Cryptography Section

1. Introduction/Scope

This section covers requirements for voting systems that use cryptographic technology to provide basic security services such as confidentiality, data integrity, authentication, etc. As some cryptographic techniques require the use of keying material, this section will cover the capabilities of voting systems to support the management of cryptographic keys. In general, a single cryptographic mechanism may provide more than one security service (e.g., the use of digital signatures can provide integrity and authentication) but not all security services to a voting system. Requirements for cryptographic voting protocols are not described in this section but can be found in the Independent Verification (IV) section of the Voluntary Voting System Guidelines (VVSG). The requirements found in this section are derived from requirements found in commercial and federal standard such as American National Standards Institute (ANSI) X9.63-2005: Public Key Cryptography: The Elliptic Curve Digital Signature Algorithm (ECDSA), NIST Special Publication 800-57: Recommendation for Key Management, Federal Information Processing Standard (FIPS) 140-2: Security Requirements for Cryptographic Modules, etc.

2. Cryptography Basics

This section provides a brief overview of some basic cryptography topics including the types of security services supported by cryptography, the different types of cryptographic algorithms, and how the security strengths of cryptographic algorithms are quantified.

2.1 Security Services Provided by Cryptography

Voting systems have several security requirements that can be met through the use of cryptographic techniques when implemented properly. The following are just a few examples of the security services a voting system might need that could be addressed with cryptographic techniques. Once a ballot has been cast in a voting system, the ballot needs to be protected from disclosure to insure that vote buying or a voter's privacy is not violated. Cast ballots also need to be protected from modification to ensure that a voter's choices are not changed once the voter has cast their ballot. In addition to the protection of cast ballots, voting systems rely on vital information/data (such as ballot definition information and other software/firmware) for proper operation. Vital information/data needed for the proper operation of a voting system not only needs to be protected from modification, but the source that created it needs to be verified. The rest of this section describes the security services that cryptographic techniques, when implemented properly, can provide to voting systems.

2.1.1 Confidentiality

Confidentiality, or secrecy, is the property whereby information or data is not disclosed to unauthorized parties. The confidentiality property can be used to help protect cast ballots from being used to buy votes and to assure privacy of a voter's choices.

Confidentiality is achieved using encryption to render the information or data unintelligible except by authorized entities. The information or data may become intelligible again by using decryption. In order for encryption to provide confidentiality, the cryptographic algorithm and mode of operation needs to be designed and implemented so that an unauthorized party cannot determine the secret keys associated with the encryption or be able to derive the plaintext directly without deriving any keys.

2.1.2 Data Integrity

Data integrity is the property whereby data has not been altered (by insertion, deletion, or substitution) in an unauthorized manner since creation, transmission or storage. The data integrity property can be used to protect cast ballots from modification to ensure that a voter's choices are not changed once the voter has cast their ballot. Cryptographic mechanisms, such as hash values or digital signatures, can be used to detect both accidental modifications (e.g., modifications that sometimes occur during noisy transmissions or by hardware memory failures), and deliberate modifications (by an adversary) with a very high probability. Non-cryptographic mechanisms, such as parity checksums or cyclic redundancy codes (CRCs), are also often used to detect accidental modifications, but cannot be relied upon to detect deliberate modifications. This section of VVSG focuses only on requirements for cryptographic mechanisms that provide data integrity.

2.1.3 Authentication

The service used to establish the origin of information is known as authentication. Authentication services verify the identity of the user or system that created information (e.g., a transaction or message). The authentication services can be used to determine the origin of vital information required for the proper operation of the voting system such as the source of ballot definition files for a specific election. Several cryptographic mechanisms may be used to provide authentication services. Most commonly, authentication is provided by digital signatures or message authentication codes; some key agreement techniques also provide authentication.

2.2 Types of Cryptographic Algorithms

Cryptographic hash algorithms (i.e., hash functions) are a type of cryptographic algorithm that does not require any keys. Hash functions generate a relatively small message digest or hash values from a (possibly) large input in a way that is fundamentally difficult to reverse (i.e., hard to find an input that will produce a given output). Hash functions are used as building blocks for key management and digital signatures, for example,

1. To provide data authentication and integrity services when they are used with keys to generate message authentication codes,
2. To compress messages for digital signature generation and verification,
3. To derive keying material,

4. To generate deterministic random numbers (e.g. pseudo random numbers).

Symmetric or secret key algorithms transform data in a way that is fundamentally difficult to undo without knowledge of a secret key. The key is "symmetric" because one key is used for encryption and decryption operations. Symmetric key algorithms are used, for example,

1. To provide data confidentiality - the same key is used to encrypt and decrypt data;
2. To provide authentication and integrity services in the form of Message Authentication Codes (MACs) (See Section 3.2),
3. As part of the key establishment process and,
4. To generate deterministic random numbers (e.g. pseudo random numbers).

Asymmetric or public key algorithms use two related keys – a public key and a private key - to perform cryptographic functions. Even though the public and private keys of a key pair are related, knowledge of the public key does not reveal the private key. The public key may be known by anyone. However, the private key is under the sole control of the entity that "owns" the key pair. Public key algorithms are used, for example,

1. To compute digital signatures,
2. To establish cryptographic keying material,
3. To generate deterministic random numbers (e.g. pseudo random numbers).

