Using BGP Communities

ISP/IXP Workshops

These materials are licensed under the Creative Commons Attribution-NonCommercial 4.0 International license (http://creativecommons.org/licenses/by-nc/4.0/)

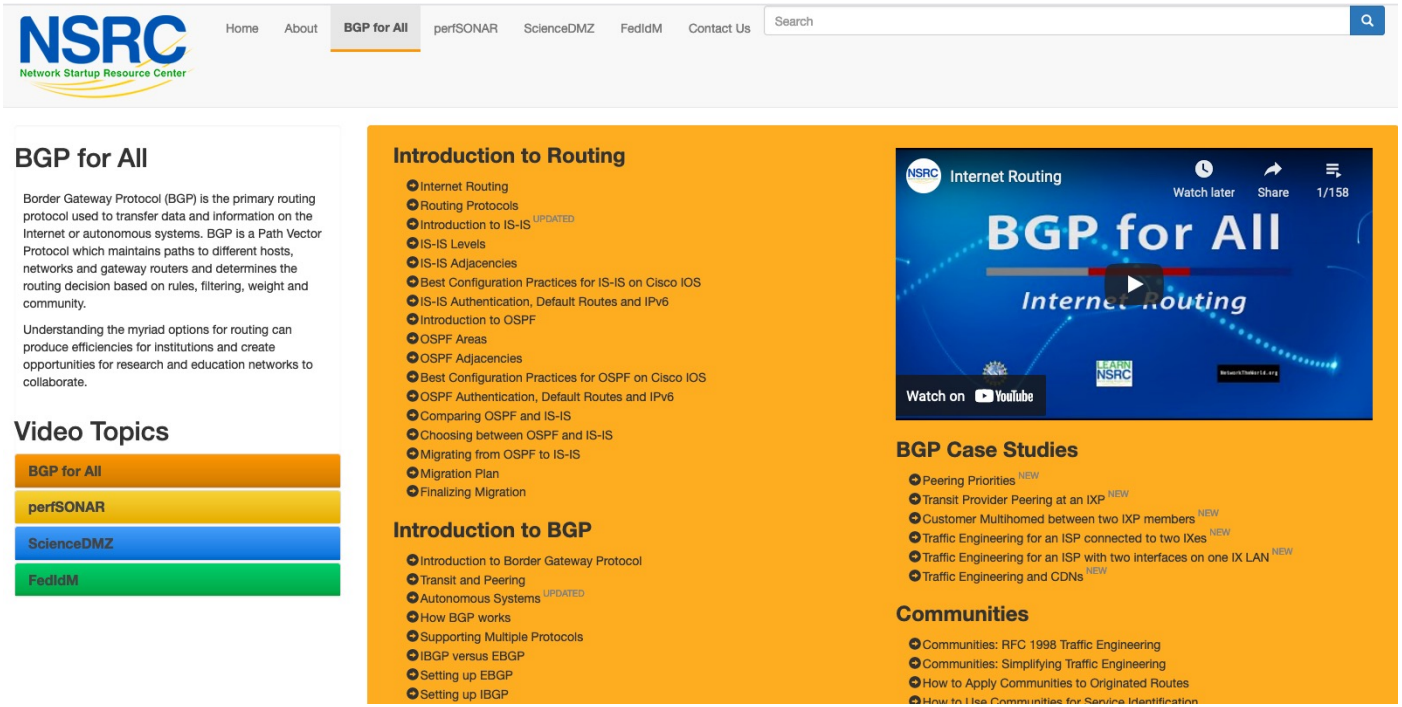
Acknowledgements

- This material originated from the Cisco ISP/IXP Workshop Programme developed by Philip Smith & Barry Greene
- Use of these materials is encouraged as long as the source is fully acknowledged and this notice remains in place
- Bug fixes and improvements are welcomed
  - Please email *workshop (at) bgp4all.com*

Philip Smith

BGP Videos

- NSRC has made a video recording of this presentation, as part of a library of BGP videos for the whole community to use:
  - https://learn.nsrc.org/bgp#communities



Using BGP Communities

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

Using BGP Communities

- Five scenarios are covered:
  - Well-known BGP communities
  - RFC1998 traffic engineering
  - Extending RFC1998 ideas for even greater customer policy options
  - Community use in Network Operator backbones
  - Customer Policy Control (aka traffic engineering)

