incomplete, IGP, EGP*)

ℹ LocalPref, MED and Origin attribute values are set with the first nonempty value in this order: 1) value from configuration or 2) value taken from incoming aggregate or 3) default value specified in RFC4271.

Communities attribute value concatenates the value taken from incoming aggregate with configuration value. The router should be configured to send no Communities attribute in case it is required that IRP announces Communities attribute that contain only the configured value.

4. BGP *next-hop* must be configured for each provider configured in IRP (please refer to [Providers configuration](#) and [Providers settings](#))

❌ Special attention is required if edge routers are configured to announce prefixes with empty AS-Path. In some cases, improvements announced by Bgpd may have an empty AS-Path. Thus, when edge router does not have any prefix-list filtering enforced, all the current improvements can be improperly advertised to routers - this may lead to policy violations and session reset. Refer to [AS-Path behavior in IRP Bgpd](#) regarding the way to disallow announcements with empty AS-Path. None of the improvements advertised by IRP should be advertised to your external peers

⚠ We recommend the routes to be injected into the edge router which runs the BGP session with the provider. This ensures that the routes are properly redistributed across the network.

For example, there are two routers: R1 and R2. R1 runs a BGP session with Level3 and R2 with Cogent. The current route is x.x.x.x/24 with the next-hop set to Level3 and all the routers learn this route via R1. The system injects the x.x.x.x/24 route to R2 with the next-hop updated to Cogent. In this case the new route is installed on the routing table and it will be properly propagated to R1 (and other routers) via iBGP.

However if the system injects the new route to R1 instead of R2, the route's next-hop will point to R2 while R2 will have the next-hop pointing to R1 as long as the injected route is propagated over iBGP to other routers. In this case a routing loop will occur.

In the following BGP session configuration example, the IRP server with IP 10.0.0.2 establishes an iBGP session to the edge router (IP: 10.0.0.1). The *local-pref* parameter for the prefixes advertised by IRP is set to 190. BGP monitoring (see [BGP Monitoring](#), [Bgpd settings](#)) is enabled.

Listing 2.39: iBGP session configuration example

```
bgpd.peer.R1.as           = 65501
bgpd.peer.R1.our_ip       = 10.0.0.2
bgpd.peer.R1.master_peer_ip = 10.0.0.1
bgpd.peer.R1.listen      = 1
bgpd.peer.R1.localpref    = 190
bgpd.peer.R1.shutdown    = 0
```

Vendor-specific router-side iBGP session configuration examples:

Vyatta routers:

Listing 2.40: Vyatta IPv4 iBGP session configuration example

```
set protocols bgp 65501 neighbor 10.0.0.2 remote-as '65501'
set protocols bgp 65501 neighbor 10.0.0.2 route-reflector-client
set protocols bgp 65501 parameters router-id '10.0.0.1'
```

Listing 2.41: Vyatta IPv6 iBGP session configuration example

```
delete system ipv6 disable-forwarding
commit
set protocols bgp 65501 neighbor 2001:db8:2::2 remote-as '65501'
set protocols bgp 65501 neighbor 2001:db8:2::2 route-reflector-client
set protocols bgp 65501 neighbor 2001:db8:2::2 address-family 'ipv6-unicast'
set protocols bgp 65501 parameters router-id '10.0.0.1'
```

Listing 2.42: Vyatta IPv4 route-map for setting local-pref on the router

```
set protocols bgp 65501 neighbor 10.0.0.2 route-map import 'RM-IRP-IN'
set policy route-map RM-IRP-IN rule 10 action 'permit'
set policy route-map RM-IRP-IN rule 10 set local-preference '190'
```

Listing 2.43: Vyatta IPv6 route-map for setting local-pref on the router

```
set protocols bgp 65501 neighbor 2001:db8:2::2 route-map import 'RM-IRP-IN'
set policy route-map RM-IRP-IN rule 10 action 'permit'
set policy route-map RM-IRP-IN rule 10 set local-preference '190'
```

Cisco routers:

Listing 2.44: Cisco IPv4 iBGP session configuration example

```
router bgp 65501
neighbor 10.0.0.2 remote-as 65501
neighbor 10.0.0.2 send-community
neighbor 10.0.0.2 route-reflector-client
```

Listing 2.45: Cisco IPv6 iBGP session configuration example

```
router bgp 65501
neighbor 2001:db8:2::2 remote-as 65501
neighbor 2001:db8:2::2 send-community
neighbor 2001:db8:2::2 route-reflector-client
```

or

```
router bgp 65501
neighbor 2001:db8:2::2 remote-as 65501
no neighbor 2001:db8:2::2 activate
address-family ipv6
neighbor 2001:db8:2::2 activate
neighbor 2001:db8:2::2 send-community
neighbor 2001:db8:2::2 route-reflector-client
```

Listing 2.46: Cisco IPv4 route-map for setting local-pref on the router

```
router bgp 65501
neighbor 10.0.0.2 route-map RM-IRP-IN input
route-map RM-IRP-IN permit 10
  set local-preference 190
```

Listing 2.47: Cisco IPv6 route-map for setting local-pref on the router

```
router bgp 65501
neighbor 2001:db8:2::2 route-map RM-IRP-IN input
route-map RM-IRP-IN permit 10
  set local-preference 190
```


Listing 2.48: Limiting the number of received prefixes for an IPv4 neighbor on Cisco

```
router bgp 65501
neighbor 10.0.0.2 maximum-prefix 10000
```

Listing 2.49: Limiting the number of received prefixes for an IPv6 neighbor on Cisco

```
router bgp 65501
neighbor 2001:db8:2::2 maximum-prefix 10000
```

Juniper equipment:

 The Cluster ID must be unique in multi-router configuration. Otherwise improvements will not be redistributed properly. Cluster ID is optional for single router networks.

Listing 2.50: Juniper IPv4 iBGP session configuration example

```
[edit]
routing-options {
  autonomous-system 65501;
  router-id 10.0.0.1;
}
protocols {
  bgp {
    group 65501 {
      type internal;
      cluster 0.0.0.1;
```

```

        family inet {
            unicast;
        }
        peer-as 65501;
        neighbor 10.0.0.2;
    }
}

```

Listing 2.51: Juniper IPv6 iBGP session configuration example

```

[edit]
routing-options {
    autonomous-system 65501;
    router-id 10.0.0.1;
}
protocols {
    bgp {
        group 65501 {
            type internal;
            cluster 0.0.0.1;
            family inet6 {
                any;
            }
            peer-as 65501;
            neighbor 2001:db8:2::2;
        }
    }
}

```

Listing 2.52: Juniper IPv4 route-map for setting local-pref on the router

```

[edit]
routing-options {
    autonomous-system 65501;
    router-id 10.0.0.1;
}
protocols {
    bgp {
        group 65501 {
            type internal;
            peer-as 65501;
            neighbor 10.0.0.2 {
                preference 190;
            }
        }
    }
}

```

Listing 2.53: Limiting the number of received prefixes for an IPv4 neighbor on Juniper

```

protocols {
    bgp {
        group 65501 {
            neighbor 10.0.0.2 {
                family inet {
                    any {
                        prefix-limit {
                            maximum 10000;
                        }
                    }
                }
            }
        }
    }
}

```


2.5.3 BGP Additional paths

Typically router sends only the best routes and that behaviour hides alternatives.

Enabling sending of additional paths (RFC 7911) allows IRP to automatically configure complete list of routes for partial routing providers and internet exchanges peering partners.

Listing 2.55: Cisco: Configure BGP Additional Paths


```
router bgp 65500
neighbor 10.1.1.2 remote-as 65500
!
address-family ipv4
  bgp additional-paths select all
  neighbor 10.1.1.2 activate
  neighbor 10.1.1.2 additional-paths send
  neighbor 10.1.1.2 advertise additional-paths all
exit-address-family
```

Listing 2.56: Juniper: Configure BGP Additional Paths

```
[edit protocols bgp group group-name family family]
add-path {
  send {
    path-count 20;
  }
}
```

2.6 Failover configuration (Not supported in IRP Lite)

Failover relies on many operating system, networking and database components to work together. All these must be planned in advance and require careful implementation.

 We recommend that you request Noction's engineers to assist with failover configuration.


Subsequent sections should be considered in presented order when configuring a fresh IRP failover setup. Refer to them when troubleshooting or restoring services.


2.6.1 Initial failover configuration

Prerequisites

Before proceeding with failover configuration the following prerequisites must be met:

- One IRP node is configured and fully functional. We will refer to this node as \$IRPMaster.
- Second IRP node is installed with the same version of IRP as on \$IRPMaster. We will refer to this node as \$IRPSlave.

 Second IRP node **MUST** run the same operating system as \$IRPMaster.

 When troubleshooting problems, besides checking matching IRP versions ensure the same versions of irp MySQL databases are installed on both failover nodes.

- IRP services, MySQL and HTTP daemons are stopped on \$IRPSLAVE node.
- Network operator can SSH to both \$IRPMaster and \$IRPSLAVE and subsequent commands are assumed to be run from a \$IRPMaster console.

 \$IRPMaster and \$IRPSLAVE must have different hostnames.

Configure communication channel from \$IRPMaster to \$IRPSLAVE


This channel is used during failover configuration and subsequently \$IRPMaster uses it to synchronize IRP configuration changes when these occur. It uses key-based authentication without a passphrase to allow automated logins on \$IRPSLAVE by \$IRPMaster processes.

 Adjust firewalls if any so that \$IRPMaster node can access \$IRPSLAVE via SSH.

Generate SSH keys pair WITHOUT passphrase:

Listing 2.57: Generate keys on \$IRPMaster

```
root@IRPMaster ~ # ssh-keygen -t rsa -b 2048 -f ~/.ssh/id_rsa -C "
failover@noction"
```

 Default keys files are used. In case your system needs additional keys for other purposes we advise that those keys are assigned a different name. If this is not possible then keys file name designated for failover use should be also specified in IRP configuration parameter `global.failover_identity_file`.

Copy public SSH key from master to slave instance:

Listing 2.58: Install public key on \$IRPSLAVE

```
root@IRPMaster ~ # cat ~/.ssh/id_rsa.pub | while read key; do ssh $IRPSLAVE
"echo $key >> ~/.ssh/authorized_keys"; done
```


Check if \$IRPMaster can login to \$IRPSLAVE without using a password:

Listing 2.59: Check SSH certificate-based authentication works

```
root@IRPMaster ~ # ssh $IRPSLAVE
```

Install certificate and keys for MySQL Multi-Master replication between \$IRPMaster and \$IRPSLAVE

MySQL Multi-Master replication uses separate communication channels that also require authentication. IRP failover uses key-based authentication for these channels too.

 Adjust firewalls if any so that \$IRPMaster and \$IRPSLAVE can communicate with each other bidirectionally.

Create Certificate Authority and server/client certificates and keys. Commands must be run on both \$IRPMaster and \$IRPSLAVE nodes:

Listing 2.60: Generate CA and certificates

```
# cd && rm -rvf irp-certs && mkdir -p irp-certs && cd irp-certs

# openssl genrsa 2048 > $(hostname -s)-ca-key.pem

# openssl req -new -x509 -nodes -days 3600 -subj "/C=US/ST=CA/L=Palo Alto/O=
=Noction/OU=Intelligent Routing Platform/CN=$(/bin/hostname) CA/
emailAddress=support@noction.com" -key $(hostname -s)-ca-key.pem -out $(
hostname -s)-ca-cert.pem

# openssl req -newkey rsa:2048 -days 3600 -subj "/C=US/ST=CA/L=Palo Alto/O=
Noction/OU=Intelligent Routing Platform/CN=$(/bin/hostname) server/
emailAddress=support@noction.com" -nodes -keyout $(hostname -s)-server-
key.pem -out $(hostname -s)-server-req.pem

# openssl rsa -in $(hostname -s)-server-key.pem -out $(hostname -s)-server-
key.pem

# openssl x509 -req -in $(hostname -s)-server-req.pem -days 3600 -CA $(
hostname -s)-ca-cert.pem -CAkey $(hostname -s)-ca-key.pem -set_serial 01
-out $(hostname -s)-server-cert.pem

# openssl req -newkey rsa:2048 -days 3600 -subj "/C=US/ST=CA/L=Palo Alto/O=
Noction/OU=Intelligent Routing Platform/CN=$(/bin/hostname) client/
emailAddress=support@noction.com" -nodes -keyout $(hostname -s)-client-
key.pem -out $(hostname -s)-client-req.pem

# openssl rsa -in $(hostname -s)-client-key.pem -out $(hostname -s)-client-
key.pem

# openssl x509 -req -in $(hostname -s)-client-req.pem -days 3600 -CA $(
hostname -s)-ca-cert.pem -CAkey $(hostname -s)-ca-key.pem -set_serial 01
-out $(hostname -s)-client-cert.pem
```

Verify certificates. Commands must be run on both \$IRPMASTER and \$IRPSLAVE nodes:

Listing 2.61: Verify certificates

```
# openssl verify -CAfile $(hostname -s)-ca-cert.pem $(hostname -s)-server-
cert.pem $(hostname -s)-client-cert.pem

server-cert.pem: OK
client-cert.pem: OK
```

Install certificates in designated directories. Commands must be run on both \$IRPMASTER and \$IRP-SLAVE nodes:

Listing 2.62: Install certificates in designated directories

```
# mkdir -p /etc/pki/tls/certs/mysql/server/ /etc/pki/tls/certs/mysql/client/
/ /etc/pki/tls/private/mysql/server/ /etc/pki/tls/private/mysql/client/

# cp $(hostname -s)-ca-cert.pem $(hostname -s)-server-cert.pem /etc/pki/tls
/certs/mysql/server/
# cp $(hostname -s)-ca-key.pem $(hostname -s)-server-key.pem /etc/pki/tls/
private/mysql/server/
# cp $(hostname -s)-client-cert.pem /etc/pki/tls/certs/mysql/client/
# cp $(hostname -s)-client-key.pem /etc/pki/tls/private/mysql/client/
```

```
# cd && rm -rvf irp-certs
```

Cross copy client key and certificates:

Listing 2.63: Cross copy client key and certificates

```
root@IRPMASTER ~# scp "/etc/pki/tls/certs/mysql/server/$IRPMASTER-ca-cert.
pem" "$IRPSLAVE:/etc/pki/tls/certs/mysql/client/"
root@IRPMASTER ~# scp "/etc/pki/tls/certs/mysql/client/$IRPMASTER-client-
cert.pem" "$IRPSLAVE:/etc/pki/tls/certs/mysql/client/"
root@IRPMASTER ~# scp "/etc/pki/tls/private/mysql/client/$IRPMASTER-client-
key.pem" "$IRPSLAVE:/etc/pki/tls/private/mysql/client/"

root@IRPMASTER ~# scp "$IRPSLAVE:/etc/pki/tls/certs/mysql/server/$IRPSLAVE-
ca-cert.pem" "/etc/pki/tls/certs/mysql/client/"
root@IRPMASTER ~# scp "$IRPSLAVE:/etc/pki/tls/certs/mysql/client/$IRPSLAVE-
client-cert.pem" "/etc/pki/tls/certs/mysql/client/"
root@IRPMASTER ~# scp "$IRPSLAVE:/etc/pki/tls/private/mysql/client/
$IRPSLAVE-client-key.pem" "/etc/pki/tls/private/mysql/client/"
```

Adjust file permissions. Commands must be run on both \$IRPMAS^TER and \$IRPSLAVE nodes:

Listing 2.64: Set file permissions for keys and certificates

```
# chown -R mysql:mysql /etc/pki/tls/certs/mysql/ /etc/pki/tls/private/mysql
/
# chmod 0600 /etc/pki/tls/private/mysql/server/* /etc/pki/tls/private/mysql
/client/*
```

Configure MySQL replication on \$IRPSLAVE

Each node of MySQL Multi-Master replication is assigned its own identifier and references previously configured certificates. There are also configuration parameters such as binary/relay log file names and auto-increment and auto-increment-offset values.

IRP includes a template config file `/usr/share/doc/irp/irp.my_repl_slave.cnf.template`. The template designates \$IRPSLAVE as second server of the Multi-Master replication and includes references to `$(hostname -s)` that need to be replaced with the actual hostname of \$IRPSLAVE before installing. Apply the changes and review the configuration file. Alternatively a command like the example below can be used to create \$IRPSLAVE config file from template. Make sure to use the actual short hostname instead of the provided variable:

Listing 2.65: Example \$IRPSLAVE configuration from template

```
root@IRPSLAVE ~# sed 's|$(hostname -s)|$IRPSLAVE|' < /usr/share/doc/irp/irp
.my_repl_slave.cnf.template > /etc/noction/mysql/irp.my_repl_slave.cnf
```

The config file created above must be included into \$IRPSLAVE node's MySQL config `my.cnf`. It is recommended that line `"include /etc/noction/mysql/irp.my_repl_slave.cnf"` is un-commented.


Check Multi-Master configuration on \$IRPSLAVE:

Listing 2.66: Check MySQL on \$IRPSLAVE works correctly

```
root@IRPSLAVE ~# service mysqld start
root@IRPSLAVE ~# tail -f /var/log/mysqld.log
root@IRPSLAVE ~# mysql irp -e "show master status \G"
root@IRPSLAVE ~# service mysqld stop
```

Configure MySQL replication on \$IRPMMASTER

Configuring \$IRPMMASTER is done with running services.

 Configuring MySQL Multi-Master replication on \$IRPMMASTER should only be done after confirming it works on \$IRPSLAVE.

Similarly to \$IRPSLAVE above IRP comes with a template configuration file for \$IRPMMASTER - /usr/share/doc/irp/irp.my_repl_master.cnf.template. The template designates \$IRPMMASTER as first server of the Multi-Master replication and includes references to \$(hostname -s) that need to be replaced with the actual hostname of \$IRPMMASTER before installing. Apply the changes and review the resulting configuration file.

Alternatively a command like the example below can be used to create \$IRPMMASTER config file from template. Make sure to use the actual hostname instead of the provided variable:


Listing 2.67: Set \$IRPMMASTER as a first node for Multi-Master replication

```
root@IRPMMASTER ~# sed 's|$(hostname -s)|$IRPMMASTER|' < /usr/share/doc/irp/
  irp.my_repl_master.cnf.template > /etc/noction/mysql/irp.my_repl_master.
  cnf
```

Again, the config file created above must be included into \$IRPMMASTER node's MySQL config my.cnf. It is recommended that line "include /etc/noction/mysql/irp.my_repl_master.cnf" is un-commented. Check MySQL Multi-Master configuration on \$IRPMMASTER:


Listing 2.68: Check MySQL on \$IRPMMASTER works correctly

```
root@IRPMMASTER ~# service mysqld restart
root@IRPMMASTER ~# tail -f /var/log/mysqld.log
root@IRPMMASTER ~# mysql irp -e "show master status \G"
```

 If Multi-Master configuration on \$IRPMMASTER fails or causes unrecoverable errors, a first troubleshooting step is to comment back the included line in /etc/my.cnf on master and slave and restart mysqld service to revert to previous good configuration.

Create replication grants on \$IRPMMASTER

Replication requires a MySQL user with corresponding access rights to replicated databases. The user must be assigned a password and is designated as connecting correspondingly from \$IRPMMASTER or \$IRPSLAVE. Unfortunately hostnames cannot be used for this and the exact IP addresses of the corresponding nodes must be specified.

 The user is created only once in our procedure since after being created the database on \$IRPMMASTER is manually transferred to \$IRPSLAVE and the user will be copied as part of this process.

Grant replication access to replication user:

Listing 2.69: Replication user and grants

```
mysql> CREATE USER 'irprepl'@'<mysql_slave1_ip_address>' IDENTIFIED BY '<
  replication_user_password>';
```

```
mysql> GRANT REPLICATION SLAVE ON *.* TO 'irprepl'@'<
mysql_masterslave1_ip_address>' REQUIRE CIPHER 'DHE-RSA-AES256-SHA';
mysql> CREATE USER 'irprepl'@'<mysql_master2_ip_address>' IDENTIFIED BY '<
replication_user_password>';
mysql> GRANT REPLICATION SLAVE ON *.* TO 'irprepl'@'<
mysql_slave2_ip_address>' REQUIRE CIPHER 'DHE-RSA-AES256-SHA';
```

Copy IRP database configuration and database to \$IRPSLAVE

IRP failover copies the irp database and corresponding configuration file from \$IRPMASTER to \$IRPSLAVE before activating second server. This avoids the need to manually verify and synchronize identifiers.

Copy root's .my.cnf config file if exists:

Listing 2.70: Copy database root user configuration file

```
root@IRPMASTER ~# scp /root/.my.cnf $IRPSLAVE:/root/
```

Copy config files:

Listing 2.71: Copy database configuration files

```
root@IRPMASTER ~# scp /etc/noction/db.global.conf /etc/noction/db.frontend.
conf $IRPSLAVE:/etc/noction/
root@IRPMASTER ~# scp /etc/noction/clickhouse/users.xml $IRPSLAVE:/etc/
noction/clickhouse/
```

Preliminary copy database files:

Listing 2.72: Copy database data files

```
root@IRPMASTER ~# rsync -av --progress --delete --delete-after --exclude="
master.info" --exclude="relay-log.info" --exclude="*-bin.*" --exclude
="*-relay.*" /var/lib/mysql/ $IRPSLAVE:/var/lib/mysql/
```

i Preliminary copy ensures that large files that take a long time to copy are synced to \$IRPSLAVE without stopping MySQL daemon on \$IRPMASTER and only a reduced number of differences will need to be synced while MySQL is stopped. This operation can be rerun one more time to reduce the duration of the downtime on \$IRPMASTER even more.


Finish copy of database files (OS with Systemd):

Listing 2.73: Copy differences of database files (OS with Systemd)

```
root@IRPMASTER ~# systemctl stop httpd24-httpd mariadb clickhouse-server #
CentOS
root@IRPMASTER ~# systemctl stop apache2 mysql clickhouse-server # Ubuntu
root@IRPMASTER ~# systemctl start irp-stop-nobgpd.target

systemctl start irp-shutdown-except-bgpd.target
systemctl start irp-shutdown.target

root@IRPMASTER ~# cd /var/lib/mysql && rm -vf ./master.info ./relay-log.
info ./*-bin.* ./*-relay.*
root@IRPMASTER ~# rsync -av --progress --delete --delete-after /var/lib/
mysql/ $IRPSLAVE:/var/lib/mysql/
```

 The procedure above tries to reduce the downtime of MySQL daemon on \$IRPMMASTER. During this time Bgpd preserves IRP Improvements. Make sure this action takes less than `bgpd.db.timeout.withdraw`.


Finish copy of database files (OS with RC-files):

Listing 2.74: Copy differences of database files (OS with RC-files)

```
root@IRPMMASTER ~# service irp stop nobgpd

root@IRPMMASTER ~# service httpd24-httpd stop
root@IRPMMASTER ~# service mysqld stop
root@IRPMMASTER ~# service clickhouse-server stop

root@IRPMMASTER ~# cd /var/lib/mysql && rm -vf ./master.info ./relay-log.
info /*-bin.* /*-relay.*
root@IRPMMASTER ~# rsync -av --progress --delete --delete-after /var/lib/
mysql/ $IRPSLAVE:/var/lib/mysql/
```

 The procedure above tries to reduce the downtime of MySQL daemon on \$IRPMMASTER. During this time Bgpd preserves IRP Improvements. Make sure this action takes less than `bgpd.db.timeout.withdraw`.

Start MySQL on \$IRPSLAVE and check if there are no errors at `/var/log/mysqld.log` on CentOS or `/var/log/mysql/error.log` on Ubuntu.


 First \$IRPSLAVE must be checked.

Start MySQL on IRP master and check if there are no errors in MySQL logs as above.

Start replication (Slaves) on both \$IRPMMASTER and \$IRPSLAVE

MySQL Multi-Master replication uses a replication scheme where a replicating master acts as both replication master and replication slave. The steps above configured IRP nodes to be capable of acting as replication masters. Here we ensure that both nodes are capable of acting as replication slaves.

IRP provides a template command that needs to be adjusted for each \$IRPMMASTER and \$IRPSLAVE and will instruct MySQL daemon to take the replication slave role. The template is `/usr/share/doc/irp/changemasterto.template`.

 The template generates a different command for each \$IRPMMASTER and \$IRPSLAVE nodes and requires multiple values to be reused from configuration settings described above. The command that is run on one node points to the other node as its master.

Make \$IRPMMASTER a replication slave for \$IRPSLAVE:

Listing 2.75: Set \$IRPMMASTER as replication slave

```
$IRPMMASTER-mysql>
CHANGE MASTER TO
MASTER_HOST=' $IRPSLAVE-ip-address',
```

```

MASTER_USER='irprepl',
MASTER_PASSWORD=' $IRPSLAVE-password>',
MASTER_PORT=3306,
MASTER_LOG_FILE= '$IRPSLAVE--bin.000001',
MASTER_LOG_POS= <$IRPSLAVE-bin-log-position>,
MASTER_CONNECT_RETRY=10,
MASTER_SSL=1,
MASTER_SSL_CAPATH='/etc/pki/tls/certs/mysql/client/',
MASTER_SSL_CA='/etc/pki/tls/certs/mysql/client/$IRPSLAVE-ca-cert.pem',
MASTER_SSL_CERT='/etc/pki/tls/certs/mysql/client/$IRPSLAVE-client-cert.pem'
',
MASTER_SSL_KEY='/etc/pki/tls/private/mysql/client/$IRPSLAVE-client-key.pem'
',
MASTER_SSL_CIPHER='DHE-RSA-AES256-SHA';

```

i You must manually check what values to assign to `$IRPSLAVE-bin.000001` and `<$IRPSLAVE-bin-log-position>` by running the following MySQL command on `$IRPSLAVE`

```
mysql> show master status
```

For the initial configuration the values for `$IRPSLAVE-bin.000001` and `<$IRPSLAVE-bin-log-position>` must be as follows:

```
Binlog file: $IRPSLAVE-bin.000001
```

```
Binlog position: 106
```

Run the commands below to run the replication and check the slave status:

Listing 2.76: Starting replication slave on `$IRPMMASTER`

```
mysql> START SLAVE \G
mysql> show slave status \G
```

i Check the `Slave_IO_State`, `Last_IO_Errno`, `Last_IO_Error`, `Last_SQL_Errno`, `Last_SQL_Error` values for errors. Make sure there are no errors.

Make `$IRPSLAVE` a replication slave for `$IRPMMASTER`:

Listing 2.77: Set `$IRPSLAVE` as replication slave

```

$IRPSLAVE-mysql>
CHANGE MASTER TO
MASTER_HOST=' $IRPMMASTER-ip-address',
MASTER_USER='irprepl',
MASTER_PASSWORD=' $IRPMMASTER-password>',
MASTER_PORT=3306,
MASTER_LOG_FILE= '$IRPMMASTER-bin.000001',
MASTER_LOG_POS= <$IRPMMASTER-bin-log-position>,
MASTER_CONNECT_RETRY=10,
MASTER_SSL=1,
MASTER_SSL_CAPATH='/etc/pki/tls/certs/mysql/client/',
MASTER_SSL_CA='/etc/pki/tls/certs/mysql/client/$IRPMMASTER-ca-cert.pem',
MASTER_SSL_CERT='/etc/pki/tls/certs/mysql/client/$IRPMMASTER-client-cert.pem'
',

```

```
MASTER_SSL_KEY='/etc/pki/tls/private/mysql/client/$IRPMaster-client-key.pem
',
MASTER_SSL_CIPHER='DHE-RSA-AES256-SHA';
```

i You must manually check what values to assign to `$IRPMaster-bin.000001` and `<$IRPMaster-bin-log-position>` by running the following MySQL command on `$IRPMaster`

```
mysql> show master status
```

For the initial configuration the values for `$IRPMaster-bin.000001` and `<$IRPMaster-bin-log-position>` must be as follows:

```
Binlog file: $IRPMaster-bin.000001
Binlog position: 106
```

Run the commands below to run the replication and check the slave status:

Listing 2.78: Starting replication slave

```
mysql> START SLAVE \G
mysql> show slave status \G
```

Run IRP and HTTPD services on `$IRPMaster` and `$IRPSlave` (OS with Systemd):

Listing 2.79: Starting IRP services and Frontend (OS with Systemd)

```
# systemctl start irp.target
```

Run IRP and HTTPD services on `$IRPMaster` and `$IRPSlave` (OS with RC-files):

Listing 2.80: Starting IRP services and Frontend (OS with RC-files)

```
# service httpd24-httpd start
# service irp start
```

i Start services on `$IRPMaster` first if the actions above took very long in order to shorten MySQL downtime.

Configure Failover using Wizard on `$IRPMaster`

These steps should only be performed once SSH communication channel and MySQL Multi-Master irp database replication have been setup and are verified to work. Refer subsections of [Failover configuration \(Not supported in IRP Lite\)](#) above.


Run failover wizard:

Login into IRP's Frontend on `$IRPMaster` node and run Configuration -> Setup wizard -> Setup Failover.

⚠ A valid failover license should be acquired before failover configuration settings become available.

Configure IRP failover:

Follow Setup Failover wizard steps and provide the required details. Refer [Setup Failover wizard \(Not supported in IRP Lite\)](#) for details. Once your configuration changes are submitted IRP will validate configuration and if it is valid the configuration will be synced to `$IRPSlave`.


 Only after this synchronization step takes place will \$IRPSLAVE node know what is its role in this setup.

Apply configuration changes to edge routers:

Ensure designated probing IPs, BGP session(s) are also setup on edge routers. Refer to corresponding sections of this document for details.

Enable failover:

Use IRP's Frontend Configuration -> Global -> Failover to configure IRP failover on both nodes. Monitor both IRP nodes and ensure everything is within expectations.

 It is recommended that after finishing the preparatory steps above both IRP master and slave nodes run with disabled failover for a short period of time (less than 1 hour) in order to verify that all IRP components and MySQL Multi-Master replication work as expected. Keep this time interval short to avoid a split-brain situation when the two nodes make contradictory decisions.

Synchronize RRD statistics to \$IRPSLAVE

RRD statistics is collected by both IRP failover nodes and needs not be synchronized during normal operation. Still, when IRP failover nodes are setup at a large time interval between them it is recommended that RRD files are synchronized too. This will ensure that past statistics is available on both IRP nodes. During short downtimes of each of the nodes synchronization is less useful but can be performed in order to cover the short downtime gaps in RRD based graphs.

Sample command to synchronize IRP's RRD statistics:

Listing 2.81: Synchronize RRD

```
root@IRPMASTER ~ # rsync -av /var/spool/irp/ $IRPSLAVE:/var/spool/irp
```

2.6.2 Re-purpose operational IRP node into an IRP failover slave

Sometimes an existing fully configured IRP node is designated to be re-purposed as a failover slave. Take the following actions to make a node as an IRP slave:

1. upgrade IRP to the version matching IRP failover master node
2. create a backup copy of your configuration
3. delete the following configuration files: `/etc/noction/irp.conf`, `/etc/noction/exchanges.conf`, `/etc/noction/policies.conf`, `/etc/noction/inbound.conf`, `/etc/noction/user_directories.conf`
4. proceed with configuration as detailed in [Initial failover configuration](#)

2.6.3 Re-purpose operational IRP failover slave into a new master

If a master node fails catastrophically and cannot be recovered an operational IRP slave can be re-purposed to become the new master. Once re-purposing is finished a new slave node can be configured as detailed in [Initial failover configuration](#).

Failover status bar (3.2.2) includes a context menu with functions to switch IRP failover node role. Simply click on the failover status bar's 'Slave node' icon and when prompted if you want to switch role confirm it.

i If you switched a node's role by error, simply change it one more time to revert to previous configuration.

2.6.4 Recover prolonged node downtime or corrupted replication

Multi-Master replication used by IRP failover is able to cope with downtime of less than 24 hours. Replication will not be able to continue in case of prolonged downtime or corrupt replications. Recovery in this case might use either:

- new server: follow configuration steps as detailed in [Initial failover configuration](#).
- same server: follow recovery steps below.

MySQL Multi-Master recovery prerequisites

Before proceeding with recovery of replication the following prerequisites must be met:

- Currently active IRP node is designated as MySQL sync 'origin'. This node currently stores reference configuration parameters and data. These will be synced to the node being recovered and we designate it as 'destination'.
- Recovery should be scheduled during non-peak hours.
- Recovery must finish before `bgpd.db.timeout.withdraw` (default 4h) expires. If recovery can not be completed in time it is required to start MySQL on the active node.

MySQL Multi-Master recovery procedure

It shall be noted that recovery closely follows actions described in [and](#) with the clarification that data and files from origin node target destination node (instead of IRP failover master files targeting IRP failover slave). Refer sections [, for](#) details about the steps below.

The steps are as follows:


1. destination: stop httpd, irp, mysqld
2. origin: sync `/etc/noction/db.*` to slave:`/etc/noction/`
3. origin: sync `/root/.my.cnf` to slave:`/root/.my.cnf`
4. origin: sync `/var/lib/mysql/` to slave:`/var/lib/mysql/`
exclude files:
`master.info relay-log.info -bin.* -relay.*`

wait until sync at (4) succeeds and continue with:
5. origin: stop httpd, irp (except bgpd), mysqld
6. origin: delete files `master.info relay-log.info -bin.* -relay.*`
7. origin: sync `/var/lib/mysql/` to slave:`/var/lib/mysql/`
8. destination: start mysqld and check `/var/log/mysqld.log` for errors
9. origin: start mysqld and check `/var/log/mysqld.log` for errors
10. origin: run `CHANGE MASTER TO` from the `/usr/share/doc/irp/changemasterto` template
11. destination: run `CHANGE MASTER TO` from the `/usr/share/doc/irp/changemasterto` template
12. destination: show slave status \G

13. origin: show slave status \G
14. origin: start IRP (bgpd should be already running), httpd
15. destination: start IRP, httpd


2.7 Frontend Configuration


The system frontend can be accessed over HTTP or HTTPS using the main IP or the FQDN of the IRP server. If the IRP server is firewalled, the TCP:80 and TCP:443 port must be open for each IP/network that should have access to the frontend.

 The default username and password are “**admin**”/”**admin**”. It is strongly recommended to change the password ASAP. Feel free to use the frontend once it becomes accessible. Alternatively local account passwords can be set using `irp-passwd` utility under root account directly on the IRP server, for example:

```
#irp-passwd.sh <username> <new-password>
```

Default “**admin**” user is created only for the initial IRP launch. Separate user accounts must be created for each user. A separate administrative account must be created for the Noction support team as well.

 Do not use the default “**admin**” user account for regular system access.

 For the frontend to function correctly, server’s time zone should be configured in `/etc/php.ini`. Example: `date.timezone=America/Chicago`.

2.8 Administrative Components

There are two additional components in the IRP system: **dbcron** and **irpstatd**. For detailed configuration parameters description, please see the [Administrative settings](#) section.

Dbcron handles periodic database maintenance tasks and statistics processing.

Irpstatd gathers interface statistics from the edge routers. For this component to function properly, the [peer.X.snmp.interfaces](#), [global.ifstats](#) and [SNMP hosts configuration](#) need to be set for each provider.

For more details, see the [Providers configuration](#) and [Providers settings](#) sections.

2.9 IRP Initial Configuration

The platform initial configuration can be performed by using the Initial Setup wizard ([Initial Setup](#)) which can be accessed via the main platform IP address over HTTP or HTTPS protocol.

Example: `https://10.11.12.13`

2.10 IRP software management

Software repository

IRP packages (current, new and old versions) are available in Noction’s repositories hosted at repo.noction.com. Coordinate getting access to the repository with Noction’s representatives.

Software installation and upgrade

Once the IRP packages repository is configured standard CentOS package management tools like yum are used to install, upgrade or downgrade IRP.

Installation:

CentOS:

i In case of upgrading from IRP Lite 3.0 to later versions, old PHP 5.6 packages should be removed prior to upgrade by running "rpm -qa rh-php56* | xargs rpm -e --nodeps"

```
yum install irplite
```

Ubuntu Server:

```
apt-get update
apt-get install irplite
```

i Optional packages *irplite-documentation* and *irplite-api-samples* should be installed separately.

Upgrade to latest version:

CentOS:

```
yum upgrade "irplite*"
```

Ubuntu:

```
apt-get update
apt-get install --only-upgrade irp\*
```

Downgrade to a specific version:

CentOS:

```
yum downgrade "irplite*1.0*"
```

Ubuntu Server:

i IRP Lite added support for Ubuntu Server starting with version 2.0. IRP Lite versions prior to 2.0 will not be offered for Ubuntu Server.

Check for available IRP versions:

```
apt-cache policy irplite
```

Edit file `/etc/apt/preferences.d/irplite` (refer to `apt_preferences` man page for package pinning) to pin desired version:

```
Package: irplite irplite-*
Pin: version 2.0.0-RELEASE~build11806~trusty
Pin-Priority: 1001
```

```
apt-get update
apt-get upgrade irplite
```

2.11 BMP configuration

IRP's BMP monitoring station passively listens on a designated TCP port for monitoring routers to establish BMP sessions. Further BMP setup is performed on monitoring router where

- BMP monitoring station's IP address and port are set and also
- filtering rules regarding what BMP data is sent to IRP are applied if needed.

i By default BMP implementations do not filter routing data sent to a monitoring station (refer to `peer.X.bmp.check_routes`).

BMP related features are configured individually for example:

- BMP monitoring station: `BMP monitoring station settings`.
- primary source of data for current route re-construction: `bgpd.as_path`.

w It is recommended that BMP is set as the primary source for current route reconstruction only when BMP data is available for all providers.

- improvement old and new provider re-probing on AS Path changes: `bgpd.retry_probing.new.bmp_path_change`, `bgpd.retry_probing.old.bmp_path_change`.

2.12 Starting, stopping and status of IRP components

In order to determine the status of IRP components or to start, stop and restart them standard Linux utilities are used.

w Critical errors preventing startup of a particular component are logged to console. Details can be obtained from logs stored in `/var/log/irp/` directory.

Managing software components in OS with systemd

Starting single component:

```
systemctl start explorer
```

i Multiple components separated by space can be listed

Stopping single component:

```
systemctl stop explorer
```

i Multiple components separated by space can be listed

Starting all components:

```
systemctl start irp.target
```

i This also starts all prerequisites

Stopping all components:

```
systemctl start irp-shutdown.target
```

Stopping all components except bgpd:

```
systemctl start irp-shutdown-except-bgpd.target
```

Restarting all components:

```
systemctl start irp-shutdown.target  
systemctl start irp.target
```

Obtaining overall status of all components:

```
systemctl list-dependencies irp.target
```

i Individual statuses can be checked by executing command `systemctl status component_name`

Chapter 3

Using IRP

3.1 Getting started

The **Frontend** is a web interface with a comprehensive set of reports, graphs and diagnostic information, which can reflect the current and historical network state, as well as the benefits of IRP network optimization.

3.1.1 Accessing the system

The system frontend is accessible over HTTPS. IRP's Frontend can be accessed using any major browser, via the main IRP server IP address, or FQDN. E.g: <http://irp-system.noction.com/>. When opening the Frontend's URL, a login form will pop up:

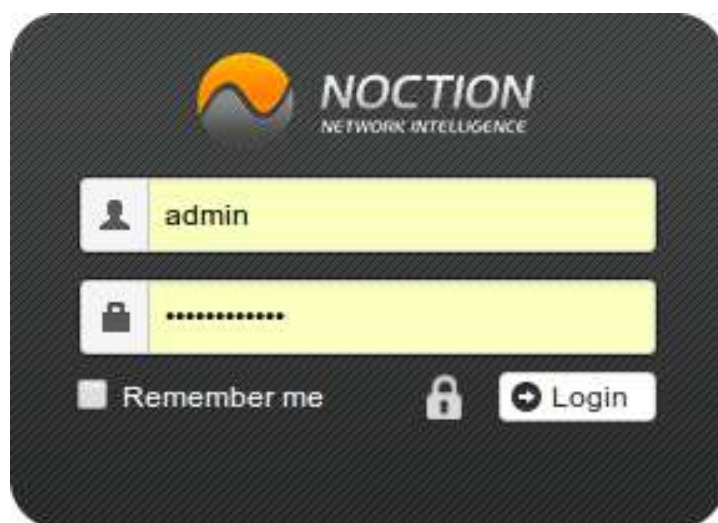


Figure 3.1.1: Frontend login form

The username and password provided by the system administrator must be used. The “Remember me” check-box can be used as well.

See also: [Frontend Configuration](#)

3.1.2 System dashboard

After a successful login, the default Dashboard will become available. It contains a standard set of gadgets - elements, which can contain a reduced version of a report or a graph.

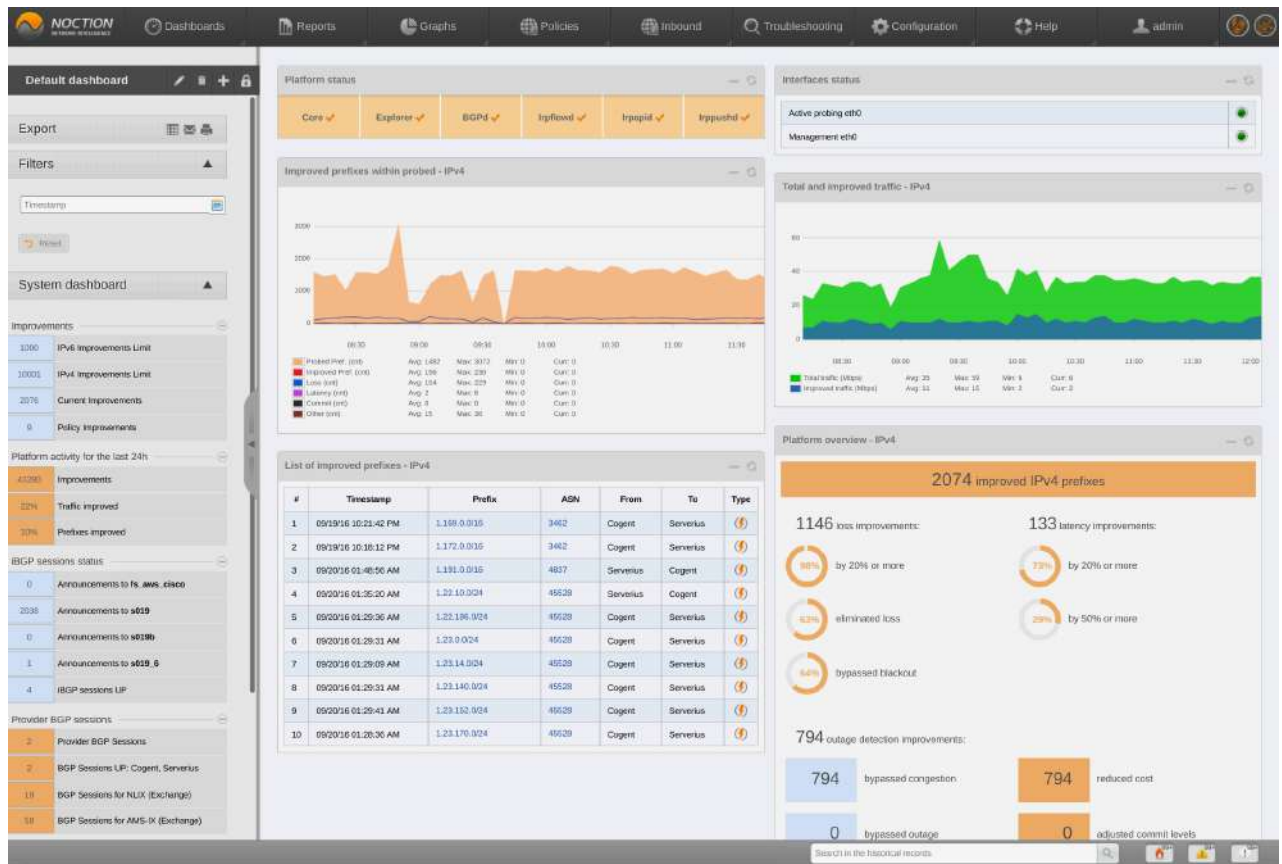


Figure 3.1.2: Default Dashboard

3.2 Frontend structure

The Frontend has several components that operate together and allow a regular user or administrator to access various reports, graphs, or configuration pages.

1. The menu bar (see Figure 3.2.1) allows quick access to any part of the system via its menu items.



Figure 3.2.1: Menu bar

2. The sidebar (See section 3.2.1) is present on the Dashboard or on any graph or report. Its contents depend on the current page, and can include filters, export functions, etc., as explained in the next section.
3. The status bar (see Figure 3.2.2), allows one to access the global search function (Section 3.2.3), as well as system events and notifications (See the [Events](#) section).



Figure 3.2.2: Status bar

3.2.1 Sidebar

All dashboards, reports, graphs and events have a sidebar, located at the left side of the screen. Depending on the current page, the sidebar has a different set of controls, allowing users to manage the dashboard components or to filter the report and graph results.

Every sidebar contains Export, Email and Print buttons. Reports can be exported in .xls, .csv, .pdf formats, while the graphs and dashboard - only as .pdf .



Figure 3.2.3: Common sidebar buttons

The “Email” button allows scheduling email delivery of several reports or graphs, or even an entire dashboard (Figure 3.2.4). Multiple destination email addresses can be used. Most elements in this form have informative tool-tips, that can be seen when hovering over a specific control. Note that Email Reports can be removed from <Username>→Emailing from the top panel.

Figure 3.2.4: Email schedule configuration

In the case of a report, custom filters can be applied, so that only specific information will be displayed in the report.

Figure 3.2.5: Report filters

On graphs, reports and dashboard, the time period which the information is being provided for, can be selected by using the “Timestamp” button. An intuitive control will open, and the desired time period can be selected.

02/12/14 13:00 - 02/13/14 13:00

January, 2014 February, 2014 March, 2014

Date range: 02/02/14 00:00 02/08/14 23:59

Custom

Cancel Save

Figure 3.2.6: Time period selection

The number of results can be adjusted on a report page, using the “Results per page” slider.

Results per page: 50

20 50 100 200 300

Figure 3.2.7: Adjusting the number of results per page

Dashboards have a different sidebar layout.

⚠ The Delete button cannot be found in the default dashboard. The default dashboard cannot be deleted.

System dashboard represents the overall system information which includes:

- Configured maximum number of improvements (Max IPv6 Improvements will be shown if IPv6 is enabled)
- Number of Routing Policies improvements
- Number of improvements performed today
- The amount of traffic improved today
- Percentage of prefixes improved today
- Number of operating iBGP sessions
- Number of announcements to the edge router
- Announcements to each from configured routers
- Number of BGP sessions with the providers
- Number of operating BGP sessions with the providers

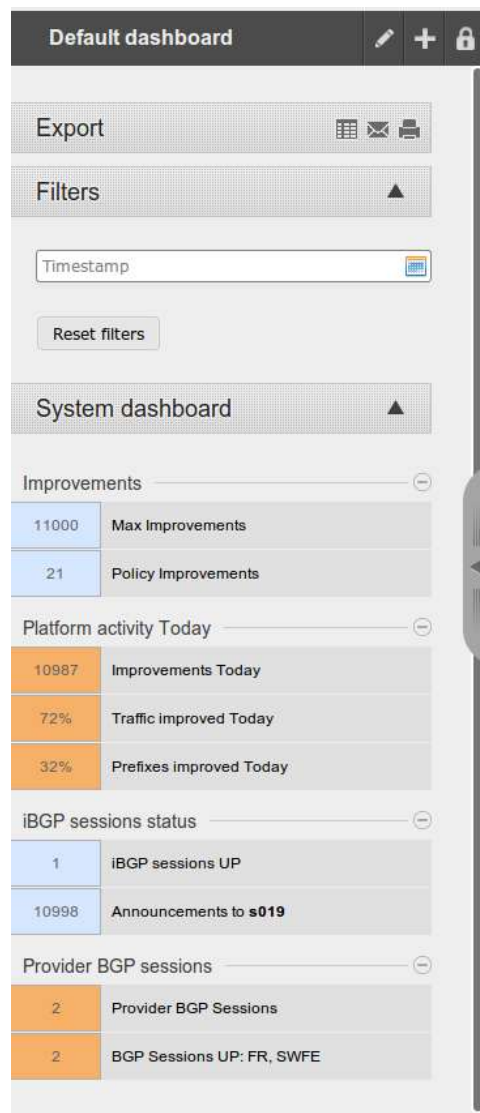


Figure 3.2.8: Dashboard buttons

See also: [Customizing the dashboard](#).

3.2.2 Failover status (Not supported in IRP Lite)

When failover is configured and operational the status of the components is highlighted below system dashboard for each master and slave nodes.

i A valid failover license is required



Figure 3.2.9: Failover status of master node

Slave node status shall be considered together with component status in the dashboard. During normal operation components Core, Explorer and Irippushd are stopped.



Figure 3.2.10: Failover status of slave node

3.2.3 Global search

The Frontend allows to search for historical information on optimized prefixes, AS Numbers and AS Names. The search function can be accessed using the Search box in the status bar.



Figure 3.2.11: Search box

Enter the needed prefix, AS number or AS name in the search box, then press Enter or click on the magnifying glass button.

While entering the search terms, results are dynamically loaded in a modal window (Figure 3.2.12).



Figure 3.2.12: Dynamical global search results

i When displaying the complete results, specific prefixes can be manually removed, or scheduled for re-probing, by using the buttons next to each result (figure 3.2.13).

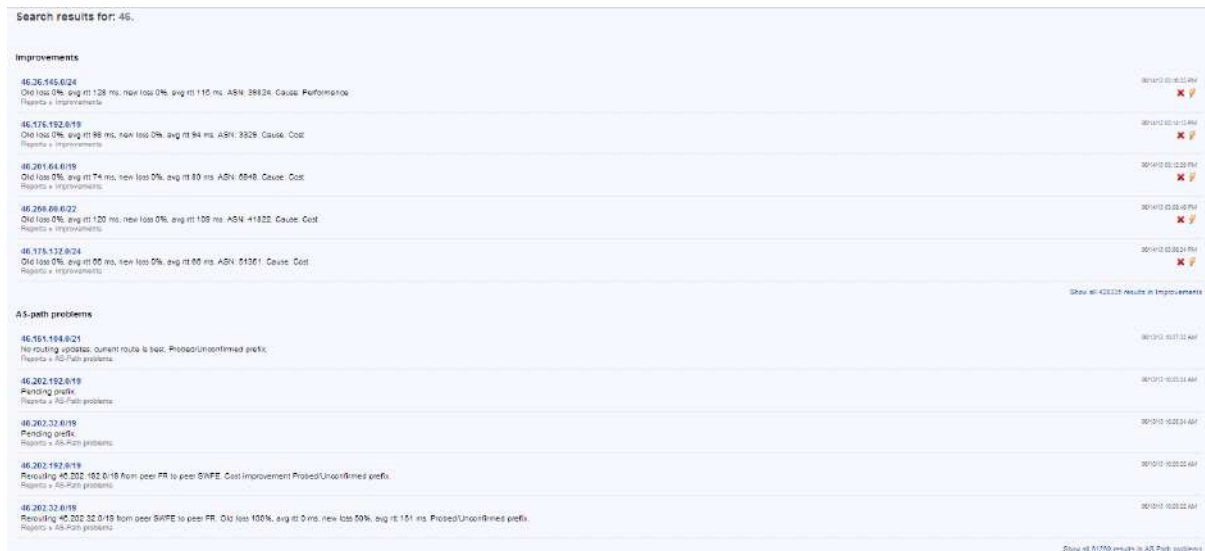


Figure 3.2.13: Global search results

The *Show all X results in Y category* buttons can also be used, to display all matching results from the selected category (Improvements or AS Path problems)

3.2.4 Warning bars

The warning bars are shown in case the attention should be drawn to operational/performance issues, some specific IRP configuration settings or license/payment issues.



Figure 3.2.14: Precedence and improve in a group warning



Figure 3.2.15: Internal Monitor warning



Figure 3.2.16: Explorer performance warning



Figure 3.2.17: License expiration warning

3.3 Customizing the dashboard

The Default dashboard is the first screen that can be seen after a successful login to the system. It contains a default set of gadgets, that can be customized by adding additional gadgets, removing unnecessary gadgets, or modifying specific gadget's settings.

3.3.1 Custom dashboard gadgets

3.3.1.1 Platform status

Platform status			
BGPd (1) ✓	Core ✓	Explorer ✓	Irpflowd ✓

Figure 3.3.1: Platform status

3.3.1.2 Interfaces status





Interfaces status	
Active probing eth0	
Management eth0	
Collector eth1	
Collector eth2	

Figure 3.3.2: Interfaces status

3.3.1.3 List of improved prefixes











List of improved prefixes						
#	Time stamp	Prefix	ASN	From	To	Type
1	10/31/13 16:49:17	94.181.20.0/22	41661	SWFE	FR	
2	10/31/13 16:49:17	92.81.224.0/19	9050	SWFE	FR	
3	10/31/13 16:49:17	88.113.64.0/19	719	SWFE	FR	
4	10/31/13 16:49:17	178.127.128.0/19	6697	FR	SWFE	
5	10/31/13 16:49:11	90.189.0.0/19	41440	SWFE	FR	
6	10/31/13 16:49:11	77.34.224.0/20	12332	SWFE	FR	
7	10/31/13 16:49:00	37.79.32.0/19	34875	SWFE	FR	
8	10/31/13 16:49:00	168.62.0.0/19	8075	SWFE	FR	
9	10/31/13 16:48:55	90.157.32.0/19	35154	FR	SWFE	
10	10/31/13 16:48:55	78.30.192.0/19	35816	SWFE	FR	

Figure 3.3.3: List of improved prefixes

3.3.1.4 AS-Path problems











AS-path problems					
#	AS-Pattern	Problem	Detection time	Problem status	
				Confirmation rate	Status
1	50384-34892	Outage	10/31/13 04:52:46 PM	20%	
2	12389-43975	Congestion	10/31/13 04:44:58 PM	1.49%	
3	12956-22085	Congestion	10/31/13 04:21:32 PM	0%	
4	31133-49718	Outage	10/31/13 04:19:22 PM	14.29%	
5	31133-34364	Outage	10/31/13 04:05:23 PM	14.29%	
6	12389-43975	Congestion	10/31/13 04:02:29 PM	1.52%	
7	6453-812	Congestion	10/31/13 03:15:14 PM	0%	
8	1273-10030	Outage	10/31/13 03:05:25 PM	66.67%	
9	8075-8069	Congestion	10/31/13 02:31:27 PM	11.11%	
10	10026-8075	Congestion	10/31/13 02:50:35 PM	22.22%	

Figure 3.3.4: AS-Path problems

3.4 Reports

The IRP Frontend comes with a comprehensive set of reports, reflecting the current state of the network as well as overall statistics on the system performance. As explained in the [Sidebar](#) section, the results can be filtered by specific criteria for each report, and the time period of the reported data can also be selected.

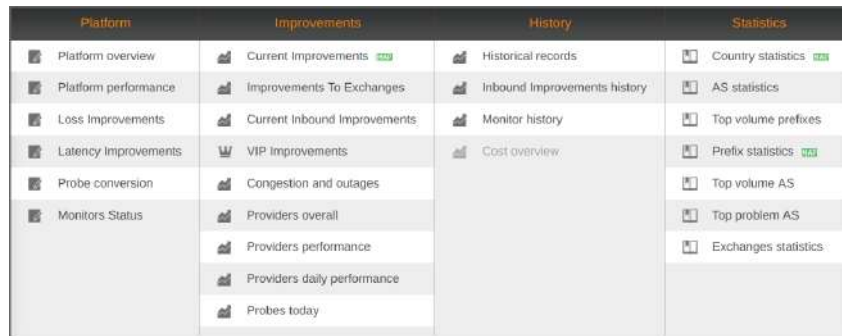


Figure 3.4.1: Reports menu

3.4.1 Platform overview

The “Platform Overview” report provides information on the overall rate of improvements based upon latency, loss and cost. This report can be used for a quick overview of the overall system performance and improvements.

One can see here that almost all the improved routes had a loss drop of 20% or more. For more than 80% of the improved routes the loss was fully eliminated. There are also 7% of the improved routes, which were redirected from a complete blackout. The report also provides the amount of the accomplished route improvements made, based upon latency, and cost (if the Cost optimization is enabled, see [Cost optimization](#)).



Figure 3.4.2: Platform overview

3.4.2 Platform performance

The “Platform Performance” report offers overall statistics on improvements made, based upon performance reasons, commit control and cost reasons on a daily, weekly and monthly basis.

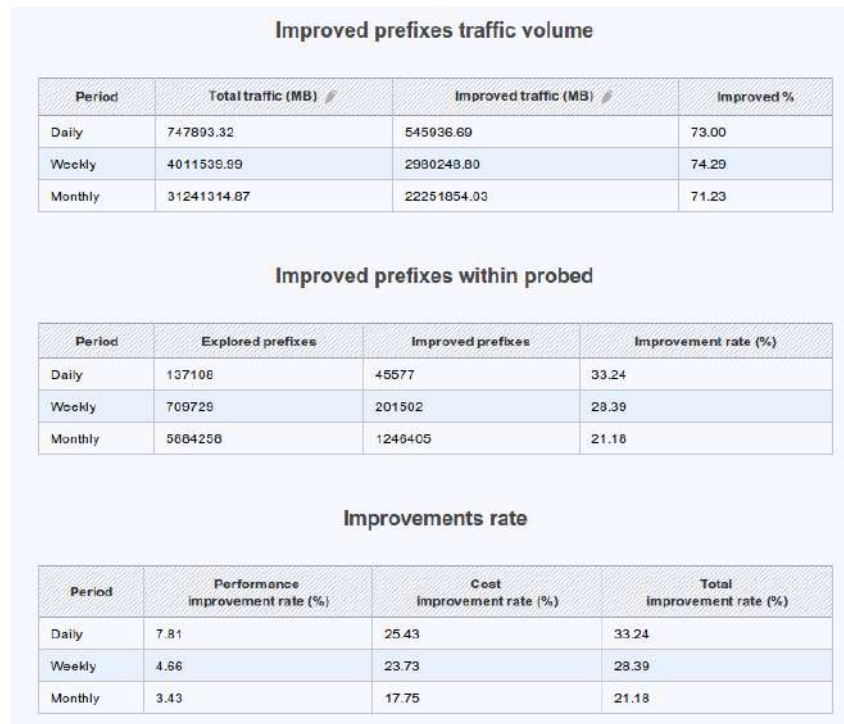


Figure 3.4.3: Platform performance

3.4.3 Loss Improvements

The Loss Improvements report highlights how much of the "Packet loss problem" IRP detected and was able to solve during a specific timeperiod.

The report depicts loss average rates for groups of destinations before and after IRP assessed and re-routed the traffic. The groups proposed and highlighted in number of prefixes depicted by the pie charts below represent:

- Sampled destinations that cover all destination prefixes - good or problematic - that have been probed by IRP;
- Problem destinations include only routes/prefixes where IRP detected packet loss and improved them. The number of such destinations, as expected, is usually significantly lower. At the same time the loss before and after IRP improved them average to a much larger values;
- Loss eliminated represents those problem destinations that IRP was able to improve to a better route with zero loss. By definition this group will have a zero remaining loss but the original average loss highlights how much better those end users perceive their service.;
- Loss reduced represents those problem destinations that IRP improved but was not able to eliminate loss completely.

In absolute values the bar chart highlights the average loss rate values before and after improvement for the groups of destinations listed above. Values of 1, 3, 5,7% are usual for the . Values exceeding 10% are a bit worrisome. While it allows IRP to boast a bigger impact for your network it also means that the routing information you receive is not very accurate and that your providers might not be making the very best effort to service your traffic.

