Stream: Internet Engineering Task Force (IETF)
RFC: 9582
Obsoletes: 6482
Category: Standards Track
Published: April 2024
ISSN: 2070-1721
Authors: J. Snijders *Fastly*, B. Maddison *Workonline*, M. Lepinski *Carleton College*, D. Kong *Raytheon*, S. Kent *Independent*

RFC 9582
A Profile for Route Origin Authorizations (ROAs)

Abstract

This document defines a standard profile for Route Origin Authorizations (ROAs). A ROA is a digitally signed object that provides a means of verifying that an IP address block holder has authorized an Autonomous System (AS) to originate routes to one or more prefixes within the address block. This document obsoletes RFC 6482.

Status of This Memo

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 system, a mechanism is needed to allow entities to verify that an Autonomous System (AS) has been given permission by an IP address block holder to advertise routes to one or more prefixes within that block. A Route Origin Authorization (ROA) provides this function.

The ROA makes use of the template for RPKI digitally signed objects [RFC6488], which defines a Cryptographic Message Syntax (CMS) wrapper [RFC5652] for the ROA content as well as a generic validation procedure for RPKI signed objects. Therefore, to complete the specification of the ROA (see Section 4 of [RFC6488]), this document defines:

- The OID that identifies the signed object as being a ROA. (This OID appears within the eContentType in the encapContentInfo object as well as the content-type signed attribute in the signerInfo object.)
- The ASN.1 syntax for the ROA eContent. (This is the payload that specifies the AS being authorized to originate routes as well as the prefixes to which the AS may originate routes.) The ROA eContent is ASN.1 encoded using the Distinguished Encoding Rules (DER) [X.690].
- Additional steps required to validate ROAs (in addition to the validation steps specified in [RFC6488]).

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.

1.2. Changes from RFC 6482

This section summarizes the significant changes between [RFC6482] and the profile described in this document.

- Clarified the requirements for the IP address and AS identifier X.509 certificate extensions.
- Strengthened the ASN.1 formal notation and definitions.
- Incorporated errata for RFC 6482.
- Added an example ROA eContent payload and a ROA.
- Specified a canonicalization procedure for the content of ipAddrBlocks.

2. Related Work

It is assumed that the reader is familiar with the terms and concepts described in "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile" [RFC5280] and "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779].

Additionally, this document makes use of the RPKI signed object profile [RFC6488]; thus, familiarity with that document is assumed. Note that the RPKI signed object profile makes use of certificates adhering to the RPKI resource certificate profile [RFC6487]; thus, familiarity with that profile is also assumed.

3. The ROA ContentType

The content-type for a ROA is defined as routeOriginAuthz and has the numerical value 1.2.840.113549.1.9.16.1.24.

