To: EncryptionModes@nist.gov
From: Matthew V. Ball, Chair of IEEE Security in Storage Working Group (P1619)
Subject: NIST's Consideration of XTS-AES as standardized by IEEE Std 1619-2007
Date: Sept 1, 2008

It is with pleasure that I have had the opportunity to facilitate this process to present the XTS-AES mode, as standardized by IEEE Std 1619-2007, to NIST for consideration as an Approved Mode of Operation under FIPS 140. This new mode of operation provides a very useful tool in the algorithm designer's toolbox.

Here are several topics relating to XTS, and why this is a good choice as an Approved Mode of Operation.

Parallel design:
In today's world of multiple core processors and clock-rate limitations, it is increasingly important that a designer be able to increase performance by instantiating multiple instances of an encryption primitive instead of increasing the clock rate of an existing encryption primitive. XTS can operate in parallel, allowing scalability in today's environment. This is not true of some other Approved modes of operation, like cipher-block chaining (CBC), cipher feedback (CFB), and output feedback (OFB) modes (see [SP 800-38A]), or even the authenticated encryption mode CCM (see [SP 800-38C]). Concerns like this is what lead to the development of the authenticated encryption mode GCM (see SP 800-38D and IEEE std 1619.1-2007).

Tweakable:
All secure encryption modes require some way to create variability so that a copy of the same plaintext is not encrypted into the same ciphertext. The ciphertext stream needs to appear as a random stream of bits, and creating duplicate ciphertext blocks (as happens under ECB mode) leaks information. This is why common modes, like CBC, require a 'nonce' or initialization vector (IV) as part of the input into the encryption mode. The 'tweak' as used by XTS serves this same purpose -- to create different ciphertext when presented with the same plaintext. If the nonce, IV, or tweak is different (or has a high probability of being different) for each invocation of the cryptographic primitive, then duplicate plaintext input will not leak information to an adversary.

Industry adoption of XTS-AES:
Ever since IEEE approved 1619 last December, industry adoption of XTS has been brisk. Maybe hardware core vendors now have XTS implementations available for purchase. Several open source disk encryption utilities also support XTS: TrueCrypt, FreeOTFE, and dmcrypt.

Several major hard drive vendors also have XTS implementations that will be available in future products.

For more information on current industry adoption of IEEE 1619 and 1619.1, see this article.

Comparing XTS to CBC for hard disk encryption
If a storage device vendor is seeking FIPS 140-2 certification today, they will typically use CBC encryption, or even ECB. CBC is a good mode, but as compared to XTS, it suffers deficiencies in that it is not parallel and has bit-wide malleability potential (if randomizing the remainder of the sector is acceptable).

When comparing XTS to CBC, the hardware complexity is only slightly more for the XTS mode. XTS essentially requires another 128 D-type flip-flops to store the masking value, and needs another 128 XOR gates for the two masking operations vs. CBC's one masking operation. However, compared to the size of the AES block cipher, these extra gates are typically a minor increment.

For software-based solutions, Intel has recently announced a new instruction that can perform a round of AES encryption. In practice, this instruction is not very efficient when used to perform a single AES encrypt or decrypt because of pipeline stalls. However, if this instruction were used to perform several parallel AES encrypt or decrypt operations, the performance substantially increases, perhaps even 6-fold (see discussion on metzdowd e-mail list). A speedup could also be possible on the VIA C7 processor, if performing 2 AES operations in parallel. For these reasons, modes that all parallel operation (like XTS) will have a substantial performance boost over serial modes (like CBC).

Security of XTS

XTS is based on the XEX mode proposed by Phillip Rogaway (see Rogaway [R04]). In his original paper, Rogaway provides a security proof for the security bounds of XTS when compared to an ideal tweakable block cipher.

The group also spent some time analyzing XTS under key-dependent plaintext inputs, such as those that broke the LRW-AES mode under practical applications (such as encrypting operating system swap files that contain the key in working RAM). For some such analysis, see Halevi [HK07]. Because of the nature of the multiply by alpha (i.e., a linear feedback shift register operation) on the tweak value, there is no simple algebraic way to cause a collision on the cipher block input when a key is present in the plaintext.

