

Effective BGP Load Balancing Using "The Metric System"

*A real-world guide to
BGP traffic engineering*

NANOG47

October 19, 2009

Dani Roisman

droisman ~ at ~ peakwebconsulting ~ dot ~ com

Introductions

Why BGP Load Balancing?

- Networks that have multiple ISP connections and are receiving the Internet routing table via BGP
- Typically multi-homed – i.e. connections to diverse ISPs, but can be multiple connections to the same ISP
- Need to balance utilization of uplinks by selecting which traffic takes which path:
 - Avoid uplink congestion (over/under utilization)
 - Control *cost* (meet commits, send overage to least expensive)
 - Improve *performance* (trouble through one ISP, but not another)

Today's Talk: Outbound Only

- Techniques are different for inbound versus outbound BGP load balancing at your network's border
- Today's talk focuses on outbound
- Typically for content networks, outbound traffic is the billable value, the benefit is that there is more control on outbound load balancing
- Technique used mostly for ISP uplinks, peering uplinks are usually considered differently
- Inbound load balancing is less precise, and often relies on behavior of upstream and further remote networks

What's the Problem?

- There is no standard method of balancing traffic over multiple ISP uplinks (a.k.a. “BGP traffic engineering”)
- No routing language where you can configure “send 300Mbps on this link, send 500Mbps on this other link”
- Some hardware/software solutions for “route optimization” that may be incompatible with network layout, or may be out-of-budget
- While there are many guides to BGP that help introduce this topic to network engineers, most only touch upon the mechanisms used for traffic engineering, without providing much guidance

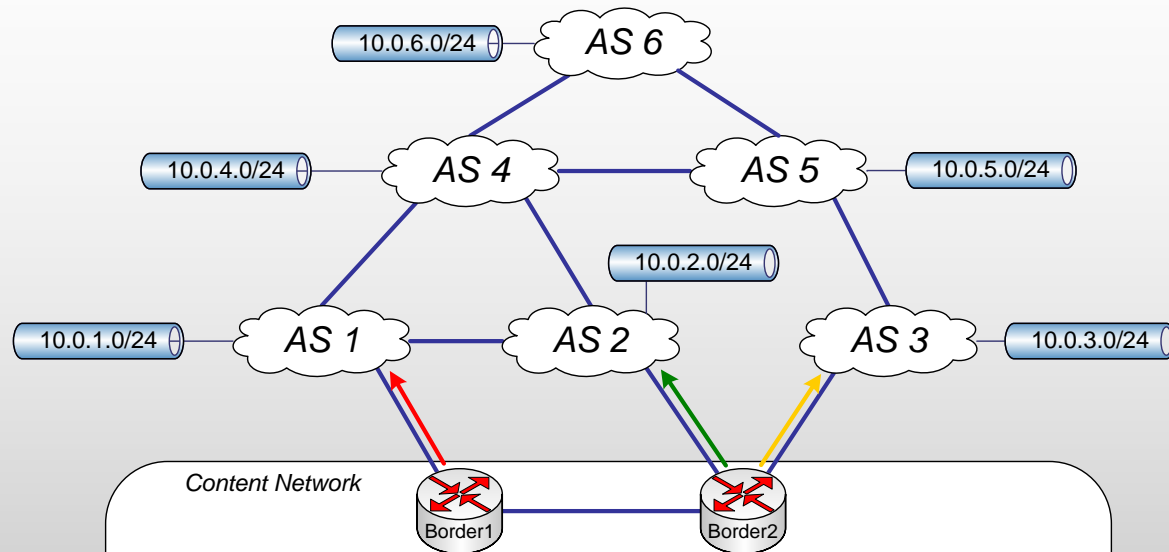
Who May Be Interested?

- Enterprise or Content networks with diverse ISP uplinks utilizing BGP
- Regional transit-purchasing ISPs with diverse upstream ISP connectivity
- BGP novices
- Senior network engineers seeking materials to help train junior staff
- Target audience is increasing as more end-user networks pursue to multi-homing to improve performance and fault-tolerance

What Are You Going to Learn?

- A BGP load-balancing method that is consistent, deterministic, and effective
 - Proven configurations that have been successful for the past 9 years
 - Implemented at over 10 networks of various sizes: 10 Mbps to 80+ Gbps

Reference: Sample Network



Base Configuration

Border 1

eBGP with ISPs AS1
iBGP with Border 2

Border 2

eBGP with ISPs AS2 & AS3
iBGP with Border 1

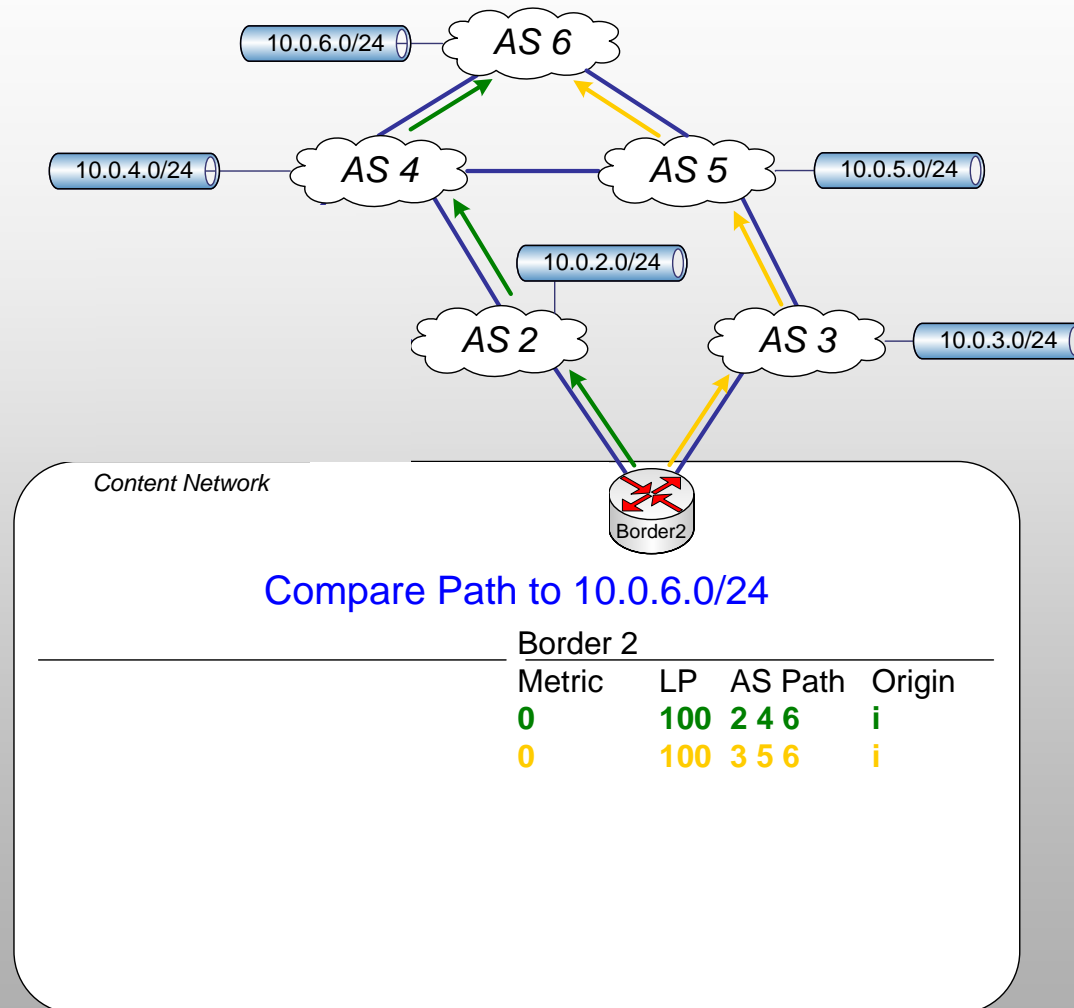
*AS1 and AS2 = "Tier 1" (transit free, fully peered)
AS3 = "Lower Tier" (partially peered, purchases transit)*

Refresher: BGP Decision Process

Refresher: BGP Routing Decision

- There are some minor differences in the way that vendors make decisions when multiple valid BGP next-hop paths are available:
 - Proprietary “weight” or “preference”
 - **BGP Local Preference**
 - **AS Path Length**
 - **BGP Origin Code (Int. / Ext. / Unknown)**
 - **MED (let’s call this “BGP metric”)**
 - How learned: eBGP versus iBGP
 - IGP metric to exit point
 - “Oldest” route entry
 - BGP neighbor router-ID IP address
 - BGP neighbor next-hop IP address

Example: Compare Two Paths



Example: Compare Two Paths

Output from Border2 as Cisco (or similar) router :

Network	Next Hop	Metric	LocPrf	Weight	Path
*10.0.6.0/24	1.1.3.1	0	100	0	3 5 6 i
	1.1.2.1	0	100	0	2 4 6 i

BGP routing table entry for 10.0.6.0/24

3 5 6

1.1.3.1 from 1.1.3.1

Origin IGP, localpref 100, external, best

(Metric 0, Weight 0)

2 4 6

1.1.2.1 from 1.1.2.1

Origin IGP, localpref 100, external

(Metric 0, Weight 0)

Example: Compare Two Paths

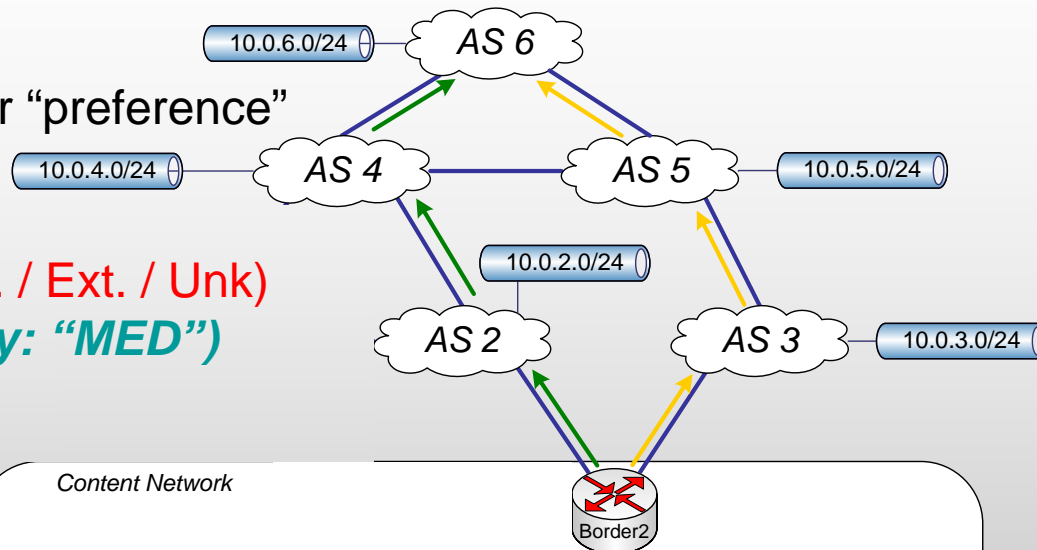
Output from Border2 as Juniper router:

Destination	Metric 1	Metric 2	Next hop	AS path
*10.0.6.0/24	100	0	>1.1.3.1	3 5 6 I
	100	0	>1.1.2.1	2 4 6 I

```
10.0.6.0/24 Source: 1.1.3.1
Next hop: 1.1.3.1 via ge-0/0/0.0
State: <Active Ext>
Age: 3w0d 23:31:10 Metric: 0
AS path: 3 5 6 I
Localpref: 100
Router ID: 1.1.3.1
```

```
Source: 1.1.2.1
Next hop: 1.1.2.1 via ge-1/0/0.0
State: <Ext>
Age: 3w4d 7:43:43 Metric: 0
AS path: 2 4 6 I
Localpref: 100
Router ID: 1.1.2.1
```

Results: Compare Two Paths



Content Network

Border2

Compare Path to 10.0.6.0/24

Border 2				
Metric	LP	AS Path	Origin	
0	100	2 4 6	i	
0	100	3 5 6	i	

Which path is selected???