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

4. The ROA eContent

The content of a ROA identifies a single AS that has been authorized by the address space holder to originate routes and a list of one or more IP address prefixes that will be advertised. If the address space holder needs to authorize multiple ASes to advertise the same set of address prefixes, the holder issues multiple ROAs, one per AS number. A ROA is formally defined as:

```
RPKI-ROA-2023 { iso(1) member-body(2) us(840) rsadsi(113549)
  pkcs(1) pkcs9(9) smime(16) mod(0) id-mod-rpkiROA-2023(75) }

DEFINITIONS EXPLICIT TAGS ::=
BEGIN

IMPORTS
  CONTENT-TYPE
  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) } ;

ct-routeOriginAttestation CONTENT-TYPE ::=
  { TYPE RouteOriginAttestation
    IDENTIFIED BY id-ct-routeOriginAuthz }

id-ct-routeOriginAuthz OBJECT IDENTIFIER ::=
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-9(9) id-smime(16) id-ct(1) routeOriginAuthz(24) }

RouteOriginAttestation ::= SEQUENCE {
  version [0]   INTEGER DEFAULT 0,
  asID          ASID,
```

```
  ipAddrBlocks  SEQUENCE (SIZE(1..2)) OF ROAIPAddressFamily }

ASID ::= INTEGER (0..4294967295)

ROAIPAddressFamily ::= SEQUENCE {
  addressFamily ADDRESS-FAMILY.&afi ({AddressFamilySet}),
  addresses     ADDRESS-FAMILY.&Addresses ({AddressFamilySet}
{@addressFamily}) }

ADDRESS-FAMILY ::= CLASS {
  &afi          OCTET STRING (SIZE(2)) UNIQUE,
  &Addresses
} WITH SYNTAX { AFI &afi ADDRESSES &Addresses }

AddressFamilySet ADDRESS-FAMILY ::=
  { addressFamilyIPv4 | addressFamilyIPv6 }

addressFamilyIPv4 ADDRESS-FAMILY ::=
  { AFI afi-IPv4 ADDRESSES ROAAddressesIPv4 }
addressFamilyIPv6 ADDRESS-FAMILY ::=
  { AFI afi-IPv6 ADDRESSES ROAAddressesIPv6 }

afi-IPv4 OCTET STRING ::= '0001'H
afi-IPv6 OCTET STRING ::= '0002'H

ROAAddressesIPv4 ::= SEQUENCE (SIZE(1..MAX)) OF ROAIPAddress{ub-IPv4}
ROAAddressesIPv6 ::= SEQUENCE (SIZE(1..MAX)) OF ROAIPAddress{ub-IPv6}

ub-IPv4 INTEGER ::= 32
ub-IPv6 INTEGER ::= 128

ROAIPAddress {INTEGER: ub} ::= SEQUENCE {
  address       BIT STRING (SIZE(0..ub)),
  maxLength     INTEGER (0..ub) OPTIONAL }

END
```

4.1. The version Element

The version number of the RouteOriginAttestation entry **MUST** be 0.

4.2. The asID Element

The asID element contains the AS number that is authorized to originate routes to the given IP address prefixes.

4.3. The ipAddrBlocks Element

The ipAddrBlocks element encodes the set of IP address prefixes to which the AS is authorized to originate routes. Note that the syntax here is more restrictive than that used in the IP Address Delegation extension defined in [RFC3779]. That extension can represent arbitrary address ranges, whereas ROAs need to represent only IP prefixes.

4.3.1. Type ROAIPAddressFamily

Within the ROAIPAddressFamily structure, the addressFamily element contains the Address Family Identifier (AFI) of an IP address family. This specification only supports IPv4 and IPv6; therefore, addressFamily **MUST** be either 0001 or 0002. IPv4 prefixes **MUST NOT** appear as IPv4-mapped IPv6 addresses (Section 2.5.5.2 of [RFC4291]).

There **MUST** be only one instance of ROAIPAddressFamily per unique AFI in the ROA. Thus, the ROAIPAddressFamily structure **MUST NOT** appear more than twice.

The addresses element represents IP prefixes as a sequence of type ROAIPAddress.

4.3.2. Type ROAIPAddress

A ROAIPAddress structure is a sequence containing an address element of type IPAddress and an optional maxLength element of type INTEGER. See Section 2.2.3.8 of [RFC3779] for more details on type IPAddress.

4.3.2.1. The address Element

The address element is of type IPAddress and represents a single IP address prefix.

4.3.2.2. The maxLength Element

When present, the maxLength element specifies the maximum length of the IP address prefix that the AS is authorized to advertise. The maxLength element **SHOULD NOT** be encoded if the maximum length is equal to the prefix length. Certification Authorities **SHOULD** anticipate that future Relying Parties will become increasingly stringent in considering the presence of superfluous maxLength elements an encoding error.

If present, the maxLength element **MUST** be:

- an integer greater than or equal to the length of the accompanying prefix, and
- less than or equal to the maximum length (in bits) of an IP address in the applicable address family: 32 in the case of IPv4 and 128 in the case of IPv6.

For example, if the IP address prefix is 203.0.113.0/24 and maxLength is 26, the AS is authorized to advertise any more-specific prefix with a maximum length of 26. In this example, the AS would be authorized to advertise 203.0.113.0/24, 203.0.113.128/25, or 203.0.113.192/26, but not 203.0.113.0/27. See [RFC9319] for more information on the use of maxLength.

When the maxLength element is not present, the AS is only authorized to advertise the exact prefix specified in the ROAIPAddress structure's address element.