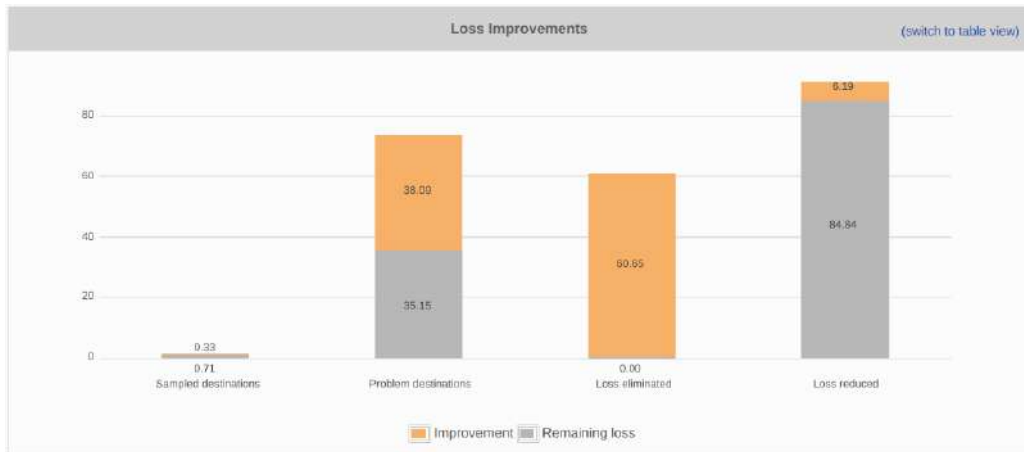


Figure 3.4.4: Loss improvements

Loss relative rates highlighting the percentage magnitude of the improvement in loss rates can be seen by opting for an 100% vertical axis on the reports filters. This view quantifies how much better loss characteristics on the network improved for various groups of end users. Notice how the ~1% loss figure above for all Sampled destinations expands to approximately ~32% of total number of lost packets being eliminated after IRP discovered the better routes and applied the improvements.

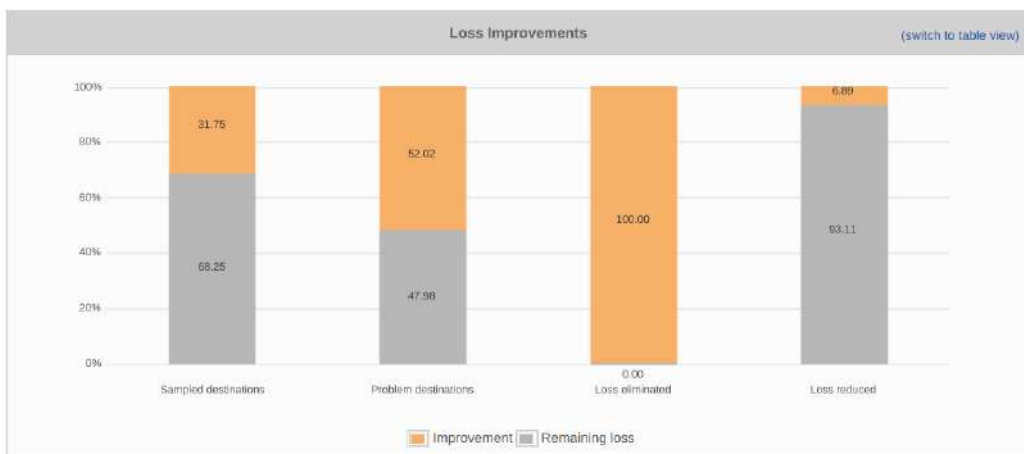


Figure 3.4.5: Loss relative rates

Problem destinations - highlights destination counts of affected routes in the groups as above. The charts present the rate of problem destinations identified by IRP and subsequently highlights how well IRP was able to improve on them. The counts allow you to infer whether there are 20 customers suffering minor packet loss (say, 5% each) or 2 customers suffering severe packet loss (for example 50% each). If for the top and middle graphs above IRP was able to reduce packet loss by 80% these pie charts allow you to infer that there were 20 or 2 customers affected and that for some of them IRP solved the problem completely and how many others are still suffering.

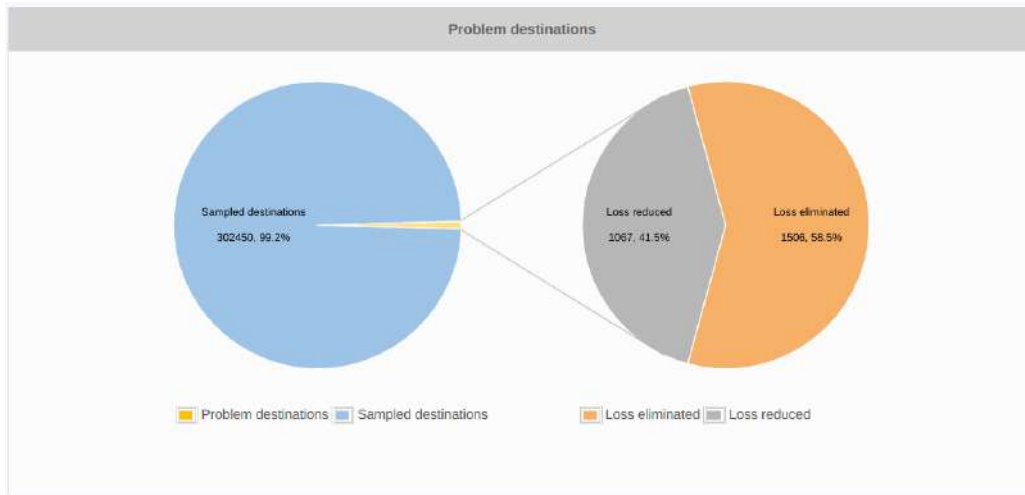


Figure 3.4.6: Loss-improved destinations

The report can be switched to table view in order to review actual numbers across the different groups and the corresponding loss average values.

3.4.4 Latency Improvements

The Latency Improvements report presents an overview for average latency values before and after IRP optimization during a specific timeperiod and depicts them across the following populations:

- Sample destinations that cover good and problematic destinations that IRP probed during selected timeperiod;
- Problematic destinations cover those routes/prefixes that IRP assessed to have excessive latency rates and identified a better route via a different upstream provider;
- 20% or more cover those destinations where IRP was able to identify a route with latency improvements of 20% or more;
- 50% or more cover those destinations where IRP was able to identify a route with latency improvements of 50% or more.

i Note that the 20% and 50% groups are depicted only to preserve similarity in the number of groups in Loss Improvement report. At the same time the 20% and 50% thresholds represent significant quality of service improvements for affected end users and should warrant special consideration.

Latency values bar-chart highlights the latency averages before and after improvement for the designated groups of destinations.



Figure 3.4.7: Latency values (ms)

Percentages of latency improvement averages can be seen by opting to use 100% vertical axis instead of the usual milliseconds. The option is available in report filters.

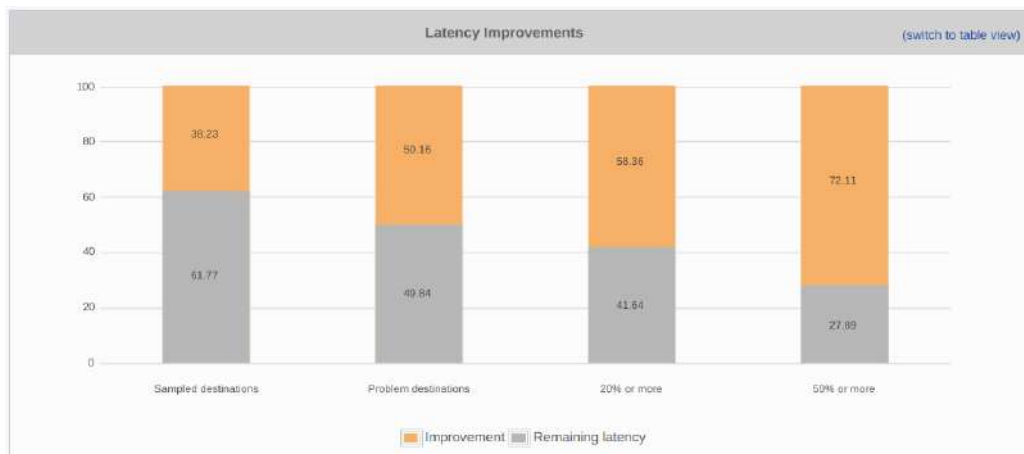


Figure 3.4.8: Latency values (%)

Latency destinations - highlights destination counts in the groups as above.

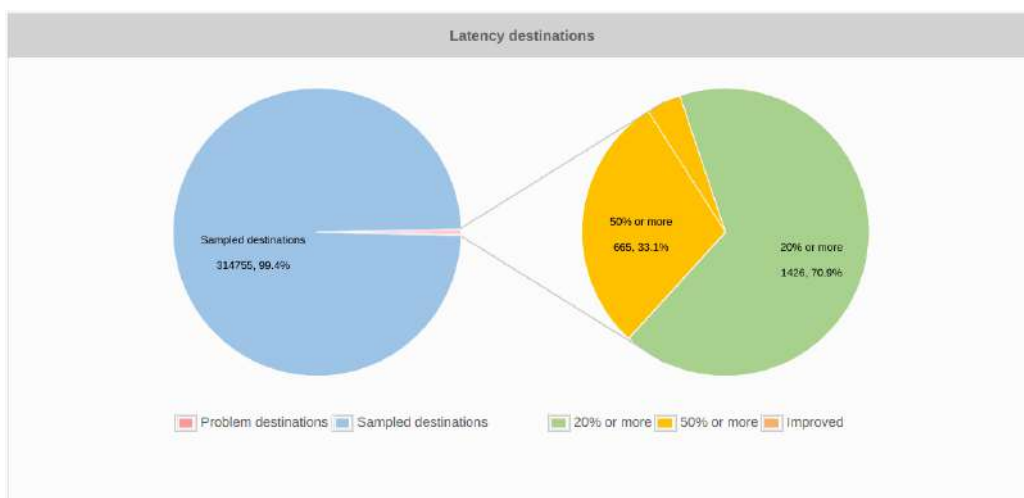


Figure 3.4.9: Latency destinations

The report can be switched to table view in order to review actual numbers across the different groups and the corresponding averages.

3.4.5 Probe conversion rates

Probe conversion rates - highlights the counts of networks/prefixes probed by IRP and their conversion rate into any of the possible improvements:

- Commit
- Performance
- Cost

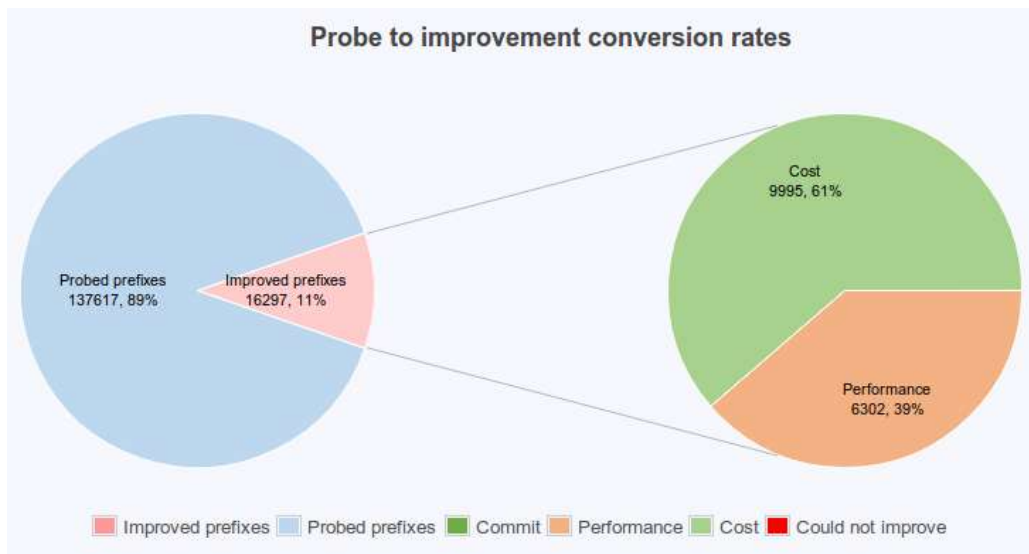


Figure 3.4.10: Probe conversion rates

3.4.6 Current improvements

The “Current Improvements” report provides a list of the currently active improvements as per network. A drill-down capability to a specific ASN or prefix is also provided. The report shows the improved prefix and the providers from and to which the traffic was redirected. It also provides the reason, which the improvement was based upon, along with the performance values before and after the traffic was rerouted.

If necessary, specific improvements can be manually deleted by clicking the checkbox next to the needed prefix, afterward using the “Remove selected” button.

Current Improvements										2017-06-07 11:05:52 - 2017-06-08 11:05:52 (switch to map view)	
#	Timestamp	Prefix	ASN	AS Name	From	To	Type	Age	Prob	Details	Rd
1	17 min 0 seconds ago	115.182.64.0/20	4847	China Networks Inter-Excha	Serverius	Cogent				Old loss 11%, avg rtt 327 ms, new loss 6%, avg rtt 375 ms.	0
2	17 min 0 seconds ago	122.200.73.0/24	4847	China Networks Inter-Excha	Cogent	Serverius				Old loss 0%, avg rtt 291 ms, new loss 0%, avg rtt 185 ms.	0
3	17 min 0 seconds ago	125.208.0.0/24	63530	Beijing Guanghuan newslett	Cogent	Serverius				Old loss 15%, avg rtt 364 ms, new loss 0%, avg rtt 363 ms.	0
4	17 min 0 seconds ago	202.165.181.0/24	4847	China Networks Inter-Excha	Cogent	Serverius				Old loss 0%, avg rtt 363 ms, new loss 0%, avg rtt 260 ms.	0
5	17 min 0 seconds ago	219.232.56.0/21	4847	China Networks Inter-Excha	Cogent	Serverius				Old loss 0%, avg rtt 361 ms, new loss 0%, avg rtt 251 ms.	0
6	17 min 0 seconds ago	68.247.0.0/16	3651	Sprint	Cogent	Serverius				Old loss 100%, avg rtt 0 ms, new loss 46%, avg rtt 104 ms.	0
7	17 min 0 seconds ago	72.60.0.0/16	3651	Sprint	Cogent	Serverius				Old loss 77%, avg rtt 94 ms, new loss 0%, avg rtt 88 ms.	0
8	17 min 5 seconds ago	117.114.192.0/21	4847	China Networks Inter-Excha	Cogent	Serverius				Old loss 9%, avg rtt 361 ms, new loss 0%, avg rtt 229 ms.	0
9	17 min 5 seconds ago	117.234.144.0/21	9829	National Internet Backbone I	Serverius	Cogent				Old loss 63%, avg rtt 180 ms, new loss 44%, avg rtt 164 ms.	0
10	17 min 5 seconds ago	121.52.160.0/24	4847	China Networks Inter-Excha	Cogent	Serverius				Old loss 0%, avg rtt 343 ms, new loss 0%, avg rtt 297 ms.	0
11	17 min 5 seconds ago	174.157.0.0/17	3651	Sprint	Cogent	Serverius				Old loss 82%, avg rtt 115 ms, new loss 0%, avg rtt 114 ms.	0
12	17 min 5 seconds ago	174.157.128.0/17	3651	Sprint	Cogent	Serverius				Old loss 41%, avg rtt 111 ms, new loss 0%, avg rtt 116 ms.	0

Figure 3.4.11: Current improvements

i In some specific cases, the “From” and “To” fields will contain the same provider. This is not a platform issue and is related to the Outage detection algorithm specifics. For more details see the [Outage detection](#) and [VIP Improvements](#) sections.

i The “ASN” field can be empty in case there is no information about the AS number (a prefix belongs to) in the IRP ASN dictionary. As soon as the ASN information is available in the dictionary, the field will be filled accordingly.

Of note is that Current Improvements report also has a map view.

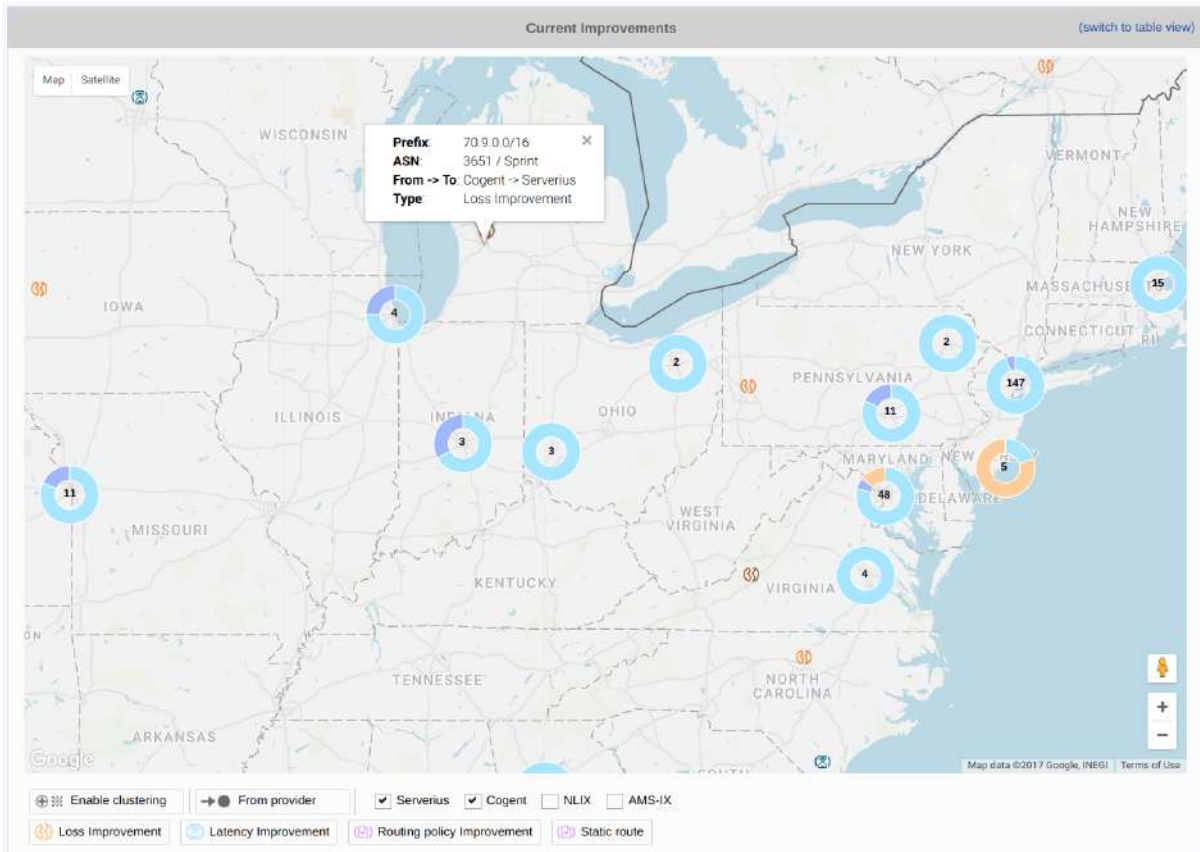


Figure 3.4.12: Current improvements map view

The map view besides offering the usual zooming and panning capabilities also offers clustering and highlighting of improvements by either provider or improvement type.

3.4.7 Improvements to Exchanges (Not supported in IRP Lite)

The “Improvements to Exchanges” report contains a complete list of current improvements where the new provider is an Internet Exchange. The report includes data about improved prefixes and their relative traffic by means of last minute and average bandwidth usage. Average bandwidth is estimated based on current hour statistics.

#	Timestamp	Prefix	Asn	From	To	Age	Prob	Details	Cause	RD	Estimated avg BW (Mbps) #	BW in last minute (Mbps) #
1	03:03:47 PM	106.51.192.0/21	24309	COGENT	NLIX	↓	⊖	Did loss 0%, avg rt 158 ms, new loss 0%, avg rt 168 ms	⚡	Global	0.02	0.00
2	03:06:11 PM	119.29.129.0/24	38182	COGENT	NLIX	↓	⊖	Did loss 0%, avg rt 271 ms, new loss 0%, avg rt 192 ms	⚡	Global	0.60	0.00
3	03:04:53 PM	117.218.0.0/20	9829	COGENT	NLIX	⬇	⊖	Did loss 0%, avg rt 213 ms, new loss 0%, avg rt 216 ms	⚡	Global	0.47	0.00
4	03:08:06 PM	122.177.176.0/20	24560	COGENT	NLIX	↓	⊖	Did loss 0%, avg rt 201 ms, new loss 0%, avg rt 170 ms	⚡	Global	0.14	0.00
5	03:05:41 PM	175.109.98.0/24	38623	COGENT	NLIX	⬇	⊖	Did loss 0%, avg rt 371 ms, new loss 0%, avg rt 328 ms	⚡	Global	0.00	0.00
6	03:06:07 PM	180.244.72.0/21	17974	SERVERIUS-PT1	NLIX	↓	⊖	Did loss 0%, avg rt 200 ms, new loss 0%, avg rt 213 ms	⚡	Global	0.00	0.00
7	03:03:59 PM	202.128.124.0/24	38913	COGENT	NLIX	↓	⊖	Did loss 0%, avg rt 181 ms, new loss 0%, avg rt 187 ms	⚡	Global	0.06	0.00
8	03:06:07 PM	292.44.232.0/21	4618	COGENT	NLIX	↓	⊖	Did loss 0%, avg rt 320 ms, new loss 0%, avg rt 245 ms	⚡	Global	0.02	0.00
9	03:06:27 PM	213.57.132.0/24	12849	COGENT	NLIX	↓	⊖	Did loss 100%, avg rt 0 ms, new loss 0%, avg rt 71 ms	⚡	Global	0.00	0.00
10	03:06:12 PM	219.57.84.0/24	12849	COGENT	NLIX	↓	⊖	Did loss 55%, avg rt 87 ms, new loss 0%, avg rt 78 ms	⚡	Global	1.05	0.00
11	03:05:41 PM	218.156.0.0/16	4766	COGENT	NLIX	⬇	⊖	Did loss 0%, avg rt 317 ms, new loss 0%, avg rt 294 ms	⚡	Global	0.00	0.00
12	03:08:06 PM	38.77.240.0/20	17974	SERVERIUS-PT1	NLIX	⬇	⊖	Did loss 0%, avg rt 228 ms, new loss 0%, avg rt 228 ms	⚡	Global	0.89	1.60

Figure 3.4.13: Improvements to Exchanges

3.4.8 Historical records

The “Historical records” report contains a complete list of active and inactive improvements that were made by the system. The report can be sorted using any of the columns by simply clicking on the column

header.

#	Timestamp	Prefix	ASN	AS Name	Old			New			Type	Prob
					Provider	Loss %	Latency (avg. ms)	Provider	Loss %	Latency (avg. ms)		
1	11/11/13 12:01:28 PM	77.52.96.0/19	21497	PrJSC MTS ...	IX2	100	0	IX2	0	1		
2	11/11/13 11:36:52 AM	77.52.96.0/19	21497	PrJSC MTS ...	A	0	283	IX2	0	1		
3	11/11/13 09:35:32 AM	213.160.192.0/19	13238	Yandex LLC	A	0	46	IX2	0	1		
4	11/11/13 04:21:31 AM	77.52.96.0/19	21497	PrJSC MTS ...	B	0	147	IX2	0	1		
5	11/11/13 03:29:51 AM	209.65.128.0/19	15169	Google Inc...	B	0	94	IX2	0	1		
6	11/10/13 07:21:06 PM	77.52.96.0/19	21497	PrJSC MTS ...	A	0	161	IX2	0	1		
7	11/10/13 06:41:20 PM	77.52.96.0/19	21497	PrJSC MTS ...	IX2	100	0	IX2	0	1		
8	11/10/13 05:41:37 PM	77.52.96.0/19	21497	PrJSC MTS ...	A	0	112	IX2	0	1		
9	11/10/13 04:21:29 PM	77.52.96.0/19	21497	PrJSC MTS ...	IX2	100	0	IX2	0	1		
10	11/10/13 04:01:09 PM	77.52.96.0/19	21497	PrJSC MTS ...	A	0	141	IX2	0	1		
11	11/10/13 02:58:13 PM	83.237.96.0/19	8339	MTS MTS OJ...	B	0	55	IX2	0	1		
12	11/10/13 02:41:24 PM	77.52.96.0/19	21497	PrJSC MTS ...	IX2	100	0	IX2	0	1		

Figure 3.4.14: Historical records

In some specific cases, the “From” and “To” fields will contain the same provider. This is not a platform issue, and is related to the Outage detection algorithm specifics. Please see the [Outage detection](#) section for more details

3.4.9 Providers overall

The “Providers Overall” report provides the number of prefixes that were rerouted from and to each of the providers. In the example below, the traffic exchange between the providers is balanced. However, if the difference between the outgoing and incoming traffic is significant, then the quality of that provider’s network should be questioned. The report also shows the number of improved prefixes, based upon performance and cost, for each of the providers.



Figure 3.4.15: Providers overall

3.4.10 Provider performance

During the day IRP makes many probes to determine performance characteristics of alternative routes.

Provider performance report presents an aggregated view of this measurements for providers. The data on the report highlights Packet loss and Average latency per provider putting the provider with best packet loss data at the top.



Figure 3.4.16: Provider performance

Provider performance report has been extended and now also offers a daily performance view. The extended Provider daily performance report adds the following capabilities:

- a Best (hypothetical) provider that estimates what might be possible to achieve on your network in terms of loss and average latency,
- classification of destinations by distance from your network with the option to review a category of interest,
- filtering by date range allowing review of past data or average performance over longer time intervals,
- a table view with exact details of the data in the report.




Figure 3.4.17: Provider daily performance

Note that while some providers seem better performing (for example, NLIX or AMS-IX in the provided screen capture) the fact that they display a much smaller number of probes clearly indicates they are Internet Exchanges with only a few peers interchanging data with our network.

For a long term distribution of this data refer chart [Provider performance history](#).

3.4.11 Cost overview

 This report is available only if the system is running in Cost optimization mode (see [Cost optimization](#)).

The “Cost Overview” report provides details regarding the cost savings, resulted from running IRP for each provider. It displays the costs that would be normally paid to the providers and the costs incurred after IRP has optimized the routing. The system calculates the total transit cost while running in Cost improvement mode and compares it with the total transit cost that would be incurred without running IRP. This difference is shown on the right side of the table as Cost Savings. The total cost savings include the ones made by the Commit Control as well. If the provider’s billing periods differ, then it is possible to choose a billing period for each provider separately.








Provider	Billing period	Cost (USD)	Default *		Improved by IRP*		Cost savings (USD)
			Billable throughput ** (Mbps) 	Cost(USD)	Billable throughput ** (Mbps) 	Cost (USD)	
A		3.00	121	363	119	357	6
B	11/01/13 - 12/01/13 	3.00	47	141	46	138	3
IX2		0	0	0	0	0	0
			168	504	165	495	9


Figure 3.4.18: Cost overview

3.4.12 Congestion and outages

This report is available only if [Outage detection](#) is enabled.

The “Congestion and outages” report contains all the as-patterns defined as problematic by the system. The problem state is shown in the “Status” column, as follows:

-  - A problem on the specified AS-Pattern is suspected
-  - The problem was not confirmed by additional re-probing
-  - The problem was confirmed, and all related prefixes were rerouted
-  - The problem was confirmed, but no better-performing route has been found

 In the past the report was named “AS pattern problems”.

#	AS-Pattern	Problem	Detection time	Affected prefixes						Problem status	
				Total	Selected for probing	Probed	Pending probing	Problem confirmed	Problem not confirmed	Confirmation rate	Status
1	2914-20377	Congestion	11/11/13 12:21:25 PM	51	51	51	0	0	51	0%	
2	9198-29355	Congestion	11/11/13 12:13:40 PM	57	56	56	0	4	52	7.02%	
3	12956-27699	Congestion	11/11/13 12:05:05 PM	111	111	111	0	0	111	0%	
4	7470-9287	Congestion	11/11/13 11:08:49 AM	2	2	2	0	1	1	50%	
5	12389-43975	Congestion	11/11/13 11:10:18 AM	149	149	149	0	3	146	2.01%	
6	1299-4788	Congestion	11/11/13 11:16:56 AM	14	14	14	0	0	14	0%	
7	9110-12400	Congestion	11/11/13 11:13:06 AM	53	53	53	0	8	45	15.09%	
8	174-12874	Congestion	11/11/13 09:21:45 AM	46	46	46	0	2	44	4.35%	
9	12389-43975	Congestion	11/11/13 09:06:41 AM	148	147	147	0	3	144	2.03%	
10	7473-4804	Congestion	11/11/13 09:01:42 AM	45	45	45	0	3	42	6.67%	
11	21219-28926	Congestion	11/11/13 08:59:31 AM	16	16	16	0	0	16	0%	
12	24708-13194	Outage	11/11/13 08:23:43 AM	37	37	37	0	1	36	2.7%	

Figure 3.4.19: Congestion and outages

3.4.13 Top volume prefixes

The “Top volume prefixes” displays a list of prefixes sorted by the total volume transferred to these remote networks.

#	Prefix	Volume(MB)
1	146.120.112.0/23	307639.45
2	82.145.220.0/22	110744.63
3	66.249.78.0/24	96863.82
4	192.187.96.0/19	93099.26
5	217.69.128.0/20	73946.92
6	188.134.32.0/19	64603.56
7	213.87.136.0/21	54826.76
8	5.10.64.0/19	49511.55
9	188.123.224.0/19	48647.03
10	91.221.66.0/23	47962.48
11	178.124.172.0/24	45565.62
12	109.207.0.0/20	39653.37

Figure 3.4.20: Top volume prefixes

3.4.14 Top volume AS

The “Top Volume AS” reports show the Autonomous Systems with the highest volume of traffic coming from the monitored network. It can help you understand your traffic flow directions. If according to this report a specific AS has significantly more traffic than the others, then the network operator may consider getting a direct connection to this AS, to reduce traffic costs.

Top volume AS				
#	ASN	AS Name	95th (Mbps)	Volume (GB)
141	14420	CORPORACION NACIONAL DE TELECOMUNICACIONES - CNT EP	390.71	0.33
142	37611	Afrihost	1271.44	0.32
143	50266	T-Mobile Thuis BV	0.66	0.32
144	26615	Tim Celular S.A.	0.00	0.32
145	37963	Hangzhou Alibaba Advertising Co.,Ltd. East Software Park, 99 Huaxing Rd. Hangzhou, Ch	190.29	0.31
146	39309	Edutel BV	2.04	0.30
147	3265	Xs4all Internet BV	225.35	0.30
148	59050	Beijing Cloud-Ark Technology Co.,Ltd.	182.05	0.29
149	23650	AS Number for CHINANET jiangsu province backbone	299.18	0.26
150	7600	Escape.net	144.84	0.26
151	45090	Shenzhen Tencent Computer Systems Company Limited Tencent Building, Kejizhongyi Av	152.31	0.26
152	1136	KPN B.V.	396.24	0.25

Figure 3.4.21: Top volume AS

3.4.15 Top problem AS

The “Top Problem AS” report reveals the most problematic Autonomous Systems that the monitored networks are sending traffic to. It does that by showing how many times the traffic going to that AS was rerouted by the IRP.

#	ASN	AS Name	Problems
1	25577	Connexions4London Ltd	3
2	6849	JSC UKRTELECOM,	2
3	48503	Tele2 Sverige AB	1
4	44640	Cactus Ltd.	1
5	21151	Ukrcom Ltd	1
6	35377	A.B.N. JSC	1
7	25592	NETIS TELECOM Inc. Yaroslavl region ISP provider Russia	1
8	34829	LLC Nauka-Svyaz	1
9	15557	Societe Francaise du Radiotelephone S.A	1
10	50780	EAST.NET Ltd.	1
11	42945	Limited liability company Insit	1
12	51004	Sakhalin Cable Telesystems Ltd	1

Figure 3.4.22: Top problem AS

3.4.16 Prefix statistics

This report lists specific prefixes with relevant monthly values such as traffic volume and number of improvements or current unique and top hosts values. To see top hosts expand the + button to expand to latest individual hosts and their corresponding traffic volume. Note that the upper limit for the number of top hosts is constrained by `collector.export.top_volume_ips`.

Prefix Statistics										
#	Month	Prefix	ASN	AS Name	Country	Volume (GB)	Improvements	Policy	Unique IPs	Top Hosts
1	Aug. 2017	89.45.8.0/24	54103	MOD Mission Critical		5033.97	0		2	
2	Aug. 2017	42.112.10.0/24	18403	The Corporation for Finan	VN	70.85	1		5	
3	Aug. 2017	118.107.176.0/24	64050	BGPNET Global ASN	SG	67.28	2		7	
4	Aug. 2017	88.198.0.0/16	24940	Hetzner Online GmbH		65.82	1		4	
#	Timestamp	IP Address			Volume (KB)	Rank				
1	20 seconds ago	88.198.68.8			561.24	1				
2	20 seconds ago	88.198.68.23			19.73	2				
3	20 seconds ago	88.198.25.3			1.91	3				
5	Aug. 2017	112.121.190.0/24	45753	NETSEC NOC	HK	64.57	15		1	
6	Aug. 2017	93.93.96.0/23	44654	Media Network Services /		61.08	4		2	
7	Aug. 2017	116.251.224.0/19	133771	Rapid Shield Company LI		60.34	7		1	
8	Aug. 2017	179.0.12.0/23	52472	FOCUS EL SALVADOR S	SV	60.01	0		2	
9	Aug. 2017	93.93.98.0/23	44654	Media Network Services /		57.74	1		2	
10	Aug. 2017	96.44.188.0/22	8100	QuadraNet, Inc	US	55.84	16		6	
11	Aug. 2017	104.223.76.0/23	8100	QuadraNet, Inc	US	55.54	4		6	
12	Aug. 2017	103.9.105.0/24	58599	Cybergate Limited	BD	54.89	1		1	

Figure 3.4.23: Prefix statistics

Note that Prefix statistics report introduced a map view highlighting with heat-maps those regions of the world where either most traffic or highest number of improvements are made.

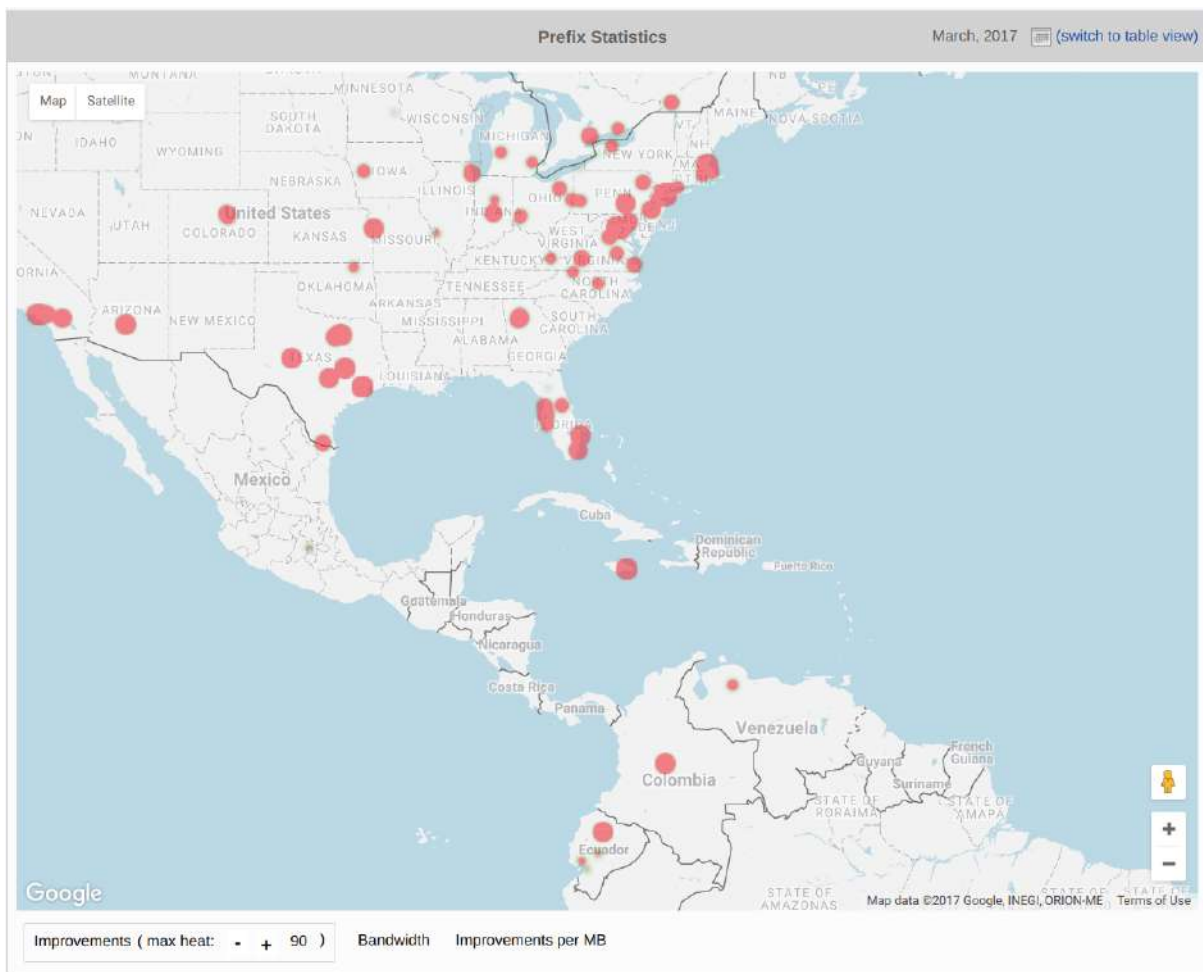


Figure 3.4.24: Prefix statistics map view

3.4.17 AS statistics

The “AS statistics” report aggregates by Autonomous System the total number of improvements, the volume of traffic and the 95th for a selected past month. Note that the choice of month for the 95th and the time period of the report are independent and they can set non-overlapping ranges.

AS statistics						
#	Date	ASN	AS Name	95th	Volume (GB)	Improvements
				July, 2017		
81	2017-11-01	24444	Shandong Mobile Communication Company	0.26	1.69	10800
82	2017-10-26	4808	China Unicom Beijing Province Network	0.14	0.29	10797
83	2017-03-20	4847	China Networks Inter-Exchange	0.51	0.00	10772
84	2017-11-15	23724	IDC. China Telecommunications Corporation	0.31	0.10	10756
85	2017-10-21	3651	Sprint	0.41	0.15	10744
86	2017-05-31	4847	China Networks Inter-Exchange	0.51	0.00	10693
87	2017-10-21	4837	CHINA UNICOM China169 Backbone	0.21	0.06	10672
88	2017-11-12	4847	China Networks Inter-Exchange	0.51	4.85	10662
89	2017-10-29	24444	Shandong Mobile Communication Company	0.26	0.54	10608
90	2017-11-13	23724	IDC. China Telecommunications Corporation	0.31	0.21	10576
91	2017-11-11	701	MCI Communications Services. Inc. d/b/a Ver	0.34	3.90	10568
92	2017-10-31	4837	CHINA UNICOM China169 Backbone	0.21	0.17	10566
93	2017-04-06	17638	ASN for TIANJIN Provincial Net of CT	0.23	0.00	10545

Figure 3.4.25: AS statistics

3.4.18 Country statistics

The “Country statistics” report distributes traffic volume and number of improvements by country. It helps see to what regions most of a network’s traffic is addressed and also the relative number of suboptimal routes towards them that IRP identified and addressed by injecting improvements.

Country Statistics (switch to map view)				
#	Country Name	Country	Volume (GB)	Improvements
1	China	CN	646.05	452027
2	United States	US	4567.72	171039
3	India	IN	171.92	98173
4	South Africa	ZA	36.26	44059
5	Australia	AU	63.19	39997
6	Rest of the world	--	5295.77	9296
7	Colombia	CO	10.18	8749
8	Ecuador	EC	2.77	5894
9	Jamaica	JM	3.49	4786
10	Singapore	SG	268.79	3227
11	Japan	JP	1.42	1144

Figure 3.4.26: Country statistics

Country statistics report also displays a map view highlighting countries with most/least number of improvements.

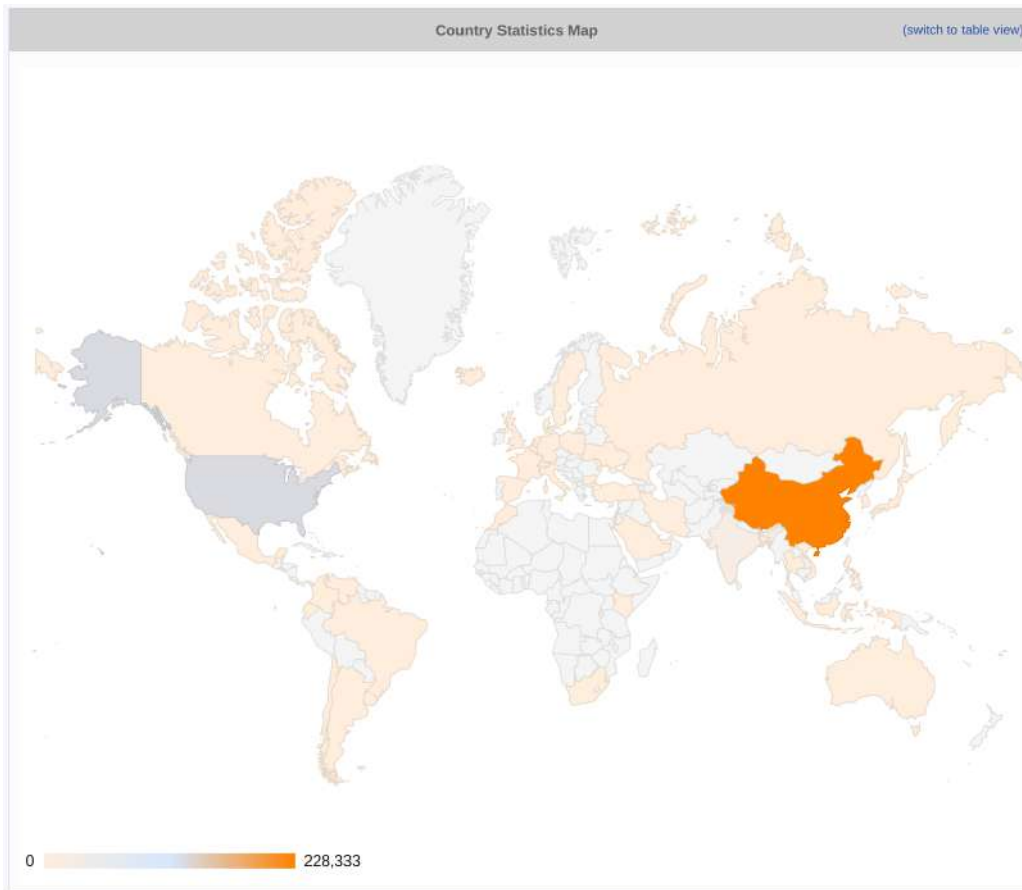


Figure 3.4.27: Country statistics map view

3.4.19 Exchanges statistics

The “Exchanges statistics” report lists statistics regarding communications with peers on Internet Exchanges. Filters and sorting help identify the required details.

i In the past this report was named “Bandwidth by Peer”

Exchanges statistics								
#	Timestamp	Peering partner	Provider	IP Version	Volume Outbound (MB) ↕	Volume Inbound (MB) ↕	Packets Outbound	Packets Inbound
1	Jun. 2017	80.249.208.90	AMS-IX	4	368.70	0.00	156450	0
2	Jun. 2017	80.249.212.146	AMS-IX	4	104.11	0.00	8648	0
3	Jun. 2017	80.249.209.150	AMS-IX	4	62.79	0.00	52525	0
4	Jun. 2017	193.239.117.165	NLIX	4	16.31	0.00	8810	0
5	Jun. 2017	193.239.117.234	NLIX	4	12.71	0.00	7098	0
6	Jun. 2017	193.239.117.110	NLIX	4	11.92	0.00	15329	0
7	Jun. 2017	80.249.208.247	AMS-IX	4	10.15	0.00	10213	0
8	Jun. 2017	80.249.210.125	AMS-IX	4	8.86	0.00	5599	0
9	Jun. 2017	193.239.116.25	NLIX	4	7.76	0.00	4305	0
10	Jun. 2017	80.249.208.200	AMS-IX	4	7.72	0.00	5835	0
11	Jun. 2017	80.249.209.20	AMS-IX	4	7.02	0.00	5217	0
12	Jun. 2017	80.249.208.50	AMS-IX	4	3.89	0.00	2177022	0

Figure 3.4.28: Exchanges statistics

3.4.20 Current inbound improvements (Not supported in IRP Lite)

The “Current inbound improvements” report provides a list of the currently active improvements. A drill-down capability to a specific ASN or prefix is also provided. The report shows the improved prefix and the providers from and to which the traffic was redirected. It also provides the reason, which the improvement was based upon, along with the performance values before and after the traffic was rerouted. If necessary, specific improvements can be manually deleted by clicking the check-box next to the needed prefix, afterward using the “Remove selected” button.

Current Inbound Improvements						
#	Timestamp	Link1	Link2	Prefix	Type	<input type="checkbox"/>
1	03/04/16 05:49:09 AM	-	-	10.1.7.0/24		<input type="checkbox"/>
2	03/04/16 04:14:10 AM	-	-	10.1.3.0/24		<input type="checkbox"/>
3	03/04/16 03:58:34 AM	2	-	10.1.8.0/24		<input type="checkbox"/>
4	03/04/16 03:23:08 AM	-	2	10.1.6.0/24		<input type="checkbox"/>
5	03/04/16 02:01:42 AM	-	2	10.1.9.0/24		<input type="checkbox"/>
6	03/04/16 02:01:34 AM	1	-	10.1.4.0/24		<input type="checkbox"/>
7	03/04/16 01:34:11 AM	2	-	10.1.2.0/24		<input type="checkbox"/>
8	03/04/16 01:30:56 AM	1	-	10.1.1.0/24		<input type="checkbox"/>

1

Remove all Remove selected

1 - 14 of 14 results found

Figure 3.4.29: Current inbound improvements

Refer [Inbound optimization \(Not supported in IRP Lite\)](#) for details.

3.4.21 Inbound Improvements history (Not supported in IRP Lite)

The “Inbound Improvements history” report provides a detailed list of past inbound improvement actions. Filtering for example by specific prefix or improvement type is provided. The report shows the improved prefix and the new preponds count towards a provider.

Inbound Improvements History							
#	Timestamp	Prefix	IP Version	Provider	Preponds	Type	Action
1	52 min 38 seconds ago	149.11.0.0/16	4	Cogent	-		Deleted
2	2017-06-22 16:13:06	185.34.201.0/24	4	Cogent	3		Updated
3	2017-06-22 15:28:30	185.34.200.0/24	4	Serverius	2		Updated
4	2017-06-22 15:24:58	185.34.202.0/24	4	Serverius	3		Updated
5	2017-06-22 14:05:10	185.34.202.0/24	4	Serverius	4		Updated
6	2017-06-22 13:21:37	185.34.200.0/24	4	Serverius	3		Updated
7	2017-06-22 11:00:21	185.34.200.0/24	4	Serverius	2		Updated
8	2017-06-22 11:00:21	185.34.202.0/24	4	Serverius	1		Updated

Figure 3.4.30: Inbound Improvements history

Refer [Inbound optimization \(Not supported in IRP Lite\)](#) for details.

3.4.22 Probes today

The “Probes Today” report provides probing details, including probes that did not warrant an improvement or failed completely. The report shows probed prefixes and actual selected probing IPs in that prefix with details about probe state and actual measurements across all providers. Missing details for some providers, for example for partial providers or Internet exchanges indicate that either the provider was stopped, the prefix is not serviced by the provider or entirely a probing IP was not identified and the probe has failed complete.

Icons and coloring are used in the report to highlight data for example if an improvement exists for the prefix, or what was the current provider during probing or whether there’s a 100% loss. Filtering and sorting can be used to highlight and group the data into different specific categories.

Probes Today									
#	Timestamp	Prefix	Probed IP	AS Number	Probe state	Cogent		Serverius	
						Loss	Latency	Loss	Latency
1	2017-04-28 12:31:42	183.210.160.0/19	183.210.178.169	56046	✔	0 %	246 ms	0 %	207 ms
2	2017-04-28 12:31:39	184.201.0.0/16	66.1.64.244	3651	✔⚡	26 %	112 ms	62 %	119 ms
3	2017-04-28 12:31:39	117.240.131.0/24	218.248.235.241	9829	✔⚡	45 %	169 ms	63 %	177 ms
4	2017-04-28 12:31:39	221.181.128.0/17	221.181.252.173	56046	✔	0 %	208 ms	0 %	211 ms
5	2017-04-28 12:31:39	1.22.221.0/24		45528	⊗⚡	100 %	-	100 %	-
6	2017-04-28 12:31:39	180.215.40.0/21	103.18.67.18	131222	✔	0 %	153 ms	0 %	153 ms
7	2017-04-28 12:31:39	218.201.112.0/22	218.201.113.107	24444	✔⚡	0 %	306 ms	0 %	234 ms
8	2017-04-28 12:31:39	221.181.224.0/21	221.181.227.60	56046	✔	0 %	222 ms	0 %	211 ms
9	2017-04-28 12:31:39	99.205.0.0/16	68.28.249.49	3651	✔	83 %	124 ms	75 %	134 ms
10	2017-04-28 12:31:39	1.22.104.0/24	113.193.240.217	45528	✔	0 %	135 ms	0 %	140 ms
11	2017-04-28 12:31:39	123.136.152.0/24		45528	⊗⚡	100 %	-	100 %	-
12	2017-04-28 12:31:39	112.25.128.0/21	112.25.135.226	56046	✔⚡	0 %	245 ms	0 %	212 ms

Figure 3.4.31: Probes today

3.4.23 Monitors status

The “Monitors Status” report highlights the current status of IRP monitors per provider. For Internet Exchanges the report highlights all the peers with a tooltip with details.

Monitors Status					
Provider	Provider Status	PBR State	PBR Diag Hop	BGP Internal Monitor	BGP External Monitor
Cogent	Enabled	OK	-	Up	Up
Serverius	Enabled	OK	-	Disabled	Up
NLIX	Enabled	Unknown	-	Unknown	Unknown

Figure 3.4.32: Monitors status

The “Monitor history” report supplements Monitors status by highlighting the time, monitor details and the new state, facilitating the tracing of various issues on the network and in IRP configuration.

Historical monitors										
#	Timestamp	Provider	Next Hop	Type	IP Version	PBR State	PBR Observed Hop	PBR Diag Hop	BGP Internal Monitor State	BGP External Monitor State
61	2017-05-16 05:01:01	Serverius		pbrTransit	6	Fail	2a00:1ca8:ad::2	2a00:1ca8:ad::1/128		
62	2017-05-16 04:59:44	Serverius		pbrTransit	6	Fail	2a00:1ca8::2:20	2a00:1ca8:ad::1/128		
63	2017-05-16 04:42:23	AMS-IX	80.249.208.173	pbrPartial		Fail				
64	2017-05-16 04:32:46	AMS-IX	80.249.209.150	bgplntPartial					Ok	
65	2017-05-16 04:30:46	AMS-IX	80.249.211.239	bgplntPartial					Ok	
66	2017-05-16 04:07:52	AMS-IX	80.249.209.20	pbrPartial		Ok				
67	2017-05-16 03:14:25	AMS-IX	80.249.211.95	pbrPartial		Ok				
68	2017-05-16 01:23:52	AMS-IX	80.249.211.95	pbrPartial		Fail				
69	2017-05-16 01:10:53	Serverius		pbrTransit	6	Fail	2a00:1ca8:ad::2	2a00:1ca8:ad::1/128		
70	2017-05-16 01:09:36	Serverius		pbrTransit	6	Fail	2a00:1ca8::2:20	2a00:1ca8:ad::1/128		
71	2017-05-16 01:06:51	Serverius		pbrTransit	6	Fail	2a00:1ca8:ad::2	2a00:1ca8:ad::1/128		
72	2017-05-16 01:05:34	Serverius		pbrTransit	6	Fail	2a00:1ca8::2:20	2a00:1ca8:ad::1/128		
73	2017-05-16 00:18:32	Serverius		pbrTransit	6	Fail	2a00:1ca8:ad::2	2a00:1ca8:ad::1/128		
74	2017-05-16 00:17:16	Serverius		pbrTransit	6	Fail	2a00:1ca8::2:20	2a00:1ca8:ad::1/128		
75	2017-05-16 00:12:21	AMS-IX	80.249.211.95	pbrPartial		Ok				
76	2017-05-15 23:52:26	AMS-IX	80.249.209.150	bgplntPartial					Fail	

Figure 3.4.33: Monitor history

3.5 Graphs

The IRP graphs are visual representations of data found in the reports. They provide an opportunity to quickly identify problems in the network, as well as forecast periodic changes in the traffic patterns and network behavior. For each graph you can adjust the time period which you want to get the results for, and export them as PDF (see the [Sidebar](#) section).

i Unless otherwise noted, all graphs are plotted using 5 minute intervals.

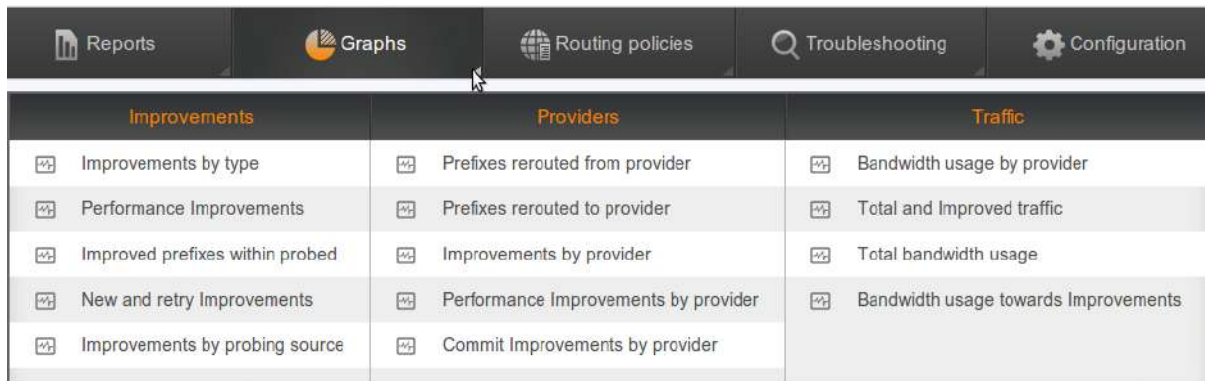


Figure 3.5.1: Graphs menu

3.5.1 Improvements by type

The “Improvements by type” graph shows the improvements made by IRP based upon: performance, cost, commit, and outage detection. Various problem patterns can be detected in compliance with the IRP activity shown in this graph. The graph provides information on the types of improvements that are more frequent for the network in the current state. Different patterns can be noticed depending on the improvement mode that is currently running.

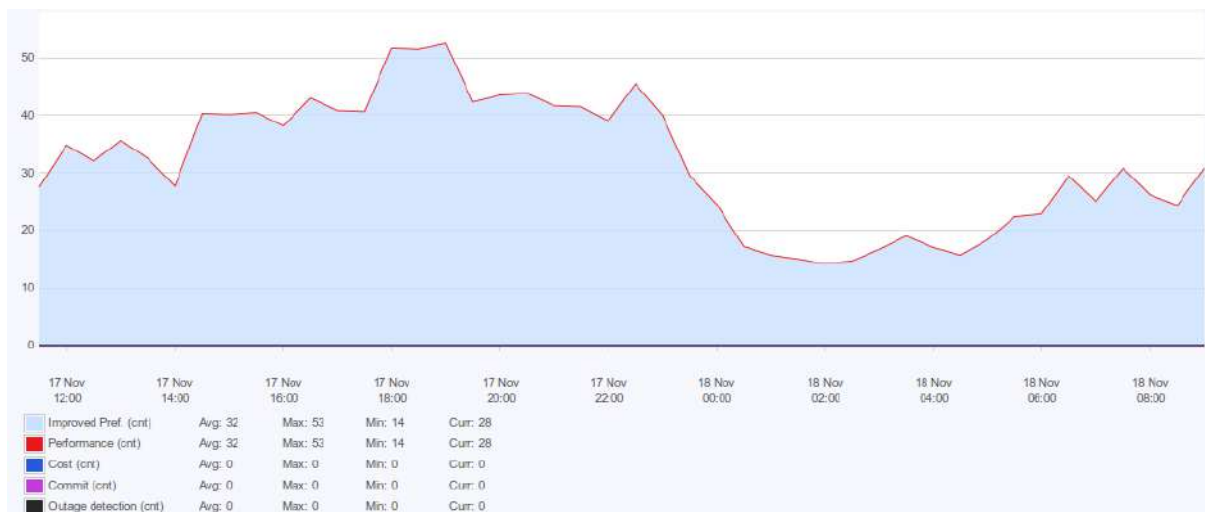


Figure 3.5.2: Improvements by type

3.5.2 Performance improvements

The “Performance improvements” graph displays the prefixes improved based upon a performance reason. By following this graph, the solved loss and latency issues, occurring in the network, can be monitored.

The graph also provides the average, maximum and minimum number of improvements for each of the improvement reasons during the selected time frame. The maximum peaks are indicators of a loss or latency problem in the network.

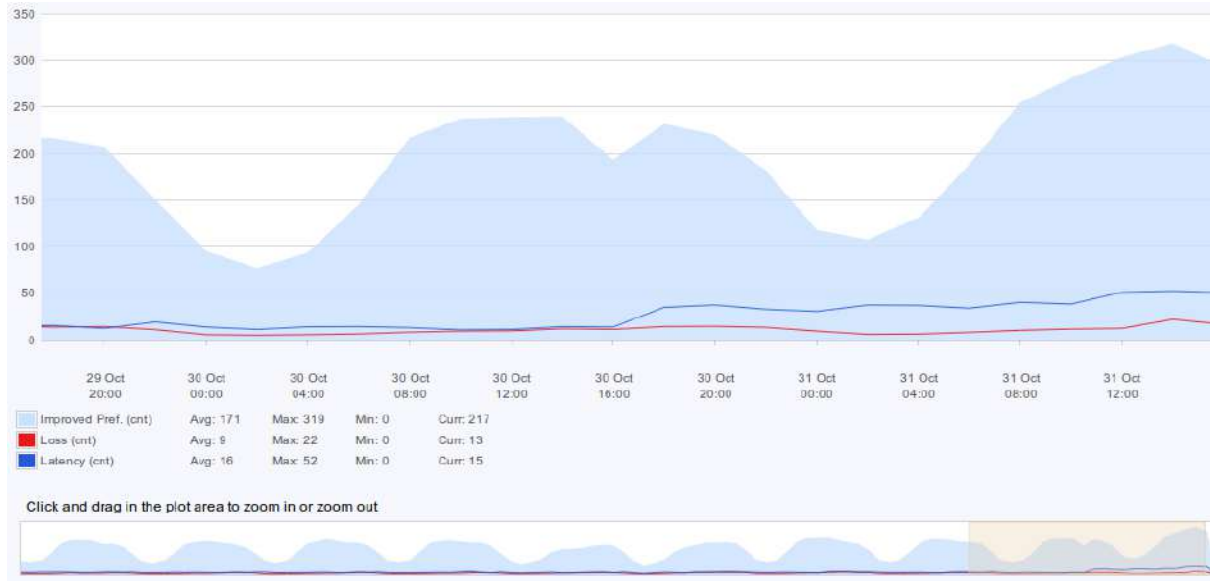


Figure 3.5.3: Performance improvements

3.5.3 Probed prefixes and Improvements

The "Probed prefixes and Improvements " graph shows the number of improvements (Loss/Latency) per time unit as reported to the total probed prefixes.



Figure 3.5.4: Probed prefixes and Improvements

3.5.4 New and retry improvements

After a particular improvement was made, IRP analyzes the path periodically. If the improvement is still valid it leaves it as it is. The "New and retry improvements" graph provides the number of improvements made for the first time as related to those, which were simply confirmed after re-probing.

