IEEE P802.15
Wireless Personal Area Networks

Project	IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)		
Title	**Mandatory ECC Security Algorithm Suite**		
Date Submitted	April 19, 2002		
Source	[Rene Struik, Certicom Corp.] 5520 Explorer Drive, 4th Floor Mississauga, ON Canada L4W 5L1 [Gregg Rasor, Motorola] 1500 Gateway Blvd. Boynton Beach, Florida 33426	Voice: Fax: E-mail: Voice: Fax: E-mail:	(905) 501-6083 (905) 507-4230 rstruik@certicom.com (561) 739-2952 (561) 739-3715 Gregg.Rasor@motorola.com
Re:	802.15.3 TG3 Security call for proposals		
Abstract	An elliptic curve cryptography (ECC) security suite is defined including a full public key cryptosystem with digital signatures and implicit certificates. Binding between a device's identity and public key using digital certificates may be established manually through direct user intervention, at device provisioning by a distributor or manufacturer, or by a trusted third party.		
Purpose	This document defines the security elements necessary to implement the mandatory and optional elliptic curve cryptography (ECC) security suite.		
Notice	This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.		
Release	The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15.		

This document provides drop-in text for inclusion in Draft D10 that will fully define the mandatory and optional elliptic curve security suites.

Table of Contents

11. Security Suite Specifications

This clause specifies the security suites that may be used when security is turned on in the piconet. The security suite specifications define the algorithms and operations that shall be performed when the security manager within selects a specific security suite for a specific security relationship. Within a piconet, the security manager shall select a specific security suite to be used by all devices that engage in device authentication and payload protection activities with that security manager.

11.1 Modes for security suites

This clause describes the modes of operation in which the piconet may operate.

11.1.1 Mode 0: No security

When in Mode 0, no security is provided. This mode is not further described, since it does not involve cryptographic operations.

11.1.2 Mode 1: Security via non-cryptographic means

When in Mode 1, access control to the piconet or establishment of a peer-to-peer security relationship is based on verifying the authorization status of a joining device by checking compliance of the device's identifying information with information in the access control list (ACL). This mode is not further described, since it does not involve cryptographic operations.

11.1.3 Mode 2: Public-key based security without certificates

When in Mode 2, access control to the piconet or establishment of a peer-to-peer security relationship is based on the successful execution of a public-key based protocol, in which each of the communicating devices presents its purported public key and device identifier (the so-called manual certificate) to the other party. In addition, the establishment of the security relationship may be based on the authorization status of a joining device by checking whether the public key information as contained in the manual certificate complies with the information in the access control list (ACL). This mode is referred to as the 'Raw' sub-mode.

11.1.4 Mode 3: Public-key based security with certificates

When in Mode 3, access control to the piconet or establishment of a peer-to-peer security relationship is based on the successful execution of a public-key based protocol, in which each of the communicating devices presents its digital certificate to the other party. In addition, the establishment of the security relationship may be based on the authorization status of a joining device by checking whether the public key information as contained in this digital certificate complies with the information in the access control list (ACL). This standard distinguishes two sub-modes of Mode 3, depending upon the type of digital certificate involved (implicit certificate vs. X509-based certificate). The first sub-mode is referred to as the Implicit sub-mode; the second as the X509 sub-mode.

11.2 Security suite selections

11.2.1 OID selections

This clause specifies the security suites that shall be used by compliant implementations. Each security suite is identified by a globally unique object identifier (OID). All OIDs are built from the following arc:

id-802-15-3-security-suites OBJECT IDENTIFIER::=
{**iso**(1) **std**(0) **iso8802**(8802) **ieee802dot15**(15)
ieee802dot15dot3 (3) **securitysuites** (1)}

Table 1 specifies the OIDs for each of the security suites in the standard.

Table 1 – Security Suites

Security Suite Name	OID Name	OID Number	DER Coding
ECMQV 283-Koblitz-1	ecmqv-sec-suite-1	Id-802-15-3-security-suites 1	0x060728C4620F030101

Table 2 specifies the OIDs for each sub-mode of the security suites specified in Table 1.

Table 2 – OIDs for sub-modes

Security Suite Name	OID Name	OID Number	DER Coding
ECMQV Raw-1	ecmqv-raw-1	ecmqv-sec-suite-1 1	0x060728C4620F03010101
ECMQV Implicit-1	ecmqv-implicit-1	ecmqv-sec-suite-1 2	0x060728C4620F03010102
ECMQV X509-1	ecmqv-X509-1	ecmqv-sec-suite-1 3	0x060728C4620F03010103

11.2.2 Symmetric cryptographic building blocks

The following cryptographic primitives and data elements are defined for use in all security suites specified in this standard.

11.2.2.1 Cryptographic hash function

The cryptographic hash function used in this standard shall be SHA-256, as specified in the Draft FIPS Pub 180-2 standard [{xref} FIPS 180-2].

11.2.2.2 Keyed hash function for message authentication

The keyed hash message authentication code (HMAC) used in this standard shall be HMAC, as specified in the FIPS Pub #HMAC [{xref} FIPS #HMAC]. The output of the HMAC function shall be truncated by omitting all but the leftmost 128 bits hereof.

