Public Comments on the Draft "Recommendation for Block Cipher Modes of Operation"

NIST received the following sets of public comments on the draft "Recommendation for Block Cipher Modes of Operation."

Commenter	Date Received	Page
Don Johnson	August 3, 2001	2
Miles Smid	August 7, 2001	4
Phillip Rogaway	August 8, 2001	5
François Rousseau	August 10, 2001	9
Cathy Robinson	August 21, 2001	10
Carol Widmayer	August 22, 2001	11
Don McGinnis	August 22, 2001	11
Sharon Shank	August 29, 2001	11
Mark Winstead	August 29, 2001	12
Patricia Black	August 30, 2001	12
Nancy Cam-Winget	August 30, 2001	12
John Smith	August 31, 2001	14
Tom Brand	August 31, 2001	14
John Savard	August 31, 2001	14
Social Security Administration	September 5, 2001	15

Don Johnson

Here are my comments on the NIST draft recommendations for block cipher modes of operation, dated July 2001. I am glad to see the CTR mode added to the four basic data confidentiality modes and an authentication mode specified.

Please get back to me with any questions, comments, etc.

Thanks, Don Johnson, Certicom

****

1) Suggest that only 1-bit, 8-bit and 128-bit (full blocksize) CFB be specified (analogous to ANSI X9.52 1-, 8- and 64-bit (full blocksize) modes) as this helps limit option proliferation.

2) I believe NIST should endorse a high security data confidentiality mode that makes every bit of plaintext affect every bit of ciphertext and vice versa. This mode (in effect) makes a big block of ciphertext out the entire plaintext, and seems to imply at least 2 passes to accomplish; therefore this mode would likely not be used except in cases of high security and/or short message length (e.g., symmetric key distribution via a symmetric key), but there will be situations where the additional processing cost could be worth the increased assurance. I propose double CBC (two CBC passes, the first as normal using a user-supplied IV, the second over the *intermediate* ciphertext using the last ciphertext block as an IV) as an initial idea towards this. The decryption is a little tricky but is straightforward to unravel. Other ideas are also possible.

3) The CBC-MAC method specified REQUIRES that all the messages to be MAC'ed be the same length n. Furthermore, this length n is a security parameter, both parties must agree to it, it is an attack concern if it is changed. Therefore it is not secure to use this MAC method for even ONE MAC if the message length is not known. It does seem curious that this very high security MAC method was specified but a very high security encryption method was not (see comment 2 above). Note that this comment should not necessarily be seen as a request to remove this high security MAC method.

These severe restrictions may be the appropriate in SOME scenarios with high security, but this same message length requirement could lead to a tremendous proliferation of keys, one for each potential message length, established and agreed to before use. This proliferation will be costly and while it may be appropriate for very high security applications, but it may cost too much for common scenarios. For example, the alternative HMAC has no such message length restriction. To handle the general case, the CBC-MAC can be truncated (as was done in ANSI X9.8), to at least half the blocksize (for AES, this is 64 bits). For example, letting A and B be different messages of one block each, then if an adversary has MAC(A) and MAC(B), then he knows he can use MAC(B) as the MAC for the previously unMAC'ed string "A || MAC(A) XOR B". Halving the output blocksize equally balances the chances of a MAC exhaustion attack with the potential message concatenation attack mentioned above. Of course, there are other ways to foil this attack concern (for example, knowing the length of the message) but at least one of the MAC'ing methods specified should work for arbitrary length messages. Another alternative might be to use a method analogous to the optional MAC method in ANSI X9.19, where 2 keys are used, the one as normal, and the second to decrypt the last CBC block, and the first again

to encrypt this result to produce the final MAC.

4) I suggest Appendix A be titled "Short Block Handling" as this is a more general title and there is a reasonably good method of handling short blocks that does not result in message expansion (that is, message padding), namely, ciphertext stealing. I believe the ciphertext stealing method should be defined/allowed, at least for CBC mode. This fills the last short block by "stealing" any needed text from the previous ciphertext block. As normally defined, the short plaintext is concatenated to the next-to-last ciphertext block and the rightmost 64 bits are used as the last plaintext block. Again, the decryption is a little tricky but is straightforward to unravel. It does not really work if the plaintext is less than one block in size (cannot really block encrypt less than one block), but otherwise this way of handling a short block results in no message expansion, which is a desirable attribute in some scenarios, such as replacing arbitrary length plaintext in EXACTLY the same location with ciphertext. It is even possible to define a way to handle less than one block, by encrypting the IV and XOR'ing the needed bits.