Goes to "Oldest Route" tie breaker

Not deterministic: Maintenance or flap will cause a shift!!!

- Proprietary "weight" or "preference"
- **BGP Local Pref**
- **AS Path Length**
- **BGP Origin Code (Int. / Ext. / Unk)**
- **BGP Metric (formerly: "MED")**
- EBGP versus IBGP
- IGP metric
- "Oldest" route entry
- BGP neighbor router-ID *
- BGP neighbor next-hop *
- * (compare IP address if same ASN)

BGP Attributes of Interest

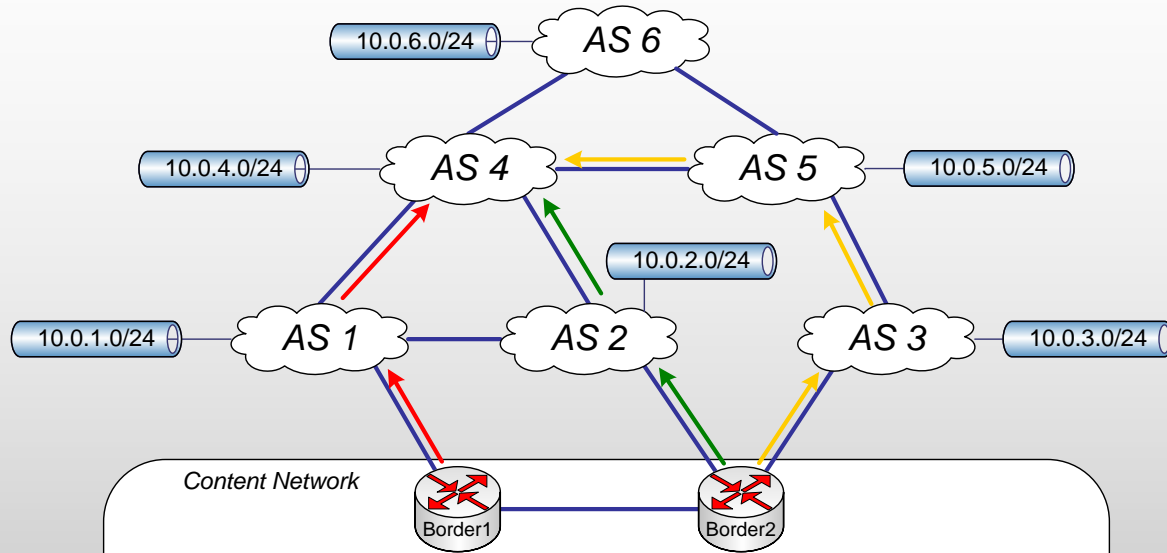
Three Route Attributes of Interest

- BGP Local Preference
 - Set within local AS, only shared to iBGP neighbors
- AS Path Length (a.k.a. hop count)
 - AS Path describes networks traversed to reach destination
 - Shared with all BGP speakers across the Internet
- BGP Origin Code (Int / Ext / Unknown)
 - *Interesting only because it can interfere with the “metric system”*
 - *We’ll end up disregarding this attribute*
- MED (let’s call this “BGP metric”)
 - Describes “cost” to destination (lower is better)
 - Shared within eBGP and iBGP
 - Often times only has relevance within a local AS
 - Often times will be removed or reset to a fixed value before a route is shared with an eBGP neighbor

Attribute: BGP Local Preference

- Default value: 100
- “Preference” type, therefore higher = better
- Often the *only* traffic engineering knob since it’s easy and immediately effective
- Heavy hammer, a decision based on LP ignores all other route attributes, especially AS Path Length
- Should be last resort for when the metric system doesn’t suffice
- Use when a hard shift of all traffic is needed, e.g. to “cost out” use of a link in its entirety

Example: Using Local Preference



Local Pref to 10.0.4.0/24 via AS3

Border 1

Metric	LP	AS Path
0	100	1 4
0	200	3 5 4

Border 2

Metric	LP	AS Path
0	100	2 4
0	200	3 5 4

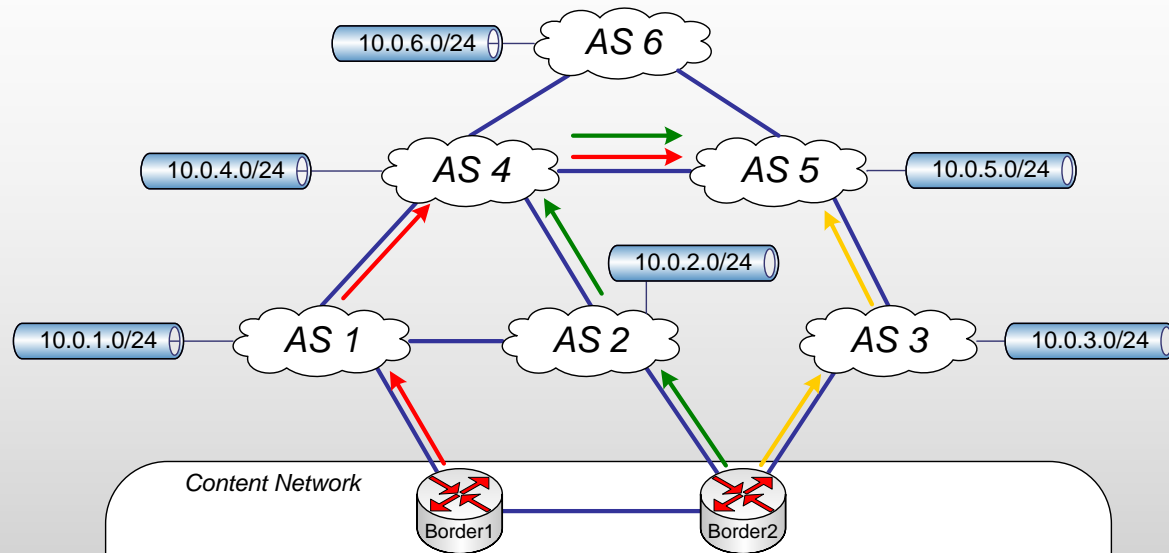
Result: Force longer path with LP

Attribute: AS Path Length

- “Count” type, therefore lower count = better
- Typically, but not always, less ASN hops means better network path
- This is the only available indicator of route or network path quality across the public Internet
- We like this to be the first selection criteria
- It’s not perfect, but it’s all we got!

- Bonus: networks can use AS prepends to hint about less desirable paths

Example: AS-Path Selection



Base Path to 10.0.5.0/24

Border 1

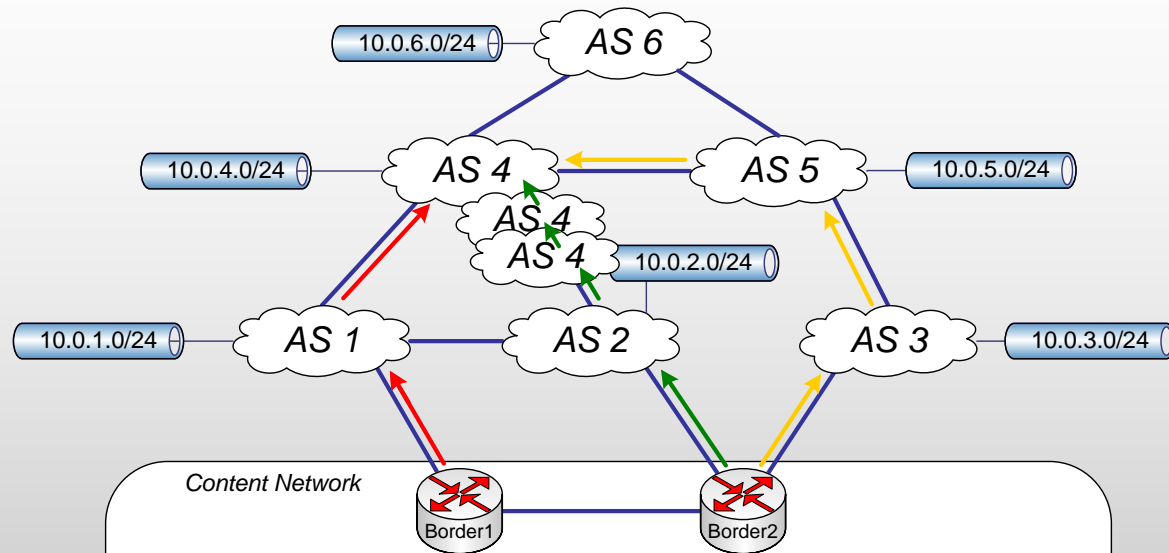
Metric	LP	AS Path
0	100	1 4 5
0	100	3 5

Border 2

Metric	LP	AS Path
0	100	2 4 5
0	100	3 5

Result: Clear winner

Example: Remote Prepend



10.0.4.0/24 With AS4 → AS2 Prepend

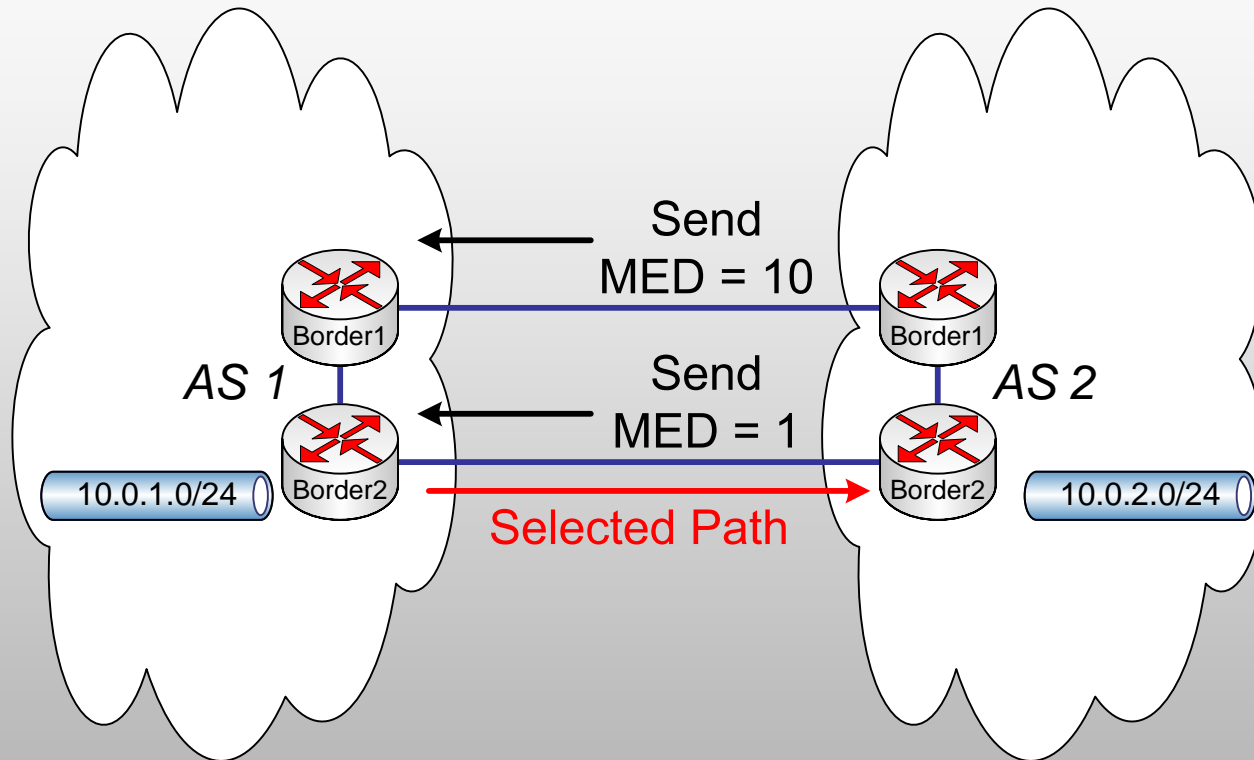
Border 1			Border 2		
Metric	LP	AS Path	Metric	LP	AS Path
0	100	1 4	0	100	1 4
			0	100	2 4 4 4
			0	100	3 5 4

