The Problem to Solve

How do we maximize our flexibility in regards to the deployment of workloads in distinct network security zones at multiple sites, while extended zones across sites? Ideally, we should be able to:

- Deploy workload into any zone at any site
- Provide local-instances
- Stretch network-security-zones ("zones") across sites

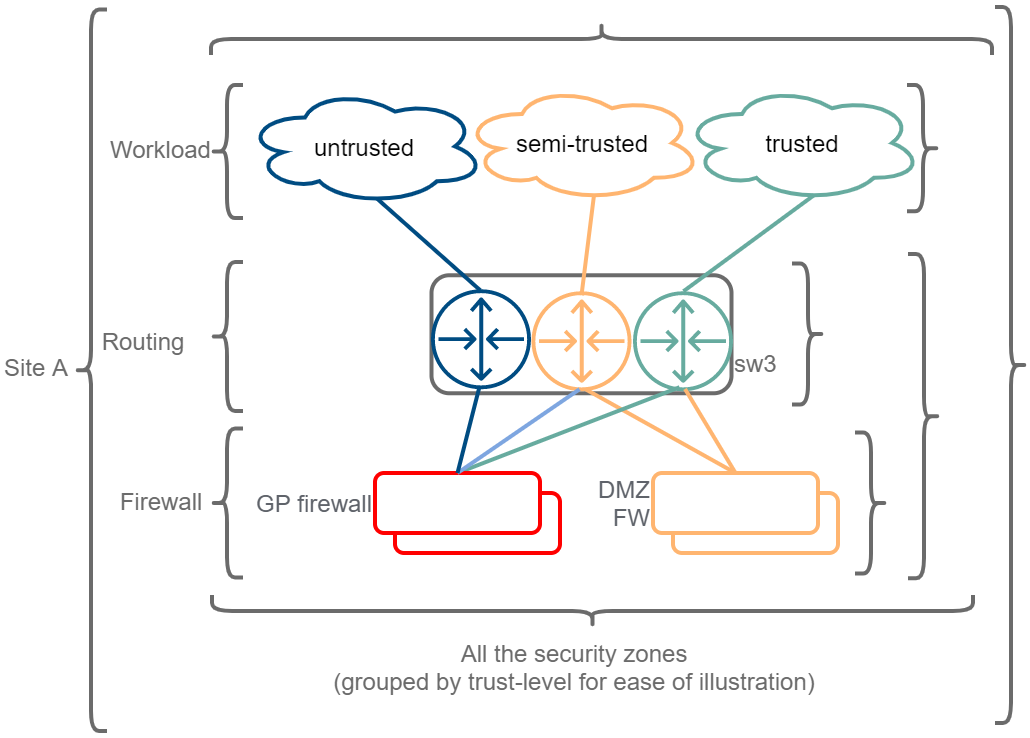
BCBSNC's current IT services deployment architecture relies on a hierarchical concept of network security zones.

- Unrusted
  - Internet
- Semi-trusted
  - B2B
  - Guest Wireless
  - BYOD wireless
  - SaaS/PaaS
  - DMZ
- Trusted
  - data-center internal
  - Internal Wireless
  - LAN (office)
  - Data-center
  - IaaS

It's not clear exactly which set of zone boundary controls are required at the edge of each, but we should assume that they will typically involve stateful inspection at Layer-4 and higher.

The existing architecture is essentially site-centric with regards to the topology of these zones and zone-boundaries. It calls for a model in which all zones are connected to a shared NGFW instance, which acts as the required "zone boundary control." Additionally, it requires the semi-trusted and Internal zones to be connected to a separate firewall instance which gates traffic between the semi-trusted and trusted zones. The addition of the dedicated DMZ FW appears to be based on an engineering design preference, rather than an architecture design constraint. Semi-trusted and internal-workload are both on the same hypervisor-based overlay network, and we wanted to avoid hairpinning traffic off of the overlay for firewall inspection, so the DMZ firewall was instantiated on the same hypervisor overlay network.)

This strategy is depicted in the following diagram

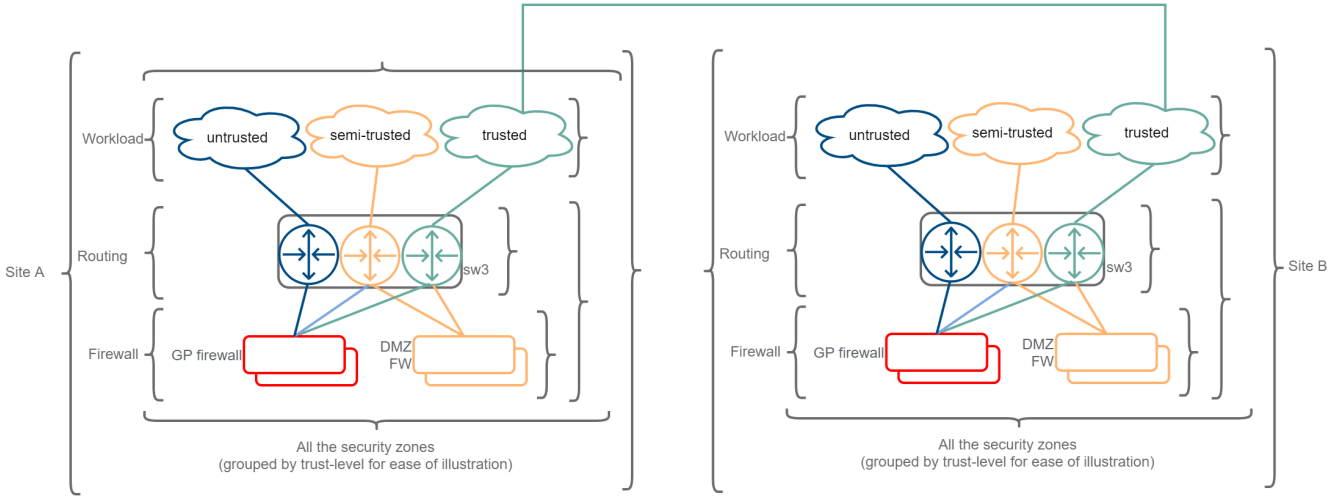


In the above model, traffic from external to DMZ zones originates in the external zone, hits the external zone router/VRF, and is routed to the GP (general purpose) firewall. From there it is routed to the DMZ router/VRF, and on to the host in the DMZ workload area. The DMZ zone is intended to host proxy functions for any ingress traffic flows from external zones to internal zones.

The connections proxied by DMZ workload create flows from the DMZ zone to the Internal zone. This path takes the DMZ VRF/router to the DMZ FW to the Internal VRF/router to the internal destination.