Well-known BGP Communities

How the “well-known” BGP communities are used

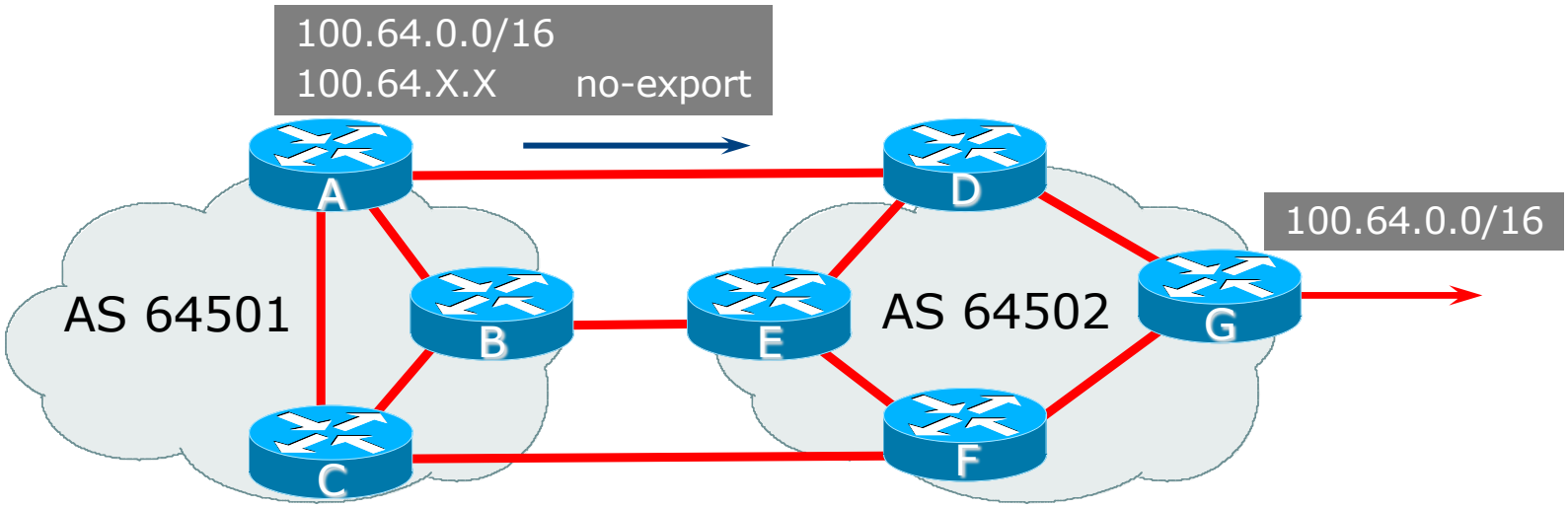
Well-Known Communities

- Several well-known communities
  - www.iana.org/assignments/bgp-well-known-communities
- Five most common:
  - *no-export* 65535:65281
    - Do not advertise to any EBGP peers
  - *no-advertise* 65535:65282
    - Do not advertise to any BGP peer
  - *no-peer* 65535:65284
    - Do not advertise to bi-lateral peers (RFC3765)
  - *blackhole* 65535:666
    - Null route the prefix (RFC7999)
  - *graceful-shutdown* 65535:0
    - Indicate imminent graceful shutdown (RFC8326)

Well-Known Communities: Notes

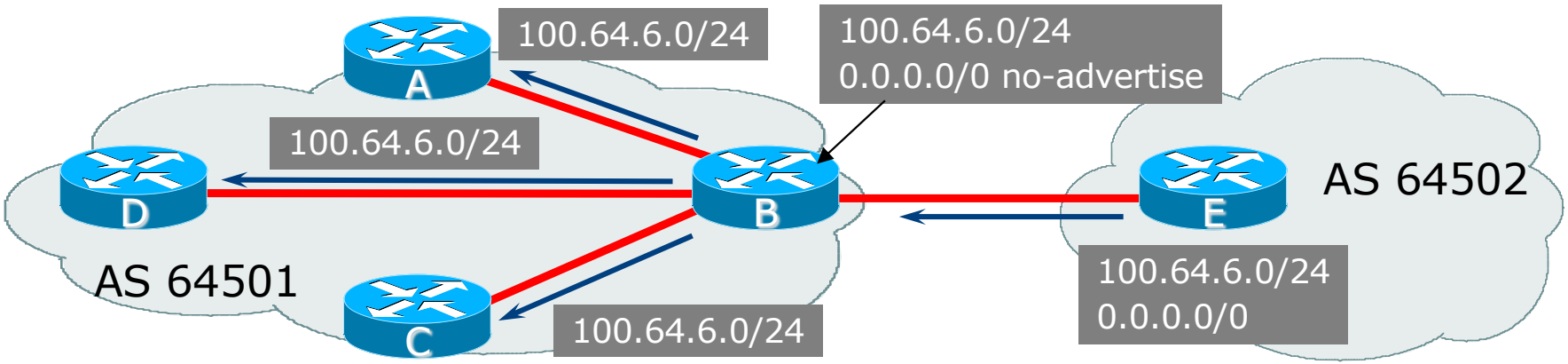
- Even though there are several well-known communities there are variations in implementation support
  - Not all vendors will create configuration key-words to support them
  - Not all vendors will automatically implement their behaviours
  - Not all vendors will allow them to be overwritten
  - *And so on*
- Check vendor documentation for implementation details
  - RFC8642 will give some idea as to the issues to be aware of
- Advice:
  - If the key-word does not exist, create a community declaration that implements the key-word (for configuration clarity & simplicity)

