Workgroup: Network Working Group
Internet-Draft: draft-ietf-sidrops-rpki-rsc-06
Published: 12 February 2022
Intended Status: Standards Track
Expires: 16 August 2022
Authors: J. Snijders Fastly
T. Harrison APNIC
B. Maddison Workonline

Resource Public Key Infrastructure (RPKI) object profile for Signed Checklist (RSC)

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 an arbitrary checksum listing with a specific 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

Copyright Notice

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 a specific set of Internet Number Resources.

Signed Checklists are expected to facilitate inter-domain business use-cases which depend on an ability to verify resource holdership. RPKI-based validation processes are expected to become the industry norm for automated Bring Your Own IP (BYOIP) on-boarding or establishment of physical interconnection between Autonomous Systems.

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 the RTA profile allows multiple signers to attest a single digital object through a checksum of its content, while the RSC profile allows a single signer to attest the existence of multiple digital objects. A single signer profile is considered a simplification for both 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 MUST be omitted from the RSC's X.509 End-Entity (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.

2.1. RSC End-Entity Certificates

The CA MUST only sign one RSC with each EE Certificate, and MUST generate a new key pair for each new RSC. This form of use of the associated EE Certificate is termed a "one-time-use" EE certificate Section 3 of [RFC6487].

3. The RSC ContentType

The ContentType for an RSC is defined as rpkiSignedChecklist, and has the numerical value of 1.2.840.113549.1.9.16.1.48.

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 a checklist for arbitrary 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-2010 -- in [RFC6268]
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
      pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }

  ASIdOrRange, IPAddressOrRange
  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) 48 }

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

FileNameAndHash ::= SEQUENCE {
  fileName        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 IPAddressFamilyItem

IPAddressFamilyItem ::= SEQUENCE {    -- AFI &amp;amp; optional SAFI --
   addressFamily        OCTET STRING (SIZE (2..3)),
```

```
   iPAddressOrRange     IPAddressOrRange }

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 FileNameAndHash objects. There is one FileNameAndHash entry for each arbitrary object referenced on the Signed Checklist. Each FileNameAndHash is an ordered pair of the name of the directory entry containing the digital object and the message digest of the digital object. The filename field is OPTIONAL.

5. RSC Validation

Before a Relying Party can use an RSC to validate a set of digital objects, the Relying Party MUST first validate the RSC. To validate an RSC, the Relying Party MUST perform all the validation checks specified in [RFC6488] (except checking for the presence of a SIA extension), and perform the following additional RSC-specific validation steps:

1. The RSC specification deviates from Section 4.8.8.2 of [RFC6487], the Subject Information Access extension MUST NOT be present on the End-Entity (EE) certificate signing the RSC.

2. The IP Addresses and AS Identifiers extension [RFC3779] is present in the end-entity (EE) certificate (contained within the RSC), and each IP address prefix(es) and/or AS Identifier(s) in the RSC is contained within the set of IP addresses specified by the EE Certificate's IP Addresses and AS Identifiers delegation extension.

3. For each FileNameAndHash entry in the RSC, if a filename field is present, the field's content MUST contain only characters specified in the Portable Filename Character Set as defined in [POSIX].

To verify a set of digital objects with an RSC:

* 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 FileNameAndHash, for the digital object to be considered as verified with the RSC.

6. Operational Considerations

When creating digital objects of a plain-text nature (such as ASCII, UTF-8, HTML, Javascript, XML, etc) it is RECOMMENDED to convert such objects into a lossless compressed form. Distributing plain-text objects within a compression envelope (such as GZIP [RFC1952]) might help avoid unexpected canonicalization at intermediate systems (which in turn would lead to checksum verification errors). Validator implementations are expected to treat a checksummed digital object as string of arbitrary single octets.

If a filename field is present, but no referenced digital object has a filename that matches the content of that field, a validator implementation SHOULD compare the message digest of each digital object to the value from the messageDigest field of the associated FileNameAndHash, and report matches to the client for further consideration.

7. Security Considerations

Relying parties are hereby warned that the data in a RPKI Signed Checklist is self-asserted. When determining the meaning of any data contained in an RPKI Signed Checklist, Relying Parties MUST NOT make any assumptions about the signer beyond the fact that it had sufficient control of the issuing CA to create the object. These data have not been verified by the Certificate Authority (CA) that issued the CA certificate to the entity that issued the EE Certificate used to validate the Signed Checklist.

RPKI Certificates are not bound to real world identities, see [I-D.ymbk-sidrops-rpki-has-no-identity] for an elaboration. Relying Parties can only associate real world entities to Internet Number Resources by additionally consulting an exogenous authority. Signed Checklists are a tool to communicate assertions 'signed with Internet Number Resources', not about any other aspect of the resource holder's business operations such as the identity of the resource holder itself.

RSC objects are not distributed through the RPKI Repository system. From this, it follows that third parties who do not have a copy of a given RSC, may not be aware of the existence of that RSC. Since RSC objects use EE Certificates, but all other currently defined types of RPKI object profiles are published in public CA repositories, an observer may infer from discrepancies in the Repository that RSC object(s) may exist. For example, if a CA does not use random serial numbers for Certificates, an observer could detect gaps between the serial numbers of the published EE Certificates. Similarly, if the CA includes a serial number on a CRL that does not match any published object, an observer could postulate an RSC EE Certificate was revoked.