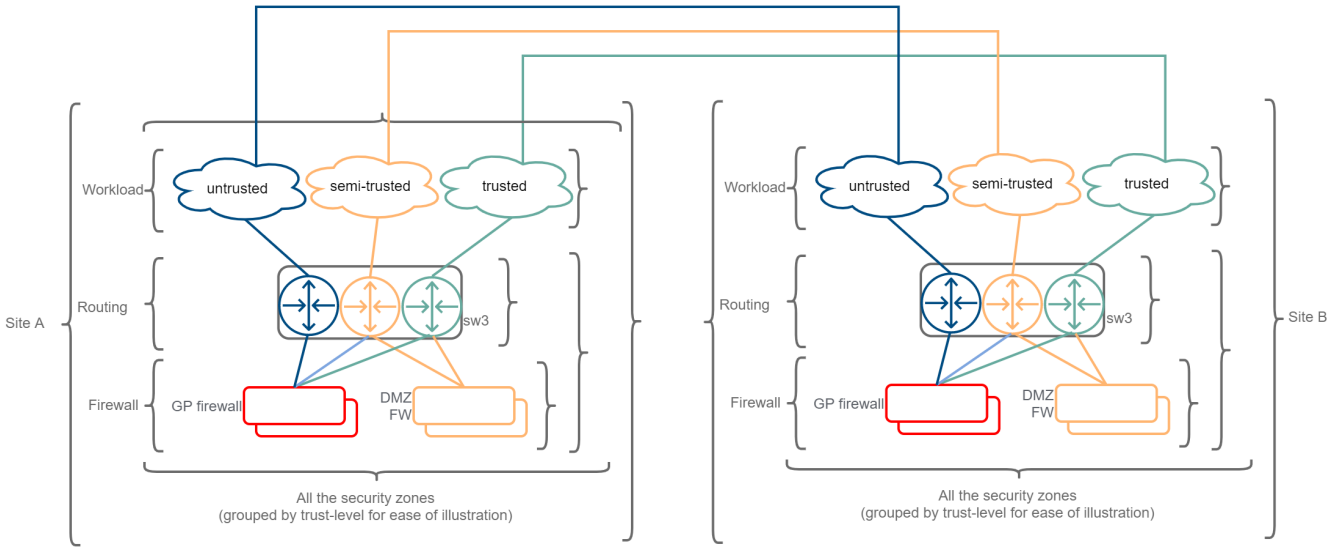
No flows should be directly permitted from external zones to internal zones.

The model above was originally extended to include a second site as depicted below, using the "Internal zone" as a transit path for **all** traffic between sites:



In the above model, traffic from "semi-trusted in site-A to "semi-trusted" in site-B traverses both the site-A DMZ FW and the site-B DMZ firewall. Traffic from "untrusted" at Site-A to "untrusted" at Site-B would have to traverse: Site-A GP firewall, and Site-B GP firewall. This requires that all traffic between sites, but in the same security zone be inspected by two separate firewall instances, where we would much prefer that it pass through ***no*** firewall instances. Additionally, this arrangement "clutters" the "trusted" zone with traffic from the untrusted and semi-trusted zones.

To achieve that desired state (no FW inspection for inter-site/intra-zone traffic), the semi-trusted and untrusted security-zones were stretched between sites A and B as depicted below:



This achieved the design objective of **not** subjecting any inter-site/intra-zone traffic to FW inspection. Unfortunately, it also raised the issue of how inter-site ***inter-zone*** could be routed consistently and efficiently. Take, for instance, a connection from site-A/semi-trusted to site-B/trusted.

Should this flow take the "semi-trusted" path to site-A, and traverse the DMZ FW at site-B en-route to the trusted zone? Or, should it take the DMZ-FW path at site-A, and take the trusted path to Site-B en-route SiteB/trusted zone?

That question boils down to: "For inter-site/inter-zone routing, should traffic traverse security-zones at the source-site, or the destination site?"

Unfortunately, a universally consistent answer for that question would result in AB traffic traversing one site's FW, and BA traffic traversing the other site's firewall (breaking inspection through the firewalls.)

Ones response to these limitations would be to limit permitted flows to restrict inter-site flows to only allow them if they are within the same zone. This, of course, represents an undesirable constraint on our ability to deploy workload across multiple sites, essentially requiring that each site have a "full stack" available.

In the context of our existing architecture (as described above), the question here is:

> *"How do we construct our network such that...":*
>
> *With:*
>
> - *Multiple "sites"*
> - *Potentially one or more security zones at each site*
> - *Multiple "security zones"*
> - *With security zones potentially extending across multiple sites*
>
> *We area able to ensure that:*
>
> - *Traffic that remains within a single security zone does not traverse any stateful perimeter inspection services*
> - *All traffic passing between security zones passes through the stateful inspection services applied at each zone's perimeter*
> - *But... traverses the same* **instance** *of the stateful inpsection services in each direction*
> - *That is, make sure traffic from A to B goes through the same firewalls as traffic from B to A*

This problem ***is*** a real one. With multiple colocation data-centers, an IaaS CSP, and more than a half-dozen security zones, we ***need*** to be able to have traffic traverse security zones (in an IaaS CSP, for instance) without getting hairpinned off a separate colo-DC just to be subjected to inspection services. Likewise, we need the flexibility to allow security zones to traverse diverse physical locations.

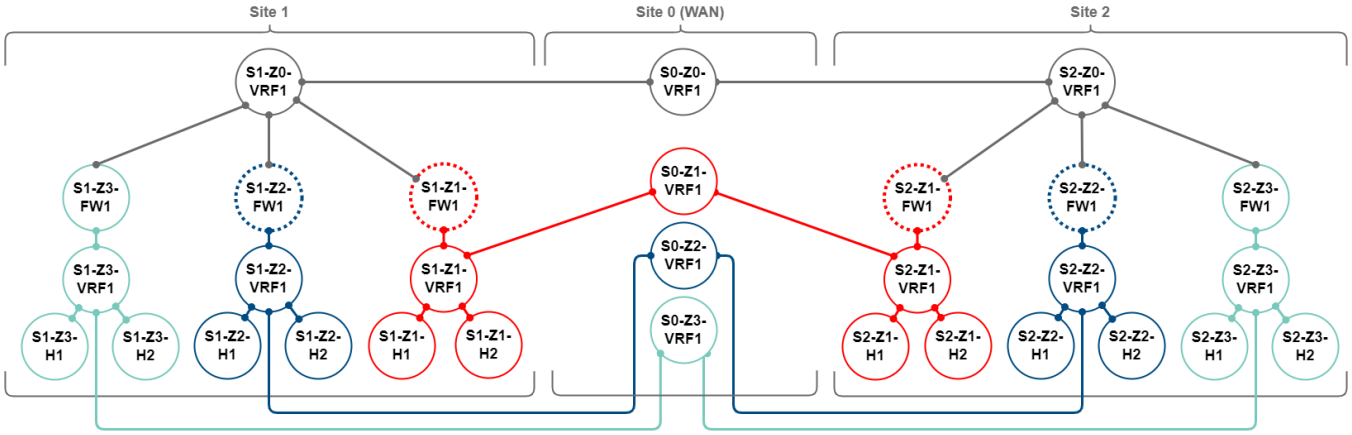
Adding a transit-zone

Another shortcoming of the existing topology is that for traffic to flow between endpoints in two different security-zones at one of those sites, at least one of those zones must be present at both sites. For instance, if If we have zones A, B, and C, with zones A&B present at site 1, and zone C present only at site-2, there is no way for a host in Site2-ZoneC to reach hosts in Site1-ZoneA or Site1-ZoneB.

With a dedicated "transit-only" (no workload hosting) zone added, the resulting topology can be represented in the following acyclic graph. This is an intentionally simplified topology, intended to depict the minimal degree of complexity at which the stated problem exists. We will demonstrate at a later point that the underlying patterns are extensible to multiple additional sites and zones, as well as hierarchical sub-zones.

