



Cisco Learning Network CCIE SP series

IOS XR RPL – Route Policy Language

Łukasz Bromirski

lukasz.bromirski@cisco.com / [@LukaszBromirski](https://twitter.com/LukaszBromirski)

Cisco Learning Network CCIE SP Series

Agenda

- Are [route-maps|prefix-lists|access-lists|filter-list|distribute-list|offset-lists|...] not enough?
- RPL introduction
- RPL basic usage & constructs
- RPL in specific use cases and troubleshooting RPL
- Q&A

Route Policy Language

Introduction

RPL brings clarity to Route-Maps

```
router bgp 100
  bgp log-neighbor-changes
  neighbor 100.64.1.1 remote-as 101
  neighbor 100.64.1.1 password CLN-WEBINARS-RULEZ
  neighbor 100.64.1.1 remove-private-as
  neighbor 100.64.1.1 soft-reconfiguration inbound
  neighbor 100.64.1.1 prefix-list INTERNET-IN in
  neighbor 100.64.1.1 prefix-list INTERNET-OUT out
  neighbor 100.64.1.1 route-map INTERNET-IN in
  neighbor 100.64.1.1 route-map INTERNET-OUT out
  neighbor 100.64.1.1 filter-list 3 in
  neighbor 100.64.1.1 filter-list 7 out
```

What is the order of processing?

Filtering order of operations – IOS/IOS-XE

- For inbound updates the order of preference is:
 - route-map
 - filter-list
 - prefix-list, distribute-list
- For outbound updates the order of preference is:
 - filter-list
 - route-map | unsuppress-map
 - advertise-map (conditional-advertisement)
 - prefix-list|distribute-list
 - ORF prefix-list (a prefix-list the neighbor sends us)
- **Note: The attributes prefix-list and distribute-list are mutually exclusive, and only one command (neighbor prefix-list or neighbor distribute-list) can be applied to each inbound or outbound direction for a particular neighbor.**

So.... new routing policy tool is needed!

- RPL developed along the IOS XR (1997-)
- Main building principles:
 - exploit modularity (think SPs, think IXPs, scale, SCALE!)
 - parametrization (SCALE again!)
 - clarity (one default, no hidden steps, explicit logic)
- Incremental changes in new releases

Let's compare live policy – SRD in action

IOS/IOS XE/NX OS

```
!  
route-map BGP-BH-IPv4 deny 10  
  match ip address prefix-list GOLDEN-NETS  
!  
route-map BGP-BH-IPv4 permit 100  
  match community bgpbh-bogons  
  set local-preference 6666  
  set weight 6666  
  set origin igp  
  set community no-advertise additive  
  set ip next-hop 192.0.2.1  
!  
route-map BGP-BH-IPv4 permit 200  
  match community bgpbh-bogons-self  
  set local-preference 6666  
  set weight 6666  
  set origin igp  
  set community no-advertise additive  
  set ip next-hop 192.0.2.1  
!
```

IOS XR

```
!  
route-policy BGP-BH-IPv4  
  if destination in GOLDEN-NETS then  
    drop  
  endif  
  if community matches-within \  
    (bgpbh-bogons, bgpbh-bogons-self) then  
    set local-preference 6666  
    set weight 6666  
    set origin igp  
    set community (no-advertise) additive  
    set next-hop discard  
  endif  
!
```

1. Do not program into FIB anything pointing to „Golden prefixes” (root DNS/NTP/local)
2. Install in FIB any routes matching communities bgpbh-bogons & bgpbh-bogons-self and set proper attributes to drop/discard them

