Stream:	Internet Engineering Task Force (IETF)
RFC:	9925
Updates:	5280
Category:	Standards Track
Published:	February 2026
ISSN:	2070-1721
Author:	D. Benjamin
	Google LLC

RFC 9925
Unsigned X.509 Certificates

Abstract

This document defines a placeholder X.509 signature algorithm that may be used in contexts where the consumer of the certificate is not expected to verify the signature. As part of this, it updates RFC 5280.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9925.

Table of Contents

1. Introduction

An X.509 certificate [RFC5280] relates two entities in the PKI: information about a subject and a proof from an issuer. Viewing the PKI as a graph with entities as nodes, as in [RFC4158], a certificate is an edge between the subject and issuer.

In some contexts, an application needs standalone subject information instead of a certificate. In the graph model, the application needs a node, not an edge. For example, certification path validation (Section 6 of [RFC5280]) begins at a trust anchor or root certification authority (root CA). The application trusts this trust anchor information out-of-band and does not require an issuer's signature.

X.509 does not define a structure for this scenario. Instead, X.509 trust anchors are often represented with "self-signed" certificates, where the subject's key signs over itself. Other formats, such as [RFC5914], exist to convey trust anchors, but self-signed certificates remain widely used.

Additionally, some TLS [RFC8446] server deployments use self-signed end entity certificates when they do not intend to present a CA-issued identity, instead expecting the relying party to authenticate the certificate out-of-band, e.g., via a known fingerprint.

These self-signatures typically have no security value, aren't checked by the receiver, and only serve as placeholders to meet syntactic requirements of an X.509 certificate.

Computing signatures as placeholders has some drawbacks:

- Post-quantum signature algorithms are large, so including a self-signature significantly increases the size of the payload.
- If the subject is an end entity, rather than a CA, computing an X.509 signature risks cross-protocol attacks with the intended use of the key.
- It is ambiguous whether such a self-signature requires the CA bit in basic constraints or keyCertSign in key usage. If the key is intended for a non-X.509 use, asserting those capabilities is an unnecessary risk.
- If the subject is an end entity, and the end entity's key is not a signing key (e.g., a Key Encapsulation Mechanism (KEM) key), there is no valid signature algorithm to use with the key.

This document defines a profile for unsigned X.509 certificates, which may be used when the certificate is used as a container for subject information, without any specific issuer.

2. 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.

3. Constructing Unsigned Certificates

This section describes how a sender constructs an unsigned certificate.

3.1. Signature

To construct an unsigned X.509 certificate, the sender **MUST** set the Certificate's signatureAlgorithm and TBSCertificate's signature fields each to an AlgorithmIdentifier with algorithm id-alg-unsigned, defined below:

```
id-alg-unsigned OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 6 36}
```

The parameters for id-alg-unsigned **MUST** be omitted. The Certificate's signatureValue field **MUST** be a BIT STRING of length zero.

3.2. Issuer

An unsigned certificate takes the place of a self-signed certificate in scenarios where the application only requires subject information. It has no issuer, so some requirements in the profile defined in [RFC5280] cannot meaningfully be applied. However, the application may have pre-existing requirements derived from [X.509] and [RFC5280], so senders **MAY** construct the certificate as if it were a self-signed certificate, if needed for interoperability.

In particular, the following fields describe a certificate's issuer:

- issuer (Section 4.1.2.4 of [RFC5280])
- issuerUniqueID (Section 4.1.2.8 of [RFC5280])

The issuer field is not optional, and both [X.509] and Section 4.1.2.4 of [RFC5280] forbid empty issuers, so such a value may not be interoperable with existing applications.

If the subject is not empty, senders **MAY** set the issuer to the subject, similar to how they would construct a self-signed certificate. This may be useful in applications that, for example, expect trust anchors to have a matching issuer and subject. This is, however, a placeholder value. The unsigned certificate is not considered self-signed or self-issued.

