Auto-export Explanation and Example

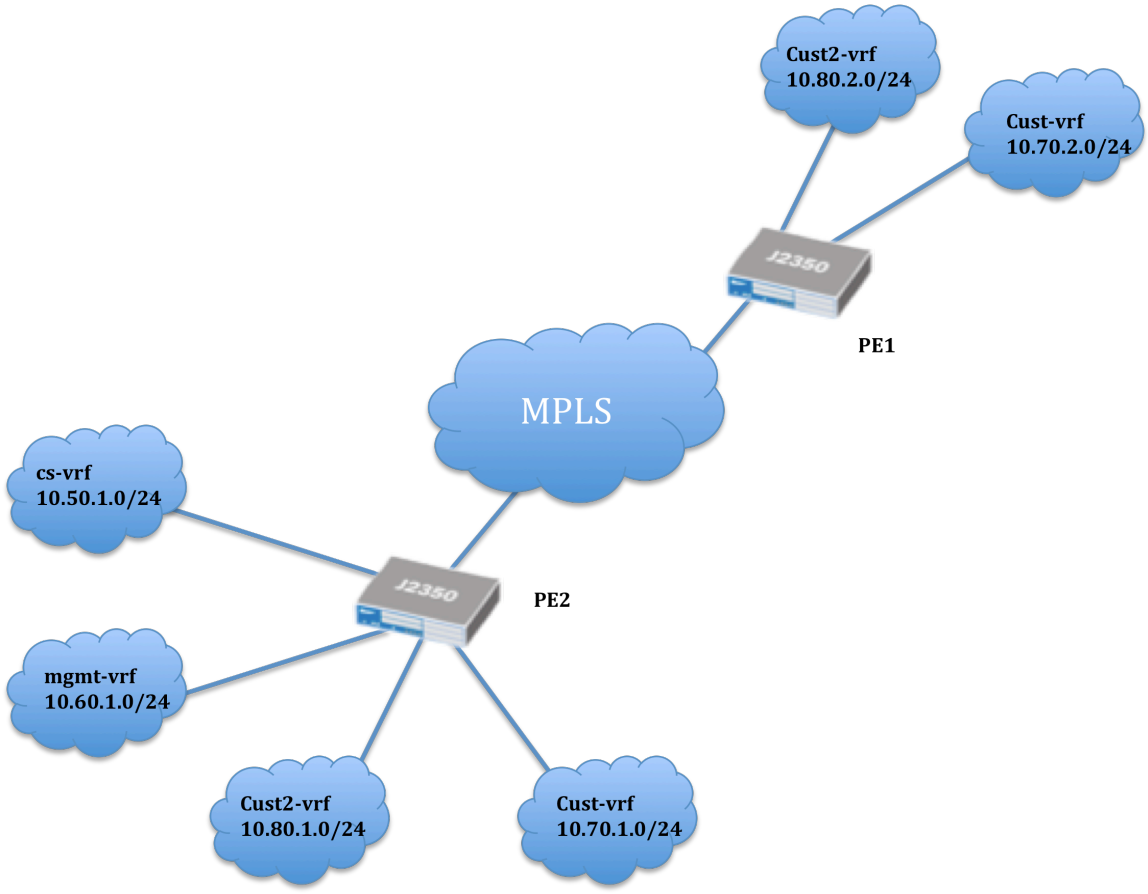
Jack W. Parks, IV
Sr. Networks Engineer
Juniper Networks
6/23/2009

Introduction

The document explains auto-export and shows how it functions. I will use the following drawing to explain the use of auto-export command statement. This document addresses the use of auto-export at a high level.

The following VRFs and address spaces are in use:
Common Services – cs-vrf – target:65000:50 – 10.50.0.0
Management – mgmt-vrf – target:65000:60 – 10.60.0.0
Customer 1 – cust-vrf – target:65000:70 – 10.70.0.0
Customer 2 – cust2-vrf – target:65000:80 – 10.80.0.0

The Common Services and Management VRF are only attached to PE2. We will be sharing prefixes between the Common Services and Management VRFs with the Customer 1 VRF. The Customer 2 VRF is used as a control VRF to show the effect of the auto-export statement on VRFs that do not use custom polices.



