Using BGP Communities

ISP Workshops

Last updated 9th December 2015
Multihoming and Communities

- The BGP community attribute is a very powerful tool for assisting and scaling BGP Policies and BGP Multihoming.
- Most major ISPs make extensive use of BGP communities:
  - Internal policies
  - Inter-provider relationships (MED replacement)
  - Customer traffic engineering
Using BGP Communities

Four scenarios are covered:

- Use of RFC1998 traffic engineering
- Extending RFC 1998 ideas for even greater customer policy options
- Community use in ISP backbones
- Customer Policy Control (aka traffic engineering)
RFC1998

An example of how ISPs use communities...
RFC1998

- Informational RFC
- Describes how to implement loadsharing and backup on multiple inter-AS links
  - BGP communities used to determine local preference in upstream’s network
- Gives control to the customer
  - Means the customer does not have to phone upstream’s technical support to adjust traffic engineering needs
- Simplifies upstream’s configuration
  - Simplifies network operation!
RFC1998

- RFC1998 Community values are defined to have particular meanings

- ASx:100 set local preference 100
  - Make this the preferred path

- ASx :90 set local preference 90
  - Make this the backup if dualhomed on ASx

- ASx :80 set local preference 80
  - The main link is to another ISP with same AS path length

- ASx :70 set local preference 70
  - The main link is to another ISP
Upstream ISP defines the communities mentioned. Their customers then attach the communities they want to use to the prefix announcements they are making.

For example:
- If upstream is AS 100
- To declare a particular path as a backup path, their customer would announce the prefix with community 100:70 to AS100
- AS100 would receive the prefix with the community 100:70 tag, and then set local preference to be 70
Sample Customer Router Configuration

```plaintext
router bgp 130
    neighbor x.x.x.x remote-as 100
    neighbor x.x.x.x description Backup ISP
    neighbor x.x.x.x route-map as100-out out
    neighbor x.x.x.x send-community

! ip as-path access-list 20 permit ^$ 
!
route-map as100-out permit 10
    match as-path 20
    set community 100:70
! 
```
Sample ISP Router Configuration

router bgp 100

neighbor y.y.y.y remote-as 130
neighbor y.y.y.y route-map customer-policy-in in

! Homed to another ISP
ip community-list 7 permit 100:70

! Homed to another ISP with equal AS_PATH length
ip community-list 8 permit 100:80

! Customer backup routes
ip community-list 9 permit 100:90

route-map customer-policy-in permit 10
  match community 7
  set local-preference 70
!
route-map customer-policy-in permit 20
  match community 8
  set local-preference 80
!
route-map customer-policy-in permit 30
  match community 9
  set local-preference 90
!
route-map customer-policy-in permit 40
  set local-preference 100
!
RFC1998 was the inspiration for a large variety of differing community policies implemented by ISPs worldwide.

There are no “standard communities” for what ISPs do.

But best practices today consider that ISPs should use BGP communities extensively for multihoming support of traffic engineering.

Look in the ISP AS Object in the IRR for documented community support.
RFC1998 Example

Two links to the same ISP, one link primary, the other link backup
Two links to the same ISP

- AS100 proxy aggregates for AS 65534
Two links to the same ISP
(one as backup only)

- Announce /19 aggregate on each link
  - primary link makes standard announcement
  - backup link sends community

- When one link fails, the announcement of the /19 aggregate via the other link ensures continued connectivity
Two links to the same ISP
(one as backup only)

- **Router A Configuration**
  
  ```
  router bgp 65534
  network 121.10.0.0 mask 255.255.224.0
  neighbor 122.102.10.2 remote-as 100
  neighbor 122.102.10.2 description RouterC
  neighbor 122.102.10.2 prefix-list aggregate out
  neighbor 122.102.10.2 prefix-list default in
  
  !
  ip prefix-list aggregate permit 121.10.0.0/19
  ip prefix-list default permit 0.0.0.0/0
  !
  ```
Two links to the same ISP
(one as backup only)