2.3 Security Services Supported By Cryptographic Algorithm Type

The following table combines the information provided in Sections 2.1 and 2.2 to map the types of cryptographic algorithms to the security services supported:

	Data Authentication	Integrity Services	Data Confidentiality	Generating Keying Material	Generating Random Numbers
Hash algorithms		Yes		Yes	Yes
Asymmetric or Public key algorithms	Yes			Yes	Yes
Symmetric or Secret key algorithms	Yes (in MACs)	Yes	Yes	Yes	Yes

2.4 Quantifying Security Strengths of Cryptographic Algorithms

Cryptographic algorithms provide different "strengths" of security, depending on the algorithm and the key size used. Two algorithms are considered to be of comparable

strength for the given key sizes (X and Y) if the amount of work needed to "break the algorithms" or determine the keys (with the given key sizes) is approximately the same using a given resource. The security strength of an algorithm for a given key size is traditionally described in terms of the amount of work it takes to try all keys for a symmetric algorithm with a key size of "X" that has no short cut attacks (i.e., the most efficient attack is to try all possible keys). In this case, the best attack is said to be the exhaustion attack. An algorithm that has a "Y" bit key, but whose strength is comparable to an "X" bit key of such a symmetric algorithm is said to have a "security strength of X bits" or to provide "X bits of security". Given a few plaintext blocks and corresponding ciphertext, an algorithm that provides X bits of security would, on average, take $2^{X-1}T$ of time to attack, where T is the amount of time that is required to perform one encryption of a plaintext value and comparison of the result against the corresponding ciphertext value.

2.5 Cryptographic Key Management

Some cryptographic algorithms require the use of keys to perform cryptographic operations. Cryptographic keys need to change over time (due to key compromises, advances in cryptographic technology, the purpose for its use, etc.) to ensure the cryptographic algorithm continues to provide the expected level of security. Since cryptographic keys need to change, they have an associated lifecycle. The initial stage in a key's lifecycle is the actual generation of the key. The key needs to be generated according to the requirements associated with the key's cryptographic algorithm to prevent the use of weak keys. Keys may go through a formal registration or activation process before they are deemed fit for use. Once keys are generated (and if necessary formally approved for use), the appropriate keys need to be securely distributed to other parties (humans, machines, applications, etc.) that will use the keys to support various security services. When keys are not being used, they need to be securely stored so that they are not modified or revealed to inappropriate parties. When keys are being used, they need to be used in an appropriate manner for the purpose (authentication, confidentiality, etc.) for which they were created. Once keys have reached the end of their lifecycle or become compromised, keys need to be properly destroyed (and if necessary formally deactivated or deregistered) so the key will not continue to be used. In addition, some cryptographic keys may need to be archived to ensure access to data that has been encrypted and/or to verify digital signatures. Given the lifecycle of generation, registration, distribution, usage, storage, deregistration, destruction, and archival, cryptographic keys need to be properly managed in order to deliver the security services provided by cryptographic techniques effectively. Section 3.5 will describe the voting system capabilities required to support key management.

3. Cryptography Requirements for Voting Systems

The requirements found in this section apply to voting systems that implement cryptographic technology to provide basic security services as described in Section 2.

3.1 Cryptographic Algorithm Requirements

3.1.1 General Requirements

3.1.1.1 Only Federal Information Processing Standard (FIPS) approved or National Institute of Standards and Technology (NIST) recommended cryptographic algorithms **shall** be used to support the basic security services (see Section 2.1) for voting systems.

Discussion: The list of FIPS-approved and NIST recommended cryptographic algorithms is subject to change over time, due to advancements in computing technology and cryptography. Information on the FIPS approved and NIST recommended cryptographic algorithms can be found at http://csrc.nist.gov/CryptoToolkit/.

3.1.1.2 Vendors **shall** provide a description of the FIPS-approved and NIST recommended cryptographic algorithms supported by the voting system, including the security strength of the algorithm and key sizes, in the technical date package (TDP) and user documentation.

3.1.1.3 Vendor **shall** provide a description of the security services provided by the cryptographic algorithms for the voting system including the detailed technical specification of how the cryptographic algorithm(s) supports the associated security service(s) in the TDP.

3.1.1.4 Vendors **shall** provide a description of how to configure the cryptographic algorithms including different key sizes supported by the voting system in user documentation.

3.1.2 Hash Functions

Five FIPS-approved or NIST recommended hash functions are specified in [FIPS 180-2] based on the Secure Hash Algorithm (SHA) family of hash functions. Each hash function provides different security strengths. The following is a table of the different hash functions with their corresponding security strengths:

Bits of Security	Hashing Functions
See 3.1.2.1	SHA-1
112	SHA-224
128	SHA-256
192	SHA-384
256	SHA-512

The list of FIPS-approved or NIST recommended hash functions is subject to change over time, due to advancements in computing technology and cryptography. Information on the list of FIPS-approved or NIST recommended hash functions can be found at http://csrc.nist.gov/CryptoToolkit/tkhash.html.

3.1.2.1 Voting systems **shall** only used SHA-1 for the generation of hash message authentication codes (HMACs), key derivation functions (KDFs), deterministic random number generators (e.g. pseudo random numbers), and hash value reference information (See Reference Information Generation Section).

Discussion: SHA-1 is susceptible to a technical attack called a second pre-image attack so must not be used to support security services (such as digital signatures) that require resistance to second pre-image attacks. Since determining the need for resistance to second pre-image attacks requires highly specialized knowledge about cryptography, the use of SHA-1 is strongly discouraged. In the case that resistance to second pre-image attacks are not a concern, SHA-1 provides 160 bits of security.