Result: AS2 will not be used

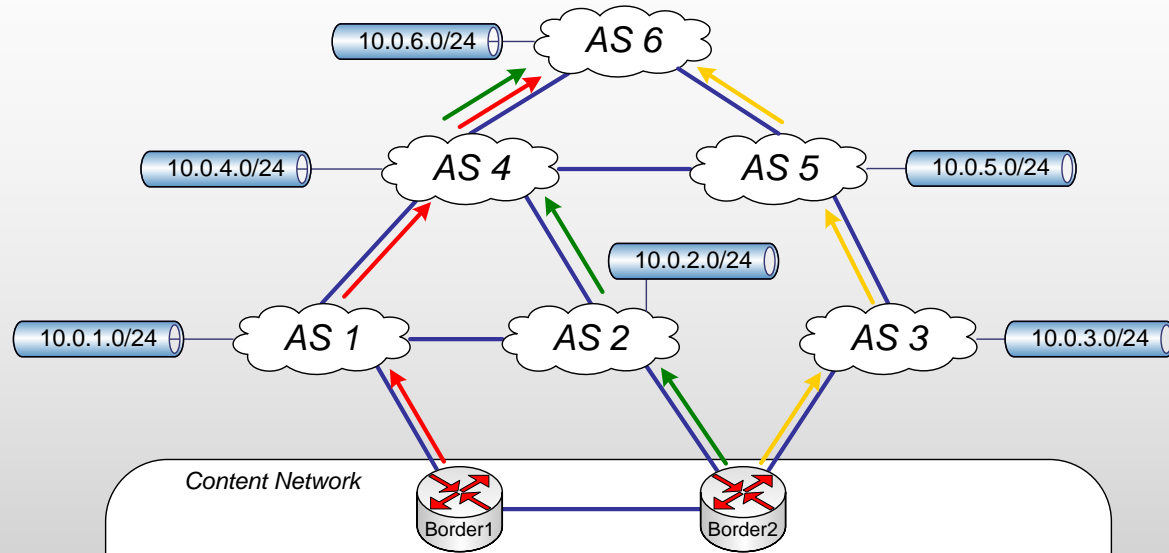
Attribute: MED or “BGP Metric”

- Default value: null (zero)
- “Cost” type, therefore lower = better
- MED = “multi-exit discriminator,” initially created to allow one ASN to tell the other ASN how they prefer to receive traffic if multiple paths (e.g. “cold potato” routing)
- Historically, the MED only consulted if same next-hop ASNs
- Since this shows up labeled as “metric” in “show route” output, we’ll call it “metric” from now on
- Key to the “Metric System” – use “metric” as a way to break ties between diverse ASNs

Example: Historical MED Use



Example: The Metric System



Base Path to 10.0.6.0/24

Border 1

Metric	LP	AS Path
100	100	1 4 6

Border 2

Metric	LP	AS Path
100	100	1 4 6
200	100	2 4 6
300	100	3 5 6

Result: Lowest metric (cost) selected

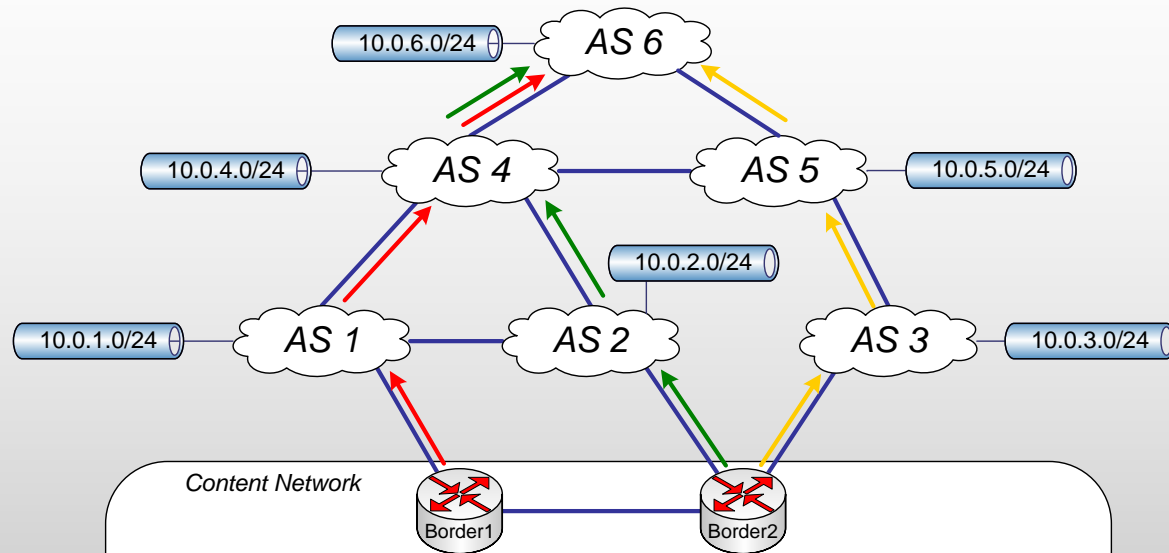
Attribute: Origin Code

- Origin code was initially intended as a “hint” to tell how the route entered into the local BGP table:
 - Internal
 - External
 - Other / Unknown
- This attribute isn’t used consistently on the Internet
 - One major US carrier: 0 (zero) count “Ext” or “Unk”
 - Another: 1367 “Ext” (0.5%), 22921 “Unk” (8.5%)
 - Another: 4007 “Ext” (1.5%), 22220 “Unk” (8.2%)
- Since use is arbitrary, determined safe to ignore

Why We Need to Fix Origin Code

- Relevant path selection section:
 - *BGP Local Preference*
 - *AS Path Length*
 - *BGP Origin Code (Int. > Ext. > Unknown)*
 - *MED (let's call this "BGP metric")*
- Because Origin is inspected *before* BGP Metric, it will interfere with the metric-based path determination

Example: Origin Interference



Simple "Metric System" to 10.0.6.0/24, different Origins

Border 1				Border 2			
Metric	LP	AS Path	Origin	Metric	LP	AS Path	Origin
100	100	1 4 6	Ext	200	100	2 4 6	Int
200	100	2 4 6	Int	300	100	3 5 6	Int

Result: Origin causes better metric to be ignored

Introducing: The “Metric System”

Goals of the “Metric System”

- Save costs by ensuring commits are met and overage is sent to less expensive ISP uplink
- Improve performance: typically, but not always, less ASN hops means better network path
- Consistent, predictable results
- Deterministic, not subject to arbitrary shifts
- Ease of administration
- Minimize manual adjustments and therefore engineer and administration time
- More “bang for your buck”
- Multi-vendor support

Are You Convinced?

- Okay, you're sold – so what's next?
 - Network preparation
 - Base configurations
 - Tie breaker selection
 - *Optional*: Additional tweaks

Preparations

Prerequisites

- Routers should have enough RAM to hold multiple full BGP tables (soft-reconfiguration), can also be effected with slightly abbreviated route tables (e.g. 256k workaround)
- Costs or capacity of transport (WAN links) are not considered, usually WAN “backhaul” is undesired
- This method works best in a network that can be divided into (usually regional) ISP transit “islands” that are usually limited to a metro area
- Network staff should stay current in communities such as NANOG/ARIN, etc.

Get Prepared

- If you're not already getting the full Internet table from all of your upstream ISPs, request it
- Request communities from your upstreams
- Request community definitions from your upstreams – this can sometimes be a challenge
- Configure metric (MED) comparison across diverse next-hop ASNs
- Configure “soft-reconfiguration inbound” if your platform requires it
- Identify best times to make changes and schedule change windows if necessary
- Clean-up / unify existing routing policy

Get Prepared: Communities

- Community tags are additional attributes associated with a route announcement that can provide useful additional information such as:
 - Type of route (peer, customer, etc.)
 - Location of route (city, state, country, region etc.)
- All ISPs use community tags internally for their own route identification purposes
- Most major ISPs do **not** send communities to their downstreams by default, but will do so on request
- There is no standard format – communities are pretty useless without definitions

Example: Community Definitions

- Actual communities used by ISPs:

Definition	nLayer	Level3
Customer, Los Angeles	4436:41718	3356:123, 3356:2003, 3356:575, 3356:3
Private Peer, Amsterdam	4436:32220	3356:666, 3356:2067, 3356:503, 3356:2

- You may have to refer to these later for fine tuning
- Definitions can be found in various places:
 - ISP website: e.g. http://nlayer.net/bgp_communities
 - Routing registry: e.g. whois -h whois.radb.net as3356
 - Ask your ISP's support staff or sales engineer
 - Unofficial collection: <http://onesc.net/communities/>

Get Prepared: Base Configuration

- Cisco:

```
# Enter BGP configuration mode
```

```
router bgp <ASNUM>
```

```
# Enable "metric" comparisons for diverse ASNs
```

```
bgp always-compare-med
```

```
# Configure "safety net" against missing metrics
```

```
bgp bestpath med missing-as-worst
```

```
# Enable received-route storage
```

```
neighbor <PeerIP> soft-reconfiguration inbound
```

```
# Enable ASN:ID format for BGP communities
```

```
ip bgp-community new-format
```

Get Prepared: Base Configuration

- Foundry:

```
# Enter BGP configuration mode
```

```
router bgp
```

```
# Enable "metric" comparisons for diverse ASNs
```

```
always-compare-med
```

```
# Configure "safety net" against missing metrics
```

```
med-missing-as-worst
```

```
# Enable received-route storage
```

```
neighbor <PeerIP> soft-reconfiguration inbound
```

Get Prepared: Base Configuration

- Force10:

```
# Enter BGP configuration mode
```

```
router bgp <ASNUM>
```

```
# Enable "metric" comparisons for diverse ASNs
```

```
bgp always-compare-med
```

```
# Enable received-route storage
```

```
neighbor <PeerIP> soft-reconfiguration inbound
```

- *Note: missing metric is treated as "worst" (highest value) by default*

Get Prepared: Base Configuration

- Juniper:

```
protocols {  
  # Enter BGP configuration  
  bgp {  
    # Enable "metric" comparisons for diverse ASNs  
    path-selection always-compare-med;  
  }  
}
```

- *Note: "soft config" is enabled by default*

Get Prepared: Identify Times

- Traffic balancing for billing purposes comes down to balancing peak traffic periods
- See daily / weekly / monthly historical graphs and identify peak times
- Prepare your management and operations staff for changes during peak periods – *this may take some finesse!*
- You may have to make some practice runs during off-peak to demonstrate that this can be done at peak periods without service disruptions