5) It is not clear why the CBC/CFB/OFB interleaved/pipelines modes are not endorsed by this document. The reason they were defined in ANSI X9.52 is that it was believed there would be scenarios where using additional symmetric cipher engines (that is, more hardware) would be a cost-effective choice, for example, for those cases where performance is critical and the additional cost is warranted. It is true that the new CTR could be used, but it does not have exactly the same properties of the others; in particular, CTR mode requires assurance that the counter not repeat, but there is no counter in CBC mode, for example. That is, it could easily be the case where interleaved CBC mode is the preferred solution and meets all customer requirements, but is not endorsed by this document. At the least, if CTR mode is thought to be the general answer to performance questions, this should be explained more.

6) There should be a short discussion on how to pick a method, giving the attributes and assumptions of each. For example, ECB mode should not be used unless the input is highly variable, etc. As it stands now, ECB is the simplest and is totally parallelizable, so why should it not be used all or most of the time, assumptions and cautions should be stated more explicitly. That is, a user could conform to this document yet still be insecure and sufficient warnings should be given. This discussion could also give performance information that allows easy comparisons between the methods, perhaps summarized in a table.

7) Constrained environment solutions should be given a priority in discussion. If doing crypto on a PC or above, any solution will likely work; in fact multiple algorithms can be implemented to ensure interoperability as needed. It is where there are limited resources that one dimension of the problem gets interesting. For example, what algorithms might be able to be omitted from a solution, yet still provide all needed functionality to address the application's needs. For example, it is a "good thing" to provide both data confidentiality and message authentication using the same building block.

Miles Smid

Comments of the NIST Recommendation for Block Cipher Modes of Operation

Miles E. Smid
Entrust-CygnaCom
July 31, 2001

1. (General) This is a very well written paper by Morris Dworkin dated July 2001, which specifies six modes of operation that may be performed with any FIPS approved block cipher algorithm. The recommendation generalizes the four standard confidentiality modes Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), and Output Feedback (OFB) that have been used for over twenty years with the Data Encryption Standard (DES) algorithm. A new confidentially mode called the Counter (CTR) mode is also provided. This mode has the advantage that plaintext blocks may be processed in parallel thus increasing throughput. The recommendation also specifies a CBC Message Authentication Code (CBC-MAC) that generalizes the former DES based MAC. This recommendation would accommodate the NIST Advanced Encryption Standard (AES) algorithm, which is currently based on 128-bit block rather than the 64-bit DES block size. Finally, many examples using AES as the block cipher are provided. Specific comments are provided below.
2. (Authority) Although this is a recommendation, it does contain requirements for correctly implementing the various modes. It should be stated that in order to claim conformance with the recommendation, one must conform to the applicable requirements.
3. (5.3 Cipher Feedback Mode) It should be made clear that if the IV is fixed, then knowledge of the first plaintext segment for one message will compromise the first segment of every other message encrypted with the same key.
4. (5.6 CBC-MAC) It should be noted that b-bit CFB could also be used to compute this MAC.
5. The requirement that every MAC computed with a given key have the same number of blocks is too restrictive. Alternatives that allow differing message lengths should be provided if possible.

Phillip Rogaway

To: Morris Dworkin
National Institute of Standards and Technology
From: Phillip Rogaway
University of California at Davis & Chiang Mai University
http://www.cs.ucdavis.edu/~rogaway
Re: Comments on: "Recommendation for Block Cipher Modes of Operation"
http://csrc.nist.gov/publications/drafts/Modes01.pdf
Today: Aug 4, 2001

First, I'd like to applaud NIST's work in this first round of updating block-cipher modes of operation. I am particularly pleased that CTR mode has made it into the draft. In general, the writeup is very clear and well written. I have just three main technical suggestions, plus a few technical-editorial comments.