Without auto-export configured

Configuration for PE2 (no auto-export applied – standard vrf)

> ***NOTE:*** *Customer 2 VRF (cust2-vrf) uses the "vrf-target" command, which eliminates the need to create a custom policy. It also prevents the Common Services and Management VRFs from "leaking" across VRF tables.*

Common Services and the Management IP prefixes should be shared into the Customer 1 VRF (cust-vrf) only.

```
jparks@J-2350-2# show routing-instances
cs-vrf {
    instance-type vrf;
    interface ge-0/0/0.50;
    route-distinguisher 10.3.3.3:50;
    vrf-target target:65000:50;
    vrf-table-label;
}
cust-vrf {
    instance-type vrf;
    interface ge-0/0/0.70;
    route-distinguisher 10.3.3.3:70;
    vrf-import cust-vrf-import;
    vrf-export cust-vrf-export;
    vrf-table-label;
}
cust2-vrf {
    instance-type vrf;
    interface ge-0/0/0.80;
    route-distinguisher 10.3.3.3:80;
    vrf-target target:65000:80;
    vrf-table-label;
}
mgmt-vrf {
    instance-type vrf;
    interface ge-0/0/0.60;
    route-distinguisher 10.3.3.3:60;
    vrf-target target:65000:60;
    vrf-table-label;
}

[edit]
jparks@J-2350-2# show policy-options
policy-statement cust-vrf-export {
    term 1 {
        from protocol direct;
        then {
            community add cust-vrf-comm;
            accept;
        }
```

```
    }
}
policy-statement cust-vrf-import {
    term 1 {
        from community [ cs-vrf-comm mgmt-vrf-comm cust-vrf-comm ];
        then accept;
    }
    term default {
        then reject;
    }
}
community cs-vrf-comm members target:65000:50;
community cust-vrf-comm members target:65000:70;
community cust2-vrf-comm members target:65000:80;
community mgmt-vrf-comm members target:65000:60;
```

Configuration for PE1 (no auto-export applied – standard vrf)

Common Services and the Management IP prefixes should be shared into the Customer 1 VRF (cust-vrf) only.

```
jparks@J-2350-1# show routing-instances
cust-vrf {
    instance-type vrf;
    interface ge-0/0/0.70;
    route-distinguisher 10.2.2.2:70;
    vrf-import cust-vrf-import;
    vrf-export cust-vrf-export;
    vrf-table-label;
}
cust2-vrf {
    instance-type vrf;
    interface ge-0/0/0.80;
    route-distinguisher 10.2.2.2:80;
    vrf-target target:65000:80;
    vrf-table-label;
}

[edit]
jparks@J-2350-1# show policy-options
policy-statement cust-vrf-export {
    term 1 {
        from protocol direct;
        then {
            community add cust-vrf-comm;
            accept;
        }
    }
}
policy-statement cust-vrf-import {
    term 1 {
        from community [ cs-vrf-comm mgmt-vrf-comm cust-vrf-comm ];
        then accept;
    }
    term default {
```

```
        then reject;
    }
}
community cs-vrf-comm members target:65000:50;
community cust-vrf-comm members target:65000:70;
community cust2-vrf-comm members target:65000:80;
community mgmt-vrf-comm members target:65000:60;
```

If we look into the route tables for each PE we can see and interesting distribution of routes. On PE2, notice the lack of the Common Services (cs-vrf) and Management (mgmt-vrf) IP Prefixes being installed into the Customer 1 (cust-vrf) VRF even though policy allows the prefixes to "leak" across the VRFs.

The PE2 VRF route tables are as follows:

```
cs-vrf.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24       *[Direct/0] 00:58:46
                    > via ge-0/0/0.50
10.50.2.1/32       *[Local/0] 00:58:46
                       Local via ge-0/0/0.50

cust-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.70.1.0/24       *[BGP/170] 00:04:22, localpref 100, from 10.2.2.2
                       AS path: I
                    > to 172.16.2.1 via ge-0/0/0.0, Push 16
10.70.2.0/24       *[Direct/0] 00:56:37
                    > via ge-0/0/0.70
10.70.2.1/32       *[Local/0] 00:56:37
                       Local via ge-0/0/0.70

cust2-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.80.1.0/24       *[BGP/170] 00:04:22, localpref 100, from 10.2.2.2
                       AS path: I
                    > to 172.16.2.1 via ge-0/0/0.0, Push 17
10.80.2.0/24       *[Direct/0] 00:56:37
                    > via ge-0/0/0.80
10.80.2.1/32       *[Local/0] 00:56:37
                       Local via ge-0/0/0.80

mgmt-vrf.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.60.2.0/24       *[Direct/0] 00:56:37
                    > via ge-0/0/0.60
10.60.2.1/32       *[Local/0] 00:56:37
                       Local via ge-0/0/0.60
```

On PE1, notice the Common Services (cs-vrf) and Management (mgmt-vrf) IP Prefixes are being installed into the Customer 1 (cust-vrf) VRF and custom policy allows the prefixes to "leak" across the VRFs.

