

© Whitepaper is supplement to the Cisco Press publication – *ISP Essentials* by Barry Raveendran Greene, and Philip Smith. Materials can be used with the permission of the authors and Cisco Press. Materials can be used with the permission of the authors and Cisco Press. Public copies are available at <http://ftp-eng.cisco.com/public/cons/isp/essentials/> or www.ispbook.com.

Service Provider Multihoming from a Disconnected Backbone

Draft 0.1

There are conditions where a Service Provider cannot build a contiguous backbone. Some countries lack the physical infrastructure between cities – making it easier to connect a city to the rest of the world vs connecting the cities to each other. Some Service Providers do not build a backbone, relying on their upstream provider to provide a *virtual backbone*. Still others require the use of the incumbent telecommunication monopoly's infrastructure to build their backbone – which often results in limited bandwidth and extremely high cost. All of these factors are valid conditions for a Service Provider to build a *Disconnected Backbone*.

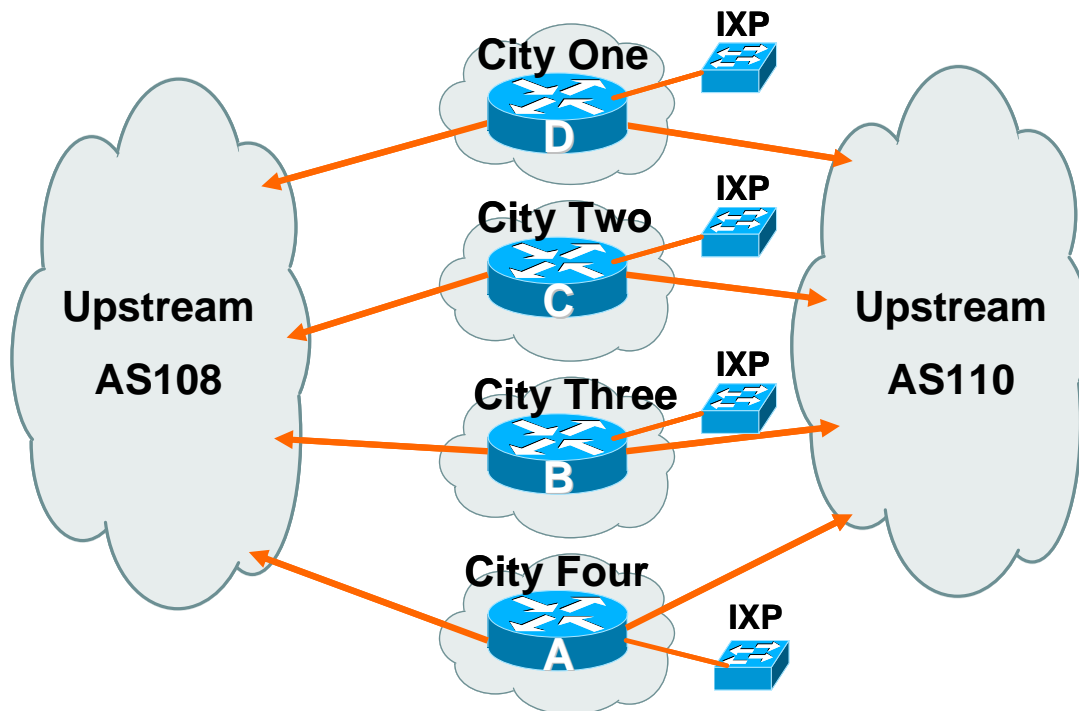


Figure 1 - Disconnected Backbone Topology Example

DISCONNECTED BACKBONE BCPs

- **Contiguous IP Allocations.** Make every effort to allocate contiguous blocks of addresses to each of the Disconnected Backbone POPs. This will take time and effort in the beginning, but will pay off in reduced operational cost and a wider range of networking options in the future.
- **Assume a backbone – even if there is no backbone.** While there are many factors that keep the creation of a backbone prohibitive, still plan on a backbone. It may not be a reality today, but sometime in the future it will. Retrofitting a backbone into a network situation that has not plans for a backbone prove – through experience – to be painful and costly transitions.
- **Connect Locally.** One of the key principles for making profit in the IP Service Provider business is to keep traffic local on cost efficient bandwidth. Disconnected Backbones need to find ways to keep traffic within a City locally in that City. That would me joining or facilitating the creation of peering points with the competition. Service Providers who do not connect locally have proven to fall down the path of business failure – so consider local interconnections as a business imperative.

ADVERTISING PREFIXES TO UPSTREAM ISPs

Disconnected Backbone's core critics revolve around their prefix advertisements to the Internet. The critics point to the additional numbers of more specific prefixes advertised into the Global Internet Route Table. These critics can be mitigated by some thoughtful use of BGP. Namely, the use of BGP communities to limit the scope of the more specific prefixes advertised to the Global Internet Route Table. The best way to illustrate this technique is through an example.

Figure 1 provides a typical Service Provider with a disconnected backbone. This service provider has four cities, each with a connection to two upstream backbones. Hopefully the Service Provider has been diligent with the IP allocations. So we'll assume that each of the four POPs have /22 out of a larger /19. We will also assume that the service provider creates and maintains a *prefix advertisement map* (see Figure 2). This is a topology map that contains diagrams how prefixes are advertised through out the network. Given these assumptions, the Service Provider needs to consider the following to insure connectivity:

1. Each Upstream ISP requires the more specific prefixes (/22s in this case) to insure packets get route to the appropriate city/POP.