3.1.3 Digital Signature Algorithms

FIPS-approved or NIST recommended Digital Signature Algorithms are specified in [FIPS 186-3]. Information on the FIPS-approved or NIST recommended digital signature algorithms is subject to change over time, due to advancements in computing technology and cryptography. Information on the FIPS-approved and NIST recommended digital signature algorithms can be found at http://csrc.nist.gov/CryptoToolkit/tkdigsigs.html. Digital signature algorithms require the use of a public key or asymmetric algorithm and a hash function as part of the signature generation and verification process. The following subsections identify FIPS-approved or NIST recommended public key algorithms that can be used to generate and verify digital signatures. See Section 3.1.2 for the FIPS-approved or NIST recommended hash functions that can be used for digital signatures. See Section 3.4 for security strength requirements when using different types of cryptographic algorithms in combination, such as a public key algorithm and hash function for the generation and verification of digital signatures.

3.1.3.1 Rivest, Shamir and Adelman (RSA) Digital Signature Algorithm

The Rivest, Shamir and Adelman (RSA) Digital Signature Algorithm is a FIPS-approved or NIST recommended cryptographic algorithm for generating and validating digital signatures. The use of RSA is specified in Federal Information Processing Standards 186-3 [FIPS 186-3] and [PKCS#1 v2.1]. The following is a table of different security strengths of RSA with different key sizes:

Bits of security	RSA Key Size in bits
112	2048
128	3072

3.1.3.2 Digital Signature Algorithm (DSA)

The Digital Signature Algorithm (DSA) is a FIPS-approved or NIST recommended cryptographic algorithm for generating and validating digital signatures. DSA is specified in Federal Information Processing Standards 186-3 [FIPS 186-3]. The following is a table of different security strengths of DSA with different-size key pairs:

Bits of security	DSA Key Size in bits, L & N
112	L = 2048 N = 224
112	L = 2048 N = 256
128	L = 3072 N = 256

3.1.3.3 Elliptic Curve Digital Signature Algorithm (ECDSA)

The Elliptic Curve Digital Signature Algorithm (ECDSA) is a FIPS approved cryptographic algorithm for generating and validating digital signatures. ECDSA is specified in Federal Information Processing Standard 186-3 [FIPS 186-3], along with NIST recommended elliptic curves that provide different security. The following is a table of different security strengths for the Prime field type NIST recommended elliptic curves:

Bits of security	Prime Field
112	len(p)=224
128	len(p)=256
192	len(p)-384
256	len(p)=512

3.1.4 Symmetric Key Algorithms

FIPS-approved or NIST recommended Symmetric Key Algorithms are specified in [FIPS 197] and [NIST SP 800-67]. Information on the FIPS-approved and NIST recommended symmetric key algorithms is subject to change over time, due to advancements in computing technology and cryptography. Information on the FIPS-approved and NIST recommended symmetric key algorithms can be found at http://csrc.nist.gov/CryptoToolkit/tkencryption.html.

3.1.4.1 Triple Data Encryption Algorithm (TDEA)

The Triple Data Encryption Algorithm (TDEA), also known as Triple Data Encryption Standard (TDES), is specified in NIST Special Publication (SP) 800-67: Recommendation for the Triple Data Encryption Algorithm Block Cipher [NIST SP 800-67]. The three key variation of TDEA (3TDEA) is a FIPS approved or NIST recommended cryptographic algorithm. The following is a table of different security strength of 3TDEA:

Bits of security	***TDEA variation***
112	3TDEA

3.1.4.2 Advance Encryption Standard (AES)

The Advance Encryption Standard (AES) is a FIPS-approved or NIST recommended symmetric key algorithm. Three different AES key sizes are specified in [FIPS 197]: 128, 192, or 256 bits. With keys of different sizes, the AES algorithm provides different levels of security. The following is a table of different security strengths of AES with keys of the three different sizes:

Bits of security	***AES***
128	AES-128
192	AES-192
256	AES-256

3.1.5 Message Authentication Codes (MACs)

Message Authentication Codes (MACs) provide data authentication and integrity. A MAC is a cryptographic checksum on the data that is used to provide assurance that the data has not changed and that the MAC was computed by an entity with the associated key. Two types of algorithms for computing a MAC have been FIPS-approved or NIST recommended: MAC algorithms that are based on block cipher algorithms (i.e., symmetric key), and MAC algorithms that are based on hash functions. [SP800-38] defines a mode to compute a MAC using FIPS-approved or NIST recommended block cipher algorithms. [FIPS198] specifies the computation of a MAC using a FIPS-approved or NIST recommended hash function (also called Hash Message Authentication Code (HMAC))

3.1.5.1 Block ciphers or hash functions used to generate MACs **shall** be a FIPS-approved or NIST recommended cryptographic algorithms. (See Section 3.1)

3.2 Validated Cryptographic Module Requirements

3.2.1 Cryptographic operations **shall** be performed within a FIPS 140-2 level 1 or higher validated cryptographic module.

3.2.2 Vendors **shall** document the FIPS 140-2 cryptographic modules used by the voting system including vendor, product name, version, and FIPS 140-2 certificate number in the TDP.

3.2.3 Vendors **shall** provide a description of the cryptographic operations performed by the FIPS 140-2 validated cryptographic modules including the detailed technical specification of how the cryptographic operations are accessed by the voting system in the TDP.

Discussion: Cryptographic operations include the encryption and decryption of information, the generation and validation of digital signatures, the generation of hash values, cryptographic key generation, and the generation of cryptographic algorithm specific parameters.

3.2.4 Vendors **shall** provide a description of how to invoke the cryptographic operations supported by the voting system in the user documentation.

3.3 Security Strength Lifetime Requirements

Section 2.4 describes how the security strength of a cryptographic technique can be described using a "bits of security" metric. The ability to compromise a fixed security strength provided by a cryptographic technique increases over time, due to changes in computing technology. Therefore, a cryptographic technique with X bits of security might be suitable to protect information when it was initially created and protected, but may not be suitable if the information needs to be protected for long periods of time. The following requirements are provided to address the relationship between the security strength of a cryptographic technique and the amount of time information needs to be protected.

