xBGP: When You can’t wait for IETF and Vendors

Thomas Wirtgen, Tom Rousseaux, Quentin de Coninck, Randy Bush, Laurent Vanbever, Axel Legay, Olivier Bonaventure

UCLouvain icteam

ETH zürich

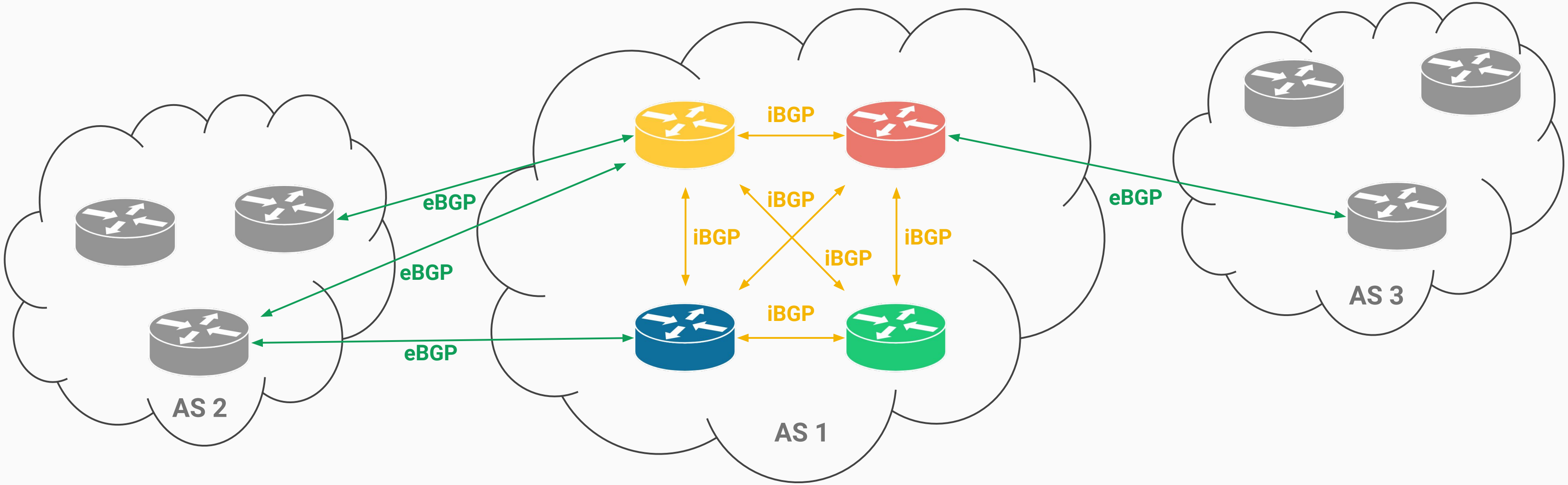
Networked Systems
ETH Zürich — seit 2015

IIJ
Internet Initiative Japan

Agenda

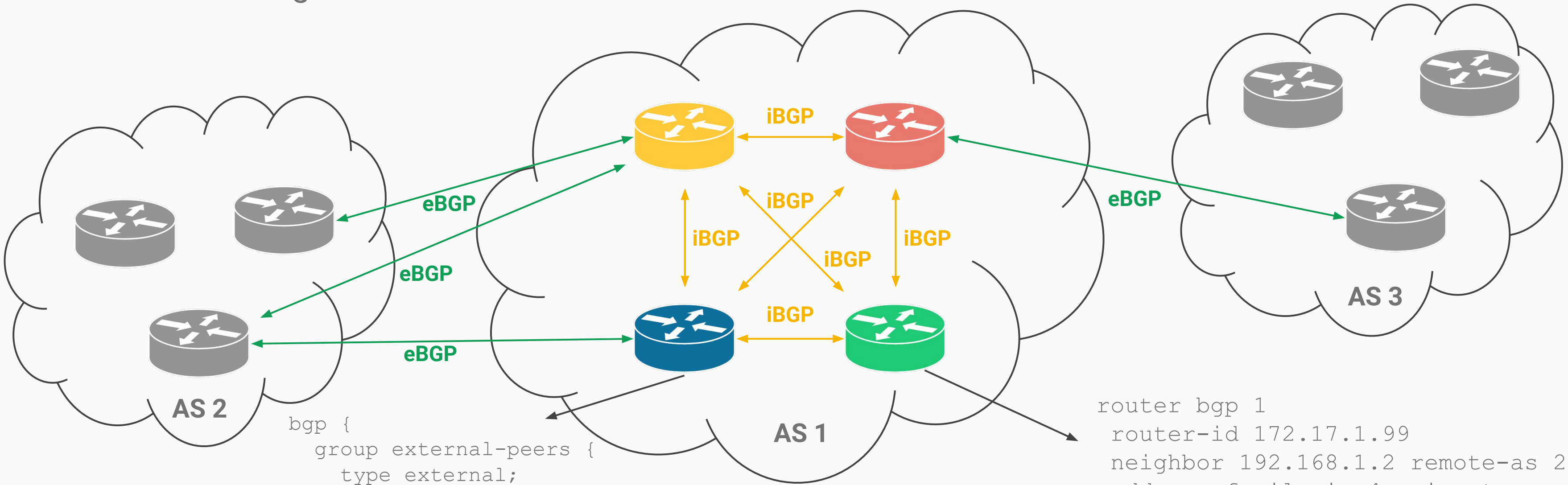
- **Why bring programmability to BGP ?**
- Inside xBGP
- Use Cases
- Verifying xBGP extensions
- Conclusion

Routing on the Internet



Routing on the Internet

Router vendors do not have an unified interface to configure routers



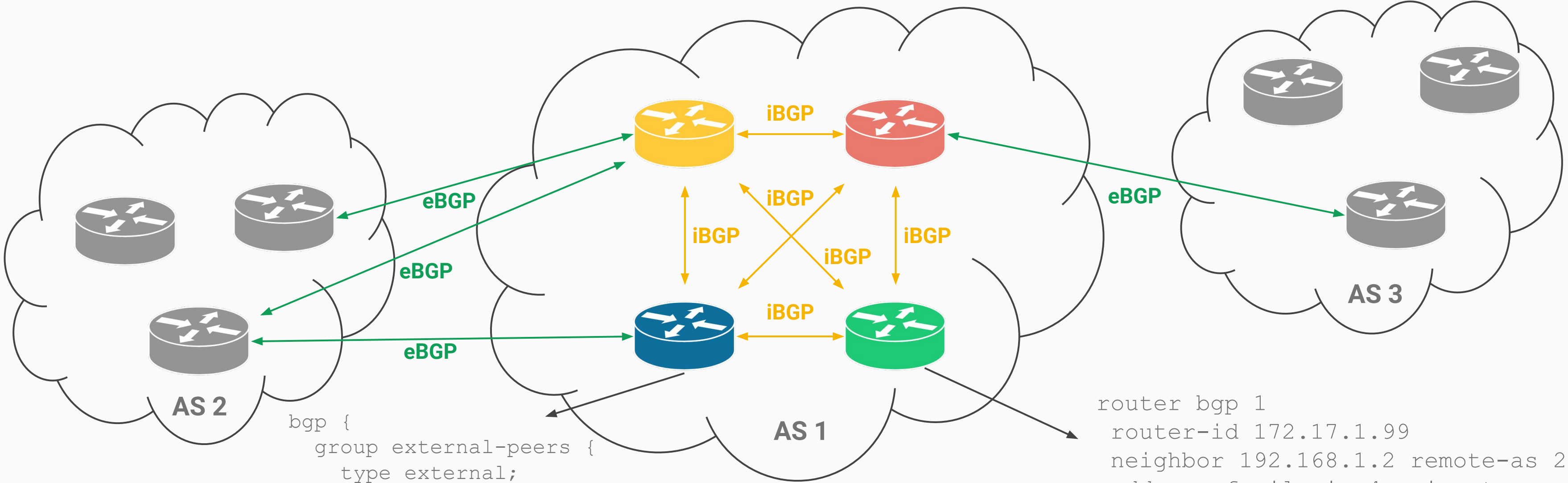
```
bgp {
  group external-peers {
    type external;
    neighbor 172.17.1.99 {
      peer-as 2;
    }}
  routing-options {
    router-id 172.17.1.99;
    autonomous-system 1;
  }}
```

```
router bgp 1
 router-id 172.17.1.99
 neighbor 192.168.1.2 remote-as 2
 address-family ipv4 unicast
  neighbor 192.168.1.2 activate
  exit-address-family
```

Routing on the Internet

Router vendors do not have an unified interface to configure routers

All routers do not implement the same set of functionalities



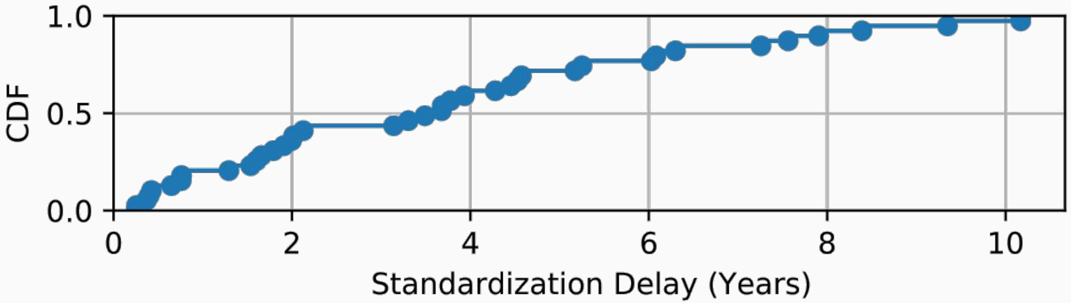
```
bgp {
  group external-peers {
    type external;
    neighbor 172.17.1.99 {
      peer-as 2;
    }}
  routing-options {
    router-id 172.17.1.99;
    autonomous-system 1;
  }}
```

```
router bgp 1
 router-id 172.17.1.99
 neighbor 192.168.1.2 remote-as 2
 address-family ipv4 unicast
  neighbor 192.168.1.2 activate
  exit-address-family
```

As networks evolve, so do routing protocols

One does not simply ask to your routers vendor...

1. Standardisation of the new feature by the IETF (3.5 year in average for BGP)
2. Implementation on the vendor OSes
3. Update your routers

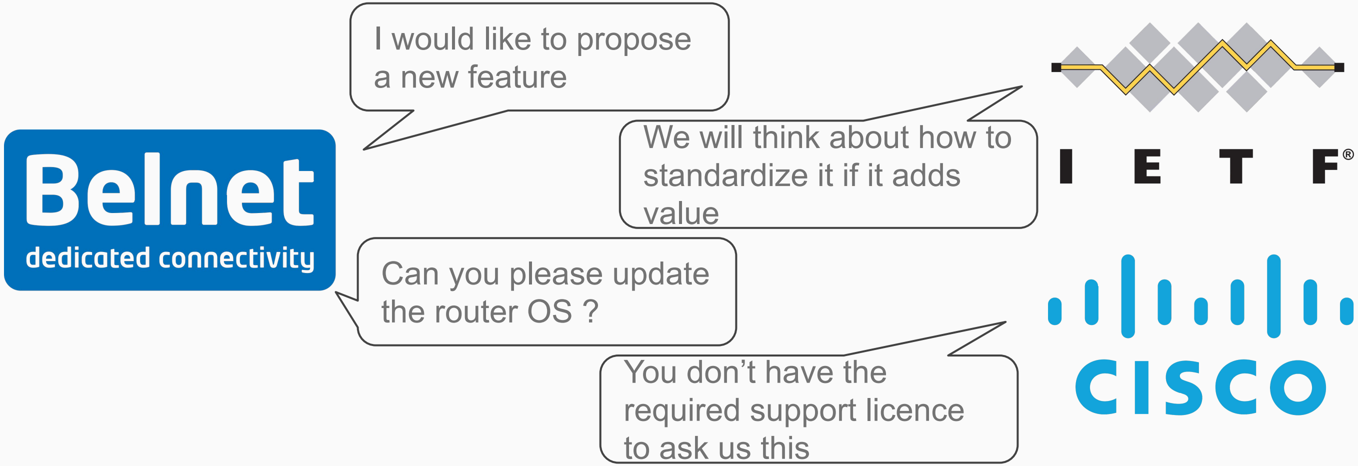


As networks evolve, so do routing protocols

One does not simply ask to your routers vendor...

1. Standardisation of the new feature by the IETF (3.5 year in average for BGP)
2. Implementation on the vendor OSes
3. Update your routers

You can not easily influence steps 1 and 2 !



I would like to propose a new feature

We will think about how to standardize it if it adds value

Can you please update the router OS ?

You don’t have the required support licence to ask us this

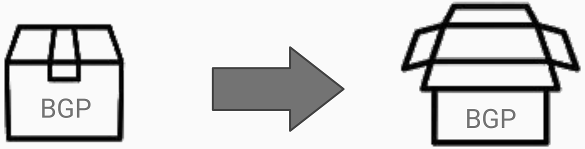
Current paradigm slows innovation

Problem #1: No consensus to propose a unified configuration model

Problem #2: Protocol extensions not implemented on all routers

Problem #3: Slow upgrade process

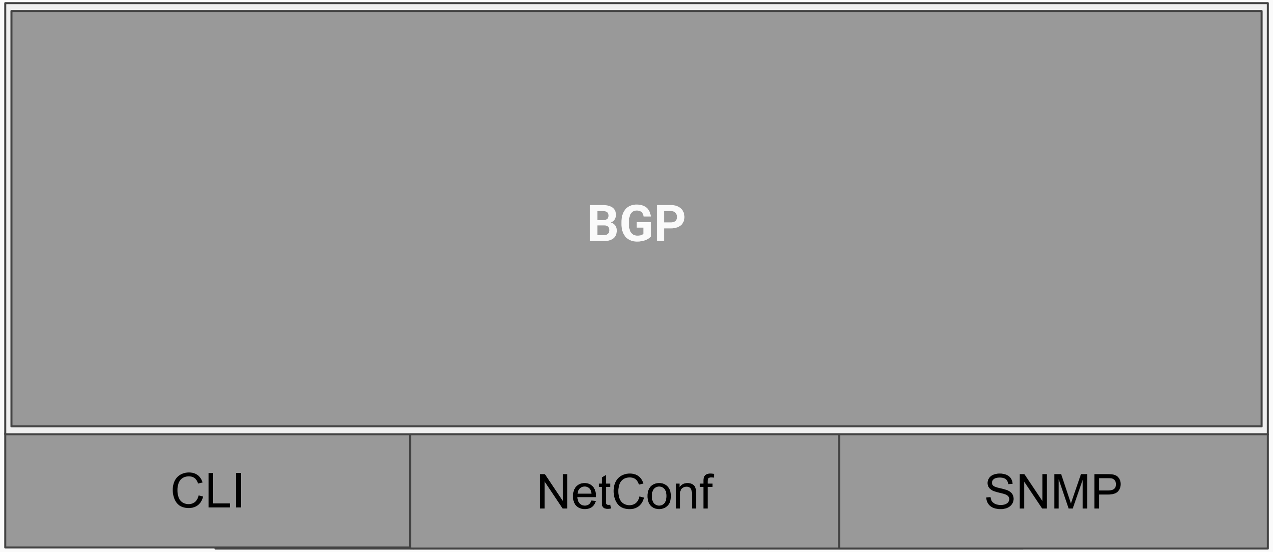
⇒ xBGP is designed to bring innovation to network engineers.

