Limited Additional Mechanisms for PKIX and SMIME
Internet-Draft
Intended status: Standards Track
Expires: 21 November 2026

D. Van Geest
CryptoNext Security
S. Turner
sn3rd

20 May 2026

Use of the FN-DSA Signature Algorithm in the Cryptographic Message Syntax (CMS)
draft-ietf-lamps-cms-fn-dsa-00

Abstract

The Fast-Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm (FN-DSA), as defined by NIST in FIPS 206, is a post-quantum digital signature scheme that aims to be secure against an adversary in possession of a Cryptographically Relevant Quantum Computer (CRQC). This document specifies the conventions for using the FN-DSA signature algorithm with the Cryptographic Message Syntax (CMS). In addition, the algorithm identifier is provided.

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://lamps.github.io/cms-fn-dsa/draft-ietf-lamps-cms-fn-dsa.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-lamps-cms-fn-dsa/.

Discussion of this document takes place on the Limited Additional Mechanisms for PKIX and SMIME Working Group mailing list (mailto:spasm@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/spasm/. Subscribe at https://www.ietf.org/mailman/listinfo/spasm/.

Source for this draft and an issue tracker can be found at https://github.com/lamps-wg/cms-fn-dsa.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

The Fast-Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm (FN-DSA) is a digital signature algorithm standardised by the US National Institute of Standards and Technology (NIST) as part of their post-quantum cryptography standardisation process. It is intended to be secure against both "traditional" cryptographic attacks, as well as attacks utilising a quantum computer. It offers smaller signatures and significantly faster runtimes than SLH-DSA [FIPS205], an alternative post-quantum signature algorithm also standardised by NIST. This document specifies the use of the FN-DSA in the CMS at two security levels: FN-DSA-512 and FN-DSA-1024. See Appendix B of [I-D.ietf-lamps-fn-dsa-certificates] for more information on the security levels and key sizes of FN-DSA.

Prior to standardisation, FN-DSA was known as Falcon. FN-DSA and Falcon are not compatible.

For each of the FN-DSA parameter sets, an algorithm identifier Object Identifier (OID) has been specified.

1.1. 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.

2. FN-DSA Algorithm Identifiers

Many ASN.1 data structure types use the AlgorithmIdentifier type to identify cryptographic algorithms. In the CMS, the AlgorithmIdentifier field is used to identify FN-DSA signatures in the signed-data content type. They may also appear in X.509 certificates used to verify those signatures. The same AlgorithmIdentifier values are used to identify FN-DSA public keys and signature algorithms. [I-D.ietf-lamps-fn-dsa-certificates] describes the use of FN-DSA in X.509 certificates. The AlgorithmIdentifier type is defined as follows:

```
AlgorithmIdentifier{ALGORITHM-TYPE, ALGORITHM-TYPE:AlgorithmSet} ::=
        SEQUENCE {
            algorithm   ALGORITHM-TYPE.&id({AlgorithmSet}),
            parameters  ALGORITHM-TYPE.
                   &Params({AlgorithmSet}{@algorithm}) OPTIONAL
        }
```

| NOTE: The above syntax is from [RFC5911] and is compatible with
| the 2021 ASN.1 syntax [X680]. See [RFC5280] for the 1988 ASN.1
| syntax.

The fields in the AlgorithmIdentifier type have the following meanings:

algorithm: The algorithm field contains an OID that identifies the cryptographic algorithm in use. The OIDs for FN-DSA are described below.

parameters: The parameters field contains parameter information for the algorithm identified by the OID in the algorithm field. Each FN-DSA parameter set is identified by its own algorithm OID, so there is no relevant information to include in this field. As such, parameters MUST be omitted when encoding an FN-DSA AlgorithmIdentifier.

The OIDs for FN-DSA are defined in the NIST Computer Security Objects Register [CSOR], and are reproduced here for convenience.

| TODO: NIST WILL ASSIGN THESE.

```
sigAlgs OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
    us(840) organization(1) gov(101) csor(3) nistAlgorithms(4) 3 }

id-fn-dsa-512 OBJECT IDENTIFIER ::= { sigAlgs TBD }

id-fn-dsa-1024 OBJECT IDENTIFIER ::= { sigAlgs TBD }
```

3. Signed-Data Conventions

3.1. Pure Mode vs Pre-hash Mode

[RFC5652] specifies that digital signatures for CMS are produced using a digest of the message to be signed and the signer’s private key. At the time of publication of that RFC, all signature algorithms supported in the CMS required a message digest to be calculated externally to that algorithm, which would then be supplied to the algorithm implementation when calculating and verifying signatures. Since then, EdDSA [RFC8032], ML-DSA [FIPS20], and SLH-DSA [FIPS205], have also been standardised, and these algorithms support both a "pure" and "pre-hash" mode. In the pre-hash mode, a message digest (the "pre-hash") is calculated separately and supplied to the signature algorithm as described above. In the pure mode, the message to be signed or verified is instead supplied directly to the signature algorithm. When EdDSA [RFC8419], SLH-DSA [RFC9814], and ML-DSA [RFC9882] are used with CMS, only the pure mode of those

algorithms is specified. This is because in most situations, CMS signatures are computed over a set of signed attributes that contain a hash of the content, rather than being computed over the message content itself. Since signed attributes are typically small, use of pre-hash modes in the CMS would not significantly reduce the size of the data to be signed, and hence offers no benefit. This document follows that convention and does not specify the use of FN-DSA's pre-hash mode ("HashFN-DSA") in the CMS.