3.3.1 Voting system information with an expected lifetime through 2030 **shall** be protected using cryptographic techniques with a minimum of 112 bits of security.

3.3.2 Voting system information with an expected lifetime beyond 2030 **shall** be protected using cryptographic techniques with a minimum of 128 bits of security.

3.3.3 Vendors **shall** provide a description of the information of the voting system that will be protected by cryptographic techniques and the expected lifetime of the information in the TDP and user documentation.

3.3.4 Vendors **shall** provide a description of how to determine and configure the appropriate the cryptographic techniques to protect voting system information based on the information's lifetime in the user documentation.

Discussion: The following table summarizes the requirements 3.3.1 and 3.3.2:

Bits of Security	Through 2030	Beyond 2030
112	Yes	No
128	Yes	Yes

The following example is provided to illustrate the application of these requirements. If information is encrypted in 2025, and the maximum expected lifetime of that data is only five years, any cryptographic technique that provides at least 112 bits of security may be used. But, if the information is protected in 2025, and the expected lifetime of the data is six years, then any cryptographic technique that provides only 112 bits of security would not be appropriate.

3.4 Cryptographic Algorithm Suite Requirements

Voting systems may require the use of several different cryptographic algorithms in combination to support the basic security services described in Section 2.1. An example is the use of a public key and hash function for digital signatures that supports data integrity and authentication. In addition, if the data requires confidentiality, a symmetric key algorithm would also be required.

3.4.1 Vendors **shall** provide a description of each security service requiring a combination of cryptographic algorithms including the specific security service, cryptographic algorithms, and their associated security strengths in the TDP. (See Requirements 3.1.1.2 and 3.1.1.3)

3.4.1.1 Vendors **shall** provide a detailed technical specification of how the combined cryptographic algorithms support each security service in the TDP.

3.4.2 Voting systems **shall** have the capability to configure the suite of cryptographic algorithms used to provide security services when more than one combination of cryptographic algorithm is supported by the voting system.

3.4.2.1 Vendors **shall** provide a description on how to configure the suite of cryptographic algorithms used to provide security services for the voting system in the user documentation.

3.4.3 Voting systems **shall** use cryptographic algorithms with at least the same level of strength (i.e. bits of security) when multiple cryptographic algorithms are used in combination to provide security services.

Discussion: The following table summarizes the requirements found in Section 3.1 and 3.3:

Security lifetimes (bits of security)	**Symmetric key algorithms**	**DSA Min. Key Size in bits, L & N**	**RSA Min. Key size**	**ECDSA Curves, Prime Field**	**SHA**
Through 2030 (112 bits of strength or higher)	3TDEA AES-128 AES-192 AES-256	$L = 2048$ $N = 224$	2048 bits	len(p)=224 len(p)=256 len(p)=384 len(p)=521	SHA-224 SHA-256 SHA-384 SHA-512
Beyond 2030 (128 bits of strength or higher)	AES-128 AES-192 AES-256	$L = 3072$ $N = 256$	3072 bits	len(p)=256 len(p)=384 len(p)=521	SHA-256 SHA-384 SHA-512

3.5 Cryptographic Key Management Requirements

Voting systems that use cryptographic algorithms that require keys to perform cryptographic operations need to support the capability to properly manage these keys to ensure the effectiveness of the security services provided by the cryptographic algorithm.

3.5.1 General Requirements

3.5.1.1 Vendors **shall** provide a description and detailed technical specifications for the implementation for all the cryptographic key management functionality supported by the voting system in the TDP.

3.5.1.2 Vendors **shall** provide a description of how to use all of the cryptographic key management functionality supported by the voting system in the user documentation.

3.5.1.3 Vendors **shall** provide the model key management policy under which the voting system was designed to operate and a description of the hazards when deviating from the policies in the user documentation.

Discussion: The model key management policy includes the frequency the keys need to be updated and key management procedures for election officials that the voting system was designed to support.

3.5.1.4 All cryptographic keys used by a voting system **shall** be generated within a FIPS 140-2 level 1 or higher validated cryptographic module.

3.5.1.5 If a voting system supports split knowledge cryptographic keys, the keys **shall** exist as multiple key components.

Discussion: Split knowledge keys requires *k* of *n* (where *k* is less than or equal to *n*) components in order for a key to be constructed. Components of split knowledge keys can be generated separately; or generated and then split into components. Note: If the multiple components of a key do not require different authentication (such as passwords or Personal Identification Numbers (PINs)) for the components (e.g. for a two component key, a different authentication is required for each component), then split knowledge has not been provided for the key.

3.5.1.5.1 Each component of a split knowledge key **shall not** provide information of the key to be constructed or of the other components.

Discussion: Simply concatenating multiple keys does not satisfy this requirement.

3.5.1.6 Voting systems **shall** log the following key management events as specified in [Event Logging Section] including who performed the event and its result: key generation (including the key generation technique when multiple key generation techniques are supported by the voting system), key import, key export, key destruction, and key access (including the purpose of the access).

3.5.1.7 Voting systems **shall** uniquely identify keys used by the voting system.

3.5.1.8 Voting systems **shall** associate a purpose (such as key transport, key wrapping, authentication, etc.) to keys used by the voting system.

3.5.1.9 Voting system software **shall not** contain any hard coded cryptographic keys.

3.5.2 Symmetric Key Management Requirements

This section covers the key management requirements for symmetric key cryptographic algorithms used to support basic security services. Symmetric key algorithms require the use of a single key that must only be known to appropriate parties to ensure basic security services, such as confidentiality, can be properly supported.