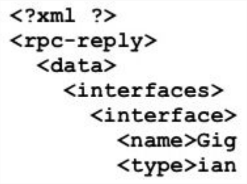


Agenda

- Why bring programmability to BGP ?
- **Inside xBGP**
- Use Cases
- Verifying xBGP extensions
- Conclusion

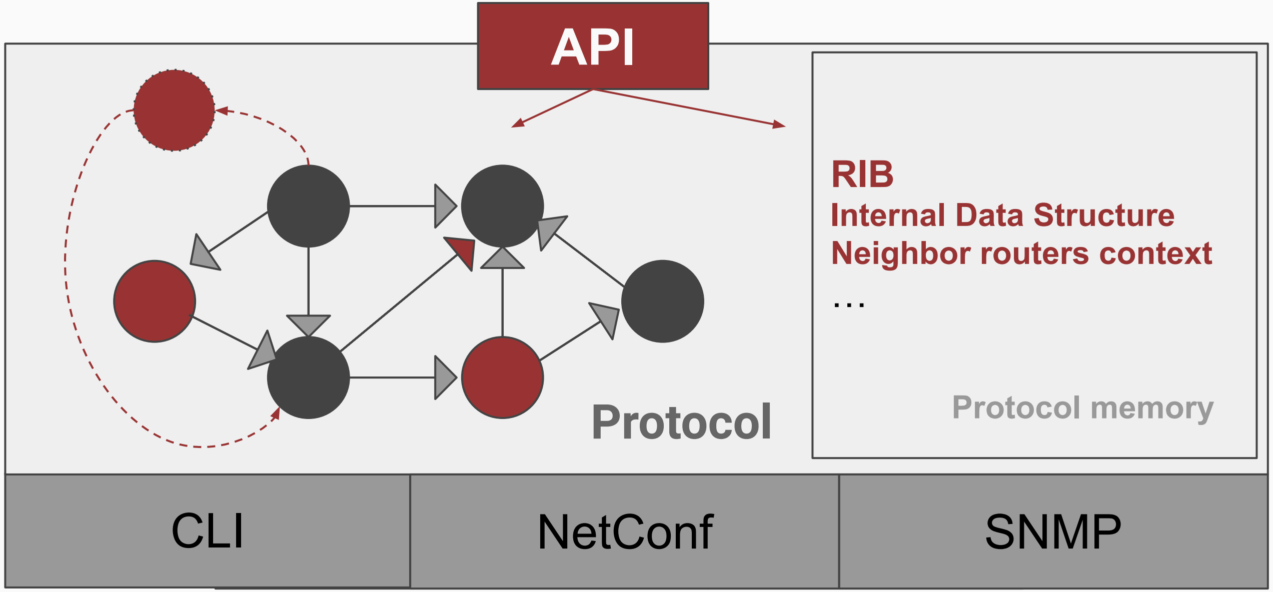
BGP implementations are opaque

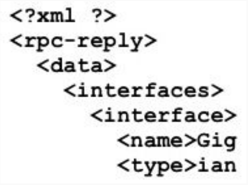




```
<?xml ?>
<rpc-reply>
  <data>
    <interfaces>
      <interface>
        <name>Gig
        <type>ian
```

BGP implementations are opaque

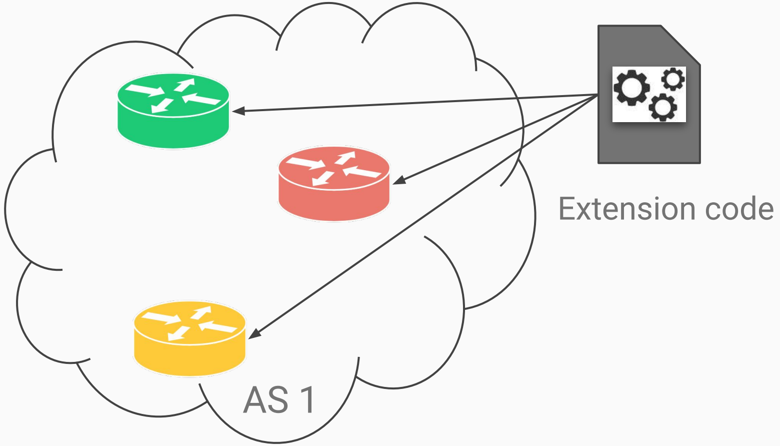




```
<?xml ?>
<rpc-reply>
  <data>
    <interfaces>
      <interface>
        <name>Gig
        <type>ian
```

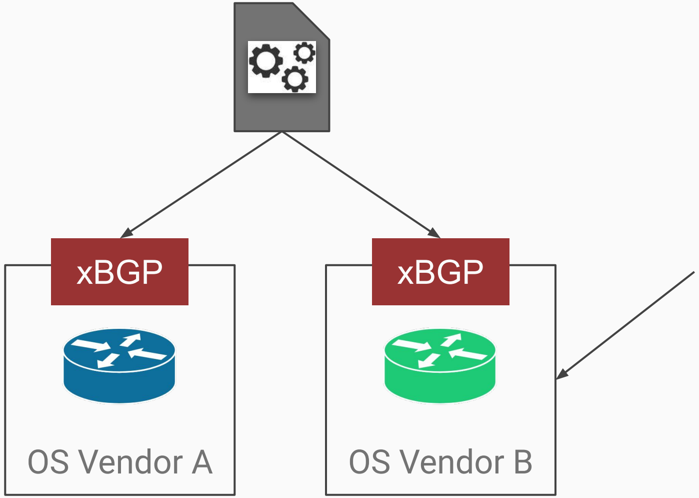
xBGP a paradigm shift

Operators can now add extension codes to their routers



xBGP propose a common interface for routers

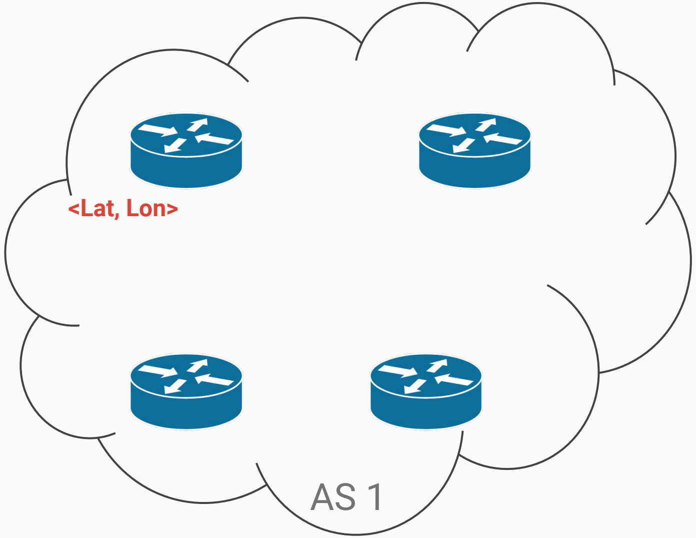
Thanks to xBGP, the same extension code can run on several implementations



Routers are now seen as microkernels

How to alter BGP to make it xBGP compliant ?

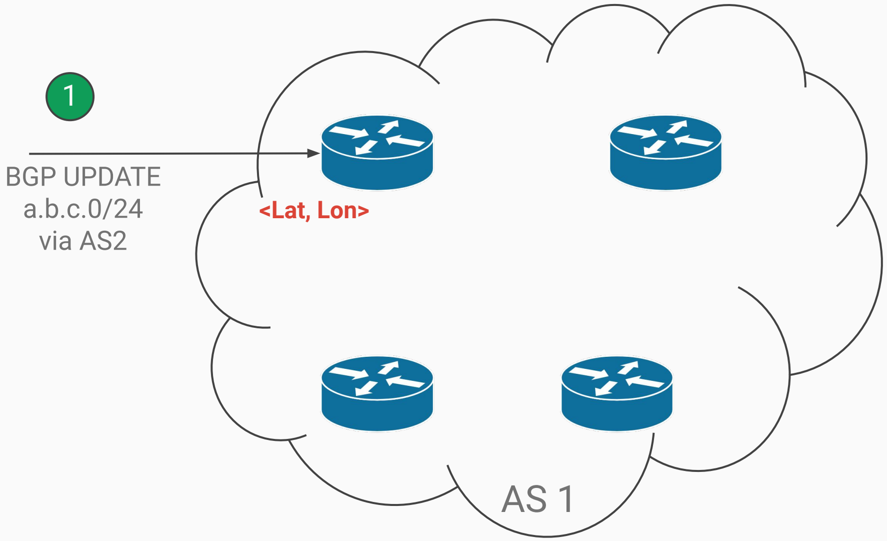
Let’s take an example of feature. The GeoLoc TLV



BGP Path Record Attribute: draft-raszuk-idr-bgp-pr-05

How to alter BGP to make it xBGP compliant ?

Let’s take an example of feature. The GeoLoc TLV

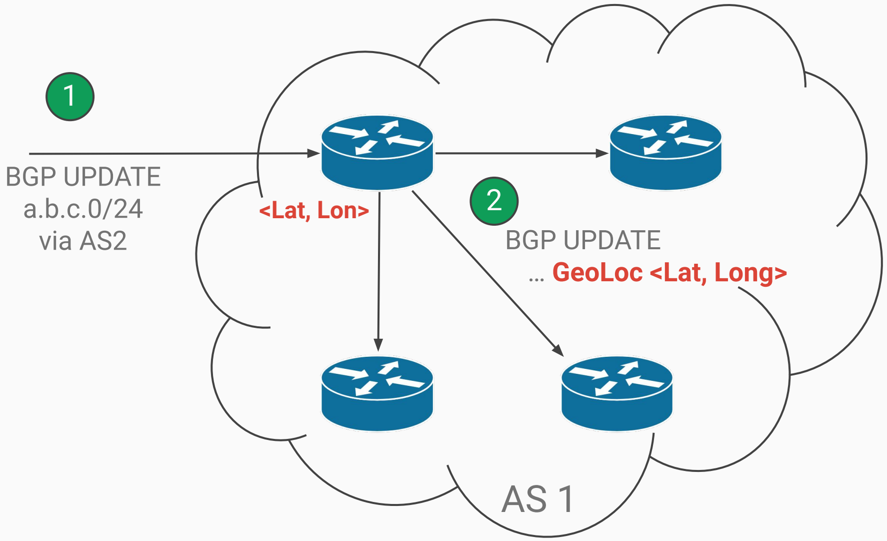


1. Add GeoLoc on the input edge routers

BGP Path Record Attribute: draft-raszuk-idr-bgp-pr-05

How to alter BGP to make it xBGP compliant ?

Let’s take an example of feature. The GeoLoc TLV

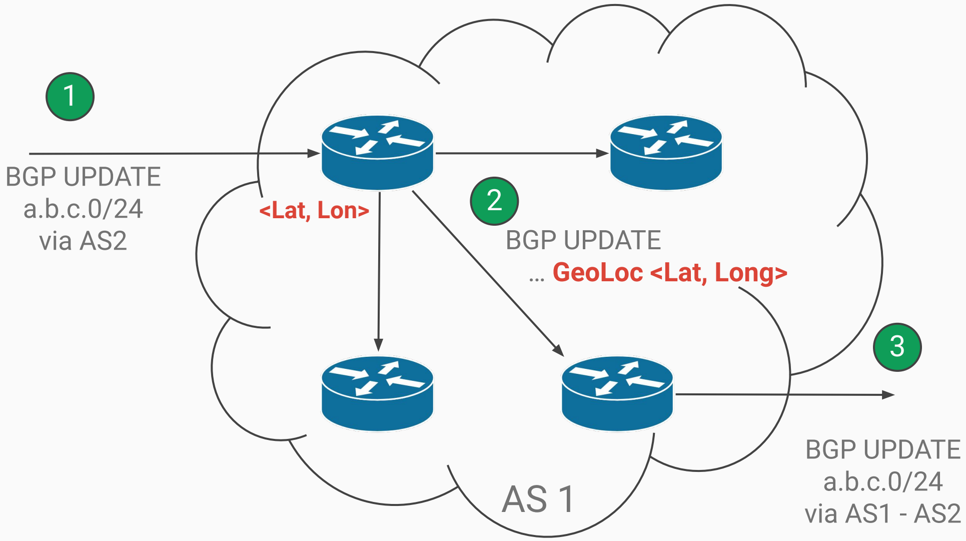


1. Add GeoLoc on the input edge routers
2. Spread the GeoLoc inside the IGP

BGP Path Record Attribute: draft-raszuk-idr-bgp-pr-05

How to alter BGP to make it xBGP compliant ?

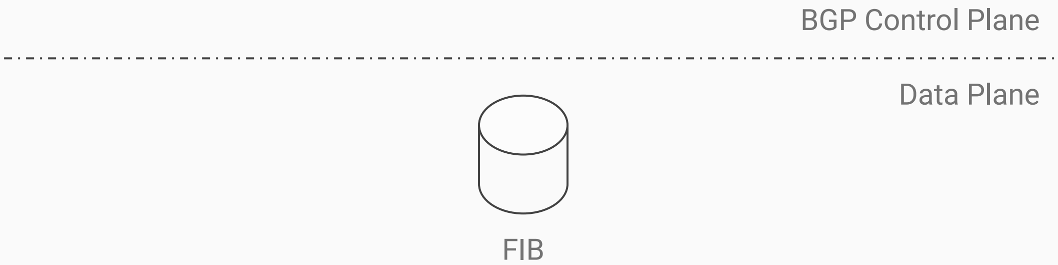
Let’s take an example of feature. The GeoLoc TLV



1. Add GeoLoc on the input edge routers
2. Spread the GeoLoc inside the IGP
3. Remove GeoLoc on the output edge routers

BGP Path Record Attribute: draft-raszuk-idr-bgp-pr-05

How to alter BGP to make it xBGP compliant ?