- **Router B Configuration**
  ```
  router bgp 65534
  network 121.10.0.0 mask 255.255.224.0
  neighbor 122.102.10.6 remote-as 100
  neighbor 122.102.10.6 description RouterD
  neighbor 122.102.10.6 send-community
  neighbor 122.102.10.6 prefix-list aggregate out
  neighbor 122.102.10.6 route-map routerD-out out
  neighbor 122.102.10.6 prefix-list default in
  neighbor 122.102.10.6 route-map routerD-in in
  ```
  ..next slide
Two links to the same ISP (one as backup only)

```
ip prefix-list aggregate permit 121.10.0.0/19
ip prefix-list default permit 0.0.0.0/0
!
route-map routerD-out permit 10
  match ip address prefix-list aggregate
  set community 100:90
route-map routerD-out permit 20
!
route-map routerD-in permit 10
  set local-preference 90
!```
Two links to the same ISP (one as backup only)

- Router C Configuration (main link)
  
  ```
  router bgp 100
  neighbor 122.102.10.1 remote-as 65534
  neighbor 122.102.10.1 default-originate
  neighbor 122.102.10.1 prefix-list Customer in
  neighbor 122.102.10.1 prefix-list default out

  ip prefix-list Customer permit 121.10.0.0/19
  ip prefix-list default permit 0.0.0.0/0
  ```
Two links to the same ISP (one as backup only)

- Router D Configuration (backup link)
  - router bgp 100
    - neighbor 122.102.10.5 remote-as 65534
    - neighbor 122.102.10.5 default-originate
    - neighbor 122.102.10.5 prefix-list Customer in
    - neighbor 122.102.10.5 route-map bgp-cust-in in
    - neighbor 122.102.10.5 prefix-list default out
  
  - !
  - ip prefix-list Customer permit 121.10.0.0/19
  - ip prefix-list default permit 0.0.0.0/0
  
  ..next slide
Two links to the same ISP (one as backup only)

```
! 
ip community-list 90 permit 100:90 
! <snip> 
route-map bgp-cust-in permit 30   
  match community 90   
  set local-preference 90   
route-map bgp-cust-in permit 40   
  set local-preference 100   
! 
```
Two links to the same ISP (one as backup only)

- This is a simple example
- It looks more complicated than the same example presented earlier which used local preference and MEDs
- But the advantage is that this scales better
  - With larger configurations, more customers, more options, it becomes easier to handle each and every requirement
Service Provider use of Communities

RFC1998 was so inspiring...
Background

- RFC1998 is okay for “simple” multihoming situations
- ISPs create backbone support for many other communities to handle more complex situations
  - Simplify ISP BGP configuration
  - Give customer more policy control
ISP BGP Communities

- There are no recommended ISP BGP communities apart from:
  - RFC1998
  - The five standard communities:
    - [www.iana.org/assignments/bgp-well-known-communities](http://www.iana.org/assignments/bgp-well-known-communities)
- Efforts have been made to document from time to time:
  - But so far... nothing more... 😞
  - Collection of ISP communities at [www.onesc.net/communities](http://www.onesc.net/communities)
  - NANOG Tutorial: [www.nanog.org/meetings/nanog40/presentations/BGPcommunities.pdf](http://www.nanog.org/meetings/nanog40/presentations/BGPcommunities.pdf)
- ISP policy is usually published:
  - On the ISP’s website
  - Referenced in the AS Object in the IRR
Typical ISP BGP Communities

- **X:80** set local preference 80
  - Backup path

- **X:120** set local preference 120
  - Primary path (over ride BGP path selection default)

- **X:1** set as-path prepend X
  - Single prepend when announced to X’s upstreams

- **X:2** set as-path prepend X X
  - Double prepend when announced to X’s upstreams

- **X:3** set as-path prepend X X X
  - Triple prepend when announced to X’s upstreams

- **X:666** set ip next-hop 192.0.2.1
  - Blackhole route – very useful for DoS attack mitigation
Sample Router Configuration (1)

```plaintext
router bgp 100
  neighbor y.y.y.y remote-as 130
  neighbor y.y.y.y route-map customer-policy-in in
  neighbor z.z.z.z remote-as 200
  neighbor z.z.z.z route-map upstream-out out
!
ip community-list 1 permit 100:1
ip community-list 2 permit 100:2
ip community-list 3 permit 100:3
ip community-list 4 permit 100:80
ip community-list 5 permit 100:120
ip community-list 6 permit 100:666
!
ip route 192.0.2.1 255.255.255.255 null0
```
Sample Router Configuration (2)