3.5.2.1 Symmetric Key Generation

3.5.2.1.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of symmetric key generation techniques and functionality supported by the voting system in the TDP.

3.5.2.1.2 Vendors **shall** provide a description of how to use the symmetric key generation techniques and functionality supported by the voting system in the user documentation.

3.5.2.1.3 Voting systems **shall** generate symmetric keys using a FIPS-approved or NIST recommended random number generation method, from a master or currently shared key with a FIPS-approved or NIST recommended key derivation function, or both.

3.5.2.1.4 When using a random number generator to generate symmetric keys, the voting systems **shall** generate keys using a FIPS-approved or NIST recommended random number generation method.

Discussion: [SP 800-90] specifies FIPS-approved or NIST recommended random number generation methods. The list of FIPS-approved or NIST recommended random number generation methods is subject to change over time, due to advancements in random number generation technology. Information on FIPS-approved and NIST recommended random number generation methods can be found at http://csrc.nist.gov/CryptoToolkit/tkrng.html.

3.5.2.1.5 When generating symmetric keys using a currently shared key, the voting system **shall** generate keys by applying a FIPS-approved or NIST recommended non-reversible function to the old uncompromised key and other data.

Discussion: A key update function generates each new key from the preceding uncompromised key (i.e., the currently uncompromised shared key) and other data. FIPS-approved or NIST recommended non-reversible functions currently include the SHA family of hash functions [FIPS 180-2], hash-based message authentication codes (HMACs) [FIPS 198] and symmetric cipher-based message authentication codes (MACs) [SP 800-38B]. Additional key derivation functions explicitly designed for key update may be specified in future FIPS or NIST recommendations.

3.5.2.1.6 When a voting system support a key update function (see 3.5.2.1.4), the voting system **shall** limit the number of times the key update function can be performed.

Discussion: Keys generated using a key update function are dependent on the previous key(s). To reduce the dependency between keys, keys need to be periodically generated from a random source (see 3.5.2.1.3), a master key (see 3.5.2.1.6), or using a key establishment technique (see 3.5.2.2.9).

3.5.2.1.7 When generating symmetric keys from a master key, the voting system **shall** generate keys by applying a FIPS-approved or NIST recommended non-reversible function to the master key and other data.

Discussion: Symmetric keys may be derived from key derivation function where a master key and additional information are provided as inputs. Other inputs typically would include information identifying the communicating components of the voting system, and may also include protocol specific information. The key derivation function must be a non-reversible function so that the master key cannot be determined from the generated key(s). FIPS-approved or NIST recommended non-reversible functions currently include the SHA family of hash functions [FIPS 180-2], hash-based message authentication codes

(HMACs) [FIPS 198] and symmetric cipher-based message authentication codes (MACs) [SP 800-38B]. Additional key derivation functions explicitly designed for key update may be specified in future FIPS or NIST recommendations.

3.5.2.1.8 When voting systems support symmetric keys of different sizes, the voting systems **shall** have the capability to configure the algorithm and size of the symmetric key to be generated.

3.5.2.2 Symmetric Key Establishment

3.5.2.2.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of symmetric key establishment techniques and functionality supported by the voting system in the TDP.

3.5.2.2.2 Vendors **shall** provide a description of how to use the symmetric key establishment techniques and functionality supported by the voting system in the user documentation.

3.5.2.2.3 Voting systems **shall** import and export symmetric keys, establish symmetric keys using key establishment techniques, or both.

Discussion: Key agreement is a method used to establish keys between two or more parties where each party separately derives the same symmetric key using data provided by each party. The key agreement method does not transmit the encrypted or plaintext form of the symmetric key between the parties. Key wrapping is a method used to establish keys between two or more parties where one party generates the symmetric key, encrypts the symmetric key, and transmits the encrypted key to the other parties. The capability of the voting system to import and export symmetric keys provides for manual distribution of the keys.

3.5.2.2.4 When voting systems support the import or export of symmetric keys, the voting systems **shall** import or export the encrypted symmetric keys using a non-proprietary standard data format via a non-proprietary standard interface, except for components of a split knowledge key (see 3.5.1.3).

Discussion: Non-proprietary standard data formats include, but are not limited to, Public Key Cryptography Standard (PKCS) #7 and #12 data formats. [PKCS#7] [PKSC#12]

3.5.2.2.5 When voting systems support the export or import symmetric keys, the voting systems **shall** export or import the symmetric key in an encrypted form using a FIPS-approved or NIST recommended key wrapping or key transport algorithm with FIPS-approved or NIST recommended cryptographic algorithms (See Section 3.1).

Discussion: FIPS approved or NIST recommended wrapping schemes include [RFC 3217] when using TDEA, also called TDES, and [AES KW]. FIPS approved or NIST recommended key transport algorithms include [PKCS#1 v2.1]. The following protocols

have been identified in [FIPS 140-2IG] as appropriate for use when using FIPS approved or NIST recommended cryptographic algorithms: Secure Socket Layer (SSL) v3.1, Transport Layer Security (TLS), Internet Protocol Security (IPSEC), and Secure Shell (SSH). The list of FIPS-approved and NIST recommended key wrapping and transport algorithms is subject to change over time, due to advancements in cryptography. Information on FIPS-approved or NIST recommended key wrapping and transport algorithms can be found at http://csrc.nist.gov/CryptoToolkit/tkkeymgmt.html.

3.5.2.2.6 When voting systems support the import of symmetric keys, the voting systems **shall** authenticate that symmetric keys are supplied from authorized sources.

3.5.2.2.7 When voting systems support the import of symmetric keys, the voting systems **shall** verify the integrity of the symmetric keys.

3.5.2.2.8 When voting systems support the export of symmetric keys, the voting systems **shall** provide information so that the source of the exported symmetric keys can be determined.