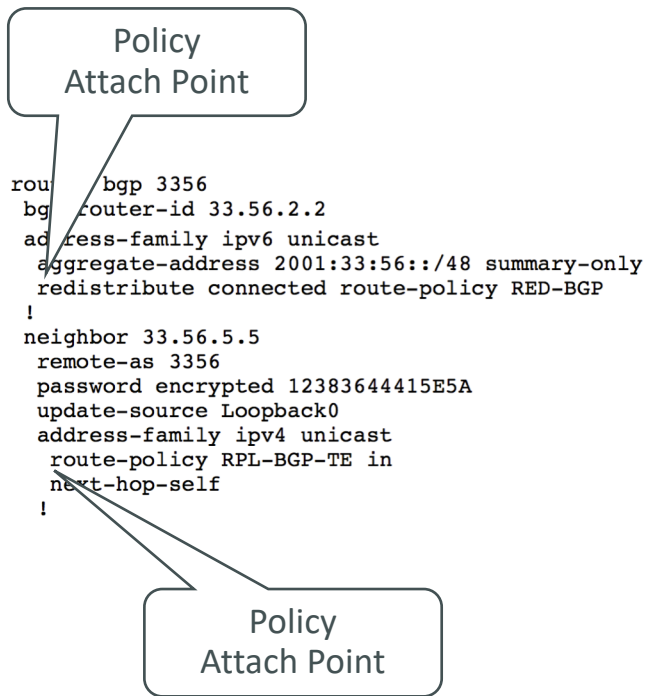
My BGP edge policies simplified!

IOS XR

```
!  
route-policy BGP-EDGE-ORANGE  
  apply BGP-F-BOGONS  
  apply BGP-BP-COMMON  
  apply BGP-BP-ORANGE-PREF  
end-policy  
!  
route-policy BGP-EDGE-TMOBILE  
  apply BGP-F-BOGONS  
  apply BGP-BP-COMMON  
  apply BGP-BP-TMOBILE-PREF  
!  
router bgp  
  neighbor x.x.x.x  
    address-family ipv4  
      route-policy BGP-EDGE-ORANGE in  
  neighbor x.x.y.y  
    address-family ipv4  
      route-policy BGP-EDGE-TMOBILE in
```

```
prefix-set PFX-BOGONS  
  0.0.0.0/8 le 32,  
  10.0.0.0/8 le 32,  
  [...]  
end-set  
!  
route-policy BGP-F-BOGONS  
  if destination in ( PFX-BOGONS ) then  
    drop  
  endif  
end-policy  
  
route-policy BGP-BP-COMMON  
  set origin igp  
  set local-preference 500  
  set med 100  
  delete community all  
end-policy  
  
route-policy BGP-BP-ORANGE-PREF  
  if destination in ( PFX-ORANGE ) then  
    set local-preference 200  
  else  
    set local-preference 90  
  endif  
end-policy
```

RPL keywords



Protocol Attribute(s)	RPL Attribute(s)	RPL Operation(s)
next-hop	source	pass / drop
weight	destination	suppress-route
local-preference	route-type	unsuppress-route
med	rib-has-route	length, unique-length
origin	traffic-index	set
as-path	dampening	apply
community	label	If, then
ext community	tag	else, elseif
rd		and, or, not
		eq, neq, le, gt
		in, is
		ios-regex

Actions in a RPL

Define action (default is drop) and may affect control flow

There is an implicit drop at the end of RPL processing.

A route must be given a **'ticket'** to ensure that it has been inspected by the RPL

Pass – prefix allowed if not later dropped

pass grants a ticket to defeat default drop

Execution **continues** after pass

Set – value changed, prefix allowed if not later dropped

Any **set** at any level grants a ticket

Execution **continues** after **set**

Values can be set more than once

Drop – prefix is discarded

Explicit drop **stops** policy execution

Implicit drop (if policy runs to end without getting a ticket)

Done – accepts prefix and **stops** processing

Things to remember when working with RPL:

Default eBGP policy (a.k.a. RFC 8212)

- eBGP sessions by default won't exchange any prefixes unless policy is configured
- There's a knob:
`bgp unsafe-ebgp-policy`
- <https://tools.ietf.org/html/rfc8212>

Things to remember when working with RPL:

Original value is stored until end of policy

- A conditional match does not occur on intermediary values during the route policy processing.

