IXP Manager & Route Servers

Route Servers Video Tutorial Series - Part 2

Barry O'Donovan - @barryo79, June 2021

Configuring Route Servers with IXP Manager

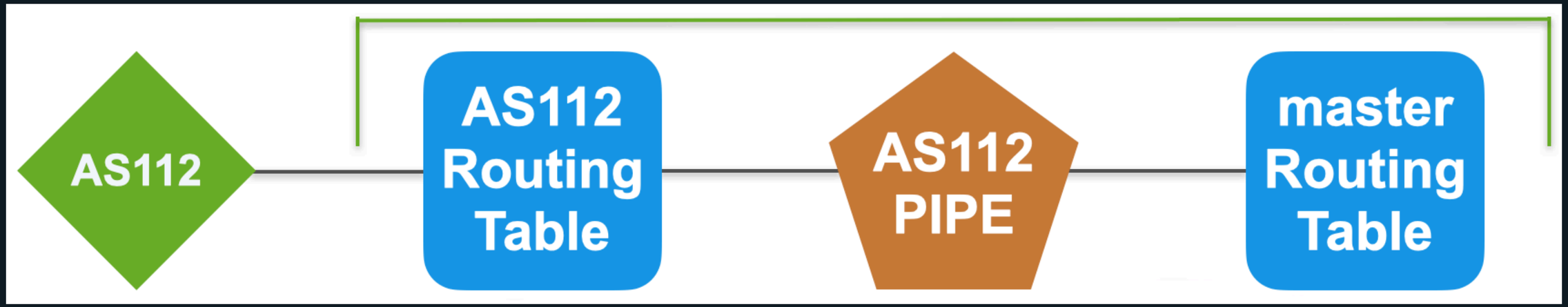
Required Parameters

— AS Number (should be dedicated for the route servers)
— Peering IPv4 and IPv6 address
— Router ID (Usually the peering IPv4 address)
— BGP software (IXP Manager has Bird v1 and v2 baked in)
— Support: MD5? Large communities?
— RPKI?
— Looking glass?

Then add the router in the IXP Manager API...

Examine Client Configuration

Remember:



Examine Client Configuration - Protocol (Peer Defn)

```
protocol bgp pb_0001_as1213 from tb_rsclient {
        description "AS1213 - HEAnet";
        neighbor 192.0.2.32 as 1213;
        ipv4 {
            import limit 100 action restart;
            import filter f_import_as1213;
            table t_0001_as1213;
            export filter f_export_as1213;
        };
        # enable rfc1997 well-known community pass through
        interpret communities off;
        password "yxtRJmDvTYNh";
}
```

Examine Client Configuration - Import Filter

```
filter f_import_as1213 {
    # Filter small prefixes
    if ( net ~ [ 0.0.0.0/0{25,32} ] ) then {
      bgp_large_community.add( IXP_LC_FILTERED_PREFIX_LEN_TOO_LONG );
        accept;
    }

    if !(avoid_martians()) then {
        bgp_large_community.add( IXP_LC_FILTERED_BOGON );
        accept;
    }

    # Peer ASN == route's first ASN?
    if (bgp_path.first != 1213 ) then {
        bgp_large_community.add( IXP_LC_FILTERED_FIRST_AS_NOT_PEER_AS );
        accept;
    }
```

Examine Client Configuration - Import Filter

```
# set of all IPs this ASN uses to peer with on this VLAN
allips = [ 192.0.2.32 ];

# Prevent BGP NEXT_HOP Hijacking
if !( from = bgp_next_hop ) then {

    # need to differentiate between same ASN next hop or actual next hop hijacking
    if( bgp_next_hop ~ allips ) then {
        bgp_large_community.add( IXP_LC_INFO_SAME_AS_NEXT_HOP );
    } else {
        # looks like hijacking (intentional or not)
        bgp_large_community.add( IXP_LC_FILTERED_NEXT_HOP_NOT_PEER_IP );
        accept;
    }
}
```

Examine Client Configuration - Import Filter

```
    # Filter Known Transit Networks
    if filter_has_transit_path() then accept;

    # Belt and braces: no one needs an ASN path with > 64 hops, that's just broken
    if( bgp_path.len > 64 ) then {
        bgp_large_community.add( IXP_LC_FILTERED_AS_PATH_TOO_LONG );
        accept;
    }

    # Skipping RPKI check -> RPKI not enabled / configured correctly.
    bgp_large_community.add( IXP_LC_INFO_RPKI_NOT_CHECKED );

    # This ASN was configured not to use IRRDB filtering
    bgp_large_community.add( IXP_LC_INFO_IRRDB_NOT_CHECKED );

    accept;
}
```

Examine Client Configuration - Protocol (Peer Defn)

```
protocol bgp pb_0001_as1213 from tb_rsclient {
        description "AS1213 - HEAnet";
        neighbor 192.0.2.32 as 1213;
        ipv4 {
            import limit 100 action restart;
            import filter f_import_as1213;
            table t_0001_as1213;
            export filter f_export_as1213;
        };
        # enable rfc1997 well-known community pass through
        interpret communities off;
        password "yxtRJmDvTYNh";
}
```