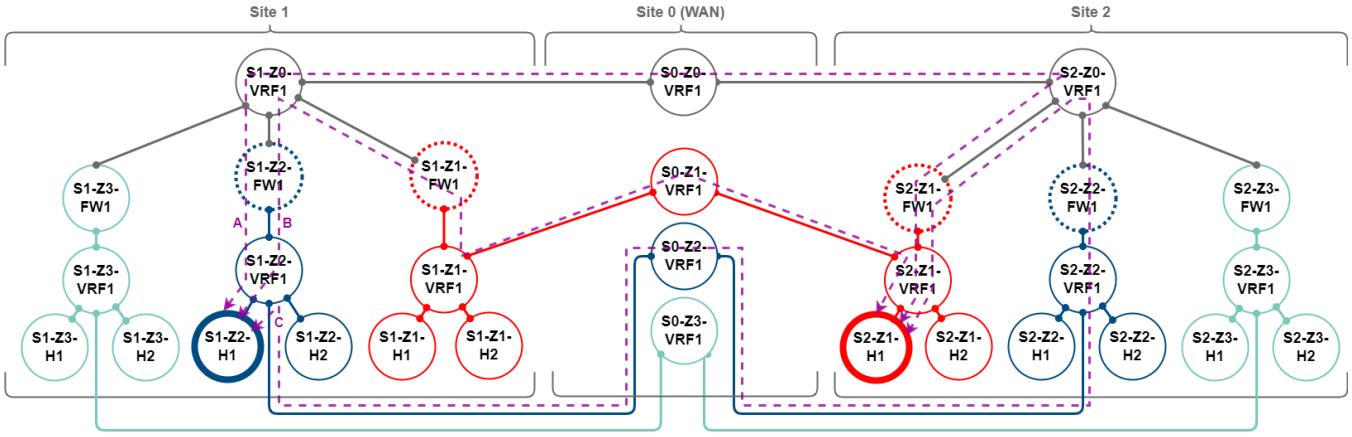
In this figure,

- We see two distinct "sites" (1 and 2)
- We see four distinct "security zones"
  - Zone 0 is a "transit zone" (no hosting permitted, only transit between other zones.)
  - Zones 1/2/3 are hosting zones for BCNC IT workload
- Elements are labeled using the following convention
  - "S*n*" = Site number *n*
  - "Z*m*" = Security zone number *m*
  - "VRF*p*" = Routed-hop (non-stateful) number *p* in the given site/zone intersection
  - "FW*q*" = Stateful routed-hop number *q* in the given site/zone intersection
  - "H*r*" = Hosting network number *r* in the given site/zone intersection



The routing protocol used in BCBS NC's data-centers and WAN is BGP, and we (for the most part) use external BGP, with most routers using a unique private ASN. As a result, AS-path length normally decides path selection within the network.

Looking at ***this*** topology, we can see that there are ***three*** separate equal-cost paths between (for instance) **S1Z2**H1 and **S2Z1**H1. Those paths (A, B, and C) are depicted in the following figure.



Each of the three best paths illustrated above has a total of seven transit hops, and each traverses a Z2 firewall instance and a Z1 firewall instance. However, each path traverses a different combination of firewall ***instances***:

- Path "A" traverses FWs: S1Z2FW1 and S2Z1FW1
- Path "B" traverses: FWs S1Z2FW1 and S1Z1FW1
- Path "C" traverses FWs: S2Z2FW1 and S2Z2FW1

In this topology, in order for a packet to start in one security zone and end in another (excluding the transit-zone), it must traverse a firewall on the perimeter of both the source and destination zones. Because there are three equal-cost paths for each inter-site/inter-zone combination, the network relies on ECMP for path selection. For any two endpoints ("A" and "B", for example), that are in different zones and different sites, this means that roughly 1/3rd of the flows will take each of the three paths from A to B. Going from B to A, the mix is the same. The result is that approximately 66% of the flows between A and B will end up being routed asymmetrically. That asymmetrical pathing will result in those flows (66% of them) passing through separate firewall instances on the AB and BA paths, causing the firewalls to drop packets for those flows due tto what they perceive as broken TCP state.

The generalized pattern we can infer here is that:

> *For every packet traversing both a security zone border and a site border, there will be ECMP routes present for the destination at both **the originating-VR of the packet** (paths B and C) and at the **zone-0 VR (paths A and B) of the site at which the packet originated***

Virtual Lab Topology (baseline/problem-statement)

A GNS3 virtual-lab implementing the previously discussed topology is accessible at: http://vlgns3d001:3080/static/web-ui/server/1/project/4cc6b26d-13d9-47f8-8eab-e2005f3a1072

The Problem, restated

Without imposing some sort of routing policy on the topology discussed here, flows that traverse both site and security-zone boundaries are likely to fail, and to fail relatively unpredictably (as a function of choices made by ECMP hashing algorithms.)

A Proposed Solution

Routing policy is the tool with which we can resolve the asymmetric pathing of inter-site/inter-zone flow, enforcing symmetric pathing through consistent sets of stateful hops for specific traffic flows. But, before we talk about policy implementation specifics, we must identify the underlying logic of our proposed routing policy.

In the problem we've set for ourselves, the key is that traffic in both directions for a given flow must traverse the same *stateful* hops. We should begin then, by considering the various equal-cost paths in terms of those stateful hops. And, because our ultimate objective is to enable symmetric paths, we should consider symmetry or reflexivity. That is, we should consider ways of describing the stateful hops in these paths that yield "mirror" results when evaluated from opposites ends of the path.

The Policy Logic Framework

We can describe our potential (AB and BA) paths with the following characteristics:

- Destination-site properties
  - The security-zone-ID in which the packet's destination is located
  - The site-ID in which the packet's destination is located
  - The number of stateful-hops in the packet's destination site
- Source-site properties
  - The security-zone-ID in which the packet originated
  - The site-ID in which the packet originated
  - The number of stateful-hops in the packet's originating site
- Whether the path traverses the transit-zone in site-0 (the WAN.)

That group of primitives gives us sufficient descriptive power to define a policy that achieves our objectives regarding path symmetry. The key to that policy is that we must be able to make comparisons between zone-ID values, classifying any zone-ID as "higher", "lower", or "equal" relative to any other zone-ID.

A Proposed Policy

A (distributed) policy that we can use, expressed with the characteristics above is:

> - *Always prefer the shortest path (as measured by AS path count)*
> - **If** *equal cost paths are present, then*
>   - *For routers with non-zero site-IDs and zone-IDs of zero, always prefer next-hops with a non-zero Zone-ID to next-hops with a zone-ID of zero.*
>   - *This prevents the zone-zero routers from each site from considering path-A in our examples above, and only considers paths B/C*
>   - *We never want to traverse path, A. This is the "one-firewall at each site" path, and the zone-0/transit-zone VRFs at each end don't have a consistent reflexive basis of comparison for of their own zone-ID's to that of the destination route.*
>   - *Always prefer the path with more stateful hops in the site of the higher security-zone ("higher" of the packet's source and destination sites)*

- *Equivalently: Always prefer the path with less stateful hops in the site of the lower security-zone ("lower" of the packet's source and destination sites)*

The "key" to that policy is that the last two rules above are essentially mirror images of each other, and they resolve to the same path from each "side." Attempting to apply that logic to the previous test-case (S1Z2H1S2Z1H1 traffic), we see that it gives us the result we want. (The hops in **bold** are the ones at which equal-cost paths are present.

- S1Z2H1 S2Z1H1 (higher-zone lower-zone)
  - S1Z2Ha is directly connected to S1Z2VR1
    - **S1Z2VR1 prefers S1Z2FW1 (because it's on path-B, which has 2 stateful hops in site-1; the site with the higher security zone)**
      - S1Z2FW1 prefers S1Z0VR1 as the next-hop (the shortest path)
        - **S1ZO-VR1 prefers S1Z1FW1 as the next hop (the path with more FW hops in site-1)**
          - S1Z1FW prefers S1Z1VR1 as the next hop (the shortest path)
            - S1Z1VR1 prefers S0Z1VR1 as the next hop (the shortest path)
              - S0Z1VR1 prefers S2Z1VR1 one as the next hop (the shortest path)
                - S2Z1VR1 is directly connected to S2Z1H1
- S2Z1H1 S1Z2H1 (lower-zone higher-zone)
  - S2Z1VR1 is directly connected to S2Z1H1
    - **S2Z1VR1 prefers S0Z1VR1 (because it's on path-B, which has 2 stateful hops in site-1; the site with the higher security zone)**
      - S0Z1VR1 prefers S1Z1VR1 as the next-hop (the shortest path)
        - **S1Z1-VR1 prefers S1Z1FW1 as the next hop (the path with more FW hops in site-1; the site with the higher security zone)**
          - S1Z1FW1 prefers S1Z0VR1 as the next hop (the shortest path)
            - S1Z0VR1 prefers S1Z2FW1 as the next hop (the shortest path)
              - S1Z2FW1 prefers S1Z2VR1 as the next hop (the shortest path)
                - S1Z2H1 is directly connected to S1Z2VR1

The section above describes an effective policy for resolving the question of what logic we'd want to apply to our routing, but it begs the question of what *mechanisms* we could use to implement it.

