Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 31, 2010

P. Mohapatra, Ed.
Cisco Systems
J. Scudder, Ed.
D. Ward, Ed.
Juniper Networks
R. Bush, Ed.
Internet Initiative Japan, Inc.
R. Austein, Ed.
Internet Systems Consortium

April 29, 2010

BGP Prefix Origin Validation
draft-pmohapat-sidr-pfx-validate-07

Abstract

A BGP route associates an address prefix with a set of autonomous systems (AS) that identify the interdomain path the prefix has traversed in the form of BGP announcements. This set is represented as the AS_PATH attribute in BGP and starts with the AS that originated the prefix. To help reduce well-known threats against BGP including prefix mis-announcing and monkey-in-the-middle attacks, one of the security requirements is the ability to validate the origination AS of BGP routes. More specifically, one needs to validate that the AS number claiming to originate an address prefix (as derived from the AS_PATH attribute of the BGP route) is in fact authorized by the prefix holder to do so. This document describes a simple validation mechanism to partially satisfy this requirement.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on October 31, 2010.

Copyright Notice

Table of Contents

1. Introduction

A BGP route associates an address prefix with a set of autonomous systems (AS) that identify the interdomain path the prefix has traversed in the form of BGP announcements. This set is represented as the AS_PATH attribute in BGP [RFC4271] and starts with the AS that originated the prefix. To help reduce well-known threats against BGP including prefix mis-announcing and monkey-in-the-middle attacks, one of the security requirements is the ability to validate the origination AS of BGP routes. More specifically, one needs to validate that the AS number claiming to originate an address prefix (as derived from the AS_PATH attribute of the BGP route) is in fact authorized by the prefix holder to do so. This document describes a simple validation mechanism to partially satisfy this requirement.

The Resource Public Key Infrastructure (RPKI) describes an approach to build a formally verifyable database of IP addresses and AS numbers as resources. The overall architecture of RPKI as defined in [I-D.ietf-sidr-arch] consists of three main components:

- A public key infrastructure (PKI) with the necessary certificate objects,

- Digitally signed routing objects,

- A distributed repository system to hold the objects that would also support periodic retrieval.

The RPKI system is based on resource certificates that define extensions to X.509 to represent IP addresses and AS identifiers [RFC3779], thus the name RPKI. Route Origin Authorizations (ROA) [I-D.ietf-sidr-roa-format] are separate digitally signed objects that define associations between ASes and IP address blocks. Finally the repository system is operated in a distributed fashion through the IANA, RIR hierarchy, and ISPs.

In order to benefit from the RPKI system, it is envisioned that relying parties either at AS or organization level obtain a local copy of the signed object collection, verify the signatures, and process them. The cache must also be refreshed periodically. The exact access mechanism used to retrieve the local cache is beyond the scope of this document.

Individual BGP speakers can utilize the processed data contained in the local cache to validate BGP announcements. The protocol details to retrieve the processed data from the local cache to the BGP speakers is beyond the scope of this document (refer to [I-D.ymbk-rpki-rtr-protocol] for such a mechanism). This document

proposes a means by which a BGP speaker can make use of the processed data in order to assign a "validity state" to each prefix in a received BGP UPDATE message.

Note that the complete path attestation against the AS_PATH attribute of a route is outside the scope of this document.

Although RPKI provides the context for this draft, it is equally possible to use any other database which is able to map prefixes to their authorized origin ASes. Each distinct database will have its own particular operational and security characteristics; such characteristics are beyond the scope of this document.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Prefix-to-AS Mapping Database

In loading the validated objects from the local cache to the BGP speaker, the BGP speaker will store this data in the form of a database that maintains the relationship between prefixes and the corresponding set of authorized origin ASes. The primary key for this database is a prefix set represented as (IP prefix)/[min. length, max. length]. The value stored against each prefix set is the set of AS numbers that is assigned or sub-allocated the corresponding IP address block. An AS may originate more than one prefix set. Thus, multiple prefix sets in the database may contain the same origin AS(es).