11.2.2.3 CBC encryption mode

The cipher-block chaining mode of encryption for block ciphers shall be used in this standard, as specified in NIST Pub 800-38A 2001 ED [{xref} NIST].

11.2.2.4 Block cipher

The block cipher used for symmetric encryption and decryption in this standard shall be the Advanced Encryption Standard AES-128, as specified in FIPS Pub 196 [{xref}FIPS 197].

11.2.3 Symmetric cryptography implementation

All of the sub-modes defined for ECMQV-Koblitz-1 shall perform all symmetric operations in the same manner.

11.2.3.1 Symmetric cryptography data formats

The following table specifies the length and intended meaning of the symmetric-cryptography-related security-suite-specific data elements from Clause 7 {xref}. The operations performed to obtain the variable data values are specified in a separate sub-clause.

Table 4 – Symmetric cryptography frame object formats

Notation	Length	Value	Description
AuthResponseType	2	See Table 7.4.1.2 {xref}	The numerical value of the HMAC-output type in Table 7.4.1.2{xref}
AuthResponseLength	2	16	Length of the output of the HMAC function as specified in Clause 11.2.2.2 {xref}
AuthResponse	16	HMAC value	Instance of HMAC output as specified in Clause 11.2.2.2 {xref}
ChallengeResponseType	2	See Table 7.4.1.2 {xref}	The numerical value of the HMAC-output type in Table 7.4.1.2{xref}
ChallengeResponseLength	2	16	Length of the output of the HMAC function as specified in Clause 11.2.2.2 {xref}
ChallengeResponse	16	HMAC value	Instance of HMAC output as specified in Clause 11.2.2.2 {xref}
KeyPurpose	1	See Table 7.4.1.6 {xref}	The numerical value of the key type in Table 7.4.1.6 {xref}
EncryptedKeyType	2	See Table 7.4.1.6 {xref}	The numerical value of the encrypted key type in Table 7.4.1.6 {xref}
EncryptedKeyLength	2	32	Length of AES-encrypted seed as specified as specified in Clause 11.2.2.4 {xref}
EncryptedKey	32	Encrypted SEED value	Instance of AES-encrypted SEED as specified in Clause 11.2.2.4 {xref}

11.2.3.2 Symmetric cryptography operations

The following table specifies the symmetric-cryptography-related operations on all secure frames as specified in Clause 7 {xref}:

Table 5 – Symmetric cryptographic operations

Use	Description
Integrity key derivation	Process by which an integrity key becomes available, as a function of a contributed SEED and a fixed string '0x00'.
Encryption key derivation	Process by which an integrity key becomes available, as a function of a contributed SEED and a fixed string '0x01'.
Challenge response generation	Communication of message authentication code value over part of the communicated strings during the protocol, to be used for entity authentication and key confirmation.
Authentication response generation	Communication of message authentication code value over part of the communicated strings during the protocol, to be used for entity authentication and key confirmation.
Beacon message authentication code generation	Communication of message authentication code value over the entire beacon via the piconet-wide data integrity key, to be used for data origin authentication relative to the authorized parties that constitute the piconet.
Command message authentication generation	Communication of message authentication code value over a command frame via the integrity key specific to the command-issuing device, to be used for data origin authentication.
ACK message authentication code generation	Communication of message authentication code value over an ACK frame via the integrity key specific to the ACK-issuing device, to be used for data origin authentication.
Data message authentication code generation	Communication of message authentication code value over a data frame via the integrity key specific to the payload-communicating device, to be used for data origin authentication.
Seed encryption operation	AES-encryption of a SEED using the encryption key of the SEED-issuing party, to be used for derivation of data integrity and data encryption keys.
Data encryption generation	AES-encryption of payload data using the data integrity and data encryption keys of the key-issuing party (i.e., the party that generated the SEED for the data keys).

11.3 ECMQV 283-Koblitz-1 security suite

The following sub-clauses define the security operations that are performed for the security suite ECMQV 283-Koblitz-1. The symmetric operations performed in this security suite are those specified in Sub-clause 11.2.3 {xref}. The public key and authentication operations are specified in the following Sub-clause 11.3.1 {xref}.

11.3.1 Public-key and authentication building blocks

The following cryptographic primitives and data elements are defined for use in all sub-modes of ECMQV 283-koblitz-1.

11.3.1.1 Elliptic curve points

All elliptic curve points shall be represented in polynomial notation as specified in Section 4.1.2.1 of ANSI X9.63-2001 [{xref} X9.63}].

11.3.1.2 Elliptic curve point compression

All transmitted elliptic curve points shall be compressed as specified in Section 4.2.2 of ANSI X9.63-2001 [{xref}X9.63].

11.3.1.3 Elliptic curve public key pair

An elliptic curve key pair consists of an integer *q* and a point *Q* on the curve determined by multiplying the generating point *G* of the curve by this integer, i.e., *Q=qG*, as specified in ANSI X9.63-2001 [{xref X9.63}]. Here, *Q* is called the public key, whereas *q* is called the private key. The private key shall be represented as specified in Section 4.3.1 of ANSI X9.63-2001 [{xref} X9.63}]. The public key shall be represented as defined in Sub-clauses 11.3.1.1{xref} and 11.3.1.2 {xref}.

