Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2021

A. Azimov
Yandex
E. Uskov
JetLend
R. Bush
Internet Initiative Japan
K. Patel
Arrcus
J. Snijders
NTT
R. Housley
Vigil Security

February 22, 2021

A Profile for Autonomous System Provider Authorization
draft-ietf-sidrops-aspa-profile-05

Abstract

This document defines a standard profile for Autonomous System Provider Authorization in the Resource Public Key Infrastructure. An Autonomous System Provider Authorization is a digitally signed object that provides a means of verifying that a Customer Autonomous System holder has authorized members of Provider set to be its upstream providers and for the Providers to send prefixes received from the Customer Autonomous System in all directions including providers and peers.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

The primary purpose of the Resource Public Key Infrastructure (RPKI) is to improve routing security. (See [RFC6480] for more information.) As part of this infrastructure, a mechanism is needed to verify that a AS has permission from a Customer AS (CAS) holder to send routes in all directions. The digitally signed Autonomous

System Provider Authorization (ASPA) object provides this verification mechanism.

The ASPA uses the template for RPKI digitally signed objects [RFC6488], which defines a Cryptographic Message Syntax (CMS) [RFC5652] wrapper for the ASPA content as well as a generic validation procedure for RPKI signed objects. As ASPAs need to be validated with RPKI certificates issued by the current infrastructure, we assume the mandatory-to-implement algorithms in [RFC6485], or its successor.

To complete the specification of the ASPA (see Section 4 of [RFC6488]), this document defines:

1. The object identifier (OID) that identifies the ASPA signed object. This OID appears in the eContentType field of the encapContentInfo object as well as the content-type signed attribute within the signerInfo structure).

2. The ASN.1 syntax for the ASPA content, which is the payload signed by the CAS. The ASPA content is encoded using the ASN.1 [X680] Distinguished Encoding Rules (DER) [X690].

3. The steps required to validate an ASPA beyond the validation steps specified in [RFC6488]).

2. The ASPA Content Type

The content-type for an ASPA is defined as id-cct-ASPA, which has the numerical value of 1.2.840.113549.1.9.16.1.TBD. This OID MUST appear both within the eContentType in the encapContentInfo structure as well as the content-type signed attribute within the signerInfo structure (see [RFC6488]).

3. The ASPA eContent

The content of an ASPA identifies the Customer AS (CAS) as well as the Set of Provider ASes (SPAS) that are authorized to further propagate announcements received from the customer. If customer has multiple providers they MUST be registered in a single ASPA object. This rule is important to avoid possible race conditions during updates. An ASPA is formally defined as:

```
ct-ASPA CONTENT-TYPE ::=
    { ASProviderAttestation IDENTIFIED BY id-ct-ASPA }

id-ct-ASPA OBJECT IDENTIFIER ::= { id-ct TBD }

ASProviderAttestation ::= SEQUENCE {
    version [0] ASPAVersion DEFAULT v0,
    aFI  AddressFamilyIdentifier,
    customerASID  ASID,
    providerASSET  SEQUENCE (SIZE(1..MAX)) OF ASID }

ASPAVersion ::= INTEGER  { v0(0) }

AddressFamilyIdentifier ::= OCTET STRING (SIZE (2))

ASID ::= INTEGER
```

Note that this content appears as the eContent within the encapContentInfo as specified in [RFC6488].

3.1. version

The version number of the ASProviderAttestation MUST be v0.

3.2. AFI

The AFI field contains Address Family Identifier for which the relation between customer and provider ASes is authorized. Presently defined values for the Address Family Identifier field are specified in the IANA's Address Family Numbers registry [IANA-AF].

3.3. customerASID

The customerASID field contains the AS number of the Autonomous System that authorizes an upstream providers (listed in the providerASSET) to propagate prefixes in the specified address family other ASes.

3.4. providerASSET

The providerASSET contains the sequence (set) of AS numbers that are authorized to further propagate announcements in the specified address family received from the customer.

4. ASPA Validation

Before a relying party can use an ASPA to validate a routing announcement, the relying party MUST first validate the ASPA object itself. To validate an ASPA, the relying party MUST perform all the validation checks specified in [RFC6488] as well as the following additional ASPA-specific validation step.

- The autonomous system identifier delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ASPA), and the customer AS number in the ASPA is contained within the set of AS numbers specified by the EE certificate's autonomous system identifier delegation extension.

5. ASN.1 Module for the ASPA Content Type

```
RPKI-ASPA-2020
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
     pkcs-9(9) smime(16) modules(0) id-mod-rpki-aspa-2020(TBD2) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
IMPORTS

CONTENT-TYPE
FROM CryptographicMessageSyntax-2010  -- RFC 6268
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
       pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) } ;

ContentSet CONTENT-TYPE ::= { ct-ASPA, ... }

--
-- ASPA Content Type
--

id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
     us(840) rsadsi(113549) pkcs(1) pkcs-9(9) 16 }

id-ct OBJECT IDENTIFIER ::= { id-smime 1 }

id-ct-ASPA OBJECT IDENTIFIER ::= { id-ct TBD }

ct-ASPA CONTENT-TYPE ::=
    { TYPE ASProviderAttestation IDENTIFIED BY id-ct-ASPA }

ASProviderAttestation ::= SEQUENCE {
  version [0] ASPAVersion DEFAULT v0,
  aFI  AddressFamilyIdentifier,
  customerASID  ASID,
  providerASSET  SEQUENCE (SIZE(1..MAX)) OF ASID OPTIONAL }

ASPAVersion ::= INTEGER  { v0(0) }

AddressFamilyIdentifier ::= OCTET STRING (SIZE (2))

ASID ::= INTEGER

END
```

6. IANA Considerations

Please add the id-mod-rpki-aspa-2018 to the SMI Security for S/MIME Module Identifier (1.2.840.113549.1.9.16.0) registry (https://www.iana.org/assignments/smi-numbers/smi-numbers.xml#security-smime-0) as follows:

Decimal	Description	Specification
TBD2	id-mod-rpki-aspa-2020	[ThisRFC]

Please add the ASPA to the SMI Security for S/MIME CMS Content Type (1.2.840.113549.1.9.16.1) registry (https://www.iana.org/assignments/smi-numbers/smi-numbers.xml#security-smime-1) as follows:

Decimal	Description	Specification
TBD	id-ct-ASPA	[ThisRFC]

Please add the ASPA to the RPKI Signed Object registry (https://www.iana.org/assignments/rpki/rpki.xhtml#signed-objects) as follows:

Name	OID	Specification
ASPA	1.2.840.113549.1.9.16.1.TBD	[ThisRFC]

7. Security Considerations

While it's not restricted, but it's highly recommended maintaining for selected Customer AS a single ASPA object that covers all its providers. Such policy should prevent race conditions during ASPA updates that might affect prefix propagation. The software that provides hosting for ASPA records SHOULD support enforcement of this rule. In the case of the transition process between different CA registries, the ASPA records SHOULD be kept identical in all registries.

8. Acknowledgments

9. References

9.1. Normative References

[IANA-AF] IANA, "Address Family Numbers", <https://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <https://www.rfc-editor.org/info/rfc3779>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6485] Huston, G., "The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure (RPKI)", RFC 6485, DOI 10.17487/RFC6485, February 2012, <https://www.rfc-editor.org/info/rfc6485>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <https://www.rfc-editor.org/info/rfc6488>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[X680] ITU-T, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, 2015.

[X690] ITU-T, "Information Technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, 2015.

9.2. Informative References

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

Authors' Addresses

Alexander Azimov
Yandex

Email: a.e.azimov@gmail.com

Eugene Uskov
JetLend

Email: eu@jetlend.ru

Randy Bush
Internet Initiative Japan

Email: randy@psg.com

Keyur Patel
Arrcus, Inc.

Email: keyur@arrcus.com

Job Snijders
NTT Communications
Theodorus Majofskistraat 100
Amsterdam 1065 SZ
The Netherlands

Email: job@ntt.net

Russ Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

Email: housley@vigilsec.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2021

A. Azimov
Yandex
E. Bogomazov
Qrator Labs
R. Bush
Internet Initiative Japan & Arrcus
K. Patel
Arrcus, Inc.
J. Snijders
NTT

February 22, 2021

Verification of AS_PATH Using the Resource Certificate Public Key Infrastructure and Autonomous System Provider Authorization
draft-ietf-sidrops-aspa-verification-07

Abstract

This document defines the semantics of an Autonomous System Provider Authorization object in the Resource Public Key Infrastructure to verify the AS_PATH attribute of routes advertised in the Border Gateway Protocol.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 26, 2021.

Table of Contents

1. Introduction

The Border Gateway Protocol (BGP) was designed without mechanisms to validate BGP attributes. Two consequences are BGP Hijacks and BGP Route Leaks [RFC7908]. BGP extensions are able to partially solve these problems. For example, ROA-based Origin Validation [RFC6483] can be used to detect and filter accidental mis-originations, and [I-D.ietf-idr-bgp-open-policy] or [I-D.ietf-grow-route-leak-detection-mitigation] can be used to detect accidental route leaks. While these upgrades to BGP are quite useful, they still rely on transitive BGP attributes, i.e. AS_PATH, that can be manipulated by attackers.

BGPSec [RFC8205] was designed to solve the problem of AS_PATH validation. Unfortunately, strict cryptographic validation brought expensive computational overhead for BGP routers. BGPSec also proved vulnerable to downgrade attacks that nullify the benefits of AS_PATH signing. As a result, to abuse the AS_PATH or any other signed transit attribute, an attacker merely needs to downgrade to 'old' BGP-4.

An alternative approach was introduced with soBGP [I-D.white-sobgp-architecture]. Instead of strong cryptographic AS_PATH validation, it created an AS_PATH security function based on a shared database of AS adjacencies. While such an approach has reasonable computational cost, the two side adjacencies don't provide a way to automate anomaly detection without high adoption rate - an attacker can easily create a one-way adjacency. SO-BGP transported data about adjacencies in new additional BGP messages, which was recursively complex thus significantly increasing adoption complexity and risk. In addition, the general goal to verify all AS_PATHs was not achievable given the indirect adjacencies at internet exchange points.