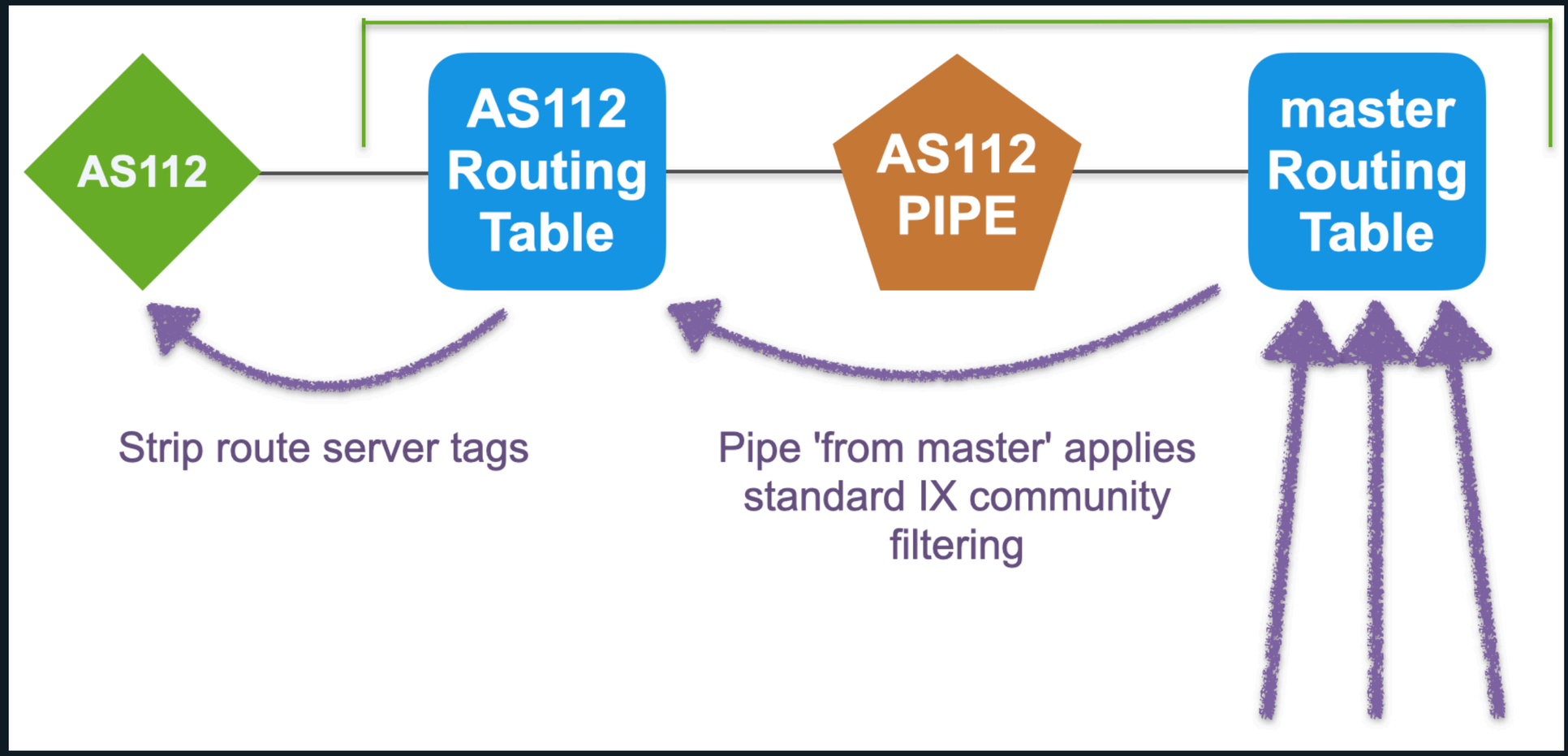
Examine Client Configuration - Export Filter

```
filter f_export_as1213 {

    # strip internal communities
    bgp_large_community.delete( [( routeserverasn, *, * )] );
    bgp_community.delete( [( routeserverasn, * )] );

    accept;
}
```

Examine Client Configuration - Export Filter



Examine Client Configuration - Pipe

```
protocol pipe pp_0001_as1213 {
        description "Pipe for AS1213 - HEAnet";
        table master4;
        peer table t_0001_as1213;
        import filter f_export_to_master;
        export where ixp_community_filter(1213);
}
```

Examine Client Configuration - Pipe

```
...
define IXP_LC_FILTERED_BOGON = ( routeserverasn, 1101, 3  );
...


filter f_export_to_master
{

    if bgp_large_community ~ [( routeserverasn, 1101, * )]
        then reject;

    accept;
}
```

Examine Client Configuration - Pipe

```
protocol pipe pp_0001_as1213 {
        description "Pipe for AS1213 - HEAnet";
        table master4;
        peer table t_0001_as1213;
        import filter f_export_to_master;
        export where ixp_community_filter(1213);
}
```

Examine Client Configuration - Pipe

```
function ixp_community_filter(int peerasn)
{
    # AS path prepending
        ...
    } else if (routeserverasn, 101, peerasn) ~ bgp_large_community then {
        bgp_path.prepend( bgp_path.first );
    }
    ...
}
```

- Client AS65503 tagged their prefix: routeserverasn:101:1213
- Route server will prepend the route from AS65503 with 65503 one extra time when advertising to AS1213

Summary

- We know what route servers are and why they are needed
- We have an understanding of why they need to be secure
- We have seen how IXP Manager generates configuration
- We've examined the basic structure of that config
- We've examined a client configuration
  - Examined import rules and internal tagging
  - Examined standard route server communities
  - Basic understanding of Bird topology
- And we have a working route server configuration!

Coming in Part 3:

Deploying Route Servers with IXP Manager

Thanks for watching!

- https://www.ixpmanager.org/
- https://docs.ixpmanager.org/
- https://www.barryodonovan.com/
- @barryo79 on Twitter
- barry.odonovan@inex.ie