11.3.1.4 Elliptic curve ansit283k1

All elliptic curve points used in this security suite shall be points on the curve ansit283k1 as specified in Appendix J4.5, Example 1, of ANSI X9.63-2001 [{xref} X9.63].

11.3.1.5 ECMQV key agreement

The ECMQV key agreement protocol shall refer to the full MQV with key confirmation scheme as specified in Section 6.11 of ANSI X9.63-2001 [{xref} X9.63], with the following instantiations:

1. Each entity shall be a DEV;
2. Each entity's identifier shall be its 48-bit IEEE MAC addresses {xref}; the parameter *entlen* shall have the integer value 48;
3. Each entity shall use the HMAC-scheme as specified in Clause 11.2.2.2 {xref};
4. Each entity shall use the cryptographic hash function as specified in Clause 11.2.2.1 {xref};
5. The parameter *keydatalen* shall have the integer value 128;
6. The parameter *SharedData* shall be the empty string; the parameter *shareddatalen* shall have the integer value 0;
7. The optional parameters *Text1* and *Text2* as specified in Sections 6.11.1 and 6.11.2 of ANSI X9.63-2001 [{xref}X9.63] shall both be the empty string.

11.3.1.6 ECC certificates

The ECMQV 283-Koblitz-1 security suite shall use different certificates, depending upon the mode of operation. The security suite uses manual certificates, as specified in Clause 11.3.1.6.1, in Mode 2, ECC implicit certificates, as specified in Clause 11.3.1.6.2, in the Implicit Sub-mode of Mode 3, and ECC X509 certificates, as specified in Clause 11.3.1.6.3, in the X509 Sub-mode of Mode 3.

11.3.1.6.1 ECC manual 'certificates'

This standard shall use the manual certificate **ManCert** specified as follows:
ManCert=(PublicKey || Subject), where

1. **PublicKey** shall be the representation of an elliptic curve point, which is a public key as specified in Clause 11.3.1.3 {xref};
2. **Subject** shall be the purported owner of the private key corresponding to the public key represented by **PublicKey**, as represented by its 48-bit IEEE MAC address.

The manual certificate is no 'real' digital certificate, since the binding between a public key and its purported owner is to be established and verified by non-cryptographic means.

11.3.1.6.2 ECC implicit certificates

This standard shall use the implicit certificate scheme, with associated implicit certificate generation protocol and implicit certificate processing protocol, as specified in Appendix A {xref}, with the following instantiations:

1. Each entity shall be a DEV;
2. Each entity's identifier shall be its 48-bit IEEE MAC address {xref}; the parameter *entlen* shall have the integer value 48;
3. Each entity shall use the cryptographic hash function as specified in Clause 11.2.2.1 {xref};
4. The format of the implicit certificate **ImplCert** shall be specified as follows: **ImplCert=(PublicReconstrKey || Subject || Issuer)**, where
   a. **ImplCert** shall be the representation of the string IC_U as specified in the implicit certificate generation protocol;
   b. **PublicReconstrKey** shall be the representation of the public-key reconstruction data *BEU* as specified in the implicit certificate generation protocol, which is an elliptic curve point as specified in Clauses 11.3.1.1 and 11.3.1.2 {xref};
   c. **Subject** shall be the identifier of the entity *U* that is bound to the public-key reconstruction data *BEU* during the execution of the implicit certificate generation protocol;
   d. **Issuer** shall be the identifier of the entity *CA* that creates the implicit certificate during the execution of the implicit certificate generation protocol (the so-called Certificate Authority).
5. The format of the string I_U as specified in Step 6 of the actions of the CA in the implicit certificate generation protocol is specified as follows: I_U=(**Subject || Issuer**), where **Subject** and **Issuer** are as specified above.

11.3.1.6.3 ECC X509 certificates

This standard shall use the ECC X509 certificate scheme, with associated ECC X509 certificate generation protocol and ECC X509 certificate processing protocol, as specified in PKIX [{xref} RFC 2459, {xref} RFC 3279, {xref}RFC 3280]. These certificates shall contain an elliptic curve public key as specified in Clause 11.3.1.3. The certificate shall use the signing algorithm ECDSA as specified in Clause 11.3.1.6.3.1. For the detailed ASN.1 specification, see Appendix B {xref}.

11.3.1.6.3.1 ECDSA Digital Signatures

The ECC X509v3 certificate scheme shall use the ECDSA signature scheme, with associated signature generation protocol and signature verification protocol, as announced in FIPS Pub 186-2 and fully specified in ANSI X9.62-1998 [{xref} X9.62]. The ECDSA key generation protocol shall use the random number generation algorithm as specified in FIPS Pub 186-2, including Change Notice 1 [{xref} FIPS186-2 Rev].

11.3.2 ECMQV Raw 1 sub-mode

ECMQV Raw 1 is a sub-mode of Mode 2 of the ECMQV 283-Koblitz-1 security suite. The cryptographic building blocks for ECMQV Raw 1 are selected from the public-key cryptographic building blocks for the ECMQV 283-Koblitz-1 security suite. The object identifier OID for this mode is specified as in Table 2 {xref} of Clause 14.2.1 {xref}. The following sub-clauses specify the public-key and authentication related objects for this sub-mode.

