Workgroup: Network Working Group
Internet-Draft: draft-ietf-sidrops-rpki-rsc-09
Published: 8 September 2022
Intended Status: Standards Track
Expires: 12 March 2023
Authors: J. Snijders Fastly
T. Harrison APNIC
B. Maddison Workonline

A profile for Resource Public Key Infrastructure (RPKI) Signed Checklists (RSC)

Abstract

This document defines a Cryptographic Message Syntax (CMS) protected content type for use with the Resource Public Key Infrastructure (RPKI) to carry a general purpose listing of checksums (a 'checklist'). The objective is to allow an attestation, termed an RPKI Signed Checklist (RSC), which contains one or more checksums of arbitrary digital objects (files) that are signed "with resources", and which, when validated, confirms that the respective Internet Resource Holder produced the RSC.

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 12 March 2023.

Copyright Notice

Table of Contents

1. Introduction

This document defines a Cryptographic Message Syntax (CMS) [RFC5652] [RFC6268] protected content type for a general purpose listing of checksums (a 'checklist'), for use with the Resource Public Key Infrastructure (RPKI) [RFC6480]. The protected CMS content type is intended to provide for the creation and validation of an RPKI Signed Checklist (RSC): a checksum listing signed with a specific set of Internet Number Resources. The objective is to allow an attestation that, when validated, provides a means to confirm a given Internet Resource Holder produced the RPKI Signed Checklist (RSC).

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 [RFC9286], 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 content 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. For example, it might be a USB stick, a web interface secured with HTTPS, a PGP-signed email, a T-shirt printed with a QR code, or a carrier pigeon.

2.1. RSC End-Entity Certificates

The Certification Authority (CA) MUST only sign one RSC with each End-Entity (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, with Object Identifier (OID) 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-2022
  { iso(1) member-body(2) us(840) rsadsi(113549)
    pkcs(1) pkcs9(9) smime(16) mod(0)
    id-mod-rpkiSignedChecklist-2022(73) }

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) }

  IPAddressOrRange, ASIdOrRange
  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              PortableFilename OPTIONAL,
  hash                  Digest }

PortableFilename ::=
  IA5String (FROM("a".."z" | "A".."Z" | "0".."9" | "." | "_" | "-"))

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

ConstrainedIPAddrBlocks ::=
  SEQUENCE (SIZE(1..MAX)) OF ConstrainedIPAddressFamily
```

```
ConstrainedIPAddressFamily ::= SEQUENCE {
  addressFamily         OCTET STRING (SIZE(2)),
  addressesOrRanges     SEQUENCE (SIZE(1..MAX)) OF IPAddressOrRange }

ConstrainedASIdentifiers ::= SEQUENCE {
  asnum [0]             SEQUENCE (SIZE(1..MAX)) OF ASIdOrRange }

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 be a subset of the set of resources listed by the EE Certificate carried in the CMS certificates field.

If the asID field is present, it MUST contain an instance of ConstrainedASIdentifiers.

If the ipAddrBlocks field is present, it MUST contain an instance of ConstrainedIPAddrBlocks.

At least one of asID or ipAddrBlocks MUST be present.

Each of ConstrainedASIdentifiers and ConstrainedIPAddrBlocks are specified such that the resulting DER-encoded data instances are binary compatible with, respectively, ASIdentifiers and IPAddrBlocks defined in [RFC3779].

Implementations encountering decoding errors whilst attempting to read DER-encoded data using this specification should be aware of the possibility that the data may have been encoded using an implementation intended for use with [RFC3779]. Such data may contain elements prohibited by the current specification.

Attempting to decode the errored data using the more permissive specification contained in [RFC3779] may enable implementors to gather additional context for use when reporting errors to the user.

However, implementations MUST NOT ignore errors resulting from the more restrictive definitions contained herein: in particular, such errors MUST cause the validation procedure described in Section 5 to fail.

4.2.1. ConstrainedASIdentifiers type

ConstrainedASIdentifiers is a SEQUENCE, consisting of a single field "asnum", itself containing a SEQUENCE OF one or more ASIdOrRange instances as defined in [RFC3779].

ConstrainedASIdentifiers is defined such that the resulting DER-encoded data are binary compatible with ASIdentifiers defined in [RFC3779].

4.2.2. ConstrainedIPAddrBlocks type

ConstrainedIPAddrBlocks is a SEQUENCE OF one or more instances of ConstrainedIPAddressFamily.

There MUST be only one instance of ConstrainedIPAddressFamily per unique AFI.

The elements of ConstrainedIPAddressFamily MUST be ordered by ascending addressFamily values (treating the octets as unsigned numbers). Thus, when both IPv4 and IPv6 addresses are specified, the IPv4 addresses MUST precede the IPv6 addresses (since the IPv4 AFI of 0001 is less than the IPv6 AFI of 0002).