Get Prepared: Clean Up

- “Scrub in” for surgery by cleaning out all existing routing policies and starting with a base “catchall” policy that does the following
 - Accept all routes
 - Reset origin to “Int”
 - Reset metric to 0
- Once this is done, you’re ready to proceed with metric-based balancing

Note on Remaining Examples

- Remaining examples will be Cisco IOS and Juniper JunOS only
- Foundry, Force10 are very similar to Cisco, please check documentation and test first
- There are many ways to write routing policies, and many different styles available, be prepared to adapt to a precedence already set within your environment

Example: Base Policy (IOS)

```
# Enter route-map mode
route-map ROUTES-IN-FROM-ISP1 permit 1000
  description Catchall policy for route defaults
  set metric 0
  set origin igp

# Apply to neighbor
router bgp <ASNUM>
  neighbor <PeerIP> route-map ROUTES-IN-FROM-ISP1 in
```

Example: Base Policy (JunOS)

```
# Enter policy mode
policy-options {
  policy-statement SET-ROUTE-ORIGIN {
    then {
      origin igp;
      next policy;
    }
  }
  policy-statement IMPORT-FROM-ISP1 {
    term CATCHALL {
      then {
        metric 0;
        accept;
      }
    }
  }
}

# Apply to neighbor
protocols {
  bgp {
    group ISP1 {
      import [ SET-ROUTE-ORIGIN IMPORT-FROM-ISP1 ];
    }
  }
}
```

Metric System Configurations

Ready to Begin

- Once preparations are completed, begin “metric system”
- First Stage: Equalize the playing field
 - Fix “route origin” attribute
 - May have to add some AS-path prepends
- Second Stage: Broad Strokes
 - Set AS hop count tie breakers
 - General selection based on ISP-provided information (e.g. community tags)
- Third Stage: Destination Networks
 - Must understand target networks
 - Value of and need for flow-based statistics
 - ASN better than prefix
- Final Stage: Fine Tuning

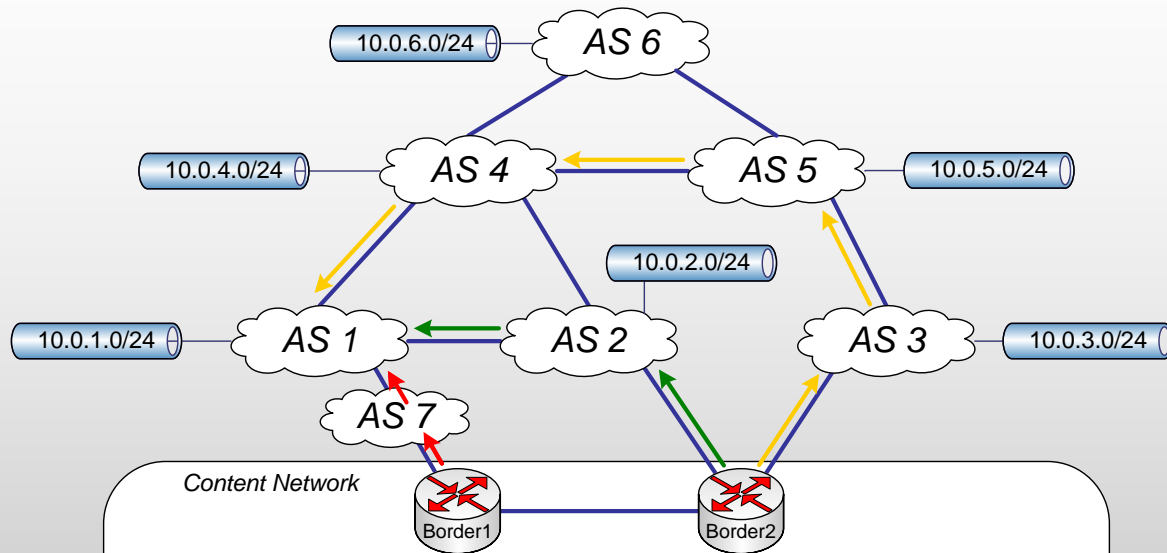
Sequence of Events

- **First Stage: Equalize the playing field**
 - **Fix “route origin” attribute**
 - **May have to add some AS-path prepends**
- Second Stage: Broad Strokes
 - Set AS hop count tie breakers
 - General selection based on ISP-provided information (e.g. community tags)
- Third Stage: Destination Networks
 - Must understand target networks
 - Value of and need for flow-based statistics
 - ASN better than prefix
- Final Stage: Fine Tuning

Equalize the Playing Field

- Fix the “route origin” attribute
 - Do this during “clean-up” stage, double-check this is complete
- Does one of your ISPs inject an extra “local” ASN before you get to their “backbone” ASN?
 - This will artificially pad AS-Path to this ISP
 - If so, add one prepend to all other ASNs
- Take baseline measurements of traffic levels and network performance

Example: Extra ASN

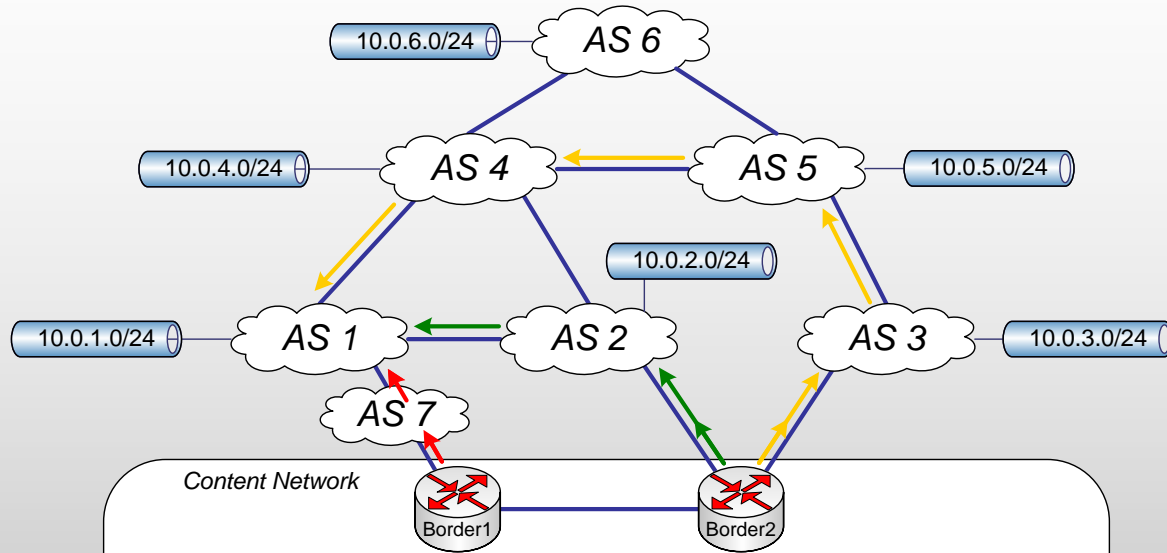


“Extra AS” Effect to 10.0.1.0/24

Border 1			Border 2		
Metric	LP	AS Path	Metric	LP	AS Path
0	100	7 1	0	100	2 1
			0	100	3 5 4 1

Result: May sway results away from AS 1

Result: Fix Extra ASN



“Extra AS” Equalize Effect to 10.0.1.0/24

Border 1			Border 2		
Metric	LP	AS Path	Metric	LP	AS Path
0	100	7 1	0	100	7 1
			0	100	2 2 1
			0	100	3 3 5 4 1

Result: AS1 via AS7 is preserved

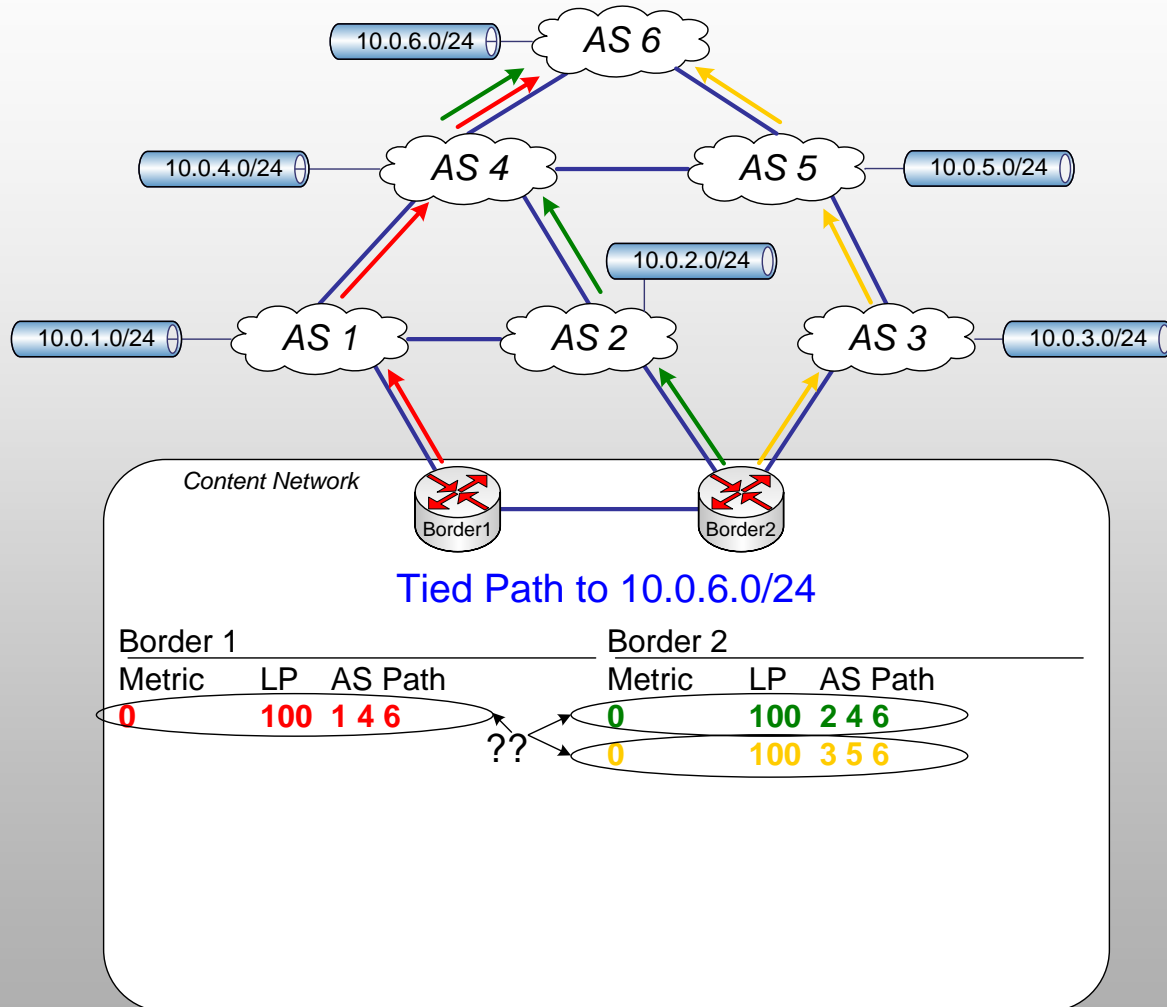
Sequence of Events

- First Stage: Equalize the playing field
 - Fix “route origin” attribute
 - May have to add some AS-path prepends
- **Second Stage: Broad Strokes**
 - **Set AS hop count tie breakers**
 - **General selection based on ISP-provided information (e.g. community tags)**
- Third Stage: Destination Networks
 - Must understand target networks
 - Value of and need for flow-based statistics
 - ASN better than prefix
- Final Stage: Fine Tuning

Broad Strokes

- Set different metric values for each ISP on the “catch all” route policy to fix AS-Path ties
- Experiment with different sequence of tie breaker metrics and take link measurements each step of the way
- Decide on final tie breakers that most closely meet your ultimate goals
- If tie breakers aren't enough, add community-based decisions
- You may even be done after completing this step!