Instead of checking AS_PATH correctness, this document focuses on solving real-world operational problems - automatic detection of malicious hijacks and route leaks. To achieve this new AS_PATH verification procedures are defined to automatically detect invalid (malformed) AS_PATHs in announcements that are received from customers, peers, providers, RS and RS-clients. This procedures uses a shared signed database of customer-to-provider relationships using a new RPKI object - Autonomous System Provider Authorization (ASPA). This technique provides benefits for participants even during early and incremental adoption.

2. Anomaly Propagation

Both route leaks and hijacks have similar effects on ISP operations - they redirect traffic, resulting in increased latency, packet loss, or possible MiTM attacks. But the level of risk depends significantly on the propagation of the anomalies. For example, a hijack that is propagated only to customers may concentrate traffic in a particular ISP's customer cone; while if the anomaly is propagated through peers, upstreams, or reaches Tier-1 networks, thus distributing globally, traffic may be redirected at the level of entire countries and/or global providers.

The ability to constrain propagation of BGP anomalies to upstreams and peers, without requiring support from the source of the anomaly (which is critical if source has malicious intent), should

significantly improve the security of inter-domain routing and solve the majority of problems.

3. Autonomous System Provider Authorization

As described in [RFC6480], the RPKI is based on a hierarchy of resource certificates that are aligned to the Internet Number Resource allocation structure. Resource certificates are X.509 certificates that conform to the PKIX profile [RFC5280], and to the extensions for IP addresses and AS identifiers [RFC3779]. A resource certificate is a binding by an issuer of IP address blocks and Autonomous System (AS) numbers to the subject of a certificate, identified by the unique association of the subject's private key with the public key contained in the resource certificate. The RPKI is structured so that each current resource certificate matches a current resource allocation or assignment.

ASPA is digitally signed object that bind, for a selected AFI, a Set of Provider AS numbers to a Customer AS number (in terms of BGP announcements not business), and are signed by the holder of the Customer AS. An ASPA attests that a Customer AS holder (CAS) has authorized Set of Provider ASes (SPAS) to propagate the Customer's IPv4/IPv6 announcements onward, e.g. to the Provider's upstream providers or peers. The ASPA record profile is described in [I-D.ietf-sidrops-aspa-profile]. For a selected Customer AS SHOULD exist only single ASPA object at any time. In this document we will use ASPA(AS1, AFI, [AS2, ...]) as notation to represent ASPA object for AS1 in the selected AFI.

4. Customer-Provider Verification Procedure

This section describes an abstract procedure that checks that a pair of ASNs (AS1, AS2) is included in the set of signed ASPAs. The semantics of its use is defined in next section. The procedure takes (AS1, AS2, AFI) as input parameters and returns one of three results: "Valid", "Invalid" and "Unknown".

A relying party (RP) must have access to a local cache of the complete set of cryptographically valid ASPAs when performing customer-provider verification procedure.

1. Retrieve all cryptographically valid ASPAs in a selected AFI with a customer value of AS1. The union of SPAS forms the set of "Candidate Providers."

2. If the set of Candidate Providers is empty, then the procedure exits with an outcome of "Unknown."

3. If AS2 is included in the Candidate Providers, then the procedure exits with an outcome of "Valid."

4. Otherwise, the procedure exits with an outcome of "Invalid."

Since an AS1 may have different set of providers in different AFI, it should also have different PCAS in corresponding ASPAs. In this case, the output of this procedure with input (AS1, AS2, AFI) may have different output for different AFI values.

5. AS_PATH Verification

The AS_PATH attribute identifies the autonomous systems through which an UPDATE message has passed. AS_PATH may contain two types of components: AS_SEQUENCEs and AS_SETs, as defined in [RFC4271].

We will use index of AS_PATH segments, where Seg(0) stands for the segment of originating AS. We will use Seg(I).value and Seg(I).type to represent Ith segment value and type respectively.

The below procedures are applicable only for 32-bit AS number compatible BGP speakers.

5.1. Upstream Paths

When a route is received from a customer, a literal peer, or by a RS at an IX, each consecutive AS_SEQUENCE pair MUST be equal (prepend policy) or belong to customer-provider or mutual transit relationship (Section 7). If there are other types of relationships, it means that the route was leaked or the AS_PATH attribute was malformed. The goal of the procedure described below is to check the correctness of this statement.

The following Python function and algorithm describes the procedure that MUST be applied on routes with AFI received from a customer, peer or RS-client:

```
    def check_upflow_path(aspath, neighbor_as, afi):
        if len(aspath) == 0:
            return Invalid

        if aspath[-1].type == AS_SEQUENCE and aspath[-1].value != neighbor_as:
            return Invalid

        semi_state = Valid

        as1 = 0
        for segment in aspath:
            if segment.type != AS_SEQUENCE:
                as1 = 0
                semi_state = Unverifiable
            elif segment.type == AS_SEQUENCE:
                if not as1:
                    as1 = segment.value
                elif as1 == segment.value:
                    continue
                else:
                    pair_check = verify_pair(as1, segment.value, afi)
                    if pair_check == Invalid:
                        return Invalid
                    elif pair_check == Unknown and semi_state == Valid:
                        semi_state = pair_check
                    as1 = segment.value
        return semi_state
```

1. If the AS_PATH has zero length then procedure halts with the outcome "Invalid";

2. If the last segment in the AS_PATH has type AS_SEQUENCE and its value isn't equal to receiver's neighbor AS then procedure halts with the outcome "Invalid";

3. If there exists I such that Seg(I-1).type and Seg(I).type equal to AS_SEQUENCE, Seg(I-1).value != Seg(I).value and customer-provider verification procedure (Section 4) with parameters (Seg(I-1).value, Seg(I).value, AFI) returns "Invalid" then the procedure also halts with the outcome "Invalid";

4. If the AS_PATH has at least one AS_SET segment then procedure halts with the outcome "Unverifiable";

5. If there exists I such that Seg(I-1).type and Seg(I).type equal to AS_SEQUENCE, Seg(I-1).value != Seg(I).value and customer-provider verification procedure (Section 4) with parameters

(Seg(I-1).value, Seg(I).value, AFI) returns "Unknown" then the procedure also halts with the outcome "Unknown";

6. Otherwise, the procedure halts with an outcome of "Valid".

5.2. Downstream Paths

When route is received from provider it may have both Upstream and Downstream fragments, where a Downstream follows an Upstream fragment. If the path differs from this rule, e.g. the Downstream fragment is followed by Upstream fragment it means that the route was leaked or the AS_PATH attribute was malformed. The first unequal pair of AS_SEQUENCE segments that has an "Invalid" outcome of the customer-provider verification procedure indicates the end of the Upstream fragment. All subsequent reverse pairs of AS_SEQUENCE segments MUST be equal (prepend policy) or belong to a customer-provider or mutual transit relationship Section 7, thus can be also verified using ASPA objects.

The following Python function and algorithm describe the procedure that MUST be applied on routes with AFI received from a provider:

```
    def check_downflow_path(aspath, neighbor_as, afi):
        if len(aspath) == 0:
            return Invalid

        if aspath[-1].type == AS_SEQUENCE and aspath[-1].value != neighbor_as:
            return Invalid
        else:
            semi_state = Valid

        as1 = 0
        upflow_fragment = True
        for segment in aspath:
            if segment.type != AS_SEQUENCE:
                as1 = 0
                semi_state = Unverifiable
            elif segment.type == AS_SEQUENCE:
                if not as1:
                    as1 = segment.value
                elif as1 == segment.value:
                    continue
                else:
                    if upflow_fragment:
                        pair_check = verify_pair(as1, segment.value, afi)
                        if pair_check == Invalid:
                            upflow_fragment = False
                        elif pair_check == Unknown and semi_state == Valid:
                            semi_state = Unknown
                    else:
                        pair_check = verify_pair(segment.value, as1, afi)
                        if pair_check == Invalid:
                            return Invalid
                        elif pair_check == Unknown and semi_state == Valid:
                            semi_state = pair_check
                    as1 = segment.value

        return semi_state
```

1. If the AS_PATH has zero length then procedure halts with the outcome "Invalid";

2. If a route is received from a provider and the last segment in the AS_PATH has type AS_SEQUENCE and its value isn't equal to receiver's neighbor AS, then the procedure halts with the outcome "Invalid";

3. Let's define I_MIN as the minimal index for which Seg(I-1).type and Seg(I).type equal to AS_SEQUENCE, its values aren't equal and the verification procedure for (Seg(I-1).value, Seg(I).value,

AFI) returns "Invalid". If I_MIN doesn't exist put the length of AS_PATH in I_MIN variable and jump to 5.

4. If there exists J > I_MIN such that both Seg(J-1).type, Seg(J).type equal to AS_SEQUENCE, Seg(J-1).value != Seg(J).value and the customer-provider verification procedure (Section 4) returns "Invalid" for (Seg(J).value, Seg(J-1).value, AFI), then the procedure halts with the outcome "Invalid";

5. If the AS_PATH has at least one AS_SET segment then procedure halts with the outcome "Unverifiable";

6. If there exists J > I_MIN such that both Seg(J-1).type, Seg(J).type equal to AS_SEQUENCE, Seg(J-1).value != Seg(J).value and the customer-provider verification procedure (Section 4) returns "Unknown" for (Seg(J).value, Seg(J-1).value, AFI), then the procedure halts with the outcome "Unknown";

7. If there exists I_MIN > J such that both Seg(J-1).type, Seg(J).type equal to AS_SEQUENCE, Seg(J-1).value != Seg(J).value and the customer-provider verification procedure (Section 4) returns "Unknown" for (Seg(J-1).value, Seg(J).value, AFI), then the procedure halts with the outcome "Unknown";

8. Otherwise, the procedure halts with an outcome of "Valid".

5.3. Paths from Route Server

A route received from a RS at IX has much in common with route received from a provider. A valid route from RS contains Upflow fragment and MAY contain Downflow fragment that contains IX AS. The ambiguity is created by transparent IXes that by default don't add their AS in the AS_PATH. In this case, a route will have only Upflow segment, though even 'transparent' IXes may support control communities that give a way to explicitly add IX AS in the path.

Routes from RS MAY be processed the same way as routes from Providers, but in the case of full IX 'transparency', it will limit the opportunity of IX members to detect and filter route leaks. This document suggests using the presence of IX AS as a token to distinguish if Upflow or Downflow path verification procedure should be applied.