11.3.2.1 Public-key and authentication data formats

The following table specifies the length and intended meaning of the public-key cryptography and authentication related security suite specific data elements from Clause 7 {xref}. The operations to obtain the variable data values are specified in a separate sub-clause.

Table 12 – Public-key frame object formats

Notation	Length	Value	Description
PublicKeyObjectType	2	See Table 7.4.1.2 {xref}	The numerical value of the ECMQV-Raw-1 public key type in Table 7.4.1.2{xref}
PublicKeyObjectLength	2	43	Length of the ECC Manual Certificate
PublicKeyObject	43	Manual certificate value	Instance of ECC Manual certificate as specified in Clause 11.3.1.6.1
OIDLength	1	10	Length of the DER encoding of the applicable OID
OID	10	OID value	DER encoding of the OID ECMQV-Raw-1 as specified in Table 2 of Clause 11.2.1 {xref}
ChallengeType	2	See Table 7.4.1.3 {xref}	The numerical value of the ChallengeType ephemeral elliptic curve public key in Table 7.4.1.3 {xref}
ChallengeLength	2	37	Length of ephemeral public key
Challenge	37	Point value	Ephemeral public key as specified in Clause 11.3.1.3

11.3.2.2 Public-key and authentication cryptographic operations

The following table specifies the public key cryptography and authentication related operations for the authentication protocol frames as specified in Clause 7 {xref}:

Table 13 – Authentication-related functions

Use	Description
Authorization of	Verification of authorization status of the other party, by checking whether

public key	the public key information as contained in the manual certificate as specified in Clause 11.3.1.6.1 complies with information in the access control list (ACL).
Challenge generation	Communication of random short-term (ephemeral) public key to the other party, to be used for key establishment and for entity authentication.
Key establishment	Derivation of a shared key, based on the ephemeral public keys contributed by both parties and long-term (static) public-keying material of both parties.
Key authentication	n.a.
Proof of possession of private key	Verification that the other party that communicated a static public key possesses the corresponding private key (as witnessed by successful key confirmation).
Mutual key confirmation	Verification of possession of the shared key by the other party, via the computation of a message authentication code over part of the data communicated with the other party.
Mutual entity authentication	Verification of the actual and real-time involvement of the other party during the key agreement protocol.

11.3.3 ECMQV Implicit 1 sub-mode

ECMQV Implicit 1 is a sub-mode of Mode 3 of the ECMQV 283-Koblitz-1 security suite. The cryptographic building blocks for ECMQV Implicit 1 are selected from the public-key cryptographic building blocks for the ECMQV 283-Koblitz-1 security suite. The object identifier OID for this mode is specified as in Table 2 {xref} of Clause 14.2.1. The following sub-clauses specify the public-key and authentication related objects for this sub-mode.

11.3.3.1 Public-key and authentication data formats

The following table specifies the length and intended meaning of the public-key cryptography and authentication related security suite specific data elements from Clause 7 {xref}. The operations to obtain the variable data values are specified in a separate sub-clause.

Table 14 – Public-key frame object formats

Notation	Length	Value	Description
PublicKeyObjectType	2	See Table 7.4.1.2 {xref}	The numerical value of the ECMQV Implicit 1 public key type in Table 7.4.1.2{xref}
PublicKeyObjectLength	2	49	Length of the ECC Implicit Certificate
PublicKeyObject	49	Implicit certificate value	Instance of ECC Implicit certificate as specified in Clause 11.3.1.6.2
OIDLength	1	10	Length of the DER encoding of the applicable OID
OID	10	OID value	DER encoding of the OID ECMQV

			Implicit 1 as specified in Table 2 of Clause 11.2.1 {xref}
ChallengeType	2	See Table 7.4.1.3 {xref}	The numerical value of the ChallengeType ephemeral elliptic curve public key in Table 7.4.1.3 {xref}
ChallengeLength	2	37	Length of ephemeral public key
Challenge	37	Point value	Ephemeral public key as specified in Clause 11.3.1.3

11.3.3.2 Public-key and authentication cryptographic operations

The following table specifies the public key cryptography and authentication related operations for the authentication protocol frames as specified in Clause 7 {xref}:

Table 15 – Authentication-related functions

Use	Description
Authorization of public key	Verification of authorization status of the other party, by checking whether the public key information as contained in the implicit certificate as specified in Clause 11.3.1.6.2 complies with information in the access control list (ACL).
Challenge generation	Communication of random short-term (ephemeral) public key to the other party, to be used for key establishment and for entity authentication.
Key establishment	Derivation of a shared key, based on the ephemeral public keys contributed by both parties and long-term (static) public-keying material of both parties.
Key authentication	Verification of the authenticity of the static key of the other party via reconstruction of the public key hereof from the implicit certificate communicated by the other party and evidence of possession of the corresponding private key hereby (as witnessed by successful key confirmation).
Mutual key confirmation	Verification of possession of the shared key by the other party, via the computation of a message authentication code over part of the data communicated with the other party.
Mutual entity authentication	Verification of the actual and real-time involvement of the other party during the authenticated key agreement protocol.

11.3.4 ECMQV X509 1 sub-mode

ECMQV X509 1 is a sub-mode of Mode 3 of the ECMQV 283-Koblitz-1 security suite. The cryptographic building blocks for ECMQV X509 1 are selected from the public-key cryptographic building blocks for the ECMQV 283-Koblitz-1 security suite. The object identifier OID for this mode is specified as in Table 2 {xref} of Clause 11.2.1. The following sub-clauses specify the public-key and authentication related objects for this sub-mode.

