Network Working Group
Request for Comments: 5480
Updates: 3279
Category: Standards Track

S. Turner
IECA
D. Brown
Certicom
K. Yiu
Microsoft
R. Housley
Vigil Security
T. Polk
NIST

March 2009

Elliptic Curve Cryptography Subject Public Key Information

Status of This Memo

Copyright Notice

Abstract

This document specifies the syntax and semantics for the Subject Public Key Information field in certificates that support Elliptic Curve Cryptography. This document updates Sections 2.3.5 and 5, and the ASN.1 module of "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279.

Table of Contents

1. Introduction

This document specifies the format of the subjectPublicKeyInfo field in X.509 certificates [PKI] that use Elliptic Curve Cryptography (ECC). It updates RFC 3279 [PKI-ALG]. This document specifies the encoding formats for public keys used with the following ECC algorithms:

- Elliptic Curve Digital Signature Algorithm (ECDSA);
- Elliptic Curve Diffie-Hellman (ECDH) family schemes; and
- Elliptic Curve Menezes-Qu-Vanstone (ECMQV) family schemes.

Two methods for specifying the algorithms that can be used with the subjectPublicKey are defined. One method allows the key to be used with any ECC algorithm, while the other method restricts the usage of the key to specific algorithms. To promote interoperability, this document indicates which is required to implement for Certification Authorities (CAs) that implement ECC algorithms and relying parties that claim to process ECC algorithms.

The ASN.1 [X.680] module in this document includes ASN.1 for ECC algorithms. It also includes ASN.1 for non-ECC algorithms defined in [PKI-ALG] and [PKI-ADALG], even though the associated text is unaffected. By updating all of the ASN.1 from [PKI-ALG] in this document, implementers only need to use the module found in this document.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [MUSTSHOULD].

2. Subject Public Key Information Fields

In the X.509 certificate, the subjectPublicKeyInfo field has the SubjectPublicKeyInfo type, which has the following ASN.1 syntax:

```
SubjectPublicKeyInfo  ::=  SEQUENCE  {
  algorithm         AlgorithmIdentifier,
  subjectPublicKey  BIT STRING
}
```

The fields in SubjectPublicKeyInfo have the following meanings:

- algorithm is the algorithm identifier and parameters for the ECC public key.

- subjectPublicKey is the ECC public key. See Section 2.2.

The AlgorithmIdentifier type, which is included for convenience [PKI], is defined as follows:

```
AlgorithmIdentifier  ::=  SEQUENCE  {
  algorithm   OBJECT IDENTIFIER,
  parameters  ANY DEFINED BY algorithm OPTIONAL
}
```

The fields in AlgorithmIdentifier have the following meanings:

- algorithm identifies the cryptographic algorithm with an object identifier. See Section 2.1.

- parameters, which are optional, are the associated parameters for the algorithm identifier in the algorithm field. See Section 2.1.1.

2.1. Elliptic Curve Cryptography Public Key Algorithm Identifiers

The algorithm field in the SubjectPublicKeyInfo structure [PKI] indicates the algorithm and any associated parameters for the ECC public key (see Section 2.2). Three algorithm identifiers are defined in this document:

- id-ecPublicKey indicates that the algorithms that can be used with the subject public key are unrestricted. The key is only restricted by the values indicated in the key usage certificate extension (see Section 3). id-ecPublicKey MUST be supported. See Section 2.1.1. This value is also included in certificates when a public key is used with ECDSA.

- id-ecDH indicates that the algorithm that can be used with the subject public key is restricted to the Elliptic Curve Diffie-Hellman algorithm. See Section 2.1.2. id-ecDH MAY be supported.

- id-ecMQV indicates that the algorithm that can be used with the subject public key is restricted to the Elliptic Curve Menezes-Qu-Vanstone key agreement algorithm. See Section 2.1.2. id-ecMQV MAY be supported.

2.1.1. Unrestricted Algorithm Identifier and Parameters

The "unrestricted" algorithm identifier is:

```
id-ecPublicKey OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) 1 }
```

The public key (ECPoint) syntax is described in Section 2.2.