3.2. Signature Generation and Verification

[RFC5652] describes the two methods that are used to calculate and verify signatures in the CMS. One method is used when signed attributes are present in the signedAttrs field of the relevant SignerInfo, and another is used when signed attributes are absent. Each method produces a different "message digest" to be supplied to the signature algorithm in question, but because the pure mode of FN-DSA is used, the "message digest" is in fact the entire message. Use of signed attributes is preferred, but the conventions for signed-data without signed attributes is also described below for completeness.

When signed attributes are absent, FN-DSA (pure mode) signatures are computed over the content of the signed-data. As described in Section 5.4 of [RFC5652], the "content" of a signed-data is the value of the encapContentInfo eContent OCTET STRING. The tag and length octets are not included.

When signed attributes are included, FN-DSA (pure mode) signatures are computed over the complete DER [X690] encoding of the SignedAttrs value contained in the SignerInfo's signedAttrs field. As described in Section 5.4 of [RFC5652], this encoding includes the tag and length octets, but an EXPLICIT SET OF tag is used rather than the IMPLICIT \[0\] tag that appears in the final message. At a minimum, the signedAttrs field MUST at minimum include a content-type attribute and a message-digest attribute. The message-digest attribute contains a hash of the content of the signed-data, where the content is as described for the absent signed attributes case above. Recalculation of the hash value by the recipient is an important step in signature verification.

Section 4 of [RFC9814] describes how, when the content of a signed-data is large, performance may be improved by including signed attributes. This is as true for FN-DSA as it is for SLH-DSA, although FN-DSA signature generation and verification is significantly faster than SLH-DSA.

FN-DSA has a context string input that can be used to ensure that different signatures are generated for different application contexts. When using FN-DSA as specified in this document, the context string is set to the empty string.

3.3. SignerInfo Content

When using FN-DSA, the fields of a SignerInfo are used as follows:

| TODO: Include text on security strength.

digestAlgorithm: Per Section 5.3 of [RFC5652], the digestAlgorithm field identifies the message digest algorithm used by the signer, and any associated parameters. Each FN-DSA parameter set ...

: Table 1 shows appropriate SHA-2 and SHA-3 digest

algorithms for each parameter set. SHA-512 [FIPS180] MUST be supported for use with the variants of FN-DSA in this document. SHA-512 is suitable for all FN-DSA parameter sets and provides an interoperable option for legacy CMS implementations that wish to migrate to use post-quantum cryptography, but that may not support use of SHA-3 derivatives at the CMS layer. However, other hash functions MAY also be supported; in particular, SHAKE256 SHOULD be supported, as this is the digest algorithm used internally in FN-DSA. When SHA-512 is used, the id-sha512 [RFC5754] digest algorithm identifier is used and the parameters field MUST be omitted. When SHAKE256 is used, the id-shake256 [RFC8702] digest algorithm identifier is used and the parameters field MUST be omitted. SHAKE256 produces 512 bits of output when used as a message digest algorithm in the CMS.

When signing using FN-DSA without including signed attributes, the algorithm specified in the digestAlgorithm field has no meaning, as FN-DSA computes signatures over entire messages rather than externally computed digests. As such, the considerations above and in Table 1 do not apply. Nonetheless, in this case implementations MUST specify SHA-512 as the digestAlgorithm in order to minimise the likelihood of an interoperability failure. When processing a SignerInfo signed using FN-DSA, if no signed attributes are present, implementations MUST ignore the content of the digestAlgorithm field.

| TODO: Verify table entries.

Signature Algorithm	Digest Algorithms
FN-DSA-512	SHA-256, SHA-384, SHA-512, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256
FN-DSA-1024	SHA-512, SHA3-512, SHAKE256

Table 1: Suitable Digest Algorithms for FN-DSA

signatureAlgorithm: The signatureAlgorithm field MUST contain one of the FN-DSA signature algorithm OIDs, and the parameters field MUST be absent. The algorithm OID MUST be one of the following OIDs described in Section 2:

Signature Algorithm	Algorithm Identifier OID
FN-DSA-512	id-fn-dsa-512
FN-DSA-10124	id-fn-dsa-1024

Table 2: Signature algorithm identifier OIDs for FN-DSA

| TODO: Verify paragraph references.

signature: The signature field contains the signature value resulting from the use of the FN-DSA signature algorithm identified by the signatureAlgorithm field. The FN-DSA (pure mode) signature-generation operation is specified in Section X.X of [FIPS206], and the signature-verification operation is specified in Section X.X of [FIPS206]. Note that Section 5.6 of [RFC5652] places further requirements on the successful verification of a signature.