Senders **MAY** alternatively use a short placeholder issuer consisting of a single relative distinguished name, with a single attribute of type id-rdna-unsigned and value a zero-length UTF8String. id-rdna-unsigned is defined as follows:

```
id-rdna-unsigned OBJECT IDENTIFIER ::= {1 3 6 1 5 5 7 25 1}
```

This placeholder name, in the string representation of [RFC4514], is:

```
1.3.6.1.5.5.7.25.1=#0C00
```

Senders **MUST** omit the issuerUniqueID field, as it is optional, not applicable, and already forbidden by Section 4.1.2.8 of [RFC5280].

3.3. Extensions

Some X.509 extensions also describe the certificate issuer and thus are not meaningful for an unsigned certificate:

- authority key identifier (Section 4.2.1.1 of [RFC5280])

- issuer alternative name (Section 4.2.1.7 of [RFC5280])

Senders **SHOULD** omit the authority key identifier and issuer alternative name extensions. Section 4.2.1.1 of [RFC5280] requires certificates to include the authority key identifier, but includes an exception for self-signed certificates used when distributing a public key. This document updates [RFC5280] to also permit omitting the authority key identifier in unsigned certificates.

Some extensions reflect whether the subject is a CA or an end entity:

- key usage (Section 4.2.1.3 of [RFC5280])
- basic constraints (Section 4.2.1.9 of [RFC5280])

Senders **SHOULD** fill in these values to reflect the subject. That is:

- If the subject is a CA, it **SHOULD** assert the keyCertSign key usage bit and **SHOULD** include a basic constraints extension that sets the cA boolean to TRUE.
- If the subject is an end entity, it **SHOULD NOT** assert the keyCertSign key usage bit, and it **SHOULD** either omit the basic constraints extension or set the cA boolean to FALSE. Unlike a self-signed certificate, an unsigned certificate does not issue itself, so there is no need to accommodate a self-signature in either extension.

4. Consuming Unsigned Certificates

X.509 signatures of type id-alg-unsigned are always invalid:

- When processing X.509 certificates without verifying signatures, receivers **MAY** accept id-alg-unsigned.
- When verifying X.509 signatures, receivers **MUST** reject id-alg-unsigned.

In particular, X.509 validators **MUST NOT** accept id-alg-unsigned in the place of a signature in the certification path.

It is expected that most unmodified X.509 applications will already be compliant with this guidance. X.509 applications are thus **RECOMMENDED** to satisfy these requirements by ignoring this document and instead treating id-alg-unsigned as the same as an unrecognized signature algorithm. An unmodified X.509 validator will be unable to verify the signature (Step (a.1) of Section 6.1.3 of [RFC5280]) and thus reject the certification path. Conversely, in contexts where an X.509 application was ignoring the self-signature, id-alg-unsigned will also be ignored but more efficiently.

In other contexts, an application may require modifications or limit itself to particular forms of unsigned certificates. For example, an application might check self-signedness to classify locally configured certificates as trust anchors or untrusted intermediates. Such an application may need to modify its configuration model or user interface before using an unsigned certificate as a trust anchor.

5. Security Considerations

It is best practice to limit cryptographic keys to a single purpose each. If a key is reused across contexts, applications risk cross-protocol attacks when the two uses collide. However, in applications that use self-signed end entity certificates, the subject's key is necessarily used in two ways: the X.509 self-signature and the end entity protocol. Unsigned certificates fix this key reuse by removing the X.509 self-signature.

If an application accepts id-alg-unsigned as part of a certification path, or in any other context where it is necessary to verify the X.509 signature, the signature check would be bypassed. Thus, Section 4 prohibits this and recommends that applications treat id-alg-unsigned the same as any other previously unrecognized signature algorithm. Non-compliant applications risk vulnerabilities analogous to those described in [JWT] and Section 1.1 of [JOSE].

The signature in a self-signed certificate is self-derived and thus of limited use to convey trust. However, some applications might use it as an integrity check to guard against accidental storage corruption, etc. An unsigned certificate does not provide any integrity check. Applications checking self-signature for integrity **SHOULD** instead use some other mechanism, such as an external hash that is verified out-of-band.