The following Python function and algorithm describe the procedure that SHOULD be applied on routes with AFI received from a RS:

```
def check_ix_path(aspath, neighbor_as, afi):
    if len(aspath) == 0:
        return Invalid

    if aspath[-1].value != neighbor_as:
        return check_upflow_path(aspath, aspath[-1].value, afi)
    else:
        return check_downflow_path(aspath, neighbor_as, afi)
```

1. If the AS_PATH has zero length then procedure halts with the outcome "Invalid";

2. If a route is received from a RS and the last segment in the AS_PATH isn't equal to receiver's neighbor AS, the result equals to the outcome of upflow verification procedure applied to AS_PATH with neighbor_as replaced with the value of the last AS_PATH segment Section 5.1;

3. If a route is received from a RS and the last segment in the AS_PATH is equal to receiver's neighbor AS, the result equals to the outcome of downflow verification procedure applied to AS_PATH Section 5.2;

5.4. Mitigation

If the output of the AS_PATH verification procedure is "Invalid" the route MUST be rejected.

If the output of the AS_PATH verification procedure is 'Unverifiable' it means that AS_PATH can't be fully checked. Such routes should be treated with caution and SHOULD be processed the same way as "Invalid" routes. This policy goes with full correspondence to [I-D.kumari-deprecate-as-set-confed-set].

The above AS_PATH verification procedure is able to check routes received from customer, peers, providers, RS, and RS-clients. The ASPA mechanism combined with BGP Roles [I-D.ietf-idr-bgp-open-policy] and ROA-based Origin Validation [RFC6483] can provide a fully automated solution to detect and filter hijacks and route leaks, including malicious ones.

6. Disavowal of Provider Authorizaion

An ASPA is a positive attestation that an AS holder has authorized its providers to redistribute received routes to the provider's providers and peers. This does not preclude the provider ASes from redistribution to its other customers. By creating an ASPA with providers set of [0], the customer indicates that no provider should

further announce its routes. Specifically, AS 0 is reserved to identify provider-free networks, Internet exchange meshes, etc.

An ASPA(AS, AFI, [0]) is a statement by the customer AS that its routes should not be received by any relying party AS from any of its customers or peers.

By convention, an ASPA(AS, AFI, [0]) should be the only ASPA issued by a given AS holder in the selected AFI; although this is not a strict requirement. An AS 0 may coexist with other provider ASes in the same ASPA (or other ASPA records in the same AFI); though in such cases, the presence or absence of the provider AS 0 in ASPA does not alter the AS_PATH verification procedure.

7. Mutual Transit (Complex Relations)

There are peering relationships which can not be described as strictly simple peer-peer or customer-provider; e.g. when both parties are intentionally sending prefixes received from each other to their peers and/or upstreams.

In this case, two corresponding records ASPA(AS1, AFI, [AS2, ...]), ASPA(AS2, AFI, [AS1, ...]) must be created by AS1 and AS2 respectively.

8. Comparison to Peerlock

ASPA has much in common with [Peerlock]. Peerlock is a BGP Flexsealing [Flexsealing] protection mechanism commonly deployed by global-scale Internet carriers to protect other large-scale carriers.

Peerlock, unfortunately, depends on a laborious manual process in which operators coordinate the distribution of unstructured Provider Authorizations through out-of-band means in a many-to-many fashion. On the other hand, ASPA's use of PKIX [RFC5280] allows for automated, scalable, and ubiquitous deployment, making the protection mechanism available to a wider range of Internet Number Resource holders.

ASPA mechanics implemented in code instead of Peerlock AS_PATH regular expressions also provides a way to detect anomalies coming from transit providers and internet exchange route servers.

ASPA is intended to be a complete solution and replacement for existing Peerlock deployments.

9. Security Considerations

The proposed mechanism is compatible only with BGP implementations that can process 32-bit ASNs in the AS_PATH. This limitation should not have a real effect on operations - such legacy BGP routers are rare and it's highly unlikely that they support integration with the RPKI.

ASPA issuers should be aware of the verification implication in issuing an ASPA - an ASPA implicitly invalidates all routes passed to upstream providers other than the provider ASs listed in the ASPA record. It is the Customer AS's duty to maintain a correct set of providers in ASPA record(s).

While it's not restricted, but it's highly recommended maintaining for selected Customer AS a single ASPA object that covers all its providers. Such policy should prevent race conditions during ASPA updates that might affect prefix propagation. The software that provides hosting for ASPA records SHOULD support enforcement of this rule. In the case of the transition process between different CA registries, the ASPA records SHOULD be kept identical in all registries.

While the ASPA is able to detect both mistakes and malicious activity for routes received from customers, RS-clients, or peers, it provides only detection of mistakes for routes that are received from upstream providers and RS(s).

Since an upstream provider becomes a trusted point, it will be able to send hijacked prefixes of its customers or send hijacked prefixes with malformed AS_PATHs back. While it may happen in theory, it's doesn't seem to be a real scenario: normally customer and provider have a signed agreement and such policy violation should have legal consequences or customer can just drop relation with such a provider and remove the corresponding ASPA record.

10. Acknowledgments

The authors wish to thank authors of [RFC6483] since its text was used as an example while writing this document. The also authors wish to thank Iljitsch van Beijnum for giving a hint about Downstream paths.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[Flexsealing] McDaniel, T., Smith, J., and M. Schuchard, "Flexsealing BGP Against Route Leaks: Peerlock Active Measurement and Analysis", November 2020, <https://arxiv.org/pdf/2006.06576.pdf>.

[I-D.ietf-grow-route-leak-detection-mitigation] Sriram, K. and A. Azimov, "Methods for Detection and Mitigation of BGP Route Leaks", draft-ietf-grow-route-leak-detection-mitigation-00 (work in progress), April 2019.

[I-D.ietf-idr-bgp-open-policy] Azimov, A., Bogomazov, E., Bush, R., Patel, K., and K. Sriram, "Route Leak Prevention using Roles in Update and Open messages", draft-ietf-idr-bgp-open-policy-05 (work in progress), February 2019.

[I-D.ietf-sidrops-aspa-profile] Azimov, A., Uskov, E., Bush, R., Patel, K., Snijders, J., and R. Housley, "A Profile for Autonomous System Provider Authorization", draft-ietf-sidrops-aspa-profile-00 (work in progress), May 2019.

[I-D.kumari-deprecate-as-set-confed-set] Kumari, W. and K. Sriram, "Deprecation of AS_SET and AS_CONFED_SET in BGP", draft-kumari-deprecate-as-set-confed-set-12 (work in progress), July 2018.

[I-D.white-sobgp-architecture] White, R., "Architecture and Deployment Considerations for Secure Origin BGP (soBGP)", draft-white-sobgp-architecture-02 (work in progress), June 2006.

[Peerlock]
Snijders, J., "Peerlock", June 2016,
<https://www.nanog.org/sites/default/files/
Snijders_Everyday_Practical_Bgp.pdf>.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <https://www.rfc-editor.org/info/rfc3779>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

[RFC6483] Huston, G. and G. Michaelson, "Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs)", RFC 6483, DOI 10.17487/RFC6483, February 2012, <https://www.rfc-editor.org/info/rfc6483>.

[RFC7908] Sriram, K., Montgomery, D., McPherson, D., Osterweil, E., and B. Dickson, "Problem Definition and Classification of BGP Route Leaks", RFC 7908, DOI 10.17487/RFC7908, June 2016, <https://www.rfc-editor.org/info/rfc7908>.

[RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/info/rfc8205>.

Authors' Addresses

Alexander Azimov
Yandex

Email: a.e.azimov@gmail.com

Eugene Bogomazov
Qrator Labs

Email: eb@qrator.net

Randy Bush
Internet Initiative Japan & Arrcus

Email: randy@psg.com

Keyur Patel
Arrcus, Inc.

Email: keyur@arrcus.com

Job Snijders
NTT Communications
Theodorus Majofskistraat 100
Amsterdam 1065 SZ
The Netherlands

Email: job@ntt.net

Network Working Group   T. Bruijnzeels
Internet-Draft   NLnet Labs
Updates: 6841, 8182 (if approved)   R. Bush
Intended status: Standards TrackInternet Initiative Japan & Arrcus, Inc.
Expires: August 26, 2021   G. Michaelson
                                                                        APNIC
                                                          February 22, 2021

Resource Public Key Infrastructure (RPKI) Repository Requirements draft-ietf-sidrops-prefer-rrdp-00

Abstract

This document formulates a plan of a phased transition to a state where RPKI repositories and Relying Party software performing RPKI Validation will use the RPKI Repository Delta Protocol (RRDP) [RFC8182] as the only mandatory to implement access protocol.

The first objective is to make RRDP the preferred access protocol, and require rsync as a fallback option only. This will greatly reduce the operational burden and concerns for RPKI repository operators.

In phase 0, today's deployment, RRDP is supported by most, but not all Repositories, and most but not all RP software.

In the proposed phase 1 RRDP will become mandatory to implement for Repositories, in addition to rsync. This phase can start as soon as this document is published.

Once the proposed updates are implemented by all Repositories phase 2 will start. In this phase RRDP will become mandatory to implement for all RP software, and rsync will be required as a fallback option only.

It should be noted that although this document currently includes descriptions and updates to RFCs for each of these phases, we may find that it will be beneficial to have one or more separate documents for these phases, so that it might be more clear to all when the updates to RFCs take effect.

Furthermore, this document currently includes an early discussion of a future objective, which would be to change the RPKI standards such that names in RPKI objects are no longer tightly coupled to rsync. By using transport independent names and validation, we will obtain the agility needed to phase out rsync altogether and/or introduce other future access protocols.

Status of This Memo

Copyright Notice

Table of Contents

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Motivation

The Resource Public Key Infrastructure (RPKI) [RFC6480] as originally defined uses rsync as its distribution protocol, as outlined in [RFC6481]. Later, the RPKI Repository Delta Protocol (RRDP) [RFC8182] was designed to provide an alternative. In order to facilitate incremental deployment RRDP has been deployed as an additional optional protocol, while rsync was still mandatory to implement.