11.3.4.1 Public-key and authentication data formats

The following table specifies the length and intended meaning of the public-key cryptography and authentication related security suite specific data elements from Clause 7 {xref}. The operations to obtain the variable data values are specified in a separate sub-clause.

Table 16 – Public-key frame object formats

Notation	Length	Value	Description
PublicKeyObjectType	2	See Table 7.4.1.2 {xref}	The numerical value of the ECMQV X509 1 public key type in Table 7.4.1.2{xref}
PublicKeyObjectLengt h	2	Variable	Length of the ECC X509 Certificate
PublicKeyObject	Variabl e	X509 certificate value	Instance of ECC X509 certificate as specified in Clause 11.3.1.6.3
OIDLength	1	10	Length of the DER encoding of the applicable OID
OID	10	OID value	DER encoding of the OID ECMQV X509 1 as specified in Table 2 of Clause 11.2.1 {xref}
ChallengeType	2	See Table 7.4.1.3 {xref}	The numerical value of the ChallengeType ephemeral elliptic curve public key in Table 7.4.1.3 {xref}
ChallengeLength	2	37	Length of ephemeral public key
Challenge	37	Point value	Ephemeral public key as specified in Clause 11.3.1.3

11.3.4.2 Public-key and authentication cryptographic operations

The following table specifies the public key cryptography and authentication related operations for the authentication protocol frames as specified in Clause 7 {xref}:

Table 17 – Authentication-related functions

Use	Description
Authorization of public key	Verification of authorization status of the other party, by checking whether the public key information as contained in the X509 certificate as specified in Clause 11.3.1.6.3 complies with information in the access control list (ACL).
Challenge generation	Communication of random short-term (ephemeral) public key to the other party, to be used for key establishment and for entity authentication.
Key establishment	Derivation of a shared key, based on the ephemeral public keys contributed by both parties and long-term (static) public-keying material of both parties.
Key authentication	Verification of the authenticity of the static key of the other party via verification of the signature provided in the X509 certificate communicated by the other party.
Mutual key	Verification of possession of the shared key by the other party, via the

confirmation	computation of a message authentication code over part of the data communicated with the other party.
Mutual entity authentication	Verification of the actual and real-time involvement of the other party during the authenticated key agreement protocol.

APPENDIX A

11.4 Implicit Certificate Scheme

This section specifies the ECQV implicit certificate scheme based on ECC supported in this standard.

Implicit certificate schemes are designed to be used by three entities – a Certification Authority *CA*, a certificate requester *U*, and a certificate processor *V*, where *U* wants to obtain an implicit certificate from *CA* in order to convey *U*'s public key to *V*.

Here implicit certificate schemes are described in terms of a certificate generation protocol, a certificate processing operation, and associated setup and *CA* key deployment procedures. *CA*, *U*, and *V* shall use the schemes as follows, when they wish to communicate.

CA, *U*, and *V* should use the setup procedure to establish which options to use the scheme with. *CA* should use the key deployment procedure to select a key pair and *U* and *V* should obtain *CA*'s public key – *CA* will use the key pair during the certificate generation protocol, *U* will use the public key during the certificate generation protocol, and *V* will use the public key during the certificate processing operation. When *U* wants to obtain an implicit certificate, *U* and *CA* should perform the certificate generation protocol to obtain a key pair known to *U* and an implicit certificate *IC*. Finally, when *V* wants to obtain *U*'s public key, *U* should convey *IC* to *V* and *V* should apply the certificate processing to *IC* under *CA*'s public key to obtain *U*'s public key. *V* concludes that the public key is genuine, provided *U* provides evidence that it possesses the corresponding private key.

The specification of ECQV in this document relies heavily on the mathematical foundations and cryptographic components specified in Sections 2 and 3 of [SEC1].

The setup procedure for ECQV is specified in Section 8.1, the key deployment procedure is specified in Section 8.2, the certificate generation protocol is specified in Section 8.3, and the certificate processing operation is specified in Section 8.4.

11.4.1 Scheme Setup

CA, *U*, and *V* shall perform the following setup procedure to prepare for the use of ECQV.

1. An infrastructure shall have been established for the operation of the scheme – including a certificate format, certificate processing rules, and unique identifiers. An example of such an infrastructure is described by PKIX [PKIX-X509].

2. Each entity has an authentic copy of the system's elliptic curve domain parameters $D=(p,a,b,G,n,h)$ or $D=(m,f(x),a,b,G,n,h)$. These parameters shall have been generated using the parameter generation primitive in Sections 3.1.1.1 or the primitive specified in Section

3.1.2.1, both of [SEC1]. Furthermore, the parameters shall have been validated using the parameter validation primitives in Sections 3.1.1.2 or Section 3.1.2.2 of [SEC1].

3. The *CA* shall have decided which cryptographic hash function to use when generating implicit certificates. Let *Hash* denote the hash function chosen, and let *hashlen* denote the length in bits of the output value of this hash function. Each entity shall have an authentic copy of this hash function.