4.3.2.3. Note on Overlapping or Superfluous Information Encoding

Note that a valid ROA may contain an IP address prefix (within a ROAIPAddress element) that is encompassed by another IP address prefix (within a separate ROAIPAddress element). For example, a ROA may contain the prefix 203.0.113.0/24 with maxLength 26, as well as the prefix

203.0.113.0/28 with maxLength 28. This ROA would authorize the indicated AS to advertise any prefix beginning with 203.0.113 with a minimum length of 24 and a maximum length of 26, as well as the specific prefix 203.0.113.0/28.

Additionally, a ROA **MAY** contain two ROAIPAddress elements, where the IP address prefix is identical in both cases. However, this is **NOT RECOMMENDED**, because in such a case, the ROAIPAddress element with the shorter maxLength grants no additional privileges to the indicated AS and thus can be omitted without changing the meaning of the ROA.

4.3.3. Canonical Form for ipAddrBlocks

As the data structure described by the ROA ASN.1 module allows for many different ways to represent the same set of IP address information, a canonical form is defined such that every set of IP address information has a unique representation. In order to produce and verify this canonical form, the process described in this section **SHOULD** be used to ensure that information elements are unique with respect to one another and sorted in ascending order. Certification Authorities **SHOULD** anticipate that future Relying Parties will impose a strict requirement for the ipAddrBlocks field to be in this canonical form. This canonicalization procedure builds upon the canonicalization procedure specified in Section 2.2.3.6 of [RFC3779].

In order to semantically compare, sort, and deduplicate the contents of the ipAddrBlocks field, each ROAIPAddress element is mapped to an abstract data element composed of four integer values:

afi
: The AFI value appearing in the addressFamily field of the containing ROAIPAddressFamily as an integer.

addr
: The first IP address of the IP prefix appearing in the ROAIPAddress address field, as a 32-bit (IPv4) or 128-bit (IPv6) integer value.

plen
: The length of the IP prefix appearing in the ROAIPAddress address field as an integer value.

mlen
: The value appearing in the maxLength field of the ROAIPAddress element, if present; otherwise, the above prefix length value.

Thus, the equality or relative order of two ROAIPAddress elements can be tested by comparing their abstract representations.

4.3.3.1. Comparator

The set of ipAddrBlocks is totally ordered. The order of two ipAddrBlocks is determined by the first non-equal comparison in the following list.

1. Data elements with a lower afi value precede data elements with a higher afi value.
2. Data elements with a lower addr value precede data elements with a higher addr value.
3. Data elements with a lower plen value precede data elements with a higher plen value.
4. Data elements with a lower mlen value precede data elements with a higher mlen value.

Data elements for which all four values compare equal are duplicates of one another.

4.3.3.2. Example Implementations

- A sorting implementation [roasort-c] in ISO/IEC 9899:1999 ("ANSI C99").
- A sorting implementation [roasort-rs] in the Rust 2021 Edition.

5. ROA Validation

Before a relying party can use a ROA to validate a routing announcement, the relying party **MUST** first validate the ROA. To validate a ROA, the relying party **MUST** perform all the validation checks specified in [RFC6488] as well as the following additional ROA-specific validation steps:

- The IP Address Delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ROA), and every IP address prefix in the ROA payload is contained within the set of IP addresses specified by the EE certificate's IP Address Delegation extension.
- The EE certificate's IP Address Delegation extension **MUST NOT** contain "inherit" elements as described in [RFC3779].
- The Autonomous System Identifier Delegation Extension described in [RFC3779] is not used in ROAs and **MUST NOT** be present in the EE certificate.
- The ROA content fully conforms with all requirements specified in Sections 3 and 4.

If any of the above checks fail, the ROA in its entirety **MUST** be considered invalid and an error **SHOULD** be logged.

6. Security Considerations

There is no assumption of confidentiality for the data in a ROA; it is anticipated that ROAs will be stored in repositories that are accessible to all ISPs, and perhaps to all Internet users. There is no explicit authentication associated with a ROA, since the PKI used for ROA validation provides authorization but not authentication. Although the ROA is a signed, application-layer object, there is no intent to convey non-repudiation via a ROA.