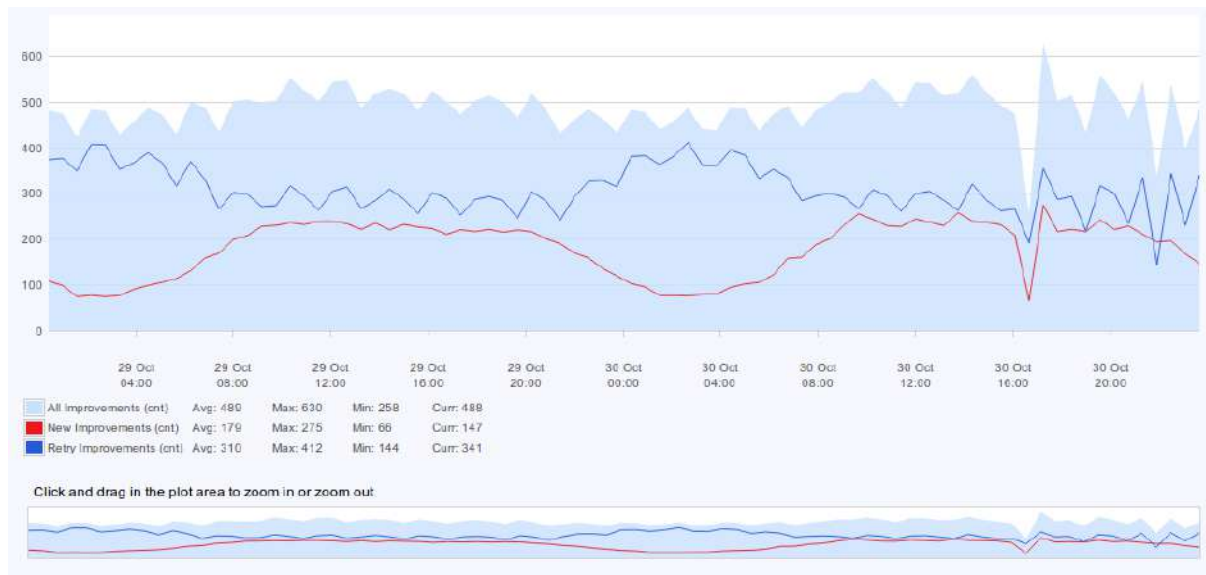


Figure 3.5.5: New and retry improvements

3.5.5 Improvements by probing source

The current graph displays the percentage of improvements that are differentiated by the probing source including Commit Control, Outage Detection, VIP Probing, Regular and Retry Probing.

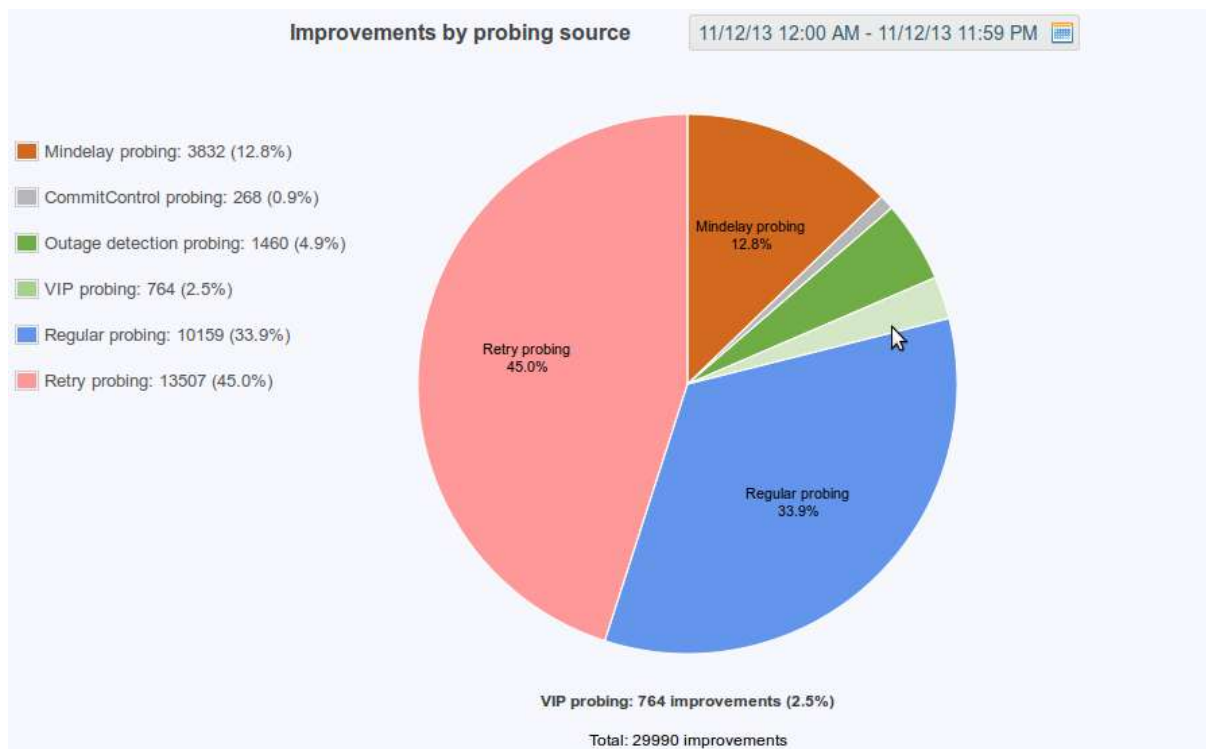


Figure 3.5.6: Improvements by probing source

3.5.6 Prefixes rerouted from provider

The current graph displays the number of prefixes that were rerouted from a specific provider in order to resolve performance or cost issues.

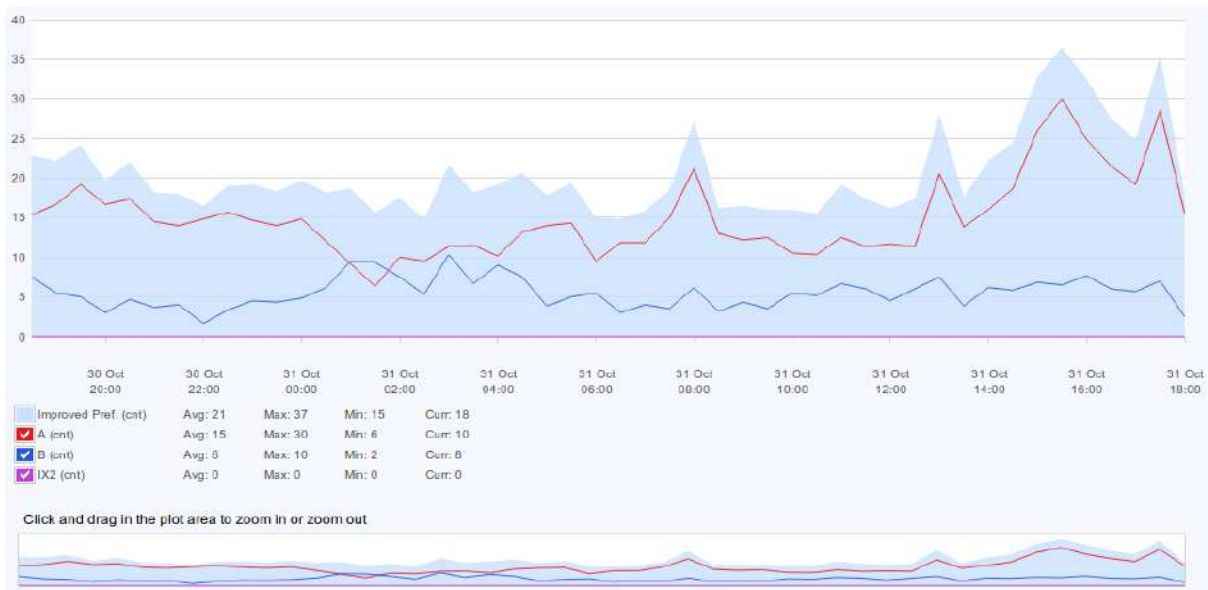


Figure 3.5.7: Prefixes rerouted from provider

3.5.7 Prefixes rerouted to provider

The graph displays the number of prefixes that were rerouted to a specific provider in order to resolve performance or cost issues.

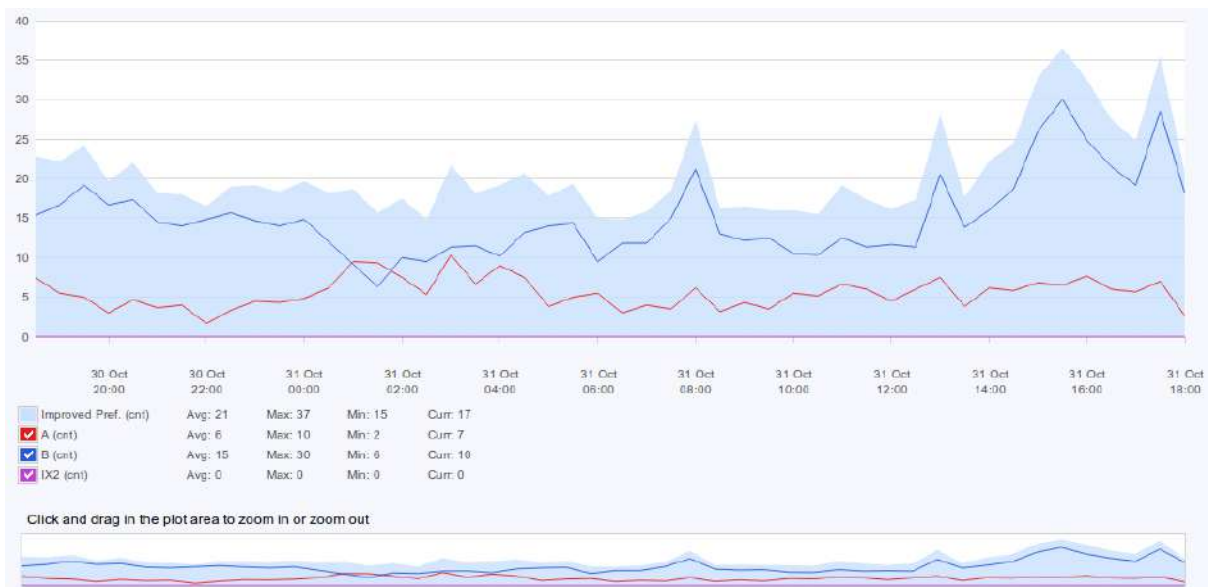


Figure 3.5.8: Prefixes rerouted to provider

3.5.8 Improvements by provider

The “Improvements by provider” graph shows all the improvements made by IRP for each of the providers. You can monitor the issues occurring in the network, which triggered IRP to reroute the traffic. You can compare the providers based on how they perform in the context of performance and cost. It also provides the average, maximum and minimum number of improvements per provider for each of the improvement reasons.

i Improvements by provider graph categorizes improvements by performance and cost. If cost mode is disabled this graph shows the same data as Prefixes rerouted to provider graph.

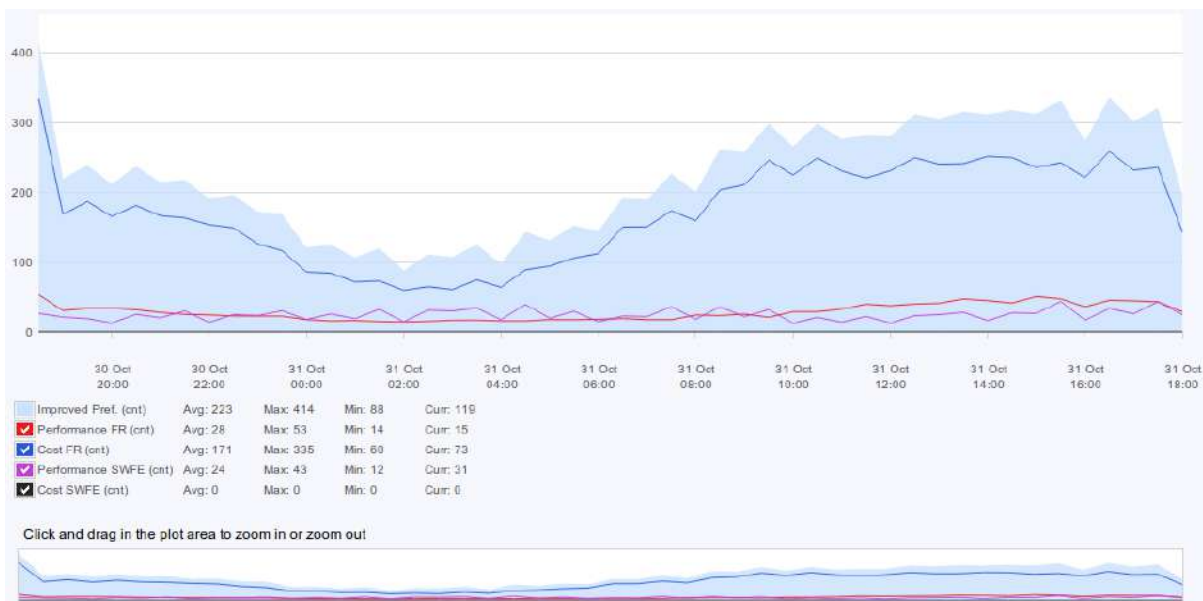


Figure 3.5.9: Improvements by provider

3.5.9 Performance improvements by provider

Similar to the previous graph, the “Performance improvements by provider” displays a graphical representation of historical values for all latency and loss-based improvements for each provider. The charts are displayed next to each other for easier comparison and identification of regularities. Keep in mind that specific provider lines can be removed from the graph. This way we are able to focus only on the interesting data.

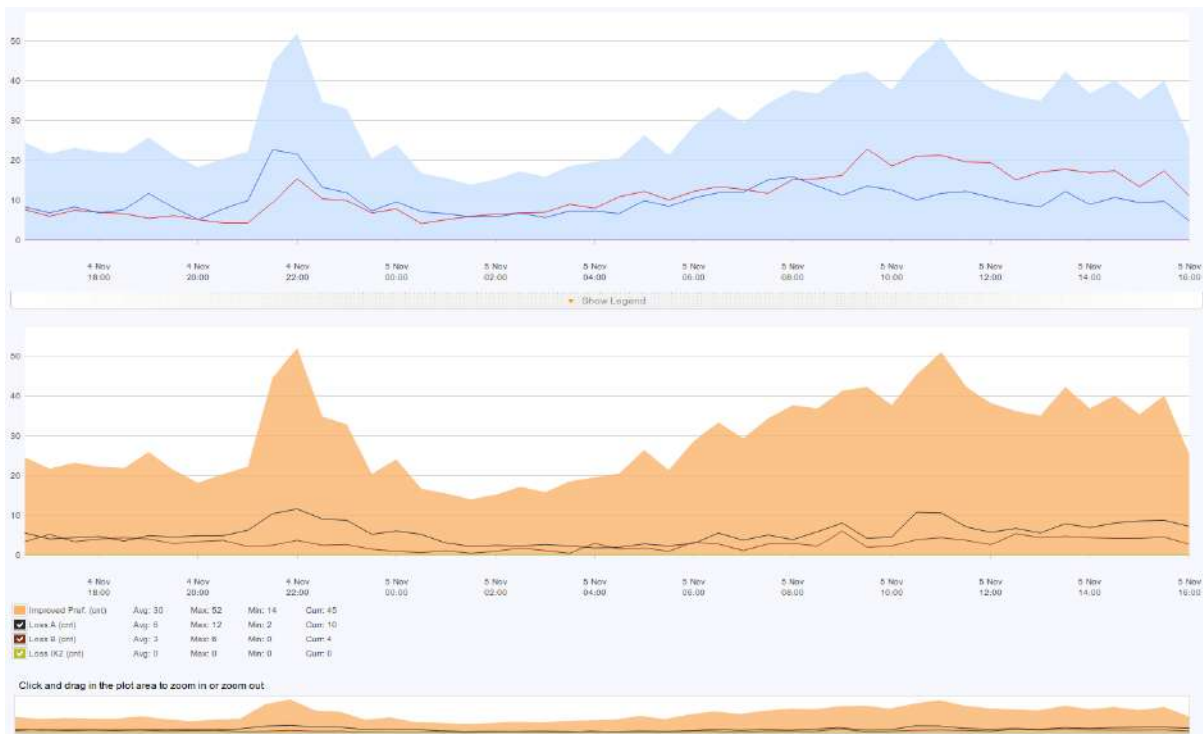


Figure 3.5.10: Performance improvements by provider

3.5.10 Commit improvements by provider(Not supported in IRP Lite)

If the Commit Control algorithm is enabled (see [Commit Control \(Not supported in IRP Lite\)](#)), this graph will display an overview of all the commit improvements that were enforced by IRP for a specific time period.

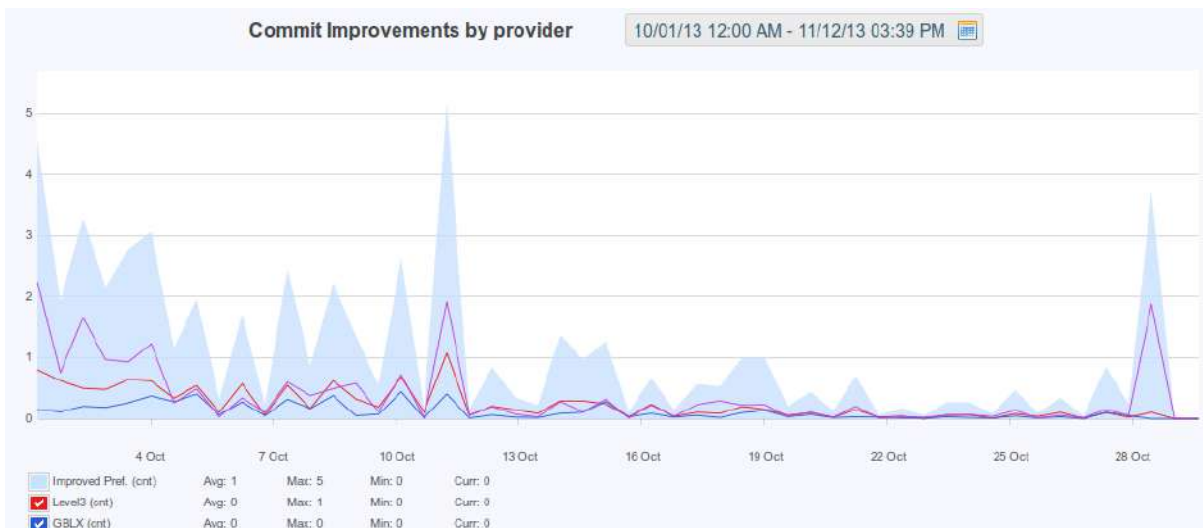


Figure 3.5.11: Commit improvements by provider

3.5.11 Total and Improved traffic

The Total and Improved Traffic graph allows you to see the positive impact of IRP on the network. It provides the amount of improved traffic, shown in blue color, as related to the total traffic, shown in green. Thus, you can monitor how much of the traffic is affected by IRP.



Figure 3.5.12: Total and improved traffic

3.5.12 Bandwidth usage by provider

The Bandwidth Usage graph shows the total usage for each of the providers. It allows you to compare the current traffic volume to the current 95th percentile and the commit 95th. The graph provides average, maximum and minimum traffic usage values for each of the providers during the selected time period.



Figure 3.5.13: Bandwidth usage by provider

3.5.13 Providers bandwidth usage

The Providers bandwidth usage graph combines all usage lines in a single chart. It allows cross-comparison between providers themselves. The chart includes options to hide/show providers of interest during selected time period. The top and bottom charts display outbound and inbound traffic accordingly.



Figure 3.5.14: Providers bandwidth usage

3.5.14 Bandwidth usage and improvements

Bandwidth usage and improvements chart superimposes two charts for easy analysis and comparison of how improvement counts correlate with bandwidth.



Figure 3.5.15: Bandwidth usage and improvements

3.5.15 Total bandwidth usage

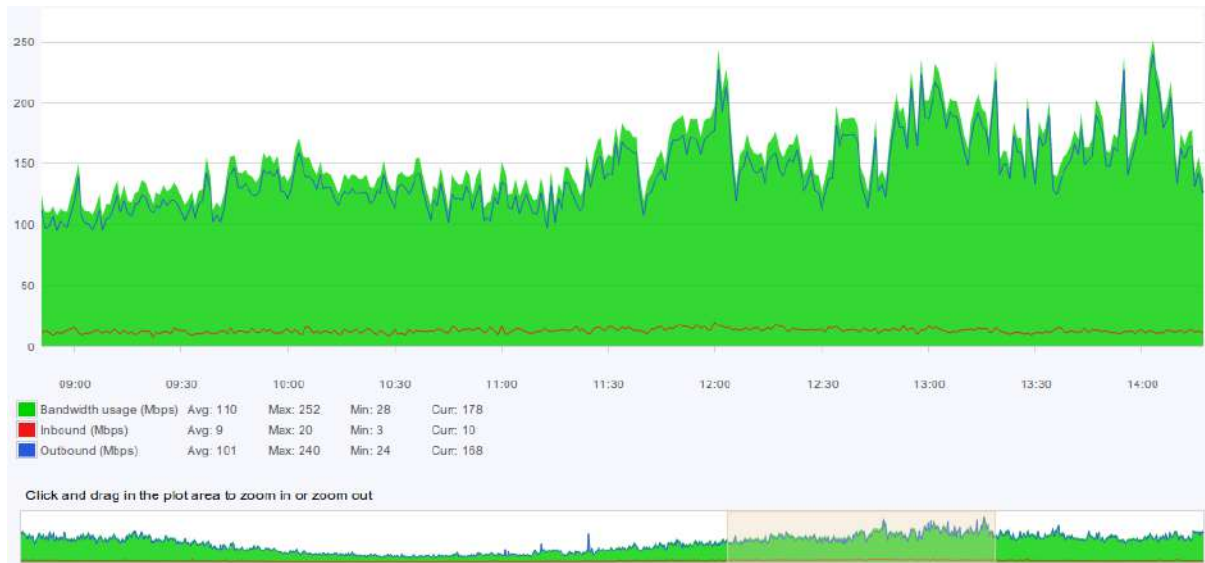


Figure 3.5.16: Total bandwidth usage

3.5.16 Inbound traffic distribution

The graph highlights distribution of inbound traffic across inbound prefixes and providers.

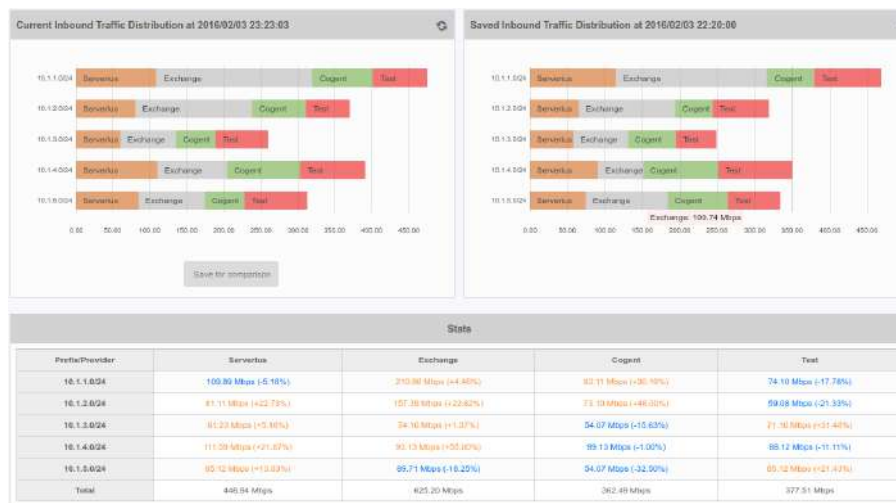


Figure 3.5.17: Inbound traffic distribution

Inbound traffic distribution report includes a feature to cross-compare current values with a saved inbound traffic distribution from recent past. To cross-compare saved inbound traffic distribution with freshly retrieved statistics, save and compare traffic shape changes and if they conform to expectations. Refer [Inbound optimization](#) (Not supported in IRP Lite) for details.

3.5.17 Provider performance history

Provider performance history uses a scatter chart to plot past provider performance data. Refer report [Provider performance](#) for a different way to display this data. While Provider daily performance report for a longer time period averages the individual metrics this chart allows spotting consistent or hectic performance for providers.

i Individual data points represent past daily averages. IRP needs to run for a few days for this chart to have any meaningful data to plot.

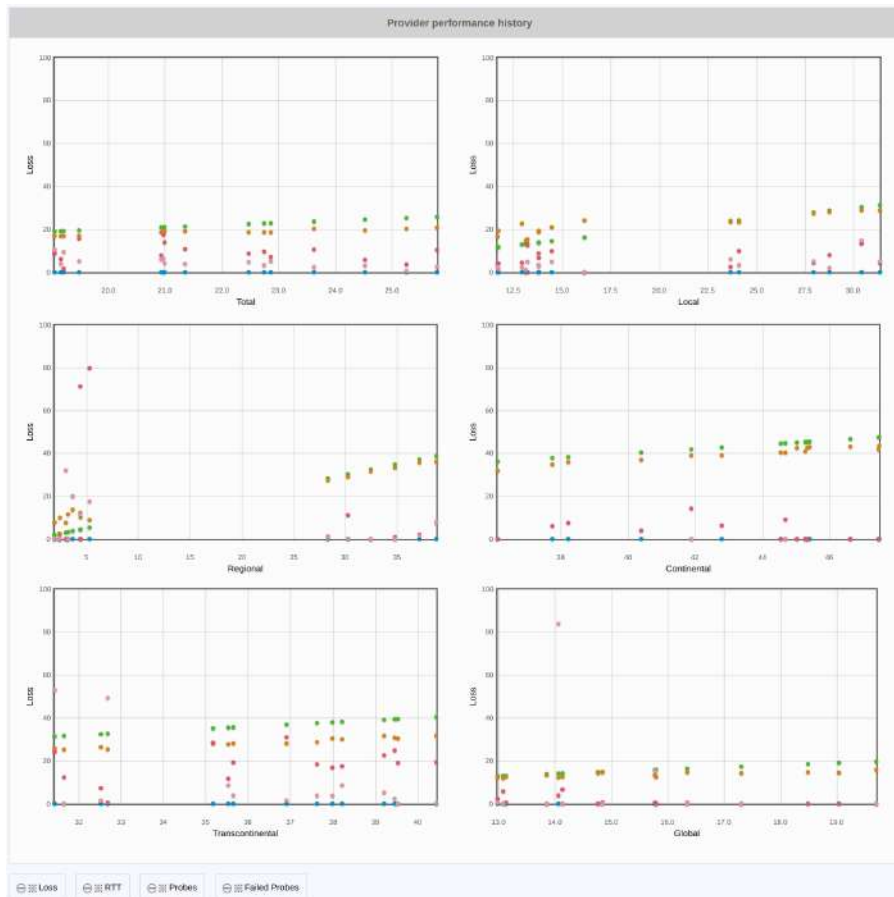


Figure 3.5.18: Provider performance history

Refer legend on the left-side sidebar for color points assigned to different providers. The same 'Best' category as in report [Provider performance](#) is distinguished to plot the absolute best loss or RTT observed during IRP probing. Designated charts plot local, regional, continental, transcontinental and worldwide data. The totals including all data points is available too.

3.6 Routing Policies

i Maximum 3 routing policies are allowed in IRP Lite

To see the full list of routing policies configured in the system, select the Routing Policies option from the main menu and click on the Routing Policies option in the submenu. The list displays all the enabled and disabled routing policies configured in IRP.

Priority	Prefix/ASN	Note	ASN	AS Name	VIP	Policy Type	Providers	Last probed	Next probe	From	To	Details	Actions
1	66.87.70.0/24 (1 prefixes)		10607	Sprint Personal Communications Systems		Allow through	NLIX, Serverius	15 minutes ago	-				
2	5.101.41.0/24 (1 prefixes)		44050	Heranburg Internet Network Ltd		Allow through	Cogent, Serverius	a few seconds ago	In next 4 minutes				
2	2.2.2.0/21 (1 prefixes)		3215	Orange		Static route	Cogent	-	-	Cogent	Cogent	Global Static route through Cogent[e]	
7	17623 (107 prefixes)		17623	China Unicom Shenzhen network		Allow through	AMS-IX, Cogent, NLIX, Serverius	17 minutes ago	In next 4 hours				
7	27.38.0.0/24		17623	China Unicom Shenzhen network		Allow through	AMS-IX, Cogent, NLIX, Serverius	17 minutes ago	In next 4 hours				
7	27.38.8.0/24		17623	China Unicom Shenzhen network		Allow through	AMS-IX, Cogent, NLIX, Serverius	17 minutes ago	In next 4 hours				

Figure 3.6.1: Routing Policies

As shown in the screenshot above, the list provides the following details:

- Priority of a policy.

i Priority specifies what policy to use in case of overlapping prefixes as might be the case of a large aggregate prefix policy extending over an ASN policy

- The country, prefix or ASN to which the policy applies.
- A note to describe the policy.
- The specific ASN and AS Name of the prefix to which the policy was applied.
- The status of the VIP reprobng (enabled/disabled).
- The type of the policy (allow/deny/static).
- Providers which are affected by the policy.
- Time passed after the last probe was performed towards this prefix.
- The time left before the next scheduled probe towards this prefix.
- From and To details of an existing improvement relating to this policy.
- The available actions for the policy. These include:

- On-demand probing
- Editing the routing policy
- Enabling the routing policy
- Disabling the routing policy

You can collapse all the policies by clicking the "Collapse All" button and then expand them back by clicking the "Expand all" button. You can use the "Remove all " button to delete all the policies. You can also select a set of policies and use the "Remove selected" button to delete them.

A new routing policy can be added by pressing the "Add routing policy" button. The following window will open up:

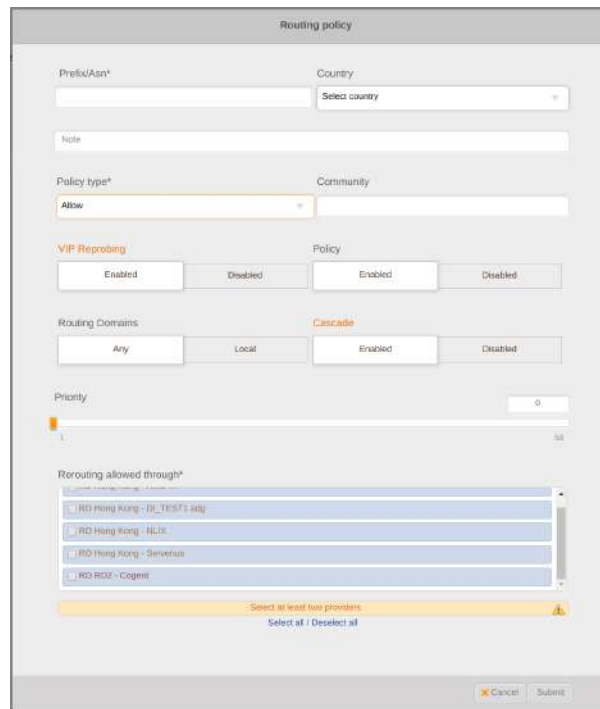


Figure 3.6.2: Routing policy window

The following parameters should be configured to add a routing policy:

- Prefix/ASN: The prefix/ASN to which the policy is applied
- Country: The country to which the policy is applied. Such a policy applies to all prefixes known to be located in the country.
- Note: A note to describe the policy
- Policy Type: The type of the policy (allow/deny/static)
- A community to mark improvements
- A flag to set if policy is VIP
- The policy status (enabled/disabled)
- A flag to indicate if an ASN policy should propagate to downstream ASN.
- A flag indicating if a global improvement allowed.
- Priority slider or text-box will set a policy’s priority.
- The Providers for which the policy should be respected
- Cascade flag for ASN policies that indicate if the policy should be enforced only for this AS or also for downstream AS. If cascading is enabled it ensures that the policy is enforced for all traffic that will eventually be routed through this AS.

Refer policy configuration parameters [Routing Policies settings](#) for details.

To see a list of Static Route policies, select Static Routes in the Routing Policies submenu. Alternatively the filter on the left side-bar can be used. A list like the one below will be displayed.

Priority	Prefix/ASN	Note	ASN	AS Name	VIP	Policy Type	Providers	Last prebid	Next probe	From	To	Details	Actions
2	2.2.2.0/24 (1 prefix)		3215	Orange		Static route	Cogent	--	--	Cogent	Cogent	Global Static route through Cogent(4)	[i] [d] [x]

Figure 3.6.3: Static Routes

The list structure and the actions available are similar to those found in the Routing Policies list. Here you can also add a new Static Route by using the "Add static route" button.

You can select the VIP option in the Routing Policies submenu to access a list of all the policies with VIP reprobing enabled.

Priority	Prefix/ASN	Note	ASN	AS Name	VIP	Policy Type	Providers	Last probed	Next probe	From	To	Details	Actions
1	66.67.70.0/24 (1 prefixes)		10507	Sprint Personal Communications Systems		Allow through	NLIX, Serverius	3 minutes ago	in next 2 minutes				
2	5.101.41.0/24 (1 prefixes)		44050	Petersburg Internet Network Ltd.		Allow through	Cogent, Serverius	4 minutes ago	in next a minute				

Figure 3.6.4: VIP policies

Here you can also enable VIP reprobing for a new prefix/ASN by using the "Add VIP" button.

3.7 Flowspec policies

To review Flowspec policies, select the Routing Policies option from the main menu and click on the Flowspec Policies option in the submenu.

Flowspec must be enabled for Bgpd to announce them.

The list displays all the enabled and disabled Flowspec policies configured in IRP.

Note	Source prefix	Source port	Destination prefix	Destination port	Rule action	Redirect IP	Protocol	Provider	Rate (Mbps)	Actions
Throttle bit torrent		6881-6889		6881-6889	Throttle				100	
		4450			Throttle		TCP		100	
	185.34.222.0/24				Redirect IP	1.2.3.4				
	185.34.200.0/24				Redirect			Cogent		
Drop IPv6 from Oz	2a14:5ec0::431:96				Drop					

Figure 3.7.1: Flowspec policies

As shown in the screenshot above, the list highlights:

- A note to describe the policy.
- A source ASN/prefix and port(s) for matching packets.
- A destination prefix and port(s) for matching packets.
- DSCP traffic classification value.
- The policy action (Throttle, Drop, Redirect or Redirect IP).
- Redirect IP or Provider for redirect policies.
- Protocols of matching packets, for example TCP, UDP or ICMP.
- Rate limits in Mbps for Throttle policies.

Flowspec policies like routing policies can be edited, enabled, disabled or deleted altogether.

A Flowspec policy is added by clicking on the designated button and specifying the following:

Figure 3.7.2: Flowspec policy popup

The following parameters should be configured to add a routing policy:

- Notes to describe the policy
- Status to set if policy is Enabled or Disabled
- Source ASN/Prefix/Port(s): The source ASN/prefix and port(s) of the IP packets that match. A prefix in CIDR notation or a single IP address should be provided. Multiple valid TCP/UDP ports can be provided as well as port ranges.

i One Flowspec rule is created for each of the prefixes in the designated AS. ASN Flowspec policies can be setup only on the source of IP packets.

- Ensure that routers are capable of processing a large number of Flowspec rules before setting policies for an AS consisting of a very large number of network segments (prefixes).

- Destination Prefix/Port: The destination prefix/port attribute of the IP packets that match. Same rules as for Source Prefix/Port(s) apply.

- Prefixes MUST not equal or include probing IP addresses. Refer to for example `global.master_probing_interface`, `peer.X.ipv4.master_probing`, `peer.X.ipv6.master_probing`.

- Protocols: Packet protocols that match the policy. Can be any combination of TCP, UDP and ICMP.
- Policy Type: The type of the policy (Throttle, Drop or Redirect)

- Provider specifies one of the provider identifiers where traffic will be redirected. The provider is set only for Redirect policies.

i In order for Redirect policies to be implemented on routers a community value must be assigned to them and VRF must be configured on the router itself. Consult Noction support for details.

- Rate limits the allowed bandwidth usage for matching traffic. The value is set only for Throttling policies. The rate specifies a number in the range 1-4200 Mbps.

i In case the provided Flowspec policy attributes are incomplete or invalid on attempting to submit it a warning will be raised similarly as depicted in the popup.

For further details refer Flowspec configuration parameters for example: `global.flowspec`, `core.flowspec.max`, `core.flowspec.max_ipv6`, `bgpd.peer.X.flowspec`, `peer.X.flowspec.ipv4.redirect_community`, `peer.X.flowspec.ipv6.redirect_`

3.8 Throttling excessive bandwidth use with Flowspec

Throttling excessive bandwidth use with Flowspec policies use the same Flowspec policies reports. Select the Routing Policies option from the main menu and click on the Flowspec Policies option in the submenu.

i Flowspec must be enabled for Bgpd to announce them.

The list displays besides all the other Flowspec policies the automatic throttling rules marked with Throttle at. These rules cannot be edited.

Note	Source	Source port(s)	Destination	Destination port(s)	Rule action	Redirect IP	Protocol	Provider	Rate (Mbps)	Announced prefixes	Actions
	60238	1.2.4.5.6.7	185.34.200.0/24		Throttle		ICMP, TCP, UDP		100	0	[edit] [delete]
	18302				Drop				0	0	[edit] [delete]
Overusage			185.34.201.0/24		Throttle at				101	0	[delete]
Overusage			185.34.200.0/24		Throttle at				45	0	[delete]
Overusage			185.34.202.0/24		Throttle at				300	0	[delete]
Overusage			185.34.212.0/24		Throttle at				15	0	[delete]

Figure 3.8.1: Throttling excessive bandwidth use with Flowspec policies

Each of the rules sets a specific destination prefix to control outbound bandwidth use and sets an Mbps rate based on past average bandwidth use.

Refer [Core configuration](#) for configuration details.

3.9 Maintenance windows

To review Maintenance windows, navigate to Policies option in main menu and click on Maintenance Window option. The list displays all the current and future maintenance windows configured in IRP. Refer [Maintenance windows](#) for details about maintenance windows.

	Begin	End	Providers	Withdraw Improvements	Seconds to Unload	Seconds to Prepend	Actions	
●	2016-09-14 16:08	2016-09-14 16:30	AMS-IX	Yes	300	0	[Edit] [Delete]	[Add]
●	2016-09-15 00:00	2016-09-28 00:00	Cogent	Yes	300	150	[Edit] [Delete]	[Add]
●	2016-09-23 10:48	2016-09-30 10:48	NL-IX	Yes	0	0	[Edit] [Delete]	[Add]

Figure 3.9.1: Maintenance windows

As shown in the screen-shot above, the list highlights:

- Current color-coded status of the maintenance window where Red and Orange depict unloading and actual maintenance phases while Blue and Green depict future planned windows and past finished windows correspondingly.
- Begin and End date-times of maintenance windows.
- Provider or router under maintenance.
- Withdraw Improvement option enables or disables withdrawal of existing Outbound Improvements to that provider.
- Seconds to Unload defines time interval in seconds prior to beginning of the maintenance window when IRP starts to unload outbound traffic from provider.

⚠ If Seconds to Unload/Seconds to Prepend time intervals set to zero IRP does not perform corresponding action.

i When seconds to unload is specified it should be at least 5 minutes (300 seconds) long with larger time intervals allowing IRP more time to assess and reroute traffic flowing through provider with scheduled maintenance window. Very short time intervals might not have the desired effect since usually an IRP optimization cycle takes 2-3 minutes to finish.

- Seconds to Prepend defines time interval in seconds prior to beginning of the maintenance window when IRP announces all inbound prefixes with maximum prepends to provider in order to deflect inbound traffic towards other providers.

Maintenance windows can be added, edited or deleted using the designated action buttons.

A maintenance window is added by specifying the following:

Figure 3.9.2: Maintenance window popup

The parameters mentioned above including the option to choose a single provider or all providers on a router are provided for Maintenance window configuration.

Maintenance windows are also highlighted on IRP Frontend sidebar with the title of the sidebar box highlighting the status of the next maintenance window so that it is visible even if the contents of the sidebar is collapsed.

	Begin	End	Provider
●	2016-09-14 16:08	2016-09-14 16:10	AMS-IX
●	2016-09-15 00:00	2016-09-28 00:00	Cogent
●	2016-09-23 10:48	2016-09-30 10:48	NL-IX

● - Planned ● - Unloading
 ● - Maintenance ● - Finished

Figure 3.9.3: Maintenance window sidebar

3.10 Events

The system events are diagnostic messages that were sent by the IRP Components (See also: [IRP Components](#)), as well as notifications caused by the BGP monitoring algorithm (See also: [BGP Monitoring](#)). The events are displayed in the status bar located at the bottom of the system frontend (See [Frontend structure](#)). Several or all the events can be marked as “Read”, so they will no longer be displayed in the status bar.

Mark all as read

<input type="checkbox"/>	#	Timestamp	Module	Event type	Notification message
<input type="checkbox"/>	1	10/30/13 18:40:44	BGPd	Error	BGP session s019/IPv4 (178.236.177.1 AS 48232) outgoing connection disconnected by peer
<input type="checkbox"/>	2	10/30/13 18:40:43	BGPd	Error	BGP session s019/IPv4 (178.236.177.1 AS 48232) disconnected by Network layer request (TCP reset...
<input type="checkbox"/>	3	10/30/13 18:22:50	BGPd	Error	BGP session s019/IPv4 (178.236.177.1 AS 48232) outgoing connection disconnected by peer
<input type="checkbox"/>	4	10/30/13 18:13:00	BGPd	Error	ExternalMon: Peer FR[2] entering FAIL state
<input type="checkbox"/>	5	10/30/13 18:13:00	BGPd	Error	ExternalMon: Peer SWFE[3] entering FAIL state

1

1 - 5 of 5 results found

Figure 3.10.1: System events

3.11 Troubleshooting

The IRP Frontend provides several troubleshooting tools, that can offer you quick information on the specific remote networks.

3.11.1 Traceroute

A traceroute to a specific IP address or hostname can be run from the Frontend. Results will be displayed in the graphical and detailed formats, as in the examples below. The “Allow automatic improvement” checkbox can also be used for automatic improvement of the provided IP address.



Figure 3.11.1: Traceroute

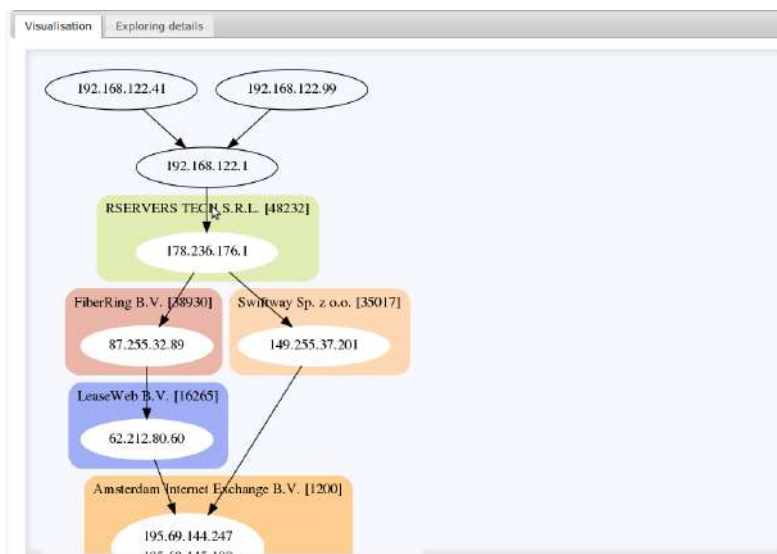


Figure 3.11.2: Traceroute results

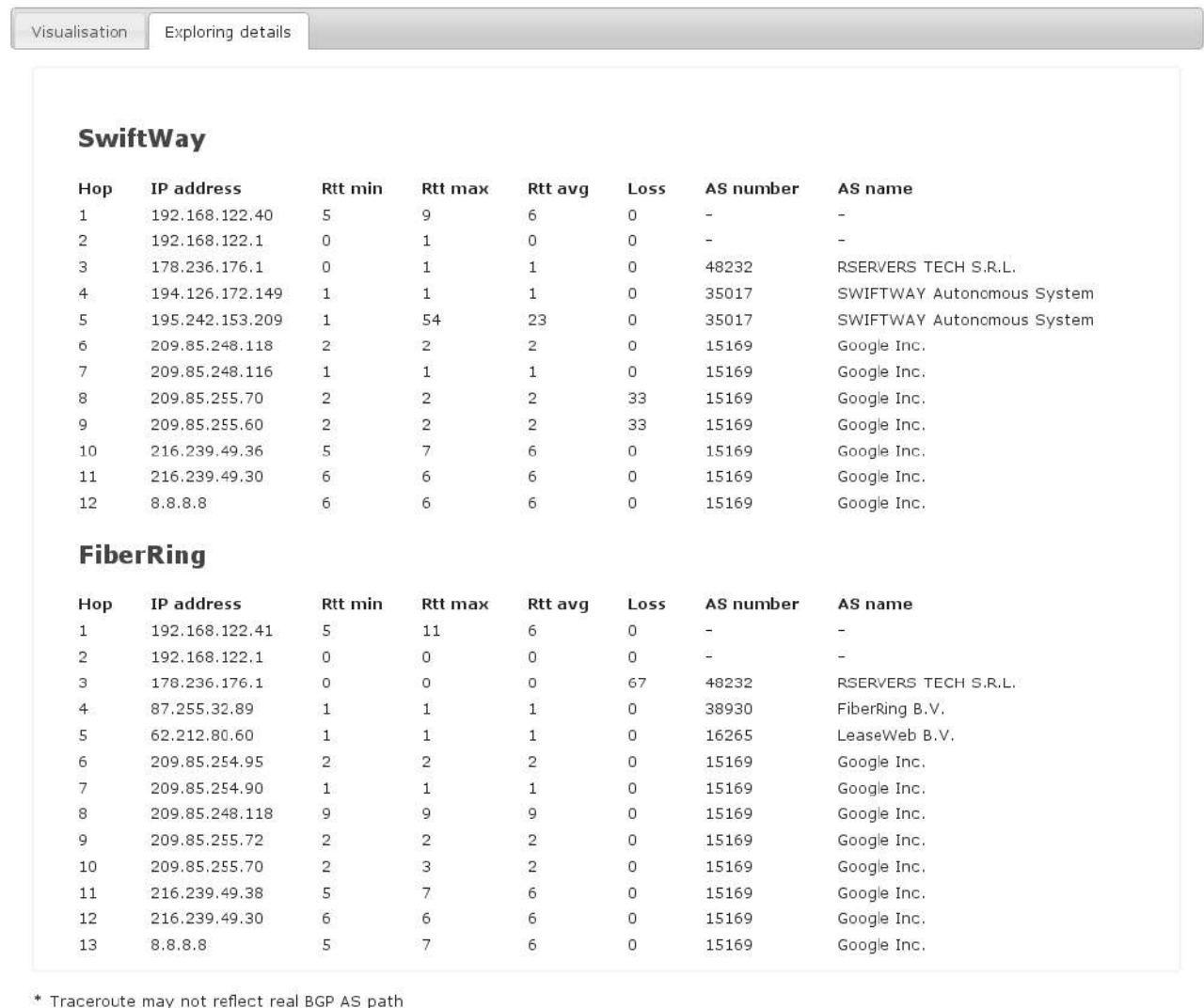


Figure 3.11.3: Traceroute exploring details results

3.11.2 Looking glass

A ping, traceroute or MTR can be run from the Frontend via the default route or via a specific provider.

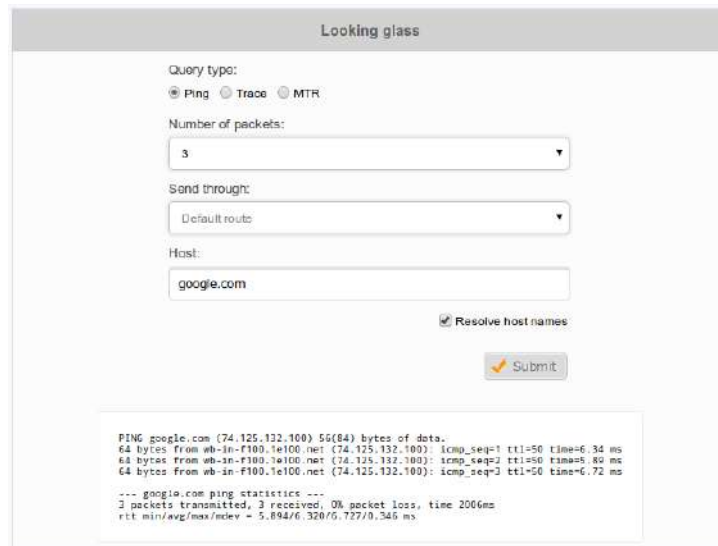


Figure 3.11.4: Looking glass

3.11.3 Whois

The “Whois” tool can query IP addresses, hostnames, network prefixes, or AS Numbers. The following valid values can be entered in the search box:

- Hostname: fully qualified domain name, e.g: “domain.com”
- IP: valid IP address, e.g: “10.20.30.40”
- IP Block: any valid prefix, in CIDR format, e.g: “10.20.30.40/24”
- ASN: valid ASN, using the following format: “AS1234”



Figure 3.11.5: Whois

3.11.4 Prefix probing

IRP features a manual prefix probing function. One can manually submit a specific IP address for probing and optimization by the system. Valid hostnames and IP addresses can be entered. The probing

results will be displayed on the same page. Please note that IRP probes the exact entered IP address. In case the IP address doesn't respond, IRP runs the indirect probing process and optimizes the whole prefix based on the indirect probing results.

Prefix probing allows to automatically or manually choose the best path for a particular IP address or prefix.

In order to check what are the possible paths for a particular prefix, enter the IP address or prefix into the field. If you prefer the prefix to be improved automatically, tick the check box "Improve automatically" and press the "Submit for probing" button. Otherwise, just press the "Submit for probing" button and wait until the system returns the probing results.



Figure 3.11.6: Prefix Probing

The system returns the probing results by displaying the loss, latency and cost values for each of the providers.

Prefix 8.8.0.0/15 probing is being performed...

Results for: 58.131.192.0/18

Global	Local	RD	ISP	Loss	Latency	Last improved	RD Latency
<input type="radio"/>	<input type="radio"/>	1	Cogent	8%	332 ms	2017-07-06 09:49:19	-
<input checked="" type="radio"/>	<input type="radio"/>	1	Serverius	0%	234 ms	2017-07-06 10:03:38	-

Recommended route: Current route: Current + Recommended route: Cancel Add a custom policy Add static route Reroute

Warning! Destination prefix is unreachable.

Results for: 1.2.3.4/32

Global	Local	RD	ISP	Loss	Latency	Last improved	RD Latency
<input type="radio"/>	<input type="radio"/>	1	Cogent	100%	unknown	-	-
<input checked="" type="radio"/>	<input type="radio"/>	1	Serverius	100%	unknown	-	-

Recommended route: Current route: Current + Recommended route: Cancel Add a custom policy Add static route Reroute

Results for: 108.120.0.0/17

Global	Local	RD	ISP	Loss	Latency	Last improved	RD Latency
<input checked="" type="radio"/>	<input type="radio"/>	1	Cogent	0%	402 ms	2017-07-06 08:59:56	-
<input type="radio"/>	<input checked="" type="radio"/>	1	Serverius	0%	243 ms	2017-07-06 10:17:10	-

Recommended route: Current route: Current + Recommended route: Cancel Add a custom policy Add static route Reroute

Figure 3.11.7: Prefix probing results

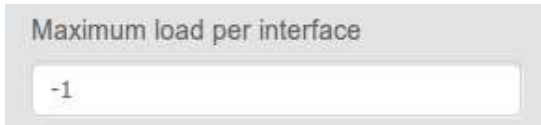
After the probing results are received, choose the path you consider the best one. Note, that IRP highlights the current and the best path. Use the "Reroute" button to redirect the traffic through the selected provider. You can also set the selected provider as static for the probed prefix by pressing the "Add static route" button or apply a specific routing policy to the probed prefix by clicking the "Add a custom policy" button.

3.12 Configuration editor

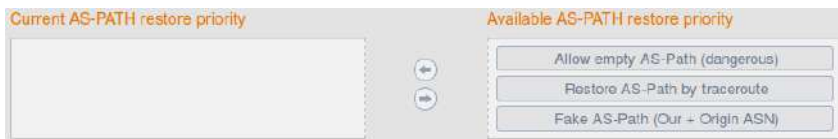
Starting with IRP version 1.7, all system parameters can be modified from the Frontend, using the “Configuration” menu.

Depending on the parameter type, several control types are available:

- Text boxes, which can hold IP addresses, provider names and numerical values



- Auto-complete boxes, used for parameters that can take a limited set of values



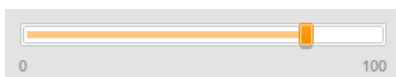
- Auto-complete list boxes, usually containing a list of IPs / prefixes (in CIDR format) / ASNs



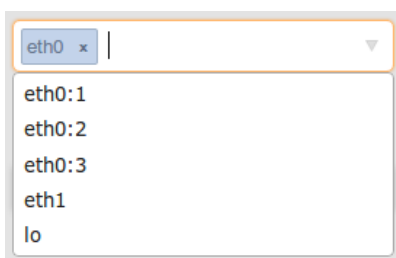
- Buttons



- Sliders, used to select a value from a predefined range



- Drop-down lists



- Toggle buttons (used for specific algorithms and BGP peers)



- Provider control buttons, can be used to suspend, resume, stop, or remove a provider



3.12.1 Global configuration

Global settings can be adjusted from the Frontend, by using the “Configuration” menu.

Please refer to [Global and Core Configuration](#) for a detailed parameters description.

Several global parameters can be configured by selecting their desired values and clicking on the “Submit” button.

Available parameters:

- **Improvement mode** (`global.improve_mode`)
- **Management interface** (`global.master_management_interface`)
- **BGP mode** (Intrusive / Non-intrusive, see `global.nonintrusive_bgp`)
- **Probing interface(s)**. See `global.master_probing_interface`
- **Prefix aggregation**. If this parameter is enabled, the system operates at aggregate level, up to the **Maximum aggregate mask** (see: `global.aggregate`, `global.agg_ipv4_max`, `global.agg_ipv6_max`)

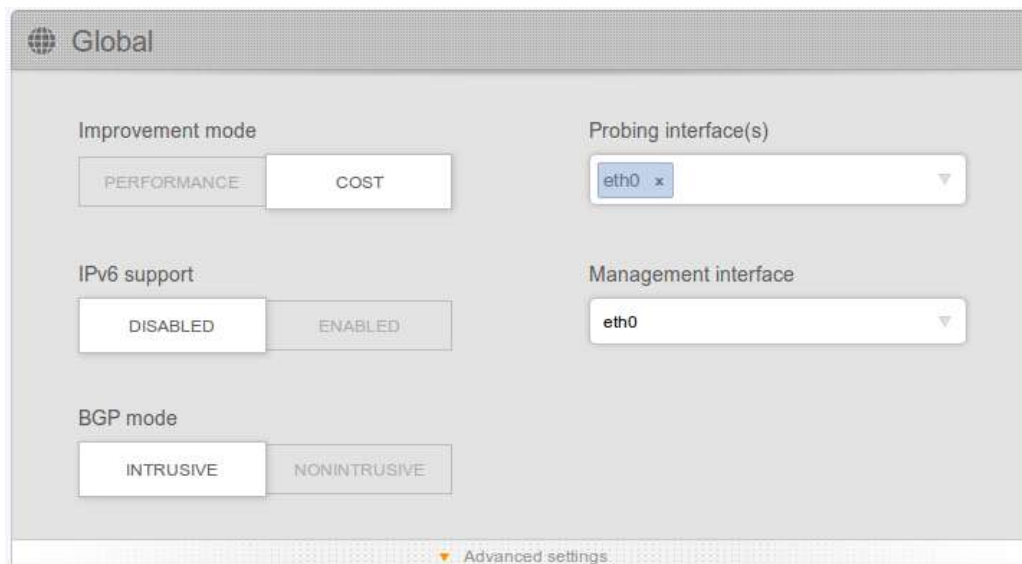


Figure 3.12.1: Configuration editor: Global settings

The list of ignored networks can be modified on the same page. The options allow listing as appropriate:

- prefixes(`global.ignorednets`),
- ASNs (`global.ignored.asn`),
- BGP Community attributes that mark prefixes/routes to ignore (`global.ignored_communities`).

Setting prefixes and ASNs is possible either manually, or by importing a text file containing one IP/network in CIDR format or ASN per line.

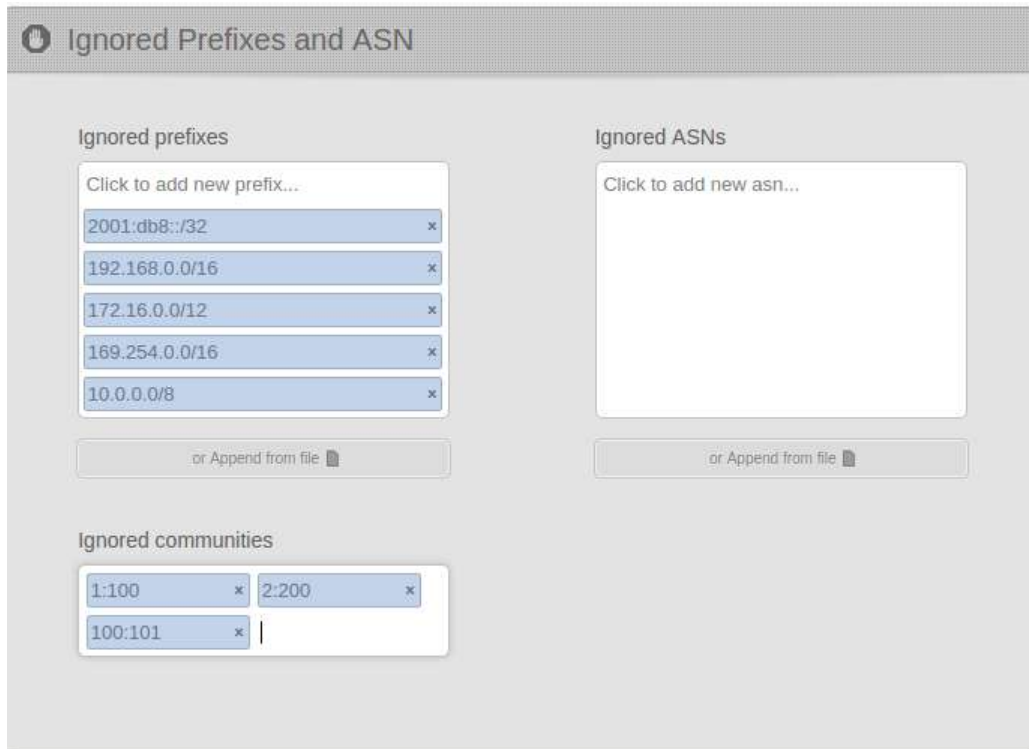


Figure 3.12.2: Configuration editor: Ignored prefixes

Failover functionality is switched on or off (`global.failover`) as part of Global configuration too.

i Failover license is required for Failover tab to be available.

Failover configuration covers the following parameters:

- Failover timer - time period in seconds during which master is considered alive and before slave becomes active if there are no announcements from master (`global.failover_timer_fail`).
- Failback timer - time period before slave node will release control back to master (`global.failover_timer_failback`). This is to protect from failover flapping between master and slave in case of master instability.
- Slave IPv4 address - indicates to master what is the slave node (`global.failover_slave.ip`). Master uses this address to setup SSH connections and sync its own configuration files.
- Slave SSH port - indicates the TCP port number for SSH on the slave node (`global.failover_slave.port`). Master uses this port to setup SSH connections and sync its own configuration files.



The image shows a configuration window titled "Failover". At the top right, there are two buttons: "ON" (highlighted) and "OFF". Below the title bar, there are four main configuration sections:

- Failover timer (s):** A numeric input field containing "30" and a slider below it. The slider has a range from 30 to 3600, with an orange indicator at the 30 mark.
- Failback timer (s):** A numeric input field containing "30" and a slider below it. The slider has a range from 30 to 3600, with an orange indicator at the 30 mark.
- Slave IPv4 address:** A text input field containing "192.168.123.77".
- Slave SSH port:** A text input field containing "22".