Technical Comments
---

1. Don't assume messages are a blocksize multiple

The reviewed document defines all five modes modes as operating on messages that are a sequence of BLOCKS, saying that padding is out of scope (although the draft does suggest three padding methods in Appendix A.) Though I appreciate that this approach in line with what has been done in the past, it still seems wrong, from both conceptual and practical considerations:

(i) The conceptual complaint is:
(a) The messages which one is encrypting/MACing are, in the user's mind, arbitrary bit strings (or arbitrary octet strings). Insisting that they have a length which is a positive multiple of some magic number (the blocksize b) amounts to surfacing in a higher-level protocol an unnecessary-to-surface implementation aspect of the mechanisms.
(b) Modern definitions of encryption often assume a message space of {0,1}^*. Indeed this is the only natural choice when you need to pin down what is the message space for an arbitrary encryption scheme.
(c) Cryptographic mechanisms of other NIST FIPS, eg., DSS, SHA-1, and HMAC, work on arbitrary bit strings. I see no conceptual or technical reason to make a standard for symmetric-encryption do otherwise.

(ii) The practical complaint is:
(a) The messages one has to encrypt are often (maybe usually) NOT a multiple of 16 bytes, so having to deal with a larger message space is more the rule than the exception.
(b) Using padding to deal with messages which are not a multiple of the blocksize is less efficient, in terms of bit complexity, than alternatives. Thus one doesn't want to construct a standard

which might seem to force one to get to a broader domain
(esp. for CBC) by way of padding.

(c) People routinely mess up on padding --particularly for the CBC-MAC-- resulting in incorrect implementations. This seems to be a consequence of earlier standards giving the issue second-class status.

I would be particularly nice if CBC mode and CTR mode were defined as operating not on BLOCK^+ but on {0,1}^* (where BLOCK={0,1}^b). This can be done in a way that is a proper extension of the functions you defined with message space BLOCK^+:

* CBC could become CBC with "ciphertext stealing", namely,
  - Partition message P into P = P_1 ... P_n, for n>=1, where |P_1|=...=|P_{n-1}|=b and |P_n| <= b and P_n is the emptystring iff P is the emptystring.
  - If |P_n|=b then apply CBC as usual: C_i = CIPH_K(C_{i-1} xor P_i) for all i in[1..n], C_0 = IV.
  - If |P_n| < n then: C_i = CIPH_K(C_{i-1} xor P_i) for i in [1..n-1], C_0=IV, and C_n = CIPH_K(C_{n-1}) #xor# P_n. Notation: A #xor# B is (the first min{|A|,|B|} bits of A) xor (the first min{|A|,|B|} bits of B)

  One doesn't really lose anything with such an extension: if one wanted to handle arbitrary-length messages by a padding scheme instead of by ciphertext stealing, then one could still apply the padding scheme and then invoke CBC as defined above; the special treatment of short final-blocks would be irrelevant.

* CTR mode (and OFB mode) are naturally defined as operating on arbitrary bit strings anyway. So fixing up the domain to {0,1}^* amounts to an editorial comment to just incorporate paragraph 2 of Appendix A into the mode's definition.

2. Deprecate ECB

ECB mode should be deprecated as a confidentiality mechanism. The mode does not achieve any reasonable definition of security. Not only is it deterministic (so it necessarily leaks message equality), it further leaks which message blocks of every message are identical to which other message blocks of every message. I think this is unacceptable in _most_ circumstances. It seems undesirable to perpetuate a fundamentally "wrong" mode like ECB.

3. Handle IVs better for CBC

The last paragraph on page 16 implies that unique IVs are enough for security. They are not; the CBC IV needs, in general, to be adversarially unpredictable. To see this, observe
that if the adversary has observed (IV1, P_1, CBC_K(IV1, P_1 ...))
and the adversary knows what will be IV2 for the next message,
then the adversary can distinguish the encryption of, for example,
messages: P_1 xor IV1 xor IV2 and any-other-block.
While this may sound inconsequential, it is enough to show that the mode does not achieve any modern definition of encryption-scheme privacy

(as defined, for example, in [Bellare, Desai, Pointcheval, Rogaway]). Whatever is achieved is unknown and, I would claim, uninteresting.

Books, standard, and implementations routinely get wrong what the CBC IV should be, suggesting, for example, that it can be a fixed value, a counter, a timestamp, or the last block of ciphertext just encrypted. While one might not get in trouble in a given application, none of these suggestions result in a correct encryption scheme (in the sense that our community now understand correctness).