Conditional Matches on Original Value

```
route-policy ORIGINAL-VALUES
  if med eq 100 then
    set med 200
  endif
  if med eq 200 then
    drop
  endif
end-policy
```

- In the example, only the original routes with a MED of 200 are dropped and the routes with values set to 200 are not dropped.

Basic RPL Examples

■ Basic Pass Policy

Example Configuration

```
route-policy PASS-ALL  
  pass  
end-policy
```

■ Basic Drop Policy

Example Configuration

```
route-policy DROP-ALL  
  drop  
end-policy
```

Somewhat redundant due to
implicit drop

RPL Examples

- Basic conditional statement

Logic	Example Configuration
if Match-Condition-One then Action-One end-if	if med eq 150 then pass endif

Conditional Match

Action

- Branching options

Logic	Example Configuration
if Match-Condition-One then Action-One else Action-Two end-if	if destination in (10.0.0.0/8 ge 8) then pass else drop endif

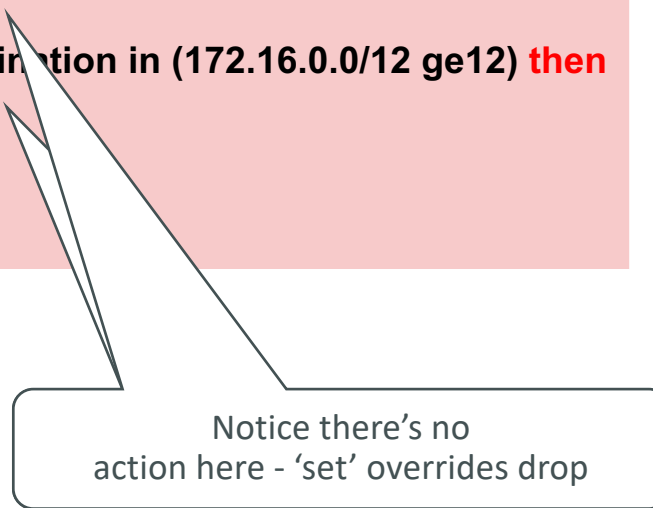
Comparison operator

Notice we are matching networks directly in the RPL.
Supports Prefix Matching or Wildcard

RPL Examples (continued)

- Multiple branching options

Logic	Example Configuration
if Match-Condition-One then Action-One elseif Match-Condition-Two then Action-Two else Action-Three end-if	if destination in (10.0.0.0/8 ge 8) then set tag 1 elseif destination in (172.16.0.0/12 ge12) then set tag 2 else drop endif



Notice there's no action here - 'set' overrides drop

Nested Conditions

- If statements within other if/elseif/else statements
Method or placing multiple conditions
- Nesting can be any depth

Logic	Example Configuration
if MATCHING-CONDITION-ONE then if MATCHING-CONDITION-TWO then ACTION-ONE end-if end-if	if as-path passes-through '100' then if destination in PREFIX-SET-RFC1918 then pass endif endif

Simplifying BGP AS-Path Conditions

AS Path Selection Criteria	Route-Map AS-Path ACL Logic (<i>ip as-path access-list 1</i>)	RPL Logic
Local Routes	<code>permit ^\$</code>	<code>if as-path is-local</code>
Only Routes From Neighbor AS 200	<code>permit ^200_</code>	<code>if as-path neighbor-is '200'</code>
Only Routes Originating From AS 200	<code>permit _200\$</code>	<code>if as-path originates-from '200'</code>
Passes Through AS200	<code>permit _200_</code>	<code>if as-path passes-through '200'</code>
Routes From 3 ASes or less away	<code>permit ^[0-9]+ [0-9]+ [0-9]+?</code>	<code>if as-path length le 3</code>

RPL Examples

Bad RPL Logic

```
route-policy METRIC-MODIFICATION
  if destination in (192.168.0.0/16 ge 16) then
    set med 100
  endif
  set med 200
end-policy
```

Overwrites setting

Good RPL Logic Option #1

```
route-policy METRIC-MODIFICATION
  if destination in (10.0.0.0/8 ge 8) then
    set med 100
    pass
  else
    set med 200
    pass
  endif
end-policy
```