```plaintext
route-map customer-policy-in permit 10
  match community 4
  set local-preference 80
!
route-map customer-policy-in permit 20
  match community 5
  set local-preference 120
!
route-map customer-policy-in permit 30
  match community 6
  set ip next-hop 192.0.2.1
!
route-map customer-policy-in permit 40
...etc...
```
Sample Router Configuration (3)

```plaintext
route-map upstream-out permit 10
    match community 1
    set as-path prepend 100

route-map upstream-out permit 20
    match community 2
    set as-path prepend 100 100

route-map upstream-out permit 30
    match community 3
    set as-path prepend 100 100 100

route-map upstream-out permit 40
    ...etc...
```
### ISP Example: Sprint

**WHAT YOU CAN CONTROL**

**AS-PATH PREPENDS**
Sprint allows customers to use AS-path prepending to adjust route preference on the network. Such prepending will be received and passed on properly without notifying Sprint of your change in announcements.

Additionally, Sprint will prepend AS1239 to eBGP sessions with certain autonomous systems depending on a received community. Currently, the following ASes are supported: 1668, 209, 2914, 3300, 3356, 3549, 3561, 4635, 701, 7018, 702 and 8220.

#### String | Resulting AS Path to ASXXX
---|---
65000:0XX | Do not advertise to ASXXX
65001:0XX | 1239 (default) ...
65002:0XX | 1239 1239 ...
65003:0XX | 1239 1239 1239 ...
65004:0XX | 1239 1239 1239 1239 ...

#### String | Resulting AS Path to ASXXX in Asia
---|---
65070:0XX | Do not advertise to ASXXX
65071:0XX | 1239 (default) ...
65072:0XX | 1239 1239 ...
65073:0XX | 1239 1239 1239 ...
65074:0XX | 1239 1239 1239 1239 ...

#### String | Resulting AS Path to ASXXX in Europe
---|---
65050:0XX | Do not advertise to ASXXX
65051:0XX | 1239 (default) ...
65052:0XX | 1239 1239 ...
65053:0XX | 1239 1239 1239 ...
65054:0XX | 1239 1239 1239 1239 ...

#### String | Resulting AS Path to ASXXX in North America
---|---
65010:0XX | Do not advertise to ASXXX

ISP Example: NTT

More info at www.us.ntt.net/about/policy/routing.cfm
### ISP Example:
#### Verizon Europe

<table>
<thead>
<tr>
<th>aut-num:</th>
<th>AS702</th>
</tr>
</thead>
<tbody>
<tr>
<td>descr:</td>
<td>Verizon Business EMEA - Commercial IP service provider in Europe</td>
</tr>
<tr>
<td>remarks:</td>
<td>Verizon Business filters out inbound prefixes longer than /24. We also filter any networks within AS702:RS-INBOUND-FILTER.</td>
</tr>
</tbody>
</table>

VzBi uses the following communities with its customers:

- **702:80** Set Local Pref 80 within AS702
- **702:120** Set Local Pref 120 within AS702
- **702:20** Announce only to VzBi AS'es and VzBi customers
- **702:30** Keep within Europe, don't announce to other VzBi AS's
- **702:1** Prepend AS702 once at edges of VzBi to Peers
- **702:2** Prepend AS702 twice at edges of VzBi to Peers
- **702:3** Prepend AS702 thrice at edges of VzBi to Peers

Advanced communities for customers:

- **702:7020** Do not announce to AS702 peers with a scope of National but advertise to Global Peers, European Peers and VzBi customers.
- **702:7001** Prepend AS702 once at edges of VzBi to AS702 peers with a scope of National.
- **702:7002** Prepend AS702 twice at edges of VzBi to AS702 peers with a scope of National.

And many more!
ISP Example:
Telia

aut-num: AS1299
descr: TeliaSonera International Carrier

rem: BGP COMMUNITY SUPPORT FOR AS1299 TRANSIT CUSTOMERS:
rem: Community Action (default local pref 200)

rem: 1299:50 Set local pref 50 within AS1299 (lowest possible)
rem: 1299:150 Set local pref 150 within AS1299 (equal to peer, backup)

rem: European peers
rem: Community Action

rem: 1299:200x All peers Europe incl:
rem: 1299:250x Sprint/1239
rem: 1299:251x Savvis/3561
rem: 1299:252x NTT/2914
rem: 1299:253x Zayo/Abovenet/6461