The purpose of a ROA is to convey authorization for an AS to originate a route to the prefix or prefixes in the ROA. Thus, the integrity of a ROA **MUST** be established. The ROA specification makes use of the RPKI signed object format; thus, all security considerations discussed in [RFC6488] also apply to ROAs. Additionally, the signed object profile uses the CMS signed message format for integrity; thus, ROAs inherit all security considerations associated with that data structure.

The right of the ROA signer to authorize the target AS to originate routes to the prefix or prefixes is established through the use of the address space and AS number PKI as described in [RFC6480]. Specifically, one **MUST** verify the signature on the ROA using an X.509 certificate issued under this PKI and check that the prefix or prefixes in the ROA are contained within those in the certificate's IP Address Delegation Extension.

7. IANA Considerations

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

IANA has updated the id-ct-routeOriginAuthz entry in the "SMI Security for S/MIME CMS Content Type (1.2.840.113549.1.9.16.1)" registry as follows:

Decimal	Description	References
24	id-ct-routeOriginAuthz	RFC 9582

Table 1

7.2. RPKI Signed Objects Registry

IANA has updated the Route Origination Authorization entry in the "RPKI Signed Objects" registry created by [RFC6488] as follows:

Name	OID	Reference
Route Origination Authorization	1.2.840.113549.1.9.16.1.24	RFC 9582

Table 2

7.3. File Extension

IANA has updated the entry for the ROA file extension in the "RPKI Repository Name Schemes" registry created by [RFC6481] as follows:

Filename Extension	RPKI Object	Reference
.roa	Route Origination Authorization	RFC 9582

Table 3

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

IANA has allocated the following entry in the "SMI Security for S/MIME Module Identifier (1.2.840.113549.1.9.16.0)" registry:

Decimal	Description	References
75	id-mod-rpkiROA-2023	RFC 9582

Table 4

7.5. Media Type

IANA has updated the media type application/rpki-roa in the "Media Types" registry as follows:

Type name: application

Subtype name: rpki-roa

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: binary

Security considerations: Carries an RPKI ROA (RFC 9582). This media type contains no active content. See Section 6 of RFC 9582 for further information.

Interoperability considerations: None

Published specification: RFC 9582

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 prefixes and an AS identifier as defined in RFC 9582.
Magic number(s): None
File extension(s): .roa
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

Change controller: IETF

8. References

8.1. 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>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

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

[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>.

[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>.

[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>.

[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>.

[X.690] 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, February 2021.

8.2. Informative References

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[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>.

[RFC9319] Gilad, Y., Goldberg, S., Sriram, K., Snijders, J., and B. Maddison, "The Use of maxLength in the Resource Public Key Infrastructure (RPKI)", BCP 185, RFC 9319, DOI 10.17487/RFC9319, October 2022, <https://www.rfc-editor.org/info/rfc9319>.

[roasort-c] "ROA sorter in C", commit 68969ea, July 2023, <https://github.com/job/roasort>.

[roasort-rs] "ROA sorter in Rust", commit 023e756, August 2023, <https://github.com/benmaddison/roasort>.

Appendix A. Example ROA eContent Payload

An example of a DER-encoded ROA eContent is provided below, with annotation following the "#" character.

```
$ echo
302402023CCA301E301C04020002301630090307002001067C208C30090307002A0EB2400000
\
  | xxd -r -ps \
  | openssl asn1parse -i -dump -inform DER
    0:d=0  hl=2 l=  36 cons: SEQUENCE                    #
RouteOriginAttestation
    2:d=1  hl=2 l=   2 prim:  INTEGER           :3CCA    # asID 15562
    6:d=1  hl=2 l=  30 cons:  SEQUENCE                   # ipAddrBlocks
    8:d=2  hl=2 l=  28 cons:   SEQUENCE                  #  ROAIPAddressFamily
   10:d=3  hl=2 l=   2 prim:    OCTET STRING             #   addressFamily
      0000 - 00 02                              ..       #    IPv6
   14:d=3  hl=2 l=  22 cons:    SEQUENCE                 #   addresses
   16:d=4  hl=2 l=   9 cons:     SEQUENCE                #    ROAIPAddress
   18:d=5  hl=2 l=   7 prim:      BIT STRING             #     address
      0000 - 00 20 01 06 7c 20 8c               . ..| .  #      2001:67c:
208c::/48
   27:d=4  hl=2 l=   9 cons:     SEQUENCE                #    ROAIPAddress
   29:d=5  hl=2 l=   7 prim:      BIT STRING             #     address
      0000 - 00 2a 0e b2 40                     .*..@    #      2a0e:b240::/48
      0007 - <SPACES/NULS>
```

