RPKI DOA*

** acronym expansion subject to change*

Job Snijders (Fastly)
Mikael Abrahamsson (NTT)
Ben Maddison (Workonline)

Background: RTBH signalling

- Sometimes, the only available response to a volumetric DDoS attack is to discard all traffic towards the victim
- Some of those times, by the time the traffic is close enough to drop, the damage is already done

Solution: ask an operator upstream of you to discard the traffic closer to the source, by announcing a specific route for the victim prefix, carrying a special-purpose BGP community

Described in detail in RFC3882 and RFC7999.

Why DOA?

We (operators) want to:

- Signal RTBH routes in eBGP
- Filter **all** ROV Invalid routes on ingress

Today, we can't have both...

Why? (cont.)

RTBH routes are typically long prefixes, for maximum granularity

"Normal" unicast routes are generally at most /24 or /48

This leaves operators with two bad choices:

1. Force peers to create ROAs with very long `maxLength`
2. Exempt any path with `BLACKHOLE` from ROV policy; or

Why? (cont..)

Bad option #1:
Force peers to create ROAs with very long `maxLength`:

- Cripples ROV sub-prefix hijack protection for the covering unicast prefix

Why? (cont...)

Bad option #2:
Exempt any path with `BLACKHOLE` from ROV policy:

- Fails to provide even limited origin-based verification
- Very easily abused or mis-configured

Why (misc.)

Plenty of smaller issues:

- Current practise places a lot of faith in correct `NO_EXPORT` handling
- No way to attribute the addition of a community if `AS_PATH` length > 1
- RFC7999 `BLACKHOLE` WKC makes semantics easier, but scoping harder

Proposal

Allow prefix-holders to signal to remote ASs the conditions for honoring RTBH requests:

- Which origins are authorised to inject RTBH routes
- Which communities will be used to signal RTBH intent
- Which prefix lengths RTBH routes may have
- Which peer ASs may an RTBH route be received from

Object processing

ROA-like object processing:

- Based on RFC6488 object template
- Prefix holder signs
- RP validates, flattens, and sends to BGP speaker via RTR protocol

Object `eContent`

High level structure:

```
DiscardOriginAuthorization ::= SEQUENCE {
     version                 [0] INTEGER DEFAULT 0,
     ipAddrBlocks            IPListRange,
     originAsID              ASId,
     peerAsIDs               [1] SEQUENCE SIZE(1..MAX) OF ASId OPTIONAL,
     communities             [2] SEQUENCE SIZE(1..MAX) OF Community
   }
```

Object `eContent` - `version`

Familiar version construct. Nothing to see here.

```
version                 [0] INTEGER DEFAULT 0,
```

Object `eContent` - `ipAddrBlocks`

List of IP prefixes covered by the object, and optional associated prefix length **ranges**

Permitted prefix length is /32 (IPv4) or /128 (IPv6) if ommitted

```
IPListRange ::= SEQUENCE (SIZE(1..MAX)) OF IPAddressFamilyRange
IPAddressFamilyRange ::= SEQUENCE {
   addressFamily        OCTET STRING (SIZE(2..3)),
   addressOrRange       IPAddressOrRange,
   prefixLengthRange    PrefixLengthRange OPTIONAL -- if omitted, assume hostroutes
}
```

Object `eContent` - `originAsID`

AS authorised to originate RTBH routes, exactly like a ROA

```
originAsID              ASId,
```

Object `eContent` - `peerAsIDs`

Optional list of ASs authorised to announce RTBH routes

If omitted, only the AS in `originAsID` may announce RTBH routes (i.e. no transit allowed)

```
peerAsIDs               [1] SEQUENCE SIZE(1..MAX) OF ASId OPTIONAL,
```

Object `eContent` - `communities`

List of BGP standard or large communities that identify a path as an RTBH route

```
Community ::= CHOICE {
   bgpCommunity          [0] OCTET STRING (SIZE(4)),
   bgpLargeCommunity     [1] OCTET STRING (SIZE(12))
}
```

BGP Route Processing

A DOA describes the conditions for a BGP route to be processed as am RTBH signal

Each received BGP path is compared to the set of validated DOAs received from the RP

BGP Route Processing (cont.)

Each route gets an "RTBH request validation state":

- **Matched**: a covering*, validated DOA object was found, and the constraints of the DOA were matched
- **Unmatched**: a covering*, validated DOA object was found , but the constraints of the DOA were not matched
- **NotFound**: no covering*, validated DOA object was found

[*]: using the definition in RFC6811

DOA Constraint Matching

A BGP route matches a validated DOA iff:

- The length of the prefix is within the `prefixLengthRange` of the DOA; *and*
- The origin AS of the route matches the `originAsID`; *and*
- The AS from which the route was received matches the `originAsID` or appears in `peerAsIDs`; *and*
- The BGP route carries at least one standard or large community contained in `communities`

ROV Co-existence & Import Policy

ROV validation state and RTBH request validation state are completely orthogonal - allowing RTBH routes to be identified up front:

```
if route.doa_state == MATCH {
    // ... check some things
    route.next_hop = /dev/null;
    return ACCEPT
} else if route.rov_state == INVALID {
    return REJECT
} else {
    // ... other policy things ...
}
```

Document Status & Next Steps

- Questions, criticisms and applause is welcome
- Document needs plenty of additional detail
- Perhaps split into separate object profile, RTR, validation docs?
- Does the WG want to discuss adoption now, or wait for a more complete product?