4. Each entity shall be bound to a unique identifier (e.g. distinguished names). All identifiers shall be bit strings of the same length *entlen* bits. Entity *U*'s identifier will be denoted by the bit string *U*. Entity *V*'s identifier will be denoted by the bit string *V*. Entity *CA*'s identifier will be denoted by the bit string *CA.*

5. The *CA* shall be bound to a static public key pair associated with the system's elliptic curve domain parameters *D*. The binding process shall include the validation of the static public key as specified in Section 3.2.2 of [SEC1]. The key binding shall include the unique identifier *CA* for the entity involved.

6. Each entity shall have decided how to represent elliptic curve points as octet strings (i.e., compressed form, uncompressed form, or hybrid form).

11.4.2 Key Deployment

CA, *U*, and *V* shall use the following key deployment procedure to prepare for the use of ECQV.

1. *CA* shall establish a static elliptic curve key pair (W_{CA}, w_{CA}) associated with *D* to use with the certificate generation and processing protocols. The key pair shall be generated using the primitive specified in Section 3.2.1 of SEC 1.

2. *U* and *V* shall obtain in an authentic manner the elliptic curve public key W_{CA} selected by *CA*.

11.4.3 Implicit Certificate Generation Protocol

This section specifies the protocol for generating implicit certificates (self-certified public keys).

Figure 12 illustrates the messaging involved in the use of the certificate generation protocol. *CA* and *U* shall generate an implicit certificate for *U* using the keys and parameters established

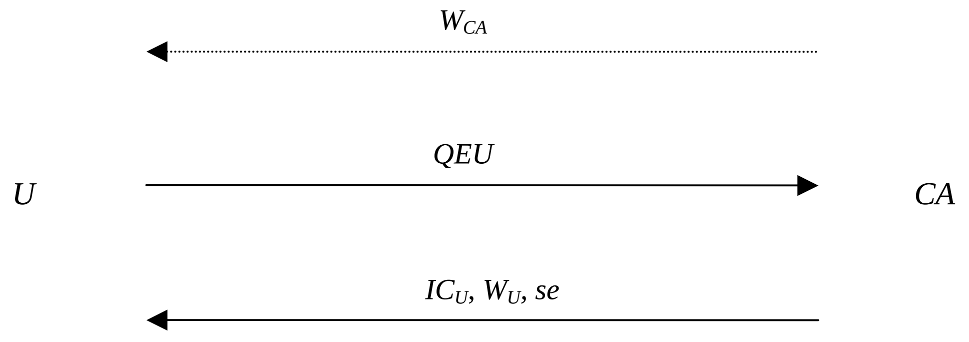

Figure 12 – Implicit Certificate Generation Scheme

during the setup procedure and the key deployment procedure as follows:

Input: None.

Output for CA: An octet string IC_U, which is U's implicit certificate, and an elliptic curve public key W_U corresponding to the elliptic curve domain parameters D, which is U's public key.

Output for U: An octet string IC_U, which is U's implicit certificate, and an elliptic curve key pair (W_U, w_U) corresponding to the elliptic curve domain parameters D, which is U's public key pair.

Actions: *U* proceeds as follows:

1. Select a random ephemeral elliptic curve key pair (Q_U, q_U) associated with the elliptic curve domain parameters D. The key pair shall be generated using the primitive specified in Section 3.2.1 of SEC 1. (q_U is referred to as U's private request value; Q_U as U's public request value.)

2. Convert Q_U to an octet string *QEU* as specified in Section 2.3.5 in SEC 1, and send it to *CA*.

3. Obtain the values IC_U and *se* from *CA*. Compute the hash value $H = Hash(IC_U)$ using the established hash function. Derive an integer e from H following the conversion routine of Step 5 in Section 4.1.3 of [SEC1].

4. Verify the content of IC_U according to the established infrastructure. This includes verifying the contents of the certificate, e.g., the subject’s name and the validity period.

5. Derive *BEU* and I_U from IC_U according to the procedures of the established infrastructure.

6. Derive W_{CA} from I_U, according to the certificate format specified during the setup procedure.

7. Convert *BEU* to an elliptic curve point B_U as specified in Section 2.3.4 of SEC 1 and convert *se* to an integer s as specified in Section 2.3.8 of SEC 1.

8. Compute the secret key $w_U := s + q_U \,.\, e \ (\mathbf{mod}\ n)$ and the public key $W_U := w_U\, G$.

9. Reconstruct the public key $W_U' = e\, B_U + W_{CA}$.

10. If $W_U' = W_U$, accept the certificate and output the computed key pair (W_U, w_U) and the implicit certificate IC_U; otherwise output *reject*.

Actions: *CA* proceeds as follows:

1. Verify the authenticity of the request received from *U* according to the procedures of the established infrastructure. The checks performed shall include, as a minimum, checking that *U* is indeed the origin of the request, and checking that *U* is authorized to obtain a certificate.

2. Receive *QEU* from *U* and convert it to an elliptic curve point Q_U as specified in Section 2.3.4 in SEC 1.

3. Select an ephemeral elliptic curve key pair (Q_{CA}, q_{CA}) associated with the elliptic curve domain parameters *D*. The key pair shall be generated using the primitive specified in Section 3.2.1 of [SEC1].