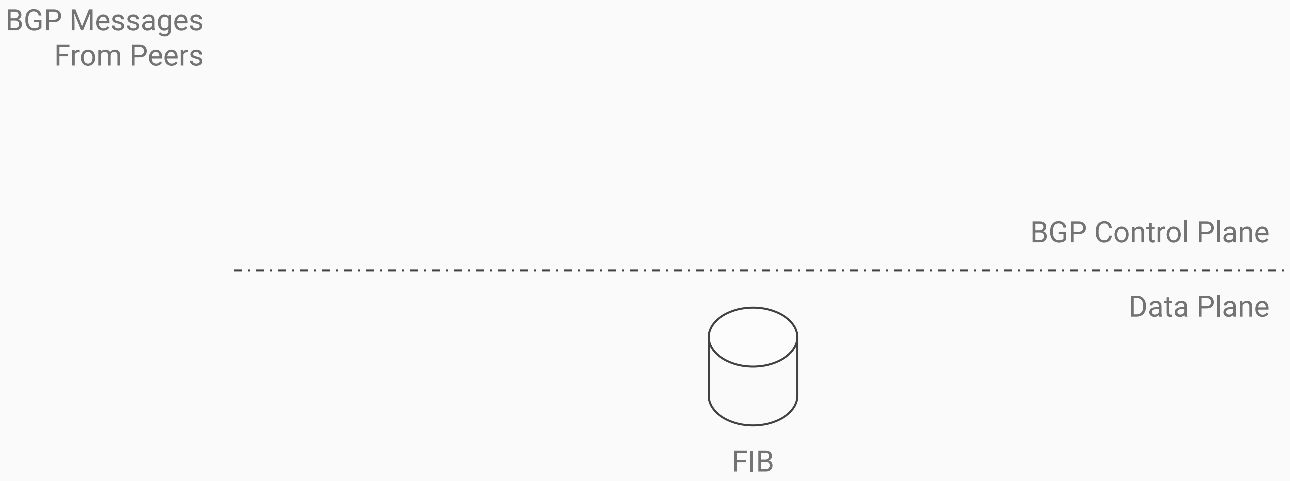
BGP Control Plane

Data Plane

FIB

RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



BGP Messages
From Peers

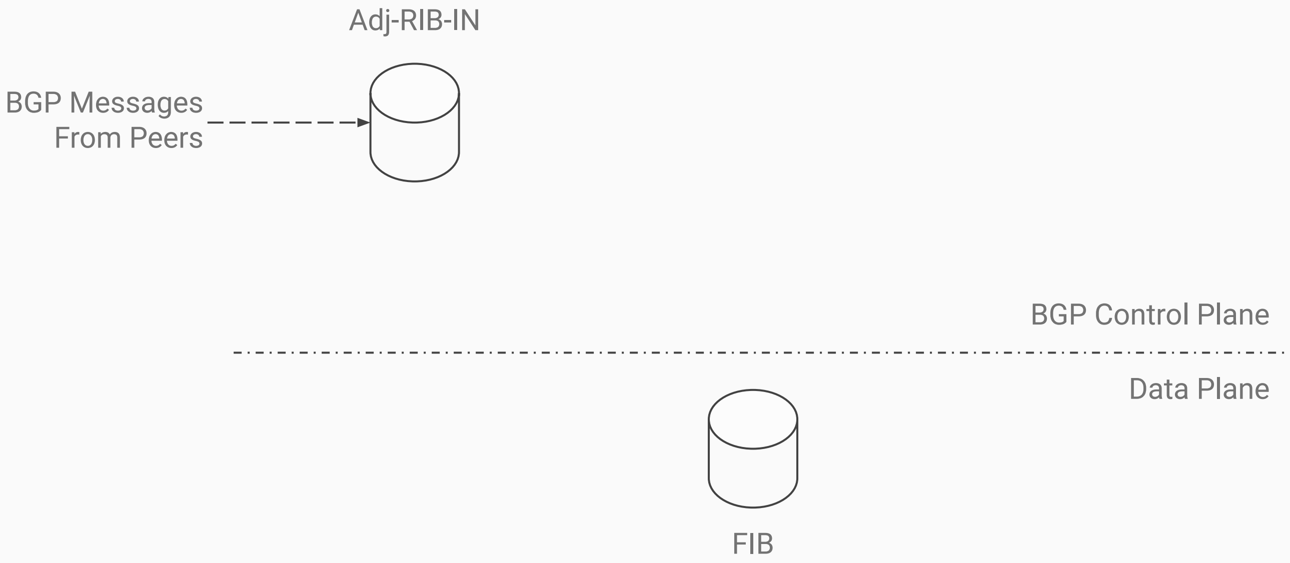
BGP Control Plane

Data Plane

FIB

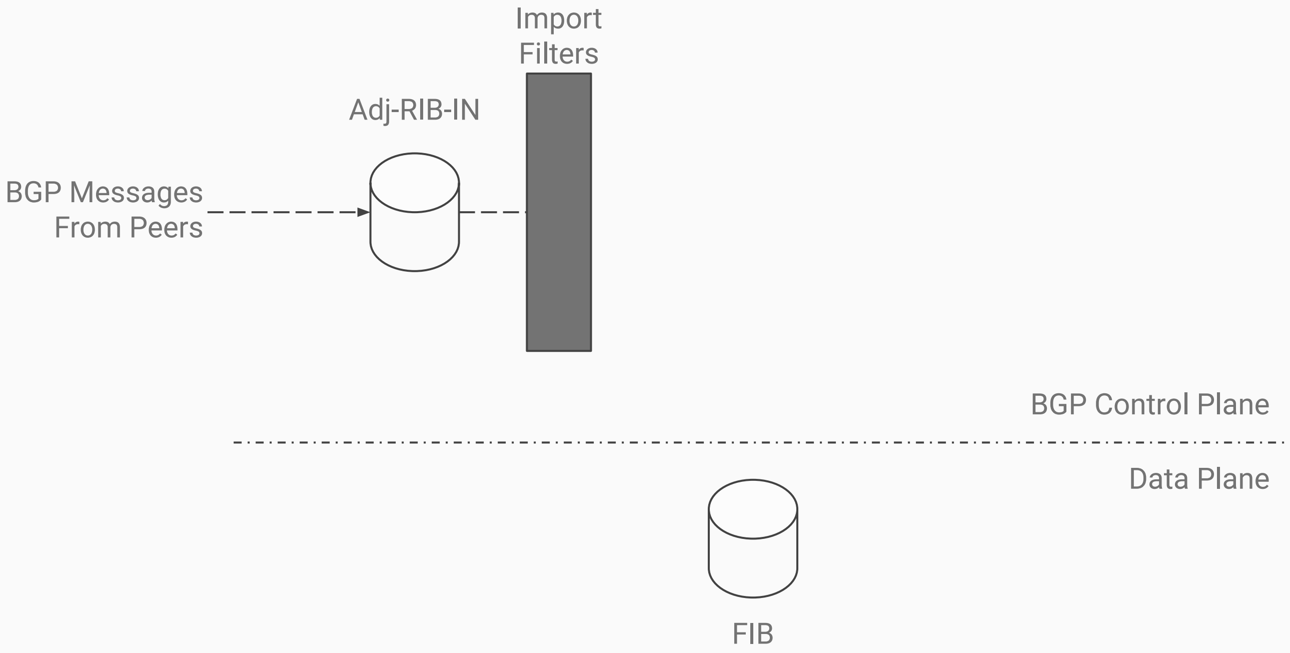
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



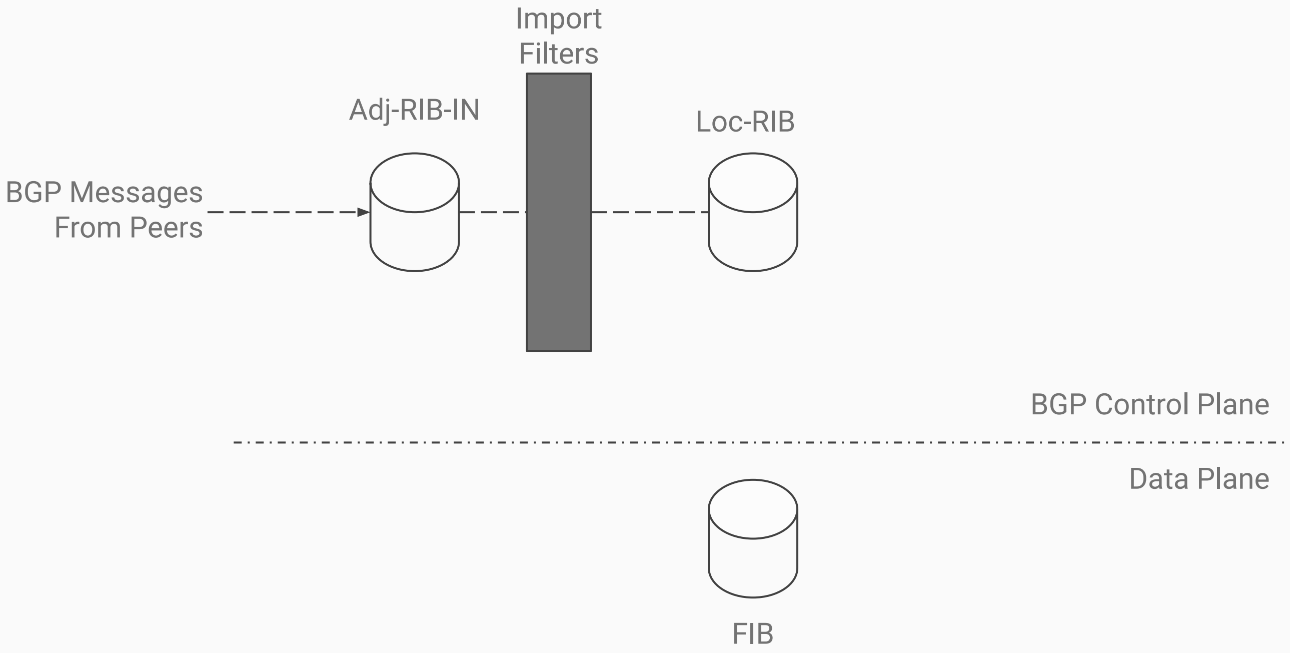
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



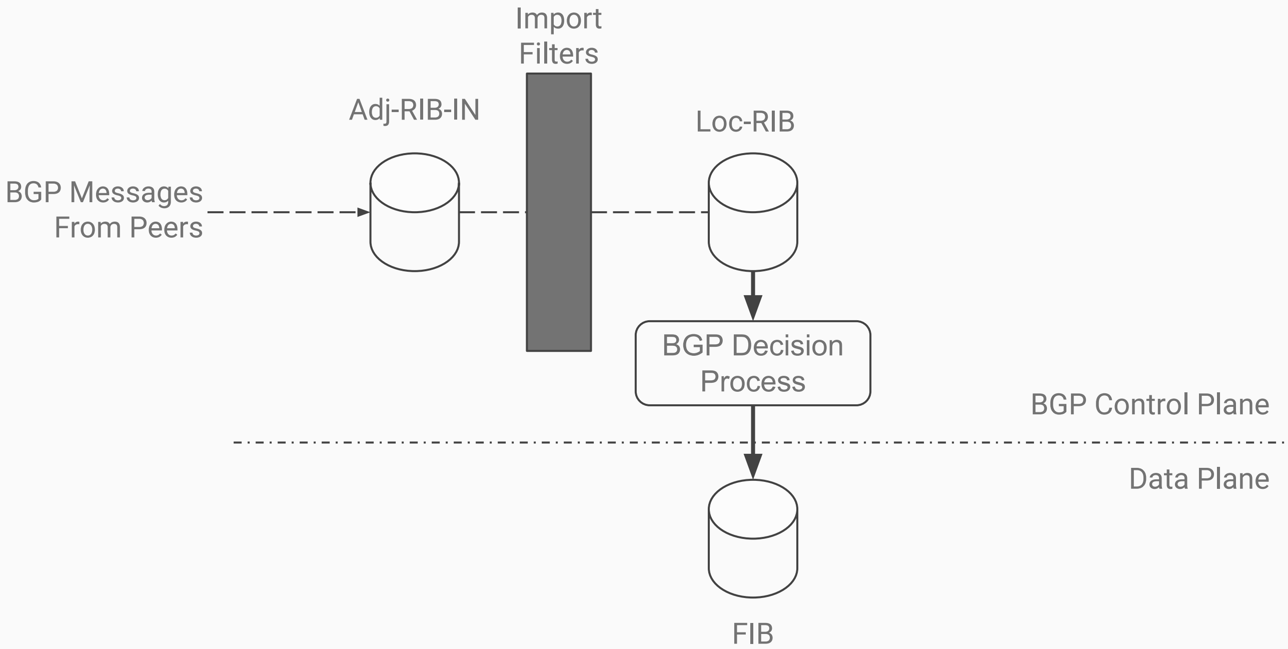
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



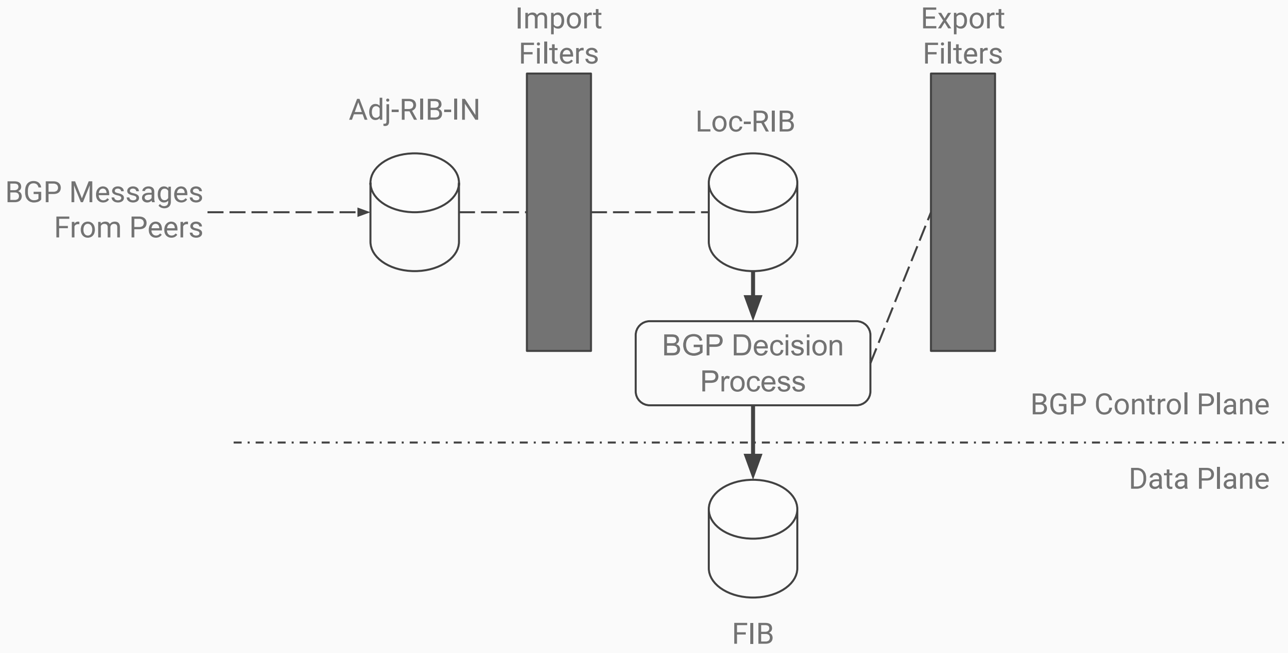
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



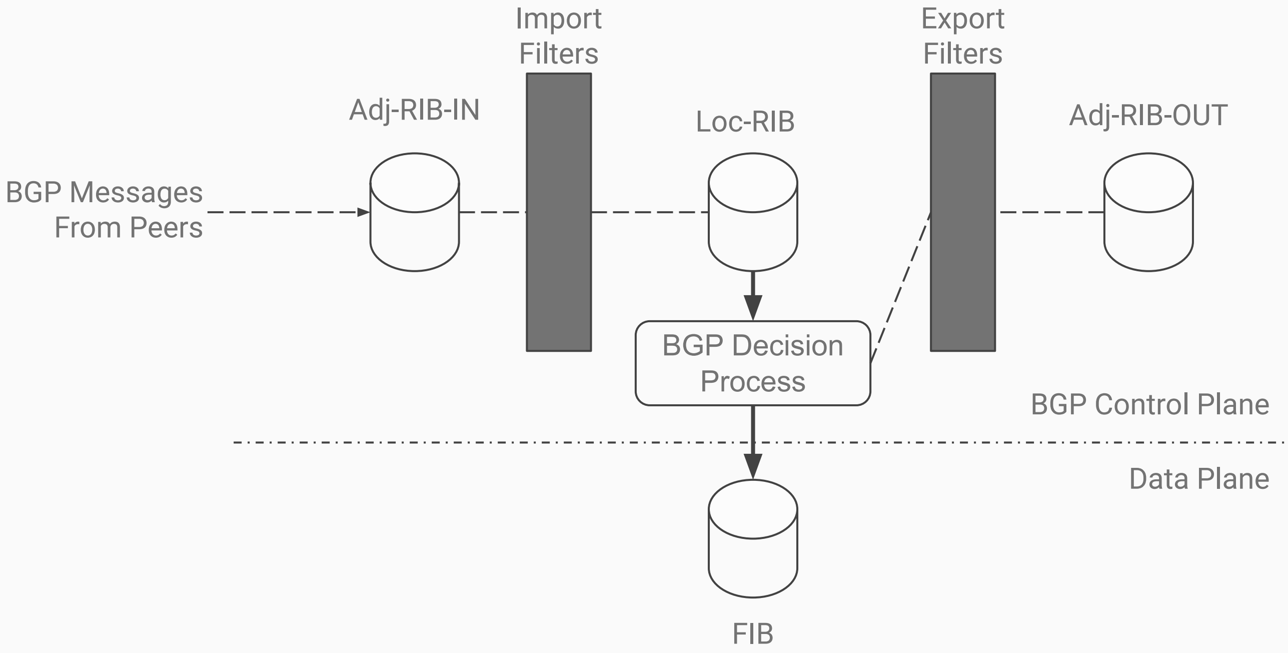
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