No-Export Community



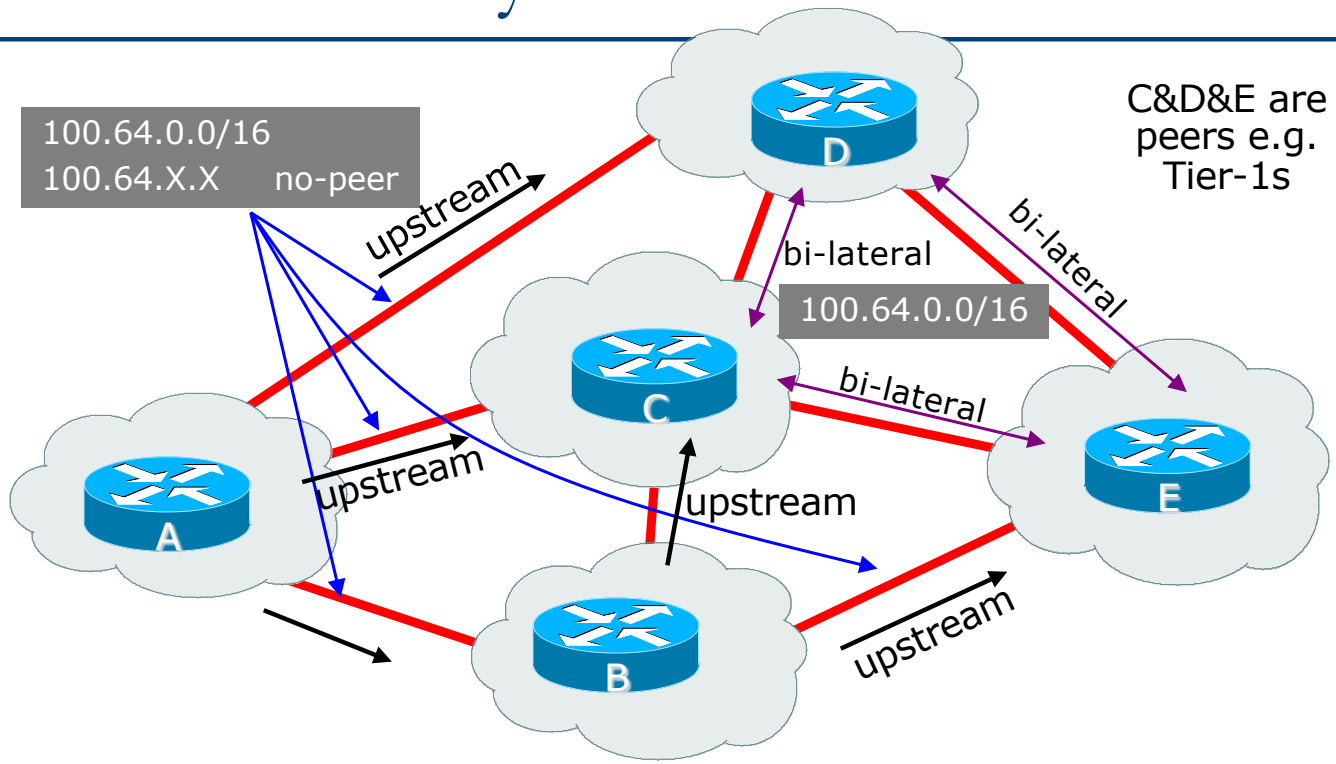
- AS64501 announces aggregate and subprefixes
  - Intention is to improve loadsharing by leaking subprefixes to upstream AS64502 only
- Subprefixes marked with no-export community
- Router G in AS64502 does not announce prefixes with no-export community set

No-Advertise Community



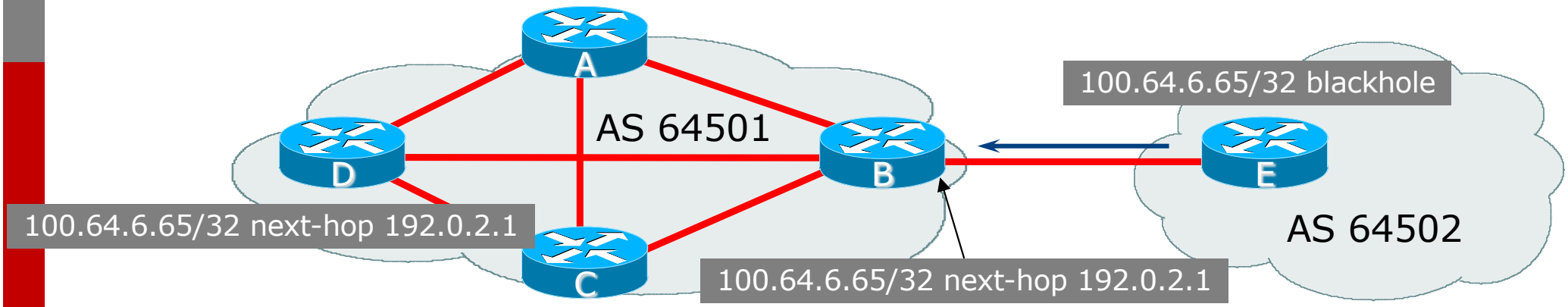
- Used to not advertise a prefix in IBGP
  - B hears 0.0.0.0/0 from EBGP peer E
  - Tags 0.0.0.0/0 as *no-advertise*
  - B will (automatically) not announce prefix to A, C or D
  - Easier/more scalable than using a prefix filter

No-Peer Community



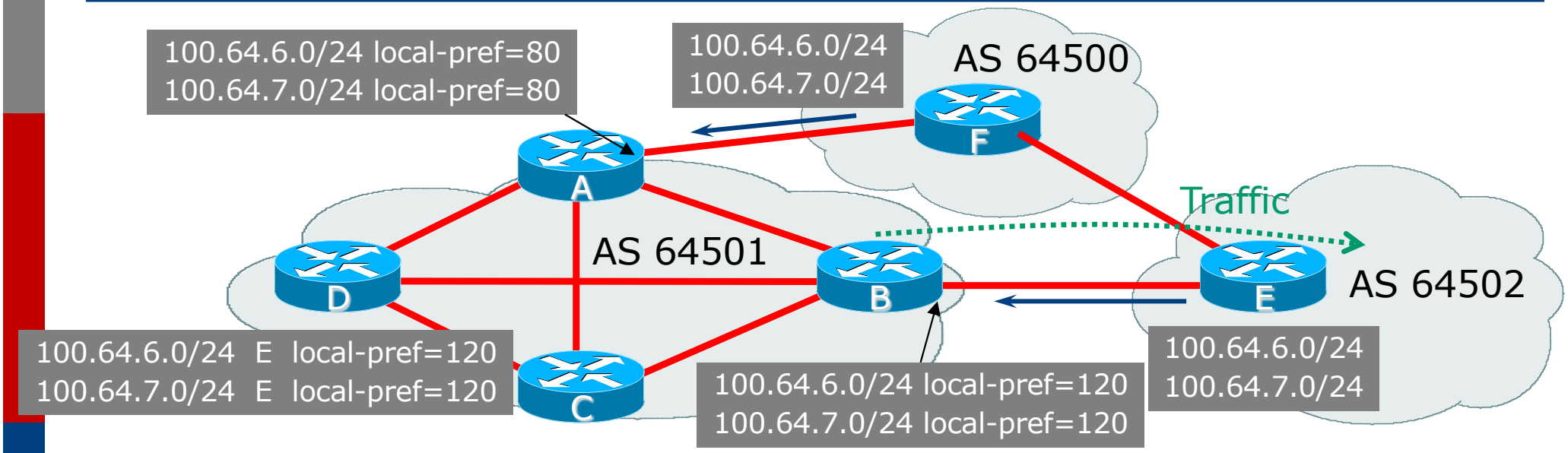
- Sub-prefixes marked with no-peer community are not sent to bi-lateral peers
  - They are only sent to upstream providers

