Network Working Group
Internet-Draft
Updates: 4211 (if approved)
Intended status: Standards Track
Expires: 23 August 2021

R. Housley
Vigil Security
19 February 2021

Algorithm Requirements Update to the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)
draft-ietf-lamps-crmf-update-algs-04

Abstract

This document updates the cryptographic algorithm requirements for the Password-Based Message Authentication Code in the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF) specified in RFC 4211.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document updates the cryptographic algorithm requirements for the Password-Based Message Authentication Code (MAC) in the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF) [RFC4211]. The algorithms specified in [RFC4211] were appropriate in 2005; however, these algorithms are no longer considered the best choices. This update specifies algorithms that are more appropriate today.

2. 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.

3. Signature Key POP

Section 4.1 of [RFC4211] specifies the Proof-of-Possession (POP) processing. This section is updated to explicitly allow the use of the PBMAC1 algorithm presented in Section 7.1 of [RFC8018].

OLD:

> algId identifies the algorithm used to compute the MAC value. All implementations MUST support id-PasswordBasedMAC. The details on this algorithm are presented in section 4.4

NEW:

> algId identifies the algorithm used to compute the MAC value. All implementations MUST support id-PasswordBasedMAC as presented in Section 4.4 of this document. Implementations MAY also support PBMAC1 presented in Section 7.1 of [RFC8018].

4. Password-Based Message Authentication Code

Section 4.4 of [RFC4211] specifies a Password-Based MAC that relies on a one-way function to compute a symmetric key from the password and a MAC algorithm. This section specifies algorithm requirements for the one-way function and the MAC algorithm.

4.1. Introduction Paragraph

Add guidance about limiting the use of the password.

OLD:

> This MAC algorithm was designed to take a shared secret (a password) and use it to compute a check value over a piece of information. The assumption is that, without the password, the correct check value cannot be computed. The algorithm computes the one-way function multiple times in order to slow down any dictionary attacks against the password value.

NEW:

> This MAC algorithm was designed to take a shared secret (a password) and use it to compute a check value over a piece of information. The assumption is that, without the password, the correct check value cannot be computed. The algorithm computes the one-way function multiple times in order to slow down any dictionary attacks against the password value. The password used to compute this MAC SHOULD NOT be used for any other purpose.

4.2. One-Way Function

Change the paragraph describing the "owf" as follows:

OLD:

> owf identifies the algorithm and associated parameters used to compute the key used in the MAC process. All implementations MUST support SHA-1.

NEW:

owf identifies the algorithm and associated parameters used to compute the key used in the MAC process. All implementations MUST support SHA-256 [SHS].

4.3. Iteration Count

Update the guidance on appropriate iteration count values.

OLD:

iterationCount identifies the number of times the hash is applied during the key computation process. The iterationCount MUST be a minimum of 100. Many people suggest using values as high as 1000 iterations as the minimum value. The trade off here is between protection of the password from attacks and the time spent by the server processing all of the different iterations in deriving passwords. Hashing is generally considered a cheap operation but this may not be true with all hash functions in the future.

NEW:

iterationCount identifies the number of times the hash is applied during the key computation process. The iterationCount MUST be a minimum of 100; however, the iterationCount SHOULD be as large as server performance will allow, typically at least 10,000 [DIGALM]. There is a trade off between protection of the password from attacks and the time spent by the server processing the iterations. As part of that tradeoff, an iteration count smaller than 10,000 can be used when automated generation produces shared secrets with high entropy.

4.4. MAC Algorithm

Change the paragraph describing the "mac" as follows:

OLD:

mac identifies the algorithm and associated parameters of the MAC function to be used. All implementations MUST support HMAC-SHA1 [HMAC]. All implementations SHOULD support DES-MAC and Triple-DES-MAC [PKCS11].

NEW:

mac identifies the algorithm and associated parameters of the MAC function to be used. All implementations MUST support HMAC-SHA256 [HMAC]. All implementations SHOULD support AES-GMAC AES [GMAC] with a 128 bit key.

For convenience, the identifiers for these two algorithms are repeated here.

The algorithm identifier for HMAC-SHA256 is defined in [RFC4231]:

```
   id-hmacWithSHA256 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
      us(840) rsadsi(113549) digestAlgorithm(2) 9 }
```

When this algorithm identifier is used, the parameters SHOULD be present. When present, the parameters MUST contain a type of NULL.

The algorithm identifier for AES-GMAC [AES][GMAC] with a 128-bit key is defined in [I-D.ietf-lamps-cms-aes-gmac-alg]:

```
   id-aes128-GMAC OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
      country(16) us(840) organization(1) gov(101) csor(3)
      nistAlgorithm(4) aes(1) 9 }
```

When this algorithm identifier is used, the parameters MUST be present, and the parameters MUST contain the GMACParameters structure as follows:

```
   GMACParameters ::= SEQUENCE {
      nonce        OCTET STRING,
      length       MACLength DEFAULT 12 }

   MACLength ::= INTEGER (12 | 13 | 14 | 15 | 16)
```

The GMACParameters nonce parameter is the GMAC initialization vector. The nonce may have any number of bits between 8 and (2^64)-1, but it MUST be a multiple of 8 bits. Within the scope of any GMAC key, the nonce value MUST be unique. A nonce value of 12 octets can be processed more efficiently, so that length for the nonce value is RECOMMENDED.