Enabling the Policy in the Control Plane

The policy described in the previous section results in the forwarding/pathing behavior we want, but it was described in terms of data-plane forwarding, and we don't want to implement policy directly in the data-plane if we can avoid it. (Not *everything* is an SDN.) However, we can achieve the desired routing policy outcomes in the control plane, using standard BGP features and a bit of applied logic.

BGP large-communities are a handy mechanism for adding an extra signaling layer to BGP route updates. We can define arbitrary key/value pairs to signal multiple properties of interest. Those key value pairs are encoded as multiple values within the BGP "large-community" attribute on a BGP route advertisement. We have to encode them as purely numerical values but an offline code-book can imbue them with whatever "meaning" we want. A BGP large-community value is canonically written as three unsigned integers, each separated by a colon.

The first integer is the "global administration" field (typically the registered ASN of the organization that's defining the way in which the large-community attributes will be used. The second integer is defined as "local data 1", and the 3rd integer is defined as "local data 2." Using that system we can implement an arbitrary number of type/value "tags" to assign to a route. For example:

Large Community Value **(*global-admin*:*local-data-1*:*local-data-2)***	**Global Administrator Field Definition**	**Local Data 1 Definition**	**Local Data 2 Definition**	**Logic to apply on learning/originating prefixes**
22593:**1**:***1-100***	"**22593**" = Blue Cross NC's registered ASN	Type: security-zone of route-origination ("ORSECZID") ("**1**")	Value: Security-zones ("**1-100**")	Only set this when originating a route. Set it to the zone-ID of the originating VRF.
22593:**2**:***1-100***	"**22593**" = Blue Cross NC's registered ASN	Type: site-id of route-origination ("ORSITEID") ("**2**")	Value: Site-IDs ("**1-100**")	Only set this when originating a route. Set it to the site-ID of the originating VRF.
22593:**3**:***1-100***	"**22593**" = Blue Cross NC's registered ASN	Type: # of stateful hops in the site-of-origination ("OSSHC") ("**3**")	Value: hop-count ("**1-100**")	Set to 0 when originating a route. Increment this when learning a route ***if*** the learning-router is a firewall and the learning-router's site-ID is the same as the originating site-ID.
22593:**4**:***1-100***	"**22593**" = Blue Cross NC's registered ASN	Type: Site-ID of router propagating the route ("PRSITEID") ("**4**")	Value: Site-IDs ("**1-100**")	Set to router's local site-ID when originating or propagating a route.

22593:5:*1-100*	"22593" = Blue Cross NC's registered ASN	Type: Security-zone-ID of router propagating the route ("PRSECZID") ("5")	Value: Site-IDs ("1-100")	Set to router's local security-zone-ID when originating or propagating a route.
22593:6:*0-1t*	"22593" = Blue Cross NC's registered ASN	Type: Propagating router's statefulness ("PRISSTATEFUL") ("1")	Value: Boolean (0 /1)	Indicates whether the propagating router is running stateful services or not (such as a firewall.) (**In case** we can't implement complex route-map policy on the firewalls BGP processes, hopefully they can at least additively insert an arbitrary large-community tag to indicate that they are stateful.)

> *"1-100" is shorthand for 100 different actual full-length large-community values in which the last octet is a value between 1 and 100 (inclusive). For example:*
>
> *225931:1:1, 225931:1:2, 225931:1:3, etc...*

Using the BGP "large-community" attribute to signal routing-policy data is particularly attractive because it can accommodate roughly 4-billion different data-types, each allowing up to 4-billion values. This leaves more than ample room to use the same mechanism for implementing additional routing policy at a later date. In contrast, attempting to encode these value in (for instance) the structure of a set of private ASNs, is far more limited.

The data about each path that we need to encode, expressed in terms of BGP primitives, is conceptually split into 2 distinct parts:

Route Origination

When a router originates a route, it must tag the route (using BGP large-community attribute) with :

- ORSECZID (with a value defined statically in the device configuration)
- ORSITEID (with a value defined statically in the device configuration)
- OSSHC (with a value of "0")
- PRSECZID (with a value defined statically in the device configuration)
- PRSITEID (with a value defined statically in the device configuration)
- PRISSTATEFUL (with a value defined statically in the device configuration)

Route Learning

- If the learning router is stateful, it must:
  - If the learning router is in the same site as which the route was *originated*, increment the OSSHC tag by 1
- The learning router must then set a BGP MED/metric value
  - If the learned route's ORSECZID is lower than the learning router's zone-ID (High-zone to Low-zone flow), then
    - Set the MED value using the OSSHC tag's value-field
    - This is a double reversal strategy, in that:
      - We want to prefer routes with a greater # of stateful hops at the current router's site (which corresponds to a lower # in the originating site.)
        - So using the USSHC tag makes the routes from sites with MORE firewalls LESS preferable
          - Which is equivalent to making routes from sites with LESS firewalls MORE preferable
  - If the learned route's ORSECZID is higher than the learning router's zone-ID (Low-zone to High-zone flow), then
    - Set the MED value using an inverse mapping of the the OSSHC tag's value-field
      - I've used 0-9 as an arbitrary range, to allow us to expand to hierarchical firewall topologies
      - This reverses the preference values, compared to a , which is what we want.

Route Propagation

- The learning router must update the PRSECZID, PRSITEID, and PRISSTATEFUL tags with it's local-configuration data when receiving a route from another router in the policy domain.
- If the router is stateful (as with a firewall) and if its zone-ID is the same as the ORSECZID tag, then
  - Increment the ORSSHC tag

The forwarding behavior that we need to produce is summarized below

Our policy objective was previously defined as:

- Always prefer the shortest path (as measured by AS path count)
  - This is the default operation for eBGP. Nothing special required on our part.
- *If* equal cost paths are present, then
  - Never prefer a path that is learned from a router in the WAN "site" (0) in the transit zone (0)
  - ~~Always prefer routes with a path through hosting security-zones' (1,2,3) WAN VRFs over routes with a path through the transit zone's (0) WAN VRF~~
    - ~~We can implement this requirement by checking the AS-path of learned routes for the ASN of the Z0Z0VR (or VRs) and setting a "least-preferred" metric~~
  - Always prefer the path with more stateful hops in the site of the higher security-zone ("higher" of the packet's source and destination sites), and with *less* stateful hops in the site of the lower security-zone ("lower" of the packet's source and destination sites)
    - For "low-to-high" flows, this translates to "prefer the path with the most stateful hops in the destination site"
      - The "originating-site stateful hop-count" ("OSSHC") value for each path has been encoded in a BGP large-community string included in each learned route
    - For "high-to-low" flows, this translates to "prefer the path with the least stateful hops in the destination site"

- The "originating-site stateful hop-count" ("OSSHC") value for each path has been encoded in a BGP large-community string included in each learned route

The desired forwarding behavior described above requires each router participating in this policy scheme to:

- Rely on AS-path length for forwarding decisions in cases of non-equal-cost routes
- Rely on a not-yet-identified metric for forwarding decisions in the case on equal-cost routes

BGP uses several "attributes" when determining which routes are loaded into the routing-table:

Priority	Attribute	Notes
2	Local Preference	Numeric value (higher = better.) Shared within an ASN (to iBGP peers); not propagated outside of the ASN (to eBGP peers.)
3	Originate	Router will prefer routes that **it** injected into BGP over routes injected by any other router
4	AS path length	Prefer path with shortest AS-path length
5	Origin code	
6	MED	Prefer path with lowest MED. Propagated to eBGP peers only.
7	eBGP path over iBGP path	
8	Shortest IGP path to BGP next hop	
9	Oldest path	
10	Router ID	
11	Neighbor IP address	

As illustrated in the preceding table, the "local preference" attribute and MED (multi-exit discriminator) can both be used to influence path selection. Critically though, "local preference" evaluations will supersede AS-path length evaluations, whereas MED evaluations are superseded by AS-path length evaluations. Since our objective is to ***only*** affect forwarding decisions when AS-path length is equal, MED is preferable to local-preference for use in implementing our policy objectives.