4. Compute the elliptic curve point $B_U := Q_U + Q_{CA}$.

5. Convert the elliptic curve point B_U to an octet string *BEU* as specified in Section 2.3.3 in SEC 1.

6. Construct the ‘to-be-signed-certificate’ data, which is an octet string I_U. I_U shall contain identification information according to the procedures of the established infrastructure and may also contain other information, such as the intended use of the public key, the serial number of the implicit certificate, and the validity period of the implicit certificate. The exact form of I_U depends on the certificate format being used.

7. Construct according to the procedures of the established infrastructure *U*'s implicit certificate, which is an octet string IC_U. IC_U should contain I_U and *BEU*. The exact form of IC_U depends on the certificate format being used.

8. Compute the hash value $H = Hash(IC_U)$ using the established hash function. Derive an integer *e* from *H* following the conversion routine of Step 5 in Section 4.1.3 of [SEC1].

9. Compute the integer $s = q_{CA}\, e + w_{CA} \ (\mathbf{mod}\ n)$. (*s* is referred to as the private-key reconstruction data.)

10. Convert *s* to an octet string *se* as specified in Section 2.3.7 of SEC 1, and send IC_U and *se* to *U*.

11. Compute $W_U = e\, B_U + W_{CA}$. Output the implicit certificate IC_U and the elliptic curve public key W_U.

11.4.4 Implicit Certificate Processing Operation

V shall process *U*'s implicit certificate using the keys and parameters established during the setup procedure and the key deployment procedure as follows:

Input: *U*'s purported implicit certificate IC_U.

Output: *U*'s purported public key W_U.

Actions: *V* proceeds as follows:

1. Verify the content of IC_U according to the established infrastructure. This includes verifying the contents of the certificate, such as the subject’s name and the validity period.

2. Derive *BEU* and I_U from IC_U, according to the certificate format specified during the setup procedure.

3. Derive W_{CA} from I_U, according to the certificate format specified during the setup procedure.

4. Convert the octet string *BEU* to an elliptic curve point B_U as specified in Section 2.3.4 of SEC 1.

5. Compute the hash value $H = Hash(IC_U)$ using the established hash function. Derive an integer e from H following the conversion routine of Step 5 in Section 4.1.3 of [SEC1].

6. Compute the public key $W_U = e\ B_U + W_{CA}$.

After performing the certificate processing operation, *V* can conclude that W_U is genuine, provided *U* evidences knowledge of the corresponding private key w_U.

APPENDIX B – ECC X509 Certificates

The IETF profile PKIX of X.509 certificates can be found in RFC 2459 and in the draft RFC 3280 (soon to be published). The various algorithm identifiers necessary to use ECC with PKIX can be found in the draft RFC 3278 (soon to be published). In this standard, the values of specific variable fields will be considerably restricted, as follows.

In this standard, all optional fields in **TBSCertificate** shall be omitted. These fields are **IssuerUniqueID**, **subjectUniqueID,** and **extensions**.

In this standard, certain fields in **TBSCertificate** and **Certificate** shall be fixed:

1. The **version** field in **TBSCertificate** shall be Version 1 (value: 0) certificates.

2. The **signature** field in **TBSCertificate** and the **signatureAlgorithm** field in **Certificate** both have the same value. As specified in RFC 3278 for certificates signed using ECDSA, both these fields shall be the **AlgorithmIdentifier** whose **algorithm** field is the object identifier **ecdsa-with-SHA1** and whose **parameters** field is omitted. The value of both these fields is thus a sequence 1 element, the object identifier above. (Note: The object identifier above is **ecdsa-with-SHA1**. For this standard, ECDSA with SHA-256, rather than ECDSA with SHA-1, shall be used. Here, SHA-256 is as specified in FIPS Pub 180-2 [{xref}FIPS 180-2]. At this moment, **ecdsa-with-SHA-256** does not have an assigned object identifier yet, however.)

3. The **validity** field in **TBSCertificate** shall have a fixed value, as follows. The default validity period shall be from the first second of January 1, 2000 until the last second of December 31, 3000. **GeneralizedTime** shall be used.

4. The **algorithm** field in **SubjectPublicKeyInfo** in **TBSCertificate** has a fixed value. As specified in RFC 3278 for certificates signed with ECDSA, this field is an **AlgorithmIdentifier** whose **algorithm** field is the object identifier **id-ecPublicKey** and whose **parameters** field has type **EcpkParameters**. In this standard, the **ImplicitlyCA** choice for **EcpkParameters** shall be used.

In this standard, the following fields shall be determined in a particular way, as follows.
The fields **issuer** and **subject** in **TBSCertificate** are both of ASN.1 type **Name** and shall include exactly one attribute that uniquely identifies the issuer and the subject, respectively. The **type** field of this attribute shall be the object identifier

device-id ::= { ieee-802-15-3 xx yy},

where **ieee-802-15-3** is the object identifier for IEEE 802.15.3 and where **xx** and **yy** are two pre-determined values (we take **xx**=1 and **yy**=1). The **value** field of this attribute shall have type

OCTET STRING. The octet string shall be the unique 48-bit IEEE MAC address of the issuer or subject.

APPENDIX C – References

12. Definitions, Abbreviations, and References

12.1 Definitions and Abbreviations

Our definitions follow those as defined in [ANSI X9.63-2001, §2.1].