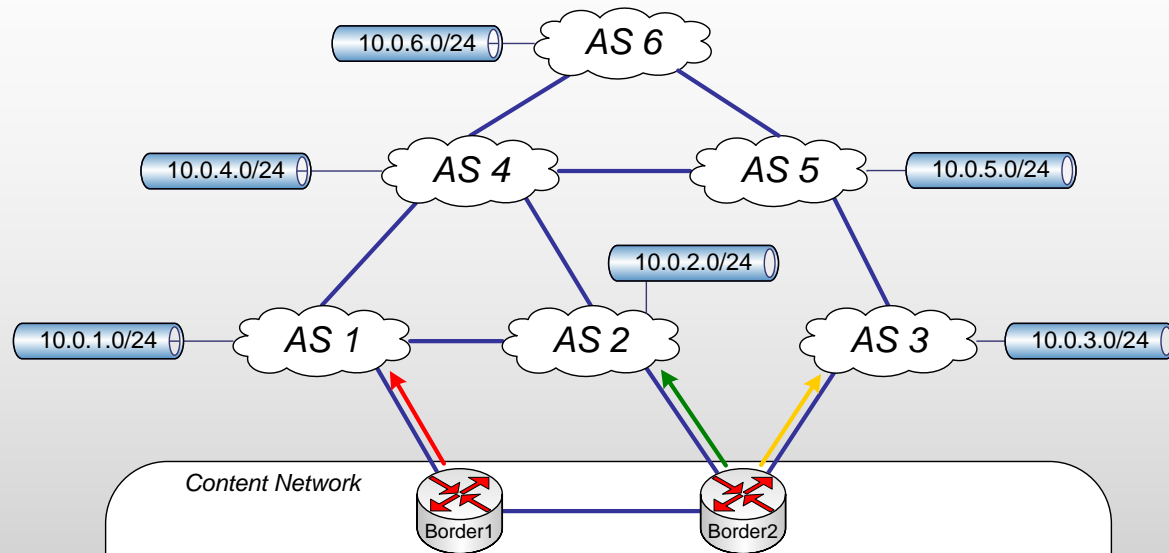
Example: AS-Path Tie



Deciding on Tie Breakers

- Understanding one definition of ISP “tier”
 - Contentious term that has some technical relevance but has been exploited by marketing
 - Think in terms of next-hop AS reachability/count
- Observe AS-Path differences to some of the larger ISPs and eyeballs
 - Poke around at the BGP table from your ISPs, look at your 5 largest target networks (if known)
- Compare traffic / performance measurements at each step, allow sufficient time for best data
- If the first tie-breaking scheme doesn't work, shuffle metrics around and try again

Example: Tie Breaker to AS 1



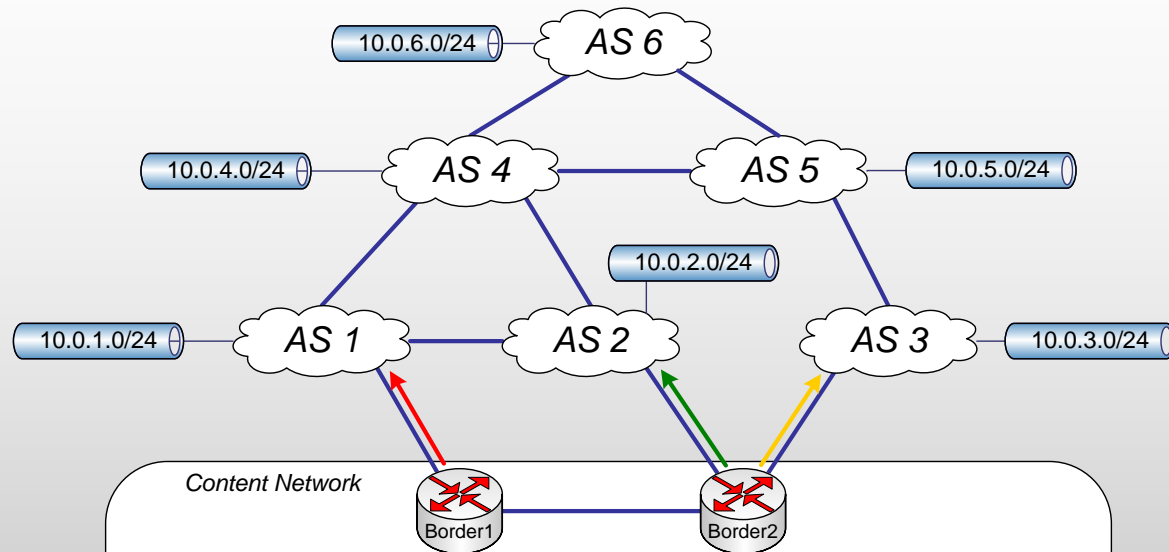
Catchall AS1: 100, AS2: 200, AS3: 300

Selected Best Paths

Prefix	Metric	LP	AS Path
10.0.1.0/24	100	100	1
10.1.2.0/24	200	100	2
10.1.3.0/24	300	100	3
10.1.4.0/24	100	100	1 4
10.1.5.0/64	300	100	3 5
10.1.6.0/24	100	100	1 4 6

Results: More paths via AS1

Example: Tie Breaker to AS 2



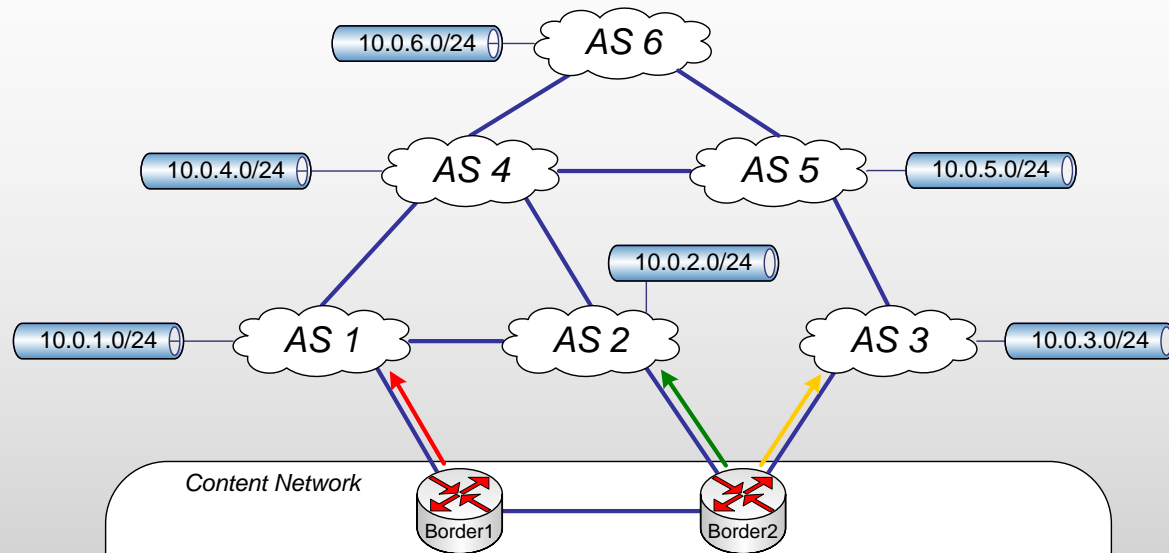
Catchall AS1: 200, AS2: 100, AS3: 300

Selected Best Paths

Prefix	Metric	LP	AS Path
10.0.1.0/24	200	100	1
10.1.2.0/24	100	100	2
10.1.3.0/24	300	100	3
10.1.4.0/24	100	100	2 4
10.1.5.0/64	300	100	3 5
10.1.6.0/24	100	100	2 4 6

Results: More paths via AS2

Example: Tie Breaker to AS 3



Catchall AS1: 200, AS2: 300, AS3: 100

Selected Best Paths

Prefix	Metric	LP	AS Path
10.0.1.0/24	200	100	1
10.1.2.0/24	300	100	2
10.1.3.0/24	100	100	3
10.1.4.0/24	200	100	1 4
10.1.5.0/64	100	100	3 5
10.1.6.0/24	100	100	3 5 6

Results: More paths via AS3, but watch 10.1.4.0/24!

Config: Tie Breakers (IOS)

```
route-map ROUTES-IN-FROM-ISP1 permit 1000
  description Catchall policy for route defaults
  set metric 100
  set origin igp
```

```
route-map ROUTES-IN-FROM-ISP2 permit 1000
  description Catchall policy for route defaults
  set metric 200
  set origin igp
```

```
route-map ROUTES-IN-FROM-ISP3 permit 1000
  description Catchall policy for route defaults
  set metric 300
  set origin igp
```

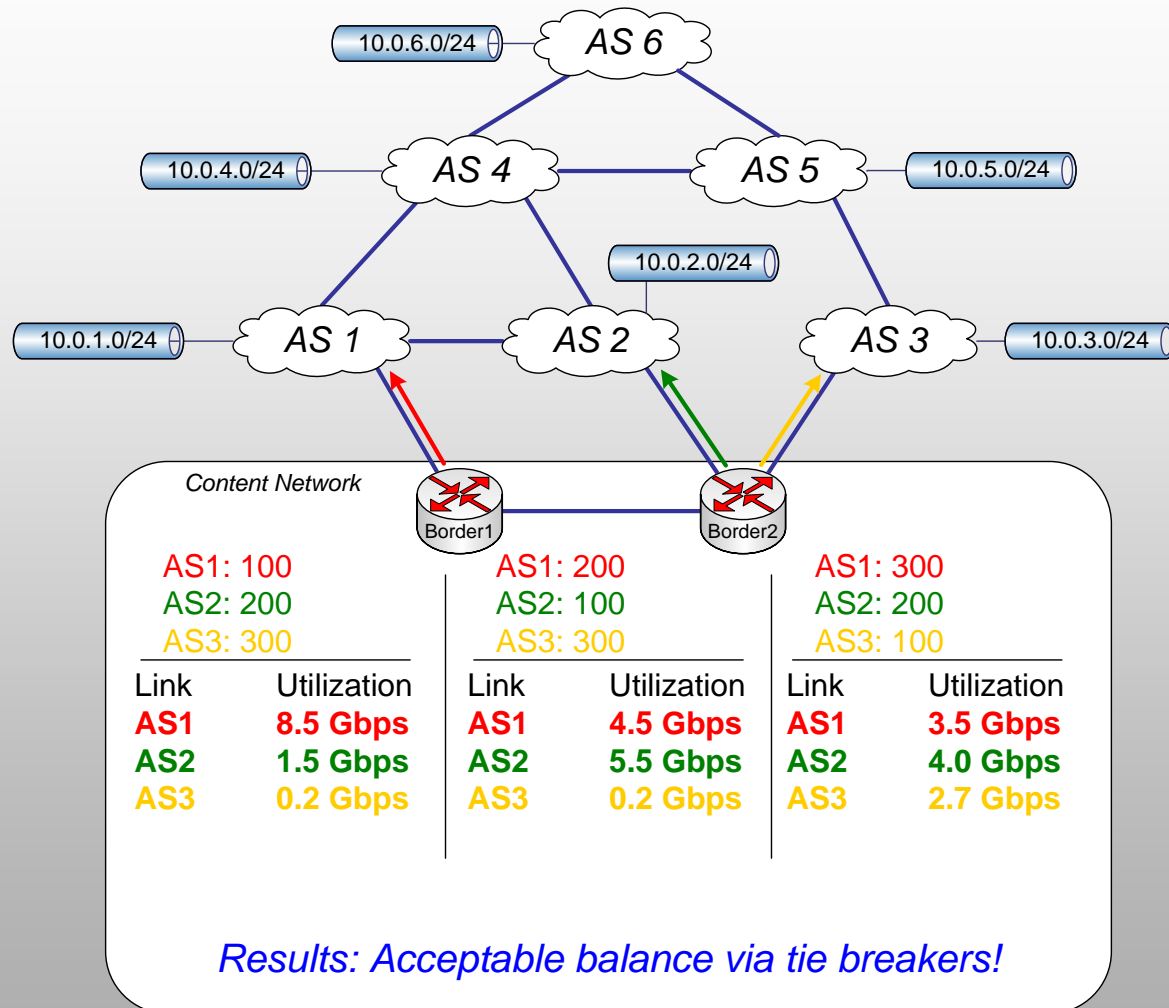
Config: Tie Breakers (JunOS)

```
policy-statement IMPORT-FROM-ISP1 {
    term CATCHALL {
        then {
            metric 100;
            accept;
        }
    }
}

policy-statement IMPORT-FROM-ISP2 {
    term CATCHALL {
        then {
            metric 200;
            accept;
        }
    }
}

policy-statement IMPORT-FROM-ISP3 {
    term CATCHALL {
        then {
            metric 300;
            accept;
        }
    }
}
```