The PE1 VRF route tables are as follows:

```
cust-vrf.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                    > to 172.16.2.2 via ge-0/0/0.0, Push 16
10.60.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                    > to 172.16.2.2 via ge-0/0/0.0, Push 19
10.70.1.0/24       *[Direct/0] 00:54:22
                    > via ge-0/0/0.70
10.70.1.1/32       *[Local/0] 00:54:22
                      Local via ge-0/0/0.70
10.70.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                    > to 172.16.2.2 via ge-0/0/0.0, Push 17

cust2-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.80.1.0/24       *[Direct/0] 00:54:22
                    > via ge-0/0/0.80
10.80.1.1/32       *[Local/0] 00:54:22
                      Local via ge-0/0/0.80
10.80.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                    > to 172.16.2.2 via ge-0/0/0.0, Push 18
```

With auto-export configured

Configuration for PE2 (auto-export applied)

```
[edit]
jparks@J-2350-2# show groups
auto-export-test {
    routing-instances {
        <*> {
            routing-options {
                auto-export;
            }
        }
    }
}
```

```
[edit]
jparks@J-2350-2# show routing-instances | display inheritance
cs-vrf {
    instance-type vrf;
    interface ge-0/0/0.50;
    route-distinguisher 10.3.3.3:50;
    vrf-target target:65000:50;
    vrf-table-label;
    ##
    ## 'routing-options' was inherited from group 'auto-export-test'
    ##
    routing-options {
        ##
        ## 'auto-export' was inherited from group 'auto-export-test'
        ##
        auto-export;
    }
}
cust-vrf {
    instance-type vrf;
    interface ge-0/0/0.70;
    route-distinguisher 10.3.3.3:70;
    vrf-import cust-vrf-import;
    vrf-export cust-vrf-export;
    vrf-table-label;
    ##
    ## 'routing-options' was inherited from group 'auto-export-test'
    ##
    routing-options {
        ##
        ## 'auto-export' was inherited from group 'auto-export-test'
        ##
        auto-export;
    }
}
cust2-vrf {
    instance-type vrf;
    interface ge-0/0/0.80;
    route-distinguisher 10.3.3.3:80;
    vrf-target target:65000:80;
    vrf-table-label;
    ##
    ## 'routing-options' was inherited from group 'auto-export-test'
    ##
    routing-options {
        ##
        ## 'auto-export' was inherited from group 'auto-export-test'
        ##
        auto-export;
    }
}
mgmt-vrf {
    instance-type vrf;
    interface ge-0/0/0.60;
    route-distinguisher 10.3.3.3:60;
    vrf-target target:65000:60;
    vrf-table-label;
```

```
    ##
    ## 'routing-options' was inherited from group 'auto-export-test'
    ##
    routing-options {
        ##
        ## 'auto-export' was inherited from group 'auto-export-test'
        ##
        auto-export;
    }
}
[edit]
jparks@J-2350-2# show policy-options
policy-statement cust-vrf-export {
    term 1 {
        from protocol direct;
        then {
            community add cust-vrf-comm;
            accept;
        }
    }
}
policy-statement cust-vrf-import {
    term 1 {
        from community [ cs-vrf-comm mgmt-vrf-comm cust-vrf-comm ];
        then accept;
    }
    term default {
        then reject;
    }
}
community cs-vrf-comm members target:65000:50;
community cust-vrf-comm members target:65000:70;
community cust2-vrf-comm members target:65000:80;
community mgmt-vrf-comm members target:65000:60;
```

Configuration for PE1 (no-auto-export applied)

```
[edit]
jparks@J-2350-1# show groups

[edit]
jparks@J-2350-1# show routing-instances
cust-vrf {
    instance-type vrf;
    interface ge-0/0/0.70;
    route-distinguisher 10.2.2.2:70;
    vrf-import cust-vrf-import;
    vrf-export cust-vrf-export;
    vrf-table-label;
}
cust2-vrf {
    instance-type vrf;
    interface ge-0/0/0.80;
    route-distinguisher 10.2.2.2:80;
    vrf-target target:65000:80;
    vrf-table-label;
```

```
}

[edit]
jparks@J-2350-1# show policy-options
policy-statement cust-vrf-export {
    term 1 {
        from protocol direct;
        then {
            community add cust-vrf-comm;
            accept;
        }
    }
}
policy-statement cust-vrf-import {
    term 1 {
        from community [ cs-vrf-comm mgmt-vrf-comm cust-vrf-comm ];
        then accept;
    }
    term default {
        then reject;
    }
}
community cs-vrf-comm members target:65000:50;
community cust-vrf-comm members target:65000:70;
community cust2-vrf-comm members target:65000:80;
community mgmt-vrf-comm members target:65000:60;
```

Auto-export doesn't need to be applied to router PE1. Auto-export only affects local route leaking based on the configured policy of that VRF. When the route tables are displayed for PE2 and PE1 it will be evident what the behavior of auto-export is.

