Routers and Routing

Routing Tables and Route Summarisation

Nick Urbanik <nicku@nicku.org>

A computing department

Contents

1Routing21

1-1

Modern Routing Tables

- Each entry in a routing table has 3 main items:
- A network address (the destination)
- A netmask length
- A next hop address

```
$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Iface
172.19.64.0     0.0.0.0         255.255.192.0   U     eth0
127.0.0.0       0.0.0.0         255.0.0.0       U     lo
0.0.0.0         172.19.127.254  0.0.0.0         UG    eth0
```

The Routing Algorithm

- For a given destination IP address
- Search the routing table for the longest prefix match for the address
- Extract the next hop address from the routing table entry
- Send the packet to the next hop address
- If no match found, report that the destination is unreachable.

Longest Prefix

- So what does “longest prefix match” mean?
- To see if the prefix matches,
  - Bitwise AND netmask with destination
  - Bitwise AND netmask with network from routing table entry
  - If the two results are equal, then the prefix matches
- If we do the same for all entries in the routing table, the match with the longest netmask wins.

Example:

- Given this routing table, where does the packet with destination 192.168.0.3 go to?

```
192.168.0.0     0.0.0.0         255.255.255.0   U     eth0
192.168.25.0    0.0.0.0         255.255.255.0   U     vmnet1
192.168.0.0     172.19.35.254   255.255.0.0     UG    ppp1
0.0.0.0         202.180.160.251 0.0.0.0         UG    ppp0
```

- How about 192.168.128.48?
- 192.168.25.10?
- 192.169.0.1?

1CIDR61

5-1

The Big Emergency

- In the early 90s, it became apparent that two problems were quickly going to become overwhelming:
  - *Address depletion* — we were running out of IP addresses
  - *Router table explosion* — the routing tables were growing too fast for the router hardware to cope

The Solution: CIDR and NAT

- Two solutions were developed:
- CIDR (Classless Internet Domain Routing), and
- NAT (Network Address Translation).
  - NAT allows a firewall or router to present one address to the outside world, but many to the inside.
  - In Linux, use iptables.
  - Use private addresses:
  - 192.168.0.0/16
  - 172.12.0.0/12
  - 10.0.0.0/8

Address Depletion

- Class C was too small for medium sized enterprises
- Class B was too big
- Many organisations asked for (and received) class B networks when they needed only a /22 or /21 network
- This used up the available 2^{32} addresses too fast
- Later there was a need for small Internet allocations of 1 or 2 addresses.
  - Class C was too wasteful for this.

Router Table Explosion

- As class B addresses became scarce, SMEs were given a number of class C network allocations
- But each class C needed a separate routing table advertisement
- Local information about the internal network structure of a company needed to be advertised world wide
- This did not scale
- By now routing would need much more CPU and RAM than is currently used, and the Internet would have slowed further.

How does CIDR solve them?

- New address allocations can be sized accurately to the need
  - When requesting addresses, the authority (`http://www.apnic.net/`) will reserve some addresses for future growth if you specify you will need them
- New address allocations are made taking into account neighbouring networks
- Aim is to *summarise* many routes into as few routes as possible.

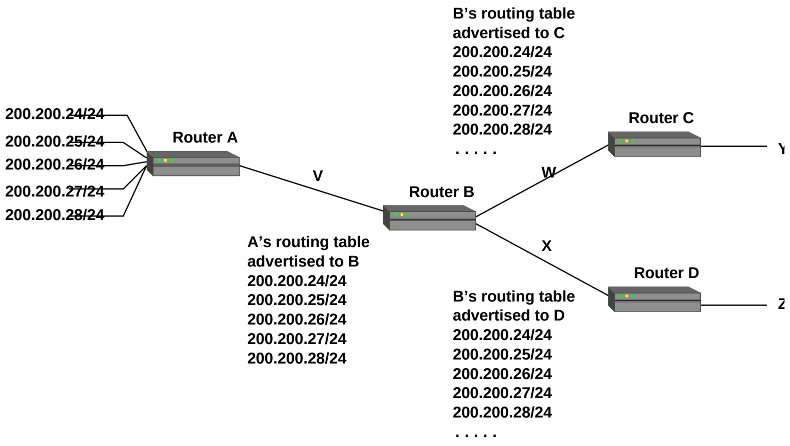
1Aggregating Routes111

10-1

Aggregating routes

- Routers summarise routes themselves when they use *classless* routing protocols such as:
  - RIP2
  - OSPF
  - BGP