The parameter for id-ecPublicKey is as follows and MUST always be present:

```
ECParameters ::= CHOICE {
  namedCurve         OBJECT IDENTIFIER
  -- implicitCurve   NULL
  -- specifiedCurve  SpecifiedECDomain
}
  -- implicitCurve and specifiedCurve MUST NOT be used in PKIX.
  -- Details for SpecifiedECDomain can be found in [X9.62].
  -- Any future additions to this CHOICE should be coordinated
  -- with ANSI X9.
```

The fields in ECParameters have the following meanings:

- namedCurve identifies all the required values for a particular set of elliptic curve domain parameters to be represented by an object identifier. This choice MUST be supported. See Section 2.1.1.1.

- implicitCurve allows the elliptic curve domain parameters to be inherited. This choice MUST NOT be used.

- specifiedCurve, which is of type SpecifiedECDomain type (defined in [X9.62]), allows all of the elliptic curve domain parameters to be explicitly specified. This choice MUST NOT be used. See Section 5, "ASN.1 Considerations".

The addition of any new choices in ECParameters needs to be coordinated with ANSI X9.

The AlgorithmIdentifier within SubjectPublicKeyInfo is the only place within a certificate where the elliptic curve domain parameters may be located. If the elliptic curve domain parameters are not present, then clients MUST reject the certificate.

2.1.1.1. Named Curve

The namedCurve field in ECParameters uses object identifiers to name well-known curves. This document publishes curve identifiers for the fifteen NIST-recommended curves [FIPS186-3]. Other documents can publish other name curve identifiers. The NIST-named curves are:

```
-- Note that in [X9.62] the curves are referred to as 'ansiX9' as
-- opposed to 'sec'. For example, secp192r1 is the same curve as
-- ansix9p192r1.

-- Note that in [PKI-ALG] the secp192r1 curve was referred to as
-- prime192v1 and the secp256r1 curve was referred to as
-- prime256v1.

-- Note that [FIPS186-3] refers to secp192r1 as P-192, secp224r1 as
-- P-224, secp256r1 as P-256, secp384r1 as P-384, and secp521r1 as
-- P-521.

secp192r1 OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
  prime(1) 1 }

sect163k1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 1 }

sect163r2 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 15 }

secp224r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 33 }

sect233k1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 26 }
```

```
sect233r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 27 }

secp256r1 OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
  prime(1) 7 }

sect283k1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 16 }

sect283r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 17 }

secp384r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 34 }

sect409k1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 36 }

sect409r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 37 }

secp521r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 35 }

sect571k1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 38 }

sect571r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 39 }
```

2.1.2. Restricted Algorithm Identifiers and Parameters

Two "restricted" algorithms are defined for key agreement algorithms: the Elliptic Curve Diffie-Hellman (ECDH) key agreement family schemes and the Elliptic Curve Menezes-Qu-Vanstone (ECMQV) key agreement family schemes. Both algorithms are identified by an object identifier and have parameters. The object identifier varies based on the algorithm, but the parameters are always ECParameters and they MUST always be present (see Section 2.1.1).

The ECDH algorithm uses the following object identifier:

```
id-ecDH OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) schemes(1)
  ecdh(12) }
```

The ECMQV algorithm uses the following object identifier:

```
id-ecMQV OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) schemes(1)
  ecmqv(13) }
```

2.2. Subject Public Key

The subjectPublicKey from SubjectPublicKeyInfo is the ECC public key. ECC public keys have the following syntax:

```
ECPoint ::= OCTET STRING
```

Implementations of Elliptic Curve Cryptography according to this document MUST support the uncompressed form and MAY support the compressed form of the ECC public key. The hybrid form of the ECC public key from [X9.62] MUST NOT be used. As specified in [SEC1]:

- The elliptic curve public key (a value of type ECPoint that is an OCTET STRING) is mapped to a subjectPublicKey (a value of type BIT STRING) as follows: the most significant bit of the OCTET STRING value becomes the most significant bit of the BIT STRING value, and so on; the least significant bit of the OCTET STRING becomes the least significant bit of the BIT STRING. Conversion routines are found in Sections 2.3.1 and 2.3.2 of [SEC1].