PE2 route tables (notice the Common Services and Management routes in cust-vrf.inet.0):

```
cs-vrf.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24       *[Direct/0] 01:18:56
                    > via ge-0/0/0.50
10.50.2.1/32       *[Local/0] 01:18:56
                      Local via ge-0/0/0.50

cust-vrf.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24       *[Direct/0] 00:00:06
                    > via ge-0/0/0.50
10.50.2.1/32       *[Local/0] 00:00:06
                      Local via ge-0/0/0.50
10.60.2.0/24       *[Direct/0] 00:00:06
                    > via ge-0/0/0.60
10.60.2.1/32       *[Local/0] 00:00:06
                      Local via ge-0/0/0.60
10.70.1.0/24       *[BGP/170] 00:00:06, localpref 100, from 10.2.2.2
```

```
                      AS path: I
                     > to 172.16.2.1 via ge-0/0/0.0, Push 16
10.70.2.0/24       *[Direct/0] 01:16:47
                     > via ge-0/0/0.70
10.70.2.1/32       *[Local/0] 01:16:47
                      Local via ge-0/0/0.70

cust2-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.80.1.0/24       *[BGP/170] 00:00:06, localpref 100, from 10.2.2.2
                      AS path: I
                     > to 172.16.2.1 via ge-0/0/0.0, Push 17
10.80.2.0/24       *[Direct/0] 01:16:47
                     > via ge-0/0/0.80
10.80.2.1/32       *[Local/0] 01:16:47
                      Local via ge-0/0/0.80

mgmt-vrf.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.60.2.0/24       *[Direct/0] 01:16:47
                     > via ge-0/0/0.60
10.60.2.1/32       *[Local/0] 01:16:47
                      Local via ge-0/0/0.60
```

The route tables for PE1 are unchanged:

```
cust-vrf.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                     > to 172.16.2.2 via ge-0/0/0.0, Push 16
10.60.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                     > to 172.16.2.2 via ge-0/0/0.0, Push 19
10.70.1.0/24       *[Direct/0] 00:54:22
                     > via ge-0/0/0.70
10.70.1.1/32       *[Local/0] 00:54:22
                      Local via ge-0/0/0.70
10.70.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
                      AS path: I
                     > to 172.16.2.2 via ge-0/0/0.0, Push 17

cust2-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.80.1.0/24       *[Direct/0] 00:54:22
                     > via ge-0/0/0.80
10.80.1.1/32       *[Local/0] 00:54:22
                      Local via ge-0/0/0.80
10.80.2.0/24       *[BGP/170] 00:11:11, localpref 100, from 10.3.3.3
```

```
                    AS path: I
                  > to 172.16.2.2 via ge-0/0/0.0, Push 18
```

Cust2-vrf is unchanged across the configurations and the vrf-import and vrf-export polices are generic (represented as vrf-target: target:65000:80) and only permit the exchange of prefixes tagged with the community target:65000:80. If we focus on the difference between the before and after look of PE2's cust-vrf-inet.0 you will see the difference between the application of the auto-export command under the VRF.

Before:

```
cust-vrf.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.70.1.0/24        *[BGP/170] 00:04:22, localpref 100, from 10.2.2.2
                       AS path: I
                     > to 172.16.2.1 via ge-0/0/0.0, Push 16
10.70.2.0/24        *[Direct/0] 00:56:37
                     > via ge-0/0/0.70
10.70.2.1/32        *[Local/0] 00:56:37
                       Local via ge-0/0/0.70
```

After:

```
cust-vrf.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0
hidden)
+ = Active Route, - = Last Active, * = Both

10.50.2.0/24        *[Direct/0] 00:00:06
                     > via ge-0/0/0.50
10.50.2.1/32        *[Local/0] 00:00:06
                       Local via ge-0/0/0.50
10.60.2.0/24        *[Direct/0] 00:00:06
                     > via ge-0/0/0.60
10.60.2.1/32        *[Local/0] 00:00:06
                       Local via ge-0/0/0.60
10.70.1.0/24        *[BGP/170] 00:00:06, localpref 100, from 10.2.2.2
                       AS path: I
                     > to 172.16.2.1 via ge-0/0/0.0, Push 16
10.70.2.0/24        *[Direct/0] 01:16:47
                     > via ge-0/0/0.70
10.70.2.1/32        *[Local/0] 01:16:47
                       Local via ge-0/0/0.70
```

Summary

The auto-export command affects the leaking of prefixes between **local** VRFs configured on a given PE based on the vrf-import and vrf-export policies applied to each VRF. Remote prefix leaking is accomplished using BGP extended communities.