Without Route Summarisation



200.200.24/24
200.200.25/24
200.200.26/24
200.200.27/24
200.200.28/24

Router A

V

Router B

W

X

Router C

Y

Router D

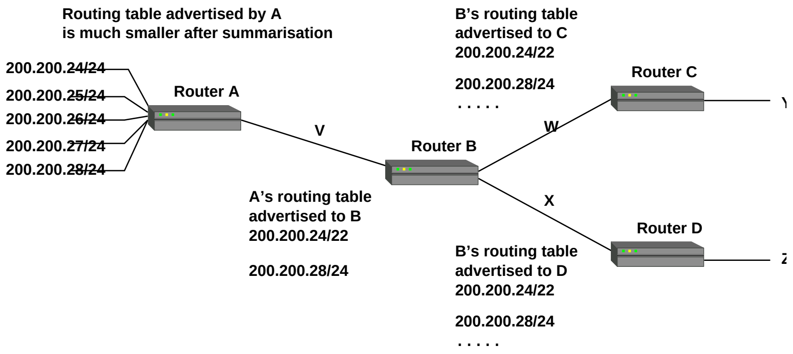
Z

B's routing table
advertised to C
200.200.24/24
200.200.25/24
200.200.26/24
200.200.27/24
200.200.28/24
. . . . .

A's routing table
advertised to B
200.200.24/24
200.200.25/24
200.200.26/24
200.200.27/24
200.200.28/24

B's routing table
advertised to D
200.200.24/24
200.200.25/24
200.200.26/24
200.200.27/24
200.200.28/24
. . . . .

With Route Summarisation



Routing table advertised by A
is much smaller after summarisation

200.200.24/24
200.200.25/24
200.200.26/24
200.200.27/24
200.200.28/24

Router A

V

Router B

W

X

Router C

Y

Router D

Z

B's routing table
advertised to C
200.200.24/22

200.200.28/24
. . . . .

A's routing table
advertised to B
200.200.24/22

200.200.28/24

B's routing table
advertised to D
200.200.24/22

200.200.28/24
. . . . .

Explanation

- The first diagram shows *all* subnets behind router A advertised everywhere
  - This is because the routers are unable to summarise the routes
- The second diagram shows the subnets behind A summarised into two routes instead of 5
  - The routers must be running a classless routing protocol such as OSPF or RIP2.

How the Routes were Summarised

- 200.200.24.0/24: 24_{10} = 00011000_2
- 200.200.25.0/24: 25_{10} = 00011001_2
- 200.200.26.0/24: 26_{10} = 00011010_2
- 200.200.27.0/24: 27_{10} = 00011011_2
  - So these can be summarised into:
  - 200.200.24.0/22
- 200.200.28.0/24: 28_{10} = 00011100_2
  - This cannot be summarised with the other routes, so it must be advertised separately.