Actual Numbers: Tie Breakers



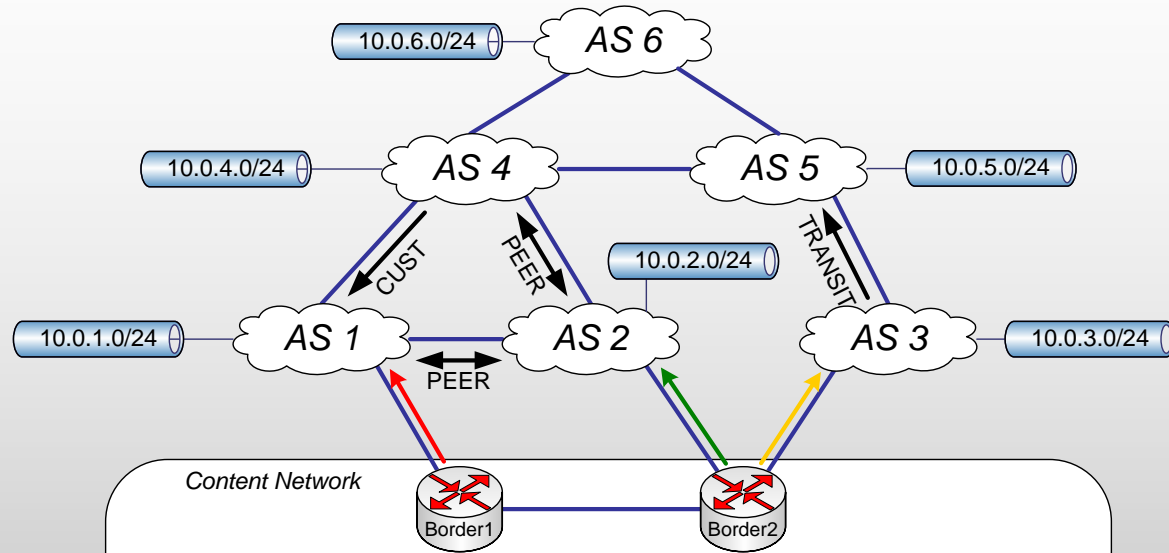
Failure Scenario: Tie Breakers

- Pay attention to the balance between the secondary / tertiary / etc. tie breakers
- If the lowest-cost tie breaker were to go down, would the resulting balance be satisfactory?
- Schedule and perform a test to confirm, and adjust if necessary

Broad Stroke: Communities

- Sometimes if you have two similar upstream ISPs, too many as-hop counts are equal, and too much traffic will be sent to the tie breaker
- Use broad community definitions to help balance, select one of these criteria and sway decision towards one ISP
 - Peer vs. Customer vs. Transit
 - Country / Continent / Region

Example: Using Communities



Communities Used

AS1 Customer:	1:100
AS1 Peer:	1:200
AS2 Customer:	2:555
AS2 Peer:	2:666
AS3 Transit:	3:4000

Config: Communities (IOS)

```
ip community-list standard AS1-CUST permit 1:100
ip community-list standard AS3-TRANSIT permit 3:4000
```

```
route-map ROUTES-IN-FROM-ISP1 permit 20
  description Prefer ISP1's Customers
  match community AS1-CUST
  set metric 50 ! better than lowest tie breaker (100)
  set origin igp
```

```
route-map ROUTES-IN-FROM-ISP3 permit 20
  description De-pref ISP3's Transit
  match community AS3-TRANSIT
  set metric 400 ! worse than highest tie breaker (300)
  set origin igp
```

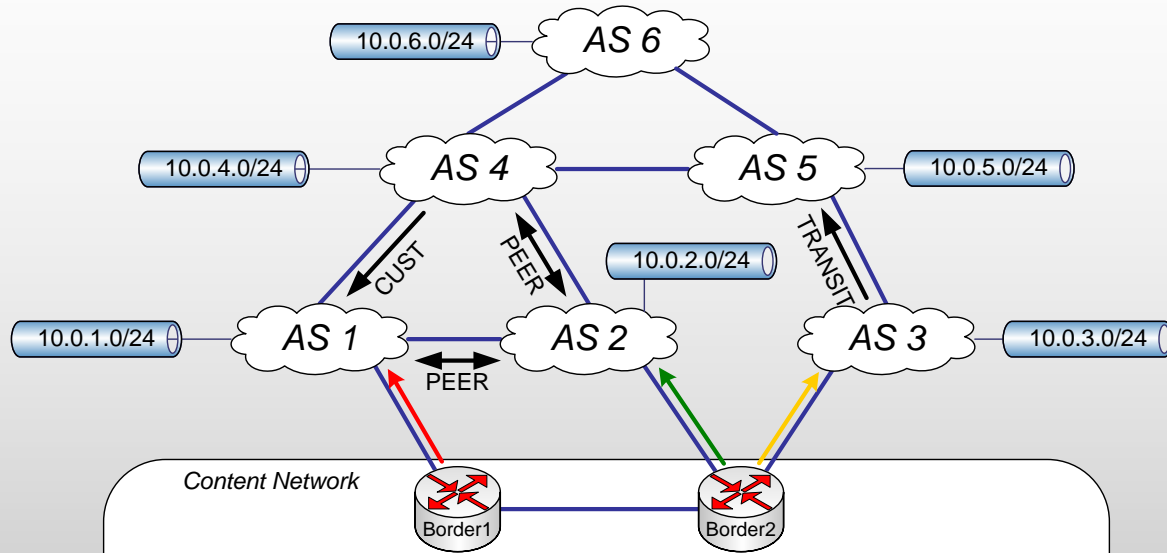
Config: Communities (JunOS)

```
community AS1-CUST members 1:100
community AS3-TRANSIT members 3:4000
```

```
policy-statement IMPORT-FROM-ISP1 {
    term PREFER-AS1-CUST {
        from community AS1-CUST ;
        then {
            metric 50; # better than lowest tie breaker (100)
            accept;
```

```
policy-statement IMPORT-FROM-ISP3 {
    term DEPREF-AS3-TRANSIT {
        from community AS3-TRANSIT ;
        then {
            metric 400; # worse than highest tie breaker (300)
            accept;
```

Example Base: Communities



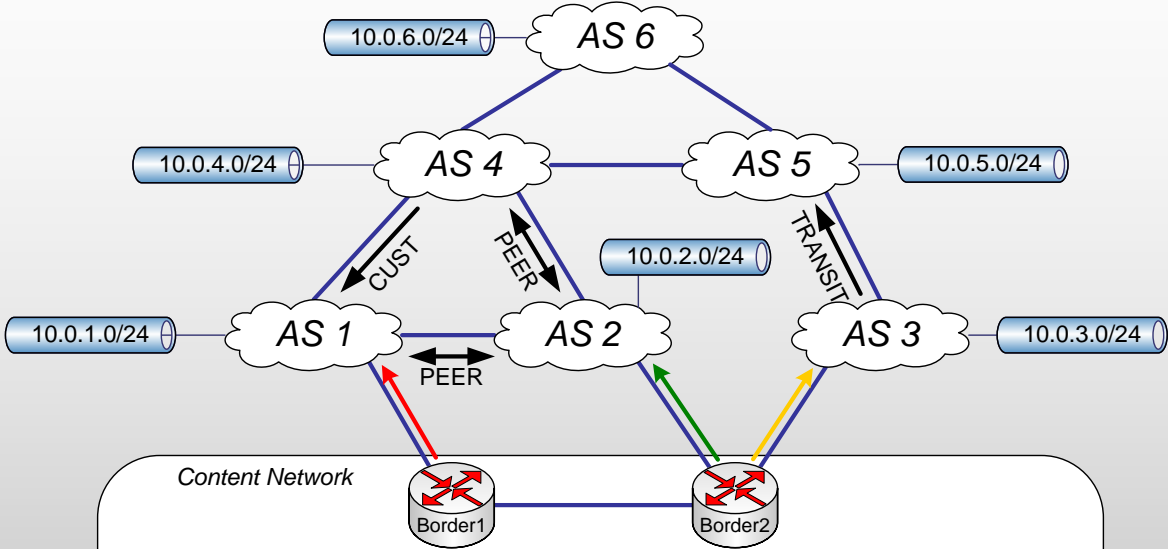
Catchall AS1: 300, AS2: 200, AS3: 100

Selected Best Paths

Prefix	Metric	LP	AS Path
10.0.1.0/24	300	100	1
10.1.2.0/24	200	100	2
10.1.3.0/24	100	100	3
10.1.4.0/24	200	100	2 4 <i>DON'T LIKE THIS...</i>
10.1.5.0/64	100	100	3 5
10.1.6.0/24	100	100	3 5 6 <i>DON'T LIKE THIS...</i>