Whenever UPDATEs are received from peers, a BGP speaker is expected to perform a lookup in this database for each of the prefixes in the UPDATE message. To aid with better description, we define terms "UPDATE prefix" and "UPDATE origin AS number" to denote the values derived from the received UPDATE message, and "database prefix set" and "database origin AS number set" to mean the values derived from the database lookup. Note that in the presence of overlapping prefixes, the database lookup against the "UPDATE prefix" may yield multiple matches.

The following are the different types of results expected from such a lookup operation:

o If the "UPDATE prefix" finds no matching or covering prefixes in the database (i.e. the "UPDATE prefix" is not a sub-block of any

of the database prefixes), the lookup result is returned as "not found". Due to incremental deployment model of the RPKI repository, it is expected that a complete registry of all IP address blocks and their AS associations is not available at a given point of time.

- o If there are "database prefix sets" that cover the "UPDATE prefix", and one of them has the "UPDATE origin AS number" in the "database origin AS number sets", then the lookup result is returned as "valid".

- o If there are "database prefix sets" which cover the "UPDATE prefix", but none of them has the "UPDATE origin AS number" in the "database origin AS number set", then the lookup result is returned as "invalid".

Depending on the lookup result, we define a property for each route, called the "validity state". It can assume the values "valid", "not found", or "invalid".

Note that all the routes, regardless of their "validity state" will be stored in the local BGP speaker's Adj-RIB-In.

Following is a sample pseudo code for prefix validation function:

```
//Input are the variables derived from a BGP UPDATE message
//that need to be validated.
//origin_as is the rightmost AS in the final AS_SEQUENCE of
//the AS_PATH attribute in the UPDATE message.
//
//If the UPDATE message carries [AS4_]AGGREGATOR attribute,
//origin_as is derived from the AS field of that attribute.
//
//origin_as is NONE if the AS_PATH begins with a non-trivial
//AS_SET and has no [AS4_]AGGREGATOR attribute.
input = {bgp_prefix, masklen, origin_as};

//Initialize result to "not found" state
result = BGP_PFXV_STATE_NOT_FOUND;

//pfx_validate_table organizes all the ROA entries retrieved
//from RPKI cache based on the IP address and the minLength
//field. There can be multiple such entries that match the
//input. Iterate through all of them.
entry = next_lookup_result(pfx_validate_table,
                           input.bgp_prefix, input.masklen);
```

```
   while (entry != NULL) {
       prefix_exists = TRUE;

       //Each entry stores multiple records sorted by the ROA
       //maxLength field. i.e. there can be multiple ROA records
       //with the same IPaddress and minLength fields, but different
       //maxLength field. Iterate through all records of the entry
       //to check if there is one range that matches the input.
       record = next_in_entry_record_list(entry);
       while (record != NULL) {
           if (input.masklen <= record->max_length) {
               if (input.origin_as == record->origin_as) {
                   result = BGP_PFXV_STATE_VALID;
                   return (result);
               }
           }
       }
   }

   //If pfx_validate_table contains one or more prefixes that
   //match the input, but none of them resulted in a "valid"
   //outcome since the origin_as did not match, return the
   //result state as "invalid". Else the initialized state of
   //"not found" applies to this validation operation.
   if (prefix_exists == TRUE) {
       result = BGP_PFXV_STATE_INVALID;
   }

   return (result);
```

3. Policy Control

An implementation MUST provide the ability to match and set the validation state of routes as part of its route policy filtering function. Use of validation state in route policy is elaborated in Section 6.

4. Route Aggregation

When an UPDATE message carries AGGREGATOR attribute, the "UPDATE origin AS number" is set to the value encoded in the AGGREGATOR instead of being derived from the AS_PATH attribute.

5. Interaction with Local Cache

Each BGP speaker supporting prefix validation as described in this document is expected to communicate with one or multiple local caches that store a database of RPKI signed objects. The protocol mechanisms used to fetch the data and store them locally at the BGP speaker is beyond the scope of this document (please refer [I-D.ymbk-rpki-rtr-protocol]). Irrespective of the protocol, the prefix validation algorithm as outlined in this document is expected to function correctly in the event of failures and other timing conditions that may result in an empty and/or partial prefix-to-AS mapping database. Indeed, if the (in-PoP) cache is not available and the mapping database is empty on the BGP speaker, all the lookups will result in "not found" state and the prefixes will be advertised to rest of the network (unless restricted by policy configuration). Similarly, if BGP UPDATEs arrive at the speaker while the fetch operation from the cache is in progress, some prefix lookups will also result in "not found" state. The implementation is expected to handle these timing conditions and MUST re-validate affected prefixes once the fetch operation is complete. The same applies during any subsequent incremental updates of the validation database.

