Network Working Group   R. Housley
Internet-Draft   Vigil Security
Intended status: Informational   March 02, 2018
Expires: September 3, 2018

Hash Of Root Key Certificate Extension
draft-housley-hash-of-root-key-cert-extn-00

Abstract

This document specifies a certificate extension that is carried in the self-signed certificate for a trust anchor, which is often called a Root Certification Authority (CA) certificate, to identify the next public key that will be used by the trust anchor.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies the Hash Of Root Key X.509 version 3 certificate extension. The extension is an optional addition to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [RFC5280]. The certificate extension facilitates the orderly transition from one Root Certification Authority (CA) public key to the next. It does so by publishing the hash value of the next generation public key in the current self-signed certificate. This allows a relying party to unambiguously recognize the next generation public key when it becomes available.

A Root CA Certificate MAY include the Hashed Root Key certificate extension to provide the hash value of the next public key that will be used by the Root CA.

1.1. Terminology

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. ASN.1

Certificates [RFC5280] are generated using ASN.1 [X680], which uses the Basic Encoding Rules (BER) and the Distinguished Encoding Rules (DER) [X690].

2. Overview

Before the initial deployment of the Root CA, the following are generated:

```
R1 = The initial Root key pair
C1 = Self-signed certificate for R1, which also contains H2
R2 = The second generation Root key pair
H2 = Thumbprint (hash) of the public key of R2
```

C1 is a self-signed certificate, and it contains H2 within the hashOfRootKey extension. C1 is distributed as part of the initial the system deployment. The hashOfRootKey certificate extension is described in Section 3.

When the time comes to replace the initial Root CA certificate, R1, the following are generated:

```
R3 = The third generation Root key pair
H3 = Thumbprint (hash) the public key of R3
C2 = Self-signed certificate for R2, which contains H3
```

This is an iterative process. That is, R4 and H4 are generated when it is time for C3 to replace C2. And so on.

The successors to the Root CA self-signed certificate can be delivered by any means. Whenever a new Root CA certificate is received, the recipient is able to verify that the potential Root CA certificate chains back to a previously authenticated Root CA certificate with the hashOfRootKey certificate extension. That is, validate the self-signed signature and verify that the hash of the DER-encoded SubjectPublicKeyInfo from the potential Root CA certificate matches the value from the hashOfRootKey certificate extension of the current Root CA certificate. If the signature does not validate or the hash values do not match, then potential Root CA certificate is not a valid replacement, and the recipient continues to use the current Root CA certificate.

3. Hash Of Root Key Certificate Extension

The HashOfRootKey certificate extension MUST NOT be critical.

The following ASN.1 [X680][X690] syntax defines the HashOfRootKey certificate extension:

```
   ext-HashOfRootKey EXTENSION ::= {    -- Only in Root CA certificates
      SYNTAX         HashedRootKey
      IDENTIFIED BY  id-ce-hashOfRootKey
      CRITICALITY    {FALSE} }

   HashedRootKey ::= SEQUENCE {
      hashAlg        AlgorithmIdentifier,  -- Hash algorithm used
      hashValue      OCTET STRING }        -- Hash of DER-encoded
                                           --   SubjectPublicKeyInfo

   id-ce-hashOfRootKey  ::=  OBJECT IDENTIFIER { 1 3 6 1 4 1 TBD 2 1 }
```

The definitions of EXTENSION and HashAlgorithm can be found in [RFC5912].

The hashAlg indicates the one-way hash algorithm that was used to compute the hash value.

The hashValue contains the hash value computed from the next generation public key. The public key is DER-encoded SubjectPublicKeyInfo as defined in [RFC5280].

4. IANA Considerations

This document makes no requests of the IANA.

5. Security Considerations

The security considerations from [RFC5280] apply, especially the discussion of self-issued certificates.

The Hash Of Root Key certificate extension facilitates the orderly transition from one Root CA public key to the next by publishing the hash value of the next generation public key in the current certificate. This allows a relying party to unambiguously recognize the next generation public key when it becomes available; however, the full public key is not disclosed until the Root CA releases the next generation certificate. In this way, attackers cannot begin to analyze the public key before the next generation Root CA certificate is released.

6. Acknowledgements

In the Secure Electronic Transaction (SET) [SET] specification published by MasterCard and VISA in 1997, a very similar certificate extension is described. The SET certificate extension has essentially the same semantics, but the syntax fairly different.

CTIA - The Wireless Association is developing a public key infrastructure that will make use of the certificate extension described in this document.

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>.

[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.

7.2. Informative References

[SET] MasterCard and VISA, "SET Secure Electronic Transaction Specification -- Book 2: Programmer's Guide, Version 1.0", May 1997.

Appendix A. ASN.1 Module

The following ASN.1 module provides the complete definition of the HashOfRootKey certificate extension.

```
   HashedRootKeyCertExtn { 1 3 6 1 4 1 51483 0 1 }

   DEFINITIONS IMPLICIT TAGS ::=
   BEGIN

   -- EXPORTS All

   IMPORTS

   AlgorithmIdentifier{}, DIGEST-ALGORITHM
     FROM AlgorithmInformation-2009  -- [RFC5912]
       { iso(1) identified-organization(3) dod(6) internet(1)
         security(5) mechanisms(5) pkix(7) id-mod(0)
         id-mod-algorithmInformation-02(58) }

   EXTENSION
     FROM PKIX-CommonTypes-2009
       { iso(1) identified-organization(3) dod(6) internet(1)
         security(5) mechanisms(5) pkix(7) id-mod(0)
         id-mod-pkixCommon-02(57) } ;

   --
   -- Expand the certificate extensions list in [RFC5912]
   --

   CertExtensions EXTENSION ::= {
      ext-HashOfRootKey, ... }

   --
   -- HashOfRootKey Certificate Extension
   --

   ext-HashOfRootKey EXTENSION ::= {    -- Only in Root CA certificates
      SYNTAX         HashedRootKey
      IDENTIFIED BY  id-ce-hashOfRootKey
      CRITICALITY    {FALSE} }

   HashedRootKey  ::=  SEQUENCE {
      hashAlg        HashAlgorithmId,   -- Hash algorithm used
      hashValue      OCTET STRING }     -- Hash of DER-encoded
                                        --   SubjectPublicKeyInfo

   HashAlgorithmId  ::=  AlgorithmIdentifier {DIGEST-ALGORITHM,{ ... }}

   id-ce-hashOfRootKey OBJECT IDENTIFIER  ::=  { 1 3 6 1 4 1 51483 2 1 }

   END
```

Author's Address

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

Email: housley@vigilsec.com