I suggest that the FIPS set C_0 = CIPH_K(IV) in the definition of CBC mode (as opposed to the current C_0 = IV). While this adds in one block-cipher call, it doesn't _really_ mean extra overhead, in most cases: CBC mode, when used correctly, would, in most cases, already have involved the user selecting IV = CIPH_K(Nonce).

I suggest to add in a discussion about the choice of IV (and also the fact that it need not be secret). One should explain that if one does choose C_0 = IV, then each IV must be adversarially unpredictable (even given all of the IVs used to date).

Editorial Comments
---

1. The notation [x]_2 emphasizes the base (2) when it would be more useful to emphasize the number of bits, s, as the subscript. Undefined if x >= 2^s.

2. Do we really need CFB and OFB?

   The error-propagation properties that motivated CFB seems mostly outmoded, while CTR is just as good as OFB for making a pseudorandom generator or stream cipher.

   CFB is the worse. It is complicated and, to my knowledge, there is no paper that proves the mode's security (under the assumption that CIPH is secure in the sense of a pseudorandom permutation).

   The IV should be unpredictable for CFB, just like for CBC. For OFB, it is not enough for the IV to be a nonce (yet unpredictability seems unimportant). I'm not sure what is the technical condition one needs.

3. It may look odd, but it makes sense to "spell solid" CBCMAC when regarding this as a function. Doing this, one can write, for example, CBCMAC-AES128, which makes better sense than CBC-MAC-AES128. (I do see you already spelled solid AES128, AES192, AES256; thanks!)

4. One absolutely needs a MAC that works across messages of arbitrary lengths. But given the clarity of the first paragraph of 5.6, I can't complain _too_ much -- it would seem you made an editorial decision to put off doing this until "round 2".

5. Given (4), one might clearly reflect in the title of this document that

"more is coming" -- eg., by appending "- Part 1" to the title.
That more would have to include a MAC with a more useful domain.

6. Appendix A.
   - Paragraph 1: The empty sequence of data blocks (0 data blocks) is not permitted by your algorithms; say explicitly you need a sequence of one or more data blocks.
   - Paragraph 3: "Permits the number of bits in the message to expand" is unclear.
   - Paragraph 4: maybe "last bit" or "least significant bit" instead of "trailing bit".
   - I can't say that I really see the point of having any padding method described but the second one, and I don't know the motivation for this last strange padding method. But I guess you have your reasons; you might wish to document them.

7. Appendix B. This leaves things rather open-ended, and I fear the generality that it suggests may encourage people to do complicated things which really aren't necessary or useful. One might add in an Appendix B.3 that gives a simple and fully specified example: drop a message counter into the top half of each T_i and put a block counter in the bottom half of each T_i.

8 Appendix C.
   - line 4. "an arbitrarily purported" -> "a random purported" and line 5, "may" -> "will". (Else the statement isn't true.)
   - I disagree with the recommendation that t should be as large as possible. Values above t=96, say, are certainly adding no practical benefit. In fact, it seems likely that using the full final block may do more harm than good, because it gives the adversary information she would not otherwise get. That is, I suspect that truncation of some bits actually improves the security, but there has been no analysis tight enough to demonstrate if this is true or false.
   - I don't think you need to _insist_ that t >= 32. While most applications would be wrong to select a smaller value of t than 32, a very small value (eg., t=8) seems appropriate for certain contexts. Imagine an application (eg., authenticating video frames) where you'd have to have to forge multiple packets in order to have a detrimental effect, and where the connection will be dropped on the first incorrectly MACed packet. A one-byte MAC may be just the thing.

9. In your appendix F, you might abandon the second hyphen in contexts like CBC-AES128-Encrypt" in favor of, say, "CBC-AES128 Encrypt" or "CBC-AES128.Encrypt".

10. Somewhere, maybe in another appendix, you should emphasize that NONE of modes CBC, CTR, CFB, OFB do anything to protect message integrity / message authenticity. One should use a separate MAC, or use a an authenticated-encryption mode instead.

11. Guidance on how to combine an encryption scheme and a MAC might be worthwhile, as this is so often done wrong. Following [Bellare, Namprempre] and [Krawczyk], you should specify MACing the ciphertext (including the IV) (using separate keys for the encryption and the MAC, of course).