2. One City/POP needs to advertise the entire /19. This insures the /19 is advertised and information on the whole /19 block is collected (i.e. Security tools like Sink Holes monitoring the network).
3. Each POP needs to allow the more specific prefixes from their sister city/POPs to insure devices in each POP can communicate with each other.
4. The Internet requires a path to the /19 to insure connectivity to the ISP. The Internet ***do not require the more specific /22s***. Internet backbones need only to forward the packet to the supporting Upstream NSPs. Once the packet forwarded to the /19 reach the Upstream NSPs, the NSP will then use the more specific /22 prefixes to forward the packet to the appropriate city/POP.

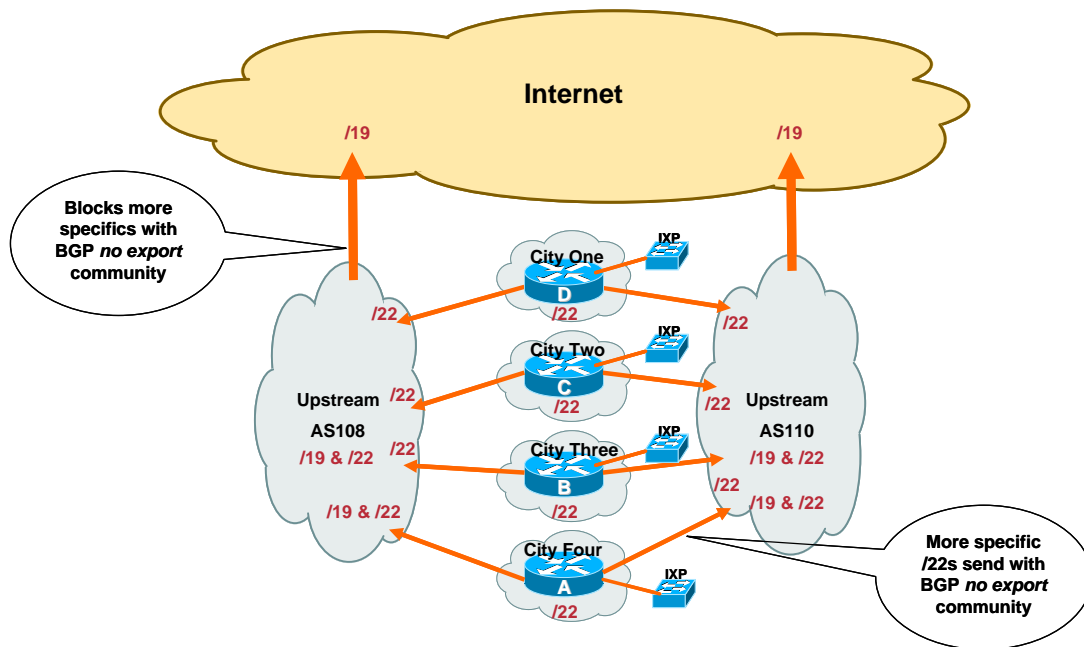


Figure 2 - Prefix Advertisement Map

The only networks that really need the more specific /22s are the two upstream NSPs (AS 108 and AS 110 in this example). Given this, the Disconnected Service marks their more specific advertisement with the BGP *no export* community. No Export is one of the four well known BGP communities. It instructs the neighboring peer to not advertise the BGP prefix. In this example (Figure 2) the /22s tagged with the no export community will be advertised to AS 108 and AS 110. AS 108 and AS 110 will then see the no export community and not advertise it to any other BGP

peer or customer. Simply, the /22 will stay in AS 108 and AS 110 and not go out to the rest of the Internet.

CISCO IOS CONFIGURATION EXAMPLE

Adding the BGP *no export* community to the advertisement can be done several ways. Each are straight forward. In the following example, the community is added with a route-map that matches only the /22 that is being advertised. That way the /19 that is advertised out Site/City A is permitted without a *no export* community.

```
router bgp 604
 network 221.10.0.0 mask 255.255.252.0
 neighbor 222.200.0.1 remote-as 108
 neighbor 222.200.0.1 description AS108 - Serial 0/0
 neighbor 222.200.0.1 send-community
 neighbor 222.200.0.1 prefix-list rfc1918-sua in
 neighbor 222.222.0.1 prefix-list my-block out
 neighbor 222.222.0.1 route-map send-community-out
 neighbor 222.222.10.1 remote-as 110
 neighbor 222.222.10.1 description AS110 - Serial 1/0
 neighbor 222.222.10.1 send-community
 neighbor 222.222.10.1 prefix-list rfc1918-sua in
 neighbor 222.222.10.1 prefix-list my-block out
 neighbor 222.222.10.1 route-map send-community-out
 neighbor 222.222.10.1 filter-list 10 in
!
ip prefix-list my-block permit 221.10.0.0/22
!
ip as-path access-list 10 permit ^(110_)+$
ip as-path access-list 10 permit ^(110_)+_[0-9]+$
!...etc to achieve outbound loadsharing
!
ip route 0.0.0.0 0.0.0.0 Serial 1/0 250
ip route 221.10.0.0 255.255.252.0 null0
!
route-map set-community permit 10match ip address 1
 set community no-export
!
route-map set community permit 20
 match ip address 2
!
access-list 1 permit 221.10.0.0 0.0.3.255
access-list 2 permit any
```