sshm
Internet-Draft
Intended status: Informational
Expires: 12 February 2026

S. Fluhrer
Cisco Systems

11 August 2025

SSH Support of ML-DSA
draft-sfluhrer-ssh-mldsa-04

Abstract

This document describes the use of ML-DSA digital signatures in the Secure Shell (SSH) protocol.

About This Document

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

The latest revision of this draft can be found at https://sfluhrer.github.io/ssh-mldsa/draft-sfluhrer-ssh-mldsa.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-sfluhrer-ssh-mldsa/.

Discussion of this document takes place on the Secure Shell Maintenance Security Area mailing list (mailto:ssh@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ssh/. Subscribe at https://www.ietf.org/mailman/listinfo/ssh/.

Source for this draft and an issue tracker can be found at https://github.com/sfluhrer/ssh-mldsa.

Status of This Memo

Table of Contents

1. Introduction

A Cryptographically Relevant Quantum Computer (CRQC) could break traditional asymmetric cryptograph algorithms: e.g RSA, ECDSA; which are widely deployed authentication options of SSH. NIST has recently published the postquantum digital signature algorithm ML-DSA [FIPS204].

This document describes how to use this algorithm for authentication within SSH [RFC4251], as a replacement for the traditional signature algorithms (RSA, ECDSA).

1.1. Background on ML-DSA

ML-DSA (as specified in FIPS 204) is a signature algorithm that is believed to be secure against attackers who have a Quantum Computer available to them. There are three strengths defined for it (with the parameter sets being known as ML-DSA-44, ML-DSA-65 and ML-DSA-87). In addition, for each defined parameter set, there are two versions, the 'pure' version (where ML-DSA directly signs the message) and a 'prehashed' version (where ML-DSA signs a hash that was computed outside of ML-DSA). For this protocol, we will always use the pure version.

In addition, ML-DSA also has a 'context' input, which is a short string that is common to the sender and the recceiver. It is intended to allow for domain separation between separate uses of the same public key. This protocol always uses an empty (zero length) context.

FIPS 204 also allows ML-DSA to be run in either determanistic or 'hedged' mode (where randomness is applied to the signature operation). We place no requirement on which is used; the implementation should select based on the quality of their random number source.

2. Conventions and Definitions

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.

The descriptions of key and signature formats use the notation introduced in [RFC4251], Section 3, and the string data type from [RFC4251], Section 5. Identifiers and terminology from ML-DSA [FIPS204] are used throughout the document.

3. Public Key Algorithms

This document describes three public key algorithms for use with SSH, as per [RFC4253], Section 6.6, corresponding to the three parameter sets of ML-DSA. The names of the algorithm are "ssh-mldsa-44", "ssh-mldsa-65" and "ssh-mldsa-87", to match the level 2, 3 and 5 parameter sets [FIPS204]. These algorithm only support signing and not encryption.

The below table lists the public key sizes and the signature size (in bytes) for the three parameter sets.

Public Key Algorithm Name	Public Key Size	Signature Size
ssh-mldsa-44	1312	2420
ssh-mldsa-65	1952	3309
ssh-mldsa-87	2592	4627

Table 1

4. Public Key Format

The key format for all three parameter sets have the following encoding:

string "ssh-mldsa-44" (or "ssh-mldsa-65" or "ssh-mldsa-87")

string key

Here, ’key’ is the public key described in [FIPS204].

Signature Algorithm

Signatures are generated according to the procedure in Section 5.2 [FIPS204], using the "pure" version of ML-DSA, with an empty context string.

5. Signature Format

The "ssh-mldsa" key format has the following encoding:

string "ssh-mldsa-44" (or "ssh-mldsa-65" or "ssh-mldsa-87")

string signature

Here, ’signature’ is the signature produced in accordance with the previous section.

6. Verification Algorithm

Signatures are verified according to the procedure in [FIPS204], Section 5.3, using the "pure" version of ML-DSA, with an empty context strong.

7. SSHFP DNS Resource Records

Usage and generation of the SSHFP DNS resource record is described in [RFC4255]. This section illustrates the generation of SSHFP resource records for ML-DSA keys, and this document also specifies the corresponding code point to "SSHFP RR Types for public key algorithms" in the "DNS SSHFP Resource Record Parameters" IANA registry [IANA-SSHFP].

The generation of SSHFP resource records keys for ML-DSA is described as follows.

The encoding of ML-DSA public keys is described in [FIPS204].

The SSHFP Resource Record for an ML-DSA key fingerprint (with a SHA-256 fingerprint) would, for example, be:

pqserver.example.com. IN SSHFP TBD 2 (
a87f1b687ac0e57d2a081a2f28267237 34d90ed316d2b818ca9580ea384d9240)

Replace TBD with the value eventually allocated by IANA.

8. IANA Considerations

This document augments the Public Key Algorithm Names in [RFC4250], Section 4.11.3.

IANA is requested to add the following entries to "Public Key Algorithm Names" in the "Secure Shell (SSH) Protocol Parameters" registry [IANA-SSH]:

Public Key Algorithm Name	Reference
ssh-mldsa-44	THIS-RFC
ssh-mldsa-65	THIS-RFC
ssh-mldsa-87	THIS-RFC

Table 2

IANA is requested to add the following entries to "SSHFP RR Types for public key algorithms" in the "DNS SSHFP Resource Record Parameters" registry [IANA-SSHFP]:

Value	Description	Reference
TBD1	ML-DSA-44	THIS RFC
TBD2	ML-DSA-65	THIS RFC
TBD3	ML-DSA-87	THIS RFC

Table 3

9. Security Considerations

The security considerations in [RFC4251], Section 9 apply to all SSH implementations, including those using ML-DSA.

The security considerations in ML-DSA [FIPS204] apply to all uses of ML-DSA, including those in SSH.

Cryptographic algorithms and parameters are usually broken or weakened over time. Implementers and users need to continously re-evaluate that cryptographic algorithms continue to provide the expected level of security.

10. References

10.1. Normative References

[FIPS204] "Module-Lattice-Based Digital Signature Standard", NIST FIPS 204, August 2024, <https://doi.org/10.6028/NIST.FIPS.204>.

[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/rfc/rfc2119>.

[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/rfc/rfc8174>.

10.2. Informative References

[IANA-SSH] "Secure Shell (SSH) Protocol Parameters", n.d., <https://www.iana.org/assignments/ssh-parameters>.

[IANA-SSHFP]
"DNS SSHFP Resource Record Parameters", n.d., <https://www.iana.org/assignments/dns-sshfp-rr-parameters)>.

[RFC4250] Lehtinen, S. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Assigned Numbers", RFC 4250, DOI 10.17487/RFC4250, January 2006, <https://www.rfc-editor.org/rfc/rfc4250>.

[RFC4251] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, DOI 10.17487/RFC4251, January 2006, <https://www.rfc-editor.org/rfc/rfc4251>.

[RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <https://www.rfc-editor.org/rfc/rfc4253>.

[RFC4255] Schlyter, J. and W. Griffin, "Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints", RFC 4255, DOI 10.17487/RFC4255, January 2006, <https://www.rfc-editor.org/rfc/rfc4255>.

Acknowledgments

The text of draft-josefsson-ssh-sphincs was used as a template for this document.

Author’s Address

Scott Fluhrer
Cisco Systems
Email: sfluhrer@cisco.com