After receiving routes and (if appropriate) modifying their respective large-community attribute tags, each router implementing this policy, must accept the learned routes, and apply a specific MED value to each (based on decoding the ORSECZID, ORSITEID, OSSHC, PRSHC, PRSID tags of the learned routes and evaluating them against the statically configured "my-site-ID", my-zone-ID", and "IAMFW" variables.)

The specific MED values to be used should guarantee that:

- If the route came from a higher security zone than that of the current router, the MEDs of learned routes should ensure that
    - OSSCH of "0" is least preferred among all equal-cost paths
    - OSSCH of "1" is more preferred than OSSCH of "0", among all equal-cost paths
    - OSSCH of "2" is more preferred than OSSCH of "1", among all equal-cost paths
    - etc...
  - If the route came from a lower security zone than that of the current router, the MEDs of learned routes should ensure that
    - OSSCH of "2" is most preferred among all equal-cost paths
    - OSSCH of "1" is more preferred than OSSCH of "0", among all equal-cost paths
    - OSSCH of "1" is more preferred than OSSCH of "1", among all equal-cost paths
    - etc...

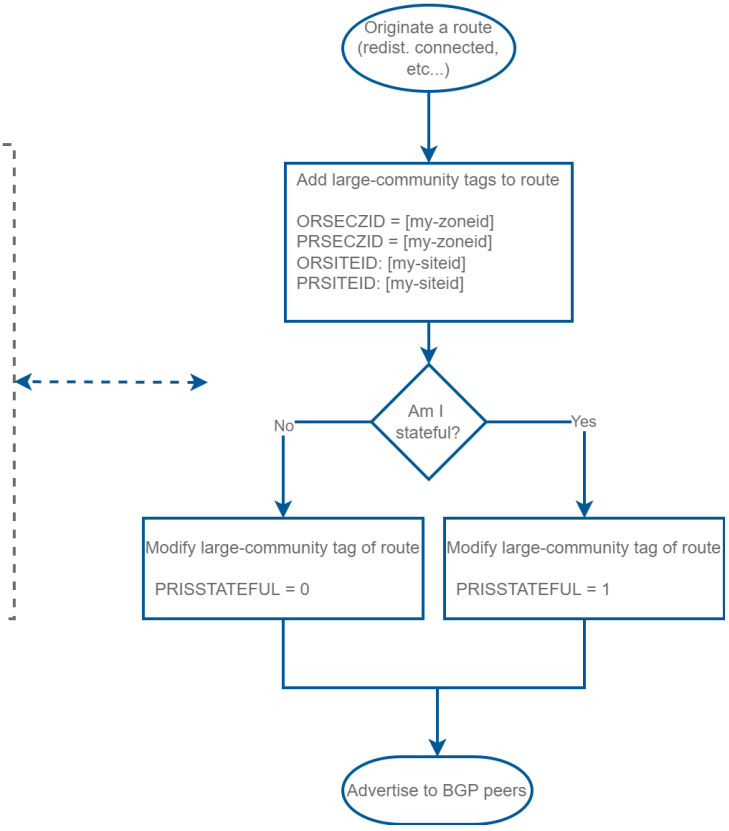
For example, S1Z2VR1 has 3 routes to S2Z1H1. (Paths A, B, and C from an earlier diagram.)

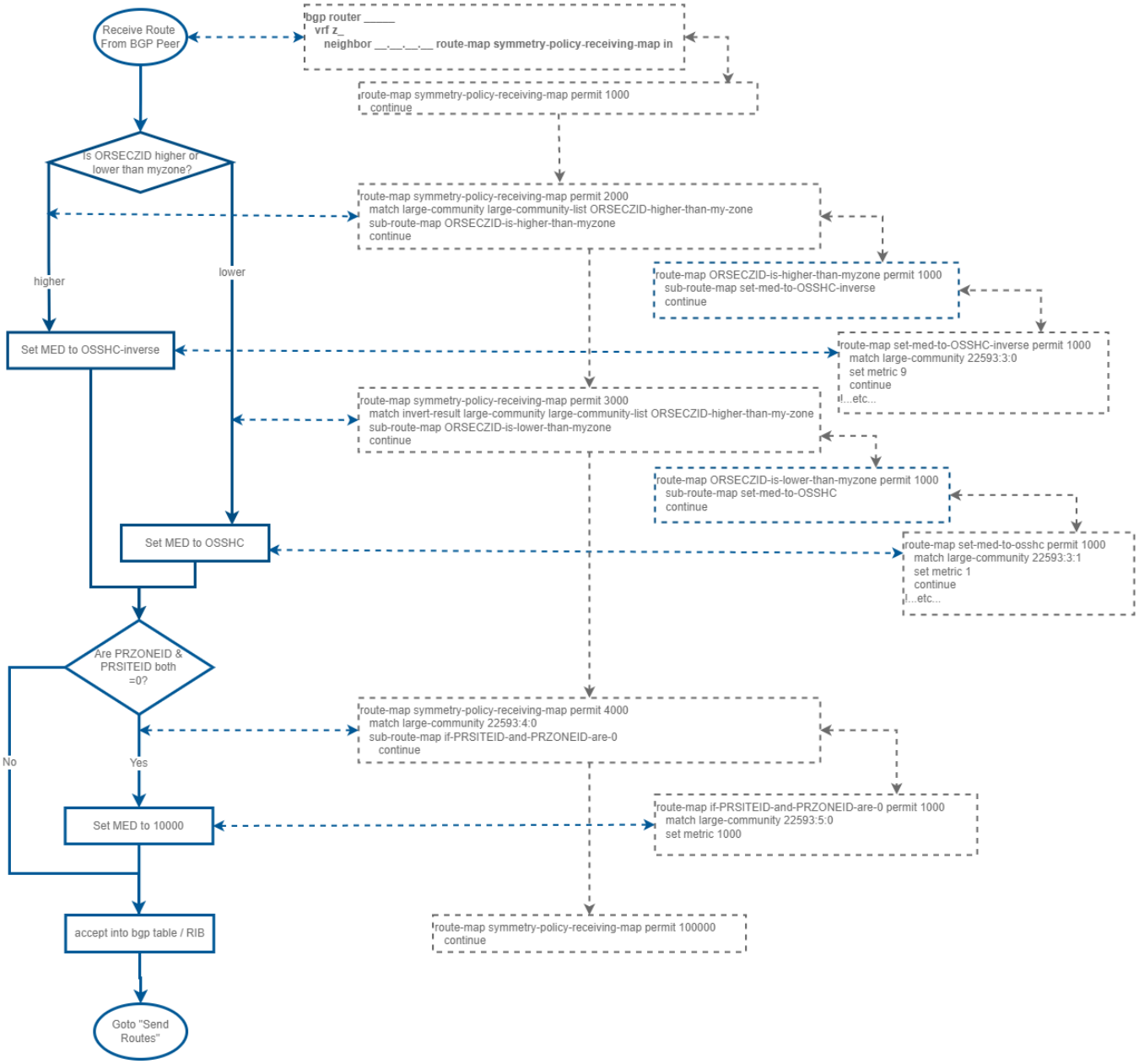
- S1Z2VR1 sees path "A" with an OSSCH of "1"
- S1Z2VR1 sees path "B" with an OSSCH of "0"
- S1Z2VR1 sees path "C" with an OSSCH of "2"