- The first octet of the OCTET STRING indicates whether the key is compressed or uncompressed. The uncompressed form is indicated by 0x04 and the compressed form is indicated by either 0x02 or 0x03 (see 2.3.3 in [SEC1]). The public key MUST be rejected if any other value is included in the first octet.

3. Key Usage Bits

If the keyUsage extension is present in a Certification Authority (CA) certificate that indicates id-ecPublicKey in SubjectPublicKeyInfo, then any combination of the following values MAY be present:

```
digitalSignature;
nonRepudiation;
keyAgreement;
keyCertSign; and
cRLSign.
```

If the CA certificate keyUsage extension asserts keyAgreement, then it MAY assert either encipherOnly or decipherOnly. However, this specification RECOMMENDS that if keyCertSign or cRLSign is present, then keyAgreement, encipherOnly, and decipherOnly SHOULD NOT be present.

If the keyUsage extension is present in an End Entity (EE) certificate that indicates id-ecPublicKey in SubjectPublicKeyInfo, then any combination of the following values MAY be present:

digitalSignature;
nonRepudiation; and
keyAgreement.

If the EE certificate keyUsage extension asserts keyAgreement, then it MAY assert either encipherOnly or decipherOnly.

If the keyUsage extension is present in a certificate that indicates id-ecDH or id-ecMQV in SubjectPublicKeyInfo, then the following MUST be present:

keyAgreement;

one of the following MAY be present:

encipherOnly; or
decipherOnly.

If the keyUsage extension is present in a certificate that indicates id-ecDH or id-ecMQV in SubjectPublicKeyInfo, then the following values MUST NOT be present:

digitalSignature;
nonRepudiation;
keyTransport;
keyCertSign; and
cRLSign.

4. Security Considerations

The security considerations in [PKI-ALG] apply.

When implementing ECC in X.509 Certificates and Certificate Revocation Lists (CRLs), there are three algorithm-related choices that need to be made for the signatureAlgorithm field in a Certificate or CertificateList:

1) What is the public key size?

2) What is the hash algorithm [FIPS180-3]?

3) What is the curve?

Consideration must be given by the CA to the strength of the security provided by each of these choices. Security is measured in bits, where a strong symmetric cipher with a key of X bits is said to provide X bits of security. It is recommended that the bits of security provided by each choice are roughly equivalent. The following table provides comparable minimum bits of security [SP800-57] for the ECDSA key sizes and message digest algorithms. It also lists curves (see Section 2.1.1.1) for the key sizes.

Minimum Bits of Security	ECDSA Key Size	Message Digest Algorithms	Curves
80	160-223	SHA-1	sect163k1
		SHA-224	secp163r2
		SHA-256	secp192r1
		SHA-384	
		SHA-512	
112	224-255	SHA-224	secp224r1
		SHA-256	sect233k1
		SHA-384	sect233r1
		SHA-512	
128	256-383	SHA-256	secp256r1
		SHA-384	sect283k1
		SHA-512	sect283r1
192	384-511	SHA-384	secp384r1
		SHA-512	sect409k1
			sect409r1
256	512+	SHA-512	secp521r1
			sect571k1
			sect571r1

To promote interoperability, the following choices are RECOMMENDED:

Minimum Bits of Security	ECDSA Key Size	Message Digest Algorithms	Curves
80	192	SHA-256	secp192r1
112	224	SHA-256	secp224r1
128	256	SHA-256	secp256r1
192	384	SHA-384	secp384r1
256	512	SHA-512	secp521r1

Using a larger hash value and then truncating it consumes more processing power than is necessary. This is more important on constrained devices. Since the signer does not know the environment that the recipient will use to validate the signature, it is better to use a hash function that provides the desired hash value output size, and no more.