Option #2

```
route-policy METRIC-MODIFICATION
  if destination in (10.0.0.0/8 ge 8) then
    set med 100
    done
  endif
  set med 200
end-policy
```

Stops all processing on
matched prefixes

Route Policy Language

Sets, nesting policies and parametrization

RPL Policy Sets

- Prefix-lists, ACLs, AS_PATH ACLs can be confusing because of permit/deny actions
- IOS XR uses policy sets to store the same information:
Prefix set, Community set, Extended Community set, AS_PATH set, RD set
- There is no 'deny' in a Policy set
- Processing occurs until the first match is made

Named and Inline Set (same behavior)

Inline Example Configuration

```
if destination in (10.0.0.0/8 ge 8, 172.16.0.0/12 ge 12, 192.168.0.0/16 ge 16) then
    pass
else
    drop
endif
```

Set Example Configuration

```
route-policy RFC1918-PREFIX-SET
    if destination in PREFIX-SET-RFC1918 then
        pass
    endif
end-policy
!
prefix-set PREFIX-SET-RFC1918
    10.0.0.0/8 ge 8,
    172.16.0.0/12 ge 12,
    192.168.0.0/16 ge 16
end-set
```

Viewing Set Based RPLS

Keyword required to see sets in the RPL

Inline Example Configuration

```
RP/0/0/CPU0:XR1#show rpl route-policy RFC1918-PREFIX-SET inline

route-policy RFC1918-PREFIX-SET
  if destination in (10.0.0.0/8 ge 8, 172.16.0.0/12 ge 12, 192.168.0.0/16 ge 16) then
    pass
  endif
end-policy
```

Parameter Passing

Single Parameter

```
route-policy PARAM ($MED)
  set med $MED
end-policy

router bgp 300
[...]
neighbor 33.56.5.1
  remote-as 49.12
  address-family ipv4 unicast
    route-policy PARAM (50) in
    route-policy PASS-ALL out
```

List of policy parameters

Accessing the passed
parameter

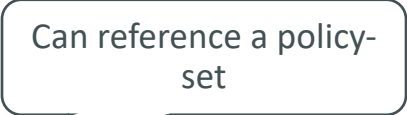
Calling policy and passing
parameter

Using Multiple Parameters

Multiple Parameters

```
route-policy SP-PEER ($AS, $PREFIX)
  if destination in $PREFIX and as-path originates-from '$AS' then
    pass
  endif
end-policy

router bgp 300
[...]
```



```
  neighbor 33.56.5.1
  remote-as 49.12
  address-family ipv4 unicast
    route-policy SP-PEER (50, CUST1-PREFIX-SET) in
  route-policy PASS-ALL out
```