Baseline before community-based policies

Result: Communities



Catchall AS1: 300 + pref cust, AS2: 200

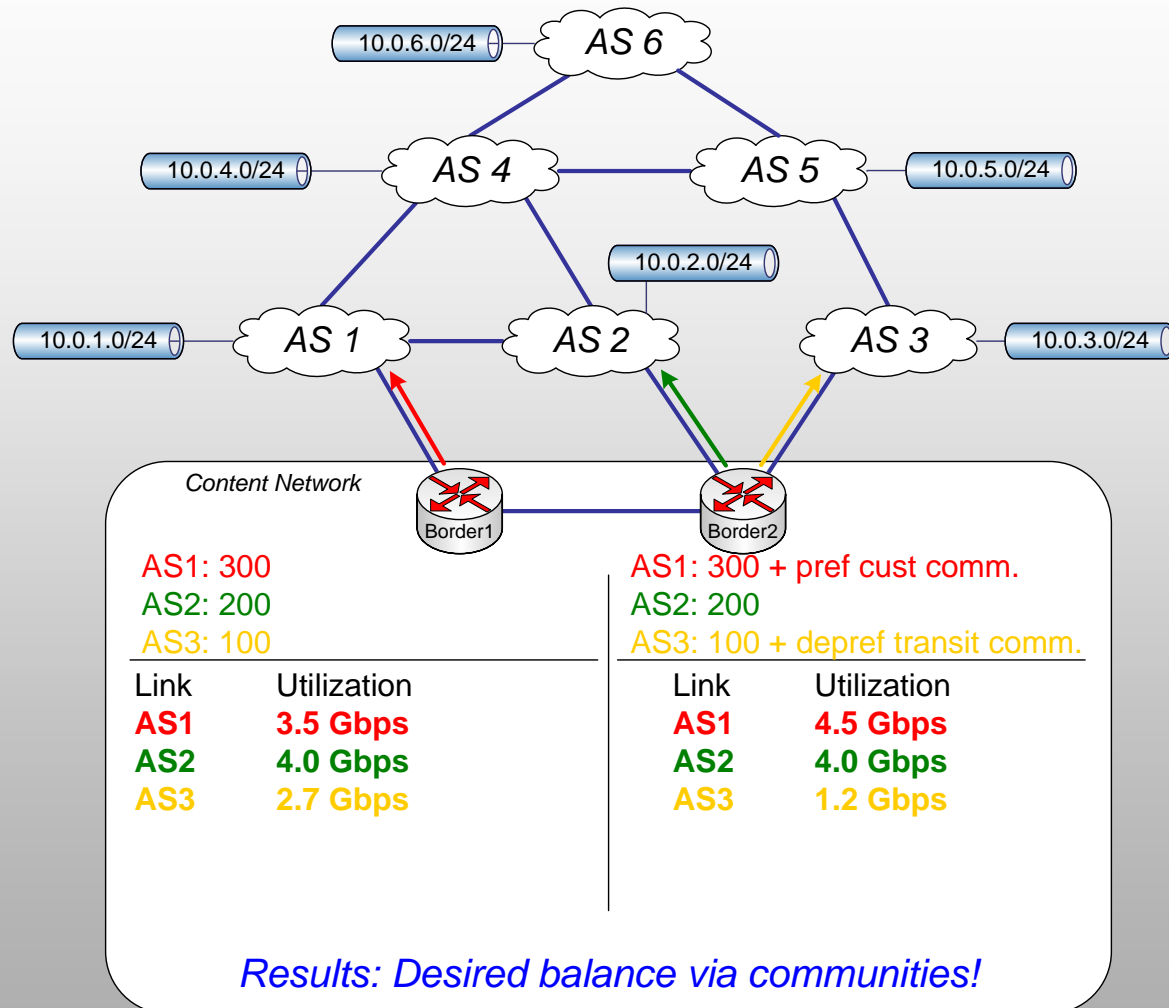
AS3: 100 +depref transit

Selected Best Paths

Prefix	Metric	LP	AS Path
10.0.1.0/24	300	100	1
10.1.2.0/24	200	100	2
10.1.3.0/24	100	100	3
10.1.4.0/24	50	100	1 4 CHANGED!
10.1.5.0/64	100	100	3 5
10.1.6.0/24	200	100	2 4 6 CHANGED!

Results: Shifted two paths

Actual Numbers: Communities



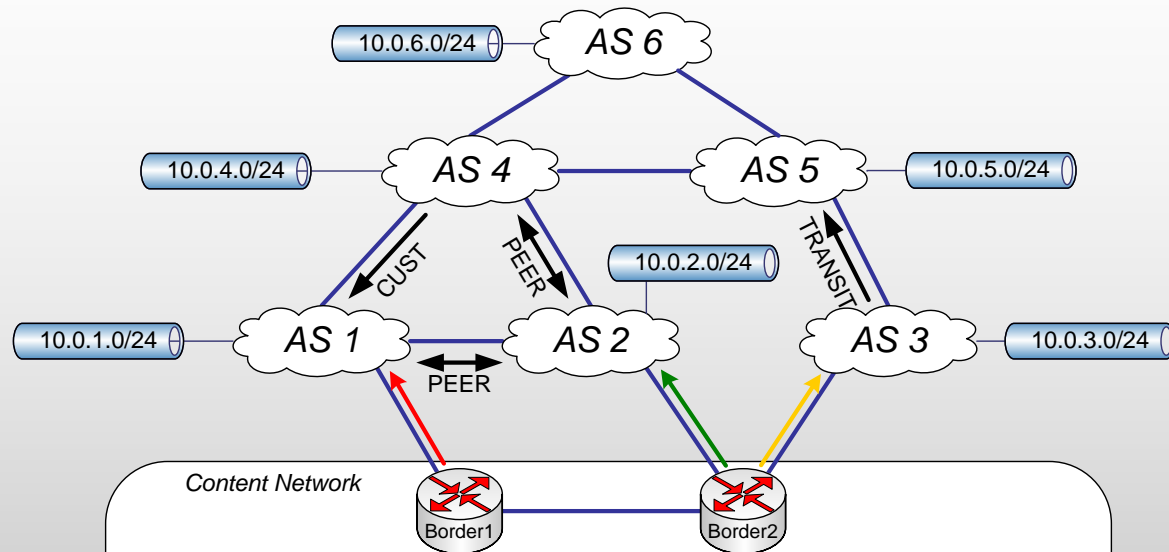
Sequence of Events

- First Stage: Equalize the playing field
 - Fix “route origin” attribute
 - May have to add some AS-path prepends
- Second Stage: Broad Strokes
 - Set AS hop count tie breakers
 - General selection based on ISP-provided information (e.g. community tags)
- **Third Stage: Destination Networks**
 - **Must understand target networks**
 - **Value of and need for flow-based statistics**
 - **ASN better than prefix**
- Final Stage: Fine Tuning

Destination Networks

- If tie breakers and broad strokes aren't achieving desired results, the next focus should be on specific destination networks
- Most networks won't need to go into this level of detail, this work is a bit more demanding
 - You should understand your largest target networks
 - Usually have to use flow-based data to identify and collect bandwidth utilization statistics for top ~ 20 destination ASNs
- Better to make changes based on destination ASNs versus focusing on IP prefixes, since IP addresses are more likely to change without notice

Example: Destination Networks



Catchall AS1: 300, AS2: 200, AS3: 100 +depref transit

Available Paths to 10.1.6.0/24

Prefix	Metric	LP	AS Path
10.1.6.0/24	300	100	1 4 6
10.1.6.0/24	200	100	2 4 6
10.1.6.0/24	400	100	3 5 6

Problem: Want to use AS1 to get to AS6

Config: Destination Nets (IOS)

```
ip as-path access-list 1 permit _6_
```

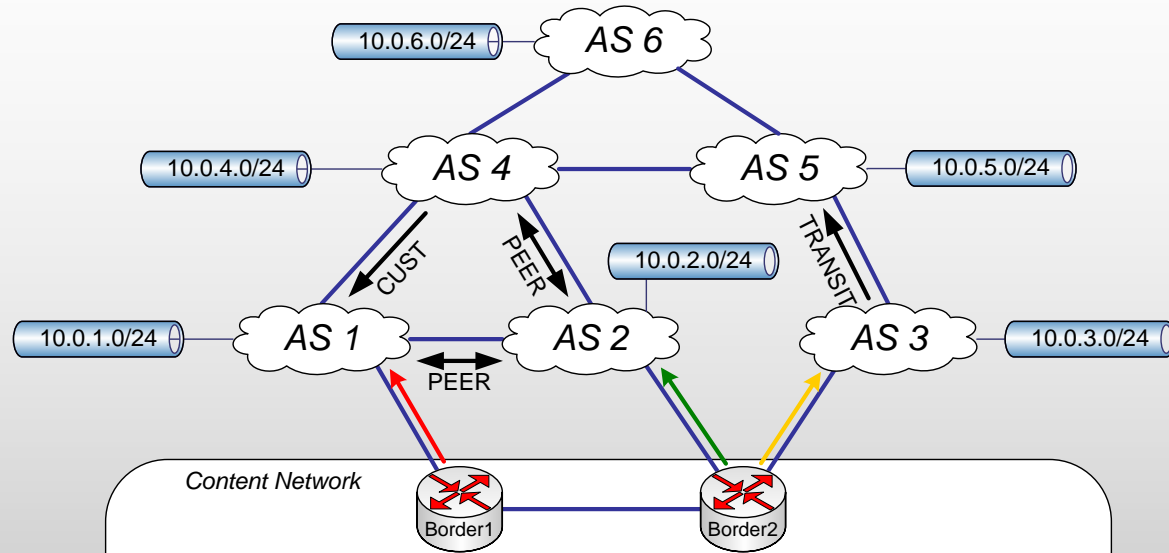
```
route-map ROUTES-IN-FROM-ISP1 permit 30  
  description Prefer AS6 via AS1  
  match as-path 1  
  set metric 50  
  set origin igp
```

Config: Destination Nets (JunOS)

```
as-path AS6 .*6.*
```

```
policy-statement IMPORT-FROM-ISP1 {  
    term PREFER-AS6-VIA-AS1 {  
        from as-path AS6 ;  
        then {  
            metric 50;  
            accept;
```

Result: Destination Networks



Content Network

Catchall AS1: 300 + Pref AS 6, AS2: 200,
 Available Paths to 10.1.6.0/24 AS3: 100 +depref transit

Prefix	Metric	LP	AS Path
10.1.6.0/24	50	100	1 4 6
10.1.6.0/24	200	100	2 4 6
10.1.6.0/24	400	100	3 5 6

Result: Using AS1 to get to AS6

Gotchas: Destination Networks

- Overall, try and minimize this step, focus on the broad strokes
- Network engineers must stay up-to-date, as the Internet is fluid and changes occur without notice, e.g. large eyeball backbone shifts
- Shift only one at a time, observe results, if unsatisfactory, be sure to undo your changes before moving on to the next target
- If you don't have useful flow data, try using a backup link and place just one ASN at a time on that path, if you don't have a backup link that is normally idle, you can remove all traffic from one temporarily

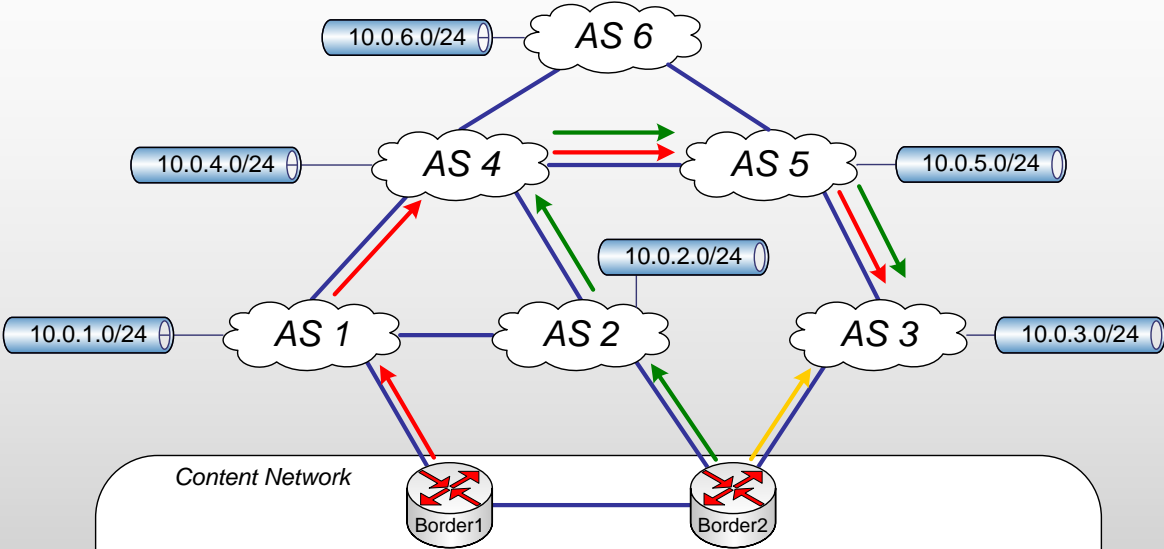
Sequence of Events

- First Stage: Equalize the playing field
 - Fix “route origin” attribute
 - May have to add some AS-path prepends
- Second Stage: Broad Strokes
 - Set AS hop count tie breakers
 - General selection based on ISP-provided information (e.g. community tags)
- Third Stage: Destination Networks
 - Must understand target networks
 - Value of and need for flow-based statistics
 - ASN better than prefix
- ***Final Stage: Fine Tuning***

Additional Fine Tuning

- There is rarely any fine tuning that is necessary once the first three stages are complete
- Due to performance concerns some additional tweaks may be necessary, where tie breaker decisions are inferior to alternate paths
- After identifying other paths, attributes to be adjusted:
 - Insert ISP's AS in prepend to shift traffic *away* from that ISP
 - When all else fails with “softer” metrics, final knob available is Local Preference: increase value
- Sometimes you may have to do this for cost trickery, too

Example: Local Preference 2



Local Pref to 10.0.3.0/24 via AS1

Border 1			Border 2		
Metric	LP	AS Path	Metric	LP	AS Path
300	200	1 4 5 3	300	200	1 4 5 3
			200	100	2 4 5 3
			100	100	3

Result: Force longer path with LP

Success!