Blackhole Community



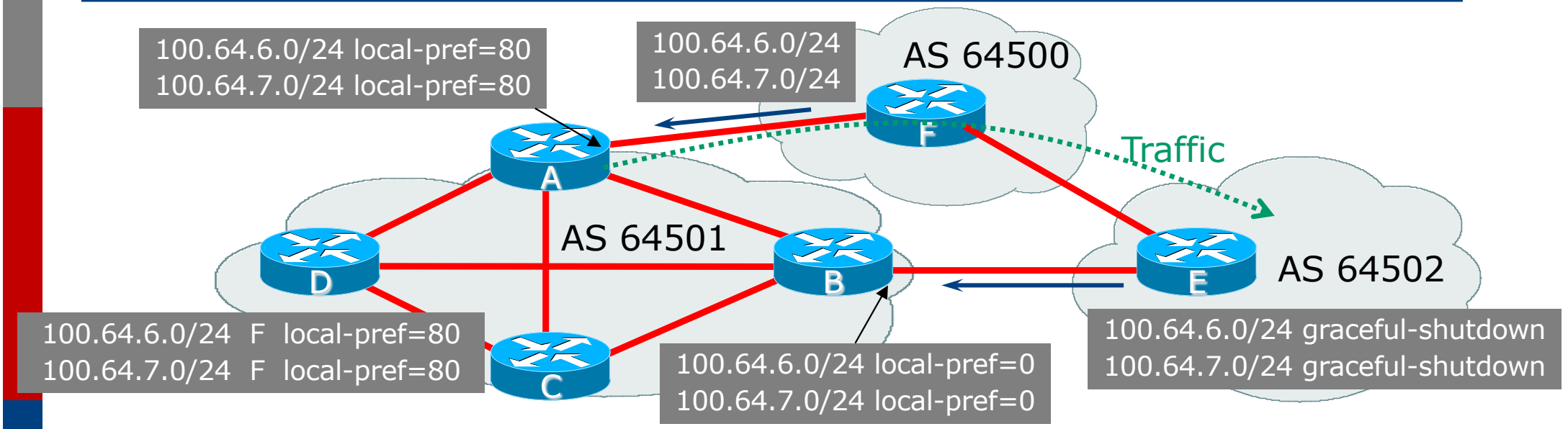
- Used to signal to a BGP neighbour to null route traffic
  - Router E sets *blackhole* community
  - Router B detects *blackhole* community on incoming EBGP announcements and sets *next-hop* to 192.0.2.1
  - 192.0.2.1 is routed to Null interface on all routers within the Autonomous System
  - All traffic to 100.64.6.65 is Null routed

Graceful-Shutdown Community (before)



- Used to inform an EBGP peer that the peering will be going down soon
  - Steady state is primary path between AS64502 and 64501 is via routers E & B
  - AS64502 wants to shutdown direct link, which means traffic will use path via AS64500
  - Graceful-Shutdown ensures that this can be achieved without traffic loss by informing AS64501 that the link is going away

Graceful-Shutdown Community (after)



- Used to inform an EBGP peer that the peering will be going down soon
  - Router E sets *graceful-shutdown* community
  - Router B detects *graceful-shutdown* community on incoming EBGP announcements and sets *local-preference* to 0
  - Best path to 100.64.6.0/24 and 100.64.7.0/24 is now via Router F
  - Allows graceful transition of external best path from Router E to Router F

RFC1998

An example of how Network Operators 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 below

Community Value	Action	Description
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 provider with the same AS path length
ASx:70	`set local preference 70`	The main link is to another provider

RFC1998

- Upstream Provider defines the communities mentioned
- Their customers then attach the communities they want to use to the prefix announcements they are making
- An example, using AS64500 as the upstream ASN:
  - To declare a particular path as a backup path, their customer would announce the prefix with community 64500:70 to AS64500
  - AS64500 would receive the prefix with the community 64500:70 tag, and then set local preference to be 70

RFC1998

- Sample End-Site Router Configuration

```
router bgp 64502
 address-family ipv4
  neighbor 100.66.32.1 remote-as 64500
  neighbor 100.66.32.1 description Backup Provider
  neighbor 100.66.32.1 route-map as64500-out out
  neighbor 100.66.32.1 send-community
  neighbor 100.66.32.1 activate
!
ip as-path access-list 20 permit ^$
!
route-map as64500-out permit 10
 match as-path 20
 set community 64500:70
!
```

RFC1998

- Sample Upstream Router Configuration

```
router bgp 64500
 address-family ipv4
  neighbor 100.66.32.2 remote-as 64502
  neighbor 100.66.32.2 route-map customer-policy-in in
  neighbor 100.66.32.2 activate
!
! Homed to another Provider
ip community-list standard rfc1998-70 permit 64500:70
! Homed to another Provider with equal ASPATH length
ip community-list standard rfc1998-80 permit 64500:80
! Customer backup routes
ip community-list standard rfc1998-90 permit 64500:90
!
```