Can reference a policy-set

Nesting in RPL

- By nesting policies we can scale RPL out

Example Configuration

```
route-policy PARENT
  apply CHILD-ONE
  apply CHILD-TWO
  pass
end-policy

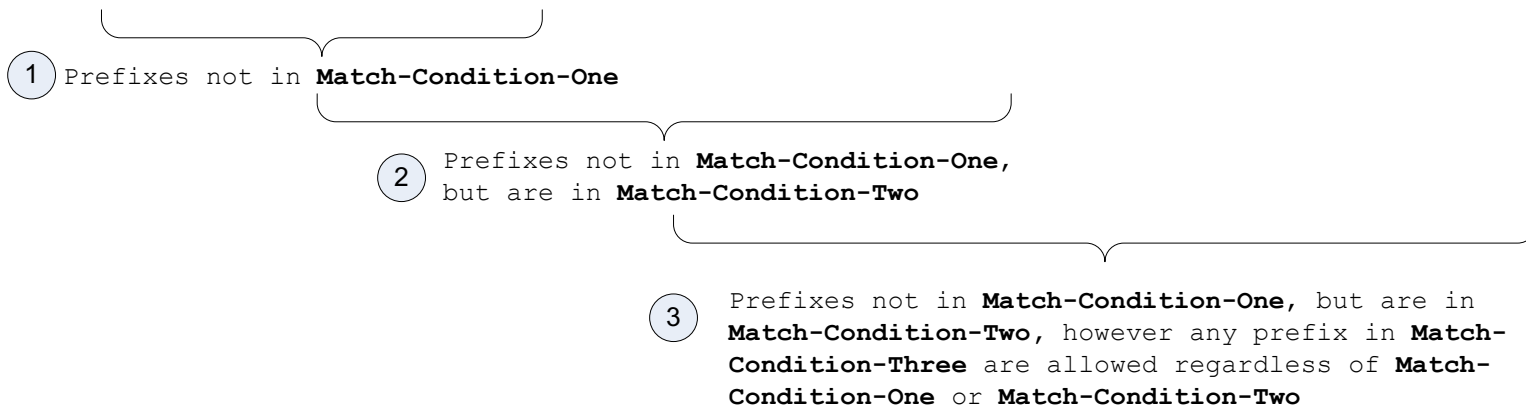
route-policy CHILD-ONE
  set weight 100
end-policy

route-policy CHILD-TWO
  set community (2:1234) additive
end-policy
```

Boolean (Logical) Operations

- Comparison operators are context sensitive
 - Semantic check not done until RPL policy use is committed
- Supported Operators - Not, And, & Or (in order of precedence)

if Not **Match-Condition-One** and **Match-Condition-Two** or **Match-Condition-Three**



Boolean (Logical) Operations

- Conditional match that requires a route to not pass through AS 100 or AS 200, and must be within the 192.168.0.0/16 network range

```
if (( (Not Match-Condition-One) and Match-Condition-Two) or Match-Condition-Three)
```

Use of parentheses

```
if not (as-path passes-through '100' or as-path passes-through '200') and destination in  
(192.168.0.0/16 ge 16)
```

Boolean Operators

Negation

```
if not destination in PREFIX-SET-RFC1918 then
    pass
endif
```

Conjunction

```
if destination in PREFIX-SET-RFC1918 and as-path passes-through '100' then
    pass
endif
```

Disjunction

```
if destination in PREFIX-SET-RFC1918 or as-path passes-through '100' then
    pass
endif
```

Route Policy Language

Corner cases, comments and troubleshooting

Community matching and manipulation

Matching specific community(-ies)

```
!  
route-policy AS100-TE  
  if community matches-any (100:748) then  
    delete community in (internet)  
  endif  
  pass  
end-policy  
!
```

Deleting / modifying communities

```
!  
route-policy AS100-TE  
  if [...]  
    delete community in (100:*)  
  endif  
  pass  
end-policy  
!
```

Nested policies in ,if' statement

- What is the behavior of policies if nested at ,if' statement?

```
!  
route-policy CHECK-MULTIPLE-OPTIONS  
  if apply CHECK-FIRST and apply CHECK-SECOND then  
    set community 3356:666  
  else  
    drop  
  endif  
  pass  
end-policy  
!
```

Remarks in policies (that survive reboot)

(and upgrade... and in order defined originally ;))

```
!  
route-policy CLN71  
  # this is specific policy for  
  # testing the ordered remarks  
  # for CLN session slides  
  #  
  # welcome  
  set med 6  
end-policy  
!
```

Profiling of RPL runs – use with care!

- “Everybody has a testing environment. Some people are lucky enough enough to have a totally separate environment to run production in.”



DevOptimist.
@stahnma

```
!  
RP/0/0/CPU0:SP1R2#debug pcl profile detail  
!  
(wait)  
!  
RP/0/0/CPU0:SP1R2#sh pcl protocol bgp speaker-0 neighbor-in-dflt default-IPv4-40.67.1.1 policy profile  
Policy profiling data  
Policy : AS100-TE  
Pass : 4  
Drop : 0  
# of executions : 4  
Total execution time : 0msec
```

Node Id	Num visited	Exec time	Policy engine operation
PXL_0_1			community delete-in <end-policy/>

Route Policy Language

Migration hints

Migrating route-maps to RPL

- Do a simple syntax translation
- Nest conditionals to reduce repetitions & comparisons
- Use inline sets to remove small indirect set references
- Parameterize to reuse common structures

Step 1: Direct syntax translation

- Each route-map becomes a route-policy
- Each clause in a route-map becomes a clause in an if-then-else sequence.
- For each clause:

Map each 'match' to the corresponding conditional.