3.5.2.2.9 When voting systems support the export of symmetric keys, the voting systems **shall** provide information so that the integrity of the exported symmetric keys can be verified.

3.5.2.2.10 When voting systems support key agreement techniques to establish keys, the voting systems **shall** use FIPS-approved or NIST recommended key agreement schemes.

Discussion: [SP 800-56A] specifies FIPS-approved and NIST recommended key agreement schemes which are restricted versions of the key agreement schemes found in [ANSI X9.42] and [ANSI X9.63]. The list of FIPS-approved and NIST recommended key agreement schemes is subject to change over time, due to advancements in cryptography. Information on the FIPS-approved and NIST recommended key agreement schemes can be found at http://csrc.nist.gov/CryptoToolkit/tkkeymgmt.html.

3.5.2.3 Symmetric Key Usage

3.5.2.3.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of symmetric key usage functionality supported by the voting system in the TDP.

3.5.2.3.2 Vendors **shall** provide a description of how symmetric keys were designed to support security services for the voting system in the user documentation.

3.5.2.3.3 Vendors **shall** provide a description of how to use the symmetric key usage functionality supported by the voting system in the user documentation.

3.5.2.3.4 Voting systems that use symmetric keys for key wrapping **shall** prevent the key from being used for other purposes, such as the general encryption and decryption of information.

3.5.2.3.5 Voting systems **shall** prevent symmetric keys from being used for the both the MAC and encryption operations.

3.5.2.3.6 Voting systems that use master keys to generate other cryptographic keys **shall** prevent the master key from being used for other purposes such as the general encryption and decryption of information.

3.5.2.3.7 Voting systems that use symmetric keys **shall** limit the use of a given symmetric key.

3.5.2.3.7.1 Voting systems **shall not** be able to use symmetric keys for more than 24 months.

3.5.2.3.8 Voting systems that use master keys to generate other cryptographic keys **shall** limit the use of a given master key.

Discussion: A master key is a symmetric key that is used only to generate other symmetric keys.

3.5.2.3.8.1 Voting systems **shall not** be able to use master keys for more than 12 months.

3.5.2.3.9 When symmetric keys are not being used, the voting systems **shall** protect symmetric keys stored outside FIPS 140-2 validated cryptographic modules or inside FIPS 140-2 validated cryptographic modules meeting level 1 physical security and access control requirements in an encrypted form using a FIPS-approved or NIST recommended key wrapping algorithm, using FIPS-approved or NIST recommended cryptographic algorithms (See Section 3.1).

Discussion: FIPS 140-2 validated cryptographic modules are assumed to be able to protect plaintext symmetric keys that are contained with the module, so are not required to store symmetric keys within the cryptographic module in encrypted form when not being used. The physical security requirements of the FIPS 140-2 validated cryptographic module is assumed to provide sufficient protection for symmetric keys stored in plaintext within the cryptographic module. In addition, FIPS 140-2 level 1 validated cryptographic modules are not required to implement access control mechanisms so access to plaintext symmetric keys within the cryptographic module cannot be explicitly limited to authorized parties. This requirement assumes that the physical security (tamper evidence techniques) and access control (role or identity based access) techniques used by FIPS 140-2 level 2 cryptographic modules are sufficient to protect plaintext symmetric keys when they are not being used by the voting system.

3.5.2.4. Symmetric Key Destruction

3.5.2.4.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of symmetric key destruction supported by the voting system in the TDP.

3.5.2.4.2 Vendors **shall** provide a description of how to use the symmetric key destruction functionality supported by the voting system in the user documentation.

3.5.2.4.3 Voting systems **shall** destroy all copies of a given symmetric key held within the voting system in a manner that removes all traces of the symmetric key so that it cannot be recovered by either physical or electronic means.

Discussion: A simple deletion of the keying material might not completely obliterate the information. For example, erasing the information might require overwriting that information multiple times with other non-related information, such as random bits, or all zero or one bits. Removing the electrical power might erase information stored in volatile memory, such as Random Access Memory (RAM). In addition, non-volatile memory, such as Programmable Read Only Memory (PROM), might require ultraviolet light or physical destruction to erase information. Keys stored in memory for a long time can become "burned in". This can be mitigated by splitting the key into components that are frequently updated (see [DiCrescenzo]). Note: symmetric keys that have been exported from the voting system cannot be destroyed by the voting system capabilities. Destruction of symmetric keys outside the voting system needs to be handled procedurally.

3.5.3 Public-Private Key Management Requirements

This section covers the key management requirements for cryptographic algorithms (see section 3.1.3) that require the use of two mathematically related keys, a private and public key pair, to support basic security services. In order to ensure that basic security services can be supported such as authentication, public keys can be known by all parties. However private keys must only be known by a single party or limited set of authorized entities (people, machines, applications, etc.).

3.5.3.1 Public-Private Key Pair Generation

3.5.3.1.1 Voting systems **shall** generate public-private key pairs in accordance with the specifications of FIPS-approved or NIST recommended asymmetric cryptographic algorithms.

Discussion: Section 3.1.3 specifies the FIPS-approved and NIST recommended cryptographic algorithms that use public-private key pairs and 3.5.1.2 requires the use of a FIPS 140-2 validated cryptographic module to generate keys, making this requirement redundant since FIPS 140-2 validated cryptographic modules must meet this requirement to be validated.

3.5.3.1.2 Vendors **shall** provide a description and detailed technical specifications for the implementation of public-private key pair generation techniques and functionality supported by the voting system in the TDP.

3.5.3.1.3 Vendors **shall** provide a description of how to use the public-private key pair generation techniques and functionality supported by the voting system in the user documentation.