Route Aggregation: `NetAddr::IP`

- There is a Perl module for working with IP addresses (of course):
- `NetAddr::IP`
- Includes the method `compact()`, which takes a list of networks and returns a list of summarised address blocks.
- The next slide shows a little program that will aggregate address blocks given on the command line or on standard input.

route-aggregate

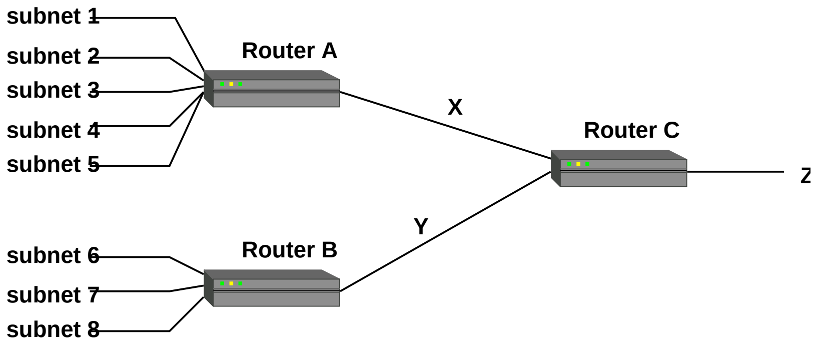
```
#! /usr/bin/perl -w
use NetAddr::IP;
$| = 1;
our ( @ips, @ip );
if ( @ARGV ) {
    @ips = @ARGV
} else {
    @ips = <STDIN>;
}
foreach my $ip ( @ips ) {
    push @ip, NetAddr::IP->new( $ip );
}
my @aggregated = NetAddr::IP::compact( @ip );
print "@aggregated\n";
```

Designing an Addressing Scheme

- Given one (or two) blocks of addresses, how do we allocate addresses to a network involving routers?
- Need also to allocate addresses to links between routers—these need their own little subnet

Example Problem

- Given a physical network layout as shown in the figure below
- Has 10 subnets (excluding the link Z)
- All three routers support CIDR addressing



Example Problem

- You are given:
  - The information on previous slide
  - Two address blocks:
    - 172.19.0/20
    - 172.19.128/28
- Requirements are:
  - Subnets 1 to 8 must each support up to 140 computers
  - Subnets must be assigned to allow maximum route aggregation
  - Any unused addresses must be kept in single blocks so that they can be used elsewhere or for future expansion

Solution — Links

- General strategy: determine the lower and upper limits on each subnet. Allocate networks in the order of smallest to largest.
- The smallest block of addresses is only suitable for allocating to the links, so allocate them first.
- Minimum size of each serial link is 4, as $2^{\lceil \log_2(2+2) \rceil} = 2^2$, giving a prefix size of $32 - 2 = 30$, i.e., /30.
- Allocate adjacent subnets to links X and Y, so that router C can aggregate routes to them.

subnet	*network*
subnet X	172.19.128.0/30
subnet Y	172.19.128.4/30

Solution — Larger Subnets — 1

- For each of the larger subnets, minimum size is 256, i.e., a /24 subnet
- 2^8 is the lowest power of 2 that contains $140 + 2$. ($2^{\lceil \log_2 (140+2) \rceil} = 2^8$; so prefix length $= 32 - 8 = 24$).

Solution — Larger Subnets — 2

- Let us allocate the lowest 8 /24 blocks from 172.19.0/20:

subnet	*network*
subnet 1	172.19.0.0/24
subnet 2	172.19.1.0/24
subnet 3	172.19.2.0/24
subnet 4	172.19.3.0/24
subnet 5	172.19.4.0/24
subnet 6	172.19.5.0/24
subnet 7	172.19.6.0/24
subnet 8	172.19.7.0/24

- In a tutorial exercise, you will determine what routes each router advertises.

order Gateway Protocol — BGP]Gateway Protocols

23-1

Gateway Protocols

Border Gateway Protocol — BGP

Classes of Routing Protocols

- Distance Vector or Link-State are two types of routing protocols.
- Another way to classify routing protocols is as follows:
- Intra-Domain routing:
  - routing of packets within the same Autonomous System (*AS*)
  - Interior Gateway Protocol IGP, RIP 2, OSPF, . . .
- Inter-Domain routing:
  - Inter-Domain routing is between multiple Autonomous Systems.
  - Exterior Gateway Protocol EGP, Border Gateway Protocol BGP
- Autonomous System (AS) refers to a group of routers (i.e. networks) administered by the same organization.
- Each AS is assigned a number. AS numbers range from