Below is a complete RPKI ROA signed object, Base64 encoded per [RFC4648].

```
MIIHCwYJKoZIhvcNAQcCoIIG/DCCBvgCAQMxDTALBglghkgBZQMEAgEwNwYLKoZIhvcNAQkQ
ARigKAQmMCQCAjzKMB4wHAQCAAIwFjAJAwcAIAEGfCCMMAkDBwAqDrJAAACgggT7MIIE9zCC
A9+gAwIBAgIDAIb5MA0GCSqGSIb3DQEBCwUAMDMxMTAvBgNVBAMTKDM4ZTE0ZjkyZmRjN2Nj
ZmJmYzE4MjM2MTUyM2FlMjdkNjk3ZTk1MmYwHhcNMjIwNjE3MDAyNDIyWhcNMjMwNzAxMDAw
MDAwWjAzMTEwLwYDVQQDEyhBM0Q5NjQyNDU3NDlCQjZERDVBQjFGMkU4MzBFMzNBNkM1MTQ2
RThGMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA4CRG1t04YFLq3fctx2ThNfr6
Vxsd2wZzcZhQJgUdlvUyfUPISWMwuPfpGjviqtCEzh5aNePGpLopkIES08egzTmJ78Is6+kW
LXwy9CcwT7gmP9qOTSEi8h4qcyajxHbAwDEjROVNSujhLGeB74S9IQTn2Ertp2Et2xPq/kXw
+eiBHtOL2h2I7/UOZxHOHuNuHby+VbhFaxgPA7rVfdlUAf9yYxQvyZtB7kHT/EwAR4c9SYWu
0rvbWNJwWehzlT74V1XaknRXQjkKYHe34Fyyx9FY86uX4uN8rPuIzkd7n6g81pUZRIuk/3tc
/DjbHNAD3qWVQ+0aqNdkunoJhQccZwIDAQABo4ICEjCCAg4wHQYDVR0OBBYEFKPZZCRXSbtt
1asfLoMOM6bFFG6PMB8GA1UdIwQYMBaAFDjhT5L9x8z7/BgjYVI64n1pfpUvMBgGA1UdIAEB
/wQOMAwwCgYIKwYBBQUHDgIwZAYDVR0fBF0wWzBZoFegVYZTcnN5bmM6Ly9jaGxvZS5zb2Jv
cm5vc3QubmV0L3Jwa2kvUklQRS1ubGpvYnNuaWpkZXJzL09PRlBrdjNIelB2OEdDTmhVanJp
ZldsLWxTOC5jcmwwZAYIKwYBBQUHAQEEWDBWMFQGCCsGAQUFBzAChkhyc3luYzovL3Jwa2ku
cmlwZS5uZXQvcmVwb3NpdG9yeS9ERUZBVUxUL09PRlBrdjNIelB2OEdDTmhVanJpZldsLWxT
OC5jZXIwDgYDVR0PAQH/BAQDAgeAMIGoBggrBgEFBQcBCwSBmzCBmDBfBggrBgEFBQcwC4ZT
cnN5bmM6Ly9jaGxvZS5zb2Jvcm5vc3QubmV0L3Jwa2kvUklQRS1ubGpvYnNuaWpkZXJzL285
bGtKRmRKdTIzVnF4OHVndzR6cHNVVWJvOC5yb2EwNQYIKwYBBQUHMA2GKWh0dHBzOi8vY2hs
b2Uuc29ib3Jub3N0Lm5ldC9ycGtpL25ld3MueG1sMCsGCCsGAQUFBwEHAQH/BBwwGjAYBAIA
AjASAwcAIAEGfCCMAwcAKg6yQAAAMA0GCSqGSIb3DQEBCwUAA4IBAQAY4bd+Y1Os1MbxGWLU
d7rNVG0c3e0FOwtUOE/Qprt5gkCHO2L19/R1jnXlAaJPID5VhUNl2y/AiwmP47vhk+fvtEdB
wniszL8wCk5b6wwufn1z5/stQ85GRmsqJw5nkOYCyWpTP8k+TUa4w32xNj1dX78FwadDVeSP
yMgJ0860mkXbV1/82/D60zrWQsVAZiYebhni1QAqmpsxZwdZceFRRVY48YDPOZ73ZBZvf0g6
Boy1+djlcAkugA92OKLzqjHWfY2iWZkcxXmFDthoeVCGQePkHMOigOyjZPcM8EXumo1rwI7N
4CPs0VkmCVCZABYVQ0HJvU08i/Wf6X1VRbNcMYIBqjCCAaYCAQOAFKPZZCRXSbtt1asfLoMO
M6bFFG6PMAsGCWCGSAFlAwQCAaBrMBoGCSqGSIb3DQEJAzENBgsqhkiG9w0BCRABGDAcBgkq
hkiG9w0BCQUxDxcNMjIwNjE3MDAyNDIyWjAvBgkqhkiG9w0BCQQxIgQgyCDmNy5kR2T3NpBX
fNhzFLNQv4PmI8kFb6VIt1kqeRswDQYJKoZIhvcNAQEBBQAEggEAWu1sxXCO/X8voU1zfvL+
My6KXb5va2CIuKD4dn/cllClWp8YizygIb+tPWfsT6DvaLOp1jE0raQyc8nUexLXSlIBGF7j
GVWYCy4Oo8mXki+YB3AP1eXiBpx8E4Aa3Rq6/FO80fqrVmUTuywGnv9m6zSIrzEPFujpRIDa
QQfDEOktRcLvNPXHfipTBzR4VSLkbZbyJBdigEPFUJVIRcAoI4tZAUVcbwANrHpZElFMBgr6
Rpn9l5nu7kUlZqXbV39Mfv8WCzctaUyc+Ag311sfWu5s6XaX3PtT9V4TnQhbSWcvR9NgM+As
NqelVbdJ/iA2SeNHU/65xf6dDE2zdHDfsw==
```