Metric System Success

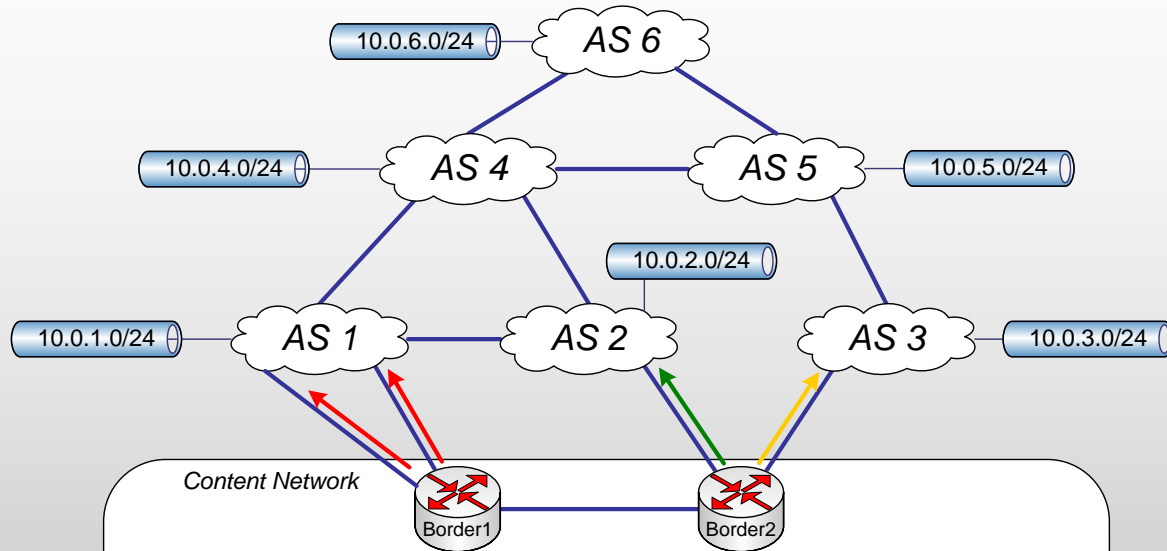
- After following the minimum necessary steps, the result should be a well-balanced network
- Future maintenance should be fairly simple
- Get monitoring system going with bandwidth and performance alarms

Alternates

Alternate Multipath Strategies

- Most modern hardware allows for automatic “multipath” balancing if connecting to the same ISP over multiple links on the same router
- Multipath Multi-AS if your hardware supports it
- Check documentation and test in a safe environment or during maintenance windows
- Not covering in detail here due to varied success on different hardware/software

Example: Multipath



Alternates: Multipath

Border 1 Path to 10.0.1.0/24

Metric	LP	AS Path
200	100	1
200	100	1

Border 2 Path to 10.0.6.0/24

Metric	LP	AS Path
100	100	2 4 6
100	100	3 5 6

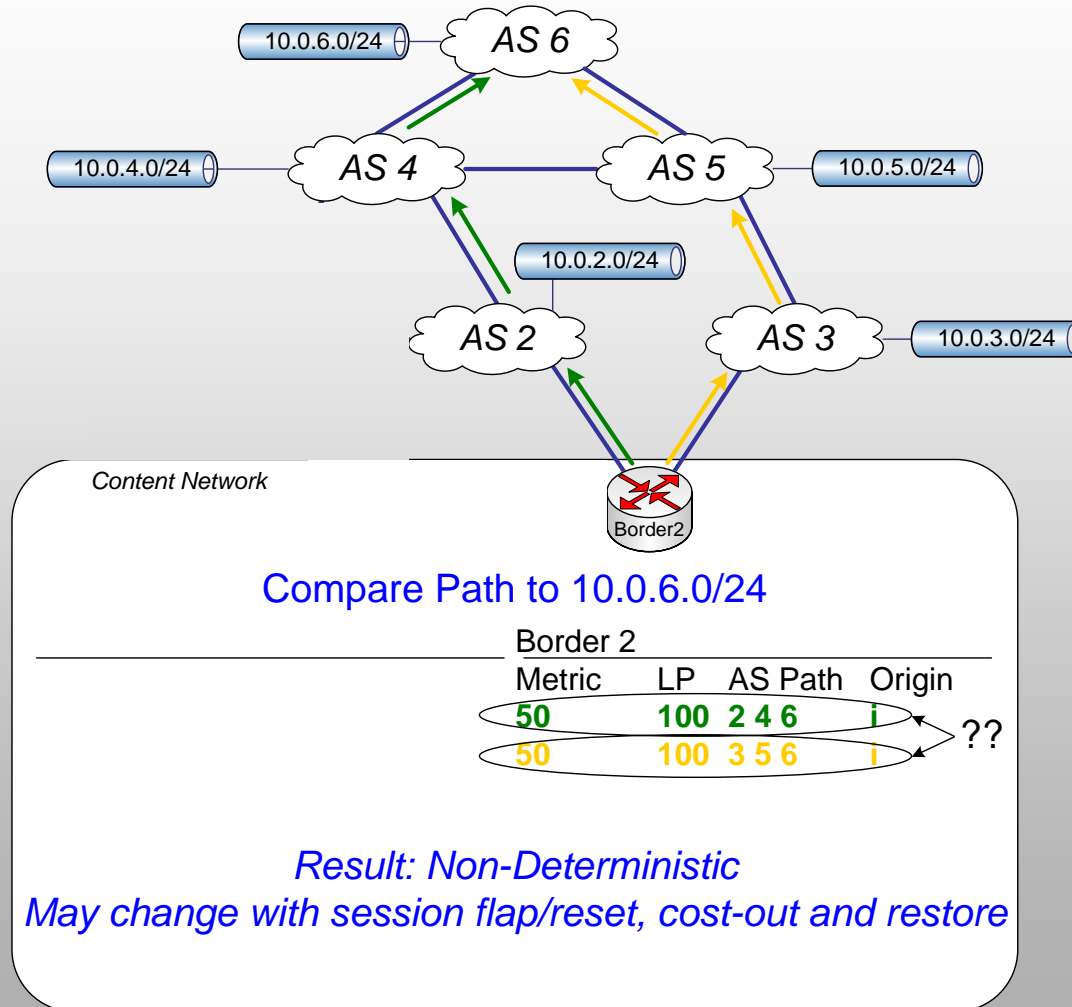
Result: Use multipath to balance same-cost paths over multiple links

Common Errors

Common Errors

- Issues that will either result in an unacceptable balance, or non-deterministic results that will lead to inconsistencies:
 - Not setting metrics for all routes from the start and relying on built-in “tie breakers”
 - Not setting *unique* (incremented) values as tie-breaker metrics
 - Inconsistent / conflicting metrics that result in different path selection on different eBGP-speaking border routers

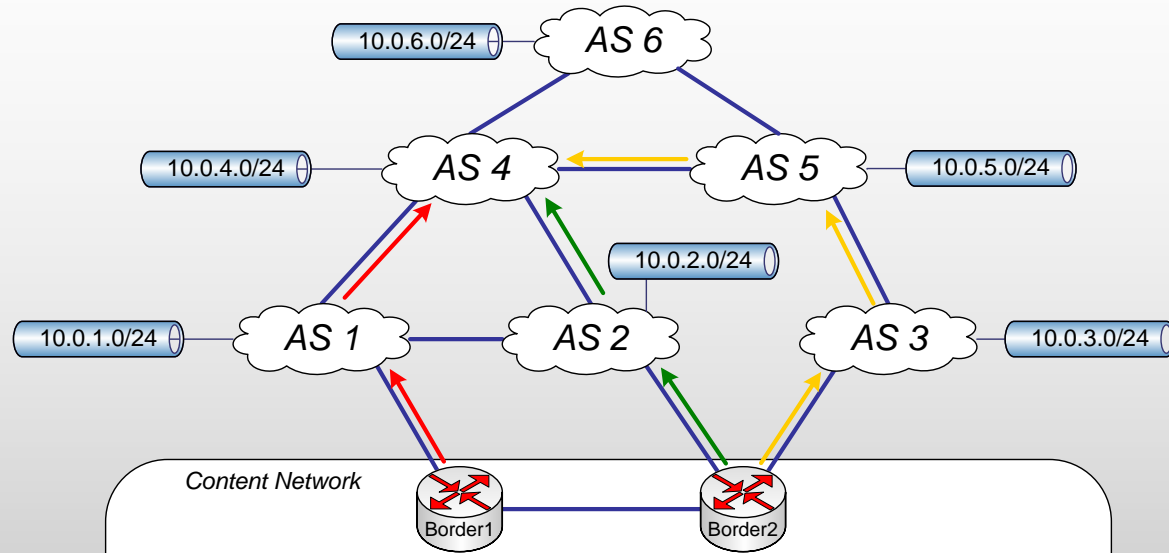
Example: Common Errors 1



Non-Deterministic Errors

- Deterministic definition:
 - having an outcome that can be predicted because all of its causes are either known or the same as those of a previous event

Example: Common Errors 2



Base Path to 10.0.4.0/24

Border 1

Metric	LP	AS Path
0	100	1 4

Border 2

Metric	LP	AS Path
0	100	2 4
0	100	3 5 4

Result: Different opinions

Different Opinions

- Having different opinions on different routers may be intentional
- We have found this confusing and inconsistent
- Usually means you're relying upon downstream ECMP for servers
- If one router goes away, traffic balance may shift more significantly than desired

Summary

- Primary decision attribute is AS length / hop count, allowing us to take advantage of the limited “performance” information that BGP can provide
- Basic balance achieved through tie-breaker adjustment
- Secondary tuning achieved through broad strokes as much as possible, e.g. based on ISP-tagged community or destination ASN
- Optional fine tuning using Local Pref (if necessary)
- Try to minimize number of fine tuning steps

Any Questions?

Thank you for listening
Peak Web Consulting is available to assist

Dani Roisman

droisman ~ at ~ peakwebconsulting ~ dot ~ com