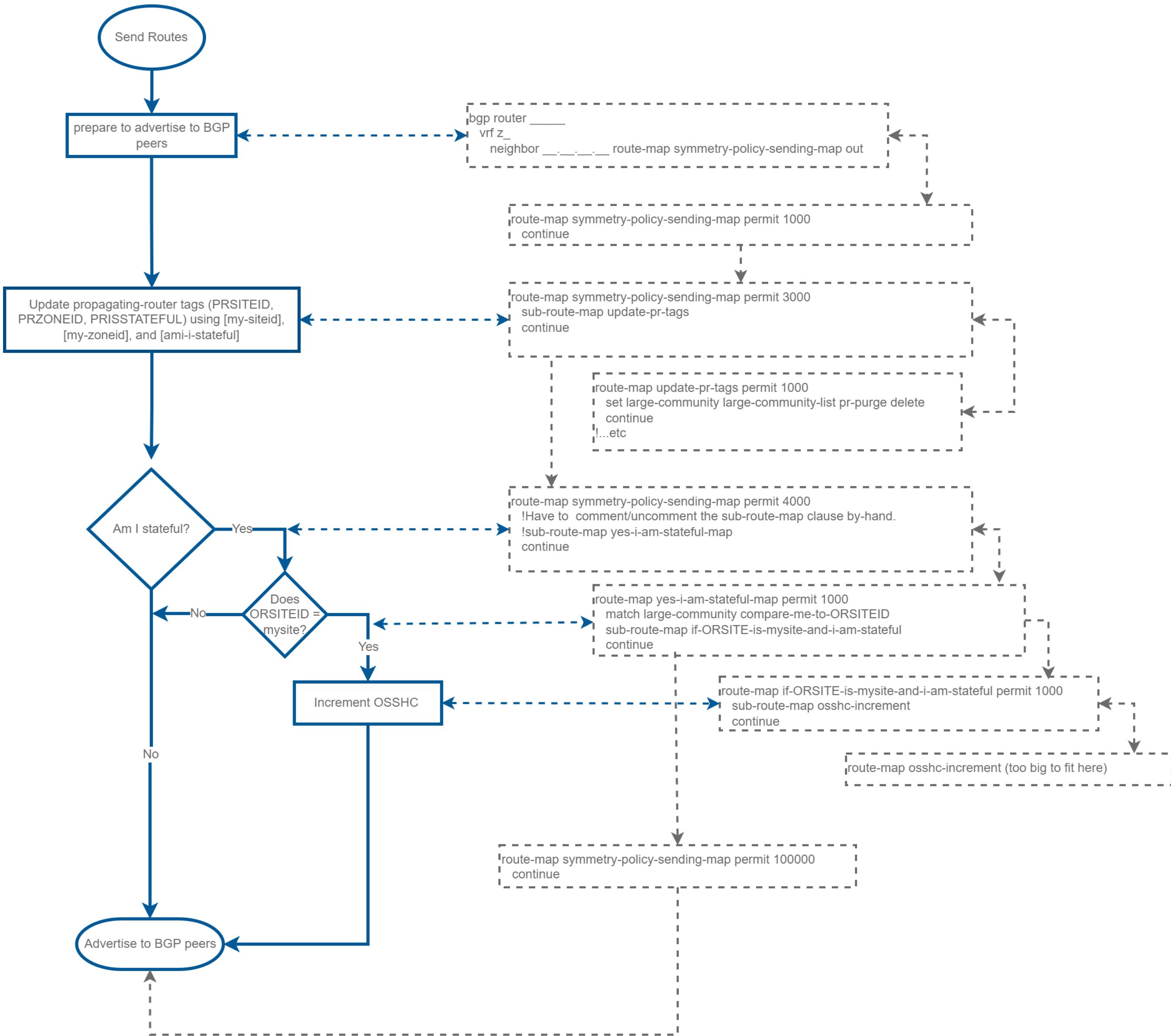
When we look at the inverse-path (S2Z1VR1 to S1Z2H1), we see:

- S1Z2VR1 sees path "A" with an OSSCH of "1"
- S1Z2VR1 sees path "B" with an OSSCH of "2""
- S1Z2VR1 sees path "C" with an OSSCH of "0""

```
route-map tag-originated-routes-for-symmetric-policy permit 1000
  set large-community large-community-list compare-me-to-ORSECZID
  continue
route-map tag-originated-routes-for-symmetric-policy permit 1100
  set large-community large-community-list compare-me-to-ORSITEID additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 1200
  set large-community large-community-list originatingstatefulhop additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 1300
  set large-community large-community-list compare-me-to-PRSITEID additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 1400
  set large-community large-community-list propagatingstatefulhop additive
  continue
!
!network commands to originate routes with the tags applied
router bgp ____
 vrf ___
  network _._._._/_ route-map tag-originated-routes-for-symmetric-policy
  network _._._._/_ route-map tag-originated-routes-for-symmetric-policy
```