There are security risks with using keys not associated with well-known and widely reviewed curves. For example, the curve may not satisfy the Menezes-Okamoto-Vanstone (MOV) condition [X9.62] or the curve may be vulnerable to the Anomalous attack [X9.62]. Additionally, either a) all of the arithmetic properties of a candidate ECC public key must be validated to ensure that it has the unique correct representation in the correct (additive) subgroup (and therefore is also in the correct EC group) specified by the associated ECC domain parameters, or b) some of the arithmetic properties of a candidate ECC public key must be validated to ensure that it is in the correct group (but not necessarily the correct subgroup) specified by the associated ECC domain parameters [SP800-56A].

As noted in [PKI-ALG], the use of MD2 and MD5 for new applications is discouraged. It is still reasonable to use MD2 and MD5 to verify existing signatures.

5. ASN.1 Considerations

[X9.62] defines additional options for ECParameters and ECDSA-Sig-Value [PKI-ALG]. If an implementation needs to use these options, then use the [X9.62] ASN.1 module. This RFC contains a conformant subset of the ASN.1 module defined in [X9.62].

If an implementation generates a PER [X.691] encoding using the ASN.1 module found in this specification, it might not achieve the same encoded output as one that uses the [X9.62] module. PER is not required by either the PKIX or S/MIME environments. If an implementation environment requires PER, then implementation concerns are less likely with the use of the [X9.62] module.

6. IANA Considerations

This document makes extensive use of object identifiers to register public key types, elliptic curves, and algorithms. Most are registered in the ANSI X9.62 arc, with the exception of the hash algorithms (which are in the NIST arc) and many of the curves (which are in the Certicom Inc. arc; these curves have been adopted by ANSI and NIST). Additionally, an object identifier is used to identify the ASN.1 module found in Appendix A. It is defined in an arc delegated by IANA to the PKIX Working Group. No further action by IANA is necessary for this document or any anticipated updates.

7. Acknowledgments

The authors wish to thank Stephen Farrell, Alfred Hoenes, Johannes Merkle, Jim Schaad, and Carl Wallace for their valued input.

8. References

8.1. Normative References

[FIPS180-3] National Institute of Standards and Technology (NIST), FIPS Publication 180-3: Secure Hash Standard, October 2008.

[FIPS186-3] National Institute of Standards and Technology (NIST), FIPS Publication 186-3: Digital Signature Standard, (draft) November 2008.

[MUSTSHOULD] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[PKI] 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, May 2008.

[PKI-ALG] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[RSAOAEP] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[SEC1] Standards for Efficient Cryptography Group (SECG), "SEC 1: Elliptic Curve Cryptography", Version 1.0, September 2000.

[X9.62] American National Standards Institute (ANSI), ANS X9.62-2005: The Elliptic Curve Digital Signature Algorithm (ECDSA), 2005.

[X.680] ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002. Information Technology - Abstract Syntax Notation One.

8.2. Informative References

[PKI-ADALG] Dang, Q., Santesson, S., Moriarty, K., Brown, D., and T. Polk, "Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA", Work in Progress, October 2008.

[SP800-56A] National Institute of Standards and Technology (NIST), Special Publication 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised), March 2007.

[SP800-57] National Institute of Standards and Technology (NIST), Special Publication 800-57: Recommendation for Key Management - Part 1 (Revised), March 2007.

[X.691] ITU-T Recommendation X.691 (2002) | ISO/IEC 8825-2:2002. Information Technology - ASN.1 Encoding Rules: Specification of Packed Encoding Rules.