ConstrainedIPAddrBlocks is defined such that the resulting DER-encoded data are binary compatible with IPAddrBlocks defined in [RFC3779].

4.2.2.1. ConstrainedIPAddressFamily type

4.2.2.1.1. addressFamily field

The addressFamily field is an OCTET STRING containing a two-octet Address Family Identifier (AFI), in network byte order. Unlike IPAddrBlocks [RFC3779], a third octet containing a Subsequent Address Family Identifier (SAFI) MUST NOT be present. AFIs are specified in the Address Family Numbers registry [IANA.ADDRESS-FAMILY-NUMBERS] maintained by IANA.

4.2.2.1.2. addressesOrRanges field

The addressesOrRanges element is a SEQUENCE OF one or more IPAddressOrRange values, as defined in [RFC3779]. The rules for canonicalization and encoding defined in Section 2.2.3.6 of [RFC3779] apply to the value of addressesOrRanges.

4.3. digestAlgorithm

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

4.4. checkList

This field is a SEQUENCE OF one or more FileNameAndHash values. There is one FileNameAndHash entry for each digital object referenced on the Signed Checklist.

4.4.1. FileNameAndHash

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 hash field is mandatory. The value of the hash field is the calculated message digest of the digital object. The hashing algorithm is specified in the digestAlgorithm field.

The fileName field is OPTIONAL. This is to allow Signed Checklists to be used in a "stand-alone" fashion in which nameless digital objects are addressed directly through their respective message digest rather than through a file system abstraction.

If the fileName field is present then its value:

* MUST contain only characters specified in the Portable Filename Character Set as defined in [POSIX].

* MUST be unique with respect to the other FileNameAndHash elements of checkList for which the fileName field is also present.

Conversely, if the fileName field is omitted, then the value of the hash field MUST be unique with respect to the other FileNameAndHash elements of checkList for which the fileName field is also omitted.

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 an SIA extension, which MUST NOT be present in the EE X.509 certificate

Section 4.8.8.2 of [RFC6487]), and perform the following additional RSC-specific validation steps:

1. The contents of the CMS eContent field MUST conform to all of the constraints described in Section 4 including the constraints described in Section 4.4.1.

2. If the asID field is present within the contents of the 'resources' field, then the AS Resources extension [RFC3779] MUST be present in the EE certificate contained in the CMS certificates field. The AS identifiers present in the eContent 'resources' field MUST be a subset of those present in the certificate extension. The EE certificate's AS Resources extension MUST NOT contain "inherit" elements.

3. If the ipAddrBlocks field is present within the contents of the 'resources' field, then the IP Resources extension [RFC3779] MUST be present in the EE certificate contained in the CMS certificates field. The IP addresses present in the eContent 'resources' field MUST be a subset of those present in the certificate extension. The EE certificate's IP Resources extension MUST NOT contain "inherit" elements.

6. Verifying files or data using RSC

To verify a set of digital objects with an RSC:

*The RSC MUST be validated according to the procedure described in Section 5. If the RSC cannot be validated, verification MUST fail. This error SHOULD be reported to the user.

*For every digital object to be verified:

1. If the verification procedure is provided with a file name for the object being verified (e.g. because the user has provided a file system path from which to read the object) then verification SHOULD proceed in "filename-aware" mode. Otherwise, verification SHOULD proceed in "filename-unaware" mode.

   Implementations MAY provide an option to override the verification mode, for example to ignore the fact that the object is to be read from a file.

2. The message digest MUST be computed from the file contents or data using the digest algorithm specified in the digestAlgorithm field of the RSC.

3. The digest computed in step 2 MUST be compared to the value appearing in the hash field of all FileNameAndHash elements of the checkList field of the RSC.

   One or more FileNameAndHash elements MUST be found with a matching hash value, otherwise verification MUST fail and the error SHOULD be reported to the user.

4. If the mode selected in step 1 is "filename-aware" then exactly one of the FileNameAndHash elements matched in step 3 MUST contain a fileName field value exactly matching the file name of the object being verified.

   Alternatively, if the mode selected in step 1 is "filename-unaware" then exactly one of the FileNameAndHash elements matched in step 3 MUST have the fileName field omitted.

   Otherwise, verification MUST fail, and the error SHOULD be reported to the user.

Note that in the above procedure, not all elements of checkList necessarily need be used. That is, it is not an error if the length of checkList is greater than the size of the set of digital objects to be verified. However, in this situation, implementations SHOULD issue a warning to the user, allowing for corrective action to be taken if necessary.

7. Operational Considerations

When creating digital objects of a plain-text nature (such as ASCII, UTF-8, HTML, Javascript, XML, etc.) converting such objects into a lossless compressed form is RECOMMENDED. 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 digital object within the set of to-be-verified digital objects 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 hash field of the associated FileNameAndHash and report matches to the user for further consideration; or report an error indicating no file by that name exists.