François Rousseau

Although I no longer work at Chrysalis-ITS, but now work at the Communications Security Establishment (CSE) in Ottawa, here are few personal comments that I found during a quick reading of this very comprehensive document, which may however not necessarily reflect the view of my new employer.

First, here are some technical comments:

a. The final version of this recommendation should probably include a new Section (e.g. under Section 5) or a more extensive Appendix (e.g. expending Appendix D) that more explicitly explains the advantages and disadvantages of each mode (e.g. parallel, errors, synchronization, impact of using the same key and/or IV, etc.) similar to Annex A to ISO/IEC 10116 on the properties of the modes of operation.

b. Section 5, fifth paragraph and Appendix A, why is the issue of formatting of the plaintext and padding bits outside the scope of this recommendation? Without an agreement on this, there can be no interoperability. Note that PKCS #11 v2.11 has a distinct mechanism for the AES CBC mode with PKCS padding, denoted CKM_AES_CBC_PAD, mandating the block cipher padding method detailed in PKCS #7, which is not even explicitly mentioned under Appendix A although you refer to the method in RFC 2630, which was copied from PKCS #7. Note also that as per Section 12.11 of PKCS #11, only block cipher mechanisms using the PKCS padding can be used to wrap and unwrap private keys.

c. Appendix F, what was the basis for only selecting the 1-bit, 8-bit and 128-bit CFB modes for AES? For example, FIPS Pub 185 defines the 1-bit, 8-bit, 16-bit, 32-bit and 64-bit CFB modes for SHIPJACK whereas PKCS #11 supports the 8-bit, 16-bit, 32-bit and 64-bit CFB modes with SHIPJACK.

d. Section 4.2.2, you should use the variable "s" instead of "x" and "y" for the least or most significant bits within the LSB and MSB operations to be consistent with the use of the variable "s" under Sections 5 and 5.3.

Than some minor editorial comments:

e. Section 2, in the last sentence of the first paragraph, there is an extra open bracket in front of "15 U.S.C. 278 g-3 (a)(5)".

f. Section 4.1, the definition for the abbreviations "CTR" and "ECB" are each missing a period at the end.

g. Section 5, in the first sentence, the reference to Sections 3.1-3.6 is incorrect.

Francois Rousseau
IT Standards, Senior Advisor - CSE
Conseiller Superieur, Normes TI - CST
francois.rousseau@cse-cst.gc.ca
(613) 991-8364
Edward Drake Building
Ottawa, Ontario

Cathy Robinson

Thank you for the opportunity to review the Draft Recommendation for Block Cipher Modes of Operations located at <http://csrc.nist.gov/publications/drafts.html>. This documents updates the modes of operation in FIPS Pub. 81, "DES Modes of Operation" for use with any FIPS-approved block cipher. The VA is pleased with the changes to FIPS 81. The changes allow our vendors to use AES encryption standards. AES is now the government preferred standard for encryption. AES is faster and stronger than triple DES (data encryption standard). The changes to FIPS 81 describe the ways that we are allowed to use the new AES standard. The key in AES is much stronger than the encryption key in triple DES. This is good for government. It keeps us ahead of the people trying to break encryption on government data.

The changes to FIPS 81 include the following:

a. It now incorporates the new Advanced Encryption Standard (AES) encryption algorithm, which is one of the principal reasons for the update. AES can be the underlying block cipher: and he allowed key sizes are 128, 192, and 256 bits. Examples are provided for 1-bit, 8-bit, and 128 bit CFB.

b. It points out the six modes of operations using symmetric key block cipher (CBC) algorithm. Five of the modes are confidentiality modes: Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter (CTR). This recommendation also defines an authentication mode based on the CBC mode (CBC-MAC). Used with an underlying block cipher algorithm that is approved in a Federal Information Processing Standard (FIPS), the modes in this recommendation can provide cryptographic protection for sensitive, but unclassified, computer data.

c. This recommendation builds on three FIPS publications that approve modes of operation of two particular FIPS-approved block cipher algorithms. FIPS Pub. 81 [4] specifies the ECB, CBC, CFB, and OFB modes of the Data Encryption Standard (DES). Modes that are essentially equivalent to the CBC-MAC modes of the DES are specified both in the appendix of FIPS Pub. 81 and in FIPS Pub. 113 [5]. FIPS Pub. 46-3 [3] approves the seven modes that are specified in ANSI X9.52 [1]. Four of these modes are equivalent to the ECB, CBC, CFB, and OFB modes with the Triple DES algorithm (TDEA) as the underlying block cipher; the other three modes in ANSI X9.52 are variants of the CBC, CFB, and OFB modes of Triple DES that use interleaving or pipelining.

d. The intro section points out -- This recommendation is neither a standard nor a guideline, and as such, is neither mandatory nor binding on Federal agencies. Federal agencies and non-government organizations may use this recommendation on a voluntary basis. It is not subject to copyright.