At the bottom of the window, there is a button with a triangle icon and the text "Hide advanced settings".

Figure 3.12.3: Configuration editor: Global failover

3.12.2 BGP and Routers configuration

BGP daemon-related parameters can be configured in the “Configuration→BGP” section.

- Bgpd parameters (See also: [Bgpd Configuration](#), [Bgpd settings](#)):
 - Split announced improvements to preserve original route attributes (`bgpd.updates.split` (Not supported in IRP Lite))
 - Remove prefixes (improvements) on next-hop update (`bgpd.improvements.remove.next_hop_eq`)
 - Strip non-IRP communities for improvements (`bgpd.improvements.strip_non_irp_communities`)
 - Remove prefixes (improvements) on aggregate withdrawal (`bgpd.improvements.remove.withdrawn`)
 - Re-probing on new (improved) or old route changes if BMP is available (`bgpd.retry_probing.new.bmp_path_change` / `bgpd.retry_probing.old.bmp_path_change`)
- BGP monitoring settings can be found under advanced settings and include but is not limited to (See also: [BGP Monitoring](#)):

i BGP monitor can be enabled/disabled. This option is useful during DoS/DDoS attacks.

- Guard time (`bgpd.mon.guardtime`)
- Holdtime (`bgpd.mon.holdtime`)
- Keepalive (`bgpd.mon.keepalive`)
- Flapping protection (`bgpd.mon.internal.flap_guardtime`)

Figure 3.12.4: Configuration editor: BGP settings

BGP routers can be added, removed or modified on the same page.

A new BGP neighbor can be added using the “Add Router” button. Several parameters must be configured:

- General settings:
 - Router name
 - ASN to be used for the iBGP session (`bgpd.peer.X.as`)
 - Improvement local-pref (`bgpd.peer.X.master_localpref`)
 - Local IP address (`bgpd.peer.X.master_our_ip`, `bgpd.peer.X.master_our_ipv6`)
 - Neighbor IP address, usually the router’s IP (`bgpd.peer.X.master_peer_ip`, `bgpd.peer.X.master_peer_ipv6`, `bgpd.peer.X.slave_peer_ip`, `bgpd.peer.X.slave_peer_ipv6`)

The screenshot shows a window titled "Add Router" with a close button in the top right corner. Below the title bar are three tabs: "General", "Advanced", and "Inbound", with "General" selected. The main area contains five input fields arranged in two columns. The first column has "Autonomous System*" (with an asterisk), "IRP's IPv4 address*" (with an asterisk), and "Router name". The second column has "Improvement localpref" and "Router IPv4 address*" (with an asterisk). At the bottom right of the window is a "Save" button with a checkmark icon.

Figure 3.12.5: Configuration editor: New Router, general settings

- Inbound settings:
 - Local IPv4/IPv6 inbound next hop (`bgpd.peer.X.inbound.ipv4.next_hop`, `bgpd.peer.X.inbound.ipv6.next_hop`)
 - Announced inbound LocalPref value (`bgpd.peer.X.inbound.master_localpref`, `bgpd.peer.X.inbound.slave_localpref`)
 - Transit SNMP host (`bgpd.peer.X.transit.snmp`)
 - Transiting traffic (`global.inbound_transit`)

The screenshot shows the same "Add Router" window, but with the "Inbound" tab selected. The main area contains four fields. The first column has "Local IPv4 inbound next_hop*" (with an asterisk) and "Transit SNMP host" (a dropdown menu showing "None"). The second column has "Announced inbound localpref value" and "Transiting traffic" (a toggle switch currently set to "DISABLED"). At the bottom right is a "Save" button with a checkmark icon.

Figure 3.12.6: Configuration editor: New Router, inbound settings

- Advanced settings:
 - Keepalive (`bgpd.peer.X.keepalive`)
 - Improvement communities to be appended to the Community attribute (`bgpd.peer.X.master_communities`)
 - Improvement local-pref (`bgpd.peer.X.master_localpref`)
 - Router ID, mandatory for IPv6 only (`bgpd.peer.X.slave_router_id`)
 - BGP session password (`bgpd.peer.X.master_password`, `bgpd.peer.X.slave_password`)
 - Improvement MED value (`bgpd.peer.X.med`)

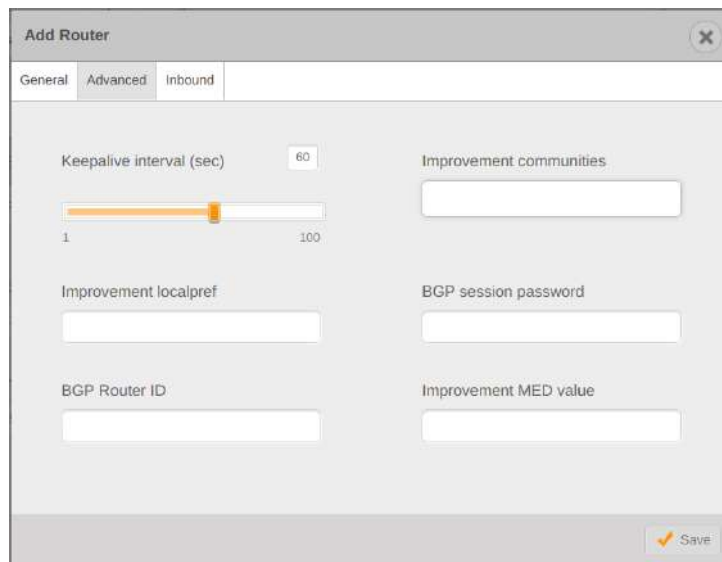


Figure 3.12.7: Configuration editor: New Router, advanced settings

See also: [Bgp Configuration](#), [Bgp settings](#), [BGP sessions settings](#)

- Failover settings - the same set of parameters shall be setup for both master and slave nodes. Use the toggle to switch between them.
 - Improvement communities to be appended to Community attribute (`bgpd.peer.X.slave_communities`, `bgpd.peer.X.slave_communities`)
 - Improvement LocalPref value (`bgpd.peer.X.master_localpref`, `bgpd.peer.X.slave_localpref`)
 - Local IPv4 address (`bgpd.peer.X.master_our_ip`, `bgpd.peer.X.slave_our_ip`)
 - BGP session password (`bgpd.peer.X.master_password`, `bgpd.peer.X.slave_password`)



Figure 3.12.8: Configuration editor: New Router, advanced settings

See also: [Bgpd Configuration](#), [Bgpd settings](#), [BGP sessions settings](#)

3.12.3 Collector configuration

Global collector parameters, as well as Span and Flow-specific settings can be adjusted by using the “Configuration→Collector” menu item.

See also: [Collector Configuration](#), [Collector settings](#).

- Global collector settings:
 - Top volume prefixes per cycle (`collector.export.volume.high.top_n`)
 - Minimal traffic volume (`collector.export.volume.min`)
 - Minimal traffic volume (%) (`collector.export.volume.min_pct`)
- Flow collector settings:
 - UDP port to listen for NetFlow or sFlow packets (`collector.flow.listen.nf`, `collector.flow.listen.sf`)
 - Flow sources (`collector.flow.sources`)
- SPAN collector settings:
 - Capture interfaces (`collector.span.interfaces`)
 - Mindelay algorithm enable/disable (`collector.span.min_delay`)
 - Mindelay probing queue slots (`collector.span.min_delay.probing_queue_size`)

The image shows a configuration editor with four panels:

- Collector:** Contains input fields for 'Top volume prefixes per cycle' (value: 50), 'Minimal traffic volume (bytes)' (value: 100000), and 'Minimal traffic volume (%)' (value: 0.01).
- Analyzed prefixes:** Features a 'Click to add new prefix...' button, a list of three prefixes: '2001:db0::32', '192.168.122.0/24', and '170.238.178.0/20', and an 'Append from file' button.
- Irpflowd:** Includes a toggle switch (ON), 'NetFlow UDP port' (value: 2055), 'sFlow UDP port' (value: 6343), and 'Flow sources' (value: 0/0).
- Irpspand:** Includes a toggle switch (OFF), 'Irpspan interfaces*' (value: eth0), 'min_delay status' (DISABLED), and 'Mindelay probing queue slots' (value: 50).

Figure 3.12.9: Configuration editor: Collector settings

The list of prefixes announced by the monitored network can be edited in the same form. Its contents can be also imported from a text file.

See also: [collector.ournets](#).

3.12.4 Core configuration

All Core parameters affecting the decision-making algorithms can be modified on the next page (“Configuration→Core”). See also: [Global and Core Configuration](#), [Core settings](#).

The following configuration parameters can be adjusted:

- **Max IPv4 improvements** (`core.improvements.max`, `core.improvements.max_ipv6`)
- **Standard reprobing period** (`core.improvements.ttl.retry_probe`) - the time period after which a specific improvement is being scheduled for re-probing.
- **VIP reprobing period** (`core.vip.interval.probe`) - defines the VIP prefixes probing interval, in seconds.
- **Minimal probe lifetime** (`core.probes.ttl.min`) - Ordinary probing will not be performed for a specific prefix if its probe age is lower than this value.
- **Allowed latency worsening** (`core.cost.worst_ms`)
- **Exploring queue slots** (`core.eventqueuelimit`)
- **Top-N relevant volume prefixes** (`core.improvements.retry_probe.volume_top_n`)
- **Relevant loss** for loss-based improvements (`core.performance.loss_pct`) - a prefix will be improved based on a loss cause only if the packet loss can be decreased by this value (in %)
- **Relevant RTT difference** (`core.performance.rtt.diff_ms`)
- **Relevant RTT difference (%)** (`core.performance.rtt.diff_pct`)
- **Maximum probe lifetime** (`core.probes.ttl.max`)



Figure 3.12.10: Configuration editor: Core configuration

[Outage detection](#) algorithm can be enabled and configured on the same page. Outage detection algorithm can be enabled or disabled using the toggle On/Off buttons at the top of each form.

The screenshot shows a configuration window for 'Outage detection'. At the top right, there are 'ON' and 'OFF' toggle buttons. Below the title, there are three settings:

- 'Outage prefix confirmation rate' with a slider set to 60.
- 'Outage round trip rate (%)' with a slider set to 50.
- 'Outage confirmation timeout (sec)' with a text input field containing 600.

Figure 3.12.11: Configuration editor: Outage detection

Throttling excessive bandwidth use with Flowspec can be enabled and configured on the same page. The feature can be enabled or disabled using the toggle On/Off buttons at the top of each form.

The screenshot shows a configuration window for 'Overusage'. At the top right, there are 'ON' and 'OFF' toggle buttons. Below the title, there are six settings:

- 'Overusage interval (sec)' with a text input field containing 60.
- 'Overusage rule retention (sec)' with a text input field containing 60.
- 'Prefix BW average time (hours)' with a text input field containing 1.
- 'Prefix relevant BW (Mbps)' with a text input field containing 50.
- 'Overusage throttle multiplier' with a text input field containing 2.
- 'Overusage threshold multiplier' with a text input field containing 10.

Figure 3.12.12: Configuration editor: Overusage

The following configuration parameters can be adjusted:

- **Overusage interval** (`core.overusage.check_interval`) sets the frequency in seconds of checking for excessive bandwidth use.
- **Overusage rule retention** (`core.overusage.hold_timer`) sets how much time a rule is kept after bandwidth use returns to normal.
- **Prefix BW average time** (`core.overusage.out.average.period`) sets the number of hours used to determine the average bandwidth use of a prefix.
- **Prefix relevant BW** (`core.overusage.out.average.relevant_min`) sets the relevant bandwidth use in Mbps by a prefix before considering any rules for it.

- **Overusage throttle multiplier** (`core.overusage.out.threshold.throttle`) sets the multiplier to apply to average prefix bandwidth use when setting a rule.
- **Overusage threshold multiplier** (`core.overusage.out.threshold.trigger`) sets the threshold multiplier used to determine excessive bandwidth use.

➖ Prefix relevant BW and Overusage threshold multiplier parameters control the number of Flowspec rules that will be generated automatically. Adjust these values to control the number of Throttling Flowspec policies.

Circuit issues detection parameters are listed and can be adjusted:

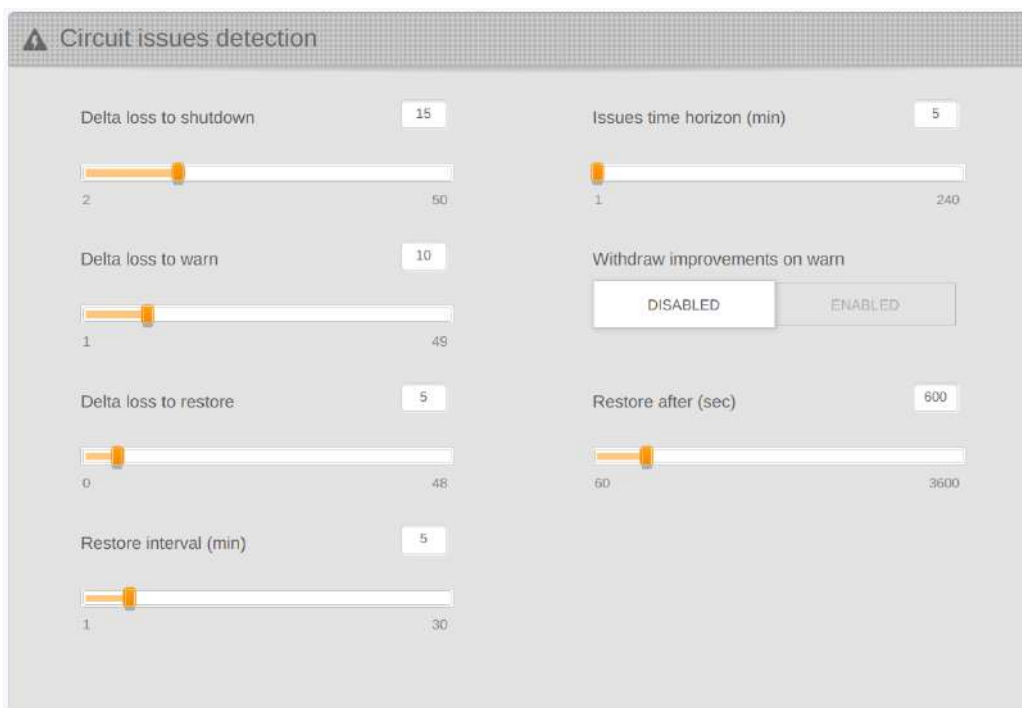


Figure 3.12.13: Configuration editor: Circuit issues detection

The following configuration parameters can be adjusted:

- **Delta loss to shutdown** (`core.circuit.high_loss_diff`) sets threshold to initiate shutting down a provider with circuit issues as a difference between examined provider's average loss and overall average loss during the given time horizon. Shutdown is attempted only for providers marked accordingly.
- **Delta loss to warn** (`core.circuit.warn_loss_diff`) sets threshold to raise a warning regarding issues with a provider as difference between examined provider's average loss and overall average loss during the given time horizon. Usually this threshold is significantly lower than the shutdown threshold.
- **Delta loss to restore** (`core.circuit.recover_loss_diff`) sets low loss level threshold when IRP will restore full functionality over provider that had circuit issues.
- **Issues time horizon** (`core.circuit.hist_interval`) sets how many minutes in the past IRP looks for probes with loss over both analyzed provider and all the other providers on the network.
- **Withdraw improvements on warn** (`core.circuit.withdraw_on_warn`) instructs IRP to withdraw outbound improvements over provider with circuit issues when warning threshold is reached. By default improvements are withdrawn only when shutdown threshold is exceeded.

- **Restore after** (`core.circuit.recover_hold_time`) sets the interval in seconds after which IRP should re-evaluate a provider's circuit loss and attempt restoring it to normal function. This will be performed only for providers that are configured to attempt to restore after shutdown.
- **Restore interval** (`core.circuit.recover_monitored_intervals`) sets the interval in minutes during which IRP will periodically re-evaluate a provider's circuit loss and attempt restoring it to normal function. This will be performed only for providers that are configured to attempt to restore after shutdown.

➤ The configuration parameters above are applied for all providers even though for individual providers different level of control can be set starting from disabling and ending with attempting both to automatically shutdown and later restore connectivity with it. See also [Providers configuration](#)

3.12.5 Commit Control configuration (Not supported in IRP Lite)

Commit Control (Not supported in IRP Lite) algorithm can be enabled and configured on the Commit Control group of pages. It can be enabled or disabled using the toggle On/Off buttons at the top of the form.

⚠ If NetFlow is used to collect statistics Routers MUST be configured to export statistics every minute (or as often as possible). Some router models have default export intervals for either inactive or active flows of up to 1800 seconds. Big delays cause IRP to react very slowly to increased load and reduce the effectiveness of Commit Control feature.

If NetFlow is used to collect statistics Routers MUST be configured to export statistics every minute (or as often as possible). Some router models have default export intervals for either inactive or active flows of up to 1800 seconds. Big delays cause IRP to react very slowly to increased load and reduce the effectiveness of Commit Control feature.

The most important parameters for Commit Control are, as follows:

- **Commit Control probing queue slots** (`core.commit_control.probing_queue_size`)
- **Minimal prefix bandwidth** in Mbps (`core.commit_control.agg_bw_min`)



Figure 3.12.14: Configuration editor: Commit Control

Providers Overall block displays Commit Control configuration for all providers including their precedence. See details in the figure below.

Individual provider parameters can be adjusted as well as detail regarding current Commit Control changes applied by IRP can be accessed by following the provided links.

Providers Overall - Commit Control							
Provider name	Commit Control	Configured 95th (Mbps)	Current bw usage (Mbps)	Monthly 95th (Mbps)	Cost (USD)	Load balancing	Precedence
B	On	80	29.09	51 (64%)	3	Disabled	80
IX2	On	5	0	-	2	Disabled	73
A	On	180	101.86	107 (59%)	3	Disabled	70

Figure 3.12.15: Providers Overall

3.12.6 Explorer configuration

To adjust the Explorer-related parameters, the “Configuration→Explorer” menu item must be accessed. The [Explorer Configuration](#) and [Explorer settings](#) sections should be consulted for detailed Explorer configuration and parameters description.

Explorer parameters:

- Infrastructure IPs (`explorer.infra_ips`)
- Explorer worker threads (`explorer.maxthreads`)
- Probing algorithms and Traceroute algorithms (`explorer.probe.algorithm`, `explorer.trace.algorithms`)
- High volume task precedence (`explorer.high_vol_precedence`)
- Process max collected IPs (`explorer.max_collector_ips`)
- First probing packets amount (`explorer.probing.sendpkts.min`)
- Adaptive probing packets count (`explorer.probing.sendpkts.adaptive_max`)
- ICMP timeout (`explorer.timeout`)
- Traceroute retry packets and packets per hop (`explorer.traceroute.sendpkts`, `explorer.traceroute.retrypkts`)
- Traceroute minimum and maximum ttl (`explorer.traceroute.ttl.min`, `explorer.traceroute.ttl.max`)

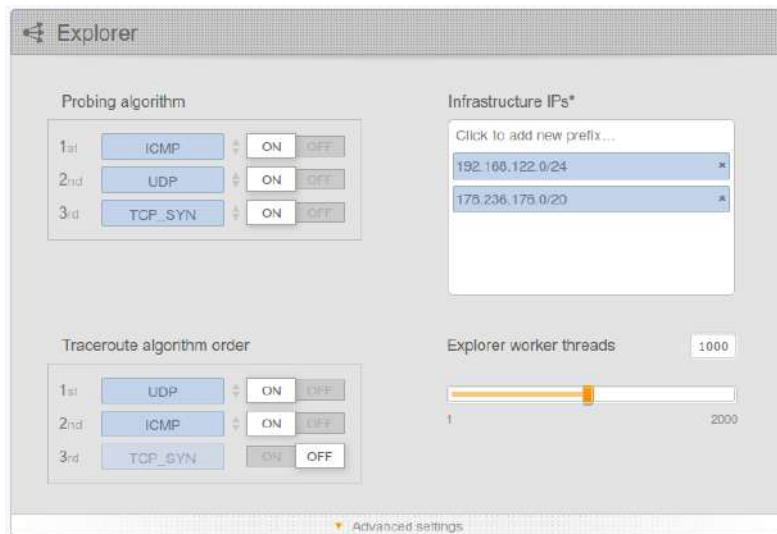


Figure 3.12.16: Configuration editor: Explorer settings

3.12.7 Providers and Peers configuration

Providers can be added and modified via the “Configuration→Providers and Peers” menu.

See also: [Providers settings](#)

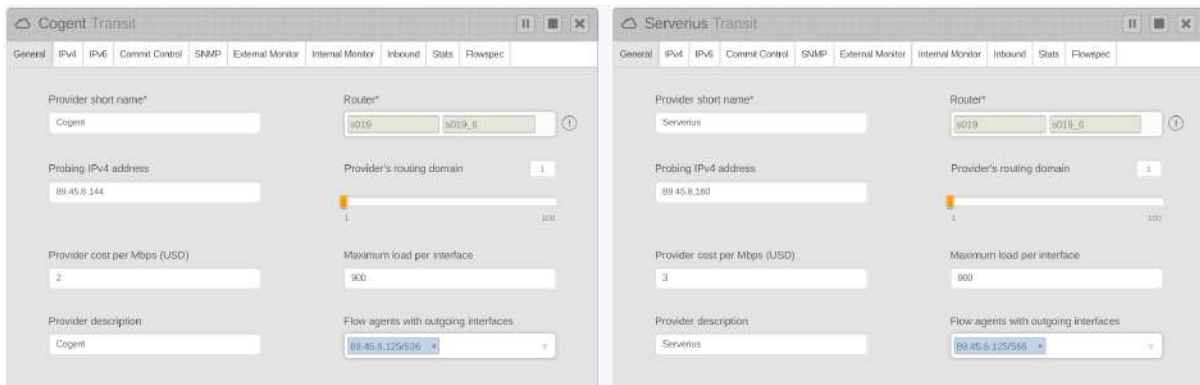


Figure 3.12.17: Configuration editor: Providers configuration

Using the provider control buttons, a specific provider can be temporarily suspended (e.g for a short maintenance in the monitored network), stopped (for long maintenance), or completely deleted. See also: [peer.X.shutdown](#). Commit Control can also be disabled or enabled for each provider specifically. ([peer.X.cc_disable](#)).

i The sections below describe the most common groups of provider configuration parameters. Some parameters are repeated on more than one of these groups for convenience.

3.12.7.1 Providers configuration

To add a new provider, the “Add provider” button must be used. Several parameters should be configured for the new provider:

- General settings:
 - Provider short name ([peer.X.shortname](#))
 - Router ([peer.X.bgp_peer](#))
 - Probing IPv4 address ([peer.X.ipv4.master_probing](#)) or Probing IPv6 address ([peer.X.ipv6.master_probing](#))
 - Provider’s routing domain ([peer.X.rd](#))
 - Provider cost per Mbps ([peer.X.cost](#))
 - Maximum load per interface ([peer.X.limit_load](#))
 - Provider description ([peer.X.description](#))
 - Flow agents ([peer.X.flow_agents](#))
 - Circuit issues detection ([peer.X.circuit.control](#)) indicating how IRP should react to excessive persistent loss over a particular provider.

The screenshot shows a configuration window titled "Add Transit Provider" with a close button in the top right. Below the title bar is a tabbed interface with the following tabs: General, IPv4, IPv6, Commit Control, SNMP, External Monitor, Internal Monitor, and Inbound. The "General" tab is selected. The form contains the following fields and controls:

- Provider short name***: A text input field.
- Router***: A dropdown menu with a warning icon to its right.
- Probing IPv4 address**: A text input field.
- Provider's routing domain**: A numeric input field with a value of "1" and a slider below it ranging from 1 to 100.
- Provider cost per Mbps (USD)**: A text input field.
- Maximum load per interface**: A dropdown menu with "No limit" selected.
- Provider description**: A text input field.
- Flow agents with outgoing interfaces**: A dropdown menu.

A "Save" button with a checkmark icon is located at the bottom right of the form.

Figure 3.12.18: Configuration editor: Adding a new provider, General settings

- Commit Control configuration:
 - Provider cost per Mbps (`peer.X.cost`)
 - Maximum load per interface (`peer.X.limit_load`)
 - Provider 95th percentile (`peer.X.95th`)
 - Provider inbound 95th percentile (`peer.X.95th.in`)
 - Provider 95th calculation mode for inbound traffic (`95th calculation modes`)
 - Commit Control status for this provider (`peer.X.cc_disable`)
 - CC provider precedence - used for Commit Control and grouping (`peer.X.precedence`)
 - Performance/Cost improvements within provider group flag in case Provider load balancing is configured, (`peer.X.improve_in_group`, `peer.X.precedence`)
 - Provider billing day (`peer.X.95th.bill_day`)
 - Centile value (`peer.X.95th.centile`)

Figure 3.12.19: Configuration editor: Adding a new provider, Commit Control

- SNMP-related settings:

Figure 3.12.20: Configuration editor: Adding a new provider, SNMP settings

- SNMP v3 uses additional parameters depending on security services used for statistics collection:
 - SNMP security services selects if Authentication and/or Privacy services are used (`snmp.X.seclevel`)
 - SNMP authentication password if authentication is used (`snmp.X.auth_password`)
 - SNMP authentication protocol if authentication is used (`snmp.X.auth_protocol`)
 - SNMP encryption password if privacy is used (`snmp.X.priv_password`)
 - SNMP encryption protocol if privacy is used (`snmp.X.priv_protocol`)
 - SNMP Username if authentication is used (`snmp.X.auth_username`)
 - SNMP version (`snmp.X.version`)
- External Monitor settings:
 - External monitor status flags for IPv4 / IPv6 (`peer.X.mon.ipv4.external.state`, `peer.X.mon.ipv6.external.state`)

- ICMP/UDP ping monitored IPv4 / IPv6 addresses (`peer.X.ipv4.mon`, `peer.X.ipv6.mon`)

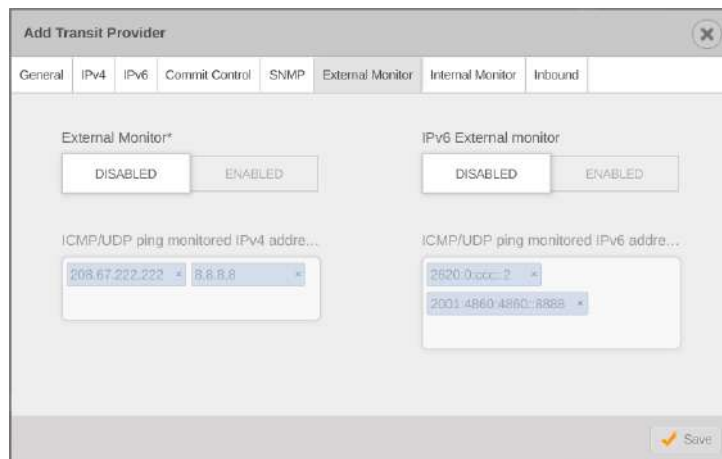


Figure 3.12.21: Configuration editor: Adding a new provider, External Monitor

- Internal Monitor settings:
 - Internal monitor status flags for IPv4 / IPv6 (`peer.X.mon.ipv4.internal.state`, `peer.X.mon.ipv6.internal.state`)
 - BGP session monitoring IPv4 / IPv6 addresses (`peer.X.mon.ipv4.bgp_peer`, `peer.X.mon.ipv6.bgp_peer`)
 - BGP MIBs for IPv4 / IPv6 (`peer.X.mon.ipv4.internal.mode`, `peer.X.mon.ipv6.internal.mode`)
 - SNMP host for BGP Internal Monitor (`peer.X.mon.snmp`)

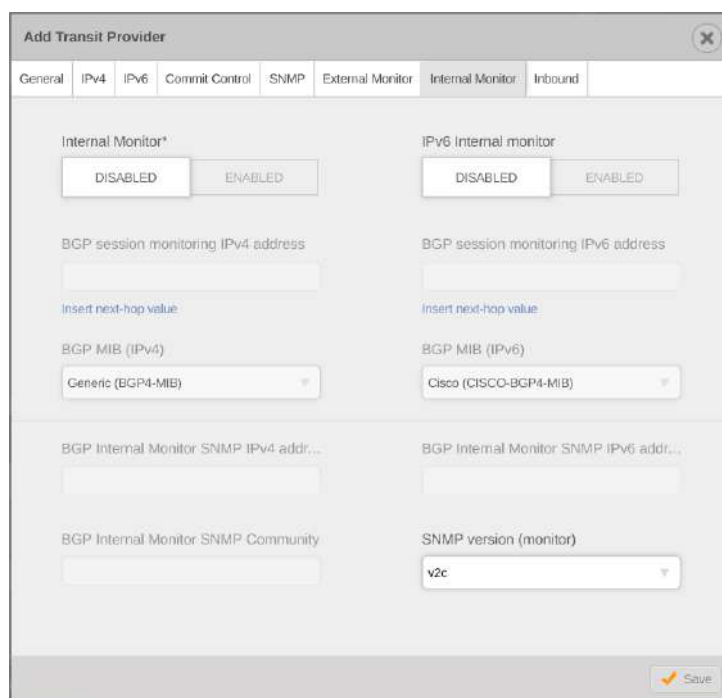


Figure 3.12.22: Configuration editor: Adding a new provider, Monitoring

- SNMP v3 uses additional parameters depending on security services used for monitoring:
- IPv4 / IPv6 settings:
 - IPv4 / IPv6 diagnostic hops (`peer.X.ipv4.diag_hop`, `peer.X.ipv6.diag_hop`)

- Probing IPv4 / IPv6 address (`peer.X.ipv4.master_probing`, `peer.X.ipv6.master_probing`)
- Remote provider ASN (`peer.X.ipv4.next_hop_as`), used by the AS-Path restoration algorithm (`bgpd.as_path`)
- Router next-hop address (`peer.X.ipv4.next_hop`), that sets the next-hop value for injected routes related to this provider

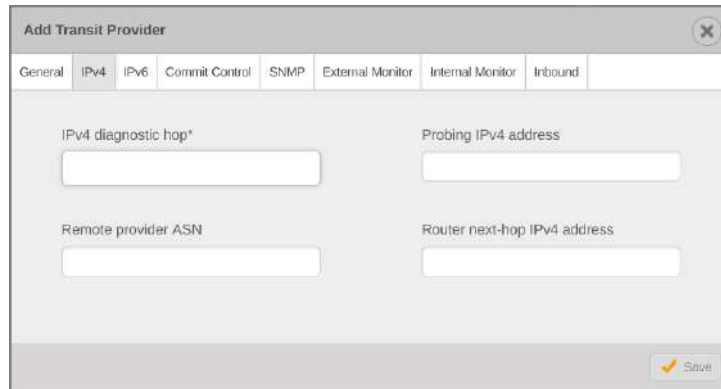


Figure 3.12.23: Configuration editor: Adding a new provider, Commit Control

The same parameters can be adjusted for the already configured providers.

3.12.7.2 Internet Exchanges configuration (Not supported in IRP Lite)

An Exchange is configured as a special type of provider. The important things to consider when setting up an Exchange are listed below.

i It is recommended that Noction systems engineers guide you through Exchange configuration process.

- IRP needs one ACL and one PBR rule for each Peering Partner on an Exchange. Internet Exchanges can have hundreds of Peering Partners. Ensure that the number of Access Lists, Access List entries and PBR entries will not exceed router hardware limitations.

- Once the Exchange is setup IRP will retrieve the routing table on the router in order to setup Exchange peering partners. IRP will require access to the router in order to do so.

- When configuring an Exchange its Diag Hop parameter must be provided. Diag Hop for an Exchange represents the list of direct-connected networks configured on the router's interface towards the Exchange network. This parameter is labeled "IPv4 diagnostic hop" on IRP Frontend.

- Probing IPs for Exchanges no longer require one IP address per peering partner since this number might be impractically big. Instead a combination of probing IP and DSCP value is used to configure the PBRs. A single probing IP in conjunction with DSCP can cover up to 64 peering partners. A sufficient number of probing IPs will be required in order to cover all peering partners on your exchange.

! As a rule of thumb it is better to list the PBR rules for transit providers before the (large number of) rules for Internet Exchange peers. If the many PBR rules assigned to peers on a large Internet Exchange are placed at the beginning of the PBR list some routers evaluate all of them before finding the terms for transit providers and this consumes valuable router CPU resources.

The screenshot shows the configuration interface for an Internet Exchange (IX2). The configuration is organized into two columns. The left column contains fields for 'Provider short name*' (IX2), 'IPv4 diagnostic hop*' (10.10.20.20), 'Provider cost per Mbps (USD)' (2), and 'Provider description' (IX2). The right column contains 'Router*' (s020) and 'Probing IPv4 address' (10.10.20.20, 10.10.20.21, 10.10.20.22). A note below the probing addresses states: '3 IP addresses can be used to setup up to 192 peering partners'. At the bottom right, there is a field for 'Maximum load per interface' set to -1.

Figure 3.12.24: An Internet Exchange is similar to a provider and it requires configuration of the Router, the Probing IPs and the other usual attributes

- Once the Exchange is configured, IRP will need to reset its BGP session towards the router interconnecting with the Exchange in order to retrieve the required routing table information about all peering partners. Once the BGP session is restarted IRP will start fetching Exchange routing tables.

i Give IRP sufficient time (~10 minutes) to fetch Exchange routing tables before continuing configuration.

- IRP offers peering partner autoconfiguration. It retrieves the list of Next Hops (peering partners) on the Exchange with the corresponding list of prefixes individual peers accept.

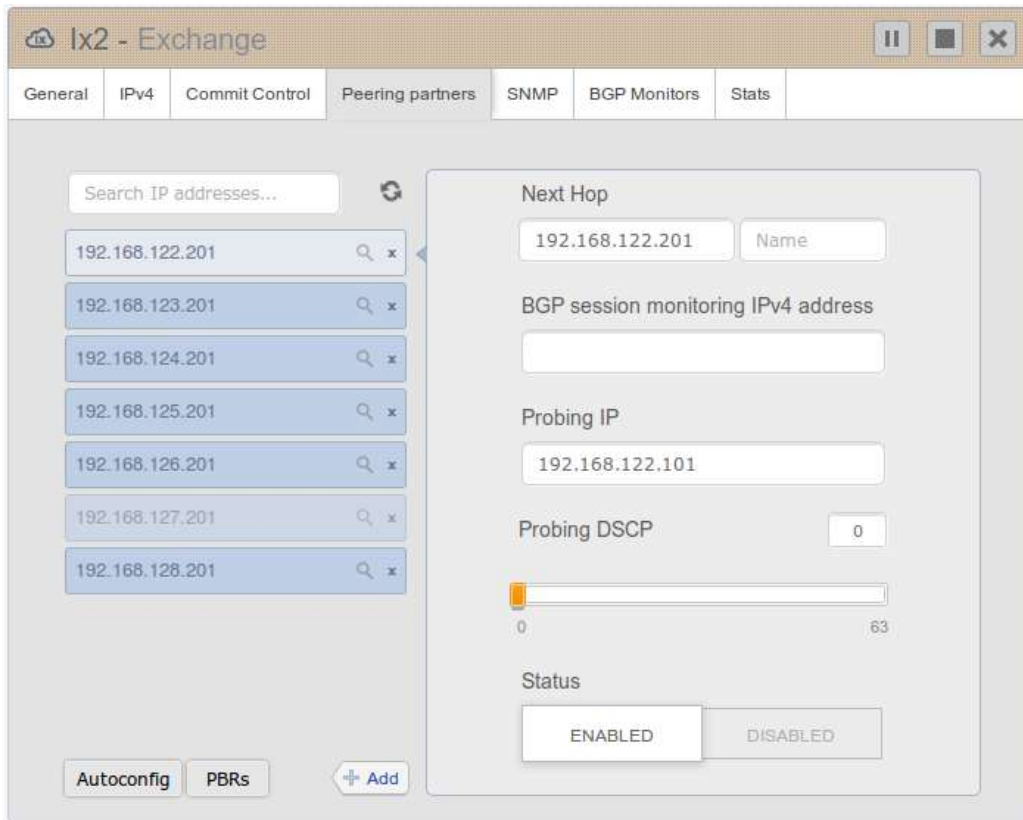


Figure 3.12.25: The list of peering partners for the Exchange shows details about each of them and offers access to IRP Autoconfiguration feature and PBR ruleset generator for the router

- Use the Autoconfiguration feature to create an initial configuration for IRP. Review Exchange peering partners before starting the Exchange. Consider enabling periodic auto re-configurations for selected IX in order to update periodically the value of BGP session monitoring IPv4 address from the Exchange and to add new peering partners. Refer [global.exchanges.auto_config_interval](#) and [peer.X.auto_config](#).

i Keep in mind that Autoconfiguration might tamper with all the changes you've made directly within IRP config files for peering partners. So, use Autoconfiguration sparingly after you've applied manual changes or try avoiding direct configuration file changes altogether.

- Once the Exchange is configured correctly within IRP you will need to apply the PBR rules on the router(s). IRP includes the functionality to generate PBR rules for different router vendors. You will need to review the generated ruleset and apply it on the router manually.

The image shows a 'Generate PBR' dialog box with the following fields:

- Router Manufacturer: Cisco
- Route Map: irp-ix
- ACL name start: 100
- Interface: ve 4

Buttons: Close, Submit

Figure 3.12.26: PBR Generator

- It is possible that the Autoconfiguration feature on the Exchange has been run with incomplete routing tables or that new peering partners have been connected. This is especially true for very big Exchanges. In this case it is possible that some peering partners are neither in IRP nor router configurations. When IRP detects such a case it raises a warning about newly discovered peering partners like in the image below. At this stage you will need to both re-autoconfigure IRP and extract the PBRs for the router.

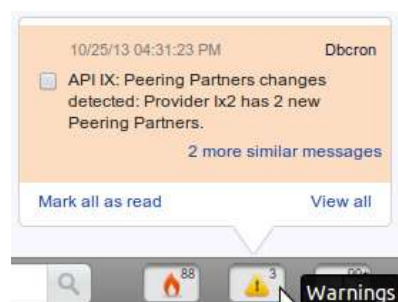


Figure 3.12.27: Warning about new peering partners that have been added to the Exchange

- After its creation and up to its complete configuration the Exchange is kept in a Stopped state. When both IRP and the Router PBRs are configured IRP is ready to start optimizing Exchange traffic too. Keep in mind that starting the Exchange will require a BGP session restart.

i We must mention that before applying changes to IRP configuration the changes are validated. If errors are detected these will be flagged and the previous good configuration will be kept intact so you will have the option to review the erroneous data and correct.

3.12.7.3 Switch a provider from one router to another

Switching a provider from one router to another is possible via manual IRP reconfiguration only.

Use the following steps to switch the provider from one router to another:

1. Suspend the provider using IRP Frontend “Providers and Peers” configuration section. IRP withdraws all existing and stops new improvements towards shutdown provider(s).

2. You can remove traffic from the provider (by denying incoming/outgoing announces in the BGP configuration) and then physically re-cable the provider from one router to another. Then configure BGP session towards the provider on new router.
3. The PBR rule for the provider should be configured on new router (move provider’s PBR settings from the old router to the new one).
4. Change IRP box local PBR rules if any.
5. Check if PBR works properly using traceroute tool.
6. Modify (/etc/noction/irp.conf) assigned router for the provider (refer to `peer.X.bgp_peer`).
7. Modify (/etc/noction/irp.conf) SNMP interface/IP/community for the provider (refer to `SNMP Hosts settings`, `peer.X.snmp.interfaces`, `peer.X.mon.snmp`).
8. Reload Bgpd configuration (affected BGP sessions could be reset).
9. Re-activate the provider using IRP Frontend “Providers and Peers” configuration section
10. Check IRP BGP Internal and External monitor states. They must be UP.

3.12.8 SNMP hosts configuration

⚠ Starting with version IRP 3.7 SNMP is configured as SNMP hosts and provider specific SNMP settings are being deprecated.

SNMP hosts are nodes on the network that provide or read SNMP data. Multiple SNMP hosts are configured in this section of configuration and references to them are used from elsewhere in the configuration. SNMP hosts can be configured by navigating via the “Configuration→SNMP hosts” menu.

See also: [SNMP Hosts settings](#)

Figure 3.12.28: Configuration editor: SNMP hosts configuration

- SNMP-related settings:
 - Flag indicating SNMP supported version (`snmp.X.version`)

- SNMP host short name that assigns a recognizable name of the SNMP host in IRP (`snmp.X.name`)
- SNMP host IPv4/IPv6 address (`snmp.X.ip`)
- SNMP v2 uses additional parameter:
 - SNMP host community (`snmp.X.community`)
- SNMP v3 uses additional parameters depending on security services used:
 - SNMP security level selects if Authentication and/or Privacy services are used (`snmp.X.seclevel`)
 - SNMP authentication password if authentication is used (`snmp.X.auth_password`)
 - SNMP authentication protocol if authentication is used (`snmp.X.auth_protocol`)
 - SNMP encryption password if privacy is used (`snmp.X.priv_password`)
 - SNMP encryption protocol if privacy is used (`snmp.X.priv_protocol`)
 - SNMP Username if authentication is used (`snmp.X.auth_username`)

3.12.9 Notifications and Events

Notifications are messages sent to alert about some events registered by IRP. There are two prerequisites in order to start using notifications:

1. Configure event parameters and channels
2. Subscribe for event notifications

IRP can send notifications as SMS, email and SNMP Trap.

3.12.9.1 Configure notification and events

Events configuration changes default threshold values when an event is raised. The most relevant events are presented in the following figure. For example Overloaded by (%) indicates that this event will fire when the aggregated outbound bandwidth usage for all providers exceeds by 10% the configured limits. Refer section 4.1.10 for details.

The screenshot shows the 'Events configuration' interface with the following settings:

Event Name	Threshold Value	Unit
Provider outbound overload by (%)	10	%
Provider outbound overload by (Mbps)	100	Mbps
Overload by (%)	10	%
Overload by (Mbps)	100	Mbps
Inbound overload by (Mbps)	100	Mbps
Inbound overload by (%)	10	%
Provider inbound overload by (Mbps)	100	Mbps
Provider inbound overload by (%)	10	%

At the bottom of the interface, there is a link to 'Show advanced settings'.

Figure 3.12.29: Configuration editor: Events configuration

IRP uses a local email server to send emails. Still, email sent by this server might trigger filtering policies enforced by some third parties that might block some important event notifications. It is possible to provide the details of another email server on your network that is protected for this. The following figure highlights the available email channel configuration parameters. Besides specifying the server email configuration sets the sender of email messages that will show in receiver's inbox.

i Only SMTP servers without authentication are currently supported.

Figure 3.12.30: Configuration editor: Email configuration

SMS configuration is mandatory for sending SMS notifications.

IRP supports the following SMS gateways:

- Twilio
- Plivo

A valid account with a supported SMS gateway is required in order to finish configuration. Check the following figure for details:

Figure 3.12.31: Configuration editor: SMS configuration

The settings are:

- SMS gateway - choose one of the supported gateways.
- Account ID - the public identifier assigned to your account by the SMS gateway. The following figure highlights this value for Plivo.
- Max message size - the maximum length of the SMS text. The SMS will be trimmed by IRP before sending. Subsequently the SMS gateway will split the SMS into multiple parts if required.
- Secret - the secret identifier assigned to your account by the SMS gateway. The following figure highlights this value for Plivo.
- From phone number - the phone number that shows as sender on the receiver's mobile device. Use a phone number supported by the SMS gateway.

Refer section 4.1.10 for complete details about configuration parameters.

⚠ Some SMS gateways enforce the From phone number and will reject numbers that do not comply with their policy.



Figure 3.12.32: Plivo account ID and secret

SNMP Traps configuration requires a list of destination IP addresses and an SNMP community parameter to deliver SNMP traps. The following figure highlights the basic configuration. There is a series of advanced configuration parameters like the SNMP Trap port and rate of packets. Refer section 4.1.10 for details.



Figure 3.12.33: Configuration editor: SNMP Traps configuration

Web Hooks configuration requires only a Webhook URL to be provided. There is a series of advanced configuration parameters in case their default settings are not satisfactory. Refer section 4.1.10 for details.



Figure 3.12.34: Configuration editor: Web Hooks configuration

The settings are:

- Webhook URL - the unique URL the team is assigned by Web hook provider. An example for Slack.com is shown.
- Avatar icon URL - optional parameter that designates an icon (usually PNG or JPEG image are supported) assigned to the Webhook bot in the channel.
- Avatar emoji - an alternative avatar specified in the form of an emoji in “:robot-face:” format.
- Bot name - optional bot name assigned to Webhook bot.

Refer section 4.1.10 for complete details about configuration parameters.

🚨 Web hooks support has been tested and verified to work for Slack.com API.

3.12.9.2 Subscriptions

Notifications are sent only if a valid subscription has been created. Subscriptions, similar to regular emailing of reports can be found under <Username> → Notifications menu.

Notifications page provides an overview of existing subscriptions with features to create, edit, delete them. Check the following figure for a preview.

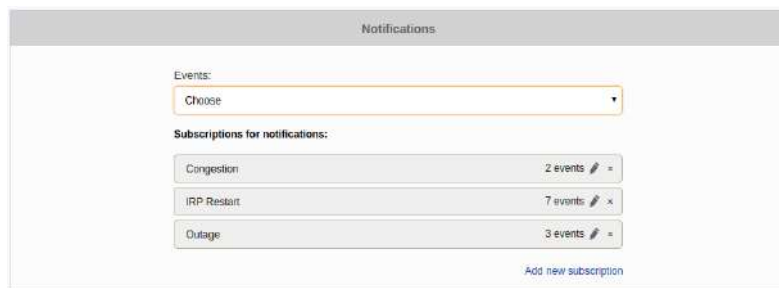


Figure 3.12.35: Notifications

Use Add new subscription or Events drop-down to create a new subscription or edit, delete existing subscriptions by clicking corresponding icons. Creating or editing a subscription will bring up a pop-up like the one in the following figure.

Figure 3.12.36: Subscribe notifications

Subscription details include:

- Topic - the title given to subscription in Notification page and also a textual part of email and SMS notifications.

i If SMS notifications are used it is advised to keep the topic short so that there is enough space left to include details about the event.

- Interval between notifications - sets a rate limit of how often an email and/or SMS notification is sent. Value 'No limit' for the interval depicts no rate limits and all events will trigger a notification.

i SNMP Traps notification are not constrained by the interval between notifications. SNMP Traps are sent immediately as they are raised.

w The rate limit is enforced per subscription so it is still possible to receive multiple notifications if the subscribed events are part of different subscriptions.

- Destinations - IP address of a trap receiver for SNMP Trap notifications, email addresses, phone numbers and webhook channels. Multiple destinations of same type can be provided. At least one valid destination must be provided per subscription.

i IRP parses and recognizes whether a provided destination is a valid IP address, email address, phone number or web hook channel. Web hook channels for example are recognized by detecting that the first character of a channel name is the “#” mark.

- Event filters - allows filtering of events by textual description or by IRP component. Remove the filters to see all the subscribed events.
- Subscribed events - the full list of events supported by IRP and tick marks for the events in this subscription. At least one event is required for a valid subscription.

3.12.9.3 Notification text

When a subscribed event is fired IRP will send notifications. A sample notification text (email) looks like the following figure:



Figure 3.12.37: Subscribe notifications

The email sample above identifies the different parts of the message:

1. IRP server name
2. Subscription topic as specified in the subscription
3. From email address - as configured in email channel
4. Time - date-time of the last event that caused the notification. In case of rate limitation this might be older than the time of the email.
5. Textual description of the event and any relating details.
6. Older events in this subscription that have been retained due to rate limitations. Keep in mind that all past events are aggregated into a single email or SMS message.

3.12.10 Security Configuration

Security configuration covers restrictions on the ranges of IP addresses that can access IRP's Frontend and also users allowed to access IRP as well as their roles.

3.12.10.1 Allowed ranges of IP addresses

IRP Frontend includes a feature to restrict access by IP address and this feature can be turned on or off as needed. When this feature is used allowed IP addresses are maintained as a list of prefixes in CIDR notation.



Figure 3.12.38: Configuration editor: Security

3.12.10.2 Users and user directories

Users who can access IRP are managed in User Directories. User directory is a generic name for an external user management provider for example LDAP (either POSIX or Active Directory). IRP can interface with as many user directories as needed in a specific environment.

IRP recognizes Admin and User privileges that can be granted to users. Once a user logs into the system IRP creates a profile that maintains his personal configurations like dashboards or notifications.

Users					
#	Username	Email	Privileges	User Directory	+ Add
1	admin	admin@noction.com	Admin	Internal	
2	noction	support@noction.com	User	Internal	
9	windowsUser	winUser@NOCTION.local	Admin	AD Test	
11	ldapUser	ldapUser@noction.com	Admin	LDAP TEST	

User Directories			
Name	Type	Hostname	+ Add
LDAP TEST	LDAP POSIX	ldap.noction.com	
AD Test	Active Directory	23.45.54.32	
Default internal user directory (MySql)	Internal	-	

Figure 3.12.39: Configuration editor: Users and Directories

3.12.10.3 LDAP and Active Directory

LDAP or AD user directories can be added, updated and removed from IRP by accessing “Configuration→Security” menu item in the User Directories tab. Each user directory takes a series of parameters specific for the protocol.

All operations with DNs (initial bind DN, group DNs, user names) are case insensitive and also strip redundant whitespace.

Refer individual protocol documentation for how to correctly configure one or another user directory. The example below offer a generic set of parameters required to configure IRP to use Active Directory for access management.

The screenshot shows a configuration window titled "Edit user directory LDAP TEST". It has three tabs: "General", "Advanced", and "Bindings". The "General" tab is active. The form contains the following fields and controls:

- User directory name:** Text input field containing "LDAP TEST".
- Type:** Dropdown menu showing "Active Directory".
- State:** Toggle buttons for "DISABLED" and "ENABLED", with "ENABLED" selected.
- Order:** A numeric input field with "0" and a slider below it ranging from 0 to 255.
- Hostname:** Text input field containing "ldap.noction.com".
- Port:** Text input field containing "389".
- Save:** A button with a checkmark icon and the text "Save" at the bottom right.

Figure 3.12.40: Configuration editor: LDAP User Directory configuration

The general tab covers:

- User directory name - the name assigned to the directory within IRP,
- User directory type - one of the supported User Directories,
- Enabling or disabling a user directory,
- Order specifies when this user directory will be examined by IRP compared to other user directories,
- User directory hostname in the form of either IP address or domain name
- User directory port

The screenshot shows the "Advanced" tab of the "Edit user directory LDAP TEST" configuration editor. The form contains the following fields and controls:

- Initial bind DN:** Text input field containing "cn=LDAP TEST,ou=app-accounts,dc:".
- Initial bind Password:** Password input field with masked characters "*****". Below it is a checkbox labeled "Show password" which is unchecked.
- Timeout:** A numeric input field with "5" and a slider below it.
- TLS:** Toggle buttons for "DISABLED" and "ENABLED", with "ENABLED" selected.
- Certificate verification:** Toggle buttons for "DISABLED" and "ENABLED", with "ENABLED" selected.
- TLS CA Certificate file:** Text input field.
- Save:** A button with a checkmark icon and the text "Save" at the bottom right.

Figure 3.12.41: Configuration editor: Users and Directories

Advanced configuration of a user directory covers:

- Initial binding user name that IRP uses to authenticate itself,
- Initial bind password assigned to IRP,

i Usually bind passwords are not required to access directories. If a password is required and configured via IRP Frontend, it will be scrambled in IRP configuration files. If the password is specified directly in IRP configuration it is provided in clear-text with the condition that it does not start/end with scrambled password encapsulation characters.

- Timeout before failing a connection to this user directory,
- TLS use,
- Certificate verification and
- CA certificate used to verify server's certificate.

Figure 3.12.42: Configuration editor: Users and Directories

Bindings maps User Directory attributes to IRP specific attributes, for example:

- Base DN and User DN specifies the root distinguished name and user subtree

i IRP recognizes BOTH short and full user identifiers. Examples below are both valid directory entries that will match user “chris” with long name “Mr. Chris Smith”:

```
cn: ops
uniqueMember: chris
```


- Admin role groups are user directory objects that list users with Administrator privileges within IRP as compared to all users. Both short and full paths are accepted for Admin role groups.
- User role, Username, Email and Common name fields map User Directory attributes to IRP user attributes,
- User filters can specify any valid LDAP search criteria (without outside parenthesis).

3.12.10.4 TACACS+ directory

TACACS+ user directories can be added, updated and removed from IRP by accessing “Configuration→Security” menu item in the User Directories tab. Each user directory takes a series of parameters specific for the protocol.

The example below highlights the parameters required to configure IRP to authenticate against a TACACS+ directory/host.

The screenshot shows a configuration window titled "New user directory TAC". The "General" tab is active. The form contains the following fields and controls:

- User directory name:** Text input field containing "TAC".
- Type:** Dropdown menu set to "TACACS+".
- State:** Toggle buttons for "DISABLED" and "ENABLED", with "ENABLED" selected.
- Order:** A slider control set to 42, with a range from 0 to 255.
- Hostname:** Text input field containing "1.2.3.4".
- Port:** Text input field containing "49".
- Secret:** Text input field containing "asecret".

At the bottom right, there are two buttons: "Check" and "Save", both with checkmark icons.

Figure 3.12.43: Configuration editor: TACACS+ configuration

The parameters include:

- User directory name - the name assigned to the directory within IRP,
- User directory type - one of the supported User Directories in our case TACACS+,
- Enabling or disabling a user directory,
- Order specifies when this user directory will be examined by IRP compared to other user directories,

- User directory hostname in the form of either IP address or domain name
- User directory port
- A shared secret used by IRP and TACACS+ host to secure communications.

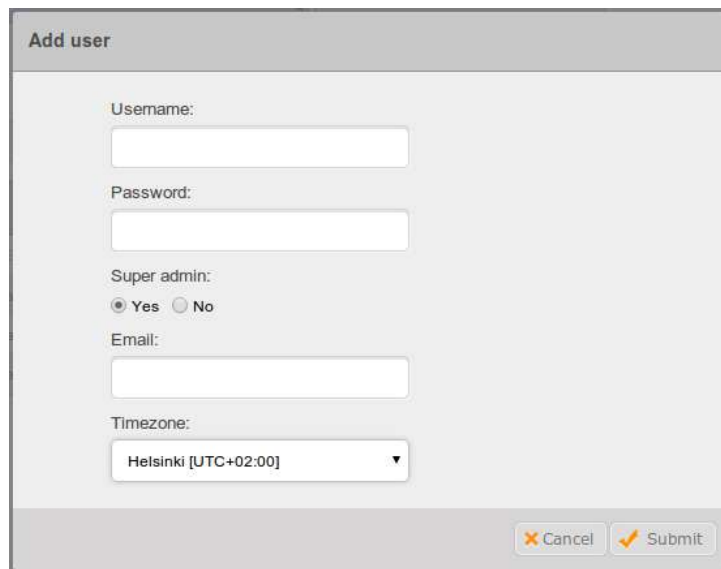
3.12.10.5 Internal user directory

IRP includes an Internal user directory that allows multiple user accounts to be setup. User accounts management is done by accessing “Configuration→Security” menu item.

i User management for external user directories is performed outside IRP using designated tools. The Internal directory is used when external user directories are not available or their use is not desirable.

To edit a specific user’s profile, press the Edit button next to it. To add a new user, click on the “Add” button. Specific users may be given Admin privileges. In such case they can also manage all system users.

The timezone specified in the user profile will be considered while displaying system Reports and Graphs.



The screenshot shows a web form titled "Add user". It contains the following fields and controls:

- Username:** A text input field.
- Password:** A text input field.
- Super admin:** Radio buttons for "Yes" (selected) and "No".
- Email:** A text input field.
- Timezone:** A dropdown menu with "Helsinki [UTC+02:00]" selected.
- Buttons:** "Cancel" and "Submit" buttons at the bottom right.

Figure 3.12.44: Configuration: Adding a new user account

3.12.11 Frontend configuration

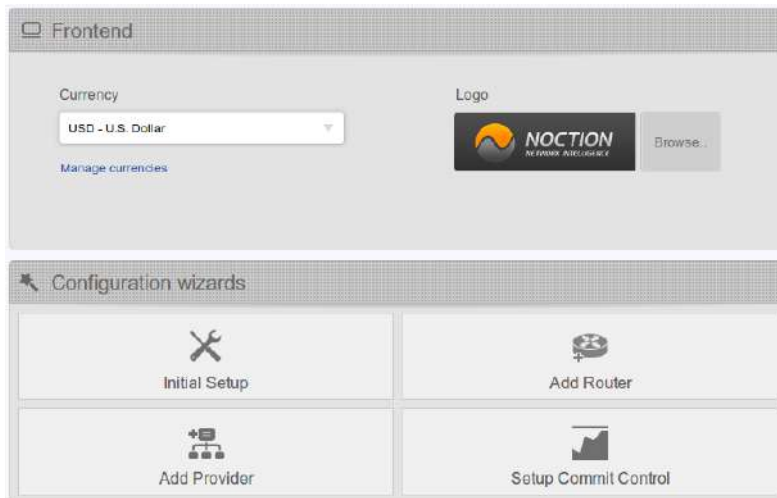


Figure 3.12.45: Configuration editor: Frontend

The currency used in various cost-related reports as well as the system logo (shown in the login form and on the menu bar) can be changed on the same page.

Currencies can be added/deleted manually by using the “Manage currencies” feature.

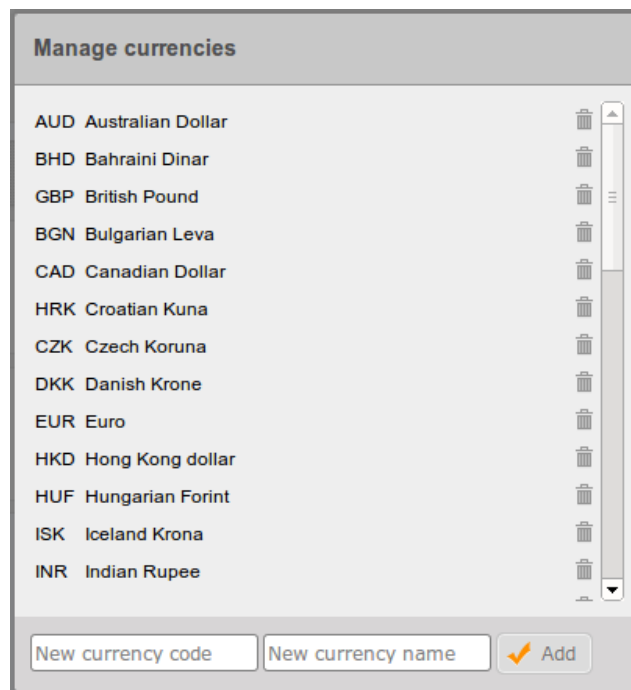


Figure 3.12.46: Configuration Editor: Manage currencies

⊖ Only a Super admin user can change the currency setting. The currency is a global system value, and cannot be adjusted per-user.

The system logo is a global setting and can be changed by the Super administrator only.

i The default currency is “USD - U.S. Dollar”.

A custom logo image can be uploaded by using the Upload button. Afterward, the image can be cropped to fit the standard logo size. The new settings are applied after clicking the “Submit” button.

3.12.12 Inbound configuration (Not supported in IRP Lite)

Refer [Inbound optimization \(Not supported in IRP Lite\)](#) for an overview of Inbound feature.

- Flow (sFlow, NetFlow) is a prerequisite for Inbound optimization. IRP collects and uses statistics about upstream provider incoming traffic targeting inbound prefixes. This statistics should be made available by properly setting up Flow and provider specific Flow agents (refer [peer.X.flow_agents](#)).

i SPAN does not carry details about the specific interface packets enter your network and as such it is impossible to distribute inbound traffic targeting Inbound prefixes by upstream provider.