Receive Route From BGP Peer
bgp router _____
vrf z_
neighbor __._.__.__ route-map symmetry-policy-receiving-map in
route-map symmetry-policy-receiving-map permit 1000
continue
Is ORSECZID higher or lower than myzone?
route-map symmetry-policy-receiving-map permit 2000
match large-community large-community-list ORSECZID-higher-than-my-zone
sub-route-map ORSECZID-is-higher-than-myzone
continue
higher
lower
route-map ORSECZID-is-higher-than-myzone permit 1000
sub-route-map set-med-to-OSSHC-inverse
continue
Set MED to OSSHC-inverse
route-map set-med-to-OSSHC-inverse permit 1000
match large-community 22593:3:0
set metric 9
continue
!...etc...
route-map symmetry-policy-receiving-map permit 3000
match invert-result large-community large-community-list ORSECZID-higher-than-my-zone
sub-route-map ORSECZID-is-lower-than-myzone
continue
route-map ORSECZID-is-lower-than-myzone permit 1000
sub-route-map set-med-to-OSSHC
continue
Set MED to OSSHC
route-map set-med-to-osshc permit 1000
match large-community 22593:3:1
set metric 1
continue
!...etc...
Are PRZONEID & PRSITEID both =0?
route-map symmetry-policy-receiving-map permit 4000
match large-community 22593:4:0
sub-route-map if-PRSITEID-and-PRZONEID-are-0
continue
No
Yes
Set MED to 10000
route-map if-PRSITEID-and-PRZONEID-are-0 permit 1000
match large-community 22593:5:0
set metric 1000
accept into bgp table / RIB
route-map symmetry-policy-receiving-map permit 100000
continue
Goto "Send Routes"




```
!New-routing-policy; v3.
!Find/replace the values below.  (Make a working-copy of this file first.)
!That will handle all of the simple string-substition customization
!    Find             Replace-with
!    [my-zoneid]        [my-zoneid]        !! Security-zone of THIS router
!    [my-siteid]        [my-siteid]        !! Site-ID of THIS router
!    [am-i-stateful]    [am-i-stateful]    !! Is this router running stateful services?  (0/1 = no/yes)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
enable
config t

!neighbor msmz-policy peer group
!neighbor msmz-policy route-map symmetry-policy-receiving-map in
!neighbor msmz-policy route-map symmetry-policy-sending-map out
!neighbor msmz-policy send-community extended large
!neighbor msmz-policy additional-paths receive
!neighbor msmz-policy additional-paths send any
```

```
!==================================READ THIS SECTION CAREFULLY - MANUAL MODIFICATION
REQUIRED==================================!
! Comment/un-comment lines in this list, based on THIS routers' security-zone ID
!(If router's zone-ID is "3", lines for ...:4-:6 should be included in the "...higher-than-my-zone" list
!ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:1
!ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:2
!ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:3
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:4
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:5
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:6
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:7
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:8
ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:9
!ip large-community-list ORSECZID-higher-than-my-zone permit 22593:1:0

route-map symmetry-policy-sending-map permit 4000
!   Manually un-comment or comment the following line based on whether THIS router is running stateful services or not.
!   sub-route-map yes-i-am-stateful-map
  continue
!=========================================END OF MANUAL MODIFICATION
SECTION=========================================!

!==================================READ THIS SECTION CAREFULLY - FIND/REPLACE MODIFICATION
REQUIRED==================================!
ip large-community-list am-i-stateful permit 22593:6:[am-i-stateful]
ip large-community-list compare-me-to-ORSECZID permit 22593:1:[my-zoneid]
ip large-community-list compare-me-to-ORSITEID permit 22593:2:[my-siteid]
ip large-community-list compare-me-to-PRSITEID permit 22593:4:[my-siteid]
ip large-community-list compare-me-to-PRSECZID permit 22593:5:[my-zoneid]
ip large-community-list originating-stateful-hop permit 22593:3:[am-i-stateful]
ip large-community-list originating-stateful-hop permit 22593:6:[am-i-stateful]
ip large-community-list originating-stateful-hop permit 22593:7:[am-i-stateful]
!=======================================END OF FIND/REPLACE MODIFICATION
SECTION=======================================!

!==================================UNIFORM/CONSISTENT CONFIGURATON FOR ALL DEVICES IN ROUTING POLICY
DOMAIN==================================!
service routing protocols model multi-agent
service routing configuration route-map set-operations sequential
ip large-community-list 22593:1:0 permit 22593:1:0
ip large-community-list 22593:1:1 permit 22593:1:1
ip large-community-list 22593:1:2 permit 22593:1:2
ip large-community-list 22593:1:3 permit 22593:1:3
ip large-community-list 22593:1:4 permit 22593:1:4
ip large-community-list 22593:1:5 permit 22593:1:5
ip large-community-list 22593:1:6 permit 22593:1:6
ip large-community-list 22593:1:7 permit 22593:1:7
ip large-community-list 22593:1:8 permit 22593:1:8
ip large-community-list 22593:1:9 permit 22593:1:9
ip large-community-list 22593:2:0 permit 22593:2:0
ip large-community-list 22593:2:1 permit 22593:2:1
ip large-community-list 22593:2:2 permit 22593:2:2
ip large-community-list 22593:2:3 permit 22593:2:3
ip large-community-list 22593:2:4 permit 22593:2:4
ip large-community-list 22593:2:5 permit 22593:2:5
ip large-community-list 22593:2:6 permit 22593:2:6
ip large-community-list 22593:2:7 permit 22593:2:7
ip large-community-list 22593:2:8 permit 22593:2:8
ip large-community-list 22593:2:9 permit 22593:2:9
ip large-community-list 22593:3:0 permit 22593:3:0
ip large-community-list 22593:3:1 permit 22593:3:1
ip large-community-list 22593:3:2 permit 22593:3:2
ip large-community-list 22593:3:3 permit 22593:3:3
ip large-community-list 22593:3:4 permit 22593:3:4
ip large-community-list 22593:3:5 permit 22593:3:5
ip large-community-list 22593:3:6 permit 22593:3:6
ip large-community-list 22593:3:7 permit 22593:3:7
ip large-community-list 22593:3:8 permit 22593:3:8
ip large-community-list 22593:3:9 permit 22593:3:9
ip large-community-list 22593:4:0 permit 22593:4:0
ip large-community-list 22593:4:1 permit 22593:4:1
ip large-community-list 22593:4:2 permit 22593:4:2
ip large-community-list 22593:4:3 permit 22593:4:3
ip large-community-list 22593:4:4 permit 22593:4:4
```

```
ip large-community-list 22593:4:5 permit 22593:4:5
ip large-community-list 22593:4:6 permit 22593:4:6
ip large-community-list 22593:4:7 permit 22593:4:7
ip large-community-list 22593:4:8 permit 22593:4:8
ip large-community-list 22593:4:9 permit 22593:4:9
ip large-community-list 22593:5:0 permit 22593:5:0
ip large-community-list 22593:5:1 permit 22593:5:1
ip large-community-list 22593:5:2 permit 22593:5:2
ip large-community-list 22593:5:3 permit 22593:5:3
ip large-community-list 22593:5:4 permit 22593:5:4
ip large-community-list 22593:5:5 permit 22593:5:5
ip large-community-list 22593:5:6 permit 22593:5:6
ip large-community-list 22593:5:7 permit 22593:5:7
ip large-community-list 22593:5:8 permit 22593:5:8
ip large-community-list 22593:5:9 permit 22593:5:9
ip large-community-list 22593:6:0 permit 22593:6:0
ip large-community-list 22593:6:1 permit 22593:6:1
ip large-community-list 22593:6:2 permit 22593:6:2
ip large-community-list 22593:6:3 permit 22593:6:3
ip large-community-list 22593:6:4 permit 22593:6:4
ip large-community-list 22593:6:5 permit 22593:6:5
ip large-community-list 22593:6:6 permit 22593:6:6
ip large-community-list 22593:6:7 permit 22593:6:7
ip large-community-list 22593:6:8 permit 22593:6:8
ip large-community-list 22593:6:9 permit 22593:6:9
ip large-community-list 22593:7:0 permit 22593:7:0
ip large-community-list 22593:7:1 permit 22593:7:1
ip large-community-list 22593:7:2 permit 22593:7:2
ip large-community-list 22593:7:3 permit 22593:7:3
ip large-community-list 22593:7:4 permit 22593:7:4
ip large-community-list 22593:7:5 permit 22593:7:5
ip large-community-list 22593:7:6 permit 22593:7:6
ip large-community-list 22593:7:7 permit 22593:7:7
ip large-community-list 22593:7:8 permit 22593:7:8
ip large-community-list 22593:7:9 permit 22593:7:9
ip large-community-list regexp pr-purge permit 22593:4:.+
ip large-community-list regexp pr-purge permit 22593:5:.+
ip large-community-list regexp pr-purge permit 22593:6:.+