Our point of contact in VA's Office of Cyber Security is Robert Pate, 202-273-7554.

Carol Widmayer

The Department of the Treasury has no comments on the draft "Recommendation for Block Cipher Modes of Operation".

Carol Ann Widmayer
Department of the Treasury
CIO
Office of Information Systems Security
(202) 622-1110

Don McGinnis

Thank you for the opportunity to review the NIST draft recommendation Recommendation for Block Cipher Modes of Operation. The Environmental Protection Agency has no comments.

Don McGinnis
Technical Information Security Staff
Office of Technology Operations and Planning
Office of Environmental Information
202-260-8671

Sharon Shank

On behalf of Howard Landon, Acting CIO for the Department of Energy, this office has reviewed the Draft "Recommendation for Block Cipher Modes of Operation" that updates the modes of operation in FIPS Pub. 81, "DES Modes of Operation" for use with any FIPS-approved block cipher.

This office concurs with the document as written and has no additional comments to offer.

Should you have any questions, please contact me via email at sharon.shank@hq.doe.gov or by phone on (301) 903-3047.

Sharon L. Shank
Telecommunications Security Program Manager
Engineering and Assessment Division
Office of Cyber Security
Office of Chief Information Officer
(301) 903-3047
FAX: (301) 903-9661

Mark Winstead

It's great to see CTR mode included. However with CBC-MAC the only recommended authentication mode, the usefulness of CTR is somewhat neutralized.

CTR's strength is speed, but if you have to wait around for the CBC-MAC to finish, the CTR advantage is lost. We hope that NIST will add a faster MAC to its recommendations as soon as is possible to do so.

Mark Winstead

Senior Mathematician

NetOctave, Inc.

P.O. Box 14824

RTP, NC 27709

919.463.9903 x328

mark.winstead@netoctave.com

Patricia Black

The Treasury Office of the Chief Information Officer, Systems Security Staff, has reviewed the subject document. At this time we have no comments to offer. Thank you for the opportunity to participate in this review.

Nancy Cam-Winget

I would like to commend NIST for anticipating the need to update the modes of operation and moving forward with the two phase approach. I have reviewed the draft "Recommendation for Block Cipher Modes of Operation" and would like to submit my general comments.

1.. The current draft appropriately describes the recommended modes and uses. However, as was discussed in the 2nd Workshop, even the most rudimentary set of guidelines will prove useful. For instance, ECB has been shown to have weaknesses as a confidentiality mechanism; I strongly encourage some mention of either it's vulnerabilities or conditions in which one could attain good confidentiality.
2.. Explicit mention of IV properties should be included. For instance, CBC mode requires that nonces must not only be unique but also unpredictable.
3.. Is Appendix A meant to be more informative than recommendation? When choosing a padding scheme, is any example more suitable for certain conditions than others? If so, please list these conditions; if not, I would suggest only citing one example (and a corresponding test vector to verify against).
4.. There is also no mention of how to deal with the instance of no data blocks in Appendix A. I know that OCB can handle null (zero) data blocks,

but perhaps for the focus of Appendix A, explicit mention of a minimum of 1 data block is required (for padding).

5.. I like the inclusion of Appendix B and encourage more as mentioned in my first comment!

6.. I would like to see more guidance as to how one might compose a set of algorithms to achieve both confidentiality and authenticity. The guidelines should also be clear that unique keys must be used for each algorithm chosen.

7.. I strongly encourage the adoption of an authenticated encryption mode. As the wireless communications market grows, so does it's throughput rate. I believe that we will be reaching a point where a dual algorithm approach (one for achieving confidentiality and one for authenticity) will become prohibitive to attain (at least) in software. I believe that having an authenticated encryption mode that employs a single key and minimizes the number of cipher block calls will soon be a requirement. More explicitly, I favor OCB mode as it has nicely minized the offset computations and more importantly, reduced the number of cipher block calls to an optimal (n-blocks + 2) number.