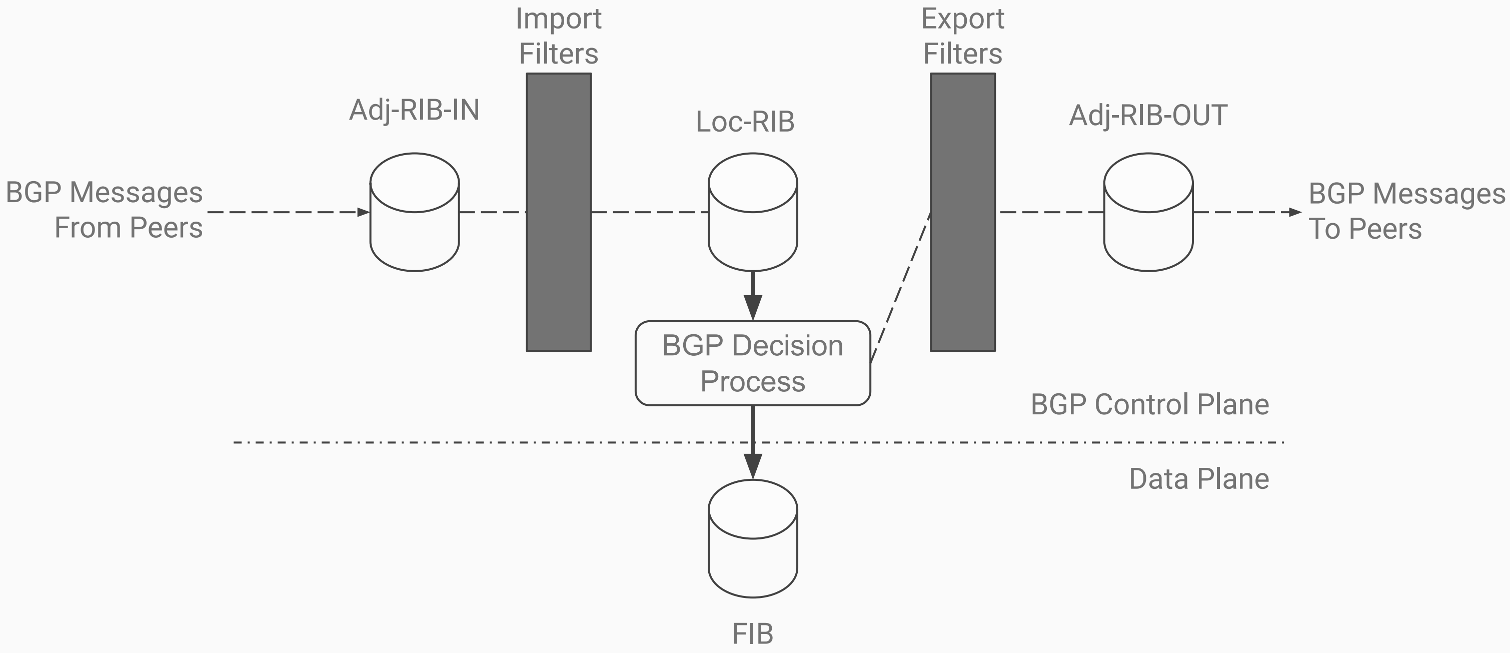
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



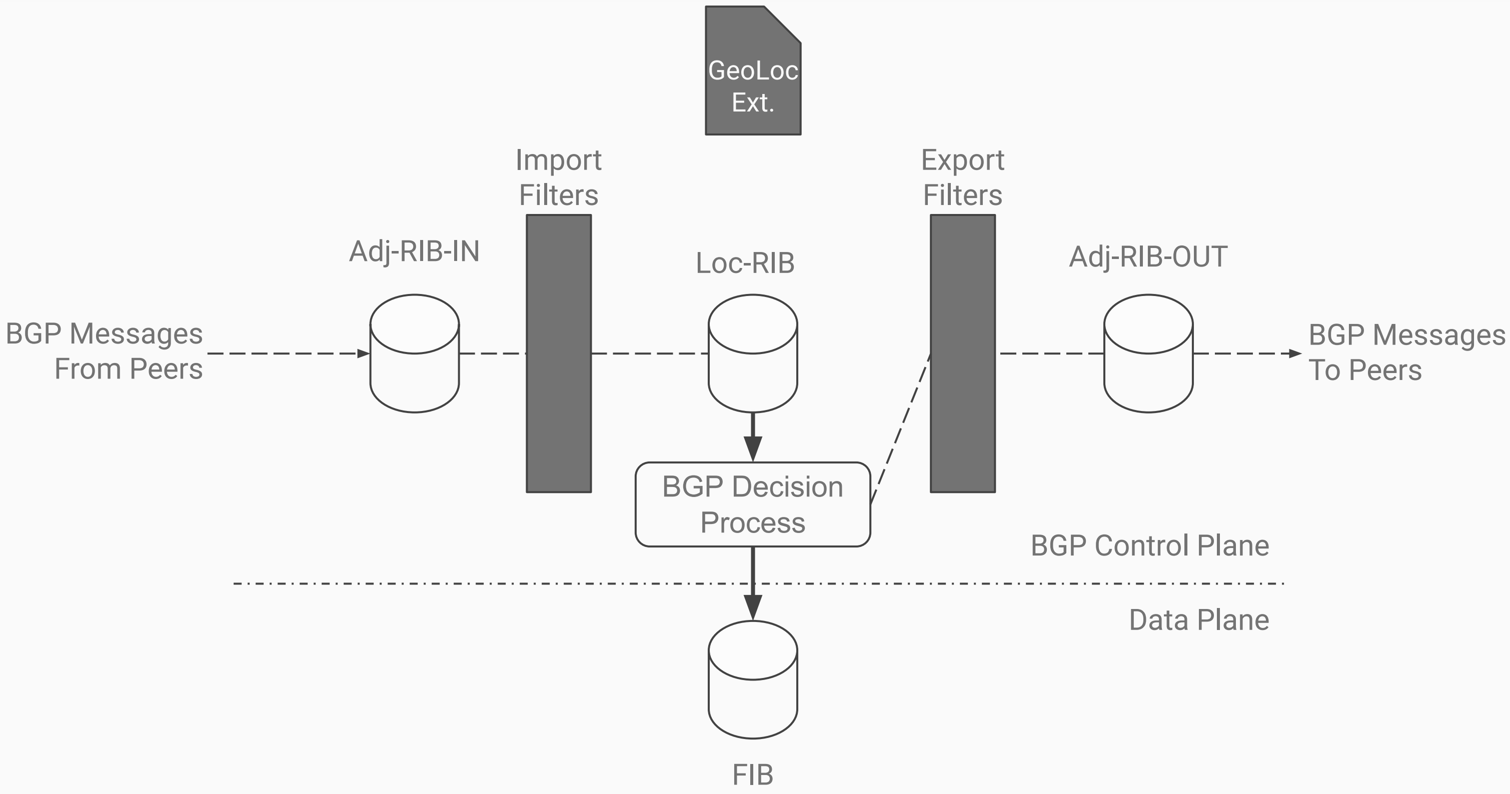
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



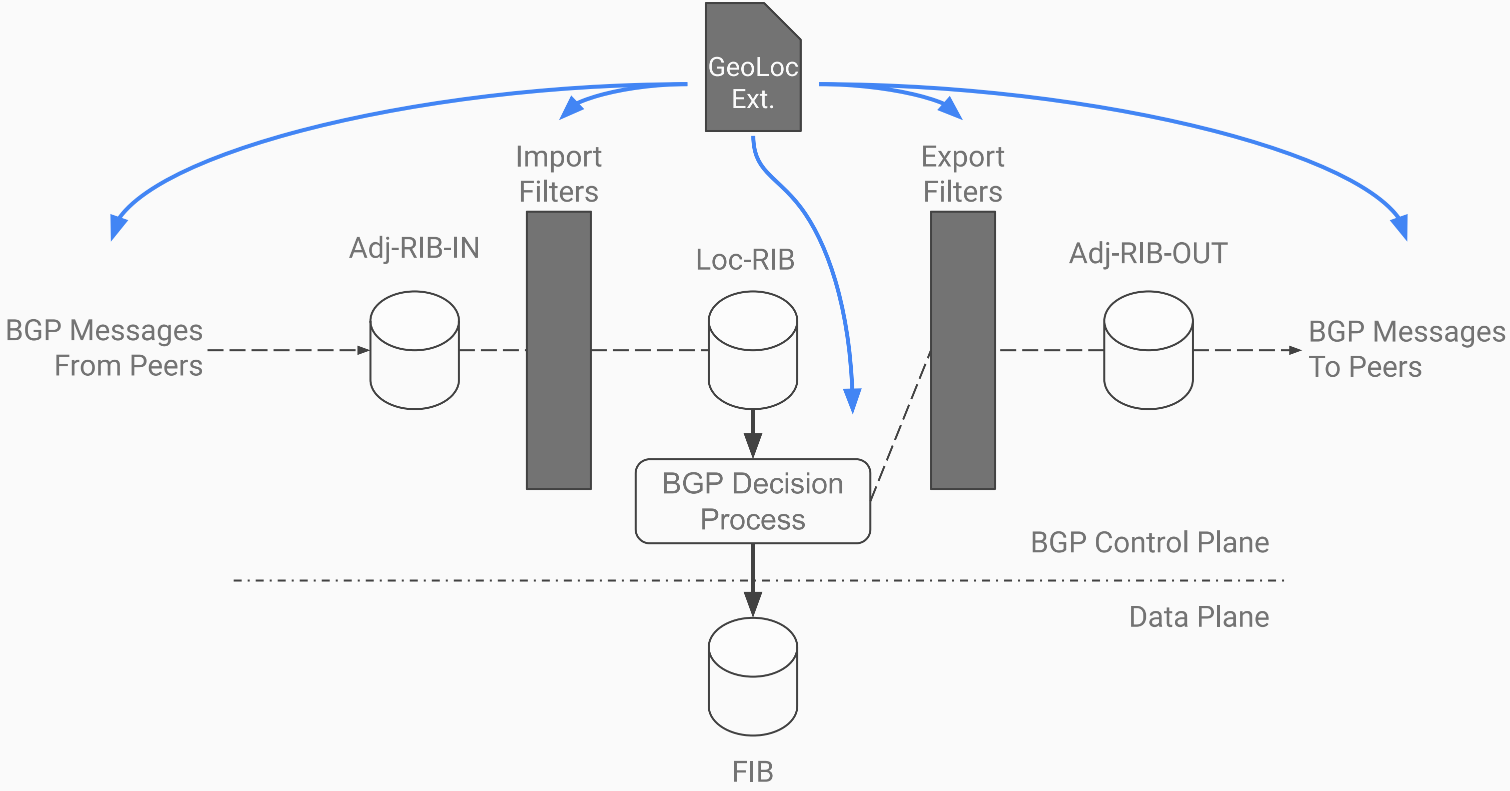
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



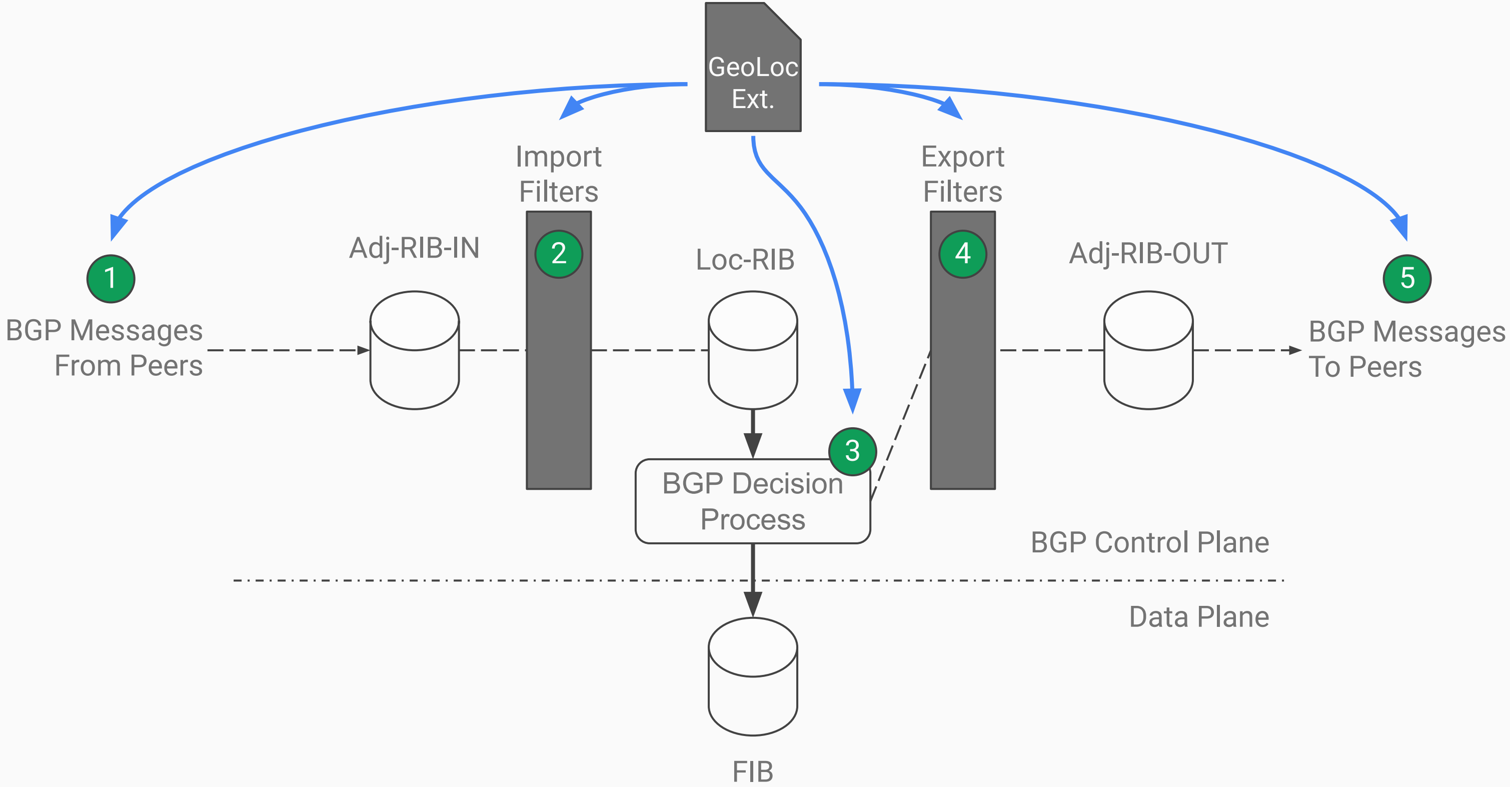
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



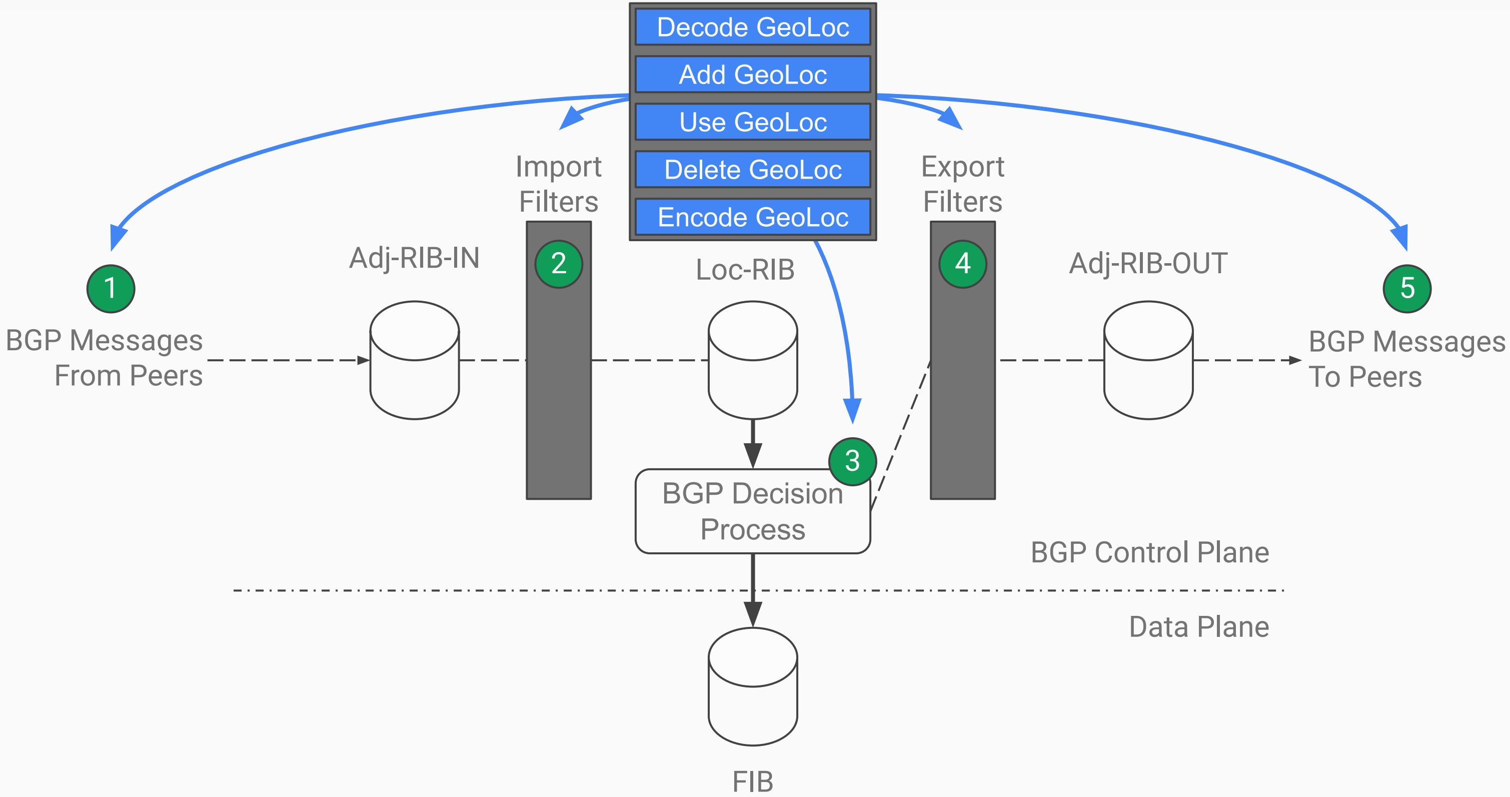
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