4. Security Considerations

The security considerations in [RFC5652] and [I-D.ietf-lamps-fn-dsa-certificates] apply to this specification.

Security of the FN-DSA private key is critical. Compromise of the private key will enable an adversary to forge arbitrary signatures.

| TODO: Verify paragraph reference.

FN-DSA depends on high quality random numbers that are suitable for use in cryptography. The use of inadequate pseudo-random number generators (PRNGs) to generate such values can significantly undermine the security properties offered by a cryptographic algorithm. For instance, an attacker may find it much easier to reproduce the PRNG environment that produced any private keys, searching the resulting small set of possibilities, rather than brute-force searching the whole key space. The generation of random numbers of a sufficient level of quality for use in cryptography is difficult; see Section X.X.X of [FIPS206] for some additional information.

| TODO: Insert references for active research.

FN-DSA signature generation uses randomness from two sources: fresh random data generated during signature generation, and precomputed random data included in the signer’s private key. Inclusion of both sources of random data can help mitigate against faulty random number generators, side-channel attacks, and fault attacks. Lack of fresh random data during FN-DSA signature generation leads to a differential fault attack [BD23]. Side channel attacks and fault attacks against FN-DSA are an active area of research XX XX. Future protection against these styles of attack may involve interoperable changes to the implementation of FN-DSA’s internal functions. Implementers SHOULD consider implementing such protection measures if it would be beneficial for their particular use cases.

To avoid algorithm substitution attacks, the CMSAlgorithmProtection attribute defined in [RFC6211] SHOULD be included in signed attributes.

5. Operational Considerations

If FN-DSA signing is implemented in a hardware device such as a hardware security module (HSM) or a portable cryptographic token, implementers might want to avoid sending the full content to the device for performance reasons. By including signed attributes, which necessarily includes the message-digest attribute and the content-type attribute as described in Section 5.3 of [RFC5652], the much smaller set of signed attributes are sent to the device for signing.

6. IANA Considerations

For the ASN.1 module in Appendix A, IANA [is requested/has assigned] the following object identifier (OID) in the "SMI Security for S/MIME Module Identifier" registry (1.2.840.113549.1.9.16.0):

Decimal	Description	Reference
TBD	id-mod-fn-dsa-2026	This RFC

Table 3: Object Identifier Assignments

7. References

7.1. Normative References

[CSOR] NIST, "Computer Security Objects Register", 20 August 2024, <https://csrc.nist.gov/projects/computer-security-objects-register/algorithm-registration>.

[FIPS206] "Fast Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm", n.d., <https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards>.

[I-D.ietf-lamps-fn-dsa-certificates] Massimo, J., Kampanakis, P., Turner, S., and B. Westerbaan, "Internet X.509 Public Key Infrastructure -- Algorithm Identifiers for the Fast-Fourier Transform over NTRU-Lattice-Based Digital Signature Algorithm (FN-DSA)", Work in Progress, Internet-Draft, draft-ietf-lamps-fn-dsa-certificates-00, 20 May 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-lamps-fn-dsa-certificates-00>.

[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>.

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

[RFC5754] Turner, S., "Using SHA2 Algorithms with Cryptographic Message Syntax", RFC 5754, DOI 10.17487/RFC5754, January 2010, <https://www.rfc-editor.org/rfc/rfc5754>.

[RFC6211] Schaad, J., "Cryptographic Message Syntax (CMS) Algorithm Identifier Protection Attribute", RFC 6211, DOI 10.17487/RFC6211, April 2011, <https://www.rfc-editor.org/rfc/rfc6211>.

[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>.

[RFC8702] Kampanakis, P. and Q. Dang, "Use of the SHAKE One-Way Hash Functions in the Cryptographic Message Syntax (CMS)", RFC 8702, DOI 10.17487/RFC8702, January 2020, <https://www.rfc-editor.org/rfc/rfc8702>.

[X690] ITU-T, "Information Technology -- Abstract Syntax Notation One (ASN.1): ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, ISO/IEC 8825-1:2021, February 2021, <https://www.itu.int/rec/T-REC-X.690>.

7.2. Informative References

[BD23] Bauer, S. and F. D. Santis, "A Differential Fault Attack against Deterministic Falcon Signatures", 2023, <https://eprint.iacr.org/2023/422>.