All Inbound related features, including configuration, moderation, and reports are listed under main menu Inbound:

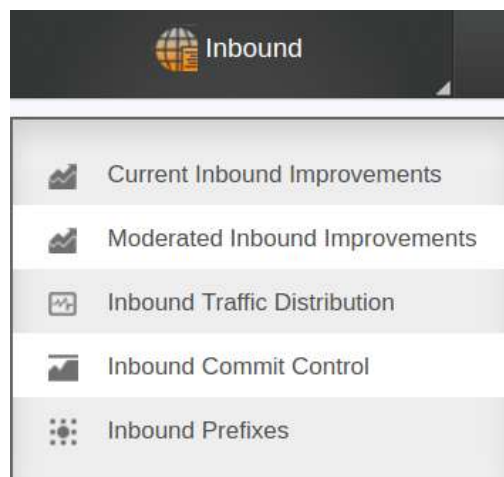


Figure 3.12.47: Main menu: Inbound

3.12.12.1 Inbound prefixes

i Note that optimization of transiting traffic feature does not rely on manually configured transit prefixes. Instead filters by ASN and/or prefix are used to constrain the specific prefixes that can be improved.

Inbound optimization manipulates how much traffic targets YOUR prefixes through different providers. In order to do this you need to configure IRP and let it know what are the Inbound prefixes. You can specify a subset of your network prefixes and you can also split larger prefixes into smaller ones in order to have a more fine-grained control over the Inbound traffic. These prefixes will be announced back to your originating routers marked with the community designated to prepends count to use when announcing to individual upstream providers.

➤ Refer for details regarding IRP announcement of inbound prefixes.

📘 Splitting larger prefixes into subprefixes offers IRP the opportunity to better control the granularity of traffic shaping changes enabling improvement of smaller chunks of overall traffic. At the same time original prefixes can be announced as before and they will guarantee continuity of the prefix for cases when improvements announced by IRP become unavailable for some reason (for example if smaller prefixes are filtered in some networks).

➤ Verify during Inbound configuration that the split inbound prefixes are not filtered by your providers and propagate as expected on the Internet.

IRP Frontend includes facilities that let you define the list of YOUR prefixes covered by Inbound traffic optimization as highlighted below and includes:

- prefix belonging to the network (`inbound.rule.X.prefix`),
- router(s) where announcements are sent (`inbound.rule.X.bgp_peer`),

➤ Inbound improvements must be advertised by IRP to core routers that originate the prefix. This way the improvements will propagate consistently through your network to edge routers and beyond.

- `prefixstatus` as each prefix can be disabled in order to exclude from further inbound optimization (`inbound.rule.X.enabled`),
- next hop used by improvements for this prefix (`inbound.rule.X.next_hop`),
- an option to instruct IRP whether to fully control the prefix or to only announce improvements when there are any. Refer the section for further details.
- providers for which this inbound prefix can be optimized (`inbound.rule.X.providers`). It must be noted that if no provider is selected then the prefix is optimized for all providers. Otherwise, the prefix will be optimized only through selected providers.

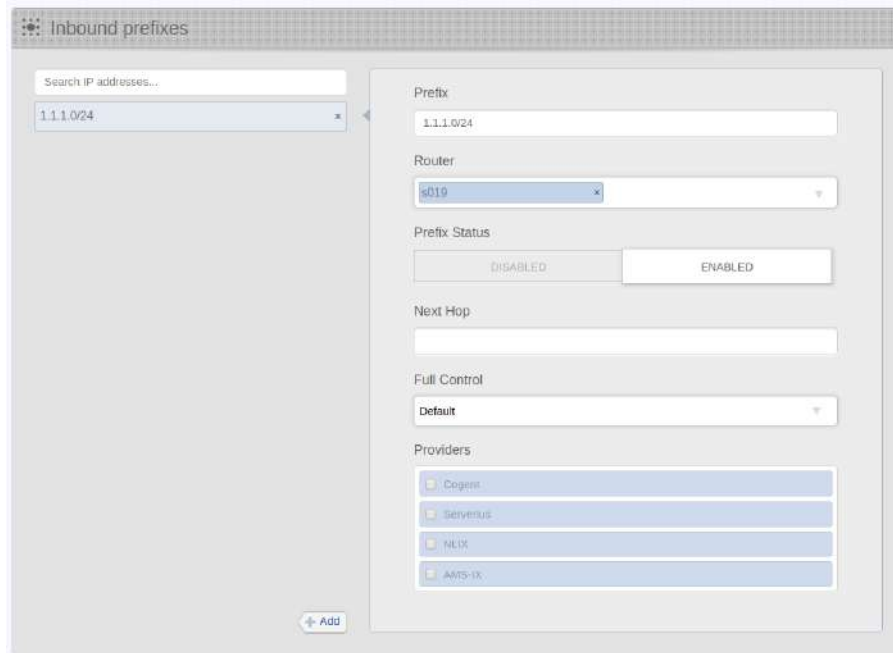


Figure 3.12.48: Configuration editor: Inbound prefixes

i Inbound prefixes can be found both via Inbound main menu or Configuration →Collectors

Inbound prefixes full control IRP operates under the assumption that the network already originates all network’s prefixes and only a subset are configured in IRP for inbound optimization purposes. This subset includes the intentionally split prefixes that allow better bandwidth control granularity by IRP. Under these assumptions IRP announces to the network only the improvements it makes. This avoids the overhead of announcing too often without real benefit for the network.

On some networks there might be a preference to guarantee that an inbound prefix is announced to all (allowed) upstreams at the same time thus making sure that they all have identical knowledge about the route with possibly different number of prepends.

Improvements Announce only improved inbound prefixes with communities for improved providers

If improved Announce only improved inbound prefixes with communities for all allowed providers

All Always announce all inbound prefixes with communities for all allowed providers

Options “If improved” and “All” are useful when the network itself does not originate a prefix, then the inbound prefix announced by IRP fully describes the route and provider preference to all upstream providers thus ensuring that all of them can route our traffic and the corresponding prepends count will influence what providers are preferred.

To enable inbound prefix full control either individual inbound prefix explicit value should be set in Configuration →Collectors →Inbound Prefixes

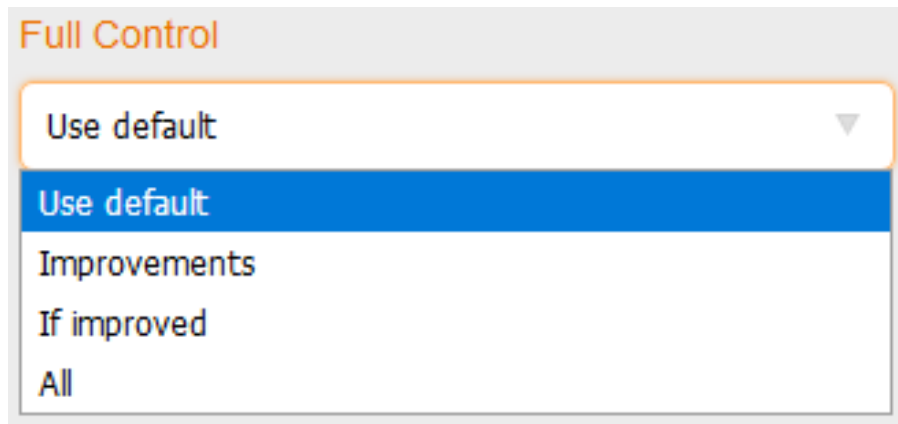


Figure 3.12.49: Configuration editor: Control of individual inbound prefix

or default IRP behavior should be specified in Configuration →BGP and Routers

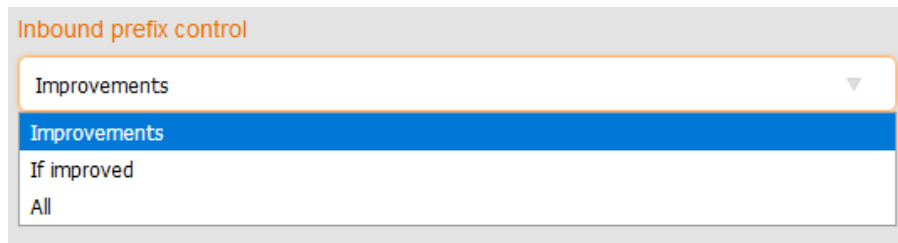


Figure 3.12.50: Configuration editor: Inbound prefixes control

3.12.12.2 Inbound prepends

Inbound optimization in IRP relies on BGP priority rules that generally elect best routes based on AS PATH length. By instructing your network to add or remove prepends for specific prefixes over different providers, IRP is able to influence how your inbound traffic flows. Inbound optimization in IRP is less effective for those regions of the Internet where your providers or other networks on the Internet use other policies for electing best routes. For example, Internet Exchange routes always take precedence for peers. This makes impossible prepend based inbound traffic optimization for Internet Exchanges. Often partial providers employ similar policies and inbound traffic flowing through them cannot be optimized. Prepends are not added directly by IRP. Instead IRP inbound improvements are marked with designated communities that are recognized and acted upon by network routers.

Base communities Inbound optimization improvements advertise larger or smaller counts of prepends to be announced to different providers. IRP uses BGP session to communicate to routers and relies on BGP communities to communicate the improvements. Each provider is assigned a base community which represents zero prepends. Additional prepends are represented by incrementing accordingly the right-side part of the provider's community.

⚠ IRP uses 8 prepend levels. This means that base community should be chosen to allow additional 7 values without intersecting with other community values.



Figure 3.12.51: Configuration editor: Base communities

i Inbound base communities are assigned for each individual provider Configuration →Providers and Peers

Additionally routemaps are applied so that routers depict the designated communities in IRP's improvements and change the announcement of the inbound prefix towards that provider appropriately. Different routemaps are produced and applied on each router in your network. See a few examples below. Request assistance from Noction's support team to prepare exact routemaps for your network configuration and router models.

3.12.12.3 Routers configuration for Inbound

w Complete sample configurations for Cisco, Juniper, Brocade and Vyatta will be provided by Noction Support on request.

3.12.12.4 Inbound Commit Control

Inbound bandwidth control is part of the Commit Control feature of IRP. You need to enable Commit Control and further to enable Inbound Commit control to make use of this feature. Besides the above mentioned settings for Inbound Commit Control separate low and high limits are configured to instruct IRP when to start and when to stop improving and also how to estimate the effects of inbound improvements.

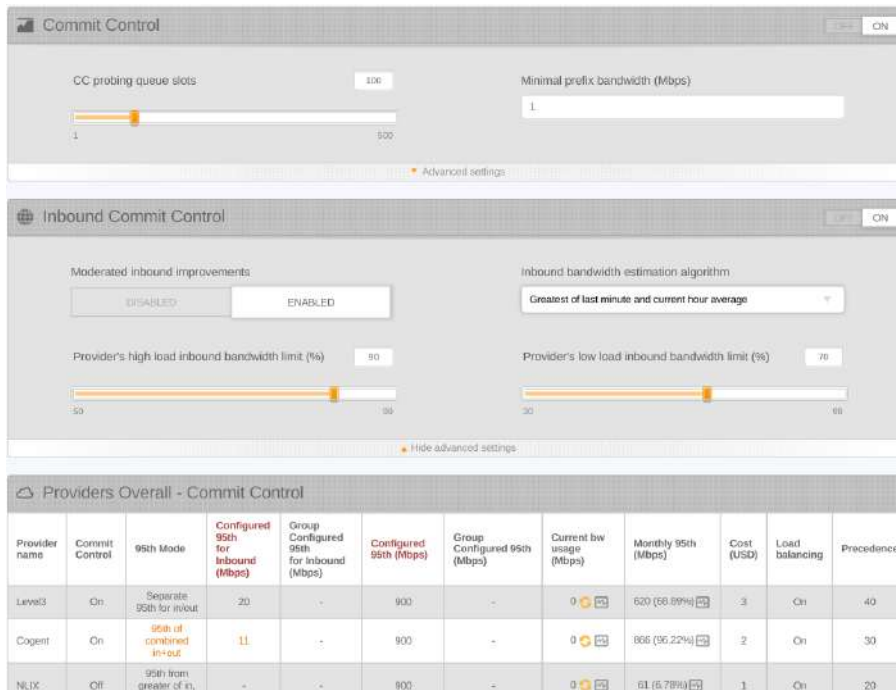


Figure 3.12.52: Configuration editor: Inbound commit control

i Inbound commit control can be found both via Inbound main menu or Configuration → Commit Control

Commit control for inbound as can be seen above screenshot highlights the relevant configuration parameters:

- whether inbound commit control is on or off
- whether review and moderation feature is enabled or disabled
- what is the bandwidth estimation algorithm for inbound and
- high and low limits for inbound in case the 95th mode warrants independent inbound vs outbound optimization.

Refer also to `peer.X.95th.mode`, `core.commit_control.inbound.enabled`, `core.commit_control.inbound.moderated`.

Estimating inbound traffic Internet traffic has high variability and adjacent measurements can differ significantly between them. In these conditions a leveling function helps mitigate the risk of generating excessive numbers of Inbound improvements due to higher traffic variability of some networks. This estimation can be fine tuned by the Inbound bandwidth estimation algorithm parameter that tells IRP what value to use as the basis of the calculation. See the possible base values below:

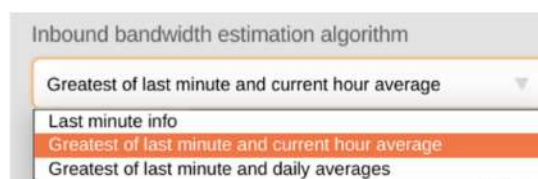


Figure 3.12.53: Configuration editor: Inbound BW estimation algorithms

Moderated Inbound Improvements IRP includes a moderation feature that allows review and approval of Inbound improvements. If this feature is enabled Inbound improvements are not automatically announced but instead they are queued for review. If an inbound improvement has not been approved in time for a subsequent cycle of improvements IRP will review the proposal itself and will adjust as needed thus making another recommendation.

See the capture below.

#	Timestamp	Link1	Link2	Prefix	Type			
1	03/03/16 04:30:56 PM	2 (+1)	4 (+2)	10.1.1.0/24	✓	Commit	Reject	
2	03/03/16 04:34:11 PM	3 (+1)	3 (+1)	10.1.2.0/24	✓	Commit	Reject	
3	03/03/16 07:14:10 PM	-	2 (+2)	10.1.3.0/24	-	Commit	Reject	
4	03/03/16 05:01:34 PM	1	2	10.1.4.0/24	✓	Commit	Reject	
5	03/03/16 06:23:06 PM	2	2	10.1.6.0/24	✓	Commit	Reject	
6	03/03/16 08:49:06 PM	-	2 (+2)	10.1.7.0/24	-	Commit	Reject	
7	03/03/16 06:58:34 PM	2	4	10.1.8.0/24	✓	Commit	Reject	
8	03/03/16 05:01:42 PM	2	2	10.1.9.0/24	✓	Commit	Reject	

Figure 3.12.54: Configuration editor: Inbound BW estimation algorithms

Inbound Improvements moderation highlights the number of additional prepends proposed by IRP and allows submitting or rejecting individual improvements as well as a batch of selected improvements.

3.12.12.5 Optimization of transiting traffic

Optimization of transiting traffic is setup as part of Inbound Commit Control.

Inbound Commit Control [OFF] [ON]

Moderated inbound improvements: DISABLED ENABLED

Transit optimization: DISABLED ENABLED

Transit ASNs:

- Click to add new asn...
- 3 x
- 2 x

 or Append from file

Transit Improvements TTL min: 14401

Top transit prefixes per cycle: 101

Inbound bandwidth estimation algorithm: Greatest of last minute and current hour average

Max transit Improvements: 101

Transit prefixes:

- Click to add new prefix...
- 1.2.0.0/16 x
- 60.240.242.217/32 x
- 2001.db8:/32 x
- 192.168.0.0/16 x
- 172.16.0.0/12 x

 or Append from file

Transit Improvements TTL max: 86401

Match transit at egress: DISABLED ENABLED

Figure 3.12.55: Configuration editor: Inbound optimization of transiting traffic parameters

i Inbound commit control can be found both via Inbound main menu or Configuration → Commit Control

Optimization of transiting traffic configuration parameters are highlighted:

- Transiting traffic toggle that enables or disables the feature (`global.inbound_transit`)
- Transit Improvements Max sets the maximum number of transit improvements (`core.improvements.inbound_transit.max`)
- Transit ASNs and Transit prefixes specify those segments of the Internet that IRP monitors and optimizes (`bgpd.prefixlist.asn`, `bgpd.prefixlist.prefixes`)
- Transit Improvements TTL min and max set the lower and upper bounds in seconds to keep a transit improvement (`core.improvements.inbound_transit.ttl.min`, `core.improvements.inbound_transit.ttl.max`)

i Note that traffic belonging to a specific prefix that fragments a large transit prefix is collected as belonging only to the specific prefix and excluded from the larger prefix traffic statistics. If the specific prefix is also filtered from IRP configuration as for example is the case of a /26 prefix, then the specific prefix will be excluded from optimization of transiting traffic and will not show up in any of the subsequent decisions or reports.

- Transit Top N prefixes sets the number of transit prefixes that are collected each cycle and considered for optimization (`collector.flow.export.inbound_transit.topn`)
- Match transit at egress enables or disable statistics collection for transit prefixes when packets exist the network (`collector.flow.process_transit_in_outbound`)

3.12.13 Wizards

Wizards can be accessed from Configuration → Setup Wizards or from Configuration → Frontend → Configuration Wizards.

3.12.13.1 Initial Setup

IRP initial setup requires to:

1. Specify Infrastructure IP addresses and list of analyzed networks
2. Enable and configure Flow or Span Collector to gather network statistics
3. Set IRP Improvement mode
4. Setup the management interface
5. Indicate interfaces that IRP uses for active probing

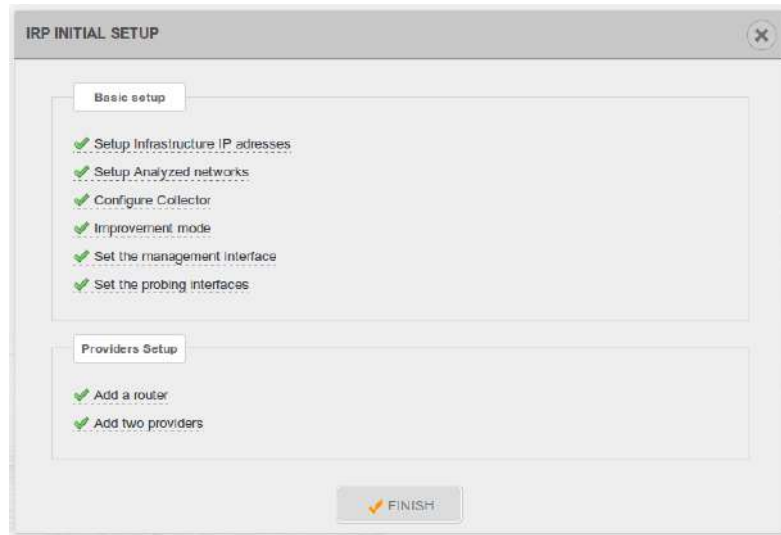


Figure 3.12.56: Configuration editor: Initial Setup

- Basic Setup
 - Setup Infrastructure IP addresses (`explorer.infra_ips`)
 - Setup Analyzed networks (`collector.ournets`)

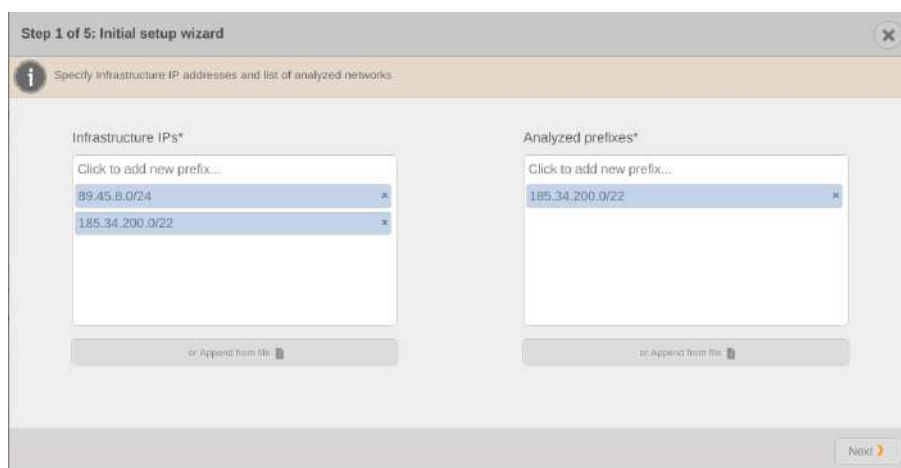


Figure 3.12.57: Configuration editor: Setup Infrastructure IP addresses and Analyzed networks

- Configure Collector:
 - * Irpflowd
 - Irpflowd enable/disable(`collector.flow.enabled`)
 - NetFlow UDP port(`collector.flow.listen.nf`)
 - sFlow UDP port(`collector.flow.listen.sf`)
 - Flow Sources(`collector.flow.sources`)
 - * Irpspand
 - Irpspand enable/disable (`collector.span.enabled`)
 - Irpspand interfaces (`collector.span.interfaces`)
 - Mindelay status (`collector.span.min_delay`)
 - Mindelay probing queue slots (`collector.span.min_delay.probing_queue_size`)

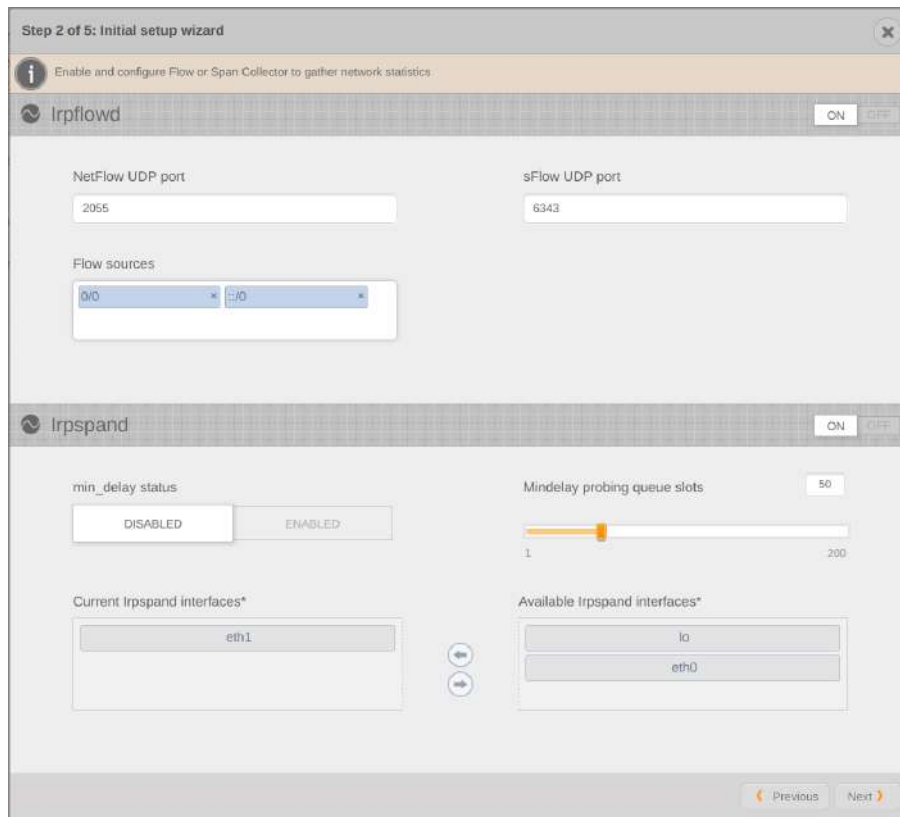


Figure 3.12.58: Configuration editor: Configure Collector

- Improvement mode (`global.improve_mode`)



Figure 3.12.59: Configuration editor: Improvement mode

- Set the managements interface (`global.master_management_interface`)



Figure 3.12.60: Configuration editor: Management Interface

- Set the probing interface (`global.master_probing_interface`)

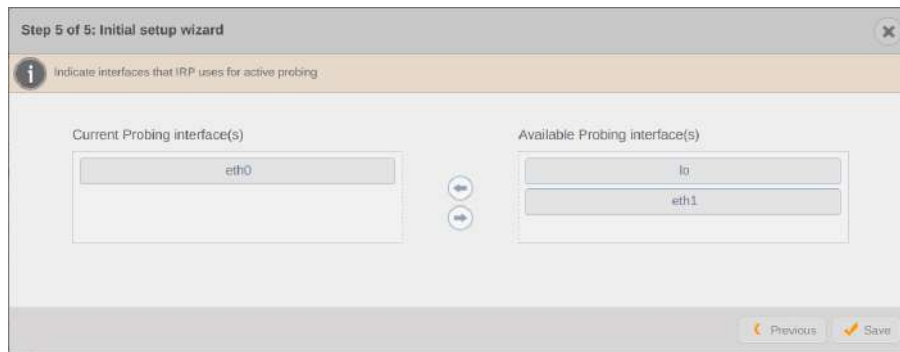


Figure 3.12.61: Configuration editor: Probing Interface

- Providers Setup
 - Add a router, see [Add Router](#)
 - Add two providers, see [Add Provider](#)

3.12.13.2 Add Router

IRP communicates improvements to your edge routers. Add Router wizard guides router configuration through the following steps:

1. Identify the router and its AS
2. Router IP address to setup BGP session
3. BGP announcement attributes to distinguish and prioritize IRP improvements on this router

Check below the corresponding wizard steps:

- Router name assigned within IRP for easy identification and
- Autonomous System number of the network (`bgpd.peer.X.as`)

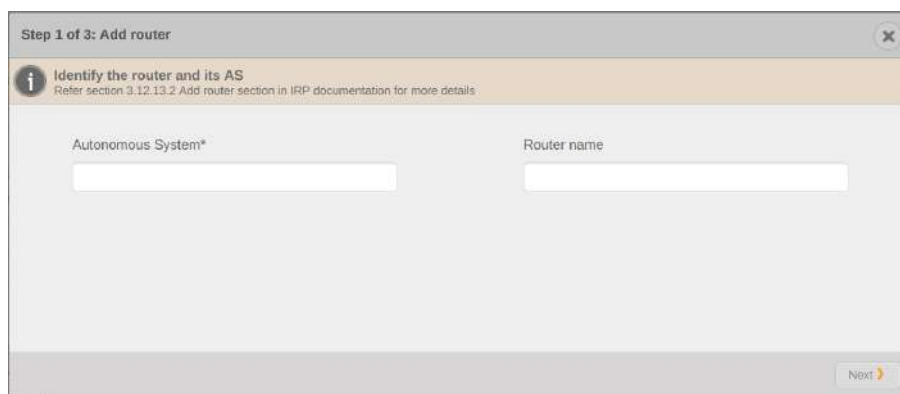


Figure 3.12.62: Configuration editor: Router name and AS

- Router IPv4 addresses
 - Local router IPv4 address (`bgpd.peer.X.master_our_ip`)
 - Router IPv4 address (`bgpd.peer.X.master_peer_ip`)

Figure 3.12.63: Configuration editor: Router IPv4 addresses

- Router announcement attributes for IRP improvements
 - Improvement LocalPref value (`bgpd.peer.X.master_localpref`)
 - Improvement communities(`bgpd.peer.X.master_communities`)
 - Improvement MED value (`bgpd.peer.X.med`)

Figure 3.12.64: Configuration editor: Router announcing

3.12.13.3 Add Provider

The wizard Add Provider covers the following:

1. Router - Identify the router and routing domain where the provider interconnects with your network
2. Provider - Specify what is a provider ASN, short name and description
3. IP addresses - Specify and assign provider network addresses that IRP will use
4. Commit Control - Optionally set provider usage thresholds to be used for Commit Control
5. SNMP host - Specify the SNMP host for this provider
6. External monitor - Indicate if an External monitor is used and designated external IP addresses used to verify this provider link status
7. Internal monitor - Indicate if an Internal monitor based on BGP session with provider will be used and the corresponding SNMP resource
8. Pre-checks - Validate given provider parameters before submitting them

- Choose a Router (`peer.X.bgp_peer`)

Figure 3.12.65: Configuration editor: Choose a Router

- Provider name
 - Provider short name (`peer.X.shortname`)
 - Provider description (`peer.X.description`)

Figure 3.12.66: Configuration editor: Provider name

- Provider IPv4 addresses
 - Probing IPv4 address (`peer.X.ipv4.master_probing`)
 - IPv4 diagnostic hop (`peer.X.ipv4.diag_hop`)
 - Router next-hop address (`peer.X.ipv4.next_hop`)
 - Router ASN (`peer.X.ipv4.next_hop_as`)

Figure 3.12.67: Configuration editor: Provider IPv4 addresses

- Provider IPv6 addresses
 - Provider probing IPv6 address (`peer.X.ipv6.master_probing`)
 - IPv6 diagnostic hop (`peer.X.ipv6.diag_hop`)
 - Router nex-hop IPv6 address (`peer.X.ipv6.next_hop`)
 - Router ASN (`peer.X.ipv6.next_hop_as`)

Figure 3.12.68: Configuration editor: Provider IPv6 addresses

- Provider Commit Control
 - Provider 95th percentile (`peer.X.95th`)
 - Provider cost per Mbps (`peer.X.cost`)
 - Commit Control provider precedence (`peer.X.precedence`)
 - Maximum allowed load per interface (`peer.X.limit_load`)
 - Commit Control status for this provider (`peer.X.cc_disable`)
 - Allow Performance/Cost Improvements within provider group members (`peer.X.improve_in_group`)

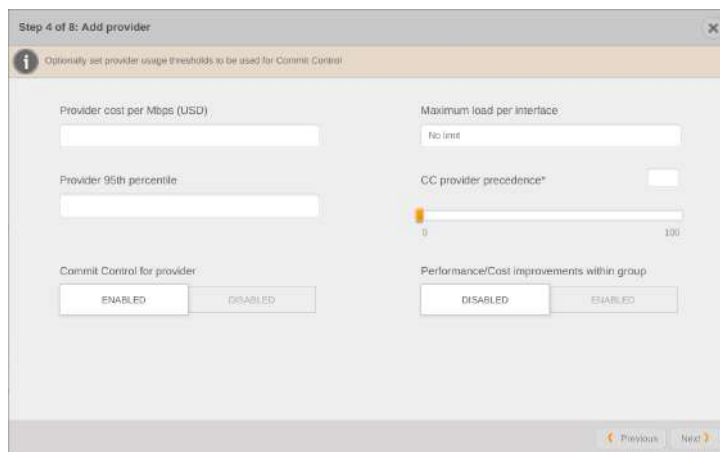


Figure 3.12.69: Configuration editor: Provider Commit Control

- Provider Monitoring setup IPv4
 - Provider SNMP interfaces (`peer.X.snmp.interfaces`, `global.ifstats`)
 - SNMP host settings (`SNMP Hosts settings`)

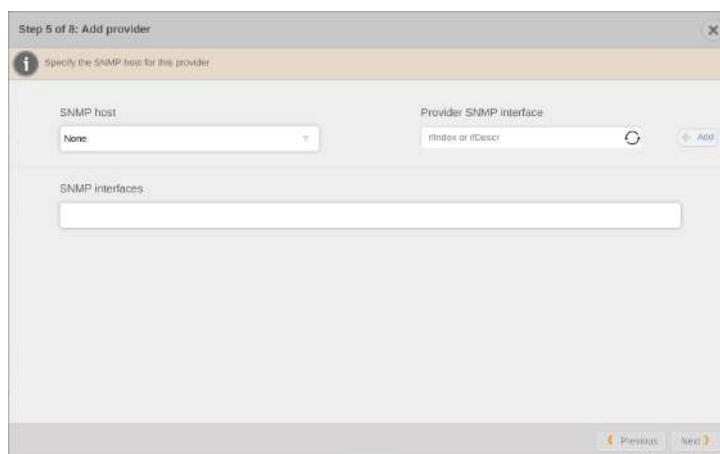


Figure 3.12.70: Configuration editor: Provider Monitoring setup IPv4

- Provider Monitoring setup IPv6
 - Provider SNMP interfaces (`peer.X.snmp.interfaces`, `global.ifstats`)
 - SNMP host settings (`SNMP Hosts settings`)

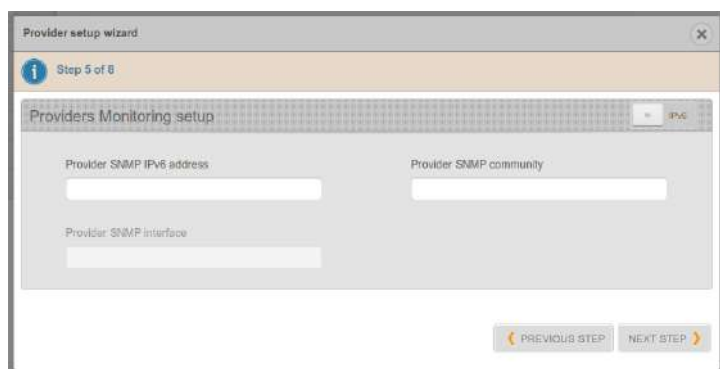


Figure 3.12.71: Configuration editor: Provider Monitoring setup IPv6

- External Monitor setup IPv4
 - External Monitor enable/disable (`peer.X.mon.ipv4.external.state`)
 - Router monitoring IPv4 address(`peer.X.ipv4.mon`)

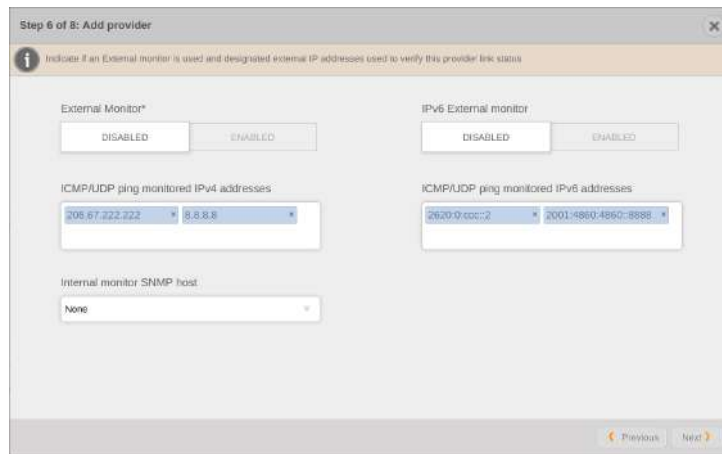


Figure 3.12.72: Configuration editor: External Monitor setup IPv4

- External Monitor setup IPv6
 - External Monitor enabled/disabled (`peer.X.mon.ipv6.external.state`)
 - Router monitoring IPv6 address`peer.X.ipv6.mon`)



Figure 3.12.73: Configuration editor: External Monitor setup IPv6

- Internal Monitor setup
 - Internal Monitor enable/disable (`peer.X.mon.ipv4.internal.state`)
 - Monitoring IPv4 address (`peer.X.mon.ipv4.bgp_peer`)

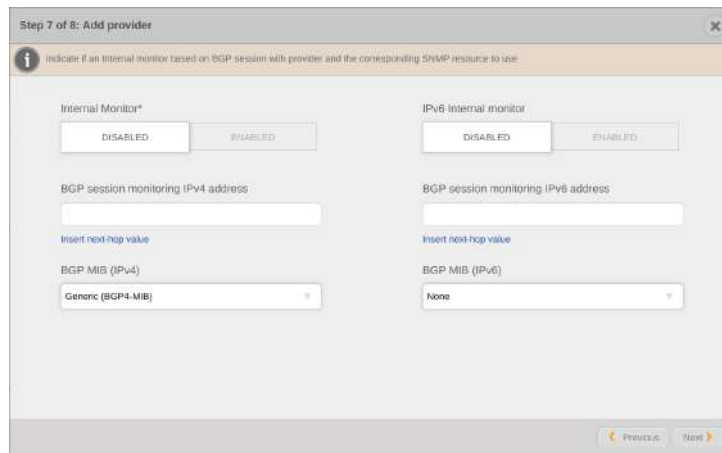


Figure 3.12.74: Configuration editor: Internal Monitor setup

- Provider pre-checks

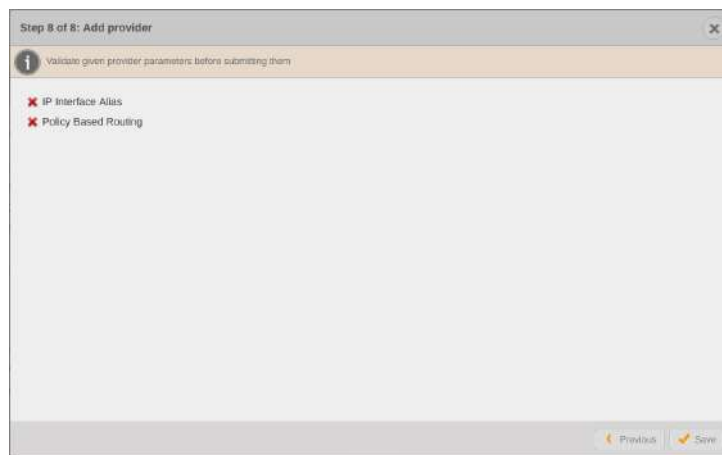


Figure 3.12.75: Configuration editor: Provider pre-checks

3.12.13.4 Setup Commit Control (Not supported in IRP Lite)

Step 1: Enable Commit Control Providers that should maintain a target 95th limit are identified and the exact limits are set. Link upper limit represents 80-90% of interface capacity and will be pre-populated for you.

- Commit control enables/disables the feature for a specific provider(`peer.X.cc_disable`)
- 95th target specifies the contracted bandwidth usage target in Mbps (`peer.X.95th`)
- Link upper limit specifies the maximum allowed bandwidth on the link in Mbps (`peer.X.limit_load`). This represents ~80-90% of interface capacity and will be pre-populated for you. Change it as appropriate.

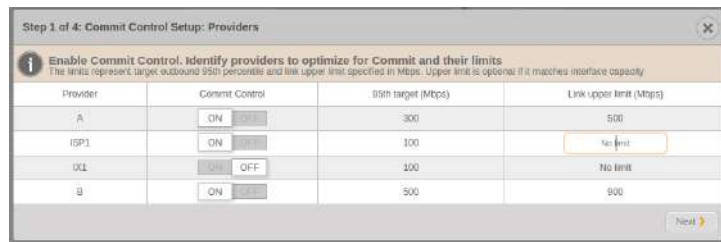


Figure 3.12.76: Configuration editor: Commit Control basic setup

Step 2: Provider precedence Precedence indicates how important it is to keep the commit levels for different providers. Traffic will be unloaded to the provider with least precedence in situations when all other providers are overloaded. If two or more providers have equal precedence they will form groups. Move the slider or use drag and drop to specify the order of providers from most important to least important.



Figure 3.12.77: Configuration editor: Commit Control provider precedence setup

Step 3: Groups configuration Groups are formed by a set of providers (as configured in IRP) that share some common characteristics, for example a redundant link or a virtual link consisting of multiple physical links towards the same upstream provider. Depending on customer's needs, groups can be configured to permit or forbid performance rerouting from one provider in the group towards another. If providers are grouped they will be assigned the same precedence. Groups are optional and can be skipped.

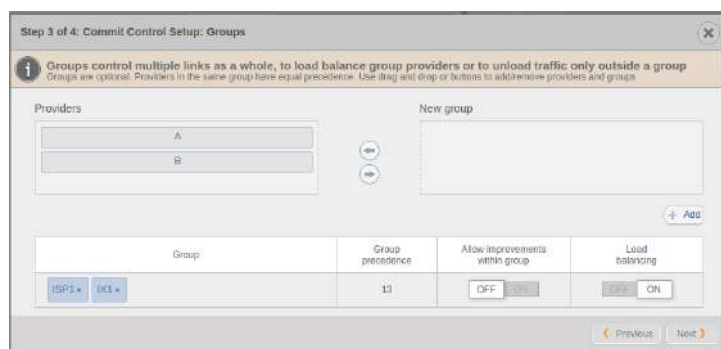


Figure 3.12.78: Configuration editor: Commit Control groups setup

Step 4: Commit Control Overview The last step of the wizard is there to review the configuration. Last minute changes can be made to some parameters. When submitted the configuration changes will be validated and applied if correct.

- Provider name (`peer.X.shortname`)

- Commit control status for a specific provider (`peer.X.cc_disable`)
- Configured 95th (`peer.X.95th`)
- Upper limit (`peer.X.limit_load`)
- Load balancing for groups
- Improve in group (`peer.X.improve_in_group`)
- Precedence (`peer.X.precedence`)

Provider	Commit Control	95th target (Mbps)	Link upper limit (Mbps)	Load balancing	Improve in group	Precedence
A	On	300	500	On	Off	40
ISP1	On	100	No limit	On	Off	13
(K)	Off	100	No limit	On	Off	13
B	On	500	800	On	Off	10

Figure 3.12.79: Configuration editor: Commit Control Provider overall

3.12.13.5 Setup Failover wizard (Not supported in IRP Lite)

Step 1: Enable failover Failover configuration has a series of mandatory global settings before enabling it.

- Failover status (`global.failover_role`)
- Slave IP address (`global.failover_slave.ip`)
- Slave SSH port (`global.failover_slave.port`)
- Failover timer in seconds (`global.failover_timer_fail`)
- Failback timer in seconds (`global.failover_timer_failback`)

Figure 3.12.80: Configuration editor: Failover setup

Step 2: Probing IP configuration Probing IPs must be assigned to all configured providers for both master and slave nodes (`peer.X.ipv4.master_probing`, `peer.X.ipv4.slave_probing`).

Provider	Master Probing IP	Slave Probing IP
A	192.168.123.44	192.168.123.79
B	192.168.123.100	192.168.123.79
rSP1	192.168.123.94	192.168.123.95
IX1	192.168.123.80	192.168.123.87
	192.168.123.81	192.168.123.88
	192.168.123.82	192.168.123.89
	192.168.123.83	192.168.123.90
	192.168.123.84	192.168.123.91
	192.168.123.85	192.168.123.92

Figure 3.12.81: Configuration editor: Failover probing IPs for providers

Step 3: Router BGP session settings Master and slave nodes of a failover configuration establish BGP sessions with all configured routers. The following parameters are required:

- Local IP address for both master and slave (`bgpd.peer.X.master_our_ip`, `bgpd.peer.X.slave_our_ip`)
- BGP session password for master and/or slave if any (`bgpd.peer.X.master_password`, `bgpd.peer.X.slave_password`)

Router	Master Local IP Address	Master Password	Slave Local IP Address	Slave Password
s019	192.168.123.42	-	192.168.123.77	-
s019v6	-	-	8.8.8.8	*****
s020	192.168.123.1	-	192.168.123.2	-
test1	192.168.123.104	-	192.168.123.105	-
test2	3.3.3.3	-	3.3.3.4	-

Figure 3.12.82: Configuration editor: Failover BGP session settings

Step 4: BGP LocalPref and Communities Master and slave nodes apply BGP attributes to announcements:

- LocalPref for master and slave (`bgpd.peer.X.master_localpref`, `bgpd.peer.X.slave_localpref`)
- Communities for master and slave (`bgpd.peer.X.master_communities`, `bgpd.peer.X.slave_communities`)

⊖ Master's LocalPref value must be greater than slave's LocalPref.

Router	Master Localpref	Master Communities	Slave Localpref	Slave Communities
s019	300	65504.1	130	65503.1
s019v6	200	-	130	-
s020	150	-	130	-
test1	846	1111.2222	130	1111.2222
test2	140	222.333	130	222.3331

Figure 3.12.83: Configuration editor: BGP announcement attributes

Step 6: Probing interfaces Failover requires configuration of Management and Probing interfaces for both master and slave nodes (`global.master_probing_interface`, `global.slave_probing_interface`). Toggle between master and slave settings to specify values for both nodes.

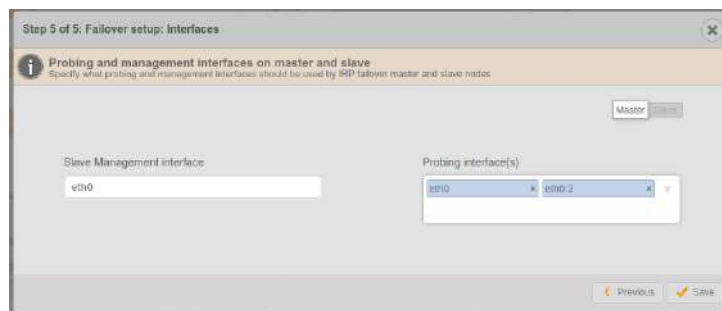


Figure 3.12.84: Configuration editor: Probing and management and interfaces

Chapter 4

Configuration parameters reference

4.1 Complete parameters list

4.1.1 Database credentials

4.1.1.1 db.clickhouse.dbname

Name of the ClickHouse database that contains the IRP tables.

Default value: `irp`

4.1.1.2 db.clickhouse.host

Database host. ClickHouse server host name or IPv4/IPv6 address.

Default value: `localhost`

4.1.1.3 db.clickhouse.password

The password used to connect to a ClickHouse server.

4.1.1.4 db.clickhouse.port

TCP port number used to connect to a ClickHouse server.

Default value: `9000`

4.1.1.5 db.clickhouse.username

User name used to connect to a ClickHouse server.

Default value: `irp`

4.1.1.6 db.dbname

Name of a MySQL database that holds the IRP tables.

Default value: `irp`

4.1.1.7 db.host

Database host. MySQL server host name or IPv4/IPv6 address.

Default value: localhost

4.1.1.8 db.ourhost

IRP host. IRP box host name or IPv4/IPv6 address.

Default value: localhost

4.1.1.9 db.password

The password used to connect to a MySQL server.

4.1.1.10 db.port

TCP port number to use for the connection to a MySQL database.

Default value: 3306

4.1.1.11 db.username

User name used to connect to a MySQL server.

Default value: irp

4.1.2 Global parameters

4.1.2.1 `global.agg_ipv4_max`

Defines the maximum IPv4 aggregate mask for the improved prefixes advertised by Bgpd. If `global.aggregate` is enabled, IRP will not announce any prefix with the mask that's less than the mask defined in `global.agg_ipv4_max`.

Possible values: 8–24

Default value: 16

4.1.2.2 `global.agg_ipv6_max`

Defines the maximum IPv6 aggregate mask for the improved prefixes advertised by Bgpd. If `global.aggregate` is enabled, IRP will not announce any prefix with the mask that's less than the mask defined in `global.agg_ipv6_max`.


Possible values: 32–48

Default value: 32

4.1.2.3 `global.aggregate`

If this parameter is ON IRP analyzes entire aggregates. When one can be improved the aggregate will be announced by Bgpd (refer to `bgpd.updates.split`, `bgpd.peer.X.updates.limit.max`). Largest possible prefixes are used when aggregates exceed `global.agg_ipv4_max` & `global.agg_ipv6_max` parameters for IPv4 and IPv6 accordingly. An aggregate dictionary is maintained by IRP in order to support this capability. The aggregate dictionary is periodically populated by `refresh_asn` or Bgpd (if Bgpd has sufficient information, by means of at least one full-view BGP peering session).

Disabling this parameter instructs IRP to operate with the smallest possible disaggregated prefixes (/24 for IPv4 and /48 IPv6).

 Switching this parameter ON/OFF might leave a large number of small prefixes in the aggregate dictionary with undesirable side effects. The contents of the dictionary should be reviewed and refreshed in case it lists undesirable prefixes. To prevent issues IRP improvements should be cleared before applying this change. Clearing improvements ensures that disaggregated prefixes announced by IRP are not present on the network.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

4.1.2.4 `global.bw_overusage`

Enables or disables automatic Flowspec throttling policies for prefixes with excessive bandwidth spikes.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.5 `global.exchanges`

Defines the path to the Exchanges configuration file.

Default value: `/etc/noction/exchanges.conf`

Recommended value: `/etc/noction/exchanges.conf`

4.1.2.6 global.exchanges.auto_config_interval

Defines the Internet Exchanges auto re-configuration interval in seconds. Auto re-configuration is applied to providers of type Internet Exchange with enabled auto re-configuration. Refer [peer.X.auto_config](#).

Possible values: 3600–2678400

Default value: 86400

4.1.2.7 global.failover

Enables and disables failover feature.

❌ Enabling failover requires extensive preparatory activities. Refer [IRP Failover \(Not supported in IRP Lite\)](#), [Failover configuration \(Not supported in IRP Lite\)](#), [Setup Failover wizard \(Not supported in IRP Lite\)](#) for details.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.8 global.failover.log

Path to failover log file.

Default value: /var/log/irp/failover.log

4.1.2.9 global.failover.use_communities

This parameter allows monitoring presence of improvements from master instance by using communities specified in [bgpd.peer.X.master_communities](#) (refer [bgpd.peer.X.slave_communities](#)).

Master and slave communities in a BGP peering should be unique if the parameter is enabled.

If a network setup requires the same community to be used on both master and slave instances then the parameter should be disabled and failover shall use [bgpd.peer.X.master_localpref](#) only.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.2.10 global.failover_identity_file

Defines the path to failover master identity file. Set this parameter only when a SSH key file other than `~/.ssh/id_rsa` is used. Refer [global.failover](#).

Possible values: Path to identity file

4.1.2.11 global.failover_role

Defines the role of the IRP instance in a failover configuration. Known roles are Master and Slave. Used only when failover is enabled. Refer [global.failover](#).

Possible values: 0 (Master), 1 (Slave)

Default value: 0

4.1.2.12 global.failover_slave.ip

Defines the IPv4 address of the slave IRP instance used for configuration sync in a failover configuration. Used only when failover is enabled. Refer [global.failover](#).

Possible values: Valid IPv4 address

Example value: 10.10.0.2

4.1.2.13 global.failover_slave.ipv6

Defines the IPv6 address of the slave IRP instance used for configuration sync in a failover configuration. Used only when failover is enabled. Refer [global.failover](#).

Possible values: Valid IPv6 address

Example value: 2001:db8::ff00:42:8329

4.1.2.14 global.failover_slave.port

Defines the SSH port of the slave node in a failover configuration. Used only when failover is enabled. See also [global.failover](#).

Possible values: 1-65535 (Valid port number)

Default value: 22

4.1.2.15 global.failover_timer_fail

Defines the period of time in seconds during which master is considered alive before the slave becomes active in a failover configuration. Used only when failover is enabled.

See also [global.failover](#).

Possible values: 30-3600

Default value: 300

4.1.2.16 global.failover_timer_failback

Defines the period of time in seconds for slave to switch back to standby after it detects master became alive in a failover configuration. Used only when failover is enabled.

See also [global.failover](#).

Possible values: 30-3600

Default value: 300

4.1.2.17 global.flowspec

Enables/disables Flowspec capability globally.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.18 global.flowspec.pbr

Enables/disables use of Flowspec policies instead of Policy Based Routing. This is available only when Flowspec is enabled. Refer [global.flowspec](#).

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.19 global.frontend_acl

Defines whether access to the IRP Frontend must be restricted and controlled by an ACL. See also [global.frontend_acl_ips](#).

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.20 global.frontend_acl_ips


Defines the IPs or networks that should be allowed to access the IRP Frontend.

Possible values: Valid IPv4, IPv6 address or subnet definition in CIDR format

Default value: 0.0.0.0/0 ::/0

4.1.2.21 global.ifstats

Defines whether interface statistics should be collected by Irpstatd or IRP collectors.

 Starting with IRP 3.8 IRP collector(s) can retrieve interface aggregated statistics. This is designed to simplify IRP by reducing its number of components (Irpstatd is being considered for deprecation) and to address measurement discrepancies issues caused by collecting detailed and aggregated data by means of separate processes (Collector(s) and Irpstatd). To preserve backwards compatibility old behavior is preserved by default. This will change in future releases.

Possible values: 0 (Irpstatd), 1 (Collector)

Default value: 1

4.1.2.22 global.ignored.asn

Defines the list of ASNs to be ignored by IRP.

Format:

1. Space-separated list of AS numbers that must be ignored by IRP.
2. Absolute path to a newline-separated file containing a list of AS numbers that must be ignored by IRP.

This parameter should list all AS numbers that must be ignored by Irpspand, Irpflowd, Explorer and the Core. No improvements will be performed by the Core for prefixes that are announced by ASNs listed within this parameter. No data will be gathered by Irpspand/Irpflowd for any source or destination IPv4/IPv6 address that are announced by ASNs that are listed within this parameter. No probes will be sent by Explorer to any destination IPv4/IPv6 address that are announced by ASNs listed within this parameter.

Refer also [global.ignored_communities](#), [global.ignorednets](#).

Possible values: 1 - 4294967295.

4.1.2.23 global.ignored.unannounced

Prefixes which aren't present in a BGP routing table wouldn't be analyzed nor optimized by Outbound optimization algorithms.

As example, spoofed IP addresses may not belong to advertized prefixes but the traffic itself may have significant volume and be optimized by IRP.

Therefore, in such a case the recommendation is to enable the parameter to do not optimize traffic from unknown origins.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.2.24 global.ignored_communities

Defines the list of BGP Community attributes the network uses to mark prefixes that IRP must ignore. IRP monitors routes marked with at least one of these BGP Community attributes and dynamically updates a list of prefixes to ignore. The decision what prefixes to mark with one of these attributes is applied on the network and network operators do not need to be explicitly set in IRP too.

This (dynamic) list of prefixes will be ignored by Irspsand, Irfpflowd, Explorer and the Core. No improvements will be performed by the Core for such prefixes. No data will be gathered by Irspsand/Irfpflowd for any source or destination IPv4/IPv6 address within such prefixes. No probes will be sent by Explorer to any destination IPv4/IPv6 address within such prefixes.

Refer also [global.ignored.asn](#), [global.ignorednets](#).

Possible values: valid BGP community attribute list.


4.1.2.25 global.ignorednets

Defines the list of networks to be ignored by IRP.

Format:

1. Space-separated list of local IPv4/IPv6 prefixes that should be ignored by IRP.
2. Absolute path to a newline-separated file containing a list of IPv4/IPv6 prefixes that should be ignored by IRP.

If netmask is not clearly specified, the system assumes /32 for IPv4 addresses, and /128 for IPv6 addresses.

-  224.0.0.0/3 is always ignored.
- IPv6 probing is performed only to 2000::/3 address range (see: [IPV6 Address Space](#))
- All IPv4/IPv6 addresses assigned to IRP server are automatically added to ignored networks

This parameter lists all IPv4/IPv6 addresses that should be ignored by Irspsand, Irfpflowd, Explorer and the Core. No improvements will be performed by the Core for /24 subnets listed within this parameter as /24 or a less specific network. No data will be gathered by Irspsand/Irfpflowd for any source or destination IPv4/IPv6 address listed within this parameter. No probes will be sent by the Explorer to any destination IPv4/IPv6 address listed within this parameter.

Refer also [global.ignored.asn](#), [global.ignored_communities](#).

Possible values: See above.

Default value: 127.0.0.0/8 10.0.0.0/8 169.254.0.0/16 172.16.0.0/12 192.168.0.0/16
100.64.0.0/10 203.0.113.0/24 198.18.0.0/15 192.0.0.0/24 192.0.2.0/24
2001:db8::/32

Recommended value: See default value.

- i** Recommended networks to ignore are:
- networks from: [Section 3 of RFC5735](#), as listed in the default value above
 - networks from: [Section 2 of RFC3849](#)

4.1.2.26 global.improve_mode

Defines the IRP operating mode (see [IRP Optimization modes](#)).

This parameter adjusts the priorities and rules for prefix improvements. Prefixes can be improved in three different ways:

1. **Performance optimization:** Decrease loss, then decrease latency;
2. **Cost optimization:** Decrease loss, then decrease cost while keeping the latency within the preconfigured level;

- i** Loss, cost and latency are improved in strict order, depending on the selected operating mode.

Possible values: 1, 2 (see above).

Default value: 1

See also: [Commit Control \(Not supported in IRP Lite\)](#), `core.commit_control`, `peer.X.cc_disable`

4.1.2.27 global.inbound.injection

Defines how IRP is used to inject Inbound improvements.

In “Pull” mode an external script is used to pull Inbound improvements from IRP API and re-configure routers’ access lists.

Example script `/usr/bin/irpTransitPull.pl` could be used to work in the Pull mode.

Possible values: 0 - Pull, 1 - BGP

Default value: 1

4.1.2.28 global.inbound_conf

Defines path to file with configured inbound prefixes.

Possible values: path to file

Default value: `/etc/noction/inbound.conf`

Recommended value: `/etc/noction/inbound.conf`

4.1.2.29 global.inbound_transit

Enables or disables inbound optimization of transiting traffic. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 0 (Disable), 1 (Enable)

Default value: 0

4.1.2.30 global.ipv6_enabled

Defines whether IPv6 is enabled in the system. Currently used for the Frontend only. Even if IPv6 is enabled via this parameter, other components configuration must be adjusted for IPv6 as well.

⊖ IRP prior to 3.3-2 allows configuration of both IPv4 and IPv6 sessions on a single router, while IRP 3.3-2 requires definition of two separate BGP sessions. It is recommended to split BGP sessions in the form of two different routers before upgrading to 3.3-2, otherwise IRP configuration will not be valid. Make sure the newly added router is linked to corresponding providers to ensure IPv6 optimization works properly. InternalMon for IPv6 session monitoring requires correct configuration of BGP MIB (IPv6) mode (see `peer.X.mon.ipv6.internal.mode` parameter). Currently support for Brocade, Cisco and Juniper MIBs is available.

Possible values: 0, 1

Default value: 1

4.1.2.31 global.master_management_interface

Defines the management network interface. In most cases it is the same as the probing interface. When failover is enabled (`global.failover`) slave's management interface (`global.slave_management_interface`) must be configured too.

Possible values: any valid system network interface name

Default value: eth0

4.1.2.32 global.master_probing_interface

Defines the probing network interface. In most cases it is the same as the management interface. When failover is enabled (`global.failover`) slave's probing interface (`global.slave_probing_interface`) must be configured too.

ⓘ This parameter is used only for displaying operating system interface(s) status in Frontend. It does not actually configure Explorer behavior, which depends on `Providers` settings.

Possible values: any valid system network interface name

Default value: eth0

4.1.2.33 global.master_rd

Specifies the Routing Domain that hosts the master node of IRP in a failover configuration. By default RD=1 hosts IRP nodes.

ⓘ It is recommended that master node of IRP is hosted in RD=1 at all times.

Possible values: 1-100

Default value: 1

Recommended value: 1

4.1.2.34 global.nonintrusive__bgp

Instructs the system to run in a non-intrusive BGP mode (see [IRP Operating modes](#)).

All improvements made in a non-intrusive mode, will not be automatically injected into the routers.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1 at first start, 0 after manual tests are performed and the system is ready to go into intrusive mode

4.1.2.35 global.offpeak__hour


Defines customer's network usual off-peak hour of the day.

Possible values: 0 - 23

Default value: 3

4.1.2.36 global.outbound

Enables or disables outbound optimization.

 Outbound optimization is disabled only for standalone Inbound optimization is used.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.2.37 global.png.datadir

Defines the file-system directory path for storing image files (Graphs).

Default value: /usr/share/irp/web/RRD

Recommended value: /usr/share/irp/web/RRD

4.1.2.38 global.policies

Defines the path to the Routing Policies (1.2.9) configuration file.

Default value: /etc/noction/policies.conf

Recommended value: /etc/noction/policies.conf

4.1.2.39 global.rd__rtt

Defines the latency distances between routing domains in the format rda:rdb:rtt where rda is the id assigned to one routing domain, rdb is the id assigned to the second routing domain and rtt represents the round trip time in milliseconds between them. rda and rdb must be different and assigned to providers. IRP assumes that distance from rda to rdb is equal to distance from rdb to rda.

The parameter takes a collection of such triplets that will define all the available inter-datacenter links between routing domains.

⊖ It is important that the RTT value between routing domains is accurate. In case the value differs significantly from the correct value IRP will make improvement decision based on incorrect information and it will make unnecessary global improvements that will reroute more traffic via inter-datacenter links.