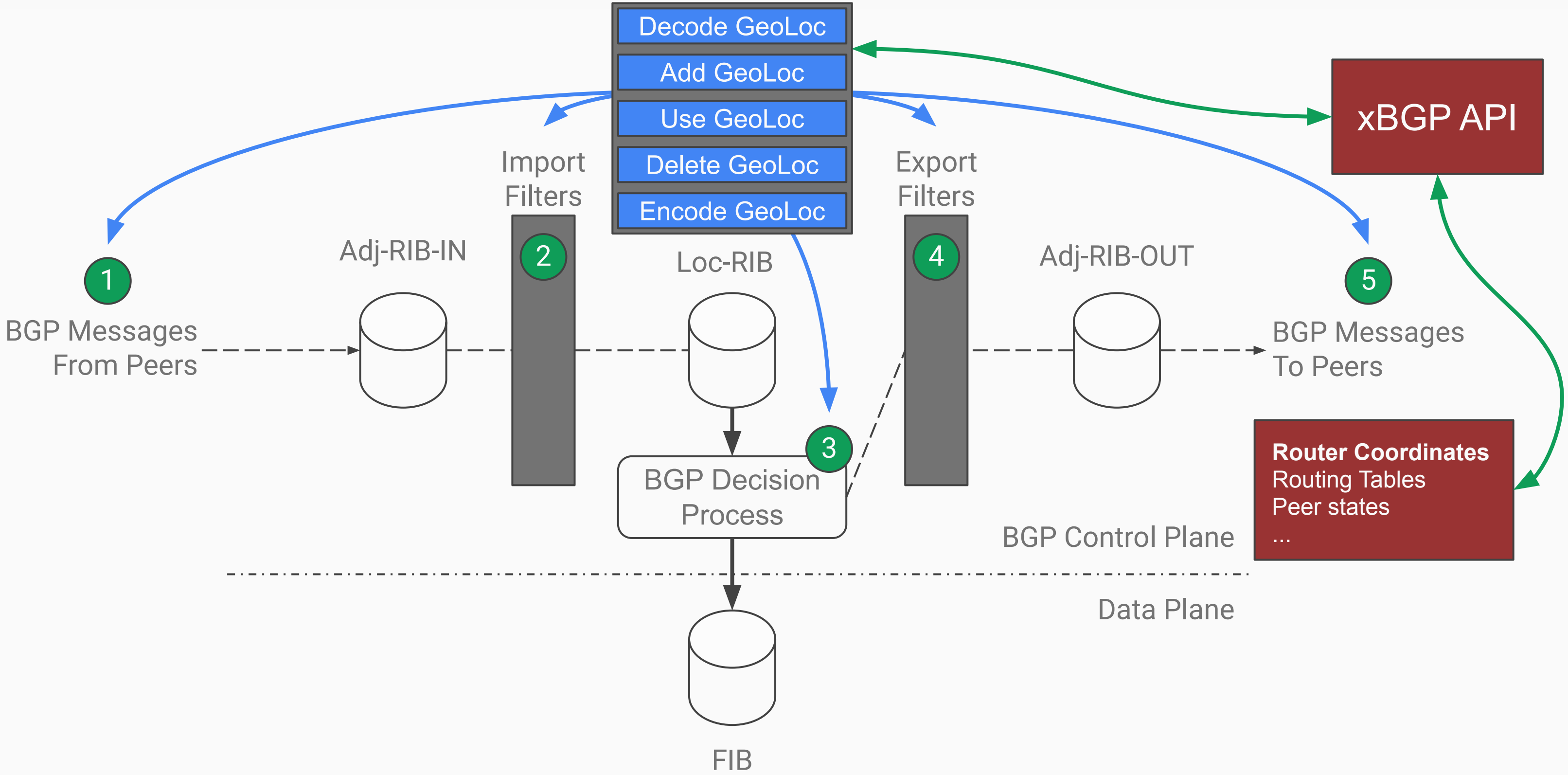
RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



RFC 4271 BGP Workflow

How to alter BGP to make it xBGP compliant ?



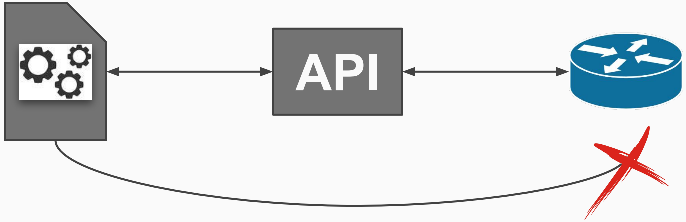
RFC 4271 BGP Workflow

The xBGP API

To communicate with BGP, xBGP extension codes **must** use the xBGP API.

The xBGP API contains :

- Send and Read BGP messages
- Setters & Getters (BGP routes, attributes, peer state, etc.)
- RIB access
- Utility Functions (memory, math, etc.)



Agenda

- Why bring programmability to BGP ?
- Inside xBGP
- **Use Cases**
- Verifying xBGP extensions
- Conclusion

Demonstrating the programmability of xBGP

xBGP requires a little adaptation to the host BGP implementation.

We have adapted both FRRouting and BIRD to be xBGP compliant

	FRRouting (LoC)	BIRD Routing (LoC)
Modification to the codebase	30	10
Insertion Points	73	66
Plugin API	624	415
`libxbgp`	3004 + dependencies	
User Space eBPF VM	2776	

https://www.pluginized-protocols.org/xbgp

Monitoring the AS Path length

We want to count the number of ASes contained in each BGP UPDATE.

It is difficult to achieve with traditional interfaces (CLI, NetConf, Yang, etc.)

Why monitoring the AS Path ?

- Filter out large AS Path
- Make analysis

Monitoring the AS Path length

```
uint64_t count_as_path(args_t *args) {
```

Monitoring the AS Path length

```c
uint64_t count_as_path(args_t *args) {
    unsigned int as_number = 0, segment_length;
    unsigned int *attribute_code = get_arg(ARG_CODE);
    unsigned int *as_path_len = get_arg(ARG_LENGTH);
    unsigned char *as_path = get_arg(ARG_DATA);

    if (!as_path || !as_path_len || !attribute_code) {
        // unable to fetch data from host implementation
        return EXIT_FAILURE;
    } else if (*attribute_code != AS_PATH_ATTR_ID) {
        return EXIT_FAILURE;
    }
```

Retrieve data from the host implementation

Monitoring the AS Path length

```c
uint64_t count_as_path(args_t *args) {
    unsigned int as_number = 0, segment_length;
    unsigned int *attribute_code = get_arg(ARG_CODE);
    unsigned int *as_path_len = get_arg(ARG_LENGTH);
    unsigned char *as_path = get_arg(ARG_DATA);

    if (!as_path || !as_path_len || !attribute_code) {
        // unable to fetch data from host implementation
        return EXIT_FAILURE;
    } else if (*attribute_code != AS_PATH_ATTR_ID) {
        return EXIT_FAILURE;
    }
    // core part of the plugin
    while (i < *as_path_len) {
        segment_length = as_path[i + 1];
        as_number += segment_length;
        i += (segment_length * 4) + 2;;
    }
```

Retrieve data from the host implementation

Parse the AS-PATH attribute

Monitoring the AS Path length

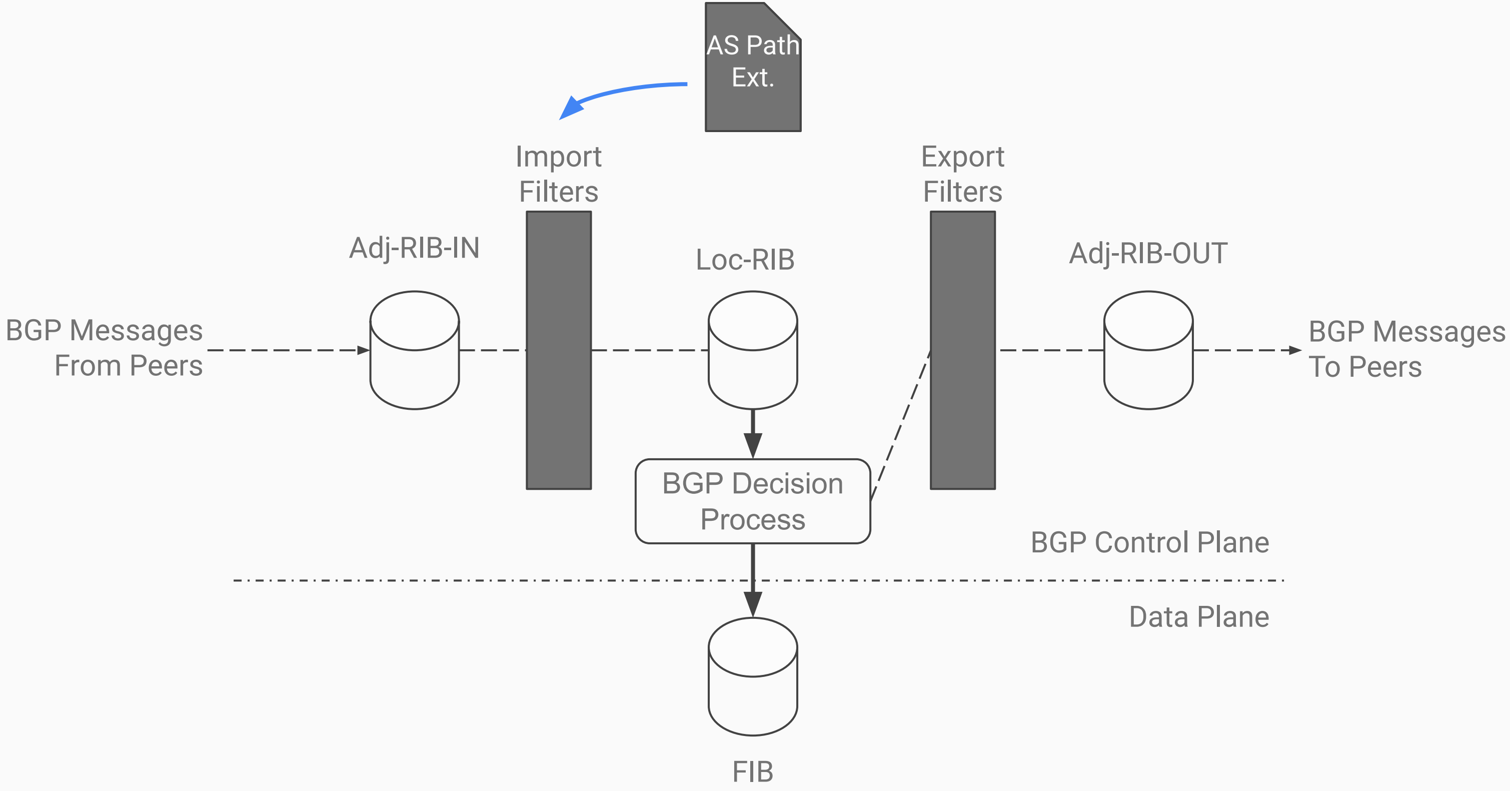
```c
uint64_t count_as_path(args_t *args) {
    unsigned int as_number = 0, segment_length;
    unsigned int *attribute_code = get_arg(ARG_CODE);
    unsigned int *as_path_len = get_arg(ARG_LENGTH);
    unsigned char *as_path = get_arg(ARG_DATA);

    if (!as_path || !as_path_len || !attribute_code) {
        // unable to fetch data from host implementation
        return EXIT_FAILURE;
    } else if (*attribute_code != AS_PATH_ATTR_ID) {
        return EXIT_FAILURE;
    }
    // core part of the plugin
    while (i < *as_path_len) {
        segment_length = as_path[i + 1];
        as_number += segment_length;
        i += (segment_length * 4) + 2;;
    }
    // log the message. If it fails, returns an error code
    if (log_msg(L_INFO "as_count:%d\n", LOG_UINT(as_number)) != 0) {
        return EXIT_FAILURE;
    }
    return EXIT_SUCCESS;
}
```

Retrieve data from the host implementation

Parse the AS-PATH attribute

Send to the logger (syslog, stdout, file, etc.)

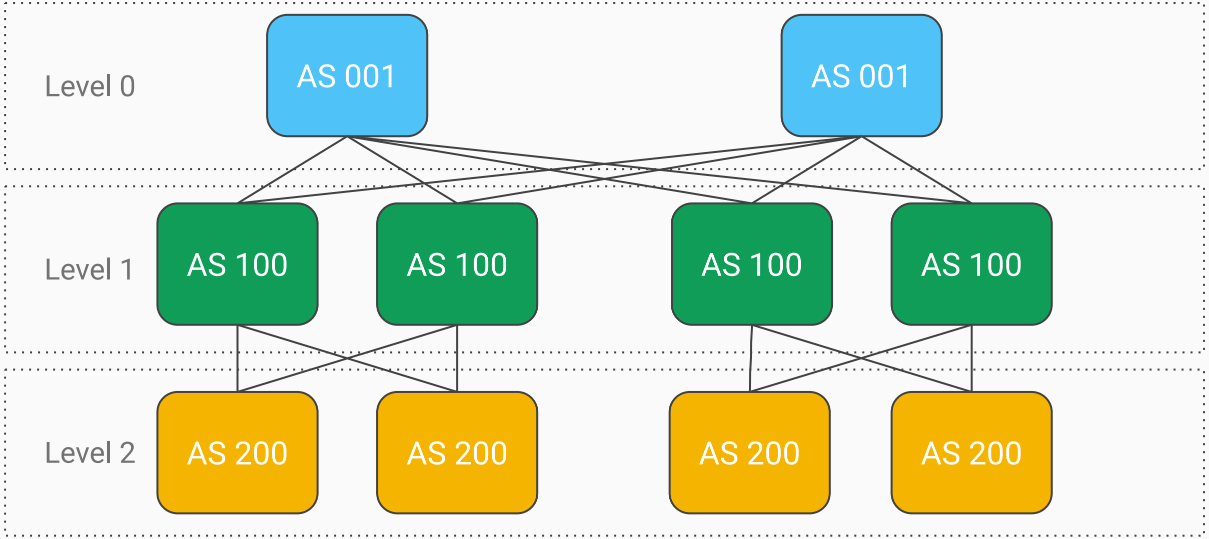
Monitoring the AS Path length



RFC 4271 BGP Workflow

Valley Free path check

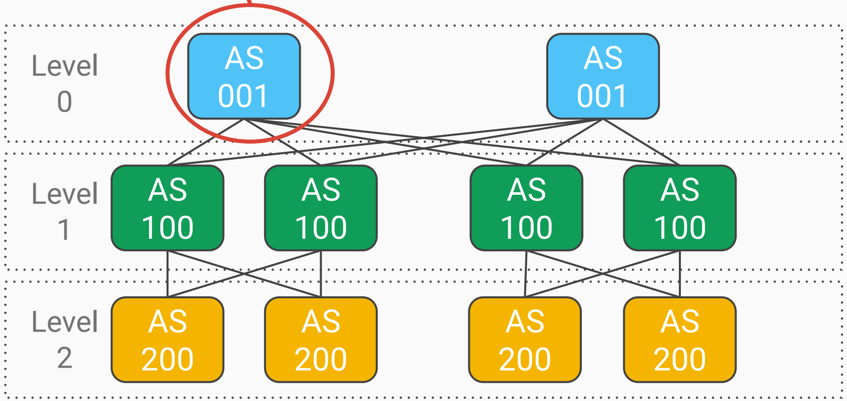
RFC7938 Use of BGP for Routing in Large-Scale Data Centers