rem: 1299:254x FT/5511
rem: 1299:255x GBLX/3549
rem: 1299:256x Level3/3356

rem: Where x is number of prepends (x=0,1,2,3) or do NOT announce (x=9)

And many many more!
ISP Example:
BT Ignite

aut-num: AS5400
descr: BT Ignite European Backbone

<snip>

remarks: The following BGP communities can be set by BT
remarks: BGP customers to affect announcements to major peers.
remarks: 5400:NXXX
remarks: N=1  not announce
remarks: N=2  prepend an extra "5400 5400" on announcement
remarks:  Valid values for XXX:
remarks: 000  All peers and transits
remarks: 500  All transits
remarks: 503  Level3 AS3356
remarks: 509  Telia AS1299
remarks: 510  NTT Verio AS2914
remarks: 002  Sprint AS1239
remarks: 003  Savvis AS3561
remarks: 004  C&W AS1273
remarks: 005  Verizon EMEA AS702
remarks: 014  DTAG AS3320
remarks: 016  Opentransit AS5511
remarks: 018  GlobeInternet Tata AS6453
remarks: 023  Tinet AS3257
remarks: 027  Telia AS1299
remarks: 045  Telecom Italia AS6762
remarks: 073  Eurorings AS286
remarks: 169  Cogent AS174

<snip>
ISP Example:
Level3

aut-num:      AS3356
descr:        Level 3 Communications
 remarks:      customer traffic engineering communities - Suppression
 remarks:      64960:XXX - announce to AS XXX if 65000:0
 remarks:      65000:0   - announce to customers but not to peers
 remarks:      65000:XXX - do not announce at peerings to AS XXX
 remarks:      customer traffic engineering communities - Prepending
 remarks:      65001:0   - prepend once  to all peers
 remarks:      65001:XXX - prepend once  at peerings to AS XXX
 remarks:      65002:0   - prepend twice to all peers
 remarks:      65002:XXX - prepend twice at peerings to AS XXX
 remarks:      customer traffic engineering communities - LocalPref
 remarks:      3356:70   - set local preference to 70
 remarks:      3356:80   - set local preference to 80
 remarks:      3356:90   - set local preference to 90
 remarks:      customer traffic engineering communities - Blackhole
 remarks:      3356:9999 - blackhole (discard) traffic
 remarks:      And many more!
Creating your own community policy

- Consider creating communities to give policy control to customers
  - Reduces technical support burden
  - Reduces the amount of router reconfiguration, and the chance of mistakes
  - Use previous ISP and configuration examples as a guideline
Using Communities for Backbone Scaling

Scaling BGP in the ISP backbone...
Communities for iBGP

- ISPs tag prefixes learned from their BGP and static customers with communities
  - To identify services the customer may have purchased
  - To identify prefixes which are part of the ISP’s PA space
  - To identify PI customer addresses
  - To control prefix distribution in iBGP
  - To control prefix announcements to customers and upstreams
  - (amongst several other reasons)
Service Identification

- ISP provides:
  - Transit via upstreams
  - Connectivity via major IXP
  - Connectivity to private peers/customers
- Customers can buy all or any of the above access options
  - Each option is identified with a unique community
- ISP identifies whether address space comes from their PA block or is their customers’ own PI space
  - One community for each
## Community Definitions

<table>
<thead>
<tr>
<th>Community</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100:1000</td>
<td>AS100 aggregates</td>
</tr>
<tr>
<td>100:1001</td>
<td>AS100 aggregate subprefixes</td>
</tr>
<tr>
<td>100:1005</td>
<td>Static Customer PI space</td>
</tr>
<tr>
<td>100:2000</td>
<td>Customers who get Transit</td>
</tr>
<tr>
<td>100:2100</td>
<td>Customers who get IXP access</td>
</tr>
<tr>
<td>100:2200</td>
<td>Customers who get BGP Customer access</td>
</tr>
<tr>
<td>100:3000</td>
<td>Routes learned from the IXP</td>
</tr>
</tbody>
</table>