Refer `peer.X.rd`, `peer.X.flow_agents`, `bgpd.rd_local_mark`

4.1.2.40 global.rrd.age_max

Defines the maximum trusted interface load data age (seconds)

Data older than this interval will not be trusted by IRP. If interface rate for each provider link has not been updated for a specified amount of time, then IRP behavior will be changed as follows:

- If provider's `limit_load` is set, no further Cost/Performance improvements will be performed to that provider
- Commit Control will not perform further in/out improvements for this provider

Possible values: 120–240

Default value: 120

Recommended value: 120. Should be increased only if frequent SNMP timeouts occur.

4.1.2.41 global.rrd.datadir

Defines the file system directory path for storing RRD database files.

Possible values: valid directory

Default value: `/var/spool/irp`

Recommended value: `/var/spool/irp`

4.1.2.42 global.slave_management_interface

Defines the management network interface for slave node in a failover configuration. In most cases it is the same as the probing interface. When failover is disabled (`global.failover`) the parameter is not used.

See also `global.master_management_interface`.

Possible values: any valid system network interface name

Default value: `eth0`

4.1.2.43 global.slave_probing_interface

Defines the probing network interface for the slave node in a failover configuration. In most cases it is the same as the management interface. When failover is disabled (`global.failover`) the parameter is not used.

See also `global.master_probing_interface`.

Possible values: any valid system network interface name

Default value: `eth0`

4.1.2.44 global.slave_rd

Specifies the Routing Domain that hosts the slave node of IRP in a failover configuration. By default RD=1 hosts IRP nodes.

Possible values: 1-100

Default value: 1

Recommended value: 1

4.1.2.45 global.user_directories_conf

Defines the path to the User Directories configuration file.

Default value: /etc/noction/user_directories.conf

Recommended value: /etc/noction/user_directories.conf

4.1.3 API daemon settings

4.1.3.1 apid.listen.master_ip

Defines the IPv4/IPv6 address of the API. Allows IRP API calls to target another node on the network. If no value provided then localhost is used. When failover is enabled ([global.failover](#)) slave's listen IP ([apid.listen.slave_ip](#)) must be configured too.

Possible values: IPv4/IPv6 address

Default value: 127.0.0.1 ::1

4.1.3.2 apid.listen.port

Defines the TCP port on which the irpapid is listening. If no value provided port 10443 is used.

Possible values: 1-65535

Default value: 10443

4.1.3.3 apid.listen.slave_ip

Defines the IPv4/IPv6 address of the API of the slave node in a failover configuration. Allows IRP API calls to target another node on the network. If no value provided then localhost is used. When failover is disabled ([global.failover](#)) slave's listen IP is not used.

See also [apid.listen.master_ip](#).

Possible values: IPv4/IPv6 address

Default value: 127.0.0.1 ::1

4.1.3.4 apid.log

Defines the file-system path to the irpapid log file.

Default value: /var/log/irp/irpapid.log

4.1.3.5 apid.log.level

Defines the logging level for the irpapid service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.3.6 apid.maxthreads

Defines the number of threads that irpapid is allowed to run. If no value provided 50 threads are used.

Possible values: 1-300


Default value: 50

4.1.3.7 apid.path.mtr

System path to MTR utility.

Possible values: /valid/path

Default value: /usr/sbin/mtr


 Default value varies depending on platform

4.1.3.8 apid.path.ping

System path to ping utility.

Possible values: /valid/path

Default value: /bin/ping


 Default value varies depending on platform

4.1.3.9 apid.path.ping6

System path to ping for IPv6 utility.

Possible values: /valid/path

Default value: /bin/ping6


 Default value varies depending on platform

4.1.3.10 apid.path.traceroute

System path to traceroute utility.

Possible values: /valid/path


Default value: /usr/bin/traceroute

 Default value varies depending on platform

4.1.4 Bgpd settings


4.1.4.1 `bgpd.as_path`

Defines the way Bgpd handles the AS-PATH attribute in the outgoing announcements. The selected options are evaluated in the configured order and the first valid AS-PATH will be used.

 Note that IRP will refuse to announce improvements when all configured options fail to produce a valid AS-PATH. Option 0 allows announcements with empty AS-PATH but this is undesirable for other reasons noted below.

Keeping a correct AS-PATH can be a requirement for some NetFlow/sFlow processing since the provider AS or the destination AS for the corresponding flow record is taken from the BGP routing table.

Some installations use outgoing filters that allow empty AS-Path while redistributing iBGP routes to upstreams. In such cases, Bgpd must be configured not to advertise the improvements with an empty AS-PATH to prevent further redistribution of the improvements to upstream routers.

 The reconstructed AS-path does not always correspond to the actual BGP AS-path.

Refer also to `peer.X.ipv4.next_hop_as`, `peer.X.ipv6.next_hop_as`, `peer.X.aspath_for_ix`, `bgpd.as_path_borrowing`.

Examples:

`bgpd.as_path=2 3` - this will instruct Bgpd to take first path from the database (reconstructed AS-Path). If that path is empty, then an AS-path with the provider's AS number and the prefix AS number should be composed.

`bgpd.as_path=3 0` - this will instruct Bgpd to compose an AS-path with the provider's AS number and the prefix AS number. If AS-Path remains empty, the improvements with an empty AS-Path are announced.

Possible values: Space separated list of options in the order of preference

- 0 - Allow empty AS-PATH
- 2 - Use non-empty reconstructed AS-PATH (Announce AS-path reconstructed from traceroute)
- 3 - Reconstruct AS path with provider ASN and prefix origin ASN
- 4 - Use AS-Path from BMP
- 5 - Use AS-Path from BGP Alternative paths (RFC 7911)

 Second algorithm produces meaningful AS-PATH only when `explorer.trace.all` is enabled

Default value: 5 4 2 3

4.1.4.2 `bgpd.as_path_borrowing`

Allows BMP to borrow AS path from other Provider with the same autonomous system number.

i The parameter modifies only 4th algo of `bgpd.as_path` parameter.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.4.3 `bgpd.db.timeout.withdraw`

Defines the time period (in seconds) before prefixes are withdrawn from the routing tables, after a database failure. This allows BGP daemon to function independently for a period of time after the IRP database becomes inaccessible due to a failure or manual intervention.

Possible values: 600 - 21600(6 hours)

Default value: 14400

Recommended values: 3600-14400

4.1.4.4 `bgpd.full_control`

Sets default behavior of IRP in regards to inbound prefixes control and specifically:

- only announce improvements or
- only announce improvements and include all allowed providers or
- fully control inbound prefixes by always announcing to allowed providers.

i This system wide default behavior can be overridden at inbound prefix level by specifying desired parameter value for `inbound.rule.X.full_control`.

Refer to for details.

Possible values: 0 (Improvements), 1 (If improved), 2 (All)

Default value: 0

4.1.4.5 `bgpd.improvements.remove.hold_time`

Defines the time interval (in seconds) for deleting (from the IRP database) the improvements affected by "Remove on next-hop eq" or "Remove on aggregate withdraw" conditions (see `bgpd.improvements.remove.next_hop_eq` and `bgpd.improvements.remove.withdrawn`).

This reduces the effects of route flapping to improvement cleanup.

Verification is performed on each BGP Scan, so that the values that are lower than the value of the `bgpd.scaninterval` parameter are not taken into account. The higher the values - the longer the time interval needed for improvements, which are still valid after aggregate route is withdrawn or on equal next-hop.

Bgpd does improvements check against the routers received and sent via iBGP in periodic time intervals. The process is referred to as the BGP Scan process.

Time interval between BGP Scannings can be configured in `bgpd.scaninterval`.

Possible values: 1 - 1000 sec

Default value: 60

Recommended values: 30 - 120

4.1.4.6 bgpd.improvements.remove.next_hop_eq

Instructs Bgpd to remove a prefix from improvements when aggregate route's next hop has changed and points to the same next hop as the improvement.

⊖ This parameter shouldn't be enabled when `bgpd.updates.split` is disabled in a multi-routing domain configuration.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 0

4.1.4.7 bgpd.improvements.remove.withdrawn

Instructs Bgpd to remove prefix from improvements when aggregate is being withdrawn from the router.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.4.8 bgpd.improvements.strip_non_irp_communities

Instructs Bgpd to exclude from Updates of IRP improvements other communities except those configured in IRP.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.4.9 bgpd.log

Defines the complete path to the Bgpd log file

Possible values: full path to log file

Default value: /var/log/irp/bgpd.log

4.1.4.10 bgpd.log.level

Defines the logging level for the Bgpd service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.4.11 bgpd.mon.guardtime

Defines the Bgpd Monitoring guard time. All the controlled IP addresses must respond within the specified amount of time, for the provider to be restored from the FAIL state. In FAIL state, all improvements are withdrawn from that provider.

It is recommended that `bgpd.mon.guardtime` is set as a double value of the `bgpd.mon.holdtime`.

Possible values: 10-600

Default value: 60

Recommended value: 60

4.1.4.12 bgpd.mon.holdtime

Defines the Bgpd Monitoring holdtime.

If any controlled IP address does not respond to all the requests during the holdtime, then corresponding provider enters the FAIL state. In FAIL state, all the improvements are withdrawn from that provider.

Possible values: 10–60

Default value: 30

Recommended value: 30

4.1.4.13 bgpd.mon.internal.flap_guardtime

Defines time interval to protect from BGP session flapping.

Internal monitor will keep FAIL state until BGP session stays established within the configured period of time.

Possible values: 0–86400

Default value: 0

Recommended value: 600

4.1.4.14 bgpd.mon.keepalive

Defines the Bgpd Monitoring keepalive interval (in seconds) between consequent ICMP Echo Requests to single controlled IP address.

Possible values: 1–10

Default value: 10

Recommended value: 5

4.1.4.15 bgpd.mon.longholdtime

Defines the Bgpd Monitoring long holdtime.

If any controlled IP address does not respond to all the requests during the long holdtime, then corresponding provider enters the FAIL state. In FAIL state, all the improvements are withdrawn from that provider.

Possible values: 60–3600


Default value: 1800

Recommended value: 1800

4.1.4.16 bgpd.monitor.type

Defines how IRP monitors Improvements to Internet Exchanges:

- by using coarse-grained internal monitors that merely validate an IX peer is live or
- by using fine-grained prefix monitors for each IX improvement that validate that the IX peer still advertises the improved prefix.

 Prefix monitors might consume significant router CPU resources when relying on SNMP to determine if an IX peer advertises the improved prefix and the number of IX improvements is large.

Refer to `bgpd.prefix.monitor.interval` for details.

Possible values: 0 (Use internal monitor), 1 (Use prefix monitor)

Default value: 0

4.1.4.17 bgpd.policy.cascade.amount

Defines the maximum number of downstream AS for cascading policies. If IRP identifies more downstream AS from the designated AS the policy for those AS will not be enforced.


Possible values: 1-1000000

Default value: 1000

Recommended value: 1000

4.1.4.18 bgpd.prefix.monitor.interval

Prefix monitor tracks at given interval in seconds if a peering partner on an Exchange is still announcing the prefix that IRP improved through it.


 This is intended to avoid cases when after IRP makes an Improvement through a peer on an IX the peer stops announcing/servicing the route.

Possible values: 1-10

Default value: 10

4.1.4.19 bgpd.prefix.monitor.search_interval

Defines the interval between retries of prefix monitor failed initialization attempts in case a prefix isn't advertised by an IX peering partner or if a SNMP error occurs.


 During this time IRP will not be able to make improvements through the affected IX peers.

Possible values: 300-3600

Default value: 300

4.1.4.20 bgpd.prefixlist.asn

Specifies a collection of AS numbers that are analyzed for inbound optimization of transiting traffic. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

 IRP ignores prefixes /25 and shorter for IPv4 and /65 and shorter for IPv6 being present in network's routing table. This means that traffic belonging to these small prefixes are accounted under the immediately larger prefix that fits the above criteria.

Possible values: list of valid AS numbers

4.1.4.21 bgpd.prefixlist.prefixes

Specifies a collection of IPv4 or/and IPv6 prefixes in CIDR notation that are analyzed for inbound optimization of transiting traffic. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: list of valid CIDR prefixes

4.1.4.22 bgpd.rd_local_mark

Specifies a marker to distinguish local improvements from global improvements in the case of multiple routing domains optimization. Parameter represents a valid value for BGP community attribute of the form X:Y. Value in `bgpd.rd_local_mark` is APPENDED to communities attribute. Refer [global.rd_rtt](#), [peer.X.rd](#), [peer.X.flow_agents](#), [rd.X.community.local](#).


Possible values: X:Y

Default value: 65535:1

4.1.4.23 bgpd.retry_probing.new.bmp_path_change

Defines on what new provider AS Path changes to re-probe an already improved prefix. The possible options are:

- 0: Disabled
- 1: On major AS-Path change
- 2: On any AS-Path change

 Major AS Path changes are those traversing a different set of Autonomous Systems. AS Path changes such as prepended ASN are ignored when only major AS Path changes are monitored.

Refer also to `bgpd.retry_probing.old.bmp_path_change`, `peer.X.bmp`.


Possible values: 0 - 2

Default value: 0

4.1.4.24 bgpd.retry_probing.old.bmp_path_change

Defines on what old provider AS Path changes to re-probe an already improved prefix. The possible options are:

- 0: Disabled
- 1: On major AS-Path change
- 2: On any AS-Path change

 Major AS Path changes are those traversing a different set of Autonomous Systems. AS Path changes such as prepended ASN are ignored when only major AS Path changes are monitored.

Refer also to `bgpd.retry_probing.new.bmp_path_change`, `peer.X.bmp`.

Possible values: 0 - 2

Default value: 0

4.1.4.25 bgpd.scaninterval

The interval in seconds between the execution of the BGP Scan process.
BGP scan is used for:

- checking the improvements belonging to aggregated routes
- checking the improvements for **aggregate withdrawn**(4.1.4.7) and for **aggregate next-hop equal to improvements next-hop**(4.1.4.6) conditions
- checking the improvements for changed BGP attributes
- checking for changes for the improvements to be announced to/withdrawn from iBGP neighbors

Possible values: 10-100

Default value: 20

Recommended value: 60

4.1.4.26 bgpd.snmp.concurrent_requests

Specifies the number of allowed concurrent (in-flight) SNMP requests sent to a router.

Possible values: 1-2000

Default value: 10

4.1.4.27 bgpd.snmp.packets_interval

Time interval in milliseconds between transit improvement's monitor SNMP packets. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 1-100

Default value: 10

4.1.4.28 bgpd.snmp.simultaneous

The number of the simultaneous PDUs that will be contained in a SNMP request.

Possible values: 1-300

Default value: 10

Recommended value: 10

4.1.4.29 bgpd.transit.monitor.election_interval

Transit monitor election interval as a factor of reconfirm intervals. Refer [bgpd.transit.monitor.fast_reconfirm_interval](#).

Possible values: 1-10

Default value: 4

4.1.4.30 bgpd.transit.monitor.fast_reconfirm_interval

Transit monitor fast reconfirm interval in seconds. The reconfirm interval sets the periodicity by which transit monitors verify presence of alternative routes from other providers for transit improvements. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 1-60

Default value: 15

4.1.4.31 bgpd.transit.monitor.retries

Number of SNMP retries before a timeout of transit improvement's monitor. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 1-2000

Default value: 3

4.1.4.32 bgpd.transit.monitor.timeout

Timeout in milliseconds of individual SNMP requests used to monitor transit improvements. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 1-2000


Default value: 1000

4.1.4.33 bgpd.updates.split

Instructs Bgpd to split advertised prefixes in two equal parts (e.g /24 is split into two /25 prefixes).

This parameter should be enabled in order to preserve the original BGP UPDATE attributes received from the corresponding aggregates.

Refer to [global.aggregate](#), [global.agg_ipv4_max](#), [global.agg_ipv6_max](#) parameters.

 If this option is enabled, the number of announced prefixes will be twice the [core.improvements.max](#).

If the [bgpd.peer.X.updates.limit.max](#) parameter value is established, then the limitation is set on the total amount of announced prefixes, AFTER split. For example, if the value of [core.improvements.max](#) is set to 10000 and [bgpd.peer.X.updates.limit.max](#) is set to 5000, then the amount of the improvements towards this particular provider is no more than 2500, split onto 5000.


Possible values: 0 (OFF), 1 (ON)


Default value: 1

Recommended value: 1

4.1.5 BGP sessions settings

4.1.5.1 bgpd.peer.X.as

 Mandatory for each iBGP session definition.

 Parameter changes cause reset of BGP session.

Defines the Autonomous System Number for the iBGP session.

Possible values: 1-4294967295

4.1.5.2 bgpd.peer.X.blackholing.ipv4.next_hop

Defines IPv4 address which will be used as next_hop in BGP UPDATE for Blackholing routes sent to this router.

The next-hop address should be known by the router.

Possible values: IPv4 address

4.1.5.3 bgpd.peer.X.blackholing.ipv6.next_hop


Defines IPv6 address which will be used as next_hop in BGP UPDATE for Blackholing routes sent to this router.

The next-hop address should be known by the router.

Possible values: IPv6 address

4.1.5.4 bgpd.peer.X.blackholing.localpref


Defines localpref value used by IRP in BGP UPDATE for Blackholing routes sent to this router.

 Avoid collisions of localpref values assigned to IRP both within its configuration and/or on customer's network.

Possible values: 0 - 4294967295

Default value: 100

4.1.5.5 bgpd.peer.X.cap_4byte_as

 Parameter changes cause reset of BGP session.

Defines usage of 16 or 32-bit autonomous system numbers. Capability can be negotiated during session setup or forced to either 16 or 32 bits.

- 1 - Negotiated on OPEN
- 2 - Always 16-bit AS path
- 3 - Always 32-bit AS path

When the router is operating in legacy mode and does not negotiate capabilities but still sends 32-bit AS Path then Bgpd considers this a malformed AS_PATH attribute and disconnects the session. To avoid this the parameter must be set to force use of 32bit AS numbers.

For example: A BGP session with IRP with “disable-capability-negotiation” option configured on Vyatta router.

As a result, the BGP session is established and then teared down with log messages:

Listing 4.1: Error log

```
Jan 29 10:20:40.777965 WARN: BGP session RTR/IPv4 (10.0.0.1 AS 65530)
  Incoming UPDATE error: Invalid elements ignored. Malformed AS_PATH
Jan 29 10:20:40.778021 ERROR: BGP session RTR/IPv4 (10.0.0.1 AS 65530)
  Incoming UPDATE error: malformed AS_PATH xxxxxxxx
```

The solution is to set `bgpd.peer.RTR.cap_4byte_as = 3`, then the BGP session succeeds.

Possible values: 1 (negotiate), 2 (force 16bit), 3 (force 32bit)

Default value: 1

4.1.5.6 bgpd.peer.X.flowspec

Enables/disables FlowSpec capability for BGP session.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.5.7 bgpd.peer.X.inbound.ipv4.next_hop

Defines IPv4 address which will be used as `next_hop` in BGP UPDATE for inbound improvements/announcements towards this router.

The next-hop address should be known by the router (refer [Routers configuration for Inbound](#)).

An inbound rule can be configured with a rule-specific next-hop. Refer to [inbound.rule.X.next_hop](#).

Possible values: IPv4 address

4.1.5.8 bgpd.peer.X.inbound.ipv6.next_hop

Defines IPv6 address which will be used as `next_hop` in BGP UPDATE for inbound improvements/announcements towards this router.

The next-hop address should be known by the router (refer [Routers configuration for Inbound](#)).


An inbound rule can be configured with a rule-specific next-hop. Refer to [inbound.rule.X.next_hop](#).

Possible values: IPv6 address

4.1.5.9 bgpd.peer.X.inbound.master_localpref

Defines localpref value used by IRP for inbound improvements for this router.

Assigned localpref value should allow Inbound Improvement to become best route.

 Avoid collisions of localpref values assigned to IRP both within its configuration and/or on customer's network.

Possible values: 0 - 4294967295

Default value: 102

4.1.5.10 bgpd.peer.X.inbound.slave_localpref

Defines localpref value used by IRP for inbound improvements for this router.

Assigned localpref value should allow Inbound Improvement to become best route.

⊖ Avoid collisions of localpref values assigned to IRP both within its configuration and/or on customer's network.

Possible values: 0 – 4294967295

Default value: 101

4.1.5.11 bgpd.peer.X.keepalive

Defines the session keepalive time (sec). See [RFC1771](#) for details. Hold time is calculated as keepalive * 3.

Possible values: 1–100

Default value: 60

Recommended value: 1/3 holdtime

4.1.5.12 bgpd.peer.X.listen

Instructs the BGP daemon to listen for incoming sessions. It must be set to 1 to be [RFC1771](#) compliant. It can be set to 0 to resolve specific issues.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.5.13 bgpd.peer.X.master_communities

Defines the BGP Community that will be appended by Bgpd to all advertised prefixes. The format is: “X:Y”.

⊖ Avoid collisions of communities values assigned to IRP both within its configuration and/or on customer's network.

When failover is enabled ([global.failover](#)) slave's BGP Community ([bgpd.peer.X.slave_communities](#)) must be configured too.


In case Bgpd receives the full or partial RIB (Routing Information Base), values for: **MED**, **Origin**, **LocalPref** and **Communities** are taken from a less-specific Aggregate route. If values for **MED**, **Origin**, **LocalPref** are set in config, it will override any value from Aggregate route. If value for **Communities** is set in config, it will be appended to communities from Aggregate route.

4.1.5.14 bgpd.peer.X.master_localpref

Defines the local-preference value for prefixes (improvements) announced by Bgpd.

⊖ Avoid collisions of localpref values assigned to IRP both within its configuration and/or on customer's network.

When failover is enabled ([global.failover](#)) slave's BGP LocalPref ([bgpd.peer.X.slave_localpref](#)) must be configured too.


 If failover is enabled, master's LocalPref value must be greater than slave's LocalPref value.


In case Bgpd received the full or partial RIB (Routing Information Base), values for **MED**, **Origin**, **LocalPref** and **Communities** will be taken from less-specific Aggregate route. If values for **MED**, **Origin**, **LocalPref** are set in config, it will override any value from Aggregate route. If value for **Communities** is set in config, it will be appended to communities from Aggregate route.

Possible values: 0 – 4294967295

Default value:

4.1.5.15 bgpd.peer.X.master_our_ip


 Mandatory for IPv4 BGP session.


 Parameter changes cause reset of BGP session.

Defines IPv4 address of IRP server end of this iBGP session. When failover is enabled (`bgpd.peer.X.slave_our_ip`) slave's IP address (`bgpd.peer.X.slave_localpref`) must be configured too.

Possible values: IPv4 address

4.1.5.16 bgpd.peer.X.master_our_ipv6

 Mandatory for IPv6 BGP session.

 Parameter changes cause reset of BGP session.

Defines IPv6 address of IRP server end of this iBGP session. When failover is enabled (`bgpd.peer.X.slave_our_ip`) slave's IPv6 address (`bgpd.peer.X.slave_our_ipv6`) must be configured too.


Possible values: IPv6 address


4.1.5.17 bgpd.peer.X.master_password

Defines iBGP session's password. When failover is enabled (`bgpd.peer.X.slave_our_ip`) slave's local IPv6 address (`bgpd.peer.X.slave_password`) must be configured too.

Possible values: up to 80 alphanumeric characters

4.1.5.18 bgpd.peer.X.master_peer_ip


 Mandatory for IPv4 BGP session.


 Parameter changes cause reset of BGP session.

Defines iBGP session's router's IPv4 address.

Possible values: IPv4 address

4.1.5.19 bgpd.peer.X.master_peer_ipv6


 Mandatory for IPv6 BGP session.


 Parameter changes cause reset of BGP session.

Defines iBGP session's router's IPv6 address.

Possible values: IPv6 address

4.1.5.20 bgpd.peer.X.master_router_id

 Mandatory for IPv6 BGP session either as standalone master or when failover is enabled and the value should be different from `bgpd.peer.X.slave_router_id`.

 Parameter changes cause reset of BGP session.

Defines IRP server's router ID. The BGP router ID is used in the BGP algorithm for determining the best path to a destination where the preference is given to the BGP router with the lowest router ID.

Possible values: 4-byte value in the IPv4 address format. Any valid IPv4 address can be used.

4.1.5.21 bgpd.peer.X.med

Defines the Multi-Exit Discriminator (MED) value for prefixes (improvements) announced by Bgpd.

In case Bgpd received the full or partial RIB (Routing Information Base), values for **MED**, **Origin**, **LocalPref** and **Communities** will be taken from less-specific Aggregate route. If values for **MED**, **Origin**, **LocalPref** are set in config, it will override any value from Aggregate route. If value for **Communities** is set in config, it will be appended to communities from Aggregate route.


Possible values: 0 - 4294967295

4.1.5.22 bgpd.peer.X.origin

Defines the Origin value for prefixes announced by Bgpd.

Possible values: 0 (IGP), 1 (EGP), 2 (INCOMPLETE)

4.1.5.23 bgpd.peer.X.shutdown

 Parameter changes cause reset of BGP session.

Defines whether the corresponding iBGP session is active or shutdown.

Possible values: 0 (Active), 1 (Shutdown)

Default value: 0

4.1.5.24 bgpd.peer.X.slave_communities

Defines the BGP Community that will be appended by Bgpd on the slave node of a failover configuration to all advertised prefixes. The format is: “X:Y”.

See also [bgpd.peer.X.master_communities](#).

4.1.5.25 bgpd.peer.X.slave_localpref


Defines the local-preference value for prefixes announced by Bgpd.


See also [bgpd.peer.X.master_localpref](#).

Possible values: 0 – 4294967295

Default value:

4.1.5.26 bgpd.peer.X.slave_our_ip

 Mandatory for IPv4 BGP session.


 Parameter changes cause reset of BGP session. Affects only BGP session of slave node in a failover configuration.


Defines IPv4 address of IRP server end of this iBGP session of slave node in a failover configuration.

See also [bgpd.peer.X.master_our_ip](#).

Possible values: IPv4 address

4.1.5.27 bgpd.peer.X.slave_our_ipv6

 Mandatory for IPv6 BGP session in failover configuration.

 Parameter changes cause reset of BGP session. Affects only BGP session of slave node in a failover configuration.

Defines IPv6 address of IRP server end of this iBGP session of slave node in a failover configuration.

See also [bgpd.peer.X.master_our_ipv6](#).

Possible values: IPv6 address

4.1.5.28 bgpd.peer.X.slave_password


Defines iBGP session's password for slave node in a failover configuration.


See also `bgpd.peer.X.master_password`.

Possible values: up to 80 alphanumeric characters

4.1.5.29 bgpd.peer.X.slave_peer_ip

Optionally distinct router IPv4 address may be specified in the parameter to be used in order to establish the BGP session from slave instance.

 Mandatory for IPv4 BGP session.

 Parameter changes cause reset of BGP session.


Defines iBGP session's router's IPv4 address.


See also `bgpd.peer.X.master_peer_ip`.

Possible values: IPv4 address

4.1.5.30 bgpd.peer.X.slave_peer_ipv6

Optionally distinct router IPv6 address may be specified in the parameter to be used in order to establish the BGP session from slave instance.

 Mandatory for IPv6 BGP session.


 Parameter changes cause reset of BGP session.


Defines iBGP session's router's IPv6 address.

See also `bgpd.peer.X.master_peer_ipv6`.

Possible values: IPv6 address

4.1.5.31 bgpd.peer.X.slave_router_id


 Mandatory for IPv6 BGP session either as standalone slave or when failover is enabled and the value should be different from `bgpd.peer.X.master_router_id`.

 Parameter changes cause reset of BGP session. Affects only BGP session of slave node in a failover configuration.

Defines IRP server's router ID. The BGP router ID is used in the BGP algorithm for determining the best path to a destination where the preference is given to the BGP router with the lowest router ID.

Possible values: 4-byte value in the IPv4 address format. Any valid IPv4 address can be used.

4.1.5.32 bgpd.peer.X.transit.mib

 Parameter changes cause reset of BGP session.

Defines the MIB used by transit improvements monitors. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

- 0 - Generic (BGP4-MIB)

Default value: 0

4.1.5.33 bgpd.peer.X.transit.snmp

i Parameter changes cause reset of BGP session.

Points to the SNMP host (and its parameters) used for transit improvements monitor on this session. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: valid SNMP host identifier

4.1.5.34 bgpd.peer.X.transit.status

i Parameter changes cause reset of BGP session.

Enables or disables transit improvements through this BGP session (router). Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 0 (Disable), 1 (Enable)

Default value: 0

4.1.5.35 bgpd.peer.X.updates.limit.max

Represents a maximum number of prefixes that can be announced simultaneously in one session.

If `bgpd.updates.split` is ON, the number of announced prefixes is twice the number of the improvements. If the BGP neighbor (typically - the edge router) has any hardware/software limitation for the number of routes in active routing table, then Bgpd can be instructed not to announce more than a specified amount of prefixes. Value 0 means no limit for current iBGP session.

Values less than maximum allowed improvements, can cause not all the improved prefixes to be injected into such peer.

Higher values can be incompatible with the router (please consult the router's vendor regarding the maximum amount of entries in routing table as well as the BGP table).

Possible values: 0-100000

Default value: 0

Recommended value: 0

4.1.5.36 bgpd.peer.X.updates.limit.ps

Defines the maximum number of updates per second that will be sent to the current BGP neighbor. Low values will slow down the improvements injection. High values can cause router to drop the improvement without installing it into the routing database.

Possible values: 1-1000000

Default value: 500

Recommended values: 100-1000

4.1.6 BMP monitoring station settings

4.1.6.1 irpbmpd.log

Defines the file-system path to the BMP monitoring station (irpbmpd) log file.

Default value: /var/log/irp/irpbmpd.log

4.1.6.2 irpbmpd.log.level

Defines the logging level for the BMP monitoring station (irpbmpd) service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.6.3 irpbmpd.port

Defines the TCP port where BMP monitoring station (irpbmpd) listens for monitoring routers to establish BMP sessions.

i When BMP information is available for all providers also consider setting BMP as a source of AS_PATH information under `bgpd.as_path`.

Default value: 7854

Recommended value: 1-65535

4.1.6.4 irpbmpd.sources

Defines IP addresses if the valid BMP sources. BMP connections coming in from other IP addresses that are not listed in this parameter will be ignored.

It is recommended to specify only trusted IP addresses.

Possible values: list of valid IPv4/IPv6 addresses

Default value: 0.0.0.0/0 ::/0

Recommended value: Trusted IPv4/IPv6 flow exporter addresses

4.1.7 Collector settings

4.1.7.1 collector.detect.explorer_ips

Instructs the collector to ignore the Explorer-generated traffic, which can initiate false IRP reaction to the network events (Excessive loss, blackout).

i This feature is used only by the SPAN collector, instructing it to ignore the IPv4 traffic when the network address translation is used to masquerade IP addresses.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 0

4.1.7.2 collector.export.interval

Defines the time interval (in seconds) between data exports in Collectors (flow and span). Lower values result in more accurate traffic data, but higher storage usage. High values cause slow statistics accumulation and less accurate calculation of prefix data.

Possible values: 10-3600

Default value: 60

Recommended value: 60-300

4.1.7.3 collector.export.top_volume_ips

Defines the maximum number of top hosts per prefix for which usage values are stored. Collector persists up to this number of IP address, usage tuples together with other prefix statistics being collected. When the value of the parameter is zero no such statistics will be collected.

Possible values: 0-10

Default value: 5

4.1.7.4 collector.export.ttl

Defines the collector-gathered data lifetime (in seconds). Larger values lead to excessive database size. Aggregated data is kept for one year, disregarding this parameter.

Possible values: 1-'unlimited'

Default value: 86400

Recommended value: 1-30days

4.1.7.5 collector.export.volume.high.top_n

Defines the number of top volume prefixes.

Top N prefixes will be marked for priority probing in descending order of volume. Lower values lead to small number of events for priority probing of high volume prefixes. Higher values lead to overflow of probing queue with jobs for unimportant prefixes.

Possible values: 0-'unlimited'

Default value: 50

Recommended value: 10-50

4.1.7.6 collector.export.volume.min

Defines the minimum collected prefix volume (bytes). The prefix will not be exported into the IRP database if its traffic volume is less than the value of the current parameter (`collector.export.volume.min`) and the percentage of the traffic volume less the `collector.export.volume.min_pct` parameter.

Lower values will lead to a higher number of prefixes exported into the IRP database.

Possible values: 1-2000000000

Default value: 100000

Recommended value: 10000-1000000

4.1.7.7 collector.export.volume.min_pct

Defines the minimum prefix volume (%) for a prefix to be exported into the IRP database. The prefix will not be exported into the IRP database if its percentage of the traffic volume is less than the value of the current parameter (`collector.export.volume.min_pct`) and the traffic volume less than `collector.export.volume.min`. The percentage of the traffic volume is calculated according to the export interval (`collector.export.interval`). Lower values will lead to a higher number of prefixes exported into the IRP database.

Possible values: 0.0001-100

Default value: 0.01

Recommended value: 0.01-0.1

4.1.7.8 collector.flow.all_outbound

The parameter allows Irpflowd to process all outbound traffic even not listed in `collector.ournets`. Enabling the feature requires `peer.X.flow_agents` to be properly configured for all providers to distinguish outbound direction.

Possible values: 0 (Ournets only), 1 (All traffic)

Default value: 0

4.1.7.9 collector.flow.buffer.size

Defines the buffer size (in packets) for the Flow collector (irpflowd). Higher values lead to extra memory used by the collector. Slightly increase this value if the network has traffic spikes that cause buffer overflows and packet drop in the collector.

Possible values: 10000-1000000

Default value: 50000

Recommended value: 50000-500000

4.1.7.10 collector.flow.enabled

Enables the Flow collector. If the parameter is enabled, Flow Collector will be used to gather network prefix data, but it is still possible to use SPAN Collector to acquire network events.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 0 - for SPAN collector (see `collector.span.enabled`), 1 - for Flow collector.

4.1.7.11 collector.flow.export.inbound_transit.topn

Specifies the number of largest volume transit prefixes that are exported each collector cycle for inbound optimization of transiting traffic. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 1-10000

Default value: 100

4.1.7.12 collector.flow.listen.nf

Defines the UDP port the Flow collector will listen to, for NetFlow traffic.

Possible values: 1-65535

Default value: 2055

Recommended value: 2055

4.1.7.13 collector.flow.listen.sf

Defines the UDP port the Flow collector will listen to, for sFlow traffic.

Possible values: 1-65535

Default value: 6343

Recommended value: 6343

4.1.7.14 collector.flow.log

Defines the file-system path to the irpflowd log file.

Default value: /var/log/irp/irpflowd.log

4.1.7.15 collector.flow.log.level

Defines the logging level for the irpflowd service.


Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.7.16 collector.flow.process__transit__in__outbound

Enables or disables matching of prefix traffic at network egress for inbound optimization of transiting traffic. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

 This should be enabled only in complex network topologies where ingress prefix statistics can be incomplete.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.7.17 collector.flow.sources

Defines the valid NetFlow/sFlow/jFlow exporters IP addresses. Flow data coming in from other IP addresses that are not listed in this parameter will be ignored.

It is recommended to specify only trusted Flow exporters IP addresses.

Possible values: list of valid IPv4/IPv6 addresses

Default value: 0.0.0.0/0 ::/0

Recommended value: Trusted IPv4/IPv6 flow exporter addresses

4.1.7.18 collector.ournets


 Mandatory if `ifcollector.span.enabled` is enabled

Defines the list of networks to be analyzed. Typically this is the list of prefixes advertised by your AS.

Format:

1. Space-separated list of local IPv4/IPv6 prefixes that should be analyzed and optimized by the IRP.
2. Absolute path to a newline-separated file containing a list of local IPv4/IPv6 prefixes that should be analyzed and optimized by the IRP.

IPv4 prefixes are treated as /24 subnets and IPv6 prefixes as /48 subnets, if netmask is not clearly specified.

 The downstream clients' networks can be indicated as well.

Possible values: See above.

4.1.7.19 collector.sessions.max

Defines the maximum number of TCP sessions inside the collector process. It can be used to minimize memory usage.

 Both Flow and Span collectors use this parameter.

Estimated memory usage can be calculated as follows: $\text{usage} = 80\text{MB} + (\text{collector.sessions.max} * 1464)$

Possible values: 10000–10000000

Default value: 2000000

Recommended value: Depends on available server memory and the maximum number of simultaneous sessions during peak hours.

4.1.7.20 collector.span.buffer.size

Defines the buffer size (in packets) for the Span collector (irpspand). Higher values lead to extra memory used by the collector. Slightly increase this value if the network has traffic spikes that cause buffer overflows and packet drop in the collector.

Possible values: 10000–5000000

Default value: 100000

Recommended value: 100000–5000000

4.1.7.21 collector.span.enabled

Enables the Span collector. Span collector will be used to gather network prefix data, if the parameter is enabled.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 0 - if no traffic analysis should be performed by the Span collector, 1 - if the Span collector should analyze mirrored traffic for network events and/or prefix statistics.

See also: [collector.flow.enabled](#)

4.1.7.22 collector.span.interfaces

 Mandatory if `collector.span.enabled` is enabled

Defines a space-separated network interfaces list for passive packet analysis by the Span collector.

Example:

```
collector.span.interfaces = eth0 eth1 eth2
```

4.1.7.23 collector.span.log

Defines the file-system path to the irpspand log file.

Default value: /var/log/irp/irpspand.log

4.1.7.24 collector.span.log.level

Defines the logging level for the irpspand service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.7.25 collector.span.min_delay

Enables fast probing tasks queuing for prefixes with one of the following issues:

- Blackouts
- Congestion
- Excessive packet delays.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 0

See also: [collector.span.min_delay.probing_queue_size](#)

4.1.7.26 collector.span.min_delay.probing_queue_size

Defines the number of slots in the probing queue that can be used by the min_delay algorithm (see: [collector.span.min_delay](#))

Possible values: 1-200


Default value: 50

Recommended value: 30-200

See also: [collector.span.min_delay](#)

4.1.7.27 collector.span.size_from_ip_header

When parameter is enabled packet size is determined from header of the IPv4/IPv6 packet. Otherwise, packet size is determined from link layer information (original packet size from PCAP/SNF).

 Enable this when source packets are stripped before entering Noction IRP's appliance SPAN interface.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.7.28 collector.span.threshold.blackout

Defines the percentage of retransmitted packets in regards to the total packets number. If retransmit is higher than this value, this prefix is considered to have a blackout.

Possible values: 1-100

Default value: 70

Recommended value: 70-90

4.1.7.29 collector.span.threshold.congestion

Defines the percentage of retransmitted packets in regards to the total packets number. If retransmit is higher than this value, this prefix is considered to have a congestion.

Possible values: 1-100

Default value: 50

Recommended value: 30-70

4.1.7.30 collector.span.threshold.delay

Percentage of excessive delayed packets by the total packets number (see [collector.span.threshold.excessive](#)). If this percentage is higher than this value, we consider this prefix to have a delay.

Possible values: 1-100

Default value: 20

Recommended value: 10-30

4.1.7.31 collector.span.threshold.excessive

Defines the excessive RTT (%). The value is compared to the average round trip time. If RTT is higher by [collector.span.threshold.excessive](#) in % than the average RTT, then the counter of excessive delay packets is incremented.

Possible values: > 0

Default value: 200

Recommended value: 100-500

4.1.7.32 collector.speaking_ips

Defines the number of speaking IPs to be stored in the database.

Possible values: 0-1000


Default value: 100

Recommended value: 100

4.1.8 Core settings

4.1.8.1 core.circuit.high_loss_diff

Defines the upper threshold of consistent packet loss over a provider when compared to other providers on the network. A provider exceeding this threshold is marked for shutdown. An event of this type will be raised and available to subscribers to act upon. Threshold loss difference is determined over a configured past time horizon and compared with all other providers on the network over the same interval.


 While the provider is marked for shutdown IRP cannot do this itself. Instead network engineers and/or other network capabilities are expected to be triggered in order to divert as much traffic as possible away from provider with circuit issues.

Possible values: 2-50

Default value: 15

4.1.8.2 core.circuit.hist_interval

Defines time interval in minutes used to determine average packet loss over a provider when compared to other providers on the network for circuit issues detection algorithm.

 Keep in mind that shorter time intervals might cause false positives where the averages are high simply because a few large and random packet loss probes are able to push the numbers above thresholds. At the same time longer time intervals will take longer to spot issues and thus will extend the time period where a circuit with issues is being used while alternatives were available.

Possible values: 1-240

Default value: 5

4.1.8.3 core.circuit.inbound

Defines if and when IRP announces Max prepends for inbound prefixes through provider with circuit issues. The options are as follows:

- No changes - any prepends through provider are not changed
- Prepend on warn - IRP announces Max Prepends once Warning level for circuit issues is reached
- Prepend on shutdown - IRP announces Max Prepends only when shutdown level for circuit issues is exceeded.

Refer also [core.circuit.transit](#).

Possible values: 0 (No changes), 1 (Prepend on warn), 2 (Prepend on shutdown)

Default value: 0

4.1.8.4 core.circuit.recover_hold_time

Defines the time interval in seconds before IRP attempts to restore a circuit with issues.

Possible values: 60-3600

Default value: 600

4.1.8.5 core.circuit.recover_loss_diff

Defines the normal threshold of packet loss when a provider with circuit issues can be considered to be ok. At this stage IRP will restore the provider to its full capacity status and will retract all other measures taken in the past in order for the network traffic to avoid the circuit with issues.

➖ This parameter is only used if the ways to react to a circuit issue include restoring it and this only can happen within the given time interval. Otherwise the circuit should be restored by manual intervention of network engineers after they have verified the issue is no longer a problem.

📌 Note that this parameter should be both smaller than `core.circuit.high_loss_diff` and `core.circuit.warn_loss_diff`

Possible values: 0-48

Default value: 5

4.1.8.6 core.circuit.recover_monitored_intervals

Defines the time interval in minutes during which IRP will continuously evaluate average loss for provider(s) with circuit issues in order to determine if it is back to normal.

➖ This parameter is only used if the ways to react to a circuit issue include restoring it and this only can happen within the given time interval. Otherwise the circuit should be restored by manual intervention of network engineers after the circuit issue is no longer a problem.

Possible values: 1-30

Default value: 5

4.1.8.7 core.circuit.transit

Defines if and when IRP announces Max preponds for transit prefixes through provider with circuit issues. The options are as follows:

- No changes - any preponds through provider are not changed
- Prepend on warn - IRP announces Max Preponds once Warning level for circuit issues is reached
- Prepend on shutdown - IRP announces Max Preponds only when shutdown level for circuit issues is exceeded.

Refer also `core.circuit.inbound`.

Possible values: 0 (No changes), 1 (Prepend on warn), 2 (Prepend on shutdown)

Default value: 0

4.1.8.8 core.circuit.warn_loss_diff

Defines the lower threshold of packet loss when a provider seems to be having circuit issues. At this stage IRP will start raising alerts that network engineers and/or network management systems can subscribe in order to act on them. At this level IRP starts taking other preventive measures such as re-probing outbound improvements made to provider with circuit issues.

i Note that this parameter should be both smaller than `core.circuit.high_loss_diff` and larger than `core.circuit.recover_loss_diff`.

Possible values: 1–49

Default value: 10

4.1.8.9 core.circuit.withdraw_on_warn

Enables or disables withdrawing outbound improvements to a provider with circuit issues at or above warning level.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.8.10 core.commit_control

Enables or disables Commit Control (see the [Commit Control \(Not supported in IRP Lite\)](#) section)

w The following parameters must be set in the configuration file to ensure proper functionality of the Commit Control feature:

1. SNMP parameters must be set for each provider:
 - (a) `SNMP Hosts settings`
 - (b) `global.ifstats`
 - (c) `peer.X.snmp.interfaces`
 - (d) `irpstatd` needs to be started (see section 2.8)
2. Each provider needs to have `peer.X.95th` set to the desired Commit level
3. `peer.X.precedence` must be set for each provider

When Commit Control is disabled, all existing Commit Control improvements are removed from current improvements. The improvements are preserved temporarily until `core.commit_control.probe_ttl` after Core service restart. If Commit Control is re-enabled before expiration these improvements will be restored into current improvements.

See also:

[Commit Control \(Not supported in IRP Lite\)](#)

`core.commit_control.rate.group`

`core.commit_control.rate.high`

`core.commit_control.rate.low`

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.8.11 core.commit_control.agg_bw_min

Defines the minimum bandwidth for a single prefix. Prefixes, whose bandwidth is less than the specified value, are ignored by Commit Control and are not being used in the Commit Control algorithm.

⚠ It is not recommended to set high values for this parameter since it limits the number of prefixes that can be rerouted by the Commit Control mechanism.

i This parameter is considered only during initial improvement. During retry probing of existing Commit Control improvements IRP might detect that the current bandwidth usage for some prefixes is below this limit. IRP will preserve these improvements as relevant if there are no other reasons to withdraw them.

Its recommended to decrease value if summary throughput is lower than 200-500 Mbps.

Possible values: 0.1-5000

Default value: 1

Recommended value: 1

4.1.8.12 core.commit_control.del_irrelevant

If enabled, Commit Control algorithm deletes improvements that have traffic volume less than value in the [4.1.8.11](#) parameter.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.8.13 core.commit_control.inbound.enabled

Enables or disables Inbound bandwidth control.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.8.14 core.commit_control.inbound.improvement.delay

The parameter specifies minimal time delay before next Inbound/Transit optimization cycle.

Time delay should be sufficient to cover route propagation time and time required to collect and process changes in bandwidth distribution.

Possible values: 60 - 1800

Default value: 300

4.1.8.15 core.commit_control.inbound.moderated

Enables or disables review and moderate feature of inbound bandwidth control.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.8.16 core.commit_control.inbound.rate.high

Defines provider's high load rate limit (%). Refer [core.commit_control.rate.high](#) for details.

This limit is used when inbound and outbound commit control operate independently. Otherwise [core.commit_control.rate.high](#) value overrides this. Refer also [peer.X.95th.mode](#).

Possible values: 50 - 99

Default value: 90

4.1.8.17 core.commit_control.inbound.rate.low

Defines provider's low load rate limit (%). Refer [core.commit_control.rate.low](#) for details.

This limit is used when inbound and outbound commit control operate independently. Otherwise [core.commit_control.rate.low](#) value overrides this. Refer also [peer.X.95th.mode](#).

Possible values: 30 - 99

Default value: 70

4.1.8.18 core.commit_control.inbound.volume_estimation

Defines method of estimating inbound bandwidth. The available options are:

- 0: Last minute data
- 1: Largest of last minute and current hour average
- 2: Largest of last minute and daily average

Possible values: 0 - 2

Default value: 1

4.1.8.19 core.commit_control.loss_override

Defines when IRP allows Commit Control improvements to adjust a provider's bandwidth considering current and new provider's loss measurements. The available options are to allow Commit Control improvements when there is:

- 0: Better or equal loss
- 1: Better or irrelevant loss difference
- 2: Any loss difference

Refer [core.performance.loss_pct](#) for details.

Possible values: 0 - 2

Default value: 0

4.1.8.20 core.commit_control.probe_ttl

Defines the TTL (time-to-live) for a specific probe. If the probe results are older than the value specified here, the system will disregard it and the Commit Control algorithm will schedule it for expedited re-probing.

Possible values: 600-86400

Default value: 7200

Recommended value: 7200

4.1.8.21 core.commit_control.probing_queue_size

Defines the number of slots in the probing queue that can be used by the Commit Control algorithm. This sets the upper limit on the number of prefixes that can be scheduled for probing by Commit Control. Note that this queue has higher priority than ordinary and retry probing. At the same time Commit Control relies heavily on existing probing results and most of the time this queue will be empty.

Possible values: 1-500

Default value: 100

Recommended value: 10-100

4.1.8.22 core.commit_control.rate.group

If using provider load balancing in group, this parameter defines allowed deviation of a provider's current bandwidth(in %) compared with other providers in the same group.

Example 1. There are three grouped providers with 95th set to 1 Gbps, 2 Gbps and 3 Gbps with a total current bandwidth of 600 Mbps. In this case, load balancing expects that each provider's bandwidth usage will be 100 Mbps, 200 Mbps and 300 Mbps accordingly. These values are proportional to individual provider's 95th settings in the group. Let's assume `core.commit_control.rate.group` is set to 5%. If a provider's bandwidth exceeds by more than the 5% limit the expected value (105 Mbps, 210 Mbps and 315 Mbps accordingly and the total for the group did not change), IRP will start active rerouting of excessive bandwidth from that provider to providers within or outside this group.

 Load balancing in a provider group is performed even when the 95th is not exceeded.

Possible values: 1-30

Default value: 5

Recommended value: 5


See also: [Provider load balancing](#)

4.1.8.23 core.commit_control.rate.high

Defines provider's high load rate limit (%).

IRP will stop Latency/Cost improvement if provider bandwidth is over `core.commit_control.rate.high` % of load (percents of 95th). The improvements will start happening again after provider's bandwidth drops below `core.commit_control.rate.low` % of load.

These parameters are used for passive 95th overload prevention as well as an additional method for Commit Control to be less aggressive.

 Provider's Latency/Cost improvement ability does not change when current bandwidth rate varies between `core.commit_control.rate.low` and `core.commit_control.rate.high`.

Example 2. if provider's current load is 1100 Mbps, 95th is set to 1000 Mbps. Commit Control will try to unload 200Mbps (to reduce current load to 90% of the 95th level) in order to prevent 95th overload as well as allow provider's bandwidth natural growth during peak hours.

See also: [Provider load balancing](#)

Possible values: 50-99, but larger or equal to `core.commit_control.rate.low`

Default value: 90

Recommended value: 90

4.1.8.24 core.commit_control.rate.low

Defines provider's low load rate limit (%).

IRP will start Latency/Cost improvements again after provider's bandwidth drops below `core.commit_control.rate.low` % of load. Latency/Cost improvements will be stopped if provider bandwidth is over `core.commit_control.rate.high` % of load (percent of 95th).

These parameters are used for passive 95th overload prevention as well as an additional method for Commit Control to be less aggressive.

i Provider's Latency/Cost improvement ability does not change when current bandwidth rate varies between `core.commit_control.rate.low` and `core.commit_control.rate.high`.

See also: Provider load balancing

Possible values: 30–99, but lower or equal to `core.commit_control.rate.high`

Default value: 80

Recommended value: 80

4.1.8.25 core.commit_control.react_on_collector

Defines if Commit Control algorithm should react immediately on collector overload values.

w Enabling this feature represents a tradeoff. Take into consideration the benefits of a faster react time vs reliance on older probes and statistics when making Commit Control improvements.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.8.26 core.commit_control.worst_loss

Defines amount of loss worsening allowed when IRP performs Commit Control improvements and `core.commit_control.loss_override` is set to "Any loss difference".

Refer `core.performance.loss_pct` for details.

Possible values: 1 - 99

Default value: 2

4.1.8.27 core.cost.worst_ms

Latency worsening.

Defines the allowed latency worsening for an improved prefix while running in `Cost optimization` mode (see `global.improve_mode`) and the `Commit Control (Not supported in IRP Lite)` feature is enabled. While operating in these modes IRP will consider as alternatives candidates with latency degradation not exceeding this limit. Of course, the candidate with least latency degradation or even with better latency will be chosen as an improvement.

When IRP detects that an existing Commit Control improvement has alternatives with latencies better than specified value it will replace it with a new performance (latency) improvement. Over-usage of the alternative route will be avoided.

i This parameter is applicable for local improvements within a Routing Domain. Global improvements will also take into account `core.global.worst_ms` values.

Possible values: 1-unlimited

Default value: 10

Recommended value: 10

4.1.8.28 core.eventqueuelimit

Defines the exploring events queue size. Lower values may result in IRP missing some important network issues.

Possible values: 1-10000

Default value: 1000

Recommended value: 100-1000

4.1.8.29 core.eventqueuelimit.retry_probe_pct

Defines a percentage of the total exploring queue length (see `core.eventqueuelimit`) to be used for retry probing.

Possible values: 1-100

Default value: 40

Recommended value: 20-60

4.1.8.30 core.flowspec.max

Defines the maximum number of IPv4 Flowspec rules that IRP is allowed to advertise.

Possible values: 1-10000

Default value: 100

4.1.8.31 core.flowspec.max_ipv6

Defines the maximum number of IPv6 Flowspec rules that IRP is allowed to advertise.

Possible values: 1-10000

Default value: 100

4.1.8.32 core.global.allow_commit

Enables or disables global commit control Improvements in a multiple routing domain configuration. By default these Improvements are enabled.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.8.33 core.global.allow_latency_cost

Enables or disables latency and cost global Improvements in a multiple routing domain configuration. By default these Improvements are enabled.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.8.34 core.global.worst_ms

Defines acceptable latency worsening for cost and commit for global improvements. Refer [core.cost.worst_ms](#)

Possible values: 1-unlimited

Default value: 30

Recommended value: 10

4.1.8.35 core.improvements.clearslots

Specifies the number of outdated improvements that will be discarded when slots are needed for new improvements. Outdated are improvements as defined by [core.improvements.clearslots.days_max](#). In case there are no outdated improvements usual improvement replacement mechanisms are employed relying on [Improvements weight](#).

Possible values: 0 - 100

Default value: 10

Recommended value: 10

4.1.8.36 core.improvements.clearslots.days_max

Sets the number of days after which an improvement is considered outdated. Outdated improvements have not been re-confirmed for a long time. The oldest of them will be discarded by the slot clearing process for outdated improvements when new slots are needed for new improvements. Refer [core.improvements.clearslots](#).


Possible values: 1 - 60

Default value: 7

Recommended value: 7

4.1.8.37 core.improvements.inbound_transit_max

Specifies the maximum number of transit improvements. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

 Increasing this limit should be done with care. Each transit improvement is continuously monitored if alternative routes from providers are still present on the router(s) and this consumes router CPU resources.


Possible values: 1-1000

Default value: 100

4.1.8.38 core.improvements.inbound_transit.ttl.clean

Specifies how IRP should treat old transit improvements. Old transit improvements are those that exceed `core.improvements.inbound_transit.ttl.max`. The options are to either

- gradually decrease the number of preponds and withdraw the transit improvement when no preponds are left or
- withdraw the inbound transit improvement immediately when it exceeds `core.improvements.inbound_transit.ttl.max`

 Transit improvements cleanup performed once a day at configured off-peak hour (`global.offpeak_hour`).

Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 0 (Decrease preponds), 1 (Remove immediately)

Default value: 0

4.1.8.39 core.improvements.inbound_transit.ttl.max

Specifies the maximum period of time to preserve a transit improvement. If IRP does not have any other reasons to adjust a transit improvement after this limitation is exceeded its preponds will be decreased or the improvement will be withdrawn. Refer [core.improvements.inbound_transit.ttl.clean](#), [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 901 - 345600

Default value: 86400

4.1.8.40 core.improvements.inbound_transit.ttl.min


Specifies the minimal period of time to preserve a transit improvement. In order to give sufficient time to the entire Internet to adjust to a transit improvement and to avoid route instability IRP blocks making changes to a transit improvement for this duration of time. IRP can still offer route changes to the same transit prefixes routed through other providers. Refer [Optimization of transiting traffic \(Not supported in IRP Lite\)](#), [Optimization of transiting traffic](#).

Possible values: 600 - 86400

Default value: 14400

4.1.8.41 core.improvements.max

Defines the maximum number of active IPv4 improvements.

 The number of announced prefixes can be twice this value if `bgpd.updates.split` is enabled.

Possible values: 1-100000

Default value: 10000

Recommended value: 10000

4.1.8.42 core.improvements.max_ipv6

Defines the maximum number of active IPv6 improvements.

Possible values: 1-100000

Default value: 2000

Recommended value: 2000

4.1.8.43 core.improvements.retry_probe.volume_top_n

Traffic volume top-N prefixes

The improvements are checked for relevance when they are improved the first time, with the relevancy flag set accordingly. All prefixes (starting from current month's first day) are sorted by traffic volume. The top-N prefixes from this list are treated as "relevant" prefixes.

In case an empty slot is required for a new improvement, the old irrelevant prefix(es) will be deleted first to free-up a slot.

Volume-relevant prefixes will be queued for retry probing before non-relevant prefixes.

Possible values: 1-20000

Default value: 5000

Recommended value: chosen for each network individually

4.1.8.44 core.improvements.safe_removal

Specifies when IRP removes/withdraws outbound improvements. Outbound improvements become irrelevant if old provider performance metrics are no longer worse than the improved provider metrics. Still, original routes for some improvements change and providers that will service those prefixes once IRP improvements are withdrawn might have worse performance metrics. Safe improvement removal accounts for this possibility and removes improvements only when ALL providers have performance metrics that are not worse than the (fresh) improvement provider metrics.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.8.45 core.improvements.ttl.retry_probe

Defines the retry probing interval (in seconds). Once an improvement is older than this value, it will be sent for re-probing.

Possible values: 600-86400

Default value: 14400

Recommended value: 7200-14400

See also: [core.improvements.retry_probe.volume_top_n](#)

4.1.8.46 core.log

Defines the file-system path to the core log file.

Default value: /var/log/irp/core.log

4.1.8.47 core.log.level

Defines the logging level for the core service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.8.48 core.outage_detection

Enables the Outage Detection algorithm.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 1

4.1.8.49 core.outage_detection.limit_pct

Defines the problem prefixes threshold (in percent) over which the system confirms an outage (if `core.outage_detection` is enabled).

See also: [Outage detection](#)

Possible values: 1-100

Default value: 60

Recommended value: 60

4.1.8.50 core.overusage.check_interval

Defines the interval to check overusage for Flowspec policies prefixes in order to apply automatic throttling Flowspec policies.

Possible values: 60-600

Default value: 60

Recommended value: 60

4.1.8.51 core.overusage.hold_timer

Defines the retention time for throttling rules after prefix average bandwidth returns to normal for automatic throttling Flowspec policies.

Possible values: 60-600

Default value: 60

Recommended value: 300

4.1.8.52 core.overusage.out.average.period

Defines the duration of the time interval in hours used to determine average prefix usage for automatic throttling Flowspec policies.

Possible values: 1-24

Default value: 1

Recommended value: 1

4.1.8.53 core.overusage.out.average.relevant_min

Defines minimum prefix bandwidth average (in Mbps) that is relevant for automatic throttling Flowspec policies.

Possible values: 1-100000

Default value: 50

Recommended value: 50

4.1.8.54 core.overusage.out.threshold.throttle

Defines the multiplier of prefix average bandwidth applied on outbound traffic for automatic throttling Flowspec policies.

Possible values: 2-10000

Default value: 2

Recommended value: 2

4.1.8.55 core.overusage.out.threshold.trigger

Defines the multiplier of prefix average bandwidth that sets the overusage threshold of a prefix. An automatic throttling Flowspec policy is created when current prefix bandwidth exceeds the average multiplied by this multiplier.

Possible values: 2-10000

Default value: 10

Recommended value: 10

4.1.8.56 core.performance.loss_pct

Defines the relevant packet loss difference (in percent) after which a loss improvement is made.

For current route loss > 50%: Do not perform rerouting, if packet loss difference between routes is less than 15%.

For current route loss <= 50%: Do not perform rerouting, if packet loss difference between routes is less than `core.performance.loss_pct`.


Possible values: 1-15

Default value: 3

Recommended value: 3-5

4.1.8.57 core.performance.rtt.diff_ms

Defines the relevant RTT difference (ms) after which a latency improvement is made. If the RTT difference between the current route and another provider is less than `core.performance.rtt.diff_ms`, then the system ignores this prefix as an improvement candidate.

 `core.performance.rtt.diff_ms` and `core.performance.rtt.diff_pct` are grouped together at decision making, using a logical AND condition.

Possible values: 1-'unlimited'

Default value: 10

Recommended value: 2-20

4.1.8.58 `core.performance.rtt.diff_pct`

Defines the relevant RTT difference (in percent) after which a latency improvement is made. If the RTT difference between the current route and another provider is less than `core.performance.rtt.diff_pct`, then the system ignores this prefix as an improvement candidate.

 `core.performance.rtt.diff_ms` and `core.performance.rtt.diff_pct` are grouped together at decision making, using a logical AND condition.