12.2 Symbols and Notation

Our notation follows [ANSI X9.63-2001, §2.2].

12.2.1 Normative References

1. ANSI X9.30-1997, The Digital Signature Algorithm (DSA) (Revised), Annex F: Parameter Settings and Security, American Bankers Association, July 1, 1999.
2. ANSI X9.62-1998, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), American Bankers Association, January 7, 1999.
3. ANSI X9.63-2001, Public Key Cryptography for the Financial Services Industry – Key Agreement and Key Transport Using Elliptic Curve Cryptography, American Bankers Association, November 20, 2001.
4. FIPS Pub 180-1, Secure Hash Standard, Federal Information Processing Standards Publication 180-1, US Department of Commerce/N.I.S.T., Springfield, Virginia, April 17, 1995 (supersedes FIPS Pub 180). Available from http://csrc.nist.gov/.
5. FIPS Pub 180-2, Draft Specification for the Secure Hash Standard, Federal Information Processing Standards Publication 180-2, US Department of Commerce/N.I.S.T., draft, 2001. Available from http://csrc.nist.gov/.
6. FIPS Pub 186-2, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-2, US Department of Commerce/N.I.S.T., Springfield, Virginia, January 27, 2000. Available from http://csrc.nist.gov/.
7. FIPS Pub 197, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, US Department of Commerce/N.I.S.T, November 26, 2001. Available from http://csrc.nist.gov/.
8. FIPS Pub #HMAC, Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards Publication #HMAC, US Department of Commerce/N.I.S.T., draft, May 30, 2001. Available from http://csrc.nist.gov/.
9. ISO/IEC 9798-1, Information Technology - Security Techniques - Entity Authentication Mechanisms – Part 1: General Model, International Standardization Organization, Geneva, Switzerland, 1991 (first edition).
10. ISO/IEC 9798-2, Information Technology - Security Techniques - Entity Authentication Mechanisms – Part 2: Mechanisms Using Symmetric Encipherment Algorithms, International Standardization Organization, Geneva, Switzerland, 1994 (first edition).

11. ISO/IEC 9798-3, Information Technology - Security Techniques - Entity Authentication Mechanisms – Part 3: Entity Authentication using a Public Key Algorithm, International Standardization Organization, Geneva, Switzerland, 1993 (first edition).
12. NIST Pub 800-38A 2001 ED, Recommendation for Block Cipher Modes of Operation – Methods and Techniques, NIST Special Publication 800-38A, 2001 Edition, US Department of Commerce/N.I.S.T., December 2001. Available from http://csrc.nist.gov/.
13. PKIX, L. Bassham, R. Housley, W. Polk, Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and CRL Profile, Internet Draft, PKIX Working Group, October 2001.
14. RFC 2104, HMAC: Keyed-Hashing for Message Authentication, Internet Request for Comments 2104, H. Krawczyk, M. Bellare, R. Canetti, February 1997.
15. RFC 2119, Key Words for Use in RFCs to Indicate Requirement Levels, Internet Request for Comments 2119, S. Bradner, March 1997.
16. RFC 2459, R. Housley, W. Ford, W. Polk, D. Solo, 'Internet X.509 Public Key Infrastructure: Certificate and CRL Profile', January 1999.
17. RFC 3279, L. Bassham, R. Housley, W. Polk, 'Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and CRL profile', PKIX Working Group Internet-Draft, October 2001. See also draft-ietf-pkix-ipki-pkalgs-05.txt.
18. RFC 3280, W. Ford, R. Housley, W. Polk, D. Solo, 'Internet X.509 Public Key Infrastructure Certificate and CRL Profile', PKIX Working Group Internet-Draft, January 2002. See also draft-ietf-pkix-new-part1-12.txt.
19. Standards for Efficient Cryptography, SEC 1: Elliptic Curve Cryptography, Version 1.0, Certicom Research, September 20, 2000. Available from http://www.secg.org/.
20. Standards for Efficient Cryptography, SEC 2: Recommended Elliptic Curve Domain Parameters, Version 1.0, Certicom Research, September 20, 2000. Available from http://www.secg.org/.

12.2.2 Informative References

1. D.R.L. Brown, R. Gallant, S.A. Vanstone, Provably Secure Implicit Certificate Schemes, in *Proceedings of Financial Cryptography 2001*, to appear.
2. IEEE Draft P802.15.3/D09, Draft Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Are Networks Specific Requirements – Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPAN), Draft P802.15.3/09, December 7, 2001.
3. L. Law, A.J. Menezes, M. Qu, J. Solinas, S.A. Vanstone, An Efficient Protocol for Authenticated Key Agreement, Technical Report CORR 1998-05, CACR, University of Waterloo, 1998. Available from http://www.cacr.math.uwaterloo.ca.
4. A.K. Lenstra, Unbelievable Security: Matching AES Security using Public Key Systems, in *Proceedings of Advances in Cryptology – ASIACRYPT 2001*, December 10-13, 2001.
5. A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, Boca Raton: CRC Press, 1997.
6. L.A. Pintsov, S.A. Vanstone, Postal Revenue Collection in the Digital Age, Technical Report CORR 2000-43, CACR, University of Waterloo, 2000. Available from http://www.cacr.math.uwaterloo.ca.