```bash
ip community-list 10 permit 100:1000
ip community-list 11 permit 100:1001
ip community-list 12 permit 100:1005
ip community-list 13 permit 100:2000
ip community-list 14 permit 100:2100
ip community-list 15 permit 100:2200
ip community-list 16 permit 100:3000
```
Aggregates and Static Customers into BGP

router bgp 100
  network 100.10.0.0 mask 255.255.224.0 route-map as100-prefixes
  redistribute static route-map static-to-bgp

! ip prefix-list as100-block permit 100.10.0.0/19 le 32
!
route-map as100-prefixes permit 10
  set community 100:1000
!
route-map static-to-bgp permit 10
  match ip address prefix-list as100-block
  set community 100:1001
route-map static-to-bgp permit 20
  set community 100:1005

---

Aggregate community set
Aggregate subprefixes community set
PI community is set
Service Identification

- AS100 has four classes of BGP customers
  - Full transit (upstream, IXP and BGP customers)
  - Upstream only
  - IXP only
  - BGP Customers only

- For BGP support, easiest IOS configuration is to create a peer-group for each class (can also use peer-templates to simplify further)
  - Customer is assigned the peer-group of the service they have purchased
  - Simple for AS100 customer installation engineer to provision
BGP Customers – creating peer-groups

```
router bgp 100
  neighbor full-transit peer-group
  neighbor full-transit route-map customers-out out
  neighbor full-transit route-map full-transit-in in
  neighbor full-transit default-originate

neighbor transit-up peer-group
  neighbor transit-up route-map customers-out out
  neighbor transit-up route-map transit-up-in in
  neighbor transit-up default-originate

neighbor ixp-only peer-group
  neighbor ixp-only route-map ixp-routes out
  neighbor ixp-only route-map ixp-only-in in

neighbor bgpcust-only peer-group
  neighbor bgpcust-only route-map bgp-cust-out out
  neighbor bgpcust-only route-map bgp-cust-in in
```
BGP Customers – creating route-maps

route-map customers-out permit 10
  match ip community 10
route-map full-transit-in permit 10
  set community 100:2000 100:2100 100:2200
route-map transit-up-in permit 10
  set community 100:2000
route-map ixp-routes permit 10
  match ip community 10 12 13 14 16
route-map ixp-only-in permit 10
  set community 100:2100
route-map bgp-cust-out permit 10
  match ip community 10 12 13 15
route-map bgp-cust-in permit 10
  set community 100:2200

Customers only get AS100 aggregates and default route

Full transit go everywhere

Customers buying IXP access only get aggregates, static & full transit customers and IXP routes

Customers buying BGP customer access only get aggregates, static & full transit customers and other BGP customers
BGP Customers – configuring customers

```plaintext
router bgp 100
  neighbor a.a.a.a remote-as 200
  neighbor a.a.a.a peer-group full-transit
  neighbor a.a.a.a prefix-list as200cust-in
  neighbor b.b.b.b remote-as 300
  neighbor b.b.b.b peer-group transit-up
  neighbor b.b.b.b prefix-list as300cust-in
  neighbor c.c.c.c remote-as 400
  neighbor c.c.c.c peer-group ixp-only
  neighbor c.c.c.c prefix-list as400cust-in
  neighbor d.d.d.d remote-as 500
  neighbor d.d.d.d peer-group bgpcust-only
  neighbor d.d.d.d prefix-list as500cust-in
```

Customers are placed into the appropriate peer-group depending on the service they paid for.

Note the specific per-customer inbound filters
BGP Customers – configuring upstream