8. 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 [RFC9255] 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 an RSC, it might have never 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 a 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.

9. 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 for a previous version of this specification.

- A validator implementation [prover] was developed by Mikhail Puzanov.

10. IANA Considerations

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

The IANA has 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.

10.2. RPKI Signed Objects sub-registry

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

Name	OID	Specification
Signed Checklist	1.2.840.113549.1.9.16.1.48	[RFC-TBD]

10.3. File Extension

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

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

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

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

The IANA has permanently allocated for this document in the "SMI Security for S/MIME Module Identifier (1.2.840.113549.1.9.16.0)" registry:

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

Upon publication of this document, IANA is requested to update the "Description" column to read "id-mod-rpkiSignedChecklist-2022", and to reference the RFC publication instead of this draft.

10.5. Media Type

The IANA has registered the media type application/rpki-checklist in the "Provisional Standard Media Type" registry as follows:

```
Type name: application
Subtype name: rpki-checklist
Required parameters: N/A
Optional parameters: N/A
Encoding considerations: binary
Security considerations: Carries an RPKI Signed Checklist
    [RFC-to-be]. This media type contains no active
    content. See Section 5 of [RFC-to-be] for further
    information.
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: IETF
```

Upon publication of this document, IANA is requested to move this registration to the "Media Types" registry and to reference the RFC publication instead of this draft.

11. References

11.1. Normative References

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

[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>.

[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>.

[RFC9286] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 9286, DOI 10.17487/RFC9286, June 2022, <https://www.rfc-editor.org/info/rfc9286>.

11.2. Informative References

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

[I-D.ietf-sidrops-rpki-rta] Michaelson, G. 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://www.ietf.org/archive/id/draft-ietf-sidrops-rpki-rta-00.txt>.

[IANA.ADDRESS-FAMILY-NUMBERS] IANA, "Address Family Numbers", 19 October 2021, <http://www.iana.org/assignments/address-family-numbers>.

[prover] Puzanov, M., "rpki-prover", September 2022, <https://github.com/lolepezy/rpki-prover/pull/126/files>.

[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>.

[RFC6268] Schaad, J. and S. Turner, "Additional New ASN.1 Modules for the Cryptographic Message Syntax (CMS) and the Public Key Infrastructure Using X.509 (PKIX)", RFC 6268, DOI 10.17487/RFC6268, July 2011, <https://www.rfc-editor.org/info/rfc6268>.

[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>.

[RFC9255] Bush, R. and R. Housley, "The 'I' in RPKI Does Not Stand for Identity", RFC 9255, DOI 10.17487/RFC9255, June 2022, <https://www.rfc-editor.org/info/rfc9255>.

[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, 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, Peter Peele, Claudio Jeker, Theo Buehler, Donald Eastlake, Erik Kline, Robert Wilton, Roman Danyliw, Éric Vyncke, Lars Eggert, Paul Wouters, and Murray S. Kucherawy for document review and suggestions.

Appendix B. Document changelog

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

B.1. changes from -10 -> -11

* Incorporate feedback from Robert Wilton.
* Incorporate feedback from Roman Danyliw.
* Incorporate feedback from Éric Vyncke.
* Add Mikhail Puzanov's implementation.
* Incorporate feedback from Lars Eggert's review.
* Incorporate feedback from Paul Wouters.
* Incorporate feedback from Murray S. Kucherawy.

B.2. changes from -09 -> -10

* Incorporate SECDIR feedback.

B.3. changes from -08 -> -09

* Updated manifests refs to RFC9286
* Added normative ref to RFC6268 (CMS)
* Cleaned up ASN.1 formatting
* Updated ASN.1 module OID following early allocation
* Updated draft-ietf-sidrops-rpki-has-no-identity to RFC9255
* Clean up IANA considerations

B.4. changes from -07 -> -08

* Theo requested explanation as to why fileName is OPTIONAL
* Russ & Randy requested implementor guidance when RFC3779-generated data fails to decode
* Added uniqueness constraints for fileName and hash contents
* Improved validation and verification procedure description
* Incorporated character-set constraints for fileName in ASN.1 module

B.5. changes from -06 -> -07

*Change wire format to allow use of commonly deployed libcrypto APIs.

B.6. changes from -05 -> -06

*Non-content-related updates.

B.7. changes from -04 -> -05

*Ties contributed clarifications.

B.8. changes from -03 -> -04

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

B.9. changes from -02 -> -03

*Reference the IANA assigned OID

*Clarify validation rules

B.10. changes from -01 -> -02

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

B.11. changes from -00 -> -01

*Readability improvements

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

B.12. 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