While rsync has been very useful in the initial deployment of RPKI, a number of issues observed with it motivated the design of RRDP, e.g.:

- rsync is CPU and memory heavy on the server side, and easy to DoS
- rsync library support is lacking, complicating RP efficiency and error logging
- we cannot ensure that RPs get atomic sets of updated objects

RRDP was designed to leverage HTTPS CDN infrastructure to provide RPKI Repository content in a resilient way, while reducing the load on the Repository server. It supports that updates are published as

atomic deltas, which can help prevent most of the issues described in section 6 of [RFC6486].

For a longer discussion please see section 1 of [RFC8182].

In conclusion: we believe that while RRDP is not perfect, and we may indeed need future work to improve on it, it is an improvement over using rsync in the context of RPKI. Therefore, this document outlines a transition plan where RRDP becomes mandatory to implement, and the operational dependency on rsync is reduced to that of a fallback option.

3. Plan to prefer RRDP

Changing the RPKI infrastructure to rely on RRDP instead of rsync is a delicate operation. There is current deployment of Certification Authorities, Repository Servers and Relying Party software which relies on rsync, and which may not yet support RRDP.

Therefore we need to have a plan that ultimately updates the relevant RFCs, but which uses a phased approach combined with measurements to limit the operational impact of doing this to (almost) zero.

The general outline of the plan is as follows. We will describe each step in more detail below.

Phase	Description
0	RPKI repositories support rsync, and optionally RRDP
1	RPKI repositories support both rsync and RRDP
2	All RP software prefers RRDP

3.1. Phase 0 - RPKI repositories support rsync, and optionally RRDP

This is the situation at the time of writing this document. Relying Parties can prefer RRDP over rsync today, but they need to support rsync until all RPKI repositories support RRDP. Therefore all repositories should support RRDP at their earliest convenience.

3.1.1. Updates to RFC 8182

Repositories which support RRDP MUST ensure that RRDP resources are available to Relying Parties (section 3.3 of [RFC8182]). Furthermore, the RRDP repository MUST include all current repository objects. Because of this the choice of falling back to alternative

repository access mechanisms was left as a local policy choice of RP software.

However, following discussions on this subject it has become clear that there is a preference to instruct RP software to make use of all possible data sources. The main motivation being that because of RPKI object security using a secondary source of data can never lead to a worse outcome in terms of validation.

The following update is therefore applicable to section 3.4.5 "Considerations Regarding Operational Failures in RRDP" of [RFC8182]:

OLD: Relying Parties could attempt to use alternative repository access mechanisms, if they are available, according to the accessMethod element value(s) specified in the SIA of the associated certificate (see Section 4.8.8 of [RFC6487]).

NEW: Relying Parties MUST attempt to use alternative repository access mechanisms, if they are available, according to the accessMethod element value(s) specified in the SIA of the associated certificate (see Section 4.8.8 of [RFC6487]).

3.1.2. Updates to RFC 6481

As noted above section 3.3 of [RFC8182] already stipulates that RRDP files MUST be made available by repositories which support RRDP. In other words the RRDP service must be treated as a critical service wherever it is supported.

During this phase the updates are applied to section 3 of [RFC6481], to make this abundantly clear:

OLD:

- The publication repository SHOULD be hosted on a highly available service and high-capacity publication platform.

- The publication repository MUST be available using rsync [RFC5781] [RSYNC]. Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

NEW:

- The publication repository MAY be available using the RPKI Repository Delta Protocol [RFC8182]. If RPDP is provided, it SHOULD be hosted on a highly available platform.

- The publication repository MUST be available using rsync [RFC5781] [RSYNC]. The rsync server SHOULD be hosted on a highly available platform.

- Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

3.2. Phase 1 - RPKI repositories support both rsync and RRDP

During this phase we will make RRDP mandatory to support for Repository Servers, and measure whether the deployed Repository Servers have been upgraded to do so, in as far as they don't support RRDP already.

3.2.1. Updates to RFC 6481

During this phase the updates are applied to section 3 of [RFC6481].

OLD:

- The publication repository SHOULD be hosted on a highly available service and high-capacity publication platform.

- The publication repository MUST be available using rsync [RFC5781] [RSYNC]. Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

NEW:

- The publication repository MUST be available using the RPKI Repository Delta Protocol [RFC8182]. The RRDP server SHOULD be hosted on a highly available platform.

- The publication repository MUST be available using rsync [RFC5781] [RSYNC]. The rsync server SHOULD be hosted on a highly available platform.

- Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

3.2.2. Measurements

We can find out whether all RPKI repositories support RRDP by running (possibly) modified Relying Party software that keeps track of this.

When it is found that Repositories do not yet support RRDP, outreach should be done to them individually. Since the number of Repositories is fairly low, and it is in their interest to run RRDP because it addresses availability concerns, we have confidence that we will find these Repositories willing to make changes.

3.3. Phase 2 - All RP software prefers RRDP

Once all Repositories support RRDP we can proceed to make RRDP mandatory to implement for Relying Party software.

3.3.1. Updates to RFC 8182

From this phase onwards the updates are applied to section 3.4.1 of [RFC8182].

OLD: When a Relying Party performs RPKI validation and learns about a valid certificate with an SIA entry for the RRDP protocol, it SHOULD use this protocol as follows.

NEW: When a Relying Party performs RPKI validation and learns about a valid certificate with an SIA entry for the RRDP protocol, it MUST use this protocol with preference.

Relying Parties MUST NOT attempt to fetch objects using alternate access mechanisms, if object retrieval through this protocol is successful.

However, as stipulated in section 3.4.5, Relying Parties MUST attempt to use alternative repository access mechanisms, if object retrieval through this protocol is unsuccessful.

3.3.2. Rsync URIs as object identifiers

Rsync URIs are used in the RPKI to name objects and hierarchies, and they are as such very useful when doing RPKI object validation, as well as for error reporting on validation issues.

Note that RRDP includes rsync URIs in its structure. See section 3.5 of [RFC8182]. Theoretically, RRDP servers could include any rsync URI. However, Relying Party software knows which RRDP server to is expected to include the rsync URIs for RPKI objects issued under any

given CA certificate, because of the id-ad-rpkiNotify SIA extenion, see section 3.2 of [RFC8182].

Thus, objects retrieved through RRDP can be mapped easily to files and URIs, similar to as though rsync would have been used to retrieve them.

3.3.3. Measurements

Although the tools may support RRDP, users will still need to install updated versions of these tools in their infrastructure. Any Repository operator can measure this transition by observing access to their RRDP and rsync repositories respectively.

But even after new versions have been available, it is expected that there will be long, low volume, tail of users who did not upgrade and still depend on rsync.

It is hard to quantify here now, what would be an acceptable moment to conclude that it's safe to move to the next phase and make rsync optional. A parallel to the so-called DNS Flag Day comes to mind.

4. Future Objective: Remove the dependency on rsync

Note that, while we discuss this here, we would probably do well to separate this section into a separate follow-up document.

4.1. Phase 3 - RPKI repositories support RRDP, and optionally rsync

The end goal of this phase would be that there will be no operational dependencies on rsync for Repositories, although they MAY still choose to operate rsync at a best effort basis.

The most pragmatic way to deal with rsync URIs in the RPKI would be to continue to use them as namespaces, but no longer require that rsync is available. Much like how https based namespaces are used in XML.

4.1.1. Updates to RFC 6481

From this phase onwards these updates are applied to section 3 of [RFC6481] as it was updated during Phase 2 described above:

OLD:

- The publication repository MUST be available using the RPKI Repository Delta Protocol [RFC8182]. The RRDP server SHOULD be hosted on a highly available platform.

- The publication repository MUST be available using rsync [RFC5781] [RSYNC]. The rsync server SHOULD be hosted on a highly available platform.

- Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

NEW:

- The publication repository MUST be available using the RPKI Repository Delta Protocol [RFC8182]. The RRDP server SHOULD be hosted on a highly available platform.

- The publication repository MAY be available using rsync [RFC5781] [RSYNC].

- Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE certificate.

Note that this means that RP software is still required to try to fall back to rsync if RRDP is unavailable, but it may find that the rsync repository is not available.

4.2. Transport agnostic RPKI object names

We could develop a new naming scheme for RPKI objects. Perhaps based on Universal Resource Names ([RFC8141]). Doing so, would allow us to use names which are independent from retrieval mechanisms, and thus they could be less confusing in some regards, and provide more agility with regards to future changes in those mechanisms. However, this would require that many updates are made to existing RFCs. An incomplete list:

- RFC6487 New names would have be allowed in the SIA, or perhaps an X509 extension, could be used. But, the latter would have a direct impact on the deployability of updated CA certificates - RP software would reject these certificates if the extension is marked as critical by the CA and not understood by the RP.

- RFC6492 New names (in whatever form) would need to be included certificate sign requests sent to a parent CA. The parent CA will need to include a 'cert_url', indicating where an issued certificate is published, in a different format.

- RFC8181 The RPKI publication protocol is based rsync URIs, and it assumes that publishers have access to a specific directory in rsync space. This would need to be changed.

- RFC8183 This RFC defines the identity exchange between an RPKI CA and Publication Server. The server's response includes an 'sia_base', in the form of an rsync directory, under which a CA is supposed to name its objects.

- RFC8182 The RRDP protocol uses rsync URIs for compatibility with rsync as a retrieval method. This would need to be updated.

Obviously this needs more discussion.

The exercise would not be trivial. But, arguably doing this work will not become easier by postponing it, and once done would leave the RPKI better positioned to use alternative access methods in future as well.

5. Appendix - Implementation Status

Note that this section is included for tracking purposes during the discussion phase of this document and is not intended to be included in an RFC.

5.1. Current RRDP Support in Repository Software

The currently known support for RRDP for repositories is as follows:

Repository Implementation	Support for RRDP
afrinic	yes
apnic	yes
arin	yes
lacnic	ongoing
ripe ncc	yes
Dragon Research Labs	yes(1,2)
krill	yes(1)

(1) in use at various National Internet Registries, as well as other resource holders under RIRs. (2) not all organizations using this software have upgraded to using RRDP.