[FIPS180] "Secure hash standard", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.180-4, 2015, <https://doi.org/10.6028/nist.fips.180-4>.

[FIPS20] "Module-lattice-based digital signature standard", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.204, August 2024, <https://doi.org/10.6028/nist.fips.204>.

[FIPS205] "Stateless hash-based digital signature standard", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.205, August 2024, <https://doi.org/10.6028/nist.fips.205>.

[I-D.ietf-lamps-dilithium-certificates] Massimo, J., Kampanakis, P., Turner, S., and B. Westerbaan, "Internet X.509 Public Key Infrastructure - Algorithm Identifiers for the Module-Lattice-Based Digital Signature Algorithm (ML-DSA)", Work in Progress, Internet-Draft, draft-ietf-lamps-dilithium-certificates-13, 30 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-lamps-dilithium-certificates-13>.

[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/rfc/rfc5280>.

[RFC5911] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, DOI 10.17487/RFC5911, June 2010, <https://www.rfc-editor.org/rfc/rfc5911>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/rfc/rfc8032>.

[RFC8419] Housley, R., "Use of Edwards-Curve Digital Signature Algorithm (EdDSA) Signatures in the Cryptographic Message Syntax (CMS)", RFC 8419, DOI 10.17487/RFC8419, August 2018, <https://www.rfc-editor.org/rfc/rfc8419>.

[RFC9814] Housley, R., Fluhrer, S., Kampanakis, P., and B. Westerbaan, "Use of the SLH-DSA Signature Algorithm in the Cryptographic Message Syntax (CMS)", RFC 9814, DOI 10.17487/RFC9814, July 2025, <https://www.rfc-editor.org/rfc/rfc9814>.

[RFC9881] Massimo, J., Kampanakis, P., Turner, S., and B. E. Westerbaan, "Internet X.509 Public Key Infrastructure -- Algorithm Identifiers for the Module-Lattice-Based Digital Signature Algorithm (ML-DSA)", RFC 9881, DOI 10.17487/RFC9881, October 2025, <https://www.rfc-editor.org/rfc/rfc9881>.

[RFC9882] Salter, B., Raine, A., and D. Van Geest, "Use of the ML-DSA Signature Algorithm in the Cryptographic Message Syntax (CMS)", RFC 9882, DOI 10.17487/RFC9882, October 2025, <https://www.rfc-editor.org/rfc/rfc9882>.

[X680] ITU-T, "Information Technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation. ITU-T Recommendation X.680 (2021) | ISO/IEC 8824-1:2021.", ITU-T Recommendation X.680, ISO/IEC 8824-1:2021, February 2021, <https://www.itu.int/rec/T-REC-X.680>.

Appendix A. ASN.1 Module

> | RFC EDITOR: Please replace the reference to
> | [I-D.ietf-lamps-fn-dsa-certificates] in the ASN.1 module below
> | with a reference the corresponding published RFC.

```
<CODE BEGINS>
FN-DSA-Module-2026
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    id-smime(16) id-mod(0) id-mod-fn-dsa-2026(TBD1) }

DEFINITIONS IMPLICIT TAGS ::= BEGIN

EXPORTS ALL;

IMPORTS SIGNATURE-ALGORITHM, SMIME-CAPS
  FROM AlgorithmInformation-2009 -- in [RFC5911]
  { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-algorithmInformation-02(58) }

sa-fn-dsa-512, sa-fn-dsa-10124
  FROM X509-FN-DSA-2026 -- From [I-D.turner-lamps-fn-dsa-certificates]
  { iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-x509-fn-dsa-2024(TBD2) } ;

--
-- Expand the signature algorithm set used by CMS [RFC5911]
--

SignatureAlgorithmSet SIGNATURE-ALGORITHM ::= {
  sa-fn-dsa-512 |
  sa-fn-dsa-1024,
  ... }

SMimeCaps SMIME-CAPS ::= {
  sa-fn-dsa-512.&smimeCaps |
  sa-fn-dsa-1024.&smimeCaps,
  ... }

END
<CODE ENDS>
```

Appendix B. Examples

This appendix contains example signed-data encodings. They can be verified using the example public keys and certificates specified in Appendix C of [I-D.ietf-lamps-fn-dsa-certificates].

The following is an example of a signed-data with a single FN-DSA-512 signer, with signed attributes included:

| TODO: Get Example.

The following is an example of a signed-data with a single FN-DSA-1024 signer, with signed attributes included:

| TODO: Get Example.

Acknowledgments

| TODO:

This document was heavily influenced by [RFC8419], [RFC9814], and [RFC9881]. Thanks go to the authors of those documents.

Authors' Addresses

Daniel Van Geest
CryptoNext Security
Email: daniel.vangeest@cryptonext-security.com

Sean
sn3rd
Email: sean@sn3rd.com