Possible values: 1-100


Default value: 10

Recommended value: 5-20

4.1.8.59 `core.performance.rtt.ix_diff_ms`

Defines the relevant RTT difference (ms) to make latency improvements across Internet Exchanges and transit providers. A latency improvement from an IX to a transit provider is allowed only when the latency is improved by at least this threshold. Inversely, a latency improvement from a transit provider to an IX is allowed even if latency degradation is less than this threshold.

 A default value of zero for this parameter disables this feature. A value smaller than or equal to `core.performance.rtt.diff_ms` isn't allowed. When the feature is disabled then the `core.performance.rtt.diff_ms` and `core.performance.rtt.diff_pct` parameters are considered to determine relevancy of latency improvements.

 `core.performance.rtt.ix_diff_ms` and `core.performance.rtt.ix_diff_pct` are grouped together at decision making, using a logical AND condition.


Possible values: 1-1000000

Default value: 0

Recommended value: 20-50

4.1.8.60 `core.performance.rtt.ix_diff_pct`

Defines the relevant RTT difference (in percent) to make latency improvements across Internet Exchanges and transit providers. Refer `core.performance.rtt.ix_diff_ms` for details.

 `core.performance.rtt.ix_diff_ms` and `core.performance.rtt.ix_diff_pct` are grouped together at decision making, using a logical AND condition.

Possible values: 1-100

Default value: 10

Recommended value: 5-20

4.1.8.61 core.probes.ttl.failed

Defines the failed probe lifetime (in seconds).

Regular and Retry probing will not be performed until the age of the failed probe is less than `core.probes.ttl.failed`

Possible values: 120–14400

Default value: 300

Recommended value: 300

4.1.8.62 core.probes.ttl.max

Defines the probe lifetime (in seconds).

Probed prefixes data is kept in the IRP database for the specified amount of time. Automatic cleanup of outdated data is performed periodically by Dbcron.

See also: [Administrative Components](#)

High values will lead to database size growth.

Possible values: 86400–604800

Default value: 86400

Recommended value: 86400

4.1.8.63 core.probes.ttl.min

Defines the relevant probe lifetime (in seconds)

Ordinary probing will not be performed for a specific prefix if its probe age is less than `core.probes.ttl.min`.

Possible values: 300–43200

Default value: 7200

Recommended value: 1800–14400

4.1.8.64 core.problem.outage_timeout

Outage detection timeout (in seconds).

The IRP will disregard a possible outage, if it was not confirmed during last `core.problem.outage_timeout`.

Possible values: 0–3600

Default value: 600

Recommended value: 600

See also: [core.outage_detection](#)

4.1.8.65 core.problem.rtt.diff_pct

Defines the RTT difference (in percent) between the current route and the new route, for Outage detection algorithm. IRP will choose a new route for problematic prefixes only if the RTT difference (in percent) is greater than this parameter.

Possible values: 1–100

Default value: 50

Recommended value: 30–60

4.1.8.66 core.vip.interval.probe

Defines the VIP prefixes probing interval, in seconds. All networks and ASNs specified in `/etc/noction/policies.co` and will be queued for priority probing every `core.vip.interval.probe` seconds.

i For probing intervals lower than 5 minutes (300 seconds) IRP uses fast probing algorithms avoiding for example long lasting tracing.

See also: [VIP Improvements](#)

Possible values: 60–14400

Default value: 3600

4.1.9 Explorer settings

4.1.9.1 explorer.aipi

If enabled, AIPi (Adaptive Inter-Packet Interval) algorithm is executed using inter-packet intervals configured for each provider.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: 0

See also: [peer.X.diag_hop.interval_min](#), [peer.X.diag_hop.interval_max](#)

4.1.9.2 explorer.algorithms

Enables or disables the use of new scanning, probing and tracing Explorer algorithms.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.9.3 explorer.high_vol_precedence

High volume tasks have priority over the tasks with variable cardinality.

The parameter establishes the balance between the tasks allocated for processing high volume prefixes (high priority for the Customer network) and the network events tasks (such as blackout, congestion, etc).

Possible values: 10–90

Default value: 50

See also: [explorer.maxthreads](#), [collector.export.volume.high.top_n](#)

4.1.9.4 explorer.infra_ips

Defines the IPv4/IPv6 addressees, used in internal infrastructure to determine the current route for a specific prefix. If netmask is not specified, then /32 is assumed for IPv4 address and /128 for IPv6 address (host addresses).

Format:

1. Space-separated list of infrastructure IPv4/IPv6 addresses.
2. Absolute path to a newline-separated file containing a list of local IPv4/IPv6 prefixes that should be analyzed and optimized by the IRP.

Possible values: See above.

Recommended value: all IPv4/IPv6 addresses used in the internal infrastructure.

4.1.9.5 explorer.interval.infra

Defines the time interval (in milliseconds) between the packets sent to infrastructure hops (4.1.9.4). By decreasing this value the Explorer performance enhances.

Possible values: 1-200

Default value: 5

Recommended value: 5

4.1.9.6 explorer.interval.other

Defines the time interval (in milliseconds) between the packets sent to destination hops (4.1.9.4).

Possible values: 1-1000

Default value: 200

Recommended value: 200

4.1.9.7 explorer.interval.other.trace

Defines the time interval (in milliseconds) between traceroute packets sent to non-infrastructure hops.

Possible values: 1-1000

Default value: 20

Recommended value: 20

4.1.9.8 explorer.ipv4_test

Defines public IPv4 address that will be used for ensuring that PBR is operational.

Possible values: IPv4 address

Default value: 8.8.8.8

Recommended value: 8.8.8.8

4.1.9.9 explorer.ipv6_test

Defines public IPv6 address that will be used to verify that PBR is operational.

Possible values: IPv6 address

Default value: 2620:0:ccc::2

Recommended value: 2620:0:ccc::2

4.1.9.10 explorer.log

Defines the file-system path to the explorer log file.

Default value: /var/log/irp/explorer.log

4.1.9.11 explorer.log.level

Defines the logging level for the explorer service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.9.12 explorer.max_collector_ips

Process max collected IPs. Maximum number of IPs to probe for a specific prefix.


Possible values: 1-255

Default value: 10

Recommended value: 10

4.1.9.13 explorer.maxthreads

Defines the number of simultaneously processed exploring tasks.

 If this value is excessively large Explorer tasks can take very long to finish. Hardware/router diagnostic packet rate limitations kick in causing all threads to wait for exploring tasks to finish.

Possible values: 10-2000

Default value: 200

Recommended value: 200-300

4.1.9.14 explorer.probe.algorithm

Defines the probing algorithm(s) used by explorer, in the preferred order.

The following algorithms can be used:

- UDP - udp requests (destination port: 33434)
- ICMP - regular ICMP ping sweep
- TCP_SYN - tcp syn scan (destination port: 33434)

Possible values: ICMP UDP TCP_SYN

Default value: ICMP UDP TCP_SYN

Recommended value: ICMP UDP TCP_SYN

4.1.9.15 explorer.probe.indirect__priority

Defines the execution priority between direct and indirect probing algorithms. Parameter value of 1 means that indirect probing will be executed prior to direct probing algorithm. Parameter value of 2 disables indirect probing.

 This parameter conflicts with `explorer.trace.all` configuration option.

Possible values: 0 (Direct then Indirect), 1 (Indirect then Direct), 2 (Direct only)

Default value: 0

Recommended value: 0

4.1.9.16 explorer.probing.sendpkts.adaptive__max

Defines the maximum number of ping packets for adaptive prefix probing.

Possible values: positive Integer value, higher or equal to `explorer.probing.sendpkts.min`

Default value: 100

Recommended value: 100

4.1.9.17 explorer.probing.sendpkts.min

Defines the default ping packets count for each probe. If any loss is detected, additional packets (up to `explorer.probing.sendpkts.adaptive__max`) are sent, for a more accurate packet loss detection.

Possible values: positive Integer value

Default value: 5

Recommended value: 5

4.1.9.18 explorer.probing.simultaneous

Defines the number of scanned IP addresses.

Possible values: 1-100

Default value: 50

Recommended value: 50

4.1.9.19 explorer.scanning.confirm__ips

Defines the number of scanned speaking IP addresses for a provider with loss.

Possible values: 1-10

Default value: 5

Recommended value: 5

4.1.9.20 explorer.scanning.replypkts.min

Defines the number of response packets from a scanned speaking IP address to qualify as a candidate.

Possible values: 1-10

Default value: 5

Recommended value: 5

4.1.9.21 explorer.scanning.rtt.dispersion_ms

Defines RTT maximum dispersion in milliseconds to qualify a speaking IP as candidate.

Possible values: 1-1000

Default value: 50

Recommended value: 50

4.1.9.22 explorer.scanning.sendpkts.factor

Defines the number of attempts to send packets to all selected speaking IPs.

Possible values: 1-10

Default value: 5

Recommended value: 5

4.1.9.23 explorer.timeout

Defines the probes ICMP timeout (in ms)

Default value: 2000

Recommended value: 2000

4.1.9.24 explorer.timeout.infra

Defines the waiting time (in milliseconds) for infrastructure hops' (4.1.9.4) responses expected during trace execution.

Possible values: 50-20000

Default value: 10000

Recommended value: 10000

4.1.9.25 explorer.trace.algorithms

Defines the traceroute algorithm(s) to be used, arranged by priority. See [explorer.probe.algorithm](#) for algorithms description.

Possible values: UDP ICMP TCP_SYN

Default value: UDP ICMP

Recommended value: UDP ICMP

4.1.9.26 explorer.trace.all

Defines tracing behaviour. If traces are forced, each probed prefix unconditionally runs traces through all configured providers. Regular tracing is needed for Outage Detection and for AS Path reconstruction. Traces can be disabled altogether by setting `explorer.trace.all` to 2.

⊖ Third algorithm should be enabled in `bgpd.as_path` when traces are fully disabled.

⚠ Forcing ALL traces (`explorer.trace.all = 1`) can significantly slow down probing for networks. Review probing times before and after changing this parameter and if probing times become unacceptable revert to previous setting.

ℹ When all traces are disabled (`explorer.trace.all = 2`) IRP is missing essential trace data and is not able to take action on some types of problems. This effectively cuts off those features.

Possible values: 0 (Regular), 1 (Force all), 2 (Disable all)

Default value: 0

Recommended value: 0

4.1.9.27 explorer.traceroute.retrypkts

Defines the number of additional traceroute packets to be sent to the intermediate hop in case it has not replied and might be an infrastructure boundary hop.

Possible values: 0-50

Default value: 10

Recommended value: 10

4.1.9.28 explorer.traceroute.sendpkts

Defines the number of traceroute packets per hop to be sent for each probe.

Possible values: 3-5

Default value: 3

Recommended value: 3

4.1.9.29 explorer.traceroute.simultaneous

Defines the number of simultaneously probed hops during trace execution initiated from the hop defined by (4.1.9.32).

Possible values: 1-30

Default value: 5

Recommended value: 5

4.1.9.30 explorer.traceroute.simultaneous.infra

Defines the number of simultaneously probed infrastructure hops during trace execution.

Possible values: 1-30

Default value: 3

Recommended value: 3

4.1.9.31 explorer.traceroute.ttl.max

Defines the maximum traceroute TTL. Essentially this defines the last hop at which explorer stops the trace.

Possible values: 1-255

Default value: 30

Recommended value: 30

4.1.9.32 explorer.traceroute.ttl.min

Defines the minimum traceroute TTL. Basically this defines the first hop after which explorer analyzes the trace.

Possible values: 1-255

Default value: 2

Recommended value: 2-5

4.1.10 Notification and events settings

4.1.10.1 pushd.email.from

Defines the From email address used for sending email notifications.

Possible values: valid email address

Default value: root@localhost

4.1.10.2 pushd.email.host

Defines the IPv4, IPv6 address or email server host name used to send emails.

Possible values: Hostname or IPv4/IPv6 address

Default value: 127.0.0.1

4.1.10.3 pushd.email.port

Defines the TCP port used for sending emails.

Possible values: 1 - 65535

Default value: 25

Recommended value: 25

4.1.10.4 pushd.listen.port

Defines the TCP listen port of irppushd service.

Possible values: 1-65535

Default value: 10499

4.1.10.5 pushd.log

Defines the complete path to the irppushd log file

Possible values: full path to log file

Default value: /var/log/irp/irppushd.log

4.1.10.6 pushd.log.level

Defines the logging level for the irppushd service.

Possible values: emerg, fatal, alert, crit, error, warn, notice, info, debug

Default value: info

Recommended value: info

4.1.10.7 pushd.sms.account_sid

Defines the public account identifier issued to user by the SMS gateway. Usually this is a random string.

Possible values: random string

4.1.10.8 pushd.sms.auth_token

Defines the secret authentication token issued by the SMS gateway to the account holder. Usually this is a longer random string.

Possible values: random string

4.1.10.9 pushd.sms.gateway


Defines the SMS gateway preferred by the user. A valid account with this SMS gateway is required to send SMS.

Possible values: twilio, plivo

Default value: none

4.1.10.10 pushd.sms.message_size

Defines the maximum size of SMS texts. Texts that are longer than this limit will be trimmed and a ... will be added. The SMS gateway will split the SMS text into multiple messages if the request includes a longer than supported SMS message.

 The SMS gateway enforces its own text size limits. In case the notification exceeds SMS gateway limits the SMS message can be either trimmed or rejected.

Possible values: 150-6000

Default value: 150

4.1.10.11 pushd.sms.phone_number

Defines the From phone number used for sending SMS notifications.

⊖ SMS gateways might have policies regarding the valid phone numbers that they will accept. Check with your SMS gateways if there are any restrictions and if yes what phone numbers can be provided in this configuration parameter.

Possible values: valid phone number

4.1.10.12 pushd.sms.uri.plivo

Defines the URL of the Plivo SMS API.

⊖ Do not change this parameter unless Plivo SMS API URL has changed.

Possible values: valid URL

Default value: `https://api.plivo.com/v1/Account/%1%/Message/`

Recommended value: `https://api.plivo.com/v1/Account/%1%/Message/`

4.1.10.13 pushd.sms.uri.twilio

Defines the URL of the Twilio SMS API.

⊖ Do not change this parameter unless Twilio SMS API URL has changed.

Possible values: valid URL

Default value: `https://api.twilio.com/2010-04-01/Accounts/%1%/Messages.json`

Recommended value: `https://api.twilio.com/2010-04-01/Accounts/%1%/Messages.json`

4.1.10.14 pushd.templates.datadir

Defines the directory for notification templates used by irppushd service to format email and sms messages.

Possible values: full path to notification templates

Default value: `/etc/noction/templates`

4.1.10.15 pushd.webhook.avatar_emoji

Defines an emoji to be used as the IRP bot's avatar image. The emoji is expected to be in the “:robot-face:” notation.


ℹ The avatar emoji is used if an avatar icon in `pushd.webhook.avatar_urlis` not specified.

Possible values: valid emoji in `:colon:` notation

Default value: `:robot-face:`

4.1.10.16 pushd.webhook.avatar_url

Defines the URL of the IRP bot's avatar image. URL should be a publicly accessible PNG or JPEG image that the webhook provider is able to retrieve and use.

 This is the default avatar image. The avatar icon will be used instead of an emoji if both are specified. Refer to `pushd.webhook.avatar_emoji`.

Possible values: valid URL

Default value: `http://www.noction.com/round-logo.png`

Recommended value: `http://www.noction.com/round-logo.png`

4.1.10.17 pushd.webhook.botname

Defines the name assigned to IRP bot.

Possible values: text


Default value: IRP

Recommended value: IRP

4.1.10.18 pushd.webhook.url

Defines the URL assigned to Web Hooks user/team. Web Hooks work by POST-ing JSON content to this designated URL for example

POST `https://hooks.slack.com/services/T00000000/B00000000/XXXXXXXXXXXXXXXXXXXXXXXXXXXX`

 This parameter is required in order for any subscription to web hooks channels to work. Currently only the Slack.com web hooks API has been tested and confirmed to work.

Possible values: valid URL

4.1.10.19 trap.bgpd__announced__rate__low.limit__pct

Defines the announcements rate limit percentage.

Possible values: 1 - 100

Default value: 60

Recommended value: 60

4.1.10.20 trap.core__cc__improvements__spike.diff__pct

Defines the size (in percent) of the spike in the number of improvement compared to preceding period average defined in `trap.core__cc__improvements__spike.period__sec`.

Possible values: 1 - 100

Default value: 50

4.1.10.21 trap.core_cc_improvements_spike.period_sec

Defines the time interval preceeding a spike. The number of improvements is averaged over this time and a spike event is triggered when the subsequent measurement exceeds the size (in percent) defined by `trap.core_cc_improvements_spike.diff_pct`.

Possible values: 30 - 86400

Default value: 300

4.1.10.22 trap.core_cc_overload.inbound_limit_mbps

Defines the overall overload limit in Mbps for the Inbound Commit Control overload by X Mbps event. The event is raised if the overall inbound traffic of the network exceeds the configured value (sum of `peer.X.95th` for all providers) by this limit.

Possible values: 1 - 10000

Default value: 100

Recommended value: 100

4.1.10.23 trap.core_cc_overload.inbound_limit_pct

Defines the overall overload limit in percent for the Inbound Commit Control overload by X% event. The event is raised if the overall inbound traffic of the network exceeds the configured value (sum of `peer.X.95th` for all providers) by this limit.

Possible values: 1 - 1000

Default value: 10

Recommended value: 10

4.1.10.24 trap.core_cc_overload.limit_mbps

Defines the overall overload limit in Mbps for the Outbound Commit Control overload by X Mbps event. The event is raised if the overall outbound traffic of the network exceeds the configured value (sum of `peer.X.95th` for all providers) by this limit.

Possible values: 1 - 10000

Default value: 100

Recommended value: 100

4.1.10.25 trap.core_cc_overload.limit_pct

Defines the overall overload limit in percent for the Outbound Commit Control overload by X% event. The event is raised if the overall outbound traffic of the network exceeds the configured value (sum of `peer.X.95th` for all providers) by this limit.

Possible values: 1 - 1000

Default value: 10

Recommended value: 10

4.1.10.26 trap.core_cc_provider_overload.inbound_limit_mbps

Defines the per provider overload limit in Mbps for the Inbound Commit Control provider X overloaded by Y Mbps event. The event is raised if the inbound traffic for a provider exceeds the configured value (refer [peer.X.95th](#)) by more than this limit.

Possible values: 1 - 10000

Default value: 100

Recommended value: 100

4.1.10.27 trap.core_cc_provider_overload.inbound_limit_pct

Defines the per provider overload limit in percent for the Inbound Commit Control provider X overloaded by Y% event. The event is raised if the inbound traffic for a provider exceeds the configured value (refer [peer.X.95th](#)) by more than this limit.

Possible values: 1 - 1000

Default value: 10

Recommended value: 10

4.1.10.28 trap.core_cc_provider_overload.limit_mbps

Defines the per provider overload limit in Mbps for the Outbound Commit Control provider X overloaded by Y Mbps event. The event is raised if the outbound traffic for a provider exceeds the configured value (refer [peer.X.95th](#)) by more than this limit.

Possible values: 1 - 10000

Default value: 100

Recommended value: 100

4.1.10.29 trap.core_cc_provider_overload.limit_pct

Defines the per provider overload limit in percent for the Outbound Commit Control provider X overloaded by Y% event. The event is raised if the outbound traffic for a provider exceeds the configured value (refer [peer.X.95th](#)) by more than this limit.

Possible values: 1 - 1000

Default value: 10

Recommended value: 10

4.1.10.30 trap.core_improvement_latency.limit_ms

Defines the packet latency limit in milliseconds.

Possible values: 1 - 10000

Default value: 300

Recommended value: 300

4.1.10.31 trap.core_improvement_loss.limit_pct

Defines the packet loss percentage limit.

Possible values: 1 - 100

Default value: 50

Recommended value: 50

4.1.10.32 trap.destination.auth_password

SNMP user's password for IRP generated traps. Applicable only when SNMP v3 with authentication is used. Refer to [trap.destination.version](#), [trap.destination.seclevel](#), [trap.destination.auth_username](#).

Possible values: text

4.1.10.33 trap.destination.auth_protocol

SNMP authentication protocol for IRP generated traps. Applicable only when SNMP v3 with authentication is used. Refer to [trap.destination.version](#), [trap.destination.seclevel](#).

- 0 - MD5
- 1 - SHA

Possible values: 0, 1

Default value: 1

4.1.10.34 trap.destination.auth_username

SNMP user name for IRP generated traps. Applicable only when SNMP v3 with authentication is used. Refer to [trap.destination.version](#), [trap.destination.seclevel](#).

Possible values: text

4.1.10.35 trap.destination.community

Defines the SNMP community used for sending SNMP traps.

Possible values: textual community name

4.1.10.36 trap.destination.port

Defines the UDP port used for sending SNMP traps.

Possible values: 1 - 65535

Default value: 162

Recommended value: 162

4.1.10.37 trap.destination.priv_password

SNMP password for IRP generated traps. Applicable only when SNMP v3 with privacy is used. Refer to [trap.destination.version](#), [trap.destination.seclevel](#).

Possible values: text

4.1.10.38 trap.destination.priv_protocol

SNMP encryption protocol for IRP generated traps. Applicable only when SNMP v3 with privacy is used. Refer to [trap.destination.version](#), [trap.destination.seclvl](#).

- 0 - DES
- 1 - AES

Possible values: 0, 1

Default value: 1

4.1.10.39 trap.destination.seclvl

SNMP security services for IRP generated traps. IRP supports either No security, Authentication only or both Authentication and Privacy. Applicable only when SNMP v3 is used ([trap.destination.version](#)). Depending on security services used further parameters should be configured, for example: [trap.destination.auth_username](#), [trap.destination.auth_password](#), [trap.destination.auth_protocol](#), [trap.destination.priv_password](#), [trap.destination.priv_protocol](#).

- 0 - Neither Authentication nor Privacy
- 1 - Authentication only
- 2 - Authentication and Privacy

Possible values: 0, 1, 2

Default value: 0

4.1.10.40 trap.destination.version

SNMP version for IRP generated traps.

- 2 - SNMP v2c
- 3 - SNMP v3

Possible values: 2, 3

4.1.11 Administrative settings

4.1.11.1 dbcron.api.log

Defines file-system path to the dbcron API log file.

Default value: /var/log/irp/api.mlog

4.1.11.2 dbcron.api.socket

Defines dbcron API listen socket.

Default value: /var/spool/irp/api

4.1.11.3 dbcron.log

Defines the file-system path to the dbcron log file.

Default value: /var/log/irp/dbcron.log

4.1.11.4 dbcron.log.level

Defines the logging level for the dbcron service.

Possible values: *emerg, fatal, alert, crit, error, warn, notice, info, debug*

Default value: *info*

Recommended value: *info*

4.1.11.5 globalcc.log

Defines the file-system path to the Globalcc log file.

Default value: */var/log/irp/globalcc.log*

4.1.11.6 globalcc.log.level

Defines the logging level for the Globalcc service.

Possible values: *emerg, fatal, alert, crit, error, warn, notice, info, debug*

Default value: *info*

Recommended value: *info*

4.1.11.7 irpdetectd.log

Defines the file-system path to the Irpdetectd log file.

Default value: */var/log/irp/irpdetectd.log*

4.1.11.8 irpdetectd.log.level

Defines the logging level for the Irpdetectd service.

Possible values: *emerg, fatal, alert, crit, error, warn, notice, info, debug*

Default value: *info*

Recommended value: *info*

4.1.11.9 irpstatd.log

Defines the file-system path to the irpstatd log file.

Default value: */var/log/irp/irpstatd.log*

4.1.11.10 irpstatd.log.level

Defines the logging level for the irpstatd service.

Possible values: *emerg, fatal, alert, crit, error, warn, notice, info, debug*

Default value: *info*

Recommended value: *info*

4.1.11.11 irpstatd.snmp.enhanced.sec

Defines during what second of a minute the SNMP enhanced algorithm should run.

Possible values: 30–58

Default value: 45

Recommended value: 45

4.1.12 Providers settings

4.1.12.1 peer.X.95th

 Mandatory if `core.commit_control` is enabled.

Defines current provider's desired 95th value (also known as Committed Interface Rate).

The value of `peer.X.limit_load`(if set to positive value) must be greater or equal to value of `peer.X.95th`. Please refer to `core.commit_control` and Commit Control (Not supported in IRP Lite) for Commit Control description.

Depending on the `peer.X.95th.bill_day` parameter value, the behavior of the 95th level calculation has two different states.

If `peer.X.95th.bill_day` is set to `-1`, then this parameter is treated as "committed interface rate limit" and actual 95th calculation is not performed. In this case, IRP will actively reroute traffic from an overloaded provider to keep it below the `peer.X.95th` level.

If `peer.X.95th.bill_day` is set to a positive value, then the system calculates the actual 95th value based on the historical traffic information from `peer.X.95th.bill_day` to the current date of this month. Then, the result is compared to the `peer.X.95th` value, and the maximum of these two is chosen as the target 95th. Thus, if the 95th for the current month has already exceeded the `peer.X.95th` value, IRP will keep the traffic usage from increasing above current 95th.

Possible values: 1–9999999

4.1.12.2 peer.X.95th.bill_day

Defines the first billing period day. If current month has less days than the specified value, then last day of the month will be taken as the start of the new billing period. The parameter is used for 95th percentile calculation.

If `-1` is specified, then `peer.X.95th` is treated as "Committed Interface Rate".

Possible values: `-1`, 1–31

Default value: 1

Recommended value: First day of the billing period.

4.1.12.3 peer.X.95th.centile

Defines the actual percentage for the 95th percentile, as some providers may have non-standard traffic billing policies.

Possible values: 1–99

Default value: 95

Recommended value: please consult your agreements with this specific provider

4.1.12.4 peer.X.95th.in


Defines the 95th level (in Mbps) for inbound when inbound and outbound commit control operate independently. Refer [peer.X.95th](#), [peer.X.95th.mode](#).

Possible values: 1-1000000

4.1.12.5 peer.X.95th.mode

Defines the way how 95th value is determined. The following modes are supported:

- 0: Separate 95th for in/out
- 2: 95th from greater of in or out
- 3: Largest of the two 95th for in/out

 Only mode with separate 95th for each inbound and outbound traffic keeps inbound and outbound commit control independent of each other.

Refer also [peer.X.95th](#), [peer.X.95th.in](#).

Possible values: 0-3

Default value: 2

4.1.12.6 peer.X.aspath_for_ix

Modifies the way IX AS number is treated for outbound improvements towards an Internet Exchange provider for 2nd and 3rd options of [bgpd.as_path](#). The supported options are:

- 0: Strip IX ASN if present
- 1: Preserve IX ASN if present


Refer also [bgpd.as_path](#).

Possible values: 0-1

Default value: 0

4.1.12.7 peer.X.auto_config


Enables or disables periodic execution of Internet Exchanges auto re-configuration process once per [4.1.2.6](#) interval.

 Explorer will be restarted after the process is completed.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.8 peer.X.bgp_peer

 Mandatory

Defines the iBGP session(s) over which an improvement made for this provider is advertised. This parameter should be set to a valid BGP neighbor name as defined in [BGP sessions settings](#). Typically, this is the session with the Edge router, which this provider is connected to. Multiple sessions should be set only if there are some additional (backup) links to this provider on multiple Edge routers.

Possible values: One or a list of valid BGP neighbor names, as defined in [BGP sessions settings](#).

4.1.12.9 peer.X.blackholing.bgp_peer

Specifies a BGP session(s) to be used to send blackholing announcements to.

Possible values: One or a list of valid BGP neighbor names, as defined in [BGP sessions settings](#).

4.1.12.10 peer.X.blackholing.community

Specifies a BGP community mark to be used in order to distinguish routes that are to be sent towards the provider's router responsible for blackhole routes.

4.1.12.11 peer.X.bmp

Defines BMP data usage for this provider.


- 0: Do not use BMP data
- 1: Use BMP data if available
- 2: Use BMP data only

Possible values: 0 - 2

Default value: 0

4.1.12.12 peer.X.bmp.check_routes

Allows using BMP to check route availability from a Provider configured as a partial routing or IX.

 IRP may unexpectedly use routes received from a Provider, but filtered by an edge router, because BGP filters aren't applied to BMP data.

The parameter must be enabled before switching `peer.X.bmp` to "Use BMP data only" mode.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.13 peer.X.cc_disable

Instructs IRP to exclude this provider from the Commit Control algorithm.

 Has effect only if `core.commit_control` is enabled.


Possible values: 0, 1

Default value: 0

4.1.12.14 peer.X.circuit.control

Defines whether provider will be monitored for excessive loss issues and how to react to them. The possible values are:

- 0: Disabled
- 1: Warn only
- 2: Warn and disconnect
- 3: Disconnect and restore


 IRP tries to induce a disconnect of BGP session via FlowSpec rules that drop any further packets on the BGP session with provider's router. It is thus recommended that the keepalive timer is set to 20 seconds for BGP sessions with providers that are expected to be disconnected if circuit issues are detected on them. This will minimize the time a circuit with excessive loss is kept operational and causes harm.

Possible values: 0-3

Default value: 0

4.1.12.15 peer.X.cost

Defines the cost per Mbps for the current provider. All `peer.X.cost` parameters should be specified in the same currency.

 Parameter's value should be the same for each provider in a provider's group (refer to 4.1.12.53) and cost mode is enabled (4.1.2.26)

Possible values: 1-'unlimited'

Default value: 0

4.1.12.16 peer.X.description

Defines the provider's description (full provider name)

Possible values: text

4.1.12.17 peer.X.diag_hop.interval_max

Defines maximum value for Adaptive Inter-Packet Interval. Adaptive Inter-Packet Interval is used while sending packets to a specific Provider's router (packets with a specific TTL). Inter-packet interval is raised up to the maximum value when the router does not respond to the packets and is decreased to the minimum one in case the router responds. The Adaptive Inter-Packet Interval is changed gradually to obtain better performance.

A value of the 4.1.9.5 parameter is used in case zero value is specified in this parameter.

Possible values: 0 - 50000000 (50 ms)

Default value: 0

4.1.12.18 peer.X.diag_hop.interval_min

Defines minimum value for Adaptive Inter-Packet Interval. Adaptive Inter-Packet Interval is used while sending packets to a specific Provider's router (packets with a specific TTL). Inter-packet interval is raised up to the maximum value when the router does not respond to the packets and is decreased to the minimum one in case the router responds. The Adaptive Inter-Packet Interval is changed gradually to obtain better performance.

Possible values: 10000 (0.01 ms) - 10000000 (10 ms)

Default value: 1000000

4.1.12.19 peer.X.disable_pbr_confirmation

Set to disable periodic transit/partial routing provider's PBR confirmation check.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

Recommended value: 0


4.1.12.20 peer.X.flow_agents

Specifies a collection of Flow agents in the form IPv4/interfaceIdentifier. Flow agents are used to assign specific Flow statistics to a designated provider. Refer [Flow agents](#), [Optimization for multiple Routing Domains](#) for details.

Possible values: forward slash separated IP address and numeric identifier in 1..2147483647 range


4.1.12.21 peer.X.flowspec.ipv4.redirect_community

Defines the BGP Community that will be appended by Bgpd to advertised Flowspec redirect policies for IPv4 sessions. The format is: "X:Y".

 Avoid collisions of communities values assigned to IRP both within its configuration and/or on customer's network.

4.1.12.22 peer.X.flowspec.ipv6.redirect_community

Defines the BGP Community that will be appended by Bgpd to advertised Flowspec redirect policies for IPv6 sessions. The format is: "X:Y".

 Avoid collisions of communities values assigned to IRP both within its configuration and/or on customer's network.

4.1.12.23 peer.X.global_group

Includes the provider into a existing Global Group.

See also: [Global Group settings](#)

Possible values: 1-100

4.1.12.24 peer.X.group_loadbalance

Defines whether commit control algorithm load balances the group of providers or optimizes it on aggregated bandwidth usage. For details refer [Provider load balancing](#) and [Commit control of aggregated groups](#).

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.12.25 peer.X.improve_in_group

Enables or disables improvements within a provider group. This parameter must be equal for all providers in a provider group.

If disabled, IRP will not make any Cost or Performance improvements inside the group.


Possible values: 0 (OFF), 1 (ON)

Default value: 0

Recommended value: depends on network infrastructure and policies


4.1.12.26 peer.X.inbound.community_base

Defines the base community of consecutive range of eight community values which IRP uses to announce inbound improvements for this provider over BGP. Base community instructs an announcement of a route without any additional prepend via a particular provider. Next communities in the range instructs for additional prepends from one up to seven.

 For example: when `peer.1.inbound.community_base` is set to `65530:50100`, the value `65530:50102` represents “prepend 2 times while advertizing to provider 1”.


Possible values: valid BGP community attribute

4.1.12.27 peer.X.ipv4.diag_hop

 Mandatory

Defines the diagnostic hop or subnet in CIDR format for the current provider. Usually, it is the IP address of the first hop of the specific provider. The parameter is used to make sure that a route actually passes through this provider.

It is also used for Internet Exchanges autoconfiguration process. The Peering Partners next-hops will be gathered based on the provided subnet(s) in CIDR format.

 Any parameter changes (via Frontend) will caused the Bgpd component restart if the provider type (`peer.X.type`) is Internet Exchanges

Possible values: valid IPv4 address (usually equal to `peer.X.ipv4.next_hop`) or subnet definition in CIDR format

4.1.12.28 peer.X.ipv4.master_probing

⚠ Mandatory

Defines the local IPv4 address, assigned for probing via the current provider. When failover is enabled (`global.failover`) slave's probing IP (`peer.X.ipv4.slave_probing`) must be configured too.

See also: [Specific PBR configuration scenarios](#)

Possible values: valid local IPv4 address

4.1.12.29 peer.X.ipv4.mon

Defines the List of IPv4 addresses to be monitored by Bgpd to ensure that the immediate upstream path through this provider is operational. Each specified IP address is probed every `bgpd.mon.keepalive` seconds to verify accessibility from Bgpd (ICMP/UDP pings are used). Highly available IP addresses that reliably answer to ICMP/UDP pings should be used. It is recommended to setup at least two IP addresses belonging to different networks.

If all monitored IP addresses do not respond for `bgpd.mon.holdtime` seconds, then any improvements designated for this provider will be withdrawn from edge router(s). Improvements will be re-announced after all IP addresses respond within `bgpd.mon.guardtime` seconds.

Possible values: space-separated list of valid IPv4 addresses

Default value: 208.67.222.222 8.8.8.8

4.1.12.30 peer.X.ipv4.next_hop**⚠ Mandatory**

Defines the next-hop IPv4 address for BGP route injection. Usually, it is the IPv4 address of the BGP partner from the provider.

Possible values: valid IPv4 address

4.1.12.31 peer.X.ipv4.next_hop_as

Defines the provider autonomous system number for IPv4. This parameter must be set in order for the 3rd algorithm of Bgpd AS-PATH to work properly (refer to `bgpd.as_path`).

If this parameter is set, then its value will be used as part of AS-PATH for outgoing improvements if the 3rd algorithm is enabled in `bgpd.as_path`.

In case of Internet Exchanges, the AS-Path begins with peering partner's AS number, instead of AS number of route server.

The first AS number will be stripped from AS-Path when advertising improvement towards Exchange in case the first AS number is equal to value set in this parameter.

See also: `provider.X.rule.Y.next_hop_as`

Possible values: 0-4294967295

Default value: 0

4.1.12.32 peer.X.ipv4.route_server

Defines the Internet Exchange Route Server(s) IP address(es) used to properly auto-configure `provider.X.rule.Y.bgp_peer` parameter.

See also `peer.X.mon.ipv4.internal.mode`, `peer.X.mon.ipv4.internal.state`

Possible values: valid IPv4 address(es)

4.1.12.33 peer.X.ipv4.slave__probing

Defines the local IPv4 address, assigned for probing via the current provider for the slave node in a failover configuration.

See also: [Specific PBR configuration scenarios](#)


4.1.12.34 peer.X.ipv4__pbr__check

Defines per-provider alternative IPv4 address to be used for PBR tests. This overrides `explorer.ipv4__test`.

See also: [peer.X.ipv6__pbr__check](#)

Possible values: valid IPv4 address


4.1.12.35 peer.X.ipv6.diag__hop

 Mandatory

Defines the IPv6 diagnostic hop for the current provider. Usually, it is the IP address of the first hop of the specific provider. The parameter is used to verify that a route actually passes through this provider.

Possible values: valid IPv6 address, usually equals to `peer.X.ipv6.next__hop`

4.1.12.36 peer.X.ipv6.master__probing

 Mandatory for IPv6

Defines the local IPv6 address assigned for probing via the current provider. When failover is enabled (`global.failover`) slave's probing IPv6 address (`peer.X.ipv4.slave__probing`) must be configured too.

See also: [Specific PBR configuration scenarios](#)

Possible values: valid local IPv6 address

4.1.12.37 peer.X.ipv6.mon


Defines the List of IPv6 addresses to be monitored by Bgpd to ensure that the immediate upstream path through this provider is operational. Each specified IP address is probed every `bgpd.mon.keepalive` seconds to verify accessibility from Bgpd (ICMP/UDP pings are used). Highly available IP addresses that reliably answer to ICMP/UDP pings should be used. It is recommended to setup at least two IP addresses belonging to different networks.

If all IP addresses do not respond for `bgpd.mon.holdtime` seconds, then any improvements designated for this provider will be withdrawn from the edge router(s). Improvements will be announced again after all IP addresses respond within `bgpd.mon.guardtime` seconds.

Possible values: space-separated list of valid IPv6 addresses

Default value: 2620:0:ccc::2 2001:4860:4860::8888

4.1.12.38 peer.X.ipv6.next__hop

 Mandatory for IPv6

Defines the next-hop IPv6 address for BGP route injection. Usually, it is the IPv6 address of the BGP partner from the provider.

Possible values: valid IPv6 address

4.1.12.39 peer.X.ipv6.next_hop_as

Defines the provider autonomous system number for IPv6. This parameter must be set in order for the 3rd algorithm of Bgpd AS-PATH to work properly (refer to [bgpd.as_path](#)).

If this parameter is set, then it's value will be used as part of an AS-PATH for outgoing improvements if the 3rd algorithm is enabled in [bgpd.as_path](#).

See also: [provider.X.rule.Y.next_hop_as](#)

Possible values: 0-4294967295

Default value: 0

4.1.12.40 peer.X.ipv6.route_server

Defines the Internet Exchange Route Server(s) IP address(es) used to properly auto-configure [provider.X.rule.Y.bgp_peer](#) parameter.

See also [peer.X.mon.ipv6.internal.mode](#), [peer.X.mon.ipv6.internal.state](#)

Possible values: valid IPv6 address(es)

4.1.12.41 peer.X.ipv6.slave_probing

Defines the local IPv6 address assigned for probing via the current provider for the slave node in a failover configuration.

See also [peer.X.ipv6.master_probing](#).

Possible values: valid local IPv6 address

4.1.12.42 peer.X.ipv6_pbr_check

Defines per-provider alternative IPv6 address to be used for PBR tests. This overrides [explorer.ipv6_test](#).

See also: [peer.X.ipv4_pbr_check](#).

Possible values: valid IPv6 address

4.1.12.43 peer.X.limit_load

Defines the load limit for current provider. IRP will improve routes to this provider as long as its load is less than the specified value in megabits per second. SNMP must be properly configured in order for this feature to work properly. If this limit is configured, but it is impossible to retrieve interface data, no new improvements will take place.

The value of [peer.X.limit_load](#)(if set to positive value) must be greater or equal to value of [peer.X.95th](#).

Possible values: -1 (unlimited), 1-1000000

Default value: -1

Recommended value: it is recommended to set it to not more than ~60-80% of the physical interface rate

See also:

[global.ifstats](#)

[peer.X.snmp.interfaces](#)

4.1.12.44 peer.X.mon.ipv4.bgp_peer

Defines the current provider's IPv4 address that must be monitored by Bgpd, usually equal to [peer.X.ipv4.next_hop](#)

Possible values: valid IPv4 address

See also: [peer.X.mon.snmp](#)

4.1.12.45 peer.X.mon.ipv4.external.state

Enables or disables external monitors for IPv4.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.46 peer.X.mon.ipv4.internal.mode

Specifies router supported BGP MIB to monitor this provider IPv4 BGP sessions.

- 0 - Generic (BGP4-MIB)
- 1 - Cisco (CISCO-BGP4-MIB)
- 2 - Juniper (BGP4-V2-MIB-JUNIPER)
- 3 - Brocade (draft-ietf-idr-bgp4-mibv2-12)

Possible values: 0, 1, 2, 3

Default value: 0

4.1.12.47 peer.X.mon.ipv4.internal.state

Enables or disables internal monitors for IPv4.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.48 peer.X.mon.ipv6.bgp_peer

Defines the current provider's IPv6 address that must be monitored by Bgpd, usually equal to [peer.X.ipv6.next_hop](#)

Possible values: valid IPv6 address

See also: [peer.X.mon.snmp](#)

4.1.12.49 peer.X.mon.ipv6.external.state

Enables or disables external monitors for IPv6.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.50 peer.X.mon.ipv6.internal.mode

Specifies router supported BGP MIB to monitor this provider IPv6 BGP sessions.

- 1 - Cisco (CISCO-BGP4-MIB)
- 2 - Juniper (BGP4-V2-MIB-JUNIPER)
- 3 - Brocade (draft-ietf-idr-bgp4-mibv2-12)

Possible values: 1, 2, 3

4.1.12.51 peer.X.mon.ipv6.internal.state

Enables or disables internal monitors for IPv6.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.52 peer.X.mon.snmp

Identifier of SNMP host to use for monitoring this provider. Refer to [SNMP Hosts settings](#), `snmp.X.name`.

Possible values: text

4.1.12.53 peer.X.precedence

 Mandatory if `core.commit_control` is enabled.

Defines this provider's precedence, used for provider grouping and defining commit priorities.

See `core.commit_control` and [Commit Control \(Not supported in IRP Lite\)](#) for Commit Control functionality.

 All providers belonging to one group must have equal cost (`peer.X.cost`)

If two different providers have the same `peer.X.precedence` value, then these providers are considered to be a group, and traffic is balanced between them.

The provider or a group with the lowest precedence value are also used by Commit Control as last resort destination (if all the providers are exceeding their 95th, then traffic is rerouted to the upstream with the smallest precedence - usually this upstream has either the highest throughput, or the lowest cost).

For example:

```
peer.1.precedence = 20
peer.2.precedence = 10
peer.3.precedence = 30
```

If the `peer.X.95th` is exceeded for all configured providers, then the excessive traffic from providers 1 and 3 will be rerouted to the 2nd provider.

If provider groups are used:

```
peer.1.precedence = 50
peer.2.precedence = 40
peer.3.precedence = 40
peer.4.precedence = 40
peer.5.precedence = 30
```

Traffic between providers 2, 3 and 4 is evenly distributed. If all providers are already overloaded, then the excessive traffic will be rerouted to the 5th provider (since it has the lowest precedence value).

Possible values: 0-100

4.1.12.54 peer.X.rd

Assigns a routing domain identifier to the provider. Providers in the same routing domains should be assigned the same identifier.

Providers with identifier 1 (one) are assumed to be in the same routing domain that hosts IRP instance. Refer `global.rd_rtt`, `peer.X.flow_agents`, `bgpd.rd_local_mark`.

Possible values: 1-100

Default value: 1

4.1.12.55 peer.X.routes_config


Defines whether Internet Exchanges auto configuration is enabled or not. In case it is enabled, IRP Bgpd gathers Peering Partners (next-hops) with their announced routes and stores the data in IRP database. Otherwise, manual configuration is required.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

Recommended value: 1

4.1.12.56 peer.X.shortname

 **Mandatory**

Defines the provider's short, abbreviated name (3-20 characters). This parameter's value is used for reports and graphs.

Possible values: text

4.1.12.57 peer.X.shutdown

Defines whether provider is Active (0), Suspended (1) or Shutdown (2).

After the provider suspend action, all the improvements will be stored for short period of time (1h). In case of short maintenance windows, use provider suspend.

After the provider shutdown action, all the improvements will be removed. In case of long maintenance windows, use provider shutdown.

After the provider reactivation (after a previous suspend), all the improvements will be sent to retry probing.

The provider state can be updated in Frontend or using the `irpPeerSuspendShutdown.pl` script.

In order to get the provider suspended, shutdown or reactivated, use the `irpPeerSuspendShutdown.pl` helper script.

```
Usage: /usr/bin/irpPeerSuspendShutdown.pl [-a] [-p] [-h]

-a|--action - Perform an action under a peer.
           Possible values:
             0 or 'activate'
             1 or 'suspend'
             2 or 'shutdown'
-p|--peerId - An ID of the peer that should be Activated, Shutdown or
             Suspended
-h|--help - Print this help
```

Possible values: 0, 1, 2

Default value: 0

4.1.12.58 peer.X.snmp.enhanced

Enables or disables heuristically enhanced SNMP collection for provider.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.12.59 peer.X.snmp.interfaces

Defines a collection of interfaces used for current provider. Individual values are in the form SNMPHostID:Interface where

- SNMPHostID is the identifier of an SNMP host configured under 4.1.18
- Interface could be specified either by SNMP index (ifIndex) or by name (ifDescr).

Possible values: collection of SNMPHostID:Interface pairs

4.1.12.60 peer.X.type

Defines the type of a provider. The following types are available:

- 0 - Transit provider
- 1 - Partial routes provider
- 2 - Exchanges provider

Possible values: 0, 1, 2

Default value: 0

4.1.13 Inbound settings

4.1.13.1 inbound.rule.X.bgp_peer

Defines the router(s) where IRP announces improvements of this inbound prefix.

Possible values: a router identifier(s) separated by space

4.1.13.2 inbound.rule.X.enabled

Enables or disables Inbound Optimization for this prefix.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.13.3 inbound.rule.X.full_control

Specifies individual inbound prefix control level by IRP. Default behavior can be set at system level under `bgpd.full_control`. The settings specify whether default behavior is inherited from the system level or set individually for this prefix to announce either Improvements only or fully announce the prefix to allowed providers.

Refer to `full_control` for details.


Possible values: -1 (Use default), 0 (Improvement), 1 (If improved), 2 (All)

Default value: -1

4.1.13.4 inbound.rule.X.next_hop

Defines a `next_hop` specific for the inbound rule. When the `next_hop` is unset, then a value from `bgpd.peer.X.inbound.ipv4.next_hop/bgpd.peer.X.inbound.ipv6.next_hop` is taken.


Next-hop should point to a router that routes/terminates traffic. Otherwise null route should be configured for the Next-Hop.


 `next_hop` value should be of the same IP address family as the inbound prefix it is assigned to. Refer `inbound.rule.X.prefix`.

Possible values: IP address

4.1.13.5 inbound.rule.X.prefix

Defines an inbound prefix. Inbound prefixes should belong to your network.

 Inbound Optimization of transit networks will be supported in future releases.

 Each inbound prefix logically extends the list of ournets when it is processed by IRP Collector. Refer `collector.ournets`.

Possible values: prefix in CIDR format

4.1.13.6 inbound.rule.X.providers

Defines the list of providers where this inbound prefix can be advertised. In case this parameter value is empty then it is treated as prefix will be advertised to ALL providers.

Possible values: providerIDs collection

4.1.14 Routing Policies settings

➔ IRP used a legacy format for policies.conf where a single policy parameters immediately followed the prefix/ASN that defined them. IRP updated policies.conf format and assigned each policy a unique identifier of the form “policy.X.” used to prefix each attribute. Notice the format changes in the samples below.

Legacy routing policy example:

```
asn=65530
vip=0
policy=deny
providers=1 3 5
enabled=1
notes=AS65530 deny via Level3, Cogent and nLayer
```

Normalized routing policy example:

```
policy.1.asn=65530
policy.1.vip=0
policy.1.policy=deny
policy.1.providers=1 3 5
policy.1.enabled=1
policy.1.notes=AS65530 deny via Level3, Cogent and nLayer
```

4.1.14.1 policy.X.asn/country/prefix

Defines the Country, ASN or a prefix to which the routing policy is applied. Only one of these parameters allowed per policy. Country represents the two letter ISO Country codes such as US, UK, MD to designate United States of America, United Kingdom or Republic of Moldova.

Possible values: 1–65535 for an ASN or Country Code for a country or a valid IPv4/IPv6 prefix.

4.1.14.2 policy.X.cascade

Defines the ASN policies that cascade to downstream AS from designated AS. The parameter applies to ASN policies (and not to prefixes).

Possible values: 0 (Disable), 1 (Enable)

Default value: 0

4.1.14.3 policy.X.community

A community to mark improvements belonging to a policy.

Possible values: X:Y

4.1.14.4 policy.X.enabled

Defines whether the policy is enabled or not

Possible values: 0 (OFF), 1 (ON)

Default value: 1

4.1.14.5 policy.X.forcelocal

A flag indicating if a global improvement allowed or not.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.14.6 policy.X.notes

Defines a description of the routing policy

Possible values: text

4.1.14.7 policy.X.policy

Defines the type of policy

Possible values: allow, deny, static

Default value: allow

4.1.14.8 policy.X.priority

Defines what policy to prioritize in case of overlapping prefixes as might be the case of a large aggregate prefix policy extending over an ASN policy. The same specific prefixes might be covered by both and priority sets which one to actually enforce in favor of the others. Policies with higher priority are prioritized.

Possible values: 0-50

Default value: 0

4.1.14.9 policy.X.providers

Defines the list of providers (IDs) affected by the policy. If the 'providers' property is not specified, then all the providers will be included in the policy by default.

Possible values: 1-unlimited

4.1.14.10 policy.X.vip

Defines whether VIP probing is enabled for the specified policy.

Possible values: 0 (OFF), 1 (ON)

Default value: 0

4.1.15 Exchanges settings

4.1.15.1 provider.X.rule.Y.bgp__peer

Defines the BGP Router-ID used for BGP session state monitoring, which is performed by the Internal BGP Monitor (BGP Monitoring).

Depending on whether a Route Server or Peering sessions are used, the parameter value should be set to Route Server Router-ID or Peering Partner Router-ID accordingly.

Possible values: IPv4 address

4.1.15.2 provider.X.rule.Y.enabled

Defines whether the rule is enabled or not.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

4.1.15.3 provider.X.rule.Y.next_hop

Defines the Peering Partner's IPv4/IPv6 next-hop address.

Possible values: IPv4/IPv6 address

4.1.15.4 provider.X.rule.Y.next_hop_as

Defines the peering partner's autonomous system number. This parameter must be set in order for the 3rd algorithm of Bgpd AS-PATH to work properly (refer to [bgpd.as_path](#)).

If this parameter is set, then it's value will be used as part of AS-PATH for outgoing improvements if the 3rd algorithm is enabled in [bgpd.as_path](#).

In case of Internet Exchanges, the AS-Path begins with peering partner's AS number, instead of AS number of route server.

The first AS number will be stripped from AS-Path when advertising improvement towards Exchange, in case the first AS number is equal to the value set in this parameter.

See also: [peer.X.ipv4.next_hop_as](#), [peer.X.ipv6.next_hop_as](#)

Possible values: 0-4294967295

Default value: 0

4.1.15.5 provider.X.rule.Y.pbr_check

Defines if PBR check for a configured Peering Partner is enabled or not.

Possible values: 0 (OFF), 1 (ON)

Default value: 1

4.1.15.6 provider.X.rule.Y.probing_dscp

Defines the DSCP (Differentiated services code point) used with [provider.X.rule.Y.probing_ip](#) for a specific Peering Partner.

Possible values: 0-63

4.1.15.7 provider.X.rule.Y.probing_ip

Defines the probing IPv4/IPv6 address used for a specific Peering Partner.

Possible values: IPv4/IPv6 address

4.1.15.8 provider.X.rule.Y.shortname


Defines a short name for a configured Peering Partner.

Possible values: text

4.1.16 Global Group settings

4.1.16.1 globalgroup.X.members

Defines a list of IP addresses or host names of all IRP instances (including Failover slave nodes) to form the Global Group.

 Value of the parameter must be the same on all instances of the Global Group.

4.1.16.2 globalgroup.X.outbound.95th

Amount of bandwidth provided by this IRP instance to the Global Group.

If the value -1 is set, then IRP automatically uses total amount of outbound bandwidth set in the `peer.X.95th` parameter for all the providers in the Global Group.

Possible values: -1 (follow Providers), 1 - 1000000

4.1.16.3 globalgroup.X.outbound.95th.reserve


Amount of bandwidth reserved to operation of this IRP instance.

Possible values: 0 (not used) - 1000000

Default value: 0

4.1.16.4 globalgroup.X.outbound.rate__high

Defines Global Group's high load rate limit (%).

 Value of the parameter must be the same on all instances of the Global Group.


See also: [core.commit_control.rate.high](#)

Possible values: 30-99, but greater than or equal to [globalgroup.X.outbound.rate_low](#)

Default value: 95

4.1.16.5 globalgroup.X.outbound.rate__low

Defines Global Group's low load rate limit (%).

 Value of the parameter must be the same on all instances of the Global Group.


See also: [core.commit_control.rate.low](#)

Possible values: 30-99, but lower than or equal to [globalgroup.X.outbound.rate_high](#)

Default value: 85

4.1.16.6 globalgroup.X.shortname

Defines short name of the Global Group.

 Value of the parameter must be the same on all instances of the Global Group.

4.1.17 User Directory settings

4.1.17.1 directory.X.admin_role_groups

Defines the specific objects/groups that list IRP users with Admin privileges. Also refer `directory.X.user_role_attr`.

Possible values: text

4.1.17.2 directory.X.base_dn

Defines the base DN (distinguished name) of users in the directory. Base DN is used in conjunction with `directory.X.username` and `directory.X.user_dn` to uniquely identify and query the directory during authentication.

Possible values: text

4.1.17.3 directory.X.email

Defines the directory attribute that stores user's email address.

Possible values: text

4.1.17.4 directory.X.fullname

Defines the directory attribute that stores user's full name.

Possible values: text

4.1.17.5 directory.X.hostname

Defines the hostname or the IP address of the User Directory.

Possible values: domain name or IP address

4.1.17.6 directory.X.initial_bind_dn

Defines User DN assigned to IRP to bind to directory. This user's password is configured in `directory.X.initial_bind_password`.

Possible values: text

4.1.17.7 directory.X.initial_bind_password

Defines the password assigned to IRP to bind to directory. Refer `directory.X.initial_bind_dn`.

Possible values: text


4.1.17.8 directory.X.name

Assigns a name to user directory within IRP.

Possible values: text

4.1.17.9 directory.X.order

Defines the sequence this user directory is queried when multiple user directories are configured. Smaller values take precedence.

 The value must be unique across configured user directories.

Possible values: 0–255

Default value: 255

4.1.17.10 directory.X.port

Defines the port used by the directory, for example 389 for LDAP.

Possible values: 0–65535

4.1.17.11 directory.X.secret

Defines the shared secret for TACACS+ host and IRP to secure communications.

Possible values: text

4.1.17.12 directory.X.state

Enables or disables use of this directory within IRP.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 1

4.1.17.13 directory.X.timeout

Defines the timeout (in seconds) after which IRP stops trying to communicate with the directory.

Possible values: 0–300

Default value: 60

4.1.17.14 directory.X.tls

Enables or disables use of TLS for this directory.

Possible values: 0 (Disabled), 1 (Enabled)

Default value: 0

4.1.17.15 directory.X.tls_cacertfile

Defines the path to a CA certificate file used when server certificate validation is enabled and CA certificate cannot be included in trusted server certificates store, for example a single purpose self-signed CA certificate is used. Refer [directory.X.verify_cert](#).

Possible values: text (path)

4.1.17.16 directory.X.type

Defines user directory type for example LDAP, Active Directory, Internal etc.

Possible values: `internal`, `ldap_posix`, `active_directory...`

Default value: `internal`

4.1.17.17 directory.X.user_dn

Defines the DN of users. User DN is used in conjunction with `directory.X.username` and `directory.X.base_dn` to uniquely identify and query the directory during authentication. Only users matching `directory.X.user_filter` are granted access to the system.

Possible values: `text`

4.1.17.18 directory.X.user_filter

Defines a filter used to match users that will be granted access. The filter uses standard LDAP filter syntax with the exception of outermost paranthesis that are expected to be dropped from configuration.

Possible values: `text`

4.1.17.19 directory.X.user_role_attr

Defines the attribute that stores the list of users in a role/group. Refer `directory.X.admin_role_groups`.

Possible values: `text`

4.1.17.20 directory.X.username

Defines the directory attribute that represents the username of users in the directory. This attribute together with `directory.X.user_dn` and `directory.X.base_dn` are used to uniquely identify and query the directory during authentication.

Possible values: `text`

4.1.17.21 directory.X.verify_cert

Enables or disables verification of directory server certificate when TLS is enabled. When verification is enabled a path to CA certificate file must be provided too. Refer `directory.X.tls`, `directory.X.tls_ca_certfile`.

Possible values: `0` (Disabled), `1` (Enabled)

Default value: `0`

4.1.18 SNMP Hosts settings

4.1.18.1 snmp.X.auth_password

User's password used to authenticate SNMP communications. Applicable only when SNMP v3 with authentication is used. Refer to `snmp.X.version`, `snmp.X.seclevel`, `snmp.X.auth_username`.

Possible values: `text`

4.1.18.2 snmp.X.auth_protocol

SNMP authentication protocol. Applicable only when SNMP v3 with authentication is used. Refer to [snmp.X.version](#), [snmp.X.seclvl](#).

- 0 - MD5
- 1 - SHA


Possible values: 0, 1

4.1.18.3 snmp.X.auth_username

SNMP user name. Applicable only when SNMP v3 with authentication is used. Refer to [snmp.X.version](#), [snmp.X.seclvl](#).

Possible values: text

4.1.18.4 snmp.X.community


 Mandatory parameter

Defines the SNMP community for retrieving interface counters.

Possible values: textual community name

4.1.18.5 snmp.X.ip

Defines IPv4/IPv6 address of the SNMP host.

 Hostnames are not supported.

Possible values: valid IPv4 or IPv6 address

4.1.18.6 snmp.X.name

Sets a shortname for SNMP host.

Possible values: text

4.1.18.7 snmp.X.priv_password

Password used to encrypt SNMP communications. Applicable only when SNMP v3 with privacy is used. Refer to [snmp.X.version](#), [snmp.X.seclvl](#).

Possible values: text

4.1.18.8 snmp.X.priv_protocol

SNMP encryption protocol. Applicable only when SNMP v3 with privacy is used. Refer to [snmp.X.version](#), [snmp.X.seclvl](#).

- 0 - DES
- 1 - AES

Possible values: 0, 1

4.1.18.9 snmp.X.seclevel

SNMP security services. IRP supports either No security, Authentication only or both Authentication and Privacy. Applicable only when SNMP v3 is used (`snmp.X.version`). Depending on security services used further parameters should be configured, for example: `snmp.X.auth_username`, `snmp.X.auth_password`, `snmp.X.auth_protocol`, `snmp.X.priv_password`, `snmp.X.priv_protocol`.

- 0 - None - neither Authentication nor Privacy is used
- 1 - Authentication only
- 2 - Authentication and Privacy

Possible values: 0, 1, 2

Default value: 0

4.1.18.10 snmp.X.version

SNMP version used by host.

- 2 - SNMP v2c
- 3 - SNMP v3

Possible values: 2, 3


Default value: 2

4.1.19 Routing domains settings

4.1.19.1 rd.X.community.local

Specifies a BGP community which will be appended to BGP community attribute of an improvement in the corresponding routing domain.

Parameter represents a valid value for BGP community attribute of the form X:Y. Value in `rd.X.community.local` is APPENDED to communities attribute.

 The value must be unique across all configured BGP communities.

Refer `global.rd_rtt`, `peer.X.rd`, `peer.X.flow_agents`, `bgpd.rd_local_mark`.

Possible values: X:Y

4.1.19.2 rd.X.community_worsening

A BGP community value. The routing domain designated community value is added to announced global outbound improvement if it degraded performance within allowed worsening thresholds.