RFC1998

```
route-map customer-policy-in permit 10
 match community rfc1998-70
 set local-preference 70
!
route-map customer-policy-in permit 20
 match community rfc1998-80
 set local-preference 80
!
route-map customer-policy-in permit 30
 match community rfc1998-90
 set local-preference 90
!
route-map customer-policy-in permit 40
 set local-preference 100
!
```

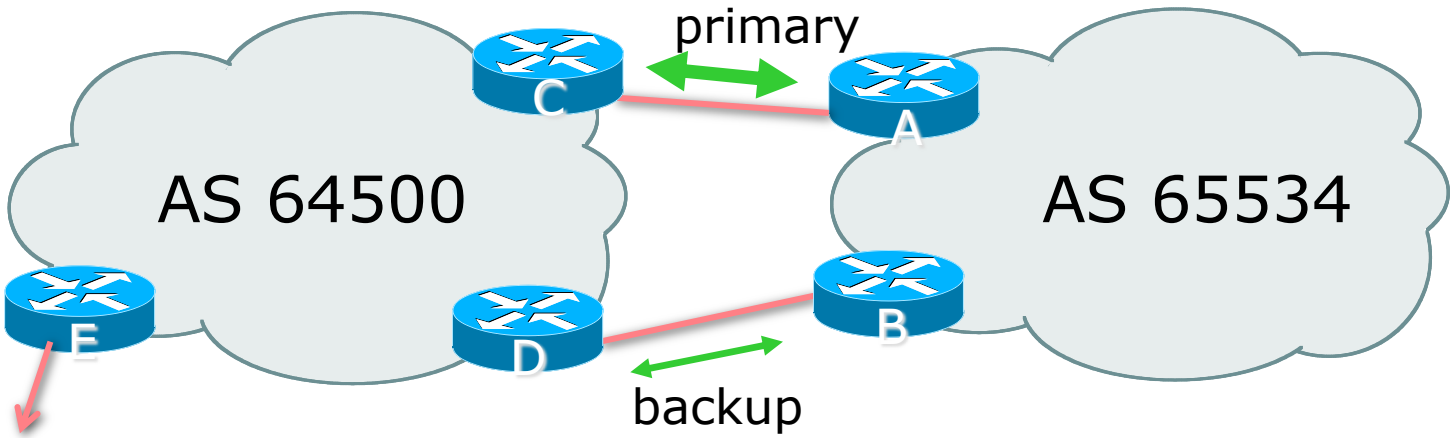
RFC1998

- RFC1998 was the inspiration for a large variety of differing community policies implemented by Network Operators worldwide
- There are no “standard communities” for Network Operators
- But best practices today consider that Network Operators should use BGP communities extensively for multihoming support of traffic engineering
- Look in the Network Operator AS Object in the IRR for documented community support

RFC1998 Example

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

Two links to the same AS



- AS64500 proxy aggregates for AS 65534

Two links to the same AS (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 AS (one as backup only)

- Router A Configuration

```
router bgp 65534
 address-family ipv4
  network 100.64.0.0 mask 255.255.224.0
  neighbor 100.66.10.2 remote-as 64500
  neighbor 100.66.10.2 description RouterC
  neighbor 100.66.10.2 prefix-list aggregate out
  neighbor 100.66.10.2 prefix-list default in
  neighbor 100.66.10.2 activate
!
ip prefix-list aggregate permit 100.64.0.0/19
ip prefix-list default permit 0.0.0.0/0
!
```

Two links to the same AS (one as backup only)

- Router B Configuration

```
router bgp 65534
 address-family ipv4
  network 100.64.0.0 mask 255.255.224.0
  neighbor 100.66.10.6 remote-as 64500
  neighbor 100.66.10.6 description RouterD
  neighbor 100.66.10.6 send-community
  neighbor 100.66.10.6 prefix-list aggregate out
  neighbor 100.66.10.6 route-map routerD-out out
  neighbor 100.66.10.6 prefix-list default in
  neighbor 100.66.10.6 route-map routerD-in in
  neighbor 100.66.10.6 activate
!
..next slide..
```

Two links to the same AS (one as backup only)

```
ip prefix-list aggregate permit 100.64.0.0/19
ip prefix-list default permit 0.0.0.0/0
!
route-map routerD-out permit 10
 match ip address prefix-list aggregate
 set community 64500:90
route-map routerD-out permit 20
!
route-map routerD-in permit 10
 set local-preference 90
!
```

Two links to the same AS (one as backup only)

- Router C Configuration (main link)

```
router bgp 64500
 address-family ipv4
  neighbor 100.66.10.1 remote-as 65534
  neighbor 100.66.10.1 default-originate
  neighbor 100.66.10.1 prefix-list Customer in
  neighbor 100.66.10.1 prefix-list default out
  neighbor 100.66.10.1 activate
!
ip prefix-list Customer permit 100.64.0.0/19
ip prefix-list default permit 0.0.0.0/0
```

Two links to the same AS (one as backup only)

- Router D Configuration (backup link)

```
router bgp 64500
 address-family ipv4
  neighbor 100.66.10.5 remote-as 65534
  neighbor 100.66.10.5 default-originate
  neighbor 100.66.10.5 prefix-list Customer in
  neighbor 100.66.10.5 route-map bgp-cust-in in
  neighbor 100.66.10.5 prefix-list default out
  neighbor 100.66.10.5 activate
!
ip prefix-list Customer permit 100.64.0.0/19
ip prefix-list default permit 0.0.0.0/0
!
...next slide...
```

Two links to the same AS (one as backup only)

```
!
ip community-list standard rfc1998-90 permit 64500:90
!
route-map bgp-cust-in permit 10
 match community rfc1998-70
 set local-preference 70
route-map bgp-cust-in permit 20
 match community rfc1998-80
 set local-preference 80
route-map bgp-cust-in permit 30
 match community rfc1998-90
 set local-preference 90
route-map bgp-cust-in permit 40
 set local-preference 100
!
```