The object in this appendix has the following properties:

```
Object
SHA256 hash:
13afbad09ed59b315efd8722d38b09fd02962e376e4def32247f9de905649b47
Size:                      1807 octets

CMS signing time:          Fri 17 Jun 2022 00:24:22 +0000

X.509 end-entity certificate
Subject key id:            A3D964245749BB6DD5AB1F2E830E33A6C5146E8F
Authority key id:          38E14F92FDC7CCFBFC182361523AE27D697E952F
Issuer:                    /CN=38e14f92fdc7ccfbfc182361523ae27d697e952f
Serial:                    86F9
Not before:                Fri 17 Jun 2022 00:24:22 +0000
Not after:                 Sat 01 Jul 2023 00:00:00 +0000
IP address delegation:     2001:67c:208c::/48, 2a0e:b240::/48

eContent
asID:                      15562
addresses:                 2001:67c:208c::/48, 2a0e:b240::/48
```


Acknowledgements

The authors wish to thank Theo Buehler, Ties de Kock, Martin Hoffmann, Charles Gardiner, Russ Housley, Jeffrey Haas, and Bob Beck for their help and contributions. Additionally, the authors thank Jim Fenton, Vijay Gurbani, Haoyu Song, Rob Austein, Roque Gagliano, Danny McPherson, Sam Weiler, Jasdip Singh, and Murray S. Kucherawy for their careful reviews and helpful comments.

Authors' Addresses

Job Snijders
Fastly
Amsterdam
Netherlands
Email: job@fastly.com

Ben Maddison
Workonline
Cape Town
South Africa
Email: benm@workonline.africa

Matthew Lepinski
Carleton College
Email: mlepinski@carleton.edu

Derrick Kong
Raytheon
Email: derrick.kong@raytheon.com

Stephen Kent
Independent
Email: kent@alum.mit.edu