The GMACParameters length parameter field tells the size of the message authentication code in octets. GMAC supports lengths between 12 and 16 octets, inclusive. However, for use with CRMF, the maximum length of 16 octets MUST be used.

5. IANA Considerations

This document makes no requests of the IANA.

6. Security Considerations

The security of the password-based MAC relies on the number of times the hash function is applied as well as the entropy of the shared secret (the password). Hardware support for hash calculation is available at very low cost [PHS], which reduces the protection provided by a high iterationCount value. Therefore, the entropy of the password is crucial for the security of the password-based MAC function. In 2010, researchers showed that about half of the real-world passwords can be broken with less than 150 million trials, indicating a median entropy of only 27 bits [DMR]. Higher entropy can be achieved by using randomly generated strings. For example, assuming an alphabet of 60 characters a randomly chosen password with 10 characters offers 59 bits a entropy, and 20 characters offers 118 bits of entropy. Using a one-time password also increases the security of the MAC, assuming that the integrity-protected transaction will complete before the attacker is able to learn the password with an offline attack.

Cryptographic algorithms age; they become weaker with time. As new cryptanalysis techniques are developed and computing capabilities improve, the work required to break a particular cryptographic algorithm will reduce, making an attack on the algorithm more feasible for more attackers. While it is unknown how cryptoanalytic attacks will evolve, it is certain that they will get better. It is unknown how much better they will become or when the advances will happen. For this reason, the algorithm requirements for CRMF are updated by this specification.

When a Password-Based MAC is used, implementations must protect the password and the MAC key. Compromise of either the password or the MAC key may result in the ability of an attacker to undermine authentication.

7. Acknowledgements

Many thanks to Hans Aschauer, Hendrik Brockhaus, Quynh Dang, Roman Danyliw, Tomas Gustavsson, Jonathan Hammell, Tim Hollebeek, Lijun Liao, Mike Ounsworth, Tim Polk, Mike StJohns, and Sean Turner for their careful review and improvements.

8. References

8.1. Normative References

[AES]   National Institute of Standards and Technology, "Advanced encryption standard (AES)", DOI 10.6028/nist.fips.197, November 2001, <https://doi.org/10.6028/nist.fips.197>.

[GMAC] National Institute of Standards and Technology, "Recommendation for block cipher modes of operation: Galois Counter Mode (GCM) and GMAC", DOI 10.6028/nist.sp.800-38d, 2007, <https://doi.org/10.6028/nist.sp.800-38d>.

[HMAC] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/rfc/rfc2104>.

[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>.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, DOI 10.17487/RFC4211, September 2005, <https://www.rfc-editor.org/info/rfc4211>.

[RFC8018] Moriarty, K., Ed., Kaliski, B., and A. Rusch, "PKCS #5: Password-Based Cryptography Specification Version 2.1", RFC 8018, DOI 10.17487/RFC8018, January 2017, <https://www.rfc-editor.org/info/rfc8018>.

[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>.

[SHS] National Institute of Standards and Technology, "Secure Hash Standard", DOI 10.6028/nist.fips.180-4, July 2015, <https://doi.org/10.6028/nist.fips.180-4>.

8.2. Informative References

[DIGALM] National Institute of Standards and Technology, "Digital identity guidelines: authentication and lifecycle management", DOI 10.6028/nist.sp.800-63b, June 2017, <https://doi.org/10.6028/nist.sp.800-63b>.

[DMR] Dell'Amico, M., Michiardi, P., and Y. Roudier, "Password Strength: An Empirical Analysis", DOI 10.1109/INFCOM.2010.5461951, March 2010, <https://doi.org/10.1109/INFCOM.2010.5461951>.

[I-D.ietf-lamps-cms-aes-gmac-alg]
Housley, R., "Using the AES-GMAC Algorithm with the Cryptographic Message Syntax (CMS)", Work in Progress, Internet-Draft, draft-ietf-lamps-cms-aes-gmac-alg-02, 30 December 2020, <http://www.ietf.org/internet-drafts/draft-ietf-lamps-cms-aes-gmac-alg-02.txt>.

[PHS] Pathirana, A., Halgamuge, M., and A. Syed, "Energy efficient bitcoin mining to maximize the mining profit: Using data from 119 bitcoin mining hardware setups", International Conference on Advances in Business Management and Information Technology, pp 1-14, November 2019.

[PKCS11] RSA Laboratories, "The Public-Key Cryptography Standards - PKCS #11 v2.11: Cryptographic Token Interface Standard", June 2001.

[RFC4231] Nystrom, M., "Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512", RFC 4231, DOI 10.17487/RFC4231, December 2005, <https://www.rfc-editor.org/info/rfc4231>.

Author's Address

Russ Housley
Vigil Security, LLC
516 Dranesville Road
Herndon, VA, 20170
United States of America

Email: housley@vigilsec.com