Valley Free path check

RFC7938 Use of BGP for Routing in Large-Scale Data Centers

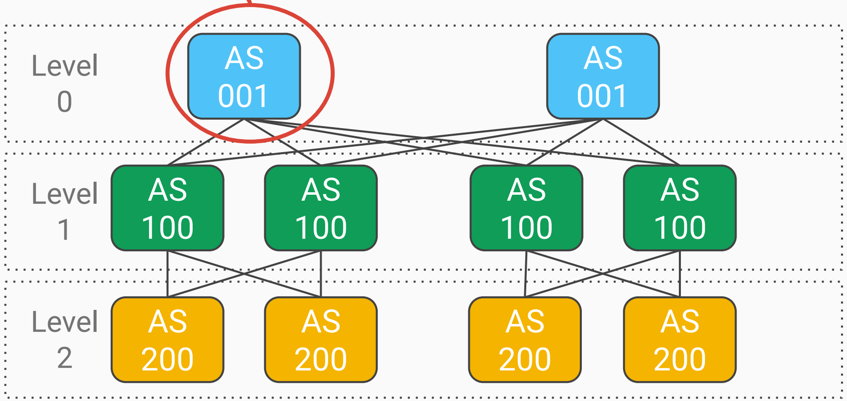
```
MyRouterCli > show ip bgp

BGP Routing table information for VRF default
Router identifier 192.168.254.5, local AS number 1

       Network          Next Hop         Metric  LocPref  Weight  Path
* >Ec 192.168.10.0/24   192.168.255.20   0       100      0       100 200 i
*  ec  192.168.10.0/24  192.168.255.4    0       100      0       100 200 i
* >Ec 192.168.254.3/32  192.168.255.4    1       100      0       100 200 i
*  ec  192.168.254.3/32 192.168.255.20   0       100      0       100 200 i
* >Ec 192.168.254.4/32  192.168.255.20   0       100      0       100 200 i
```



Valley Free path check

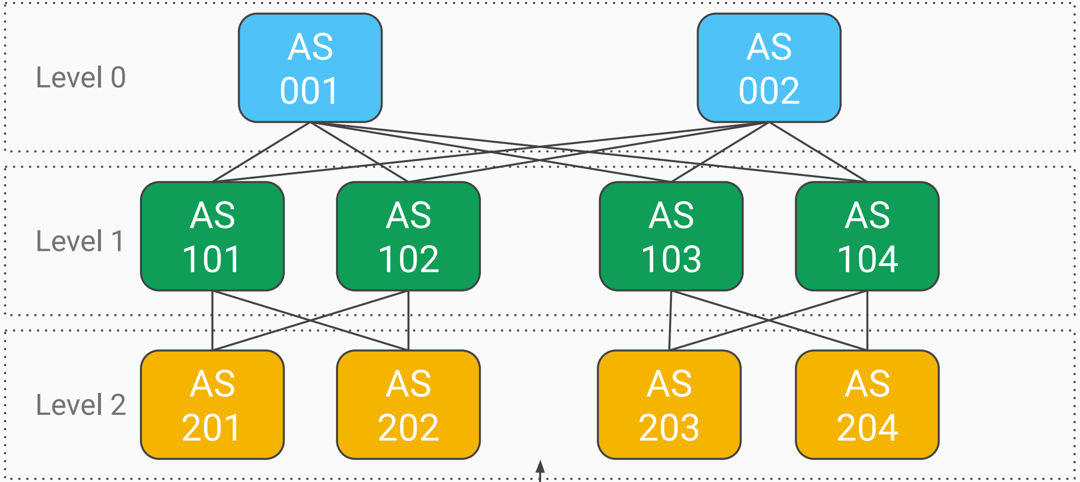
RFC7938 Use of BGP for Routing in Large-Scale Data Centers

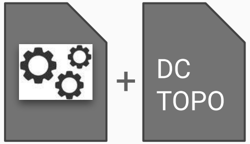
```
MyRouterCli > show ip bgp

BGP Routing table information for VRF default
Router identifier 192.168.254.5, local AS number 1

        Network          Next Hop        Metric  LocPref  Weight  Path
* >Ec 192.168.10.0/24    192.168.255.20  0       100      0       100 200 i
*  ec  192.168.10.0/24   192.168.255.4   0       100      0       100 200 i
* >Ec 192.168.254.3/32   192.168.255.4   1       100      0       100 200 i
*  ec  192.168.254.3/32  192.168.255.20  0       100      0       100 200 i
* >Ec 192.168.254.4/32   192.168.255.20  0       100      0       100 200 i
```



Where are these routes sourced from ?

Valley Free path check with xBGP

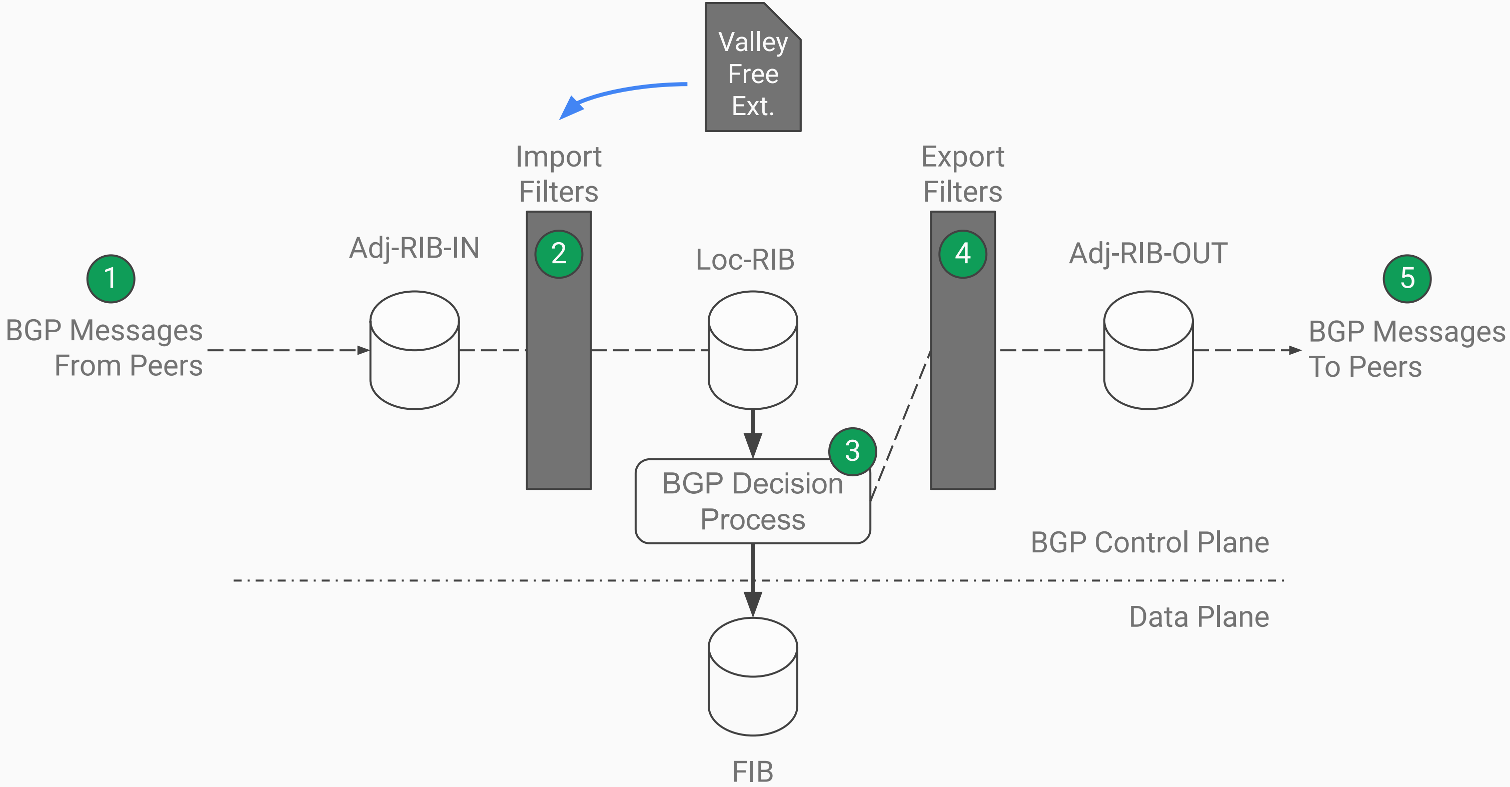




(81 LoC)

One plugin + one topology manifest
for all routers !