Appendix A. ASN.1 Module

```
PKIX1Algorithms2008 { iso(1) identified-organization(3) dod(6)
  internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) 45 }

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL;

IMPORTS

-- From RFC 4055 [RSAOAEP]

id-sha224, id-sha256, id-sha384, id-sha512
  FROM PKIX1-PSS-OAEP-Algorithms
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkix1-rsa-pkalgs(33) }

;

--
-- Message Digest Algorithms
--

-- MD-2
-- Parameters are NULL

id-md2  OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) 2 }

-- MD-5
-- Parameters are NULL

id-md5  OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549)digestAlgorithm(2) 5 }

-- SHA-1
-- Parameters are preferred absent

id-sha1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) oiw(14) secsig(3)
  algorithm(2) 26 }

-- SHA-224
-- Parameters are preferred absent
```

```
   -- id-sha224 OBJECT IDENTIFIER ::= {
   --   joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
   --   csor(3) nistalgorithm(4) hashalgs(2) 4 }
   -- SHA-256
   -- Parameters are preferred absent

   -- id-sha256 OBJECT IDENTIFIER ::= {
   --   joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
   --   csor(3) nistalgorithm(4) hashalgs(2) 1 }

   -- SHA-384
   -- Parameters are preferred absent

   -- id-sha384 OBJECT IDENTIFIER ::= {
   --   joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
   --   csor(3) nistalgorithm(4) hashalgs(2) 2 }

   -- SHA-512
   -- Parameters are preferred absent

   -- id-sha512 OBJECT IDENTIFIER ::= {
   --   joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
   --   csor(3) nistalgorithm(4) hashalgs(2) 3 }

   --
   -- Public Key (PK) Algorithms
   --

   -- RSA PK Algorithm and Key

   rsaEncryption OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 1 }

   RSAPublicKey ::= SEQUENCE {
     modulus         INTEGER, -- n
     publicExponent  INTEGER  -- e
   }

   -- DSA PK Algorithm, Key, and Parameters

   id-dsa OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) x9-57(10040) x9algorithm(4) 1 }

   DSAPublicKey ::= INTEGER --  public key, y
```

```
   DSS-Parms ::= SEQUENCE {
     p  INTEGER,
     q  INTEGER,
     g  INTEGER
   }

   -- Diffie-Hellman PK Algorithm, Key, and Parameters

   dhpublicnumber OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-x942(10046) number-type(2) 1 }

   DHPublicKey ::= INTEGER  -- public key, y = g^x mod p

   DomainParameters ::= SEQUENCE {
     p                INTEGER,           -- odd prime, p=jq +1
     g                INTEGER,           -- generator, g
     q                INTEGER,           -- factor of p-1
     j                INTEGER OPTIONAL,  -- subgroup factor, j>= 2
     validationParms  ValidationParms OPTIONAL
   }

   ValidationParms ::= SEQUENCE {
     seed         BIT STRING,
     pgenCounter  INTEGER
   }

   -- KEA PK Algorithm and Parameters

   id-keyExchangeAlgorithm OBJECT IDENTIFIER ::= {
     joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
     dod(2) infosec(1) algorithms(1) 22 }


   KEA-Parms-Id ::= OCTET STRING

   -- Sec 2.1.1 Unrestricted Algorithm ID, Key, and Parameters
   -- (ECDSA keys use id-ecPublicKey)

   id-ecPublicKey OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) 1 }

   ECPoint ::= OCTET STRING

   -- Parameters for both Restricted and Unrestricted

   ECParameters ::= CHOICE {
     namedCurve         OBJECT IDENTIFIER
     -- implicitCurve   NULL
```

```
      -- specifiedCurve  SpecifiedECDomain
    }
      -- implicitCurve and specifiedCurve MUST NOT be used in PKIX.
      -- Details for SpecifiedECDomain can be found in [X9.62].
      -- Any future additions to this CHOICE should be coordinated
      -- with ANSI X9.

    -- Sec 2.1.2 Restricted Algorithm IDs, Key, and Parameters: ECDH

    id-ecDH OBJECT IDENTIFIER ::= {
      iso(1) identified-organization(3) certicom(132) schemes(1)
      ecdh(12) }

    -- ECPoint ::= OCTET STRING

    -- Parameters are ECParameters.

    -- Sec 2.1.2 Restricted Algorithm IDs, Key, and Parameters: ECMQV

    id-ecMQV OBJECT IDENTIFIER ::= {
      iso(1) identified-organization(3) certicom(132) schemes(1)
      ecmqv(13) }

    -- ECPoint ::= OCTET STRING

    -- Parameters are ECParameters.

    --