Map each 'set' to the corresponding 'action'.

```
route-map PROCESS_INBOUND deny 5
  match as-path 150
!
route-map PROCESS_INBOUND permit 10
  match as-path 10
  match community 1
  set local-preference 70
  set community 100:500 100:505 100:999 additive
!
route-map PROCESS_INBOUND permit 20
  match as-path 10
  match community 2
  set local-preference 80
  set community 100:500 100:505 100:999 additive
!
route-map PROCESS_INBOUND permit 30
  set local-preference 90
  set community 100:500 100:505 100:999 additive
!
```

```
route-policy PROCESS_INBOUND
  if (as-path in aspath_150) then
    drop
  elseif ((community matches-any comm_1) and (as-path in aspath_10)) then
    set local-preference 70
    set community (100:500, 100:505, 100:999) additive
  elseif ((community matches-any comm_2) and (as-path in aspath_10)) then
    set local-preference 80
    set community (100:500, 100:505, 100:999) additive
  else
    set local-preference 90
    set community (100:500, 100:505, 100:999) additive
  endif
end-policy
```

Step 2: Nest Conditionals

- Collect similar conditions into nested 'if' statements.

```
route-policy PROCESS_INBOUND
  if (as-path in as_path_150) then
    drop
  elseif (as-path in as_path_10) then
    if (community matches-any comm_1) then
      set local-preference 70
      set community (100:500, 100:505, 100:999) additive
    elseif (community matches-any comm_2) then
      set local-preference 80
      set community (100:500, 100:505, 100:999) additive
    endif
  else
    set local-preference 90
    set community (100:500, 100:505, 100:999) additive
  endif
end-policy
```

Step 3: Use inline sets (when it makes sense)

- Small sets (AS-Path set, Community set, etc.) can be replaced with inline sets.

```
route-policy PROCESS_INBOUND
  if (as-path in '_701_', '_3561_') then
    drop
  elseif (as-path in '^21409_') then
    if (community matches-any '5511:70') then
      set local-preference 70
      set community (100:500, 100:505, 100:999) additive
    elseif (community matches-any '5511:80') then
      set local-preference 80
      set community (100:500, 100:505, 100:999) additive
    endif
  else
    set local-preference 90
    set community (100:500, 100:505, 100:999) additive
  endif
end-policy
```

Step 4: Parameterize

- Similar actions can be grouped into a common policy with parameters.

```
route-policy set_attributes ($pref)
  set local-preference $pref
  set community (100:500:, 100:505, 100:999) additive
end-policy
!
route-policy PROCESS_INBOUND
  if (as-path in '_701_', '_3561_') then
    drop
  elseif (as-path in '^21409_') then
    if (community matches-any '5511:70') then
      apply set_attributes (70)
    elseif (community matches-any '5511:80') then
      apply set_attributes (80)
    endif
  else
    apply set_attributes (90)
  endif
end-policy
```

Policy Lists with mixed entries.

- Recall, that sets within RPL do not convey the concept of permit or deny - only membership.
- So, how does the following policy get converted ?

```
ip prefix-list martians seq 10 permit 0.0.0.0/0
ip prefix-list martians seq 20 permit 127.0.0.0/8 le 32
ip prefix-list martians seq 30 deny 10.192.0.0/10 ge 12 le 21
ip prefix-list martians seq 40 permit 10.0.0.0/8 le 32
ip prefix-list martians seq 50 permit 172.16.0.0/12 le 32
ip prefix-list martians seq 60 permit 192.168.0.0/16 le 32
ip prefix-list martians seq 70 permit 128.0.0.0/16 le 32
ip prefix-list martians seq 80 permit 192.0.0.0/24 le 32
ip prefix-list martians seq 90 permit 223.255.255.0/24 le 32
ip prefix-list martians seq 100 permit 224.0.0.0/3 le 32
ip prefix-list martians seq 110 permit 192.157.69.0/24 le 32
!
route-map CUST-FACE deny 10
  match ip address prefix-list martians
```

Policy Lists with mixed entries.