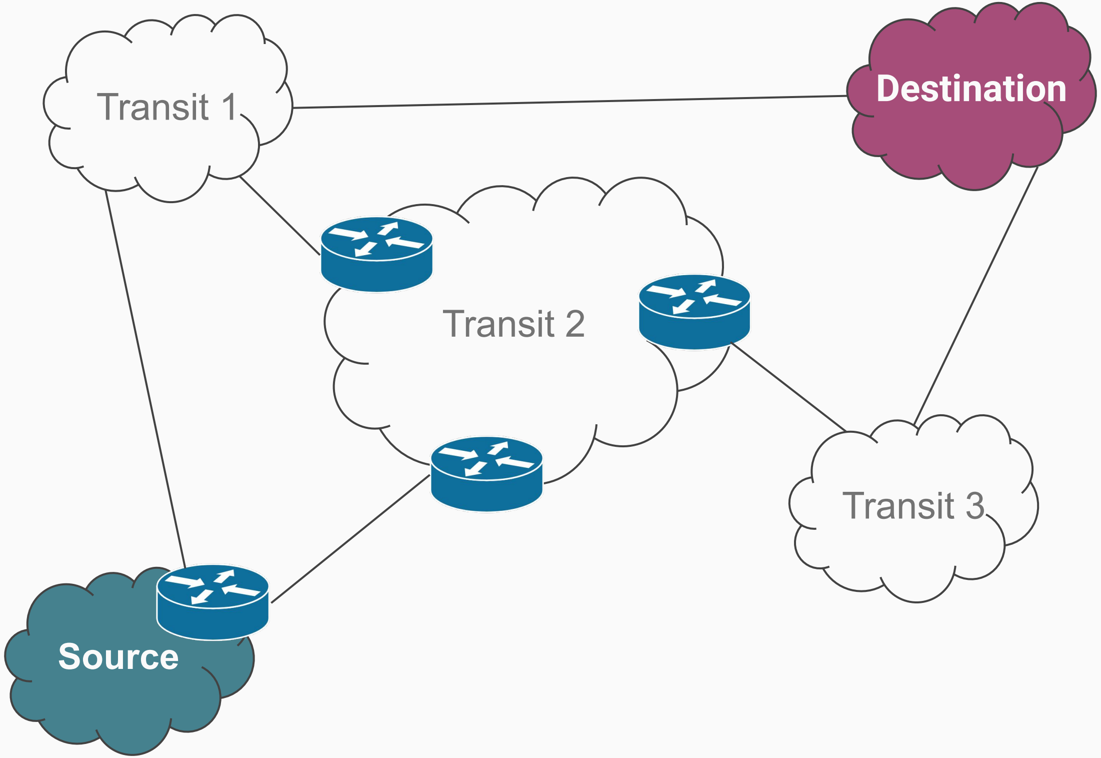
Valley Free Path Check



RFC 4271 BGP Workflow

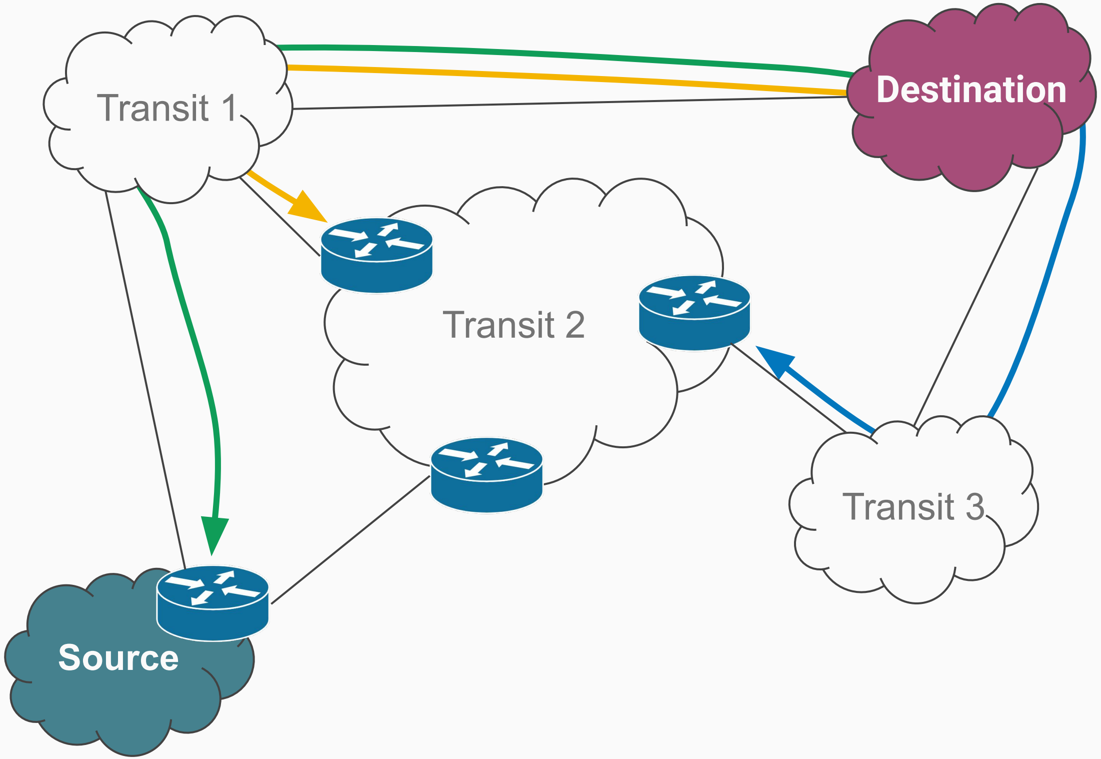
The Path Selection Service

“Source AS” can not influence “Transit 2” route selection



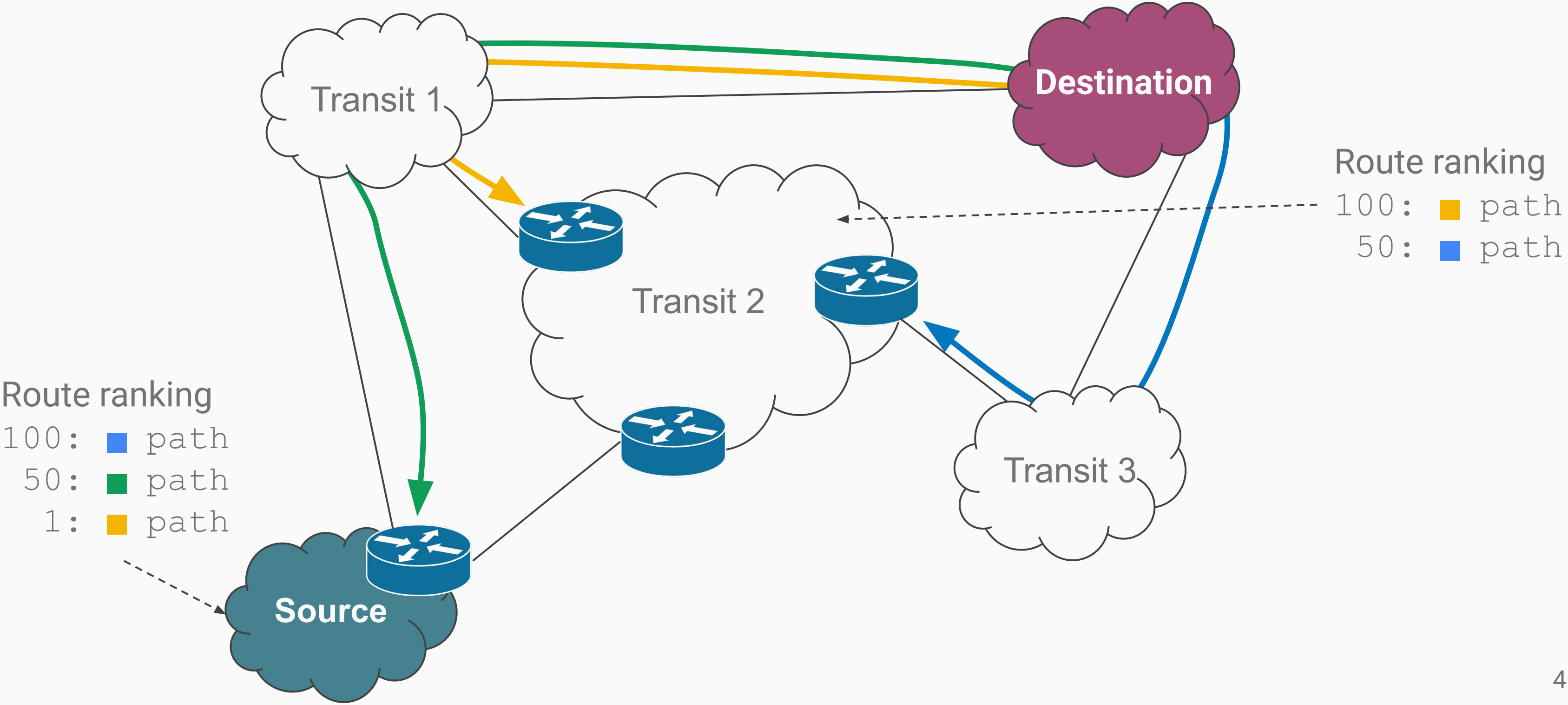
The Path Selection Service

“Source AS” can not influence “Transit 2” route selection



The Path Selection Service

“Source AS” can not influence “Transit 2” route selection

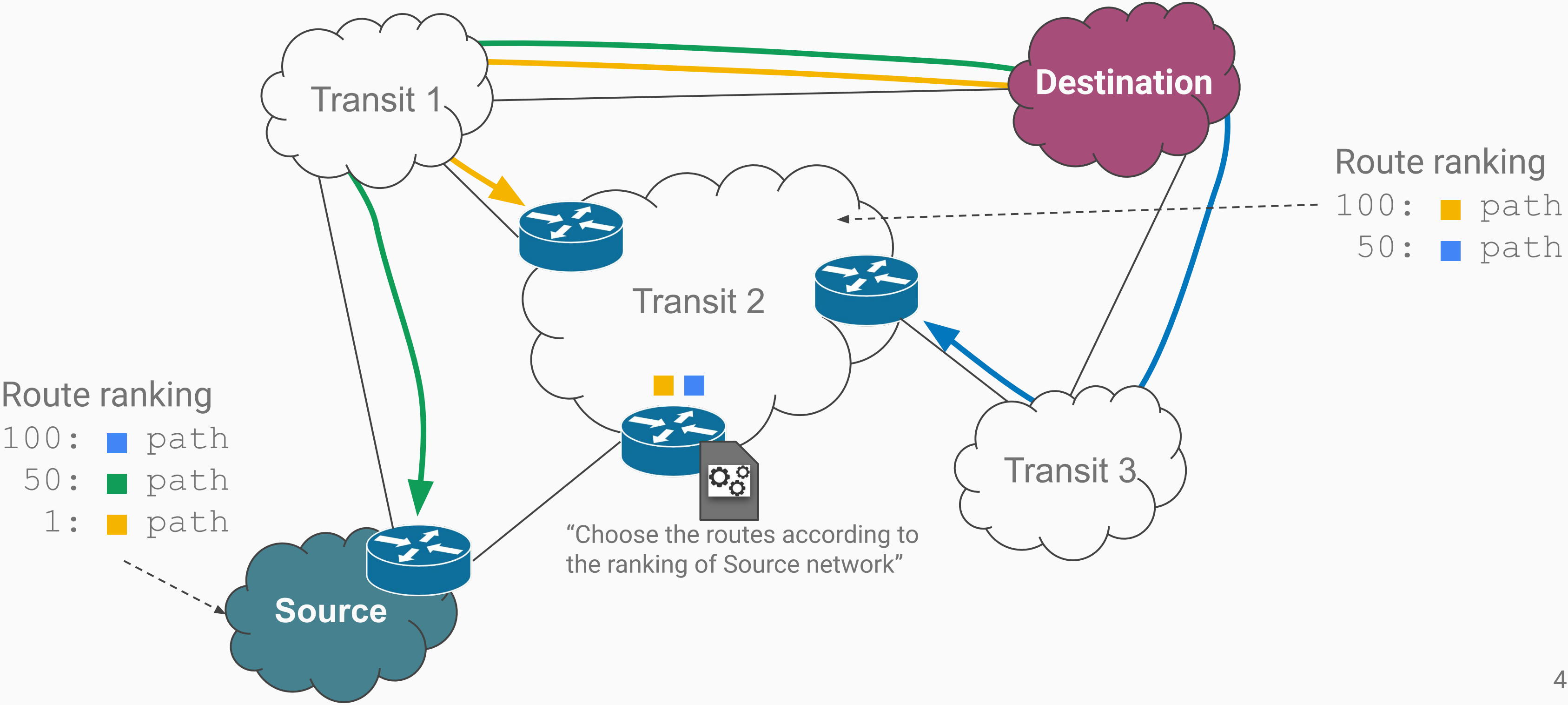


Route ranking
100: 🟨 path
50: 🟦 path

Route ranking
100: 🟦 path
50: 🟩 path
1: 🟨 path

The Path Selection Service

“Source AS” can not influence “Transit 2” route selection



Route ranking

100: 🟨 path

50: 🟦 path

Route ranking

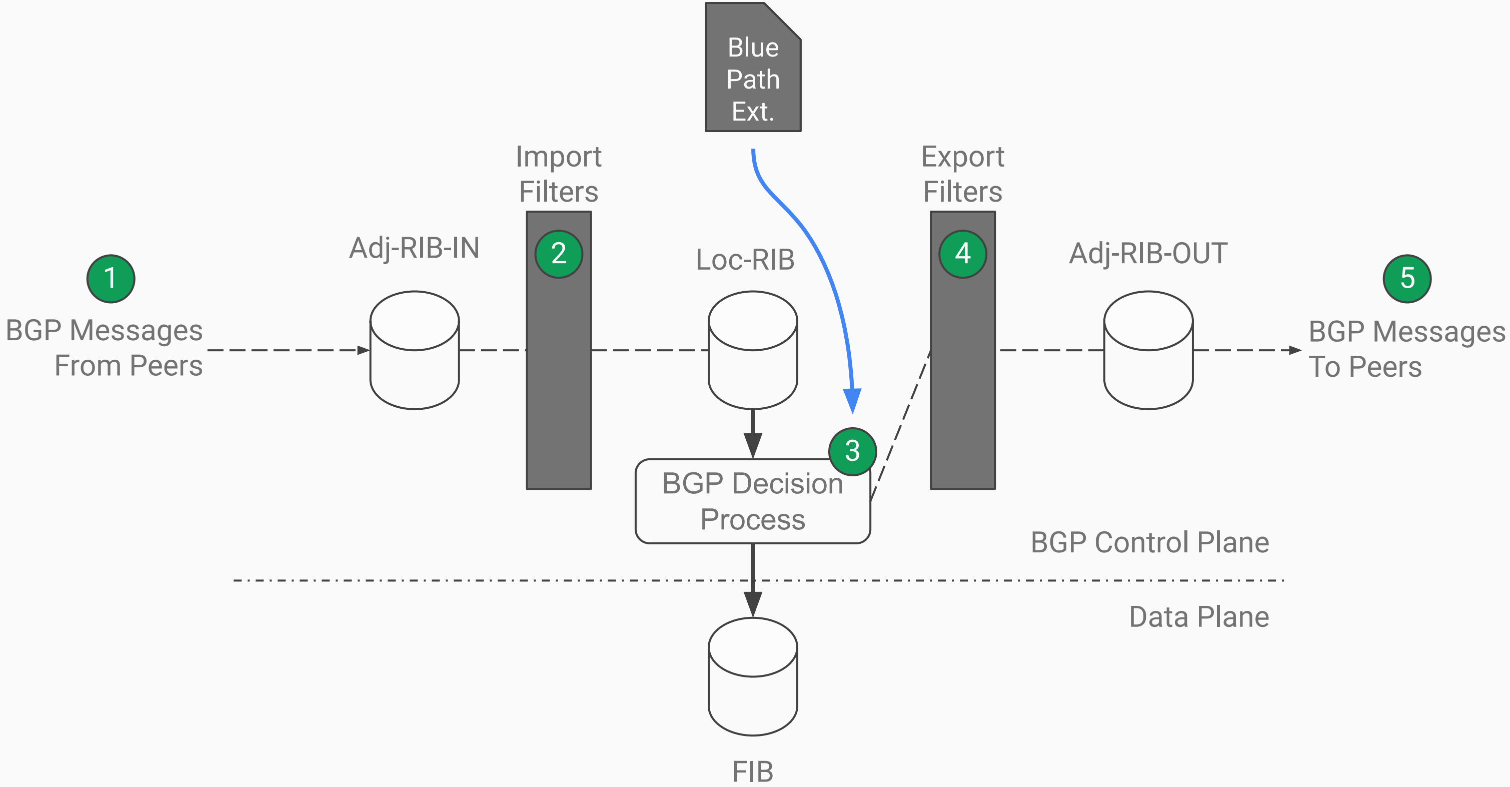
100: 🟦 path

50: 🟩 path

1: 🟨 path

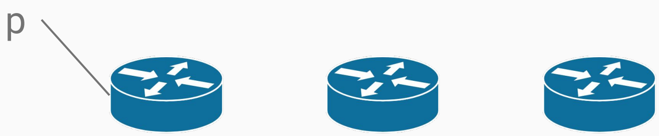
“Choose the routes according to the ranking of Source network”

The Path Selection Service

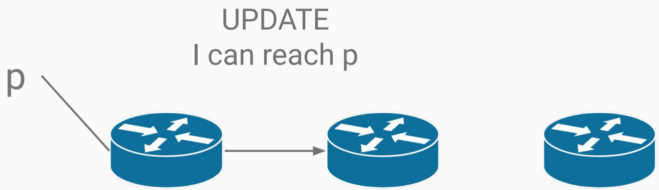


RFC 4271 BGP Workflow

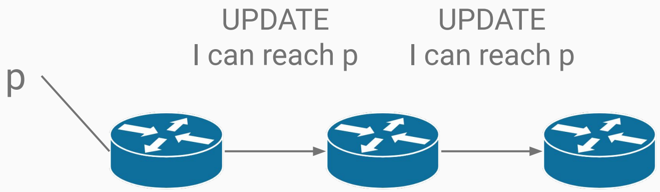
Detecting BGP Zombies



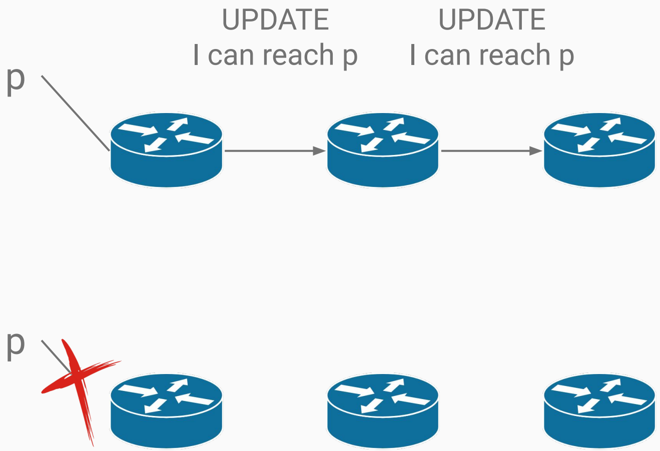
Detecting BGP Zombies



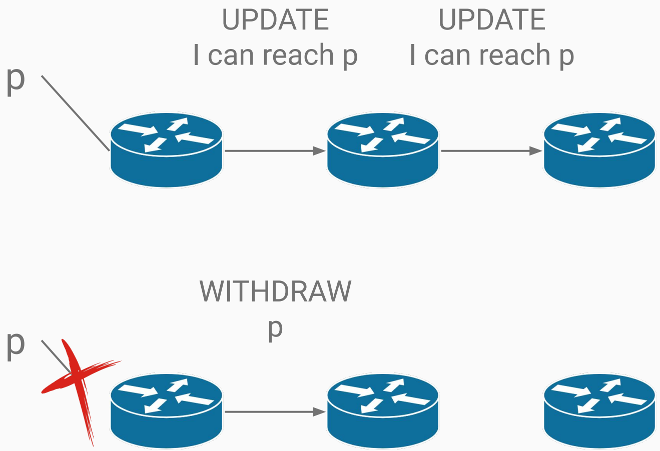
Detecting BGP Zombies



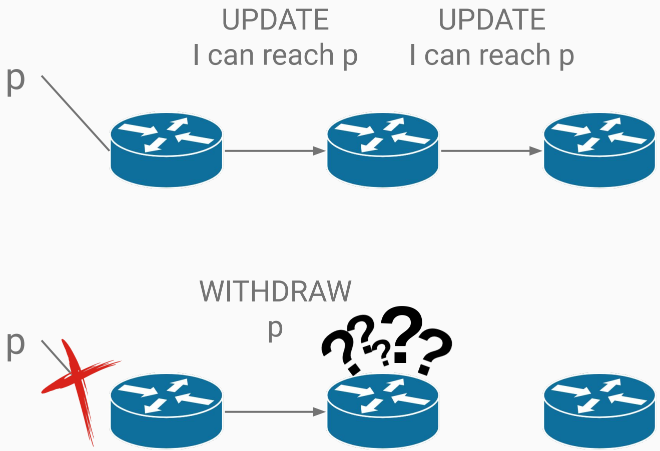
Detecting BGP Zombies



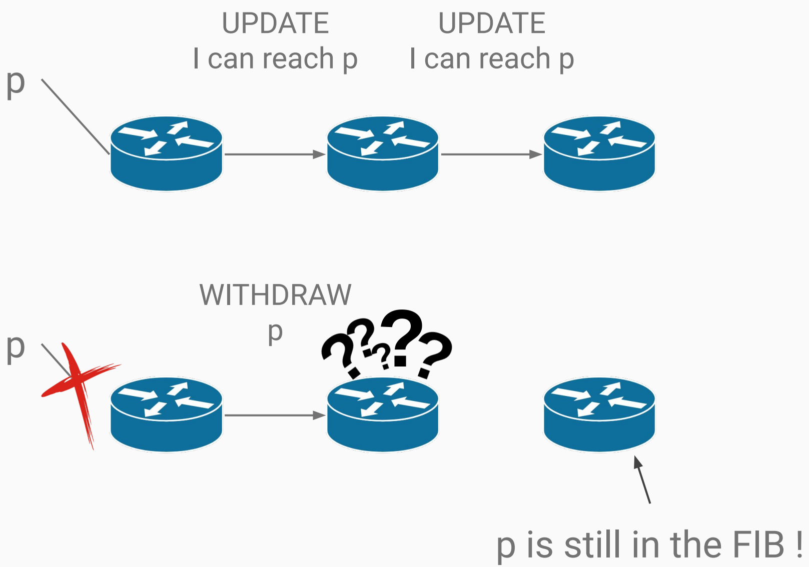
Detecting BGP Zombies



Detecting BGP Zombies



Detecting BGP Zombies



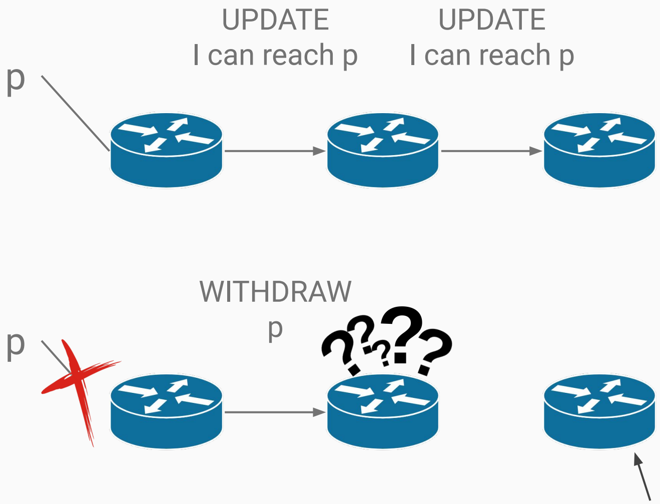
UPDATE
I can reach p

UPDATE
I can reach p

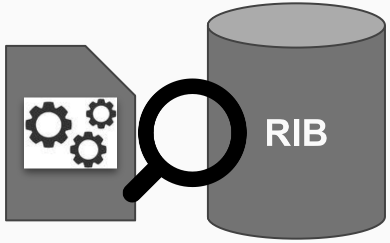
WITHDRAW
p

p is still in the FIB !