Two links to the same AS (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
- Network Operators create backbone support for many other communities to handle more complex situations
  - Simplify Network Operator BGP configuration
  - Give customer more policy control

Network Operator BGP Communities

- There are no recommended Network Operator BGP communities apart from
  - RFC1998
  - The well-known communities
    - www.iana.org/assignments/bgp-well-known-communities
- Efforts have been made to document from time to time
  - https://www.ietf.org/archive/id/draft-bonaventure-quoitin-bgp-communities-00.txt
  - But so far… nothing more… ☹
  - Collection of Network Operator communities at www.onesc.net/communities
  - NANOG Tutorial: www.nanog.org/meetings/nanog40/presentations/BGPcommunities.pdf
- Network Operator policy is usually published
  - On the Operator's website
  - Referenced in the AS Object in the IRR

Typical Network Operator BGP Communities

Community Value	Action	Description
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 (RFC7999)

Sample Router Configuration (1)

```
router bgp 64500
 address-family ipv4
  neighbor 100.66.32.2 remote-as 64502
  neighbor 100.66.32.2 route-map customer-policy-in in
  neighbor 100.66.32.2 activate
  neighbor 100.65.8.9 remote-as 64510
  neighbor 100.65.8.9 route-map upstream-out out
  neighbor 100.65.8.9 activate
!
ip community-list standard prepend-1 permit 64500:1
ip community-list standard prepend-2 permit 64500:2
ip community-list standard prepend-3 permit 64500:3
ip community-list standard lp-80     permit 64500:80
ip community-list standard lp-120    permit 64500:120
ip community-list standard RTBH      permit 64500:666
!
ip route 192.0.2.1 255.255.255.255 null0
```

Customer BGP

Upstream BGP

Black hole route (on all routers)

Sample Router Configuration (2)

```
route-map customer-policy-in permit 10
 match community lp-80
 set local-preference 80
!
route-map customer-policy-in permit 20
 match community lp-120
 set local-preference 120
!
route-map customer-policy-in permit 30
 match community RTBH
 set ip next-hop 192.0.2.1
!
route-map customer-policy-in permit 40

...etc...
```

Sample Router Configuration (3)

```
route-map upstream-out permit 10
 match community prepend-1
 set as-path prepend 64500
!
route-map upstream-out permit 20
 match community prepend-2
 set as-path prepend 64500 64500
!
route-map upstream-out permit 30
 match community prepend-3
 set as-path prepend 64500 64500 64500
!
route-map upstream-out permit 40

...etc...
```

Community	Local-Pref	Description
(default)	120	customer
65520:nnnn	50	this community will only set the local preference within the connected country, not beyond
65530:nnnn	50	this community will only set the local preference within the connected region, not beyond
2914:435	50	only beyond the connected country
2914:436	50	only beyond the connected region
2914:450	96	customer fallback
2914:460	98	peer backup
2914:470	100	peer
2914:480	110	customer backup
2914:490	120	customer default
2914:666		**blackhole**

Customers wanting to alter their route announcements to other customers

NTT BGP customers may choose to prepend to all other NTT BGP customers with the following communities:

Community	Description
2914:411	prepends o/b to customer 1x
2914:412	prepends o/b to customer 2x
2914:413	prepends o/b to customer 3x

Example: NTT

More info at https://www.gin.ntt.net/support-center/policies-procedures/routing/

Example: Verizon Europe

```
aut-num:        AS702
descr:          Verizon Business EMEA - Commercial IP service provider in Europe
<snip>
remarks:        ----------------------------------------------------------
                Verizon Business filters out inbound prefixes longer than /24.
                We also filter any networks within AS702:RS-INBOUND-FILTER.
                ----------------------------------------------------------
                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.
<snip>
                ----------------------------------------------------------
                Additional details of the VzBi communities are located at:
                http://www.verizonbusiness.com/uk/customer/bgp/
```

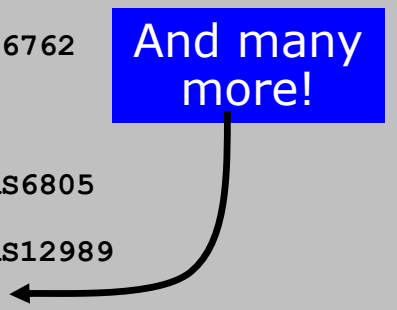
And many more!

Example: Arelion

```
aut-num:        AS1299
descr:          Arelion, f/k/a Telia Carrier
<snip>
remarks:        BGP COMMUNITY SUPPORT FOR AS1299 TRANSIT CUSTOMERS:
remarks:
remarks:        Community Action (default local pref 200)
remarks:        ----------------------------------------
remarks:        1299:50 Set local pref 50 within AS1299 (lowest possible)
remarks:        1299:150 Set local pref 150 within AS1299 (equal to peer, backup)
remarks:        1299:1y050 Set local pref 50 in region y
remarks:        1299:1y150 Set local pref 150 in region y
remarks:        Where y is:
remarks:        0= outside own continent
remarks:        2= Europe
remarks:        5= North America
remarks:        7= Asia Pacific
<snip>
remarks:        European peers
remarks:        Community Action
remarks:        --------- ------
remarks:        1299:200x All peers Europe incl:
remarks:
remarks:        1299:252x NTT/2914
remarks:        1299:253x Zayo/6461
remarks:        1299:254x Orange/5511
remarks:        1299:256x Lumen/3356
remarks:        1299:257x Verizon/702
<snip>
remarks:        Where x is number of prepends (x=0,1,2,3) or do NOT announce (x=9)
```