3.5.3.2 Private Key Establishment

3.5.3.2.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of private key establishment techniques and functionality supported by the voting system in the TDP.

3.5.3.2.2 Vendors **shall** provide a description of how to use the private key establishment techniques and functionality supported by the voting system in the user documentation.

3.5.3.2.3 If voting systems import or export private keys, the voting systems **shall** import or export the encrypted private keys using a non-proprietary standard data format via a non-proprietary standard interface.

Discussion: Non-proprietary standard data formats include, but are not limited to, Public Key Cryptography Standard (PKCS) #7 and #12 data formats. [PKCS#7] [PKSC#12]

3.5.3.2.4 When a voting system export or import private keys, the voting systems **shall** export or import the private key in an encrypted form using a FIPS-approved or NIST recommended key wrapping or key transport algorithm using FIPS-approved or NIST recommended cryptographic algorithms (See Section 3.1).

Discussion: FIPS approved or NIST recommended wrapping algorithms include [RFC 3217] when using TDEA, also called TDES, and [AES KW]. FIPS approved or NIST recommended key transport algorithms include [PKCS#1 v2.1]. The following protocols have been identified in [FIPS 140-2IG] as appropriate for use when using FIPS approved or NIST recommended cryptographic algorithms: Secure Socket Layer (SSL) v3.1, Transport Layer Security (TLS), Internet Protocol Security (IPSEC), and Secure Shell (SSH). The list of FIPS-approved and NIST recommended key wrapping and transport algorithms is subject to change over time, due to advancements in cryptography. Information on FIPS-approved or NIST recommended key wrapping and transport algorithms can be found at http://csrc.nist.gov/CryptoToolkit/tkkeymgmt.html.

3.5.3.2.5 When voting systems import private keys, the voting system **shall** authenticate that private keys are supplied from authorized sources before being imported and used by the voting system.

3.5.3.2.6 When voting systems import private keys, the voting system **shall** determine that the private key is associated with the appropriate public key.

Discussion: For example, a public-private key pair association for a key transport key pair can be checked by encrypting data using the public key and verifying that the private key can decrypt the data.

3.5.3.3 Public Key Establishment

3.5.3.3.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of public key establishment techniques and functionality supported by the voting system in the TDP.

3.5.3.3.2 Vendors **shall** provide a description of how to use the public key establishment techniques and functionality supported by the voting system in the user documentation.

3.5.3.3.3 If voting systems import or export public keys, the voting systems **shall** import or export the public keys using an authenticated non-proprietary standard data format via a non-proprietary standard interface.

Discussion: Non-proprietary standard data formats include, but are not limited to, Public Key Cryptography Standard (PKCS) #7 and X.509 certificate data formats. [PKCS#7] [ISO X.509]

3.5.3.3.4 When voting systems import public keys, the voting systems **shall** authenticate that public keys are supplied from authorized sources before being imported and used by the voting system.

3.5.3.3.5 When voting systems import public keys, the voting systems **shall** determine that the public key is associated with the appropriate private key.

Discussion: For example, a public-private key pair association can be checked by generating a digital signature using the private key and establishing that the public key can verify the digital signature.

3.5.3.3.6 When voting systems use ephemeral public keys, the voting systems **shall** determine the validity of the ephemeral public key.

Discussion: [SP 800-56A] specifies techniques that can be used to determine the validity of ephemeral public keys used for key establishment.

3.5.3.4 Private Key Usage

3.5.3.4.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of private key usage functionality supported by the voting system in the TDP.

3.5.3.4.2 Vendors **shall** provide a description of how public-private key pairs were designed to support security services for the voting system in the user documentation.

3.5.3.4.3 Vendors **shall** provide a description of how to use the private key usage functionality supported by the voting system in the user documentation.

3.5.3.4.4 Voting systems that use private keys **shall** limit the use of a given private key.

3.5.3.4.4.1 Voting systems **shall not** be able to use private keys for more than 24 months.

3.5.3.4.5 Voting systems that use private keys for key transport **shall** prevent the private keys from being used for other purposes such as authentication.

3.5.3.4.6 When private keys are not being used, the voting systems **shall** protect private keys stored outside FIPS 140-2 validated cryptographic modules or inside FIPS 140-2 validate cryptographic modules meeting level 1 physical security and access control requirements in an encrypted form using a non-proprietary standard key wrapping algorithm using FIPS-approved or NIST recommended cryptographic algorithms (See Section 3.1).

Discussion: FIPS 140-2 validated cryptographic modules are assumed to be able to protect plaintext private keys that are contained within the module, so are not required to store private keys within the cryptographic module in encrypted form when not being used. The physical security requirements of the FIPS 140-2 validated cryptographic module is assumed to provide sufficient protection for private keys stored in plaintext within the cryptographic module. In addition, FIPS 140-2 level 1 validated cryptographic modules are not required to implement access control mechanisms so access to plaintext private keys within the cryptographic module cannot be explicitly limited to authorized parties. This requirement assumes that the physical security (tamper evidence techniques) and access control (role or identity based access) techniques used by FIPS 140-2 level 2 cryptographic modules are sufficient to protect plaintext private keys when they are not being used by the voting system.

3.5.3.5 Private Key Destruction

3.5.3.5.1 Vendors **shall** provide a description and detailed technical specifications for the implementation of private key destruction supported by the voting system in the TDP.

3.5.3.5.2 Vendors **shall** provide a description of how to use the private key destruction functionality supported by the voting system in the user documentation.

3.5.3.5.3 Voting systems **shall** destroy all copies of a given private key held within the voting system in a manner that removes all traces of the private key so that it cannot be recovered by either physical or electronic means.