There is a weak key issue if the tweak value ever starts at zero. In this case, there is no masking and the security reduces to that of ECB mode. However, the probability of this occurrence is 1 in 2^128, which is reasonable for practical encryption applications. NIST requires at least 80 bits of security until 2010, and 112 bits until 2030, so this weak tweak is within acceptable security bounds for the foreseeable future.

Optional Changes to XTS for NIST Adoption:

There are a couple changes that NIST could make to XTS without at most minor effects on the security:

- *Allow using one AES key for both AES block ciphers*: XTS as defined in IEEE 1619 requires two different keys for the AES cipher used to create the tweak, and for the AES cipher used for ECB encryption. In Rogaway's original description of XEX, he allowed using the same AES key for both AES ciphers, and showed that the security is reasonable in this configuration. NIST may consider allowing the same key to be used in both ciphers for XTS.
- *Allow switching the order of the blocks created through Ciphertext Stealing*: There is no security-related reason for the ciphertext stealing order as specified in IEEE 1619. If a particular implementation of XTS does not lend itself to interchange (e.g., internal encryption in a hard disk), then it should be acceptable to change the ordering of the ciphertext stealing blocks when writing the ciphertext to disk.
- *Allow 192-bit AES keys*. IEEE 1619 only allows 128-bit and 256-bit keys for the AES block cipher, to make interchange and implementation easier. However, there is no security-related reason to restrict the use of 192-bit keys, if there was some application for this.

Note that it is not a good idea to allow substituting 3DES for AES. Using anything other than a 128-bit wide cipher forces a change of the finite field and puts the birthday bounds at about 64 GB, which is unacceptably low for today's common storage capacities. As with any mode that follows the construction outlined in the original LRW paper [LRW02], XTS leaks information when encrypting more information than allowed by the birthday bound (which is 2^32 cipherblocks for 3DES).

Intellectual Property (IP) Concerns of XTS:
Phillip Rogaway (the inventor of XEX, the basis for XTS), has no IP claims on XEX, nor knows of anyone else who does (see this e-mail). NeoScale (now nCipher -- soon to be acquired by Thales) submitted a blanket statement of owning undisclosed IP for all of IEEE P1619, but there is no reason to believe that this IP covers XTS specifically. I am currently seeking clarification from nCipher, and should have this within the next couple months.

Replacing Electronic Code Book (ECB) Mode:
By accepting XTS, NIST has an opportunity to start the process to remove ECB as an Approved Mode of Operation. ECB is currently the only insecure mode of operation allowed under FIPS 140-2, when used as described. Many products have passed FIPS 140-2 certification by changing an otherwise secure (but not Approved) mode to ECB mode. However, removing ECB in the past has been difficult because no other mode of operation can operate both in parallel, but not allow bit-level malleability. By using XTS, both of these requirements can be met.

I strongly recommend that if NIST accepts XTS, then NIST should start the sunset process on ECB and have it completely removed in time for FIPS 140-3, or shortly thereafter.

References:

- [LRW02] M. Liskov, R. Rivest, and D. Wagner. "Tweakable Block Ciphers", 2002.
- [HK07] S. Halevi, and H. Krawczyk. "Security under Key Dependent Inputs", August 2007.
- [IEEE 1619] IEEE Std 1619-2007, "Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices", December 2007.
- [IEEE 1619.1] IEEE Std 1619.1-2007, "Standard for Authenticated Encryption with Length Expansion for Storage Devices", December 2007.
- [IEEE 1619.2] IEEE P1619.2 Draft 7, "Draft Standard for Wide-Block Encryption for Shared Storage Media", August 2008.
- [R04] P. Rogaway. "Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC", September 2004.
- [SP 800-38A] M. Dworkin. "Recommendation for Block Cipher Modes of Operation: Methods and Techniques", December 2001.
- [SP 800-38B] M. Dworkin. "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication", May 2005.
- [SP 800-38C] M. Dworkin. "Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality", May 2004.
- [SP 800-38D] M. Dworkin. "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", November 2007.