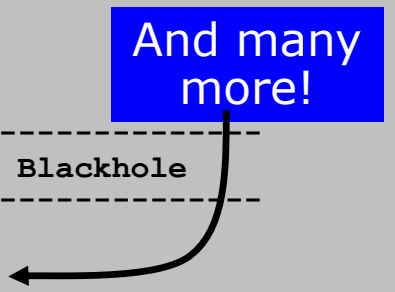
Example: BT Ignite

```
aut-num:      AS5400
descr:        BT
<snip>
remarks:        Communities scheme:
remarks:        The following BGP communities can be set by BT
remarks:        BGP customers to affect announcements to major peers.
remarks:
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         Colt AS3356
remarks:        509         Arelion AS1299
remarks:        002         Sprint AS1239
remarks:        004         Vodafone Global Network AS1273
remarks:        005         Verizon EMEA AS702
remarks:        014         DTAG AS3320
remarks:        016         Orange AS5511
remarks:        018         Tata Communications Ltd AS6453
remarks:        023         GTT Communications AS3257
remarks:        045         Telecom Italia Sparkle AS6762
remarks:        073         GTT Communications AS286
remarks:        169         Cogent AS174
remarks:        177         Telxius Cable AS12956
remarks:        177         Telefonica Germany GmbH AS6805
remarks:        190         Comcast AS7922
remarks:        191         Highwinds Network Group AS12989
<snip>
```



Example: Level3

```
aut-num:      AS3356
descr:        Level 3 Communications
<snip>
remarks:        ------------------------------------------------------
remarks:        customer traffic engineering communities - Suppression
remarks:        ------------------------------------------------------
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:        ------------------------------------------------------
remarks:        customer traffic engineering communities - Prepending
remarks:        ------------------------------------------------------
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
<snip>
remarks:        ------------------------------------------------------
remarks:        customer traffic engineering communities - LocalPref
remarks:        ------------------------------------------------------
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:        ------------------------------------------------------
remarks:        customer traffic engineering communities - Blackhole
remarks:        ------------------------------------------------------
remarks:        3356:9999 - blackhole (discard) traffic
<snip>
```



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 Network Operator and configuration examples as a guideline

Using Communities for Backbone Scaling

Scaling BGP in the Service Provider backbone…

Communities for IBGP

- Network Operators 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 Provider'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

- Network Operator 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
- Network Operator identifies whether address space comes from their PA block or is their customers' own PI space
  - One community for each

Community Definitions

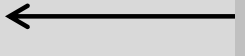
64500:1000	AS64500 aggregates
64500:1001	AS64500 aggregate subprefixes
64500:1005	Static Customer PI space
64500:2000	Customers who get Transit
64500:2100	Customers who get IXP access
64500:2200	Customers who get BGP Customer access
64500:3000	Routes learned from the IXP

```
ip community-list standard aggregates permit 64500:1000
ip community-list standard subnets    permit 64500:1001
ip community-list standard pi         permit 64500:1005
ip community-list standard transits   permit 64500:2000
ip community-list standard ixp-access permit 64500:2100
ip community-list standard bgp-cust   permit 64500:2200
ip community-list standard ixp-routes permit 64500:3000
```

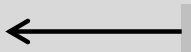
Aggregates and Static Customers into BGP

```
router bgp 64500
 address-family ipv4
  network 100.64.0.0 mask 255.255.224.0 route-map as64500-prefixes
  redistribute static route-map static-to-bgp
!
ip prefix-list as64500-block permit 100.64.0.0/19 le 32
!
route-map as64500-prefixes permit 10
 set community 64500:1000
!
route-map static-to-bgp permit 10
 match ip address prefix-list as64500-block
 set community 64500:1001
route-map static-to-bgp permit 20
 set community 64500:1005
```

Aggregate community set



Aggregate subprefixes community set



PI community is set

Service Identification

- AS64500 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 AS64500 customer installation engineer to provision

BGP Customers
Creating peer-groups

```
router bgp 64500
 address-family ipv4
  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 upstream-only peer-group
  neighbor upstream-only route-map customers-out out
  neighbor upstream-only route-map upstream-only-in in
  neighbor upstream-only 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 aggregates
!
route-map full-transit-in permit 10
 set community 64500:2000 64500:2100 64500:2200
!
route-map upstream-only-in permit 10
 set community 64500:2000
!
route-map ixp-routes permit 10
 match ip community aggregates pi transits ixp-access ixp-routes
!
route-map ixp-only-in permit 10
 set community 64500:2100
!
route-map bgp-cust-out permit 10
 match ip community aggregates pi transits bgp-custs
!
route-map bgp-cust-in permit 10
 set community 64500:2200
```

Customers only get AS64500 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

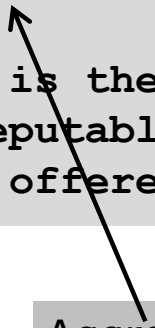
```
router bgp 64500
 address-family ipv4
  neighbor 100.67.3.2 remote-as 64501
  neighbor 100.67.3.2 peer-group full-transit
  neighbor 100.67.3.2 prefix-list as64501cust-in
  neighbor 100.67.3.2 activate
  neighbor 100.67.3.6 remote-as 64502
  neighbor 100.67.3.6 peer-group upstream-only
  neighbor 100.67.3.6 prefix-list as64502cust-in
  neighbor 100.67.3.6 activate
  neighbor 100.67.3.10 remote-as 64503
  neighbor 100.67.3.10 peer-group ixp-only
  neighbor 100.67.3.10 prefix-list as64503cust-in
  neighbor 100.67.3.10 activate
  neighbor 100.67.3.14 remote-as 64504
  neighbor 100.67.3.14 peer-group bgpcust-only
  neighbor 100.67.3.14 prefix-list as64504cust-in
  neighbor 100.67.3.14 activate
```