```
router bgp 100
    neighbor x.x.x.x remote-as 130
    neighbor x.x.x.x prefix-list full-routes in
    neighbor x.x.x.x route-map upstream-out out

route-map upstream-out permit 10
    match ip community 10 12 13

! IP prefix-list full-routes is the standard bogon
! prefix filter - or use a reputable bogon
! route-service such as that offered by Team Cymru
```
BGP Customers – configuring IXP peers

router bgp 100
    neighbor y.y.y.1 remote-as 901
    neighbor y.y.y.1 route-map ixp-peers-out out
    neighbor y.y.y.1 route-map ixp-peers-in in
    neighbor y.y.y.1 prefix-list AS901-peer in
    neighbor y.y.y.2 remote-as 902
    neighbor y.y.y.2 route-map ixp-peers-out out
    neighbor y.y.y.2 route-map ixp-peers-in in
    neighbor y.y.y.2 prefix-list AS902-peer in

route-map ixp-peers-out permit 10
    match ip community 10 12 13 14

route-map ixp-peers-in permit 10
    set community 100:3000

Aggregates, PI customers full transit and IXP customers are announced to the IXP
While the community set up takes a bit of thought and planning, once it is implemented:

- eBGP configuration with customers is simply a case of applying the appropriate peer-group
- eBGP configuration with IXP peers is simply a case of announcing the appropriate community members to the peers
- eBGP configuration with upstreams is simply a case of announcing the appropriate community members to the upstreams

All BGP policy internally is now controlled by communities

- No prefix-lists, as-path filters, route-maps or other BGP gymnastics are required
What about iBGP itself?

- We’ve made good use of communities to handle customer requirements
  - But what about iBGP
- Most ISPs deploy Route Reflectors as a means of scaling iBGP
- In transit networks:
  - Core routers (the Route Reflectors) carry the full BGP table
  - Edge/Aggregation routers carry domestic prefixes & customers
iBGP core router/route reflector

router bgp 100
neighbor rrc peer-group
neighbor rrc descr Route Reflector Clients
neighbor rrc remote-as 100
neighbor rrc route-reflector-client
neighbor rrc route-map ibgp-filter out
neighbor rrc send-community
neighbor ibgp-peer peer-group
neighbor ibgp-peer Standard iBGP peers
neighbor ibgp-peer remote-as 100
neighbor ibgp-peer send-community
neighbor n.n.n.a peer-group ibgp-peer
neighbor n.n.n.b peer-group rrc
!
route-map ibgp-filter permit 10
  match community 10 11 12 13 14 15 16
!

The filter to restrict client iBGP to just domestic prefixes

Must NOT forget to send community to iBGP peers

Allow all prefixes coming from the domestic network & IXP
iBGP in the core

- Notice that the filtering of iBGP from the core to the edge is again achieved by a simple route-map applying a community match
  - No prefix-lists, as-path filters or any other complicated policy
  - Once the prefix belongs to a certain community, it has the access across the backbone determined by the community policy in force
Using Communities for Customers Policy

Giving policy control to customers...
Customer Policy Control

- ISPs have a choice on how to handle policy control for customers
- No delegation of policy options:
  - Customer has no choices
  - If customer wants changes, ISP Technical Support handles it
- Limited delegation of policy options:
  - Customer has choices
  - ISP Technical Support does not need to be involved
- BGP Communities are the only viable way of offering policy control to customers
Policy Definitions

- Typical definitions:
  - Nil: No community set, just announce everywhere
  - X:1: 1x prepend to all BGP neighbours
  - X:2: 2x prepend to all BGP neighbours
  - X:3: 3x prepend to all BGP neighbours
  - X:80: Local pref 80 on customer prefixes
  - X:120: Local pref 120 on customer prefixes
  - X:666: Black hole this route please!
  - X:5000: Don’t announce to any BGP neighbour
  - X:5AA0: Don’t announce to BGP neighbour AA
  - X:5AAB: Prepend B times to BGP neighbour AA
Policy Implementation

- The BGP configuration for the initial communities was discussed at the start of this slide set.
- But the new communities, X:5MMN, are worth covering in more detail.
  - The ISP in AS X documents the BGP transits and peers that they have (MM can be 01 to 99).
  - The ISP in AS X indicates how many prepends they will support (N can be 1 to 9, but realistically 4 prepends is usually enough on today’s Internet).
  - Customers then construct communities to do the prepending or announcement blocking they desire.
- If a customer tags a prefix announcement with:
  - 100:5030 don’t send prefix to BGP neighbour 03
  - 100:5102 2x prepend prefix announcement to peer 10
Community Definitions

- Example: ISP in AS 100 has two upstreams. They create policy based on previously slide to allow no announce and up to 3 prepends for their customers.