5.2. Current RRDP Support in Relying Party software

The currently known support for RRDP in Relying Party software is as follows:

Relying Party Implementation	RRDP	version	since
FORT	yes	1.2.0	02/2021
OctoRPKI	yes	1.0.0	02/2019
rcynic	yes	?	?
RIPE NCC RPKI Validator 2.x	yes	2.18	07/2015
RIPE NCC RPKI Validator 3.x	yes	3.0	03/2018
Routinator	yes	0.6.0	09/2019
rpki-client	ongoing	?	?
RPSTIR2	yes	2.0	04/2020

The authors kindly request Relying Party software implementers to let us know in which version of their tool support for RRDP was introduced, and when that version was released.

6. IANA Considerations

This document has no IANA actions.

7. Security Considerations

TBD

8. Acknowledgements

TBD

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, DOI 10.17487/RFC5781, February 2010, <https://www.rfc-editor.org/info/rfc5781>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <https://www.rfc-editor.org/info/rfc6481>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <https://www.rfc-editor.org/info/rfc6486>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8182] Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein, "The RPKI Repository Delta Protocol (RRDP)", RFC 8182, DOI 10.17487/RFC8182, July 2017, <https://www.rfc-editor.org/info/rfc8182>.

9.2. Informative References

[RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017, <https://www.rfc-editor.org/info/rfc8141>.

Authors' Addresses

Tim Bruijnzeels
NLnet Labs

Email: tim@nlnetlabs.nl
URI: https://www.nlnetlabs.nl/

Randy Bush
Internet Initiative Japan & Arrcus, Inc.

Email: randy@psg.com

George Michaelson
APNIC

Email: ggm@apnic.net
URI: http://www.apnic.net

Network Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: August 26, 2021

Y. Gilad
Hebrew University of Jerusalem
S. Goldberg
Boston University
K. Sriram
USA NIST
J. Snijders
Fastly
B. Maddison
Workonline Communications
February 22, 2021

The Use of Maxlength in the RPKI
draft-ietf-sidrops-rpkimaxlen-06

Abstract

This document recommends ways to reduce forged-origin hijack attack surface by prudently limiting the set of IP prefixes that are included in a Route Origin Authorization (ROA). One recommendation is to avoid using the maxLength attribute in ROAs except in some specific cases. The recommendations complement and extend those in RFC 7115. The document also discusses creation of ROAs for facilitating the use of Distributed Denial of Service (DDoS) mitigation services. Considerations related to ROAs and origin validation in the context of destination-based Remote Triggered Black Hole (RTBH) filtering are also highlighted.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 26, 2021.

Table of Contents

1. Introduction

The RPKI [RFC6480] uses Route Origin Authorizations (ROAs) to create a cryptographically verifiable mapping from an IP prefix to a set of autonomous systems (ASes) that are authorized to originate that prefix. Each ROA contains a set of IP prefixes, and an AS number of an AS authorized to originate all the IP prefixes in the set [RFC6482]. The ROA is cryptographically signed by the party that holds a certificate for the set of IP prefixes.

The ROA format also supports a maxLength attribute. According to [RFC6482], "When present, the maxLength specifies the maximum length of the IP address prefix that the AS is authorized to advertise." Thus, rather than requiring the ROA to list each prefix the AS is authorized to originate, the maxLength attribute provides a shorthand that authorizes an AS to originate a set of IP prefixes.

However, measurements of current RPKI deployments have found that use of the maxLength in ROAs tends to lead to security problems. Specifically, measurements have shown that 84% of the prefixes specified in ROAs that use the maxLength attribute, are vulnerable to a forged-origin subprefix hijack [HARMFUL]. The forged-origin prefix or subprefix hijack involves inserting the legitimate AS as specified in the ROA as the origin AS in the AS_PATH, and can be launched against any IP prefix/subprefix that has a ROA. Consider a prefix/subprefix that has a ROA but is unused, i.e., not announced in BGP by a legitimate AS. A forged origin hijack involving such a prefix/subprefix can propagate widely throughout the Internet. On the other hand, if the prefix/subprefix were announced by the legitimate AS, then the propagation of the forged-origin hijack is somewhat limited because of its increased AS_PATH length relative to the legitimate announcement. Of course, forged-origin hijacks are harmful in both cases but the extent of harm is greater for unannounced prefixes.

For this reason, this document recommends that, whenever possible, operators SHOULD use "minimal ROAs" that authorize only those IP prefixes that are actually originated in BGP, and no other prefixes. Further, it recommends ways to reduce forged-origin attack surface by prudently limiting the address space that is included in Route Origin Authorizations (ROAs). One recommendation is to avoid using the maxLength attribute in ROAs except in some specific cases. The recommendations complement and extend those in [RFC7115]. The document also discusses creation of ROAs for facilitating the use of Distributed Denial of Service (DDoS) mitigation services. Considerations related to ROAs and origin validation in the context of destination-based Remote Triggered Black Hole (RTBH) filtering are also highlighted.

One ideal place to implement the ROA related recommendations is in the user interfaces for configuring ROAs. Thus, this document further recommends that designers and/or providers of such user interfaces SHOULD provide warnings to draw the user’s attention to the risks of using the maxLength attribute.

Best current practices described in this document require no changes to the RPKI specification and will not increase the number of signed ROAs in the RPKI, because ROAs already support lists of IP prefixes [RFC6482].

1.1. Requirements

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 [RFC2119].

1.2. Documentation Prefixes

The documentation prefixes recommended in [RFC5737] are insufficient for use as example prefixes in this document. Therefore, this document uses [RFC1918] address space for constructing example prefixes.

2. Suggested Reading

It is assumed that the reader understands BGP [RFC4271], RPKI [RFC6480], Route Origin Authorizations (ROAs) [RFC6482], RPKI-based Prefix Validation [RFC6811], and BGPsec [RFC8205].

3. Forged-Origin Subprefix Hijack

A detailed description and discussion of forged-origin subprefix hijacks are presented here, especially considering the case when the subprefix is not announced in BGP. The forged-origin subprefix hijack is relevant to a scenario in which:

(1) the RPKI [RFC6480] is deployed, and

(2) routers use RPKI origin validation to drop invalid routes [RFC6811], but

(3) BGPsec [RFC8205] (or any similar method to validate the truthfulness of the BGP AS_PATH attribute) is not deployed.

Note that this set of assumptions accurately describes a substantial, and growing, number of large Internet networks at the time writing.

The forged-origin subprefix hijack [RFC7115] [GCHSS] is described here using a running example.

Consider the IP prefix 192.168.0.0/16 which is allocated to an organization that also operates AS 64496. In BGP, AS 64496 originates the IP prefix 192.168.0.0/16 as well as its subprefix 192.168.225.0/24. Therefore, the RPKI should contain a ROA authorizing AS 64496 to originate these two IP prefixes.

Suppose, however, the organization issues and publishes a ROA including a maxLength value of 24:

ROA:(192.168.0.0/16-24, AS 64496)

We refer to the above as a "loose ROA" since it authorizes AS 64496 to originate any subprefix of 192.168.0.0/16 up to and including

length /24, rather than only those prefixes that are intended to be announced in BGP.

Because AS 64496 only originates two prefixes in BGP: 192.168.0.0/16 and 192.168.225.0/24, all other prefixes authorized by the "loose ROA" (for instance, 192.168.0.0/24), are vulnerable to the following forged-origin subprefix hijack [RFC7115] [GCHSS]:

> The hijacker AS 64511 sends a BGP announcement "192.168.0.0/24: AS 64511, AS 64496", falsely claiming that AS 64511 is a neighbor of AS 64496 and falsely claiming that AS 64496 originates the IP prefix 192.168.0.0/24. In fact, the IP prefix 192.168.0.0/24 is not originated by AS 64496.
>
> The hijacker's BGP announcement is valid according to the RPKI, since the ROA (192.168.0.0/16-24, AS 64496) authorizes AS 64496 to originate BGP routes for 192.168.0.0/24.
>
> Because AS 64496 does not actually originate a route for 192.168.0.0/24, the hijacker's route is the *only* route to the 192.168.0.0/24. Longest-prefix-match routing ensures that the hijacker's route to the subprefix 192.168.0.0/24 is always preferred over the legitimate route to 192.168.0.0/16 originated by AS 64496.

Thus, the hijacker's route propagates through the Internet, the traffic destined for IP addresses in 192.168.0.0/24 will be delivered to the hijacker.

The forged-origin *subprefix* hijack would have failed if a "minimal ROA" described below was used instead of the "loose ROA". In this example, a "minimal ROA" would be:

> ROA:(192.168.0.0/16, 192.168.225.0/24, AS 64496)

This ROA is "minimal" because it includes only those IP prefixes that AS 64496 originates in BGP, but no other IP prefixes [RFC6907].

The "minimal ROA" renders AS 64511's BGP announcement invalid, because:

> (1) this ROA "covers" the attacker's announcement (since 192.168.0.0/24 is a subprefix of 192.168.0.0/16), and
>
> (2) there is no ROA "matching" the attacker's announcement (there is no ROA for AS 64511 and IP prefix 192.168.0.0/24) [RFC6811].

If routers ignore invalid BGP announcements, the minimal ROA above ensures that the subprefix hijack will fail.

Thus, if a "minimal ROA" had been used, the attacker would be forced to launch a forged-origin *prefix* hijack in order to attract traffic, as follows:

> The hijacker AS 64511 sends a BGP announcement "192.168.0.0/16: AS 64511, AS 64496", falsely claiming that AS 64511 is a neighbor of AS 64496.

This forged-origin *prefix* hijack is significantly less damaging than the forged-origin *subprefix* hijack:

> AS 64496 legitimately originates 192.168.0.0/16 in BGP, so the hijacker AS 64511 is not presenting the *only* route to 192.168.0.0/16.
>
> Moreover, the path originated by AS 64511 is one hop longer than the path originated by the legitimate origin AS 64496.

As discussed in [LSG16], this means that the hijacker will attract less traffic than he would have in the forged-origin *subprefix* hijack, where the hijacker presents the *only* route to the hijacked subprefix.

In summary, a forged-origin subprefix hijack has the same impact as a regular subprefix hijack, despite the increased AS_PATH length of the illegitimate route. A forged-origin *subprefix* hijack is also more damaging than forged-origin *prefix* hijack.

4. Measurements of Today's RPKI

Network measurements have shown that 12% of the IP prefixes authorized in ROAs have a maxLength longer than their prefix length. Of these, the vast majority (84%) are non-minimal, as they include subprefixes that are not announced in BGP by the legitimate AS, and are thus vulnerable to forged origin subprefix hijacks. See [GSG17] for details.