1BGP: AS Types271

26-1

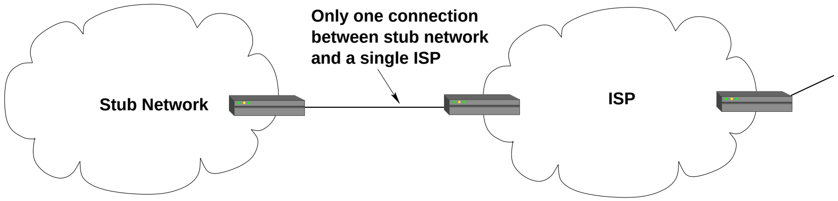
Gateway Protocols

- Inter-domain and Intra-domain routing protocols are also referred as Exterior and Interior routing protocols respectively.
- The first widely used exterior gateway protocol is called Exterior Gateway Protocol (EGP), it was designed to communicate reachability among the core routers of ARPANET.
- EGP is more a reachability protocol than a routing protocol, it only tests reachability but not makes intelligent routing decisions.
- EGP is replaced by the Border Gateway Protocol (BGP). The current version of BGP is version 4
  - earlier versions don't support CIDR, so are obsolete

Border Gateway Protocol BGP

- BGP is an inter-domain (inter-AS) routing protocol. However, BGP can also be used within an AS.
- When used between AS, BGP is referred as Exterior BGP (eBGP).
- When used within an AS, BGP is referred as Interior BGP (iBGP).
- BGP is mainly used in core routers in the Internet, for connections between Internet Service Providers. Large networks (universities and big enterprises) also use BGP to connect to ISPs. Within these networks, however, other Interior Gateway Protocols (such as RIP or OSPF) are used rather than iBGP.

Single-homed Autonomous Systems



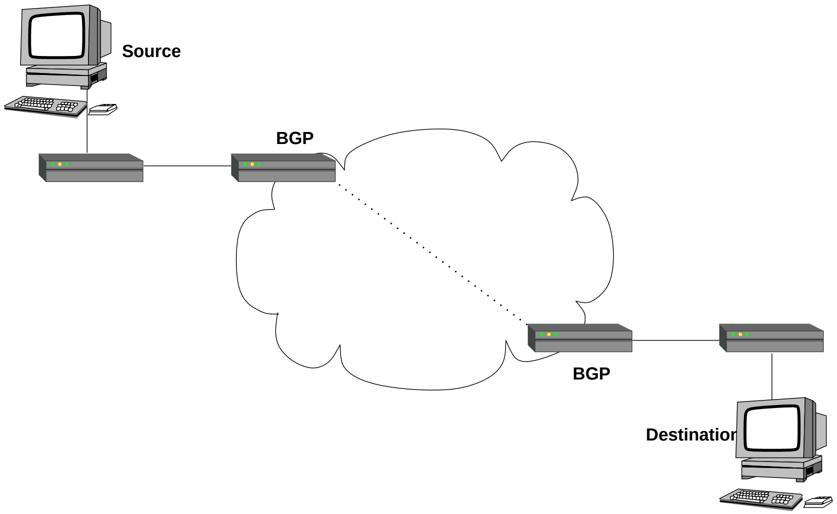
Single-homed Autonomous Systems

- Single homed AS, or *stub* AS
  - An AS has only one exit point to outside networks. Quite often, a single-homed AS is referred as a *stub network*.
- An ISP can use three different methods to advertise a customer's network, a single-homed AS, so that the Internet community can learn about such a network.
  - Using static/default routes
  - Using IGP, such as OSPF and RIP
  - Using EGP, such as BGP
- In most cases, simple static routes are used.
- BGP is not commonly used due to the difficulty stub networks have with getting a registered AS number.