6. IANA Considerations

6.1. Module Identifier

IANA has added the following entry in the "SMI Security for PKIX Module Identifier" registry, defined by [RFC7299]:

Decimal	Description	Reference
122	id-mod-algUnsigned-2025	RFC 9925

Table 1

6.2. Algorithm

IANA has added the following entry to the "SMI Security for PKIX Algorithms" registry [RFC7299]:

Decimal	Description	Reference
36	id-alg-unsigned	RFC 9925

Table 2

6.3. Relative Distinguished Name Attribute

To allocate id-rdna-unsigned, this document introduces a new PKIX OID arc for relative distinguished name attributes:

IANA has added the following entry to the "SMI Security for PKIX" registry [RFC7299]:

Decimal	Description	Reference
25	Relative Distinguished Name Attribute	RFC 9925

Table 3

IANA has created the "SMI Security for PKIX Relative Distinguished Name Attribute" registry within the "Structure of Management Information (SMI) Numbers (MIB Module Registrations)" registry group.

The new registry's description is "iso.org.dod.internet.security.mechanisms.pkix.rdna (1.3.6.1.5.5.7.25)".

The new registry has three columns and is initialized with the following values:

Decimal	Description	Reference
1	id-rdna-unsigned	RFC 9925

Table 4

Future updates to this table are to be made according to the Specification Required policy as defined in [RFC8126].

7. References

7.1. 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>.

[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>.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <https://www.rfc-editor.org/info/rfc5912>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

7.2. Informative References

[JOSE] Madden, N., "JOSE: Deprecate 'none' and 'RSA1_5'", Work in Progress, Internet-Draft, draft-ietf-jose-deprecate-none-rsa15-03, 19 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-jose-deprecate-none-rsa15-03>.

[JWT] Sanderson, J., "How Many Days Has It Been Since a JWT alg:none Vulnerability?", <https://www.howmanydayssinceajwtalgnonevuln.com/>.

[RFC4158] Cooper, M., Dzambasow, Y., Hesse, P., Joseph, S., and R. Nicholas, "Internet X.509 Public Key Infrastructure: Certification Path Building", RFC 4158, DOI 10.17487/RFC4158, September 2005, <https://www.rfc-editor.org/info/rfc4158>.

[RFC4514] Zeilenga, K., Ed., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", RFC 4514, DOI 10.17487/RFC4514, June 2006, <https://www.rfc-editor.org/info/rfc4514>.

[RFC5914] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Format", RFC 5914, DOI 10.17487/RFC5914, June 2010, <https://www.rfc-editor.org/info/rfc5914>.

[RFC7299] Housley, R., "Object Identifier Registry for the PKIX Working Group", RFC 7299, DOI 10.17487/RFC7299, July 2014, <https://www.rfc-editor.org/info/rfc7299>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[X.509] ITU-T, "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", ITU-T Recommendation X.509, ISO/IEC 9594-8:2020, October 2019, <https://www.itu.int/rec/t-rec-x.509/en>.

Appendix A. ASN.1 Module

```
SignatureAlgorithmNone
  { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-algUnsigned-2025(122) }

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

IMPORTS
  SIGNATURE-ALGORITHM
  FROM AlgorithmInformation-2009  -- in [RFC5912]
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58) }
  ATTRIBUTE
  FROM PKIX-CommonTypes-2009 -- in [RFC5912]
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkixCommon-02(57) } ;

-- Unsigned Signature Algorithm

id-alg-unsigned OBJECT IDENTIFIER ::= { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) alg(6) 36 }

sa-unsigned SIGNATURE-ALGORITHM ::= {
   IDENTIFIER id-alg-unsigned
   PARAMS ARE absent
}

id-rdna-unsigned OBJECT IDENTIFIER ::= { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) rdna(25) 1 }

at-unsigned ATTRIBUTE ::= {
   TYPE UTF8String (SIZE (0))
   IDENTIFIED BY id-rdna-unsigned
}

END
```

Acknowledgements

Thanks to Bob Beck, Nick Harper, and Sophie Schmieg for reviewing an early iteration of this document. Thanks to Alex Gaynor for providing a link to cite for [JWT]. Thanks to Russ Housley for additional input.

Author's Address

David Benjamin
Google LLC
Email: davidben@google.com