Keep all of the 'permit's ?

```
prefix-set pfx_martians
 0.0.0.0/0,
127.0.0.0/8 le 32,
10.0.0.0/8 le 32,
172.16.0.0/12 le 32,
192.168.0.0/16 le 32,
128.0.0.0/16 le 32,
192.0.0.0/24 le 32,
223.255.255.0/24 le 32,
224.0.0.0/3 le 32,
192.157.69.0/24 le 32
end-set
!
route-policy CUST_FACE
  if (destination in pfx_martians) then
    drop
  else
    pass
  endif
end-policy
!
```

Keep all of the 'deny's ?

```
prefix-set pfx_martians
 10.192.0.0/10 ge 12 le 21,
end-set
!
route-policy CUST_FACE
  if (destination in pfx_martians) then
    pass
  else
    drop
  endif
end-policy
!
```

Policy Lists with mixed entries.

The answer is: **BOTH** !

- 1) Partition the prefix-list into separate sections - each containing a string of 'permit' or 'deny' entries.
- 2) Create a prefix-set to correspond to each section.
- 3) Adjust the route-policy to process each partition in turn.

Keeping the partitions in order is important to preserve the original logic with respect to overlapping entries.

The same process can be applied to as-path-set(s) & community-set(s)

```
prefix-set pfx_martians_p1_permit
 0.0.0.0/0
 127.0.0.0/8 le 32
end-set
!
prefix-set pfx_martians_p2_deny
 10.192.0.0/10 ge 12 le 21
end-set
!
prefix-set pfx_martians_p3_permit
 10.0.0.0/8 le 32,
 172.16.0.0/12 le 32,
 192.168.0.0/16 le 32,
 128.0.0.0/16 le 32,
 191.255.0.0/16 le 32,
 192.0.0.0/24 le 32,
 223.255.255.0/24 le 32,
 224.0.0.0/3 le 32,
 192.157.69.0/24 le 32
end-set
!
route-policy CUST_FACE
 if (destination in pfx_martians_p1_permit) then
   drop
 elseif (destination in pfx_martians_p2_deny) then
   pass
 elseif (destination in pfx_martians_p3_permit) then
   drop
 endif
end-policy
```

Route Policy Language

Follow up – where to look for information

Resources

- Understanding and using IOS XR RPL:
<https://supportforums.cisco.com/t5/service-providers-documents/asr9000-xr-understanding-and-using-rpl-route-policy-language/ta-p/3117050>
- Using IOS XR RPL for BGP:
<https://learning.nil.com/assets/Tips-/Using-the-IOS-XR-Routing-Policy-Language-for-BGP.pdf>
- Great site for IOS XR geeks:
<https://xrdocs.github.io/>
- Cisco Press IOS XR fundamentals book:
<http://www.ciscopress.com/store/cisco-ios-xr-fundamentals-9781587052712>
- Cisco Press IP routing on IOS, IOS XE and IOS XR book:
<http://www.ciscopress.com/store/ip-routing-on-cisco-ios-ios-xe-and-ios-xr-an-essential-9781587144233>
- CCIE SP Study Group home page:
<https://learningnetwork.cisco.com/groups/ccie-sp-study-group>



Cisco Learning Network CCIE SP series

IOS XR RPL – Route Policy Language

Thank You!

Łukasz Bromirski

lukasz.bromirski@cisco.com / [@LukaszBromirski](https://twitter.com/LukaszBromirski)

Cisco Learning Network CCIE SP Series