Multi-homed Non-transit AS

- An AS is a multi-homed system if it has more than one exit point to the outside networks. An AS connected to the Internet can be multi-homed to a single ISP or multiple ISPs.
- Non-transit refers to the fact that transit traffic does not pass through the AS. A non-transit AS would advertise only its own routes to the ISPs to which it connects, it would not advertise routes that it learned from one ISP to another.
- A multi-homed Non-transit AS does not really need to run BGP with their ISPs. Other routing methods can be used instead. However, some ISPs may prefer the customers to use BGP.

Multi-homed Transit AS



Multi-homed Transit AS

- A multi-homed transit AS can be used for transit traffic of other autonomous systems. BGP can be used internally so that multiple border routers in the same AS can share BGP information.
- iBGP is run inside the AS. Routers that route iBGP traffic are *transit routers*.
- eBGP is run between the local and the external ASs. Routers on the boundary of an AS that use eBGP to exchange information with the ISP are *border* (or *edge*) routers.

BGP: to use or not to use

- If the routing policy of an AS is consistent with the ISP's policy, it is not necessary to use BGP to exchange routing information with the ISP. If the AS and ISP's policy are different, BGP is preferred.
- If the AS uses different ISPs for redundancy, (or load sharing) a combination of static and default routes could be used instead of BGP.
- If the AS uses multiple connections to ISPs that are active at the same time, BGP is preferred.

BGP Attributes

BGP

- BPG is designed to be used on the Internet. Many route parameters, called *attributes*, can be used with BGP so that better routing policies are provided.
- BGP supports CIDR which helps reduce the routing table size.
- BGP packets are carried through TCP connection. When two neighbor routers wish to exchange BGP route information, a TCP connection is established first.
- BGP routers do not send periodic updates. Full routing information are exchanged when the TCP connection is first established, afterward, only *changed* routes will be advertised. Also, only the *optimal path* (i.e. there are alternate paths) to a destination network is advertised through routing updates.

BGP Attributes

- Routes learned via BGP have associated properties that are used to determine the best route to a destination when multiple paths exist. These properties are referred to as BGP attributes.
- The following BGP attributes can be used to determine the best path:
  - Weight (Cisco proprietary, highest priority)
  - Local Preference
  - AS Path
  - Origin
  - Multi-Exit Discriminator (lowest priority)

BGP Weight Attribute

- *Weight* is a Cisco-defined attribute that is local to a router. The *weight* attribute is not advertised to neighboring routers.
- If the router learns about more than one route to the same destination, the route with the highest *weight* will be preferred.
- When there are two routes/paths to a destination, both will be maintained in the BGP routing table. However, only the route with the highest *weight* will be installed in the IP routing table. That is, when forwarding IP packets, the route with the highest *weight* is used.