Detecting BGP Zombies

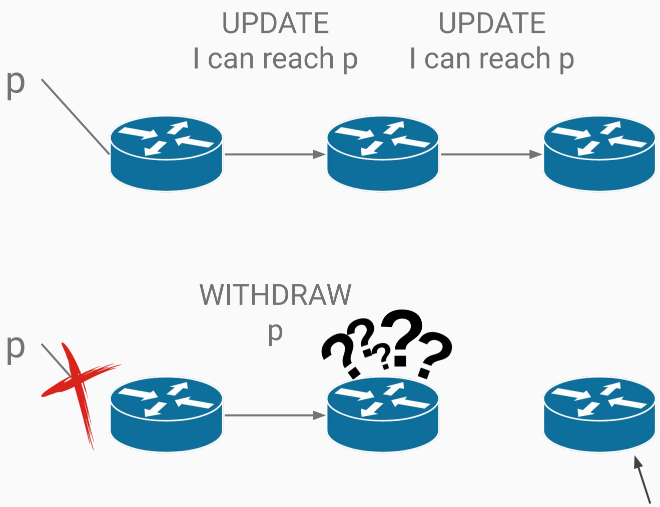


p is still in the FIB !



Checks routes older than
<x> <unit of time>

Detecting BGP Zombies



p

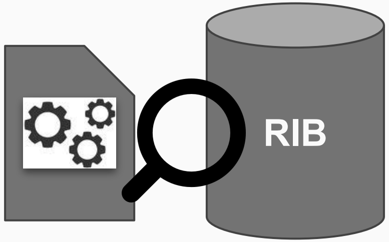
UPDATE
I can reach p

UPDATE
I can reach p

p

WITHDRAW
p

p is still in the FIB !

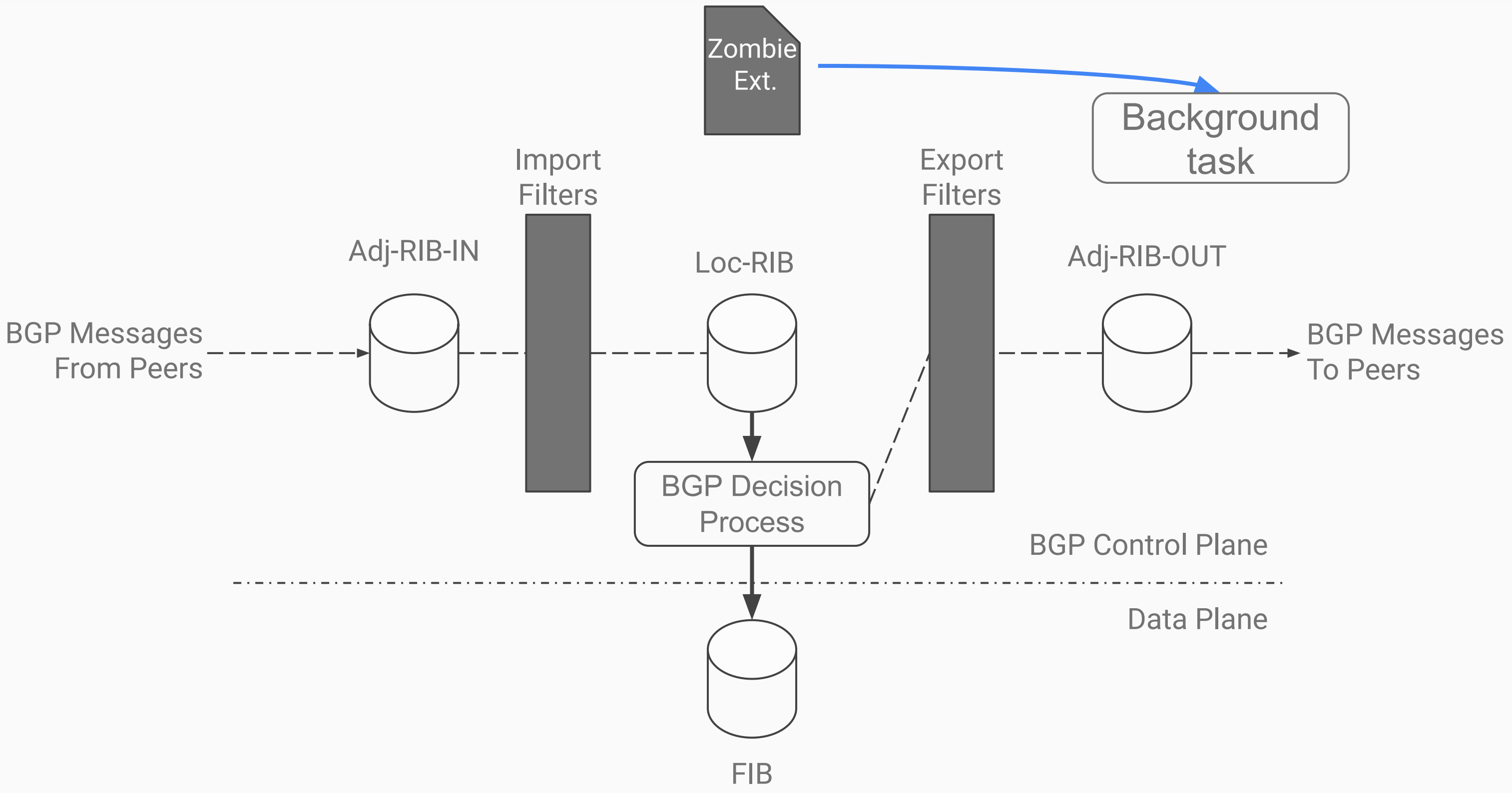


Checks routes older than
<x> <unit of time>

p since 4w 7h 36m 2s

Ask the upstream router to confirm if the route is still valid

Detecting BGP Zombies



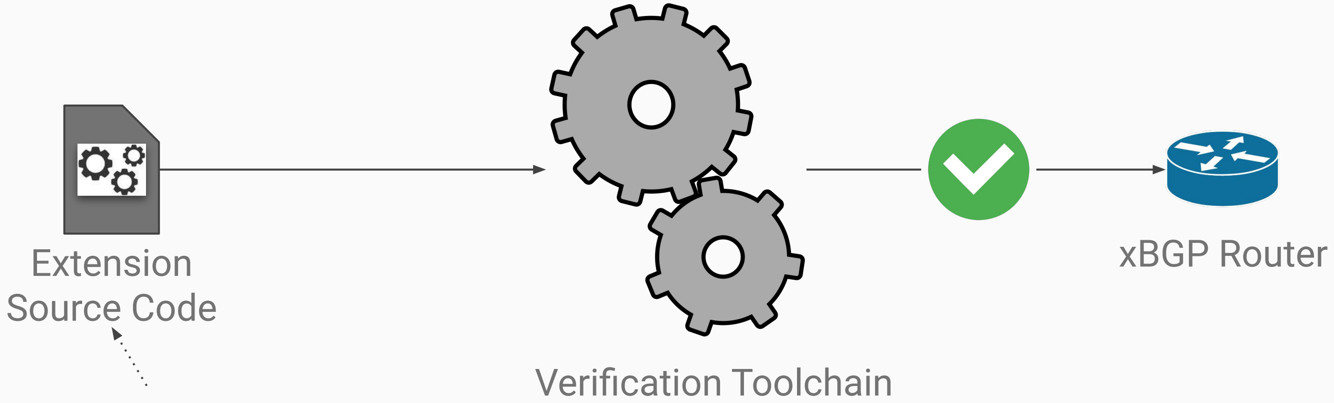
RFC 4271 BGP Workflow

Agenda

- Why bring programmability to BGP ?
- Inside xBGP
- Use Cases
- **Verifying xBGP extensions**
- Conclusion

Executing arbitrary code is dangerous

The code executed by xBGP is untrusted. Could it break BGP ?

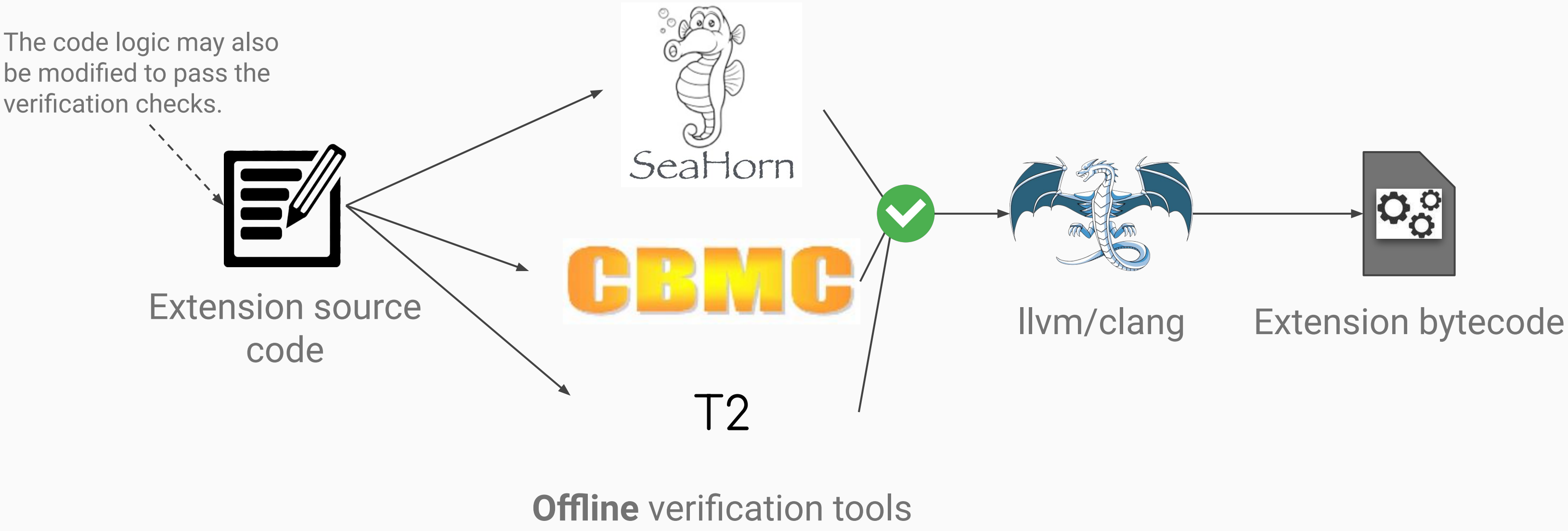


The code should satisfy :

1. Termination
2. Memory Isolation
3. BGP Syntax
4. API Restriction

How to verify the properties ?

The code should be annotated manually, and then passed to the verification tools.


The code logic may also be modified to pass the verification checks.
Extension source code
SeaHorn
CBMC
T2
Offline verification tools
llvm/clang
Extension bytecode

The right tool to the right property

T2

- Termination property

libxBGP

- xBGP API restriction (offline)

CBMC

- Buffer overflow
- Memory isolation
- Memory leak
- Conversion errors
- ...

SeaHorn

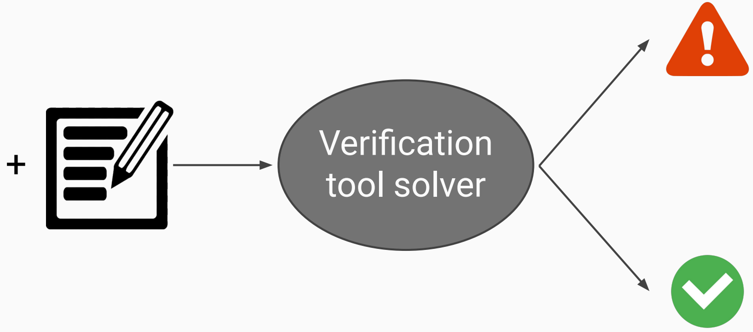
- BGP Related properties (i.e. BGP syntax)

Extension codes are guaranteed to not violate the properties we defined

Example: verifying the BGP syntax

If the xBGP extension adds Geographic coordinates, it must respect the TLV format defined in the draft.

```
attribute.type.flags.optional == 1
attribute.type.flags.transitive == 0
attribute.type.flags.partial == 0
attribute.type.flags.extended == 0

attribute.type.code == GeoTLV Identifier

attribute.length == 8

lo_latitude <= attribute.data.latitude <= hi_latitude
lo_longitude <= attribute.data.longitude <= hi_longitude
```



Conclusion

With xBGP, BGP implementations can become truly extensible

T. Wirtgen, Q. De Coninck, L. Vanbever, R. Bush, O. Bonaventure, *xBGP: When You Can't Wait for the IETF and Vendors*, Hotnets'20, Nov. 2020

See https://www.pluginized-protocols.org/xbgp for running source code

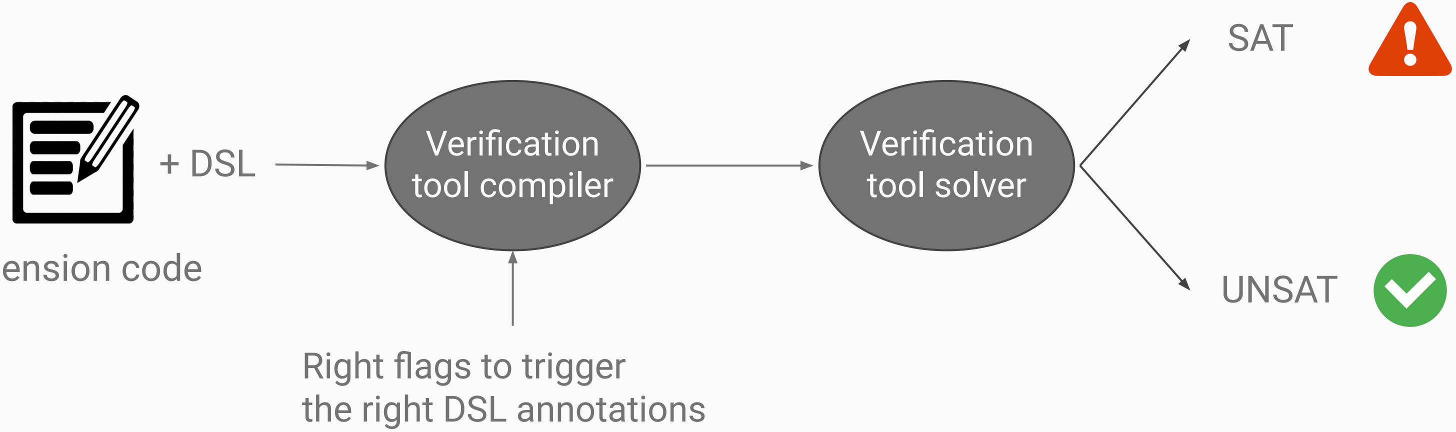
xBGP provides new opportunities with other routing protocols

thomas.wirtgen@uclouvain.be

Backup Slides

Using several tools is cumbersome

We propose a kind of DSL that unifies the annotations of all verification tools


+ DSL
Extension code
Verification tool compiler
Right flags to trigger the right DSL annotations
Verification tool solver
SAT
UNSAT