Conversely, a gap in serial numbers does not imply that an RSC exists. Nor does an arbitrary (to the RP unknown) serial in a CRL imply an RSC object exists: the implicitly referenced object might not be a RSC, it might never have been published, or was revoked before it was visible to RPs. In general, it is not possible to confidently infer the existence or non-existence of RSCs from the Repository state without access to a given RSC.

While an one-time-use EE Certificate must only be used to generate and sign a single RSC object, CAs technically are not restricted from generating and signing multiple different RSC objects with a single keypair. Any RSC objects sharing the same EE Certificate can not be revoked individually.

8. Implementation status

This section is to be removed before publishing as an RFC.

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] written in Perl based on OpenSSL was provided by Tom Harrison from APNIC.

*A signer implementation [rpkimancer] written in Python was developed by Ben Maddison.

*Example .sig files were created by Job Snijders with the use of OpenSSL.

*A validator implementation based on OpenBSD rpki-client and LibreSSL was developed by Job Snijders.

*A validator implementation [FORT] based on the FORT validator was developed by Alberto Leiva.

9. IANA Considerations

9.1. SMI Security for S/MIME CMS Content Type (1.2.840.113549.1.9.16.1)

The IANA has permanently allocated for this document in the SMI Security for S/MIME CMS Content Type (1.2.840.113549.1.9.16.1) registry:

Decimal	Description	References
48	id-ct-signedChecklist	[draft-ietf-sidrops-rpki-rsc]

Upon publication of this document, IANA is requested to reference the RFC publication instead of this draft.

9.2. RPKI Signed Objects sub-registry

The IANA is requested to register the OID for the RPKI Signed Checklist in the registry created by [RFC6488] as following:

Name	OID	Specification
Signed Checklist	1.2.840.113549.1.9.16.1.48	[draft-ietf-sidrops-rpki-r

9.3. File Extension

The IANA is requested to add an item for the Signed Checklist file extension to the "RPKI Repository Name Scheme" registry created by [RFC6481] as follows:

Filename Extension	RPKI Object	Reference
.sig	Signed Checklist	[draft-ietf-sidrops-rpki-rsc]

9.4. SMI Security for S/MIME Module Identifier (1.2.840.113549.1.9.16.0)

The IANA is requested to add an item to the "SMI Security for S/MIME Module Identifier" registry as follows:

Decimal	Description	References
TBD	id-mod-rpkiSignedChecklist-2021	[draft-ietf-sidrops-rpki-rsc]

9.5. Media Type

The IANA is requested to register the media type application/rpki-checklist in the Provisional Standard Media Type registry 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>.

[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>.

[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>.

[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>.

[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

[FORT] LACNIC and NIC.MX, "FORT", May 2021, <https://github.com/NICMx/FORT-validator>.

[I-D.ietf-sidrops-rpki-rta] Michaelson, G., Huston, G., Harrison, T., Bruijnzeels, T., and M. Hoffmann, "A profile for Resource Tagged Attestations (RTAs)", Work in Progress, Internet-Draft, draft-ietf-sidrops-rpki-rta-00, 21 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-rpki-rta-00>.

[I-D.ymbk-sidrops-rpki-has-no-identity] Bush, R. and R. Housley, "The I in RPKI does not stand for Identity", Work in Progress, Internet-Draft, draft-ymbk-sidrops-rpki-has-no-identity-00, 16 March 2021, <https://datatracker.ietf.org/doc/html/draft-ymbk-sidrops-rpki-has-no-identity-00>.

[POSIX] IEEE and The Open Group, "The Open Group's Base Specifications, Issue 7", 2016, <https://publications.opengroup.org/standards/unix/c165>.

[RFC1952] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, DOI 10.17487/RFC1952, May 1996, <https://www.rfc-editor.org/info/rfc1952>.

[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>.

[rpkimancer] Maddison, B., "rpkimancer", May 2021, <https://github.com/benmaddison/rpkimancer>.

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

Appendix A. Acknowledgements

The authors wish 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 authors thank Russ Housley for reviewing the ASN.1 notation and providing suggestions. The authors would like to thank Nimrod Levy, Tim Bruijnzeels, Alberto Leiva, Ties de Kock, and Peter Peele for document review and suggestions.

Appendix B. Document changelog

This section is to be removed before publishing as an RFC.

B.1. changes from -05 -> -06

*Non-content-related updates.

B.2. changes from -04 -> -05

*Ties contributed clarifications.

B.3. changes from -03 -> -04

*Alberto pointed out the asID validation also needs to be documented.

B.4. changes from -02 -> -03

*Reference the IANA assigned OID

*Clarify validation rules

B.5. changes from -01 -> -02

*Clarify RSC is part of a puzzle, not panacea. Thanks Randy & Russ.

B.6. changes from -00 -> -01

*Readability improvements

*Update document category to match the registry allocation policy requirement.

B.7. individual submission phase

*On-the-wire change: the 'Filename' switched from 'required' to 'optional'. Some SIDROPS Working Group participants proposed a checksum itself is the most minimal information required to address digital objects.

Authors' Addresses

Job Snijders
Fastly
Amsterdam
Netherlands

Email: job@fastly.com

Tom Harrison
Asia Pacific Network Information Centre
6 Cordelia St
South Brisbane QLD 4101
Australia

Email: tomh@apnic.net

Ben Maddison
Workonline Communications
Cape Town
South Africa

Email: benm@workonline.africa