These measurements suggest that operators commonly misconfigure the maxLength attribute, and unwittingly open themselves up to forged-origin subprefix hijacks. That is, they are exposing a much larger attack surface for forged-origin hijacks than necessary.

5. Recommendations about Minimal ROAs and maxLength

Operators SHOULD use "minimal ROAs" whenever possible. A minimal ROA contains only those IP prefixes that are actually originated by an AS in BGP, and no other IP prefixes. (See Section 3 for an example.)

In general, except in some special cases, operators SHOULD avoid using the maxLength attribute in their ROAs, since its inclusion will usually make the ROA non-minimal.

One such exception may be when all more specific prefixes permitted by the maxLength are actually announced by the AS in the ROA. Another exception is where: (a) the maxLength is substantially larger compared to the specified prefix length in the ROA, and (b) a large number of more specific prefixes in that range are announced by the AS in the ROA. This case should occur rarely in practice (if at all). Operator discretion is necessary in this case.

This practice requires no changes to the RPKI specification and need not increase the number of signed ROAs in the RPKI, because ROAs already support lists of IP prefixes [RFC6482]. See also [GSG17] for further discussion of why this practice will have minimal impact on the performance of the RPKI ecosystem.

5.1. Facilitating Ad-hoc Routing Changes and DDoS Mitigation

Operational requirements may require that a route for an IP prefix be originated on an ad-hoc basis, with little or no prior warning. An example of such a situation arises where an operator wishes to make use of DDoS mitigation services that use BGP to redirect traffic via a "scrubbing center".

In order to ensure that such ad-hoc routing changes are effective, there should exist a ROA validating the new route. However a difficulty arises due to the fact that newly created objects in the RPKI are made visible to relying parties considerabley more slowly than routing updates in BGP.

Ideally, it would not be necessary to pre-create the ROA which validates the ad-hoc route, and instead create it "on-the-fly" as required. However, this is practical only if the latency imposed by the propagation of RPKI data is guaranteed to be within acceptable limits in the circumstances. For time-critical interventions such as responding to a DDoS attack, this is unlikely to be the case.

Thus, the ROA in question will usually need to be created well in advance of the routing intervention, but such a ROA will be non-

minimal, since it includes an IP prefix that is sometimes (but not always) originated in BGP.

In this case, the ROA SHOULD include:

(1) the set of IP prefixes that are always originated in BGP, and

(2) the set IP prefixes that are sometimes, but not always, originated in BGP.

The ROA SHOULD NOT include any IP prefixes that the operator knows will not be originated in BGP. Whenever possible, the ROA SHOULD also avoid the use of the maxLength attribute unless doing so has no impact on the set of included prefixes.

The running example is now extended to illustrate one situation where it is not possible to issue a minimal ROA.

Consider the following scenario prior to deployment of RPKI. Suppose AS 64496 announced 192.168.0.0/16 and has a contract with a Distributed Denial of Service (DDoS) mitigation service provider that holds AS 64500. Further, assume that the DDoS mitigation service contract applies to all IP addresses covered by 192.168.0.0/22. When a DDoS attack is detected and reported by AS 64496, AS 64500 immediately originates 192.168.0.0/22, thus attracting all the DDoS traffic to itself. The traffic is scrubbed at AS 64500 and then sent back to AS 64496 over a backhaul data link. Notice that, during a DDoS attack, the DDoS mitigation service provider AS 64500 originates a /22 prefix that is longer than AS 64496's /16 prefix, and so all the traffic (destined to addresses in 192.168.0.0/22) that normally goes to AS 64496 goes to AS 64500 instead. In some deployments, the origination of the /22 route is performed by AS 64496 and announced only to AS 64500, which then announces transit for that prefix. This variation does not change the properties considered here.

First, suppose the RPKI only had the minimal ROA for AS 64496, as described in Section 3. But if there is no ROA authorizing AS 64500 to announce the /22 prefix, then the DDoS mitigation (and traffic scrubbing) scheme would not work. That is, if AS 64500 originates the /22 prefix in BGP during DDoS attacks, the announcement would be invalid [RFC6811].

Therefore, the RPKI should have two ROAs: one for AS 64496 and one for AS 64500.

ROA:(192.168.0.0/16, 192.168.225.0/24, AS 64496)

ROA:(192.168.0.0/22, AS 64500)

Neither ROA uses the maxLength attribute. But the second ROA is not "minimal" because it contains a /22 prefix that is not originated by anyone in BGP during normal operations. The /22 prefix is only originated by AS 64500 as part of its DDoS mitigation service during a DDoS attack.

Notice, however, that this scheme does not come without risks. Namely, all IP addresses in 192.168.0.0/22 are vulnerable to a forged-origin subprefix hijack during normal operations, when the /22 prefix is not originated. (The hijacker AS 64511 would send the BGP announcement "192.168.0.0/22: AS 64511, AS 64500", falsely claiming that AS 64511 is a neighbor of AS 64500 and falsely claiming that AS 64500 originates 192.168.0.0/22.)

In some situations, the DDoS mitigation service at AS 64500 might want to limit the amount of DDoS traffic that it attracts and scrubs. Suppose that a DDoS attack only targets IP addresses in 192.168.0.0/24. Then, the DDoS mitigation service at AS 64500 only wants to attract the traffic designated for the /24 prefix that is under attack, but not the entire /22 prefix. To allow for this, the RPKI should have two ROAs: one for AS 64496 and one for AS 64500.

ROA:(192.168.0.0/16, 192.168.225.0/24, AS 64496)

ROA:(192.168.0.0/22-24, AS 64500)

The second ROA uses the maxLength attribute because it is designed to explicitly enable AS 64500 to originate *any* /24 subprefix of 192.168.0.0/22.

As before, the second ROA is not "minimal" because it contains prefixes that are not originated by anyone in BGP during normal operations. As before, all IP addresses in 192.168.0.0/22 are vulnerable to a forged-origin subprefix hijack during normal operations, when the /22 prefix is not originated.

The use of maxLength in this second ROA also comes with an additional risk. While it permits the DDoS mitigation service at AS 64500 to originate prefix 192.168.0.0/24 during a DDoS attack in that space, it also makes the *other* /24 prefixes covered by the /22 prefix (i.e., 192.168.1.0/24, 192.168.2.0/24, 192.168.3.0/24) vulnerable to a forged-origin subprefix attacks.

6. ROAs and Origin Validation for RTBH Filtering Scenario

Considerations related to ROAs and origin validation [RFC6811] for the case of destination-based Remote Triggered Black Hole (RTBH) filtering are addressed here. In RTBH filtering, highly specific

prefixes (greater than /24 in IPv4 and greater than /48 in IPv6; possibly even /32 (IPv4) and /128 (IPv6)) are announced in BGP. These announcements are tagged with a BLACKHOLE Community [RFC7999]. It is obviously not desirable to use large maxlength or include any such highly specific prefixes in the ROAs to accommodate destination-based RTBH filtering, for the reasons set out above.

As a result, RPKI based route origin validation [RFC6811] is a poor fit for the validation of RTBH routes. Specification of new procedures to address this use case through the use of the RPKI is outside the scope of this document.

Therefore:

- Operators SHOULD NOT create non-minimal ROAs (either by creating additional ROAs, or through the use of maxLength) for the purpose of advertising RTBH routes; and

- Operators providing a means for operators of neighboring autonomous systems to advertise RTBH routes via BGP MUST NOT make the creation of non-minimal ROAs a pre-requisite for its use.

7. Acknowledgments

The authors would like to thank the following people for their review and contributions to this document: Omar Sagga (Boston University) and Aris Lambrianidis (AMS-IX). Thanks are also due to Matthias Waehlisch (Free University of Berlin) for comments and suggestions.

8. References

8.1. Normative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <https://www.rfc-editor.org/info/rfc1918>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <https://www.rfc-editor.org/info/rfc6482>.

[RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <https://www.rfc-editor.org/info/rfc6811>.

8.2. Informative References

[GCHSS] Gilad, Y., Cohen, A., Herzberg, A., Schapira, M., and H. Shulman, "Are We There Yet? On RPKI's Deployment and Security", in NDSS 2017, February 2017, <https://eprint.iacr.org/2016/1010.pdf>.

[GSG17] Gilad, Y., Sagga, O., and S. Goldberg, "Maxlength Considered Harmful to the RPKI", in ACM CoNEXT 2017, December 2017, <https://eprint.iacr.org/2016/1015.pdf>.

[HARMFUL] Gilad, Y., Sagga, O., and S. Goldberg, "MaxLength Considered Harmful to the RPKI", 2017, <https://eprint.iacr.org/2016/1015.pdf>.

[LSG16] Lychev, R., Shapira, M., and S. Goldberg, "Rethinking Security for Internet Routing", in Communications of the ACM, October 2016, <http://cacm.acm.org/magazines/2016/10/207763-rethinking-security-for-internet-routing/>.

[NIST-800-189] Sriram, K. and D. Montgomery, "Resilient Interdomain Traffic Exchange: BGP Security and DDoS Mitigation", NIST Special Publication, NIST SP 800-189, December 2019, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-189.pdf>.

[RFC6907] Manderson, T., Sriram, K., and R. White, "Use Cases and Interpretations of Resource Public Key Infrastructure (RPKI) Objects for Issuers and Relying Parties", RFC 6907, DOI 10.17487/RFC6907, March 2013, <https://www.rfc-editor.org/info/rfc6907>.

[RFC7115] Bush, R., "Origin Validation Operation Based on the Resource Public Key Infrastructure (RPKI)", BCP 185, RFC 7115, DOI 10.17487/RFC7115, January 2014, <https://www.rfc-editor.org/info/rfc7115>.

[RFC7999] King, T., Dietzel, C., Snijders, J., Doering, G., and G. Hankins, "BLACKHOLE Community", RFC 7999, DOI 10.17487/RFC7999, October 2016, <https://www.rfc-editor.org/info/rfc7999>.

[RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/info/rfc8205>.