route-map tag-originated-routes-for-symmetric-policy permit 1000
  set large-community large-community-list compare-me-to-ORSECZID
  continue
route-map tag-originated-routes-for-symmetric-policy permit 2000
  set large-community large-community-list compare-me-to-ORSITEID additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 3000
  set large-community large-community-list originating-stateful-hop additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 4000
  set large-community large-community-list compare-me-to-PRSITEID additive
  continue
route-map tag-originated-routes-for-symmetric-policy permit 100000

route-map ORSECZID-is-higher-than-myzone permit 1000
  sub-route-map set-med-to-OSSHC-inverse
  continue
route-map ORSECZID-is-higher-than-myzone permit 100000
route-map ORSECZID-is-lower-than-myzone permit 1000
  sub-route-map set-med-to-OSSHC
  continue
route-map ORSECZID-is-lower-than-myzone permit 100000
route-map am-i-stateful-map permit 100000
route-map if-ORSITE-is-mysite-and-i-am-stateful permit 1000
  sub-route-map osshc-increment
  continue
route-map if-ORSITE-is-mysite-and-i-am-stateful permit 100000
route-map if-PRSITEID-and-PRSECZID-are-0 permit 1000
  match large-community 22593:5:0
  set metric 1000
route-map osshc-increment permit 1000
  match large-community 22593:3:8
  continue
  set large-community 22593:3:9 additive
route-map osshc-increment permit 2000
  match large-community 22593:3:8
  continue
```

```
   set large-community 22593:3:8 delete
route-map osshc-increment permit 3000
   match large-community 22593:3:7
   continue
   set large-community 22593:3:8 additive
route-map osshc-increment permit 4000
   match large-community 22593:3:7
   continue
   set large-community 22593:3:7 delete
route-map osshc-increment permit 5000
   match large-community 22593:3:6
   continue
   set large-community 22593:3:7 additive
route-map osshc-increment permit 6000
   match large-community 22593:3:6
   continue
   set large-community 22593:3:6 delete
route-map osshc-increment permit 7000
   match large-community 22593:3:5
   continue
   set large-community 22593:3:6 additive
route-map osshc-increment permit 8000
   match large-community 22593:3:5
   continue
   set large-community 22593:3:5 delete
route-map osshc-increment permit 9000
   match large-community 22593:3:4
   continue
   set large-community 22593:3:5 additive
route-map osshc-increment permit 10000
   match large-community 22593:3:4
   continue
   set large-community 22593:3:4 delete
route-map osshc-increment permit 11000
   match large-community 22593:3:3
   continue
   set large-community 22593:3:4 additive
route-map osshc-increment permit 11800
   match large-community 22593:3:0
   continue
   set large-community 22593:3:0 delete
route-map osshc-increment permit 12000
   match large-community 22593:3:3
   continue
   set large-community 22593:3:3 delete
route-map osshc-increment permit 13000
   match large-community 22593:3:2
   continue
   set large-community 22593:3:3 additive
route-map osshc-increment permit 14000
   match large-community 22593:3:1
   continue
   set large-community 22593:3:2 additive
route-map osshc-increment permit 15000
   match large-community 22593:3:1
   continue
   set large-community 22593:3:2 additive
route-map osshc-increment permit 16000
   match large-community 22593:3:1
   continue
   set large-community 22593:3:1 delete
route-map osshc-increment permit 17000
   match large-community 22593:3:0
   continue
   set large-community 22593:3:1 additive
route-map osshc-increment permit 100000
route-map set-med-to-OSSHC permit 1000
   match large-community 22593:3:0
   continue
   set metric 0
route-map set-med-to-OSSHC permit 2000
   match large-community 22593:3:1
   continue
   set metric 1
route-map set-med-to-OSSHC permit 3000
```

```
  match large-community 22593:3:2
  continue
  set metric 2
route-map set-med-to-OSSHC permit 4000
  match large-community 22593:3:3
  continue
  set metric 3
route-map set-med-to-OSSHC permit 5000
  match large-community 22593:3:4
  continue
  set metric 4
route-map set-med-to-OSSHC permit 6000
  match large-community 22593:3:5
  continue
  set metric 5
route-map set-med-to-OSSHC permit 7000
  match large-community 22593:3:6
  continue
  set metric 6
route-map set-med-to-OSSHC permit 8000
  match large-community 22593:3:7
  continue
  set metric 7
route-map set-med-to-OSSHC permit 9000
  match large-community 22593:3:8
  continue
  set metric 8
route-map set-med-to-OSSHC permit 10000
  match large-community 22593:3:9
  continue
  set metric 9
route-map set-med-to-OSSHC permit 100000
route-map symmetry-policy-receiving-map permit 1000
  continue
route-map symmetry-policy-receiving-map permit 2000
  match large-community ORSECZID-higher-than-my-zone
  sub-route-map ORSECZID-is-higher-than-myzone
  continue
route-map symmetry-policy-receiving-map permit 3000
  match invert-result large-community ORSECZID-higher-than-my-zone
  sub-route-map ORSECZID-is-lower-than-myzone
  continue
route-map symmetry-policy-receiving-map permit 4000
  match large-community 22593:4:0
  sub-route-map if-PRSITEID-and-PRSECZID-are-0
  continue
route-map symmetry-policy-receiving-map permit 100000
route-map symmetry-policy-sending-map permit 1000
  continue
route-map symmetry-policy-sending-map permit 3000
  sub-route-map update-pr-tags
  continue
route-map symmetry-policy-sending-map permit 100000
route-map update-pr-tags permit 1000
  continue
  set large-community large-community-list pr-purge delete
route-map update-pr-tags permit 2000
  continue
  set large-community large-community-list compare-me-to-PRSITEID additive
route-map update-pr-tags permit 3000
  continue
  set large-community large-community-list compare-me-to-PRSECZID additive
route-map update-pr-tags permit 4000
  continue
  set large-community large-community-list am-i-stateful additive
route-map update-pr-tags permit 100000
route-map yes-i-am-stateful-map permit 1000
  match large-community compare-me-to-ORSITEID
  sub-route-map if-ORSITE-is-mysite-and-i-am-stateful
  continue
route-map yes-i-am-stateful-map permit 100000
route-map set-med-to-OSSHC-inverse permit 1000
  match large-community 22593:3:0
  continue
  set metric 9
```

```
route-map set-med-to-OSSHC-inverse permit 2000
  match large-community 22593:3:1
  continue
  set metric 8
route-map set-med-to-OSSHC-inverse permit 3000
  match large-community 22593:3:2
  continue
  set metric 7
route-map set-med-to-OSSHC-inverse permit 4000
  match large-community 22593:3:3
  continue
  set metric 6
route-map set-med-to-OSSHC-inverse permit 5000
  match large-community 22593:3:4
  continue
  set metric 5
route-map set-med-to-OSSHC-inverse permit 6000
  match large-community 22593:3:5
  continue
  set metric 4
route-map set-med-to-OSSHC-inverse permit 7000
  match large-community 22593:3:6
  continue
  set metric 3
route-map set-med-to-OSSHC-inverse permit 8000
  match large-community 22593:3:7
  continue
  set metric 2
route-map set-med-to-OSSHC-inverse permit 9000
  match large-community 22593:3:8
  continue
  set metric 1
route-map set-med-to-OSSHC-inverse permit 10000
  match large-community 22593:3:9
  continue
  set metric 0
route-map set-med-to-OSSHC-inverse permit 100000
```

Virtual Lab Topology (BGP policy proof-of-concept)

A GNS3 virtual-lab implementing the policy configuration discussed above (built on the previously linked "baseline" lab) is available athttp://vlgns3d001:3080/static/web-ui/server/1/project/6813805e-90e7-4243-b52f-2c97c8ab9591

Remaining Work

- We should establish enterprise-wide standards/practices/documentation on the use of BGP large-community attribute in routing policy.
- Update framework to account for non-symmetric inter-site transit paths on different zones.
    - Rely on BGP local-preference to weight same-zone paths over other paths?
- Reduce scope of footprint so that policy only needs to be implemented on the firewalls and the routers immediately north/south of them (ideally, not even on the firewalls.)
- The POC virtual-lab and configurations should be extended to include additional scenarios
    - No discrete "transit zone" / no-man's land
        - Flattened firewalls (multi-zone firewalls, not separate firewall per zone)
    - Hierarchical security zones
    - Look into implementing policy via Arista "Routing Control Functions" rather than route-maps
        - Richer functionality; probably more manageable configurations