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 64500
 address-family ipv4
  neighbor 100.66.32.1 remote-as 64510
  neighbor 100.66.32.1 prefix-list full-routes in
  neighbor 100.66.32.1 route-map upstream-out out
  neighbor 100.66.32.1 activate
!
route-map upstream-out permit 10
 match ip community aggregates pi transits
!
! 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
```



Aggregates, PI customers and full transit customers are announced to upstream

BGP Customers – configuring IXP peers

```
router bgp 64500
 address-family ipv4
  neighbor 100.70.0.1 remote-as 65536
  neighbor 100.70.0.1 route-map ixp-peers-out out
  neighbor 100.70.0.1 route-map ixp-peers-in in
  neighbor 100.70.0.1 prefix-list AS65536-peer in
  neighbor 100.70.0.1 activate
  neighbor 100.70.0.2 remote-as 65537
  neighbor 100.70.0.2 route-map ixp-peers-out out
  neighbor 100.70.0.2 route-map ixp-peers-in in
  neighbor 100.70.0.2 prefix-list AS65537-peer in
  neighbor 100.70.0.2 activate
!
route-map ixp-peers-out permit 10
 match ip community aggregates pi transits ixp-access
!
route-map ixp-peers-in permit 10
 set community 64500:3000
```

Aggregates, PI customers full transit and IXP customers are announced to the IXP

Service Identification

- 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 Network Operators 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 64500
 address-family ipv4
  neighbor rrc peer-group
  neighbor rrc descr Route Reflector Clients
  neighbor rrc remote-as 64500
  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 64500
  neighbor ibgp-peer send-community
  neighbor 100.64.0.1 peer-group ibgp-peer
  neighbor 100.64.0.1 activate
  neighbor 100.64.0.2 peer-group rrc
  neighbor 100.64.0.2 activate
!
route-map ibgp-filter permit 10
 match community aggregates subnets pi transits ixp-access bgp-cust ixp-routes
!
```

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

- Network Operators have a choice on how to handle policy control for customers
- No delegation of policy options:
  - Customer has no choices
  - If customer wants changes, the operator's Technical Support handles it
- Limited delegation of policy options:
  - Customer has choices
  - The operator's 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:

Community	Action
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 preference set to 80 on customer prefixes
X:120	Local preference set to 120 on customer prefixes
X:666	Black hole this route please! (RFC7999)
X:5000	Don’t announce to any BGP neighbour
X:5MM0	Don’t announce to BGP neighbour MM
X:5MMN	Prepend N times to BGP neighbour MM

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 operator in AS X documents the BGP transits and peers that they have (MM can be 01 to 99)
  - The operator 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:
  - 64500:5030 don't send prefix to BGP neighbour 03
  - 64500:5102 2x prepend prefix announcement to peer 10

Community Definitions

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

```
ip community-list standard all-noann   permit 64500:5000
ip community-list standard all-pre1    permit 64500:5001
ip community-list standard all-pre2    permit 64500:5002
ip community-list standard all-pre3    permit 64500:5003
ip community-list standard peer1-noann permit 64500:5010
ip community-list standard peer1-pre1  permit 64500:5011
ip community-list standard peer1-pre2  permit 64500:5012
ip community-list standard peer1-pre3  permit 64500:5013
ip community-list standard peer2-noann permit 64500:5020
ip community-list standard peer2-pre1  permit 64500:5021
ip community-list standard peer2-pre2  permit 64500:5022
ip community-list standard peer2-pre3  permit 64500:5023
```

Don’t announce anywhere

Single prepend to all

Don’t announce to peer 1

Single prepend to peer 2

Creating route-maps – neighbour 1

```
route-map bgp-neigh-01 deny 10
 match ip community all-noann peer1-noann
!
route-map bgp-neigh-01 permit 20
 match ip community all-pre1 peer1-pre1
 set as-path prepend 64500
!
route-map bgp-neigh-01 permit 30
 match ip community all-pre2 peer1-pre2
 set as-path prepend 64500 64500
!
route-map bgp-neigh-01 permit 40
 match ip community all-pre3 peer1-pre3
 set as-path prepend 64500 64500 64500
!
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 all-noann peer2-noann
!
route-map bgp-neigh-02 permit 20
 match ip community all-pre1 peer2-pre1
 set as-path prepend 64500
!
route-map bgp-neigh-02 permit 30
 match ip community all-pre2 peer2-pre2
 set as-path prepend 64500 64500
!
route-map bgp-neigh-02 permit 40
 match ip community all-pre3 peer2-pre3
 set as-path prepend 64500 64500 64500
!
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

Operator BGP configuration

```
router bgp 64500
 address-family ipv4
  neighbor 100.67.3.2 remote-as 64501
  neighbor 100.67.3.2 route-map bgp-neigh-01 out
  neighbor 100.67.3.2 route-map policy-01 in
  neighbor 100.67.3.2 activate
  neighbor 100.67.3.6 remote-as 64502
  neighbor 100.67.3.6 route-map bgp-neigh-02 out
  neighbor 100.67.3.6 route-map policy-02 in
  neighbor 100.67.3.6 activate
```

- 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 64508
 address-family ipv4
  neighbor 100.69.1.1 remote-as 64500
  neighbor 100.69.1.1 route-map upstream out
  neighbor 100.69.1.1 prefix-list default in
  neighbor 100.69.1.1 activate
!
route-map upstream permit 10
 match ip address prefix-list blockA
 set community 64500:5010 64500:5023
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
  - Operator Technical Support does not need to be involved
- Disadvantages
  - Customer could upset the operator's 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/IXP Workshops