8.. When evaluating authenticated encryption algorithms, they must also be able to handle authenticating data that is not necessarily included in the ciphertext. For instance, given plaintext data P and Q; we need to generate ciphertext Q" but authenticate both P and Q. This is a requirement for some networking datagrams.

9.. I strongly encourage the adoption of parallelizable algorithms. Again, as throughput rates increase, so will the need to increase the performance for encryption algorithms. Adopting parallelizable encryption algorithms will greatly improve our ability to strive for higher performance.

With Best Regards,

Nancy.

Nancy Cam-Winget

Atheros Communications, Inc.

529 Almanor Ave

Sunnyvale, Ca. 94085

(408)773-5317

John Smith

Thanks for the opportunity to review the Draft NIST "Recommendation for Block Cipher Modes of Operation". HUD's comments are below.

Comments on this draft

1) Page Three should cite the "Computer Security Act of 1987"

2) Introduction section should explain in layman's terms what is changed because of this new recommended standard. That is not clear to the reader who is not emersed in encryption terminology.

3) Several examples of how the new recommended standard can be used in sensitive unclassified systems would also be helpful. HUD has some systems that use encryption already and it would be nice to be able to recommend modifications based on the new standard.

John S

Tom Brand

This note is in regard to your recent publication "Recommendation for Block Cipher Modes of Operation". Please check the 1-bit CFB decryption examples. I believe the three examples for 128,192 and 256-bit decryption are in error. It appears the feedback bit applied to the shift register is always zero, when it should be the next cipher text bit.

Regards,

Tom Brand
Asic Engineer
SafeNet, Inc.
tbrand@safenet-inc.com
978-539-4808

John Savard

Should Counter Mode be officially accepted as one of the standard modes of encryption to be used with the new AES?

When I first read of the standard modes of encryption for DES, instinctively I thought that counter mode would be preferable to output feed-back mode, the keystream generator mode actually recommended for DES, because the danger of short cycles was obvious.

However, I realize now that there was a reason that counter mode was considered unacceptable. If initial counter values are not necessarily identical, but merely productive of overlapping sequences, then the messages involved are obviously vulnerable to attack.

However, even with OFB, while overlapping sequences are no longer trivially

obvious, they can still be detected by Kerckhoffs superimposition, so indeed it could be argued that this is not a valid objection to counter mode.

Still, it does make sense that requiring users to keep track of previous IV values, instead of simply choosing them at random each time, would unacceptably complicate the description of an encryption mode.

(It seems to me that the danger of short cycles with OFB could be reduced, and the presumed problems of counter mode avoided, if the two modes were combined: a counter value can be XORed with the block cipher output when it is fed back, but a comment is not the place for a proposal.)

Social Security Administration

Comments on National Institute of Standards and Technology (NIST) Special Publication 800-XX, *Recommendation for Block Cipher Modes of Operation*

Although this NIST "recommendation is neither a standard nor a guideline", and therefore "neither mandatory or binding", it is a welcome expansion on several previous FIPS publications. Its detailed explanations will be useful in helping determine the appropriate modes of operation for algorithms that Federal Agencies will use in future symmetric encryption applications.

The Social Security Administration (SSA) has already implemented at least one symmetric encryption application, Triple Data Encryption Standard (3DES) for its Virtual Private Network (VPN), which links remote users to SSA's internal data processing environment. While the VPN algorithm and mode of operation were established before this NIST publication was available, SSA can adapt VPN to meet the recommendation.

The NIST recommendation supports statutory requirements pertaining to overall computer security in federal agencies and federal agency participation in electronic commerce. It promotes secure networks operating effectively between federal agencies and with outside organizations.

If SSA needs to share information with other agencies or organizations who have adopted symmetric encryption and certain aspects of the NIST recommendation, SSA would have to agree to use the same key, algorithm, and mode of operation. The fact that the algorithms and modes of operation are specified and approved by NIST should make the agreement process easier.

SSA could comply with NIST guidelines and exchange information with another Agency without having to make extensive changes to existing software applications. It would have to insert an encryption or decryption software procedure before transmitting, or after receiving secured information.