Discussion: A simple deletion of the keying material might not completely obliterate the information. For example, erasing the information might require overwriting that information multiple times with other non-related information, such as random bits, or all zero or one bits. Removing the electrical power might erase information stored in volatile memory, such as Random Access Memory (RAM). In addition, non-volatile memory, such as Programmable Read Only Memory (PROM), might require ultraviolet light or physical destruction to erase information. Keys stored in memory for a long time can become "burned in". This can be mitigated by splitting the key into components that are frequently updated (see [DiCrescenzo]). Note: private keys that have been exported from the voting system cannot be destroyed by the voting system capabilities. Destruction of private keys outside the voting system needs to be handled procedurally.

3.6 Cryptography Application Requirements

3.6.1 General Electronic Communications Requirements

3.6.1.1 If voting systems use electronic communications between two of its components, then the integrity of the communications **shall** be protected by cryptographic means (e.g. digital signatures or message authentication codes) unless either: (a) the communications channel between the components is entirely within a protected physical enclosure of the voting system, or (b) the integrity of the communications is documented not to be necessary for the reliability and security of the voting system.

Discussion: Electronic communications includes the physical transport of storage media (e.g. a CD or memory card). T-coil transmissions are excluded from the requirement since one component, the recipient's hearing aid, is not part of the voting system. Physical connection cables such as video and printer cables may be excluded from the requirement only if both cable endpoints are physically protected. Wireless communications may be excluded from the requirement only if wireless signals from outside the voting system are not capable of interfering with the communications between the components (because of the physical enclosure of the components and of the space used by the wireless signals). Cryptographic key management requirements and guidelines should at minimum require cryptographic keys be changeable according to a policy and procedures specified by election officials.

3.6.1.2 If voting systems use electronic communications between two of its components, then confidentiality (i.e. secrecy) of the communications **shall** be protected by cryptographic means (i.e. encryption) unless either: (a) the communications channel between the components is entirely within a protected physical enclosure of the voting system, or (b) the integrity of the communications is documented not to be necessary for the reliability and security of the voting system.

Discussion: Electronic communications includes the physical transport of storage media (e.g. a CD or memory card). T-coil transmissions are excluded from the requirement since one component, the recipient's hearing aid, is not part of the voting system. Physical connection cables such as video and printer cables may be excluded from the requirement

only if both cable endpoints are physically protected. Wireless communications may be excluded from the requirement only if wireless signals from outside the voting system are not capable of interfering with the communications between the components (because of the physical enclosure of the components and of the space used by the wireless signals). Cryptographic key management requirements and guidelines should at minimum require cryptographic keys be changeable according to a policy and procedures specified by election officials.

4. References

[AES KW] National Institute of Standards and Technology, DRAFT AES Key Wrap Specification, November 2001. Available at http://csrc.nist.gov/CryptoToolkit/kms/AES_key_wrap.pdf.

[ANSI X9.31] American National Standards Institute, X9.31-1998, Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA), September 1998.

[ANSI X9.42] American National Standards Institute, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using discrete Logarithm Cryptography, 2001.

[ANSI X9.62] American National Standards Institute, Public Key Cryptography: Public Key Cryptography: The Elliptic Curve Digital Signature Algorithm (ECDSA), November 2005

[ANSI X9.63] American National Standards Institute, Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography, November 2001

[DiCrescenzo] How to forget a secret, G. Di Crescenzo, N. Ferguson, R. Impagliazzo, and M Jakobsson, STACS '99. Available via www.macfergus.com/pub/forget.pdf.

[FIPS 140-2] Federal Information Processing Standard (FIPS) 140-2: Security Requirements for Cryptographic Modules, May 2001.

[FIPS 140-2C] DRAFT Annex C: Approved Random Number Generators for FIPS 140-2, Security Requirements for Cryptographic Modules, January 31, 2005.

[FIPS 140-2IG] National Institute of Standard and Technology and Communications Security Establishment, Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program, December 2001.

[FIPS 180-2] Federal Information Processing Standard (FIPS) 180-2: Secure Hash Standard (SHS), August 2002.

[FIPS 186-2] Federal Information Processing Standard (FIPS) 186-2: Digital Signature Standard (DSS), January 2000.

[FIPS 197] Federal Information Processing Standard (FIPS) 197: Advanced Encryption Standard, November 2001.

[FIPS 198] Federal Information Processing Standard (FIPS) 198: The Keyed-Hash Message Authentication Code (HMAC), March 2002.

[PKCS#1 v2.1] RSA Laboratories, PKCS #1 v 2.1: RSA Cryptography Standard, June 2002.

[PKCS#7] RSA Laboratories, PKCS #7: Cryptographic Message Syntax Standard, November 1993.

[PKCS#12] RSA Laboratories, PKCS 12 v1.0: Personal Information Exchange Syntax, June 1999.

[RFC 3217] Housely, R., Triple-DES and RC2 Key Wrapping, Internet Engineering Task Force (IETF) Request For Comment (RFC) 3217, December 2001.

[SP 800-38] National Institute of Standards and Technology, Special Publication 800-38: Recommendation for Block Cipher Modes of Operations – Methods and Techniques, December 2001.

[SP 800-38B] National Institute of Standards and Technology, Special Publication 800-38B: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, May 2005

[SP 800-56A] National Institute of Standards and Technology, Draft Special Publication 800-56: Recommendation on Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, July 2005.

[SP 800-57] National Institute of Standards and Technology, Special Publication 800-57: Recommendation for Key Management, August 2005.

[ISO X.509] ISO/IEC 9594-8/ITU-T Recommendation X.509, "Information Technology - Open Systems Interconnection: The Directory: Authentication Framework," 2001 edition.