Authors' Addresses

Yossi Gilad
Hebrew University of Jerusalem
Rothburg Family Buildings, Edmond J. Safra Campus
Jerusalem 9190416
Israel

EMail: yossigi@cs.huji.ac.il

Sharon Goldberg
Boston University
111 Cummington St, MCS135
Boston, MA 02215
USA

EMail: goldbe@cs.bu.edu

Kotikalapudi Sriram
USA National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899
USA

EMail: kotikalapudi.sriram@nist.gov

Job Snijders
Fastly
Amsterdam
Netherlands

EMail: job@fastly.com

Ben Maddison
Workonline Communications
114 West St
Johannesburg 2196
South Africa

EMail: benm@workonline.africa

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 24, 2021

J. Snijders
Fastly

February 20, 2021

RPKI Signed Checklists
draft-spaghetti-sidrops-rpki-rsc-03

Abstract

This document defines a Cryptographic Message Syntax (CMS) profile for a general purpose listing of checksums (a 'checklist'), for use with the Resource Public Key Infrastructure (RPKI). The objective is to allow an attestation, in the form of a listing of one or more checksums of arbitrary digital objects (files), to be signed "with resources", and for validation to provide a means to confirm a specific Internet Resource Holder produced the signed checklist. The profile is intended to provide for the signing of a checksum listing with an arbitrary set of Internet Number Resources.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

Table of Contents

1. Introduction

This document defines a Cryptographic Message Syntax (CMS) [RFC5652] profile for a general purpose listing of checksums (a 'checklist'), for use with the Resource Public Key Infrastructure (RPKI) [RFC6480]. The objective is to allow an attestation, in the form of a listing of one or more checksums of arbitrary files, to be signed "with resources", and for validation to provide a means to confirm a given Internet Resource Holder produced the RPKI Signed Checklist (RSC).

The profile is intended to provide for the signing of a checksum listing with an arbitrary set of Internet Number Resources.

RSC files are expected to facilitate Bring Your Own IP (BYOIP) authentication, inter-domain interconnection provisioning, and resource holdership verification processes.

The RSC concept borrows heavily from RTA [I-D.ietf-sidrops-rpki-rta], Manifests [RFC6486], and OpenBSD's [signify] utility. The main difference between RSC and RTA is that an RTA enables multiple signers to attest a single anonymous digital object through a checksum of its content, while an RSC allows a single signer to attest the checksums of multiple named digital objects. This difference is expected to represent a simplification for implementers and operators.

2. RSC Profile and Distribution

RSC follows the Signed Object Template for the RPKI [RFC6488] with one exception. Because RSCs MUST NOT be distributed through the global RPKI repository system, the Subject Information Access (SIA) extension is omitted from the RSC's X.509 EE certificate.

What constitutes suitable transport for RSC files is deliberately unspecified. It might be a USB stick, a web interface secured with conventional HTTPS, PGP-signed email, a T-shirt printed with a QR code, or a carrier pigeon.

3. The RSC ContentType

The ContentType for an RSC is defined as rpkiSignedChecklist, and has the numerical value of 1.2.840.113549.1.9.16.1.TBD.

This OID MUST appear both within the eContentType in the encapContentInfo object as well as the ContentType signed attribute in the signerInfo object (see [RFC6488]).

4. The RSC eContent

The content of an RSC indicates that an a checklist for arbitrary named digital objects has been signed "with resources". An RSC is formally defined as:

```
RpkiSignedChecklist-2021
  { iso(1) member-body(2) us(840) rsadsi(113549)
    pkcs(1) pkcs9(9) smime(16) mod(0) TBD }

DEFINITIONS EXPLICIT TAGS ::=
```

```
BEGIN

IMPORTS
  CONTENT-TYPE, Digest, DigestAlgorithmIdentifier
  FROM CryptographicMessageSyntax-2009 -- in [RFC5911]
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
      pkcs-9(9) smime(16) modules(0) id-mod-cms-2004-02(41) }

  ASIdOrRange, IPAddressFamily
  FROM IPAddrAndASCertExtn -- in [RFC3779]
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) mod(0)
      id-mod-ip-addr-and-as-ident(30) } ;

ct-rpkiSignedChecklist CONTENT-TYPE ::=
    { TYPE RpkiSignedChecklist IDENTIFIED BY
      id-ct-signedChecklist }

id-ct-signedChecklist OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
      pkcs-9(9) id-smime(16) id-ct(1) TBD }

RpkiSignedChecklist ::= SEQUENCE {
  version  [0]          INTEGER DEFAULT 0,
  resources             ResourceBlock,
  digestAlgorithm       DigestAlgorithmIdentifier,
  checkList             SEQUENCE SIZE (1..MAX) OF FileAndHash }

FileAndHash ::= SEQUENCE {
  file            IA5String OPTIONAL,
  hash            Digest }

ResourceBlock ::= SEQUENCE {
    asID         [0]       AsList OPTIONAL,
    ipAddrBlocks [1]       IPList OPTIONAL }
    -- at least one of asID or ipAddrBlocks MUST be present
    ( WITH COMPONENTS { ..., asID PRESENT} |
      WITH COMPONENTS { ..., ipAddrBlocks PRESENT } )

AsList ::= SEQUENCE (SIZE(1..MAX)) OF ASIdOrRange

IPList ::= SEQUENCE (SIZE(1..MAX)) OF IPAddressFamily

END
```

4.1. version

The version number of the RpkiSignedChecklist MUST be 0.

4.2. resources

The resources contained here are the resources used to mark the attestation, and MUST match the set of resources listed by the EE certificate carried in the CMS certificates field.

4.3. digestAlgorithm

The digest algorithm used to create the message digest of the attested digital object. This algorithm MUST be a hashing algorithm defined in [RFC7935].

4.4. checkList

This field is a sequence of FileAndHash objects. There is one FileAndHash entry for each arbitrary object referenced from the RSC. Each FileAndHash is an ordered pair consisting an optional name of the file containing the object, and the message digest of the digital object.

5. Operational Considerations

When working with objects of a plain-text nature (ASCII, UTF-8, HTML, Javascript, XML, etc) it is RECOMMENDED to distribute such objects in a lossless compressed form, and sign the compressed form. Wrapping plain-text objects in a compression envelope can help make those appear as a single octet string to any intermediate systems, which hopefully discourages in-transit modification of the file contents. The use of lossless compression can help avoid checksum verification errors.

6. RSC Validation

To validate an RSC the relying party MUST perform all the validation checks specified in [RFC6488] as well as the following additional RSC-specific validation steps.

- The message digest of each referenced digital object, using the digest algorithm specified in the the digestAlgorithm field, MUST be calculated and MUST match the value given in the messageDigest field of the associated FileAndHash.

- If a filename is present, the filename MUST NOT contain a '/' (slash) or '\' (backslash) character.

7. Security Considerations

Relying parties are hereby warned that the data in a RPKI Signed Checklist is self-asserted. These data have not been verified by the CA that issued the CA certificate to the entity that issued the EE certificate used to validate the Signed Checklist.

8. Implementation status - RFC EDITOR: REMOVE BEFORE PUBLICATION

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 7942. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

- A signer and validator implementation [rpki-rsc-demo] based on perl and OpenSSL was provided by Tom Harrison from APNIC.

- A validator implementation based on OpenBSD's rpki-client is expected to be published after IANA Early Allocation of the OIDs.

9. IANA Considerations

9.1. OID

The IANA has registered the OID for the RPKI Signed Checklist in the registry created by [RFC6488] as follows:

Name	OID	Specification
Checklists	1.2.840.113549.1.9.16.1.TBD	[RFC-TBD]

9.2. File Extension

The IANA has added an item for the signed Checklist file extension to the "RPKI Repository Name Scheme" created by [RFC6481] as follows:

Filename Extension	RPKI Object	Reference
.sig	Signed Checklist	[RFC-TBD]

9.3. Media Type

The IANA has registered the media type application/rpki-checklist as follows:

```
Type name: application
Subtype name: rpki-checklist
Required parameters: None
Optional parameters: None
Encoding considerations: binary
Security considerations: Carries an RPKI Signed Checklist
                         [RFC-TBD].
Interoperability considerations: None
Published specification: This document.
Applications that use this media type: RPKI operators.
Additional information:
  Content: This media type is a signed object, as defined
      in [RFC6488], which contains a payload of a list of
      checksums as defined above in this document.
  Magic number(s): None
  File extension(s): .sig
  Macintosh file type code(s):
Person & email address to contact for further information:
  Job Snijders <job@fastly.com>
Intended usage: COMMON
Restrictions on usage: None
Author: Job Snijders <job@fastly.com>
Change controller: Job Snijders <job@fastly.com>
```

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <https://www.rfc-editor.org/info/rfc6481>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <https://www.rfc-editor.org/info/rfc6486>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <https://www.rfc-editor.org/info/rfc6488>.

[RFC7935] Huston, G. and G. Michaelson, Ed., "The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure", RFC 7935, DOI 10.17487/RFC7935, August 2016, <https://www.rfc-editor.org/info/rfc7935>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[I-D.ietf-sidrops-rpki-rta]
Michaelson, G., Huston, G., Harrison, T., Bruijnzeels, T., and M. Hoffmann, "A profile for Resource Tagged Attestations (RTAs)", draft-ietf-sidrops-rpki-rta-00 (work in progress), January 2021.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

[rpki-rsc-demo]
Harrison, T., "A proof-of-concept for constructing and validating RPKI Signed Checklists (RSCs).", February 2021, <https://github.com/APNIC-net/rpki-rsc-demo>.

[signify] Unangst, T. and M. Espie, "signify - cryptographically sign and verify files", May 2014, <https://man.openbsd.org/signify>.

Appendix A. Acknowledgements

The author wishes to thank George Michaelson, Tom Harrison, Geoff Huston, Randy Bush, Stephen Kent, Matt Lepinski, Rob Austein, Ted Unangst, and Marc Espie for prior art. The author thanks Russ Housley for reviewing the ASN.1 notation and providing suggestions. The author would like to thank Nimrod Levy, Tom Harrison, Ben Maddison, and Tim Bruijnzeels for document review and suggestions.