In the event that connectivity to the cache is lost, the router should make a reasonable effort to fetch a new validation database (either from the same, or a different cache), and SHOULD wait until the new validation database has been fetched before purging the previous one. A configurable timer MUST be provided to bound the length of time the router will wait before purging the previous validation database.

6. Deployment Considerations

Once a route is received from an EBGP peer it is categorized according the procedure given in Section 2. Subsequently, routing policy as discussed in Section 3 can be used to take action based on the validation state.

Policies which could be implemented include filtering routes based on validation state (for example, rejecting all "invalid" routes) or adjusting a route's degree of preference in the selection algorithm based on its validation state. The latter could be accomplished by adjusting the value of such attributes as LOCAL_PREF.

In some cases (particularly when the selection algorithm is influenced by the adjustment of a route property that is not propagated into IBGP) it could be necessary for routing correctness to propagate the validation state to the IBGP peer. This can be

accomplished on the sending side by setting a community or extended community based on the validation state, and on the receiving side by matching the (extended) community and setting the validation state.

7. Contributors

Rex Fernando rex@cisco.com
Keyur Patel keyupate@cisco.com
Cisco Systems

Miya Kohno mkohno@juniper.net
Juniper Networks

Shin Miyakawa miyakawa@nttv6.jp
Taka Mizuguchi
Tomoya Yoshida
NTT Communications

Russ Housley housley@vigilsec.com
Vigil Security

Junaid Israr jisra052@uottawa.ca
Mouhcine Guennoun mguennou@uottawa.ca
Hussein Mouftah mouftah@site.uottawa.ca
University of Ottawa School of Information Technology and
Engineering(SITE) 800 King Edward Avenue, Ottawa, Ontario, Canada,
K1N 6N5

8. Acknowledgements

Junaid Israr's contribution to this specification is part of his PhD research work and thesis at University of Ottawa, Canada.

9. IANA Considerations

10. Security Considerations

Although this specification discusses one portion of a system to validate BGP routes, it should be noted that it relies on a database (RPKI or other) to provide validation information. As such, the security properties of that database must be considered in order to determine the security provided by the overall solution. If "invalid" routes are blocked as this specification suggests, the overall system provides a possible denial-of-service vector, for

example if an attacker is able to inject one or more spoofed records into the validation database which lead a good route to be declared invalid. In addition, this system is only able to provide limited protection against a determined attacker -- the attacker need only prepend the "valid" source AS to a forged BGP route announcement in order to defeat the protection provided by this system. This mechanism does not protect against "AS in the middle attacks" or provide any path validation. It only attempts to verify the origin. In general, this system should be thought of more as a protection against misconfiguration than as true "security" in the strong sense.

11. References

11.1. Normative References

[I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-09 (work in progress), October 2009.

[I-D.ietf-sidr-roa-format]
Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", draft-ietf-sidr-roa-format-06 (work in progress), October 2009.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

11.2. Informative References

[I-D.ymbk-rpki-rtr-protocol]
Bush, R. and R. Austein, "The RPKI/Router Protocol", draft-ymbk-rpki-rtr-protocol-04 (work in progress), July 2009.

Authors' Addresses

Pradosh Mohapatra (editor)
Cisco Systems
170 W. Tasman Drive
San Jose, CA 95134
USA

Email: pmohapat@cisco.com

John Scudder (editor)
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA 94089
USA

Email: jgs@juniper.net

David Ward (editor)
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA 94089
USA

Email: dward@juniper.net

Randy Bush (editor)
Internet Initiative Japan, Inc.
5147 Crystral Springs
Bainbridge Island, Washington 98110
USA

Email: randy@psg.com

Rob Austein (editor)
Internet Systems Consortium
950 Charter Street
Redwood City, CA 94063
USA

Email: sra@isc.org