Possible values: valid BGP community attribute

4.1.19.3 rd.X.shortname

A shortname assigned to each routing domain for human readability.

Possible values: text

Chapter 5

Appendixes

5.1 Frontend labels for configuration parameters

provider.X.rule.Y.next_hop_as
AS Path rules for IX ASN peer.X.aspath_for_ix
AS-PATH restore priority bgpd.as_path
Account ID pushd.sms.account_sid
Adaptive packets count explorer.probing.sendpkts.adaptive_max
Allow borrowing of AS-PATH if can't restore bgpd.as_path_borrowing
Allowed IP addresses global.frontend_acl_ips
Allowed latency worsening (ms) core.cost.worst_ms
Allowed loss worsening (%) core.commit_control.worst_loss
Alternative PBR test IPv4 address peer.X.ipv4_pbr_check
Analyzed prefixes collector.ournets
Announced blackholing localpref value bgpd.peer.X.blackholing.localpref
Announced inbound localpref value bgpd.peer.X.inbound.master_localpref, bgpd.peer.X.inbound.slave_localpref
Autonomous System bgpd.peer.X.as
Autonomous system number ??
Avatar emoji pushd.webhook.avatar_emoji
Avatar icon URL pushd.webhook.avatar_url
BGP MIB (IPv4) peer.X.mon.ipv4.internal.mode
BGP MIB (IPv6) peer.X.mon.ipv6.internal.mode
BGP Router ID bgpd.peer.X.master_router_id, bgpd.peer.X.slave_router_id
BGP community policy.X.community
BGP mode global.nonintrusive_bgp
BGP session monitoring IPv4 address peer.X.mon.ipv4.bgp_peer
BGP session monitoring IPv6 address peer.X.mon.ipv6.bgp_peer
BGP session password bgpd.peer.X.master_password, bgpd.peer.X.slave_password
BGPd monitoring guard time (sec) bgpd.mon.guardtime
BGPd monitoring holdtime (sec) bgpd.mon.holdtime
BGPd monitoring keepalive (sec) bgpd.mon.keepalive
BGPd monitoring long holdtime (sec) bgpd.mon.longholdtime
BW reserved for local instance operation globalgroup.X.outbound.95th.reserve
Base community for inbound improvements peer.X.inbound.community_base
Blackholing IPv4 next hop bgpd.peer.X.blackholing.ipv4.next_hop
Blackholing IPv6 next_hop bgpd.peer.X.blackholing.ipv6.next_hop

Blackholing Routers peer.X.blackholing.bgp_peer
Blackholing community peer.X.blackholing.community
Bot name pushd.webhook.botname
CC allowed for core.commit_control.loss_override
CC probing TTL (sec) core.commit_control.probe_ttl
CC probing queue slots core.commit_control.probing_queue_size
CC provider precedence peer.X.precedence
Cascade policy policy.X.cascade
Cascading policy max AS bgpd.policy.cascade.amount
Centile value peer.X.95th.centile
Circuit issues detection peer.X.circuit.control
Commit Control core.commit_control
Commit Control for inbound core.commit_control.inbound.enabled
Commit Control for provider peer.X.cc_disable
Commit Control moderated inbound improvements core.commit_control.inbound.moderated
Community mark for RD rd.X.community.local
Community marker for local improvements bgpd.rd_local_mark
Control inbound.rule.X.full_control
Country prefix ??
Delete irrelevant CC improvements core.commit_control.del_irrelevant
Delta loss to restore core.circuit.recover_loss_diff
Delta loss to shutdown core.circuit.high_loss_diff
Delta loss to warn core.circuit.warn_loss_diff
Directory Base DN directory.X.base_dn
Directory CA certificate directory.X.tls_cacertfile
Directory TLS use directory.X.tls
Directory User DN directory.X.user_dn
Directory admin role DNs directory.X.admin_role_groups
Directory bind DN directory.X.initial_bind_dn
Directory bind password directory.X.initial_bind_password
Directory certificate verification directory.X.verify_cert
Directory email attribute directory.X.email
Directory full name attribute directory.X.fullname
Directory hostname directory.X.hostname
Directory name directory.X.name
Directory order directory.X.order
Directory port directory.X.port
Directory state directory.X.state
Directory timeout directory.X.timeout
Directory type directory.X.type
Directory user role attribute directory.X.user_role_attr
Directory username attribute directory.X.username
Email server pushd.email.host
Enable policy policy.X.enabled
Explorer algorithms explorer.algorithms
Explorer worker threads explorer.maxthreads
Exploring queue slots core.eventqueuelimit

Failback timer (s) `global.failover_timer_failback`
 Failed probe lifetime (sec) `core.probes.ttl.failed`
 Failover `global.failover`
 Failover master identity file `global.failover_identity_file`
 Failover timer (s) `global.failover_timer_fail`
 Flapping protection `bgpd.mon.internal.flap_guardtime`
 Flow Collector `collector.flow.enabled`
 Flow agents `peer.X.flow_agents`
 Flow sources `collector.flow.sources`
 Flowspec `global.flowspec`
 Flowspec PBR `global.flowspec.pbr`
 Flowspec status `bgpd.peer.X.flowspec`
 From email `pushd.email.from`
 From phone number `pushd.sms.phone_number`
 Frontend access restriction `global.frontend_acl`
 Global Group high load bandwidth limit (%) `globalgroup.X.outbound.rate_high`
 Global Group local member 95th percentile `globalgroup.X.outbound.95th`
 Global Group low load bandwidth limit (%) `globalgroup.X.outbound.rate_low`
 Global Group member list `globalgroup.X.members`
 Global Group name `globalgroup.X.shortname`
 Global commit improvements `core.global.allow_commit`
 Global improvements max worsening (ms) `core.global.worst_ms`
 Global latency/cost improvements `core.global.allow_latency_cost`
 High volume precedence `explorer.high_vol_precedence`
 Holdtime of BGP updates (sec) `bgpd.db.timeout.withdraw`
 ICMP timeout (ms) `explorer.timeout`
 ICMP/UDP ping monitored IPv4 addresses `peer.X.ipv4.mon`
 ICMP/UDP ping monitored IPv6 addresses `peer.X.ipv6.mon`
 IP prefix ??
 IPv4 External monitor `peer.X.mon.ipv4.external.state`
 IPv4 Internal monitor `peer.X.mon.ipv4.internal.state`
 IPv4 Redirect community `peer.X.flowspec.ipv4.redirect_community`
 IPv4 aggregate size `global.agg_ipv4_max`
 IPv4 diagnostic hop `peer.X.ipv4.diag_hop`
 IPv6 External monitor `peer.X.mon.ipv6.external.state`
 IPv6 Internal monitor `peer.X.mon.ipv6.internal.state`
 IPv6 Redirect community `peer.X.flowspec.ipv6.redirect_community`
 IPv6 aggregate size `global.agg_ipv6_max`
 IPv6 diagnostic hop `peer.X.ipv6.diag_hop`
 IPv6 support `global.ipv6_enabled`
 IRP's IPv4 address `bgpd.peer.X.master_our_ip`, `bgpd.peer.X.slave_our_ip`
 IRP's IPv6 address `bgpd.peer.X.master_our_ipv6`, `bgpd.peer.X.slave_our_ipv6`
 IX auto re-configuration `peer.X.auto_config`
 IX auto re-configuration interval (s) `global.exchanges.auto_config_interval`
 Ignore unannounced prefixes `global.ignored.unannounced`
 Ignored ASNs `global.ignored.asn`
 Ignored communities `global.ignored_communities`

Ignored prefixes `global.ignorednets`
Improvement MED value `bgpd.peer.X.med`
Improvement communities `bgpd.peer.X.master_communities`, `bgpd.peer.X.slave_communities`
Improvement localpref `bgpd.peer.X.master_localpref`, `bgpd.peer.X.slave_localpref`
Improvement mode `global.improve_mode`
Inbound overload by (%) `trap.core_cc_overload.inbound_limit_pct`
Inbound Injection method `global.inbound.injection`
Inbound Prefix Next Hop `inbound.rule.X.next_hop`
Inbound bandwidth estimation algorithm `core.commit_control.inbound.volume_estimation`
Inbound improvement delay (sec) `core.commit_control.inbound.improvement.delay`
Inbound overload by (Mbps) `trap.core_cc_overload.inbound_limit_mbps`
Inbound prefix control `bgpd.full_control`
Indirect probing precedence `explorer.probe.indirect_priority`
Infrastructure IPs `explorer.infra_ips`
Internal monitor SNMP host `peer.X.mon.snmp`
Irpspan interfaces `collector.span.interfaces`
Issues time horizon (min) `core.circuit.hist_interval`
Keepalive interval (sec) `bgpd.peer.X.keepalive`
List of providers `policy.X.providers`
Local IPv4 inbound next_hop `bgpd.peer.X.inbound.ipv4.next_hop`
Local IPv6 inbound next_hop `bgpd.peer.X.inbound.ipv6.next_hop`
Local improvement only `policy.X.forcelocal`
Management interface `global.master_management_interface`
Master routing domain `global.master_rd`
Max IPv4 Flowspec rules `core.flowspec.max`
Max IPv4 Improvements `core.improvements.max`
Max IPv6 Flowspec rules `core.flowspec.max_ipv6`
Max IPv6 Improvements `core.improvements.max_ipv6`
Max collected IPs `explorer.max_collector_ips`
Max message size `pushd.sms.message_size`
Maximum load per interface (Mbps) `peer.X.limit_load`
Maximum probe lifetime (sec) `core.probes.ttl.max`
Min loss detection packets `explorer.probing.sendpkts.min`
Mindelay probing queue slots `collector.span.min_delay.probing_queue_size`
Minimal prefix bandwidth (Mbps) `core.commit_control.agg_bw_min`
Minimal probe lifetime (sec) `core.probes.ttl.min`
Minimal traffic volume (%) `collector.export.volume.min_pct`
Minimal traffic volume (bytes) `collector.export.volume.min`
NetFlow UDP port `collector.flow.listen.nf`
Outage confirmation timeout (sec) `core.problem.outage_timeout`
Outage detection `core.outage_detection`
Outage prefix confirmation rate `core.outage_detection.limit_pct`
Outage round trip rate (%) `core.problem.rtt.diff_pct`
Outbound optimization `global.outbound`
Outbound traffic matching `collector.flow.all_outbound`
Overload by (%) `trap.core_cc_overload.limit_pct`
Overload by (Mbps) `trap.core_cc_overload.limit_mbps`

Overusage interval (sec) `core.overusage.check_interval`
 Overusage policies `global.bw_overusage`
 Overusage rule retention (sec) `core.overusage.hold_timer`
 Overusage threshold multiplier `core.overusage.out.threshold.trigger`
 Overusage throttle multiplier `core.overusage.out.threshold.throttle`
 PBR check for provider `peer.X.disable_pbr_confirmation`
 PUSHd TCP listen port `pushd.listen.port`
 Packet size from IP header `collector.span.size_from_ip_header`
 Peering Partner PBR check `provider.X.rule.Y.pbr_check`
 Peering Partner name `provider.X.rule.Y.shortname`
 Peering Partner next-hop `provider.X.rule.Y.next_hop`
 Peering Partner status `provider.X.rule.Y.enabled`
 Performance/Cost improvements within group `peer.X.improve_in_group`
 Plivo API uri `pushd.sms.uri.plivo`
 Policy notes `policy.X.notes`
 Policy priority `policy.X.priority`
 Policy type `policy.X.policy`
 Prefix `inbound.rule.X.prefix`
 Prefix BW average time (hours) `core.overusage.out.average.period`
 Prefix aggregation `global.aggregate`
 Prefix announcement rate (%) `trap.bgpd_announced_rate_low.limit_pct`
 Prefix latency limit (ms) `trap.core_improvement_latency.limit_ms`
 Prefix loss limit (%) `trap.core_improvement_loss.limit_pct`
 Prefix relevant BW (Mbps) `core.overusage.out.average.relevant_min`
 Prefix status `inbound.rule.X.enabled`
 Prepend inbound prefixes `core.circuit.inbound`
 Prepend transit prefixes `core.circuit.transit`
 Probing DSCP `provider.X.rule.Y.probing_dscp`
 Probing IP address `provider.X.rule.Y.probing_ip`
 Probing IPv4 address `peer.X.ipv4.master_probing`, `peer.X.ipv4.slave_probing`
 Probing IPv6 address `peer.X.ipv6.master_probing`
 Probing algorithm `explorer.probe.algorithm`
 Probing interface(s) `global.master_probing_interface`
 Provider 95th Calculation Mode (inbound) `peer.X.95th.mode`
 Provider 95th percentile `peer.X.95th`
 Provider ASN `peer.X.ipv4.next_hop_as`, `peer.X.ipv6.next_hop_as`
 Provider Shutdown `peer.X.shutdown`
 Provider cost per Mbps (USD) `peer.X.cost`
 Provider description `peer.X.description`
 Provider inbound overload by (%) `trap.core_cc_provider_overload.inbound_limit_pct`
 Provider overload by (%) `trap.core_cc_provider_overload.limit_pct`
 Provider overload by (Mbps) `trap.core_cc_provider_overload.limit_mbps`
 Provider overload inbound by (Mbps) `trap.core_cc_provider_overload.inbound_limit_mbps`
 Provider probing IPv6 address `peer.X.ipv6.slave_probing`
 Provider type `peer.X.type`
 Provider's 95th (inbound) `peer.X.95th.in`
 Provider's bandwidth max deviation (%) `core.commit_control.rate.group`

Provider's billing day `peer.X.95th.bill_day`
 Provider's global group ID `peer.X.global_group`
 Provider's high load bandwidth limit (%) `core.commit_control.rate.high`
 Provider's high load inbound bandwidth limit (%) `core.commit_control.inbound.rate.high`
 Provider's low load bandwidth limit (%) `core.commit_control.rate.low`
 Provider's low load inbound bandwidth limit (%) `core.commit_control.inbound.rate.low`
 Provider's routing domain `peer.X.rd`
 Provider/link name `peer.X.shortname`
 Providers `inbound.rule.X.providers`
 Public IPv4 address for PBR tests `explorer.ipv4_test`
 Public IPv6 address for PBR tests `explorer.ipv6_test`, `peer.X.ipv6_pbr_check`
 RD community `rd.X.community_worsening`
 RD shortname `rd.X.shortname`
 RTT between RDs `global.rd_rtt`
 Re-probe on new path change `bgpd.retry_probing.new.bmp_path_change`
 Re-probe on old path change `bgpd.retry_probing.old.bmp_path_change`
 React on Irfpflowd stats `core.commit_control.react_on_collector`
 Relevant RTT difference (%) `core.performance.rtt.diff_pct`
 Relevant RTT difference (ms) `core.performance.rtt.diff_ms`
 Relevant RTT: IX vs transit (%) `core.performance.rtt.ix_diff_pct`
 Relevant RTT: IX vs transit (ms) `core.performance.rtt.ix_diff_ms`
 Relevant loss (%) `core.performance.loss_pct`
 Remove next hop update `bgpd.improvements.remove.next_hop_eq`
 Remove on aggregate withdrawal `bgpd.improvements.remove.withdrawn`
 Restore after (sec) `core.circuit.recover_hold_time`
 Restore interval (min) `core.circuit.recover_monitored_intervals`
 Role `global.failover_role`
 Route Server or Peering session IPv4/IPv6 address `provider.X.rule.Y.bgp_peer`
 Route server's IPv4 addresses `peer.X.ipv4.route_server`
 Route server's IPv6 addresses `peer.X.ipv6.route_server`
 Router `inbound.rule.X.bgp_peer`, `peer.X.bgp_peer`
 Router IPv4 address `bgpd.peer.X.master_peer_ip`, `bgpd.peer.X.slave_peer_ip`
 Router IPv6 address `bgpd.peer.X.master_peer_ipv6`, `bgpd.peer.X.slave_peer_ipv6`
 Router next-hop IPv4 address `peer.X.ipv4.next_hop`
 Router next-hop IPv6 address `peer.X.ipv6.next_hop`
 Routes configuration mode `peer.X.routes_config`
 SMS gateway `pushd.sms.gateway`
 SNMP Traps community `trap.destination.community`
 SNMP Traps destination port `trap.destination.port`
 SNMP Username (traps) `trap.destination.auth_username`
 SNMP authentication (traps) `trap.destination.auth_protocol`
 SNMP authentication password (traps) `trap.destination.auth_password`
 SNMP community `snmp.X.community`
 SNMP encryption `snmp.X.priv_protocol`
 SNMP encryption (traps) `trap.destination.priv_protocol`
 SNMP encryption password `snmp.X.priv_password`
 SNMP encryption password (traps) `trap.destination.priv_password`

SNMP host address `snmp.X.ip`
 SNMP host short name `snmp.X.name`
 SNMP interface counters processing `global.ifstats`
 SNMP interfaces `peer.X.snmp.interfaces`
 SNMP security (traps) `trap.destination.seclvl`
 SNMP v3 Username `snmp.X.auth_username`
 SNMP v3 authentication password `snmp.X.auth_password`
 SNMP v3 authentication protocol `snmp.X.auth_protocol`
 SNMP v3 security level `snmp.X.seclvl`
 SNMP version `snmp.X.version`
 SNMP version (traps) `trap.destination.version`
 SPAN Collector `collector.span.enabled`
 Safe removal of improvements `core.improvements.safe_removal`
 Scanning attempts `explorer.scanning.sendpkts.factor`
 Secret `pushd.sms.auth_token`
 Server port `pushd.email.port`
 Shutdown `bgpd.peer.X.shutdown`
 Slave IPv4 address `global.failover_slave.ip`
 Slave IPv6 address `global.failover_slave.ipv6`
 Slave Management interface `global.slave_management_interface`
 Slave Probing interface(s) `global.slave_probing_interface`
 Slave SSH port `global.failover_slave.port`
 Slave routing domain `global.slave_rd`
 Speaking IP responses for candidates `explorer.scanning.replypkts.min`
 Speaking IPs RTT dispersion (ms) `explorer.scanning.rtt.dispersion_ms`
 Speaking IPs to scan on loss `explorer.scanning.confirm_ips`
 Spike preceding interval (seconds) `trap.core_cc_improvements_spike.period_sec`
 Spike size (%) `trap.core_cc_improvements_spike.diff_pct`
 Split announcements to preserve original route attributes `bgpd.updates.split`
 Standard reprobing period (sec) `core.improvements.ttl.retry_probe`
 Strip non-IRP communities `bgpd.improvements.strip_non_irp_communities`
 TACACS+ directory secret `directory.X.secret`
 Top volume prefixes per cycle `collector.export.volume.high.top_n`
 Top-N relevant volume prefixes `core.improvements.retry_probe.volume_top_n`
 Trace all providers `explorer.trace.all`
 Traceroute algorithms order `explorer.trace.algorithms`
 Traceroute max hops `explorer.traceroute.ttl.max`
 Traceroute min hops `explorer.traceroute.ttl.min`
 Traceroute packets per hop `explorer.traceroute.sendpkts`
 Traceroute retry packets `explorer.traceroute.retrypkts`
 Transit ASNs `bgpd.prefixlist.asn`
 Transit Improvement TTL max (sec) `core.improvements.inbound_transit.ttl.max`
 Transit Improvement TTL min (sec) `core.improvements.inbound_transit.ttl.min`
 Transit Improvements Max `core.improvements.inbound_transit.max`
 Transit SNMP hosts `bgpd.peer.X.transit.snmp`
 Transit matching at egress `collector.flow.process_transit_in_outbound`
 Transit prefixes `bgpd.prefixlist.prefixes`

Transit top N prefixes `collector.flow.export.inbound_transit.topn`
Transiting traffic `bgpd.peer.X.transit.status`, `global.inbound_transit`
Twilio API uri `pushd.sms.uri.twilio`
Use BMP data `peer.X.bmp`
Use BMP to search for routes `peer.X.bmp.check_routes`
Users filter `directory.X.user_filter`
VIP probing policy `X.vip`
VIP reprobing period (sec) `core.vip.interval.probe`
Webhook URL `pushd.webhook.url`
Withdraw improvements on warn `core.circuit.withdraw_on_warn`
load_balance/aggregated BW within peer group `peer.X.group_loadbalance`
min_delay status `collector.span.min_delay`
sFlow UDP port `collector.flow.listen.sf`

5.2 Configuration parameter index

glo

bgpd.peer.X.flowspec, 71, 135, 223, 293

global.agg_ipv4_max, 144, 203, 221, 293

global.agg_ipv6_max, 144, 203, 221, 293

global.aggregate, 51, 144, 203, 221, 295

global.bw_overusage, 203, 295

global.exchanges, 203

global.exchanges.auto_config_interval, 165, 204, 293

global.failover, 145, 204, 205, 209, 211, 212, 224, 274, 275, 293

global.failover.log, 204

global.failover.use_communities, 204

global.failover_identity_file, 78, 204, 293

global.failover_role, 198, 204, 296

global.failover_slave.ip, 145, 198, 205, 297

global.failover_slave.ipv6, 205, 297

global.failover_slave.port, 145, 198, 205, 297

global.failover_timer_fail, 37, 145, 198, 205, 293

global.failover_timer_failback, 37, 145, 198, 205, 293

global.flowspec, 71, 135, 205, 206, 293

global.flowspec.pbr, 11, 58, 71, 206, 293

global.frontend_acl, 206, 293

global.frontend_acl_ips, 206, 291

global.ifstats, 88, 194, 206, 240, 277, 297

global.ignored.asn, 76, 144, 206, 207, 293

global.ignored.unannounced, 207, 293

global.ignored_communities, 76, 144, 206, 207, 293

global.ignorednets, 76, 144, 206, 207, 294

global.improve_mode, 50, 144, 189, 208, 244, 294

global.inbound.injection, 208, 294

global.inbound_conf, 208

global.inbound_transit, 148, 187, 208, 298

global.ipv6_enabled, 209, 293

global.master_management_interface, 144, 189, 209, 211, 294

global.master_probing_interface, 134, 144, 189, 200, 209, 211, 295

global.master_rd, 36, 209, 294

global.nonintrusive_bgp, 50, 71, 144, 210, 291

global.offpeak_hour, 210, 247

global.outbound, 210, 294

global.png.datadir, 210

global.policies, 210

global.rd_rtt, 27, 30, 210, 219, 279, 290, 296

global.rrd.age_max, 211

global.rrd.datadir, 211

global.slave_management_interface, 209, 211, 297

global.slave_probing_interface, 200, 209, 211, 297

global.slave_rd, 36, 212, 297

global.user_directories_conf, 212

COM

db.clickhouse.dbname, 201
db.clickhouse.host, 201
db.clickhouse.password, 201
db.clickhouse.port, 201
db.clickhouse.username, 201
db.dbname, 201
db.host, 202
db.ourhost, 202
db.password, 202
db.port, 202
db.username, 202

bgp

bgpd.as_path, 76, 90, 163, 214, 215, 231, 258, 269,
274, 276, 284, 291
bgpd.as_path_borrowing, 214, 291
bgpd.db.timeout.withdraw, 83, 87, 215, 293
bgpd.full_control, 215, 281, 294
bgpd.improvements.remove.hold_time, 215
bgpd.improvements.remove.next_hop_eq, 147, 215,
216, 296
bgpd.improvements.remove.withdrawn, 147, 215, 216,
296
bgpd.improvements.strip_non_irp_communities, 147,
216, 297
bgpd.log, 216
bgpd.log.level, 216
bgpd.mon.guardtime, 15, 147, 216, 274, 275, 291
bgpd.mon.holdtime, 15, 147, 216, 217, 274, 275, 291
bgpd.mon.internal.flap_guardtime, 147, 217, 293
bgpd.mon.keepalive, 15, 147, 217, 274, 275, 291
bgpd.mon.longholdtime, 15, 217, 291
bgpd.monitor.type, 217
bgpd.policy.cascade.amount, 218, 292
bgpd.prefix.monitor.interval, 218
bgpd.prefix.monitor.search_interval, 218
bgpd.prefixlist.asn, 187, 218, 297
bgpd.prefixlist.prefixes, 187, 219, 297
bgpd.rd_local_mark, 28, 211, 219, 279, 290, 292
bgpd.retry_probing.new.bmp_path_change, 90, 147,
219, 296
bgpd.retry_probing.old.bmp_path_change, 90, 147,
219, 296
bgpd.scaninterval, 215, 220
bgpd.snmp.concurrent_requests, 220
bgpd.snmp.packets_interval, 220
bgpd.snmp.simultaneous, 220
bgpd.transit.monitor.election_interval, 220
bgpd.transit.monitor.fast_reconfirm_interval, 220,
221
bgpd.transit.monitor.retries, 221
bgpd.transit.monitor.timeout, 221
bgpd.updates.split, 147, 203, 221, 230, 247, 297

bgp1

bgpd.peer.X.as, 148, 190, 221, 291
bgpd.peer.X.blackholing.ipv4.next_hop, 44, 222, 291
bgpd.peer.X.blackholing.ipv6.next_hop, 44, 222, 291
bgpd.peer.X.blackholing.localpref, 44, 222, 291
bgpd.peer.X.cap_4byte_as, 222
bgpd.peer.X.inbound.ipv4.next_hop, 148, 223, 281, 294
bgpd.peer.X.inbound.ipv6.next_hop, 148, 223, 281, 294
bgpd.peer.X.inbound.master_localpref, 148, 223, 291
bgpd.peer.X.inbound.slave_localpref, 148, 223, 291
bgpd.peer.X.keepalive, 149, 224, 294
bgpd.peer.X.listen, 224
bgpd.peer.X.master_communities, 149, 191, 199, 204, 224, 227, 294
bgpd.peer.X.master_localpref, 148, 149, 191, 199, 204, 224, 227, 294
bgpd.peer.X.master_our_ip, 148, 149, 190, 199, 225, 227, 293
bgpd.peer.X.master_our_ipv6, 148, 225, 227, 293
bgpd.peer.X.master_password, 149, 199, 225, 228, 291
bgpd.peer.X.master_peer_ip, 148, 190, 225, 229, 296
bgpd.peer.X.master_peer_ipv6, 148, 226, 229, 296
bgpd.peer.X.master_router_id, 226, 229, 291
bgpd.peer.X.med, 149, 191, 226, 294
bgpd.peer.X.origin, 226
bgpd.peer.X.shutdown, 226, 297
bgpd.peer.X.slave_communities, 149, 199, 204, 224, 227, 294
bgpd.peer.X.slave_localpref, 149, 199, 224, 225, 227, 294
bgpd.peer.X.slave_our_ip, 149, 199, 225, 227, 293
bgpd.peer.X.slave_our_ipv6, 225, 227, 293
bgpd.peer.X.slave_password, 149, 199, 225, 228, 291
bgpd.peer.X.slave_peer_ip, 148, 229, 296
bgpd.peer.X.slave_peer_ipv6, 148, 229, 296
bgpd.peer.X.slave_router_id, 149, 226, 229, 291
bgpd.peer.X.transit.mib, 229
bgpd.peer.X.transit.snmp, 148, 230, 297
bgpd.peer.X.transit.status, 230, 298
bgpd.peer.X.updates.limit.max, 203, 221, 230
bgpd.peer.X.updates.limit.ps, 230

col

- collector.detect.explorer_ips, 231
- collector.export.interval, 231, 232
- collector.export.top_volume_ips, 114, 232
- collector.export.ttl, 232
- collector.export.volume.high.top_n, 150, 232, 253, 297
- collector.export.volume.min, 150, 232, 294
- collector.export.volume.min_pct, 150, 232, 294
- collector.flow.all_outbound, 233, 294
- collector.flow.buffer.size, 233
- collector.flow.enabled, 53, 188, 233, 235, 293
- collector.flow.export.inbound_transit_topn, 187, 233, 298
- collector.flow.listen.nf, 53, 150, 188, 233, 294
- collector.flow.listen.sf, 53, 150, 188, 233, 298
- collector.flow.log, 234
- collector.flow.log.level, 234
- collector.flow.process_transit_in_outbound, 187, 234, 297
- collector.flow.sources, 53, 150, 188, 234, 293
- collector.ournets, 53, 57, 151, 188, 233, 234, 281, 291
- collector.sessions.max, 235
- collector.span.buffer.size, 235
- collector.span.enabled, 57, 188, 233–235, 297
- collector.span.interfaces, 56, 57, 150, 188, 235, 294
- collector.span.log, 236
- collector.span.log.level, 236
- collector.span.min_delay, 57, 150, 188, 236, 298
- collector.span.min_delay.probing_queue_size, 150, 188, 236, 294
- collector.span.size_from_ip_header, 57, 236, 295
- collector.span.threshold.blackout, 237
- collector.span.threshold.congestion, 237
- collector.span.threshold.delay, 237
- collector.span.threshold.excessive, 237
- collector.speaking_ips, 237

COR

core.circuit.high_loss_diff, 154, 238–240, 292
core.circuit.hist_interval, 154, 238, 294
core.circuit.inbound, 238, 239, 295
core.circuit.recover_hold_time, 155, 238, 296
core.circuit.recover_loss_diff, 154, 239, 240, 292
core.circuit.recover_monitored_intervals, 155, 239, 296
core.circuit.transit, 238, 239, 295
core.circuit.warn_loss_diff, 154, 239, 292
core.circuit.withdraw_on_warn, 154, 240, 298
core.commit_control, 46, 47, 51, 208, 240, 268, 270, 278, 292
core.commit_control.agg_bw_min, 19, 49, 156, 240, 294
core.commit_control.del_irrelevant, 241, 292
core.commit_control.inbound.enabled, 185, 241, 292
core.commit_control.inbound.improvement.delay, 241, 292
core.commit_control.inbound.moderated, 185, 241, 292
core.commit_control.inbound.rate.high, 242, 296
core.commit_control.inbound.rate.low, 242, 296
core.commit_control.inbound.volume_estimation, 242, 294
core.commit_control.probe_ttl, 240, 242, 292
core.commit_control.probing_queue_size, 156, 243, 292
core.commit_control.rate.group, 240, 243, 295
core.commit_control.rate.high, 240, 242–244, 285, 296
core.commit_control.rate.low, 46, 240, 242–244, 285, 296
core.commit_control.react_on_collector, 47, 244, 296
core.cost.worst_ms, 152, 244, 246, 291
core.eventqueuelimit, 152, 245, 292
core.eventqueuelimit.retry_probe_pct, 19, 245
core.flowspec.max, 135, 245, 294
core.flowspec.max_ipv6, 135, 245, 294
core.global.allow_commit, 245, 293
core.global.allow_latency_cost, 246, 293
core.global.worst_ms, 30, 245, 246, 293
core.improvements.clearslots, 246
core.improvements.clearslots.days_max, 246
core.improvements.inbound_transit.max, 187, 246, 297
core.improvements.inbound_transit.ttl.clean, 247
core.improvements.inbound_transit.ttl.max, 187, 247, 297
core.improvements.inbound_transit.ttl.min, 187, 247, 297
core.improvements.max, 19, 152, 221, 247, 294
core.improvements.max_ipv6, 19, 152, 248, 294
core.improvements.retry_probe.volume_top_n, 19, 152, 248, 297
core.improvements.safe_removal, 248, 297
core.improvements.ttl.retry_probe, 18, 19, 47, 152, 248, 297
core.log, 19, 248
core.log.level, 249
core.outage_detection, 51, 249, 252, 294
core.outage_detection.limit_pct, 249, 294
core.overusage.check_interval, 153, 249, 295
core.overusage.hold_timer, 153, 249, 295
core.overusage.out.average.period, 153, 249, 295
core.overusage.out.average.relevant_min, 153, 250, 295
core.overusage.out.threshold.throttle, 154, 250, 295
core.overusage.out.threshold.trigger, 154, 250, 295
core.performance.loss_pct, 152, 242, 244, 250, 296
core.performance.rtt.diff_ms, 152, 250, 251, 296
core.performance.rtt.diff_pct, 152, 250, 251, 296
core.performance.rtt.ix_diff_ms, 251, 296
core.performance.rtt.ix_diff_pct, 251, 296
core.probes.ttl.failed, 252, 293
core.probes.ttl.max, 152, 252, 294
core.probes.ttl.min, 152, 252, 294
core.problem.outage_timeout, 252, 294
core.problem.rtt.diff_pct, 252, 294
core.vip.interval.probe, 18, 152, 253, 298

exp

- explorer.aipi, 253
- explorer.algorithms, 253, 292
- explorer.high_vol_precedence, 158, 253, 293
- explorer.infra_ips, 70, 158, 188, 253, 294
- explorer.interval.infra, 254
- explorer.interval.other, 254
- explorer.interval.other.trace, 254
- explorer.ipv4_test, 254, 275, 296
- explorer.ipv6_test, 254, 276, 296
- explorer.log, 255
- explorer.log.level, 255
- explorer.max_collector_ips, 158, 255, 294
- explorer.maxthreads, 158, 253, 255, 292
- explorer.probe.algorithm, 158, 255, 257, 295
- explorer.probe.indirect_priority, 256, 294
- explorer.probing.sendpkts.adaptive_max, 158, 256, 291
- explorer.probing.sendpkts.min, 158, 256, 294
- explorer.probing.simultaneous, 256
- explorer.scanning.confirm_ips, 256, 297
- explorer.scanning.replypkts.min, 257, 297
- explorer.scanning.rtt.dispersion_ms, 257, 297
- explorer.scanning.sendpkts.factor, 257, 297
- explorer.timeout, 158, 257, 293
- explorer.timeout.infra, 257
- explorer.trace.algorithms, 158, 257, 297
- explorer.trace.all, 214, 256, 258, 297
- explorer.traceroute.retrypkts, 158, 258, 297
- explorer.traceroute.sendpkts, 158, 258, 297
- explorer.traceroute.simultaneous, 258
- explorer.traceroute.simultaneous.infra, 259
- explorer.traceroute.ttl.max, 158, 259, 297
- explorer.traceroute.ttl.min, 158, 259, 297

adm

dbcron.api.log, 266

dbcron.api.socket, 266

dbcron.log, 266

dbcron.log.level, 267

globalcc.log, 267

globalcc.log.level, 267

irpdetectd.log, 267

irpdetectd.log.level, 267

irpstatd.log, 267

irpstatd.log.level, 267

irpstatd.snmp.enhanced.sec, 268

ups

peer.X.95th, 160, 193, 196, 198, 240, 263, 264, 268, 269, 276, 279, 285, 295

peer.X.95th.bill_day, 160, 268, 296

peer.X.95th.centile, 160, 268, 292

peer.X.95th.in, 160, 269, 295

peer.X.95th.mode, 185, 242, 269, 295

peer.X.aspath_for_ix, 214, 269, 291

peer.X.auto_config, 165, 204, 269, 293

peer.X.bgp_peer, 159, 167, 192, 269, 296

peer.X.blackholing.bgp_peer, 270, 292

peer.X.blackholing.community, 44, 270, 292

peer.X.bmp, 219, 270, 298

peer.X.bmp.check_routes, 18, 90, 270, 298

peer.X.cc_disable, 47, 159, 160, 193, 196, 198, 208, 270, 292

peer.X.circuit.control, 159, 271, 292

peer.X.cost, 159, 160, 193, 271, 278, 295

peer.X.description, 159, 192, 271, 295

peer.X.diag_hop.interval_max, 253, 271

peer.X.diag_hop.interval_min, 253, 272

peer.X.disable_pbr_confirmation, 272, 295

peer.X.flow_agents, 52, 159, 180, 211, 219, 233, 272, 279, 290, 293

peer.X.flowspec.ipv4.redirect_community, 135, 272, 293

peer.X.flowspec.ipv6.redirect_community, 135, 272, 293

peer.X.global_group, 272, 296

peer.X.group_loadbalance, 47, 49, 273, 298

peer.X.improve_in_group, 160, 193, 198, 273, 295

peer.X.inbound.community_base, 273, 291

peer.X.ipv4.diag_hop, 162, 192, 273, 293

peer.X.ipv4.master_probing, 134, 159, 163, 192, 199, 273, 295

peer.X.ipv4.mon, 16, 17, 162, 195, 274, 293

peer.X.ipv4.next_hop, 15, 163, 192, 273, 274, 277, 296

peer.X.ipv4.next_hop_as, 163, 192, 214, 274, 284, 295

peer.X.ipv4.route_server, 274, 296

peer.X.ipv4.slave_probing, 199, 274, 275, 295

peer.X.ipv4_pbr_check, 275, 276, 291

peer.X.ipv6.diag_hop, 162, 193, 275, 293

peer.X.ipv6.master_probing, 134, 159, 163, 193, 275, 276, 295

peer.X.ipv6.mon, 16, 17, 162, 195, 275, 293

peer.X.ipv6.next_hop, 15, 193, 275, 277, 296

peer.X.ipv6.next_hop_as, 193, 214, 276, 284, 295

peer.X.ipv6.route_server, 276, 296

peer.X.ipv6.slave_probing, 276, 295

peer.X.ipv6_pbr_check, 275, 276, 296

peer.X.limit_load, 159, 160, 193, 196, 198, 268, 276, 294

peer.X.mon.ipv4.bgp_peer, 17, 162, 195, 277, 291

peer.X.mon.ipv4.external.state, 16, 161, 195, 277, 293

peer.X.mon.ipv4.internal.mode, 162, 274, 277, 291

peer.X.mon.ipv4.internal.state, 16, 162, 195, 274, 277, 293

peer.X.mon.ipv6.bgp_peer, 17, 162, 277, 291

peer.X.mon.ipv6.external.state, 161, 195, 278, 293

peer.X.mon.ipv6.internal.mode, 162, 209, 276, 278, 291

peer.X.mon.ipv6.internal.state, 162, 276, 278, 293

peer.X.mon.snmp, 17, 162, 167, 277, 278, 294

peer.X.precedence, 45, 47, 160, 193, 198, 240, 278, 292

peer.X.rd, 27, 159, 211, 219, 279, 290, 296

peer.X.routes_config, 279, 296

peer.X.shortname, 159, 192, 197, 279, 296

peer.X.shutdown, 159, 279, 295

peer.X.snmp.enhanced, 280

peer.X.snmp.interfaces, 88, 167, 194, 240, 277, 280, 297

peer.X.type, 273, 280, 295

rou

policy.X.asn/country/prefix, 282
policy.X.cascade, 282, 292
policy.X.community, 282, 291
policy.X.enabled, 282, 292
policy.X.forcelocal, 283, 294
policy.X.notes, 283, 295
policy.X.policy, 283, 295
policy.X.priority, 283, 295
policy.X.providers, 283, 294
policy.X.vip, 283, 298

exc

provider.X.rule.Y.bgp_peer, 274, 276, 283, 296
provider.X.rule.Y.enabled, 284, 295
provider.X.rule.Y.next_hop, 284, 295
provider.X.rule.Y.next_hop_as, 274, 276, 284, 291
provider.X.rule.Y.pbr_check, 284, 295
provider.X.rule.Y.probing_dscp, 284, 295
provider.X.rule.Y.probing_ip, 284, 295
provider.X.rule.Y.shortname, 284, 295

tra

pushd.email.from, 259, 293
pushd.email.host, 259, 292
pushd.email.port, 259, 297
pushd.listen.port, 260, 295
pushd.log, 260
pushd.log.level, 260
pushd.sms.account_sid, 260, 291
pushd.sms.auth_token, 260, 297
pushd.sms.gateway, 260, 296
pushd.sms.message_size, 260, 294
pushd.sms.phone_number, 261, 293
pushd.sms.uri.plivo, 261, 295
pushd.sms.uri.twilio, 261, 298
pushd.templates.datadir, 261
pushd.webhook.avatar_emoji, 261, 262, 291
pushd.webhook.avatar_url, 261, 262, 291
pushd.webhook.botname, 262, 292
pushd.webhook.url, 262, 298

trap.bgpd_announced_rate_low.limit_pct, 262, 295
trap.core_cc_improvements_spike.diff_pct, 262, 263, 297
trap.core_cc_improvements_spike.period_sec, 262, 263, 297
trap.core_cc_overload.inbound_limit_mbps, 263, 294
trap.core_cc_overload.inbound_limit_pct, 263, 294
trap.core_cc_overload.limit_mbps, 263, 294
trap.core_cc_overload.limit_pct, 263, 294
trap.core_cc_provider_overload.inbound_limit_mbps, 264, 295
trap.core_cc_provider_overload.inbound_limit_pct, 264, 295
trap.core_cc_provider_overload.limit_mbps, 264, 295
trap.core_cc_provider_overload.limit_pct, 264, 295
trap.core_improvement_latency.limit_ms, 264, 295
trap.core_improvement_loss.limit_pct, 265, 295
trap.destination.auth_password, 265, 266, 296
trap.destination.auth_protocol, 265, 266, 296
trap.destination.auth_username, 265, 266, 296
trap.destination.community, 265, 296
trap.destination.port, 265, 296
trap.destination.priv_password, 265, 266, 296
trap.destination.priv_protocol, 266, 296
trap.destination.seclvl, 265, 266, 297
trap.destination.version, 265, 266, 297

api

apid.listen.master_ip, 212
apid.listen.port, 212
apid.listen.slave_ip, 212
apid.log, 212
apid.log.level, 213
apid.maxthreads, 213
apid.path.mtr, 213
apid.path.ping, 213
apid.path.ping6, 213
apid.path.traceroute, 213

inb

bgpd.peer.X.inbound.ipv4.next_hop, 148, 223, 281, 294
bgpd.peer.X.inbound.ipv6.next_hop, 148, 223, 281, 294
bgpd.peer.X.inbound.master_localpref, 148, 223, 291
bgpd.peer.X.inbound.slave_localpref, 148, 223, 291

core.commit_control.inbound.enabled, 185, 241, 292
core.commit_control.inbound.improvement.delay, 241, 294
core.commit_control.inbound.moderated, 185, 241, 292
core.commit_control.inbound.rate.high, 242, 296
core.commit_control.inbound.rate.low, 242, 296
core.commit_control.inbound.volume_estimation, 242, 294
core.commit_control.loss_override, 46, 242, 244, 292
core.commit_control.worst_loss, 244, 291

global.inbound.injection, 208, 294
global.inbound_conf, 208

inbound.rule.X.bgp_peer, 181, 280, 296
inbound.rule.X.enabled, 181, 281, 295
inbound.rule.X.full_control, 215, 281, 292
inbound.rule.X.next_hop, 181, 223, 281, 294
inbound.rule.X.prefix, 181, 281, 295
inbound.rule.X.providers, 181, 281, 296

peer.X.95th.in, 160, 269, 295
peer.X.95th.mode, 185, 242, 269, 295
peer.X.aspath_for_ix, 214, 269, 291
peer.X.circuit.control, 159, 271, 292
peer.X.inbound.community_base, 273, 291

use

directory.X.admin_role_groups, 286, 288, 292
directory.X.base_dn, 286, 288, 292
directory.X.email, 286, 292
directory.X.fullname, 286, 292
directory.X.hostname, 286, 292
directory.X.initial_bind_dn, 286, 292
directory.X.initial_bind_password, 286, 292
directory.X.name, 286, 292
directory.X.order, 287, 292
directory.X.port, 287, 292
directory.X.secret, 287, 297
directory.X.state, 287, 292
directory.X.timeout, 287, 292
directory.X.tls, 287, 288, 292
directory.X.tls_cacertfile, 287, 288, 292
directory.X.type, 288, 292
directory.X.user_dn, 286, 288, 292
directory.X.user_filter, 288, 298
directory.X.user_role_attr, 286, 288, 292
directory.X.username, 286, 288, 292
directory.X.verify_cert, 287, 288, 292

snm

snmp.X.auth_password, 161, 168, 288, 290, 297
snmp.X.auth_protocol, 161, 168, 289, 290, 297
snmp.X.auth_username, 161, 168, 288–290, 297
snmp.X.community, 168, 289, 296
snmp.X.ip, 168, 289, 297
snmp.X.name, 168, 278, 289, 297
snmp.X.priv_password, 161, 168, 289, 290, 296
snmp.X.priv_protocol, 161, 168, 289, 290, 296
snmp.X.seclvl, 161, 168, 288–290, 297
snmp.X.version, 161, 167, 288–290, 297

rou1

rd.X.community.local, 219, 290, 292
rd.X.community_worsening, 30, 290, 296
rd.X.shortname, 290, 296

bmp

irpbmpd.log, 230
irpbmpd.log.level, 231
irpbmpd.port, 231
irpbmpd.sources, 231

glo1

globalgroup.X.members, 285, 293
globalgroup.X.outbound.95th, 285, 293
globalgroup.X.outbound.95th.reserve, 285, 291
globalgroup.X.outbound.rate_high, 285, 293
globalgroup.X.outbound.rate_low, 285, 293
globalgroup.X.shortname, 285, 293

List of Figures

1.2.1	IRP Components	8
1.2.2	System Dashboard	16
1.2.3	Error: Internal BGP Monitor not configured	17
1.2.4	Support for Centralized Route Reflectors	23
1.2.5	IRP configuration in a multi-homed network connected to transit providers as well as and Internet Exchange	24
1.2.6	City wide network	25
1.2.7	Regional network	25
1.2.8	Intercontinental network	25
1.2.9	Multiple routing domains	26
1.2.10	Failover high level overview	35
1.2.11	Unbalanced inbound/outbound peering	39
1.2.12	Fluctuating inbound traffic	39
1.3.1	Performance optimization algorithm	44
1.3.2	Cost optimization algorithm	45
1.3.3	More aggressive when actual exceeds schedule	46
1.3.4	Provider load balancing	48
1.3.5	Commit control of aggregated groups	48
1.3.6	Sorted Commit Control improvements	49
2.3.1	Flow export configuration	53
2.3.2	Span port configuration (on the router)	57
2.3.3	Span port configuration (on the switch)	57
2.4.1	PBR configuration: single router and separate probing Vlan	59
2.4.2	PBR configuration: two routers and separate probing Vlan	62
2.4.3	PBR configuration via GRE tunnels	63
2.4.4	PBR configuration: single router and separate probing Vlan	70
3.1.1	Frontend login form	92
3.1.2	Default Dashboard	93
3.2.1	Menu bar	94
3.2.2	Status bar	94
3.2.3	Common sidebar buttons	94
3.2.4	Email schedule configuration	95
3.2.5	Report filters	95
3.2.6	Time period selection	96
3.2.7	Adjusting the number of results per page	96
3.2.8	Dashboard buttons	97
3.2.9	Failover status of master node	97
3.2.10	Failover status of slave node	98
3.2.11	Search box	98
3.2.12	Dynamical global search results	98
3.2.13	Global search results	99
3.2.14	Precedence and improve in a group warning	99
3.2.15	Internal Monitor warning	99
3.2.16	Explorer performance warning	99

3.2.17	License expiration warning	99
3.3.1	Platform status	100
3.3.2	Interfaces status	100
3.3.3	List of improved prefixes	100
3.3.4	AS-Path problems	101
3.4.1	Reports menu	102
3.4.2	Platform overview	102
3.4.3	Platform performance	103
3.4.4	Loss improvements	104
3.4.5	Loss relative rates	104
3.4.6	Loss-improved destinations	105
3.4.7	Latency values (ms)	106
3.4.8	Latency values (%)	106
3.4.9	Latency destinations	106
3.4.10	Probe conversion rates	107
3.4.11	Current improvements	108
3.4.12	Current improvements map view	109
3.4.13	Improvements to Exchanges	109
3.4.14	Historical records	110
3.4.15	Providers overall	110
3.4.16	Provider performance	111
3.4.17	Provider daily performance	111
3.4.18	Cost overview	112
3.4.19	Congestion and outages	113
3.4.20	Top volume prefixes	113
3.4.21	Top volume AS	114
3.4.22	Top problem AS	114
3.4.23	Prefix statistics	115
3.4.24	Prefix statistics map view	115
3.4.25	AS statistics	116
3.4.26	Country statistics	116
3.4.27	Country statistics map view	117
3.4.28	Exchanges statistics	117
3.4.29	Current inbound improvements	118
3.4.30	Inbound Improvements history	118
3.4.31	Probes today	119
3.4.32	Monitors status	119
3.4.33	Monitor history	120
3.5.1	Graphs menu	121
3.5.2	Improvements by type	121
3.5.3	Performance improvements	122
3.5.4	Probed prefixes and Improvements	122
3.5.5	New and retry improvements	123
3.5.6	Improvements by probing source	123
3.5.7	Prefixes rerouted from provider	124
3.5.8	Prefixes rerouted to provider	124
3.5.9	Improvements by provider	125
3.5.10	Performance improvements by provider	126
3.5.11	Commit improvements by provider	126
3.5.12	Total and improved traffic	127
3.5.13	Bandwidth usage by provider	127
3.5.14	Providers bandwidth usage	128
3.5.15	Bandwidth usage and improvements	128
3.5.16	Total bandwidth usage	129
3.5.17	Inbound traffic distribution	129
3.5.18	Provider performance history	130
3.6.1	Routing Policies	131
3.6.2	Routing policy window	132

3.6.3	Static Routes	132
3.6.4	VIP policies	133
3.7.1	Flowspec policies	133
3.7.2	Flowspec policy popup	134
3.8.1	Throttling excessive bandwidth use with Flowspec policies	135
3.9.1	Maintenance windows	136
3.9.2	Maintenance window popup	137
3.9.3	Maintenance window sidebar	137
3.10.1	System events	138
3.11.1	Traceroute	139
3.11.2	Traceroute results	139
3.11.3	Traceroute exploring details results	140
3.11.4	Looking glass	141
3.11.5	Whois	141
3.11.6	Prefix Probing	142
3.11.7	Prefix probing results	142
3.12.1	Configuration editor: Global settings	144
3.12.2	Configuration editor: Ignored prefixes	145
3.12.3	Configuration editor: Global failover	146
3.12.4	Configuration editor: BGP settings	147
3.12.5	Configuration editor: New Router, general settings	148
3.12.6	Configuration editor: New Router, inbound settings	148
3.12.7	Configuration editor: New Router, advanced settings	149
3.12.8	Configuration editor: New Router, advanced settings	150
3.12.9	Configuration editor: Collector settings	151
3.12.10	Configuration editor: Core configuration	152
3.12.11	Configuration editor: Outage detection	153
3.12.12	Configuration editor: Overusage	153
3.12.13	Configuration editor: Circuit issues detection	154
3.12.14	Configuration editor: Commit Control	156
3.12.15	Providers Overall	157
3.12.16	Configuration editor: Explorer settings	158
3.12.17	Configuration editor: Providers configuration	159
3.12.18	Configuration editor: Adding a new provider, General settings	160
3.12.19	Configuration editor: Adding a new provider, Commit Control	161
3.12.20	Configuration editor: Adding a new provider, SNMP settings	161
3.12.21	Configuration editor: Adding a new provider, External Monitor	162
3.12.22	Configuration editor: Adding a new provider, Monitoring	162
3.12.23	Configuration editor: Adding a new provider, Commit Control	163
3.12.24	An Internet Exchange is similar to a provider and it requires configuration of the Router, the Probing IPs and the other usual attributes	164
3.12.25	The list of peering partners for the Exchange shows details about each of them and offers access to IRP Autoconfiguration feature and PBR ruleset generator for the router	165
3.12.26	PBR Generator	166
3.12.27	Warning about new peering partners that have been added to the Exchange	166
3.12.28	Configuration editor: SNMP hosts configuration	167
3.12.29	Configuration editor: Events configuration	168
3.12.30	Configuration editor: Email configuration	169
3.12.31	Configuration editor: SMS configuration	169
3.12.32	Plivo account ID and secret	170
3.12.33	Configuration editor: SNMP Traps configuration	170
3.12.34	Configuration editor: Web Hooks configuration	170
3.12.35	Notifications	171
3.12.36	Subscribe notifications	172
3.12.37	Subscribe notifications	173
3.12.38	Configuration editor: Security	174
3.12.39	Configuration editor: Users and Directories	174
3.12.40	Configuration editor: LDAP User Directory configuration	175

3.12.41	Configuration editor: Users and Directories	175
3.12.42	Configuration editor: Users and Directories	176
3.12.43	Configuration editor: TACACS+ configuration	177
3.12.44	Configuration: Adding a new user account	178
3.12.45	Configuration editor: Frontend	179
3.12.46	Configuration Editor: Manage currencies	179
3.12.47	Main menu: Inbound	180
3.12.48	Configuration editor: Inbound prefixes	182
3.12.49	Configuration editor: Control of individual inbound prefix	183
3.12.50	Configuration editor: Inbound prefixes control	183
3.12.51	Configuration editor: Base communities	184
3.12.52	Configuration editor: Inbound commit control	185
3.12.53	Configuration editor: Inbound BW estimation algorithms	185
3.12.54	Configuration editor: Inbound BW estimation algorithms	186
3.12.55	Configuration editor: Inbound optimization of transiting traffic parameters	186
3.12.56	Configuration editor: Initial Setup	188
3.12.57	Configuration editor: Setup Infrastructure IP addresses and Analyzed networks	188
3.12.58	Configuration editor: Configure Collector	189
3.12.59	Configuration editor: Improvement mode	189
3.12.60	Configuration editor: Management Interface	189
3.12.61	Configuration editor: Probing Interface	190
3.12.62	Configuration editor: Router name and AS	190
3.12.63	Configuration editor: Router IPv4 addresses	191
3.12.64	Configuration editor: Router announcing	191
3.12.65	Configuration editor: Choose a Router	192
3.12.66	Configuration editor: Provider name	192
3.12.67	Configuration editor: Provider IPv4 addresses	193
3.12.68	Configuration editor: Provider IPv6 addresses	193
3.12.69	Configuration editor: Provider Commit Control	194
3.12.70	Configuration editor: Provider Monitoring setup IPv4	194
3.12.71	Configuration editor: Provider Monitoring setup IPv6	194
3.12.72	Configuration editor: External Monitor setup IPv4	195
3.12.73	Configuration editor: External Monitor setup IPv6	195
3.12.74	Configuration editor: Internal Monitor setup	196
3.12.75	Configuration editor: Provider pre-checks	196
3.12.76	Configuration editor: Commit Control basic setup	197
3.12.77	Configuration editor: Commit Control provider precedence setup	197
3.12.78	Configuration editor: Commit Control groups setup	197
3.12.79	Configuration editor: Commit Control Provider overall	198
3.12.80	Configuration editor: Failover setup	198
3.12.81	Configuration editor: Failover probing IPs for providers	199
3.12.82	Configuration editor: Failover BGP session settings	199
3.12.83	Configuration editor: BGP announcement attributes	199
3.12.84	Configuration editor: Probing and management and interfaces	200

List of configuration examples

1.1	Stop IRP Core and purge existing improvements	12
1.2	Switch to Intrusive Mode and adjust IRP Core and Bgpd parameters	12
1.3	Inserting test improvements	12

1.4	Remove next-hop-self route-map (RM-NHS) example	13
1.5	Restart IRP Bgpd	13
1.6	Show BGP information for specified IP address or prefix	13
1.7	Traceroute destination networks	14
1.8	Delete test improvements	14
1.9	Configure the maximum improvements limit and revert Bgpd configuration	14
1.10	Restart IRP Core and Bgpd	14
1.11	Delete test improvements	14
1.12	Switch the system to non-intrusive mode	14
1.13	Restart IRP Core and Bgpd	14
2.1	Global MLS settings configuration	53
2.2	Global NetFlow settings and export configuration	54
2.3	Per-interface NetFlow settings configuration	54
2.4	Flexible NetFlow monitor configuration	54
2.5	Flexible NetFlow exporter configuration	54
2.6	Flexible NetFlow sampler configuration	54
2.7	Per-interface Flexible NetFlow settings configuration	54
2.8	NetFlow configuration on Cisco 7200/3600 series routers	55
2.9	Ingress/egress flow export configuration on peering interfaces	55
2.10	Ingress flow export configuration	55
2.11	Configuring NetFlow export on Vyatta	55
2.12	Configuration of an interface for the flow accounting	56
2.13	Juniper flow export configuration	56
2.14	Per-interface NetFlow sampling configuration	56
2.15	Cisco IPv4 PBR configuration: 1 router, 2 providers	59
2.16	Cisco IPv4 PBR configuration for IOS XR: 1 router, 2 providers	59
2.17	Juniper IPv4 PBR configuration: 1 router, 2 providers	60
2.18	Vyatta IPv4 PBR configuration example	61
2.19	Vyatta (VC6.5 and up) IPv4 PBR configuration example	61
2.20	IRP server source-based routing using the 'ip' command	62
2.21	IRP server source-based routing using standard CentOS configuration files	62
2.22	Cisco IPv4 PBR configuration: 2 routers, 2 providers, separate Vlan	63
2.23	IRP-side IPv4 GRE tunnel CLI configuration	63
2.24	IRP-side IPv4 GRE tunnel configuration using standard CentOS configs	64
2.25	Router-side IPv4 GRE tunnel configuration (Vyatta equipment)	64
2.26	Router-side IPv4 GRE tunnel configuration (Cisco equipment)	64
2.27	Router-side IPv4 GRE tunnel configuration (Juniper equipment)	64
2.28	IRP server IPv4 source-based routing via GRE using the 'ip' command	64
2.29	IRP server IPv4 source-based routing via GRE using standard CentOS configuration files	65
2.30	Cisco IPv4 PBR configuration: 2 routers, 2 providers, separate Vlan	65
2.31	Cisco IPv4 PBR configuration template	65
2.32	Cisco IPv4 PBR configuration template	66
2.33	Juniper IPv4 PBR configuration template	67
2.34	PBR validation using 'traceroute'	69
2.35	PBR validation using Explorer self check	69
2.36	PBR validation using 'iptables' and 'traceroute'	69
2.37	Sample provider configuration	70
2.38	SNMP parameters validation	71
2.39	iBGP session configuration example	73
2.40	Vyatta IPv4 iBGP session configuration example	73
2.41	Vyatta IPv6 iBGP session configuration example	73
2.42	Vyatta IPv4 route-map for setting local-pref on the router	73
2.43	Vyatta IPv6 route-map for setting local-pref on the router	73
2.44	Cisco IPv4 iBGP session configuration example	73
2.45	Cisco IPv6 iBGP session configuration example	74
2.46	Cisco IPv4 route-map for setting local-pref on the router	74
2.47	Cisco IPv6 route-map for setting local-pref on the router	74
2.48	Limiting the number of received prefixes for an IPv4 neighbor on Cisco	74

2.49	Limiting the number of received prefixes for an IPv6 neighbor on Cisco	74
2.50	Juniper IPv4 iBGP session configuration example	74
2.51	Juniper IPv6 iBGP session configuration example	75
2.52	Juniper IPv4 route-map for setting local-pref on the router	75
2.53	Limiting the number of received prefixes for an IPv4 neighbor on Juniper	75
2.54	Reload IRP Bgpd configuration	76
2.55	Cisco: Configure BGP Additonal Paths	77
2.56	Juniper: Configure BGP Additonal Paths	77
2.57	Generate keys on \$IRPMaster	78
2.58	Install public key on \$IRPSlave	78
2.59	Check SSH certificate-based authentication works	78
2.60	Generate CA and certificates	79
2.61	Verify certificates	79
2.62	Install certificates in designated directories	79
2.63	Cross copy client key and certificates	80
2.64	Set file permissions for keys and certificates	80
2.65	Example \$IRPSlave configuration from template	80
2.66	Check MySQL on \$IRPSlave works correctly	80
2.67	Set \$IRPMaster as a first node for Multi-Master replication	81
2.68	Check MySQL on \$IRPMaster works correctly	81
2.69	Replication user and grants	81
2.70	Copy database root user configuration file	82
2.71	Copy database configuration files	82
2.72	Copy database data files	82
2.73	Copy differences of database files (OS with Systemd)	82
2.74	Copy differences of database files (OS with RC-files)	83
2.75	Set \$IRPMaster as replication slave	83
2.76	Starting replication slave on \$IRPMaster	84
2.77	Set \$IRPSlave as replication slave	84
2.78	Starting replication slave	85
2.79	Starting IRP services and Frontend (OS with Systemd)	85
2.80	Starting IRP services and Frontend (OS with RC-files)	85
2.81	Synchronize RRD	86
4.1	Error log	223