```plaintext
ip community-list 100 permit 100:5000  # Don’t announce anywhere
ip community-list 101 permit 100:5001  # Single prepend to all
ip community-list 102 permit 100:5002
ip community-list 103 permit 100:5003
ip community-list 110 permit 100:5010  # Don’t announce to peer 1
ip community-list 111 permit 100:5011
ip community-list 112 permit 100:5012
ip community-list 113 permit 100:5013
ip community-list 120 permit 100:5020
ip community-list 121 permit 100:5021  # Single prepend to peer 2
ip community-list 122 permit 100:5022
ip community-list 123 permit 100:5023
```
Creating route-maps – neighbour 1

route-map bgp-neigh-01 deny 10
match ip community 100 110
!
route-map bgp-neigh-01 permit 20
match ip community 101 111
set as-path prepend 100
!
route-map bgp-neigh-01 permit 30
match ip community 102 112
set as-path prepend 100 100
!
route-map bgp-neigh-01 permit 40
match ip community 103 113
set as-path prepend 100 100 100
!
route-map bgp-neigh-01 permit 50

Don’t announce these prefixes to neighbour 01
Single prepend of these prefixes to neighbour 01
Double prepend of these prefixes to neighbour 01
Triple prepend of these prefixes to neighbour 01
All other prefixes remain untouched
Creating route-maps – neighbour 2

route-map bgp-neigh-02 deny 10
   match ip community 100 120
!
route-map bgp-neigh-02 permit 20
   match ip community 101 121
   set as-path prepend 100
!
route-map bgp-neigh-02 permit 30
   match ip community 102 122
   set as-path prepend 100 100
!
route-map bgp-neigh-02 permit 40
   match ip community 103 123
   set as-path prepend 100 100 100
!
route-map bgp-neigh-02 permit 50

Don’t announce these prefixes to neighbour 02
Single prepend of these prefixes to neighbour 02
Double prepend of these prefixes to neighbour 02
Triple prepend of these prefixes to neighbour 02
All other prefixes remain untouched
ISP’s BGP configuration

router bgp 100
neighbor a.a.a.a remote-as 200
neighbor a.a.a.a route-map bgp-neigh-01 out
neighbor a.a.a.a route-map policy-01 in
neighbor b.b.b.b remote-as 300
neighbor b.b.b.b route-map bgp-neigh-02 out
neighbor b.b.b.b route-map policy-02 in

- The route-maps are then applied to the appropriate neighbour
- As long as the customer sets the appropriate communities, the policy will be applied to their prefixes
Customer BGP configuration

```
router bgp 600
    neighbor c.c.c.c remote-as 100
    neighbor c.c.c.c route-map upstream out
    neighbor c.c.c.c prefix-list default in

! route-map upstream permit 10
    match ip address prefix-list blockA
    set community 100:5010 100:5023
route-map upstream permit 20
    match ip address prefix-list aggregate
```

- This will:
  - 3x prepend of blockA towards their upstream’s 2nd BGP neighbour
  - Not announce blockA towards their upstream’s 1st BGP neighbour
  - Let the aggregate through with no specific policy
Customer Policy Control

- Notice how much flexibility a BGP customer could have with this type of policy implementation

- Advantages:
  - Customer has flexibility
  - ISP Technical Support does not need to be involved

- Disadvantages
  - Customer could upset ISP loadbalancing tuning

- Advice
  - This kind of policy control is very useful, but should only be considered if appropriate for the circumstances
Conclusion
Communities

- Communities are fun! 😊
- And they are extremely powerful tools
- Think about community policies, e.g. like the additions described here
- Supporting extensive community usage makes customer configuration easy
- Watch out for routing loops!
Using BGP Communities

ISP Workshops