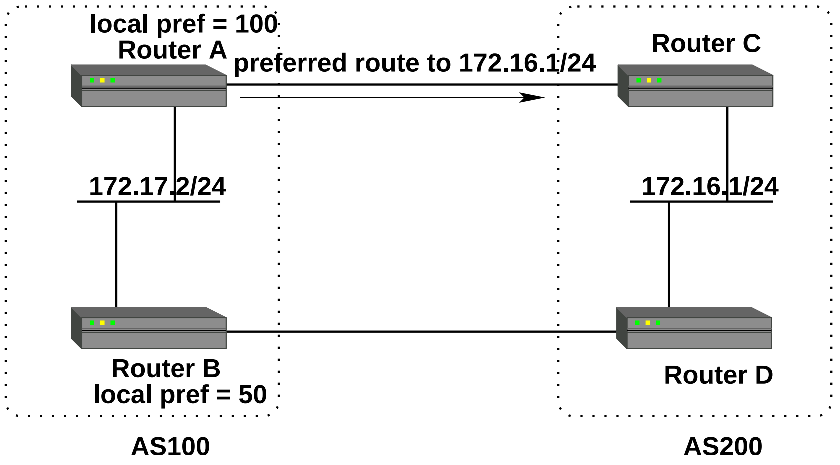
1Preferring One Link381

37-1

BGP Local Preference Attribute

- The *local preference* attribute is used to prefer an exit point from the local autonomous system AS. If there are multiple exit points from the AS, the *local preference* attribute is used to select the exit point for a specific route.
- For example, two routers (A & B) connect a local AS100 to another AS200, and both routers receive route advertisement for a particular network 10.0.0.0/8. If router A is set a *local preference* value of 50 while router B is set a value of 55, the route through router B will be used to forward traffic from local AS to the particular network 10.0.0.0/8.
- *Weight* attribute is similar to the *local preference* attribute in that they are used to set an outgoing path. Their difference is that *weight* attribute is local to a router while *local preference* attribute is propagated throughout the

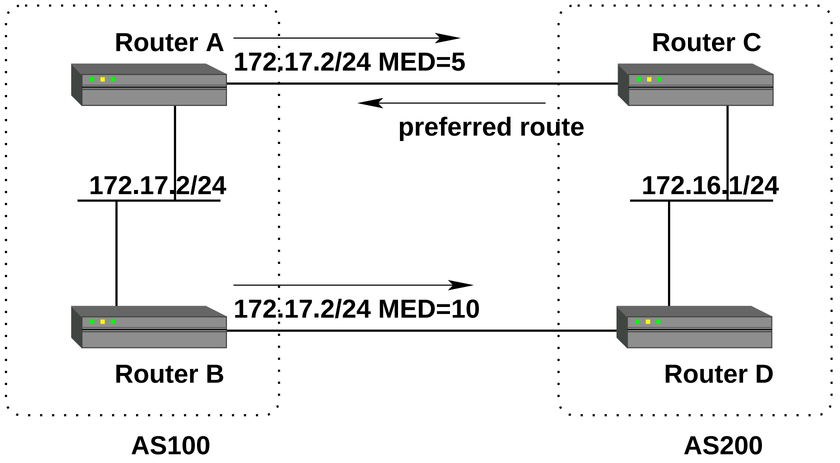
BGP LOCAL_PREF



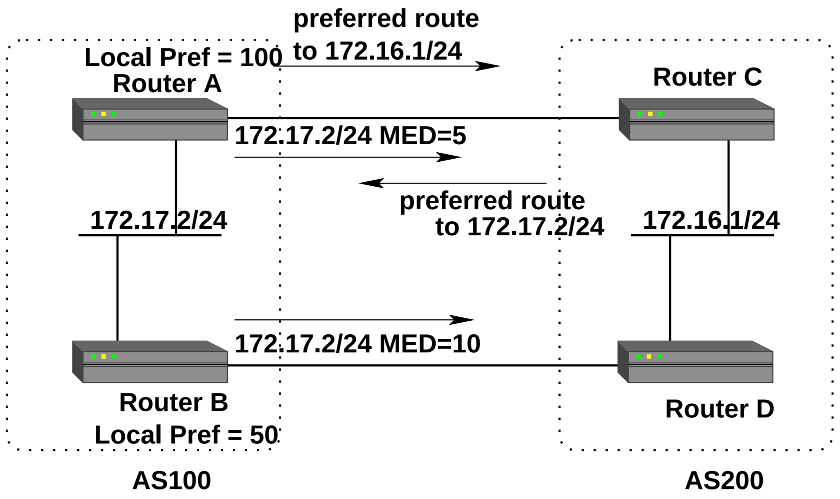
BGP MED Attribute

- The *multi-exit discriminator* (MED) is used to suggest an external AS regarding the preferred route into the local AS that is advertising the route.
- The external AS, which receive the MEDs, may not take the “suggestion” and may use other BGP attributes for route selection.
- MEDs are advertised throughout the local AS.