Author's Address

Job Snijders
Fastly
Amsterdam
Netherlands

Email: job@fastly.com

Network Working Group
Internet-Draft
Obsoletes: 8360 (if approved)
Updates: 6482,6487 (if approved)
Intended status: Standards Track
Expires: August 26, 2021

J. Snijders
Fastly
B. Maddison
Workonline

February 22, 2021

RPKI Validation Re-reconsidered
draft-spaghetti-sidrops-rpki-validation-update-00

Abstract

This document describes an improved validation procedure for Resource Public Key Infrastructure (RPKI) signed objects. This document updates RFC 6482. This document updates RFC 6487. This document obsoletes RFC 8360.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 26, 2021.

Table of Contents

1. Introduction

[RFC8360] describes an improved validation algorithm for signed objects published in the RPKI. The improved validation algorithm would help in situations such as described in this [Report]. However, operational experience has shown the described procedure for deploying updates to the validation algorithm, as described in [RFC6487] Section 9, is impractical. This document deprecates the original [RFC6487] section 7 algorithm in favour of the [RFC8360] algorithm, and obsoletes [RFC8360] because a migration via those codepoints is infeasible. This document also deprecates the procedure set out in [RFC6487] section 9 for future changes to the validation algorithm.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Deprecation of RFC 8360

[RFC8360] defines several alternative OIDs for use in Resource Certificates [RFC6487]:

- id-cp-ipAddr-asNumber-v2 - Section 4.2.1 [RFC8360]
- id-pe-ipAddrBlocks-v2 - Sections 4.2.2.1 and 4.2.2.2 [RFC8360]
- id-pe-autonomousSysIds-v2 - Sections 4.2.2.3 and 4.2.2.4 [RFC8360]

The stated purpose of the above OIDs is rendered obsolete by the updated specifications contained in this document.

Therefore:

- Issuing CAs MUST NOT include the above OIDs in newly issued Resource Certificates; and
- Relying parties encountering the above OIDs in Resource Certificates MUST proceed according to the updated procedures described below.

3. Updates to RFC 6482

This section updates Section 4 [RFC6482]. The following text:

> The IP address delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the set of IP addresses specified by the EE certificate's IP address delegation extension.

Is replaced with:

> Either the IP Address Delegation extension described in [RFC3779] or the alternative IP Address Delegation extension described in [RFC8360] (but not both) is present in the end entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the VRS-IP set that is specified as an outcome of EE certificate validation described in Section 7.2 (as updated by this document) [RFC6487].

Note that this ensures that ROAs can be valid only if all IP address prefixes in the ROA are encompassed by the VRS-IP of all certificates along the path to the trust anchor used to verify it.

Operators MAY issue separate ROAs for each IP address prefix, so that the loss of one or more IP address prefixes from the VRS-IP of any certificate along the path to the trust anchor would not invalidate authorizations for other IP address prefixes.

4. Updates to RFC 6487

This section updates [RFC6487] to specify an improved behavior of a Relying Party implementation.

4.1. Updates to Section 7.2

The following section replaces Section 7.2 [RFC6487] (Resource Certification Path Validation) in its entirety.

Validation of signed resource data using a target resource certificate consists of verying that the digital signature of the signed resource data is valid, using the public key of the target resource certificate, and also validating the resource certificate in the context of the RPKI, using the path validation process.

There are two inputs to the validation algortihm:

1. A trust anchor

2. A certificate to be validated

The algorithm is initialized with two new variables for use in the RPKI: Verified Resource Set-IP (VRS-IP) and Verified Resource Set-AS (VRS-AS). These sets are used to track the set of INRs (IP address space and AS numbers) that are considered valid for each CA certificate. The VRS-IP and VRS-AS sets are initially set to the IP Address Delegation and AS Identifier Delegation values, respectively, from the trust anchor used to perform validation.

This path validation algorithm verifies, among other things, that a prospective certification path (a sequence of n certificates) satisfies the following conditions:

a. for all 'x' in {1, ..., n-1}, the subject of certificate 'x' is the issuer of certificate ('x' + 1);

b. certificate '1' is issued by a trust anchor;

c. certificate 'n' is the certificate to be validated; and

d. for all 'x' in {1, ..., n}, certificate 'x' is valid.

Certificate validation requires verifying that all of the following conditions hold, in addition to the certification path validation criteria specified in Section 6 of [RFC5280].

1. The signature of certificate x (x>1) is verified using the public key of the issuer's certificate (x-1), using the signature algorithm specified for that public key (in certificate x-1).

2. The current time lies within the interval defined by the NotBefore and NotAfter values in the Validity field of certificate x.

3. The Version, Issuer, and Subject fields of certificate x satisfy the constraints established in Sections 4.1 to 4.7 of RFC 6487.

4. If certificate x uses the Certificate Policy defined in Section 4.8.9 of [RFC6487], then the certificate MUST contain all extensions defined in Section 4.8 of [RFC6487] that must be present. The value(s) for each of these extensions MUST satisfy the constraints established for each extension in the respective sections. Any extension not thus identified MUST NOT appear in certificate x.

5. If certificate x uses the Certificate Policy defined in Section 4.2.4.1 [RFC8360], then all extensions defined in Section 4.8 of [RFC6487], except Sections 4.8.9, 4.8.10, and 4.8.11 MUST be present. The certificate MUST contain an extension as defined in Sections 4.2.4.2 or 4.2.4.3 [RFC8360], or both. The value(s) for each of these extensions MUST satisfy the constraints established for each extension in the respective sections. Any extension not thus identified MUST NOT appear in certificate x.

6. Certificate x MUST NOT have been revoked, i.e., it MUST NOT appear on a Certificate Revocation List (CRL) issued by the CA represented by certificate x-1.

7. Compute the VRS-IP and VRS-AS set values as indicated below:

   If the IP Address Delegation extension is present in certificate x and x=1, set the VRS-IP to the resources found in this extension.

   If the IP Address Delegation extension is present in certificate x and x>1, set the VRS-IP to the intersection of the resources between this extension and the value of the VRS-IP computed for certificate x-1.

   If the IP Address Delegation extension is absent in certificate x, set the VRS-IP to NULL.

If the IP Address Delegation extension is present in certificate x and x=1, set the VRS-IP to the resources found in this extension.

If the AS Identifier Delegation extension is present in certificate x and x>1, set the VRS-AS to the intersection of the resources between this extension and the value of the VRS-AS computed for certificate x-1.

If the AS Identifier Delegation extension is absent in certificate x, set the VRS-AS to NULL.

8. If there is any difference in resources in the VRS-IP and the IP Address Delegation extension on certificate x, or the VRS-AS and the AS Identifier Delegation extension on certificate x, then a warning listing the overclaiming resources for certificate x SHOULD be issued.

These rules allow a CA certificate to contain resources that are not present in (all of) the certificates along the path from the trust anchor to the CA certificate. If none of the resources in the CA certificate are present in all certificates along the path, no subordinate certificates could be valid. However, the certificate is not immediately rejected as this may be a transient condition. Not immediately rejecting the certificate does not result in a security problem because the associated VRS sets accurately reflect the resources validly associated with the certificate in question.

4.2. Updates to Section 9

Section 9 "Operational Considerations for Profile Agility" is removed.

5. Implementation status - RFC EDITOR: REMOVE BEFORE PUBLICATION

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC7942. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

As of today these changesets have been produced for commonly used Relying Party implementations:

NLnet Labs Routinator [routinator]

OpenBSD rpki-client [rpkiclient]

FORT Validator [fort]

The 'public' OpenSSL X509v3_addr_validate_path() and X509v3_asid_validate_path() interfaces do not read the Policy OIDs. Also, these interfaces are not referenced outside OpenSSL itself: [codesearch] and [github].

At the time of writing there are zero (0) certificates in the RPKI carrying the extensions and policy defined in [RFC8360].

6. Security Considerations

The authors believe that the revised validation algortihm introduces no new security vulnerabilities into the RPKI, because it cannot lead to any ROA and/or router certificates to be accepted if they contain resources that are not held by the issuer.

7. IANA Considerations

IANA is requested to reference this document in the "SMI Security for PKIX Certificate Policies" registry at:

id-cp-ipAddr-asNumber-v2

IANA is requested to reference this document in the "SMI Security for PKIX Certificate Extensions" registry at:

id-pe-ipAddrBlocks-v2

id-pe-autonomousSysIds-v2

IANA is requested to reference this document in the "SMI Security for PKIX Module Identifier" registry at:

id-mod-ip-addr-and-as-ident-v2

id-mod-ip-addr-and-as-ident-2v2

8. Acknowledgements

The authors would like to thank Tim Bruijnzeels, Mikael Abrahamsson, and Nick Hilliard for their helpful review of this document.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <https://www.rfc-editor.org/info/rfc3779>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <https://www.rfc-editor.org/info/rfc6482>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <https://www.rfc-editor.org/info/rfc6487>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8360] Huston, G., Michaelson, G., Martinez, C., Bruijnzeels, T., Newton, A., and D. Shaw, "Resource Public Key Infrastructure (RPKI) Validation Reconsidered", RFC 8360, DOI 10.17487/RFC8360, April 2018, <https://www.rfc-editor.org/info/rfc8360>.

9.2. Informative References

[codesearch]
Debian, "Debian Codesearch", February 2021, <https://codesearch.debian.net/search?q=X509v3_addr_validate_path&literal=1>.

[fort] Snijders, J., "Harmonize RFC 8360 and RFC 6487 in FORT", February 2021, <https://github.com/job/FORT-validator/commit/ff5f4b9313d5c553fa13bae427acb69665977727>.

[github] Github, "Github Search", February 2021, <https://github.com/search?q=X509v3_addr_validate_path&type=commits>.

[Report] Snijders, J., "[routing-wg] RFC 8360 should be the default (Was: RPKI Outage Post-Mortem)", January 2021, <https://www.ripe.net/ripe/mail/archives/routing-wg/2021-January/004220.html>.

[routinator]
Snijders, J., "Harmonize RFC 8360 and RFC 6487 in rpki-rs", February 2021, <https://github.com/job/rpki-rs/commit/d9fa8c72cf83ed6f25e4420eaaa9054078f15bc3>.

[rpkiclient]
Jeker, C., "rpki-client check IP and ASnum coverage only on ROAs", January 2021, <https://marc.info/?l=openbsd-tech&m=161011710120123&w=2>.

Authors' Addresses

Job Snijders
Fastly
Amsterdam
Netherlands

Email: job@fastly.com

Ben Maddison
Workonline
Cape-Town
South Africa

Email: benm@workonline.africa