    -- Signature Algorithms
    --

    -- RSA with MD-2
    -- Parameters are NULL

    md2WithRSAEncryption OBJECT IDENTIFIER ::= {
      iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 2 }

    -- RSA with MD-5
    -- Parameters are NULL

    md5WithRSAEncryption OBJECT IDENTIFIER ::= {
      iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 4 }

    -- RSA with SHA-1
    -- Parameters are NULL

    sha1WithRSAEncryption OBJECT IDENTIFIER ::= {
      iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5 }
```

```
   -- DSA with SHA-1
   -- Parameters are ABSENT

   id-dsa-with-sha1 OBJECT IDENTIFIER ::=  {
     iso(1) member-body(2) us(840) x9-57(10040) x9algorithm(4) 3 }

   -- DSA with SHA-224
   -- Parameters are ABSENT

   id-dsa-with-sha224 OBJECT IDENTIFIER  ::=  {
     joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101)
     csor(3) algorithms(4) id-dsa-with-sha2(3) 1 }

   -- DSA with SHA-256
   -- Parameters are ABSENT

   id-dsa-with-sha256 OBJECT IDENTIFIER  ::=  {
     joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101)
     csor(3) algorithms(4) id-dsa-with-sha2(3) 2 }

   -- ECDSA with SHA-1
   -- Parameters are ABSENT

   ecdsa-with-SHA1 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) 1 }

   -- ECDSA with SHA-224
   -- Parameters are ABSENT

   ecdsa-with-SHA224 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
     ecdsa-with-SHA2(3) 1 }

   -- ECDSA with SHA-256
   -- Parameters are ABSENT

   ecdsa-with-SHA256 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
     ecdsa-with-SHA2(3) 2 }

   -- ECDSA with SHA-384
   -- Parameters are ABSENT

   ecdsa-with-SHA384 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
     ecdsa-with-SHA2(3) 3 }
```

```
   -- ECDSA with SHA-512
   -- Parameters are ABSENT

   ecdsa-with-SHA512 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
     ecdsa-with-SHA2(3) 4 }

   --
   -- Signature Values
   --

   -- DSA

   DSA-Sig-Value ::= SEQUENCE {
     r  INTEGER,
     s  INTEGER
   }

   -- ECDSA

   ECDSA-Sig-Value ::= SEQUENCE {
     r  INTEGER,
     s  INTEGER
   }

   --
   -- Named Elliptic Curves
   --

   -- Note that in [X9.62] the curves are referred to as 'ansiX9' as
   -- opposed to 'sec'.  For example secp192r1 is the same curve as
   -- ansix9p192r1.

   -- Note that in [PKI-ALG] the secp192r1 curve was referred to as
   -- prime192v1 and the secp256r1 curve was referred to as prime256v1.

   -- Note that [FIPS186-3] refers to secp192r1 as P-192, secp224r1 as
   -- P-224, secp256r1 as P-256, secp384r1 as P-384, and secp521r1 as
   -- P-521.

   secp192r1 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
     prime(1) 1 }

   sect163k1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 1 }
```

```
   sect163r2 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 15 }

   secp224r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 33 }

   sect233k1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 26 }

   sect233r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 27 }

   secp256r1 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
     prime(1) 7 }

   sect283k1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 16 }

   sect283r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 17 }

   secp384r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 34 }

   sect409k1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 36 }

   sect409r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 37 }

   secp521r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 35 }

   sect571k1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 38 }

   sect571r1 OBJECT IDENTIFIER ::= {
     iso(1) identified-organization(3) certicom(132) curve(0) 39 }

   END
```

Authors' Addresses

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031
USA

EMail: turners@ieca.com

Kelvin Yiu
Microsoft
One Microsoft Way
Redmond, WA 98052-6399
USA

EMail: kelviny@microsoft.com

Daniel R. L. Brown
Certicom Corp
5520 Explorer Drive #400
Mississauga, ON L4W 5L1
CANADA

EMail: dbrown@certicom.com

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

EMail: housley@vigilsec.com

Tim Polk
NIST
Building 820, Room 426
Gaithersburg, MD 20899

EMail: wpolk@nist.gov