BGP MULTI_EXIT_DISC



BGP: Selecting one Link



BGP AS_path Attribute

- When a route advertisement passes through an autonomous system, the AS number is added to an ordered list of AS numbers that the route advertisement has traversed.
- The *AS_path* attribute can be used to detect routing loops.
  - If a router receives a route advertisement with an ordered list containing an AS number the same as the AS that the router belongs to, it ignores the route advertisement.
- The *AS_path* attribute can be used to select the better path.
  - The route that contains the shortest *AS_path* (i.e. the order list that contains the shortest list of AS numbers) is selected.

43-1

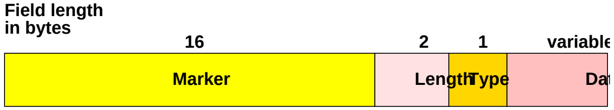
1BGP Messages441

BGP Message Types

- Four BGP message types are specified in RFC 1771 (i.e. BGP version 4).
- The *Open Message* opens a BGP communication session between peers and is the first message sent by each side after a TCP connection is established.
- The *Update Message* is used to provide routing updates to other BGP systems, allowing routers to construct a consistent view of the network topology. Update messages can withdraw one or more unfeasible routes from the routing table and simultaneously can advertise a route.
- The *Notification Message* is sent when error condition is detected. Notifications are used to close an active session.
- The *Keep-alive Message* notifies BGP peers that a device is active

BGP Packet Formats

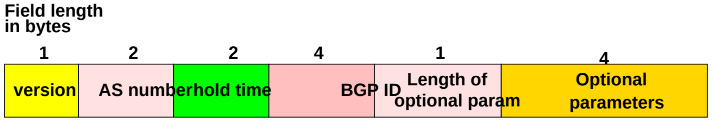
- All BGP message types use the *basic packet header*



- The basic packet header contains:
  - a 16-byte *marker* field which contains authentication value
  - a 2-byte *length* field which contains the total length of the message
  - a 1-byte *type* field which specifies the message type
  - data of variable length, this field carry the upper-layer information

Additional Fields: Open Message

- Open, update and notification messages have additional fields, but keep-alive messages use only the basic packet header.
- Additional fields of the *Open Message* contains:
  - BGP *version number* (i.e., 4)
  - *AS number* of sender
  - *hold-time*
  - BGP *identifier* of the sender (IP address)
  - *optional parameters* such as authentication data.

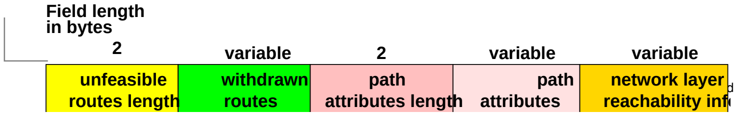


Field length in bytes

1	2	2	4	1	4
version	AS number	hold time	BGP ID	Length of optional param	Optional parameters

BGP Additional Fields: Update Message

- Additional fields of *Update Message* contains:
  - *Withdrawn routes*: a list of IP address prefixes for the routes being withdrawn
  - *Network layer reachability* information: a list of IP address prefixes (e.g. 10.1.1.0/24) for the advertised routes
  - *Path attributes* (such as origin, AS_path, MED, LOCAL_PREF, . . .) that describe the characteristics of the advertised path.
  - *Unfeasible routes length*, i.e., length of withdrawn routes field
  - *Total path attribute length*, i.e., length of the path attributes field

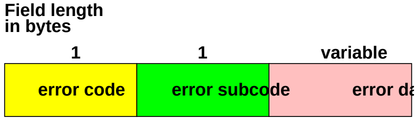


Field length in bytes

2	variable	2	variable	variable
unfeasible routes length	withdrawn routes	path attributes length	path attributes	network layer reachability inf

GP Additional Fields: Notification Message

- Additional fields of *Notification Message* contains:
  - *Error code* that indicates the type of error that occurred.
  - Error *sub code*
  - *error data*.



Field length in bytes

1	1	variable
error code	error subcode	error da