Comments Received on the Draft HMAC Standard

Date: Thu, 29 Mar 2001 15:40:28 -0500
From: Carl Campbell <Carl_Campbell@compuserve.com>
Subject: HMAC FIPS PUB Draft
To: Elaine Barker <elaine.barker@nist.gov>

[...Personal greeting text deleted..]
I am writing you now about the "HMAC" FIPS PUB, especially concerning the MAC length.
In the mid '80's I was on the ANSI and ISO working groups for security of retail banking, and helped develop X9.19, which became IS 9807 (message authentication for retail banking), and is now incorporated in the new ISO 16609. These standards require only a 32-bit MAC, on the premise that adversaries who do not know the secret key could not predict the correct MAC value for a given message, and thus could only "guess" at what the MAC should be. Thus MAC determination was presumed to require and exhaustive "trial-and-error" procedure. With a 32-bit MAC, such a procedure would require the average of 2 billion trials if the adversaries could limit themselves to a single message, and the average of 4 billion trials if they were forced to try many different messages. This seems clearly a non-feasible number of trials, so 32 bits appears more than sufficient. The current NIST "HMAC" document at least permits the MAC to be as short as 32 bits (unlike ANSI X9.71, which mandates that it must be at least 80 bits), but states that the MAC "shall be at least L/2 bytes [80 bits for SHA-1] unless an application or protocol makes numerous trials impractical". The question is: How many trials are "numerous trials"? Presumably it is considerable less than 2-to-4 billion, because the NIST "HMAC" document goes on to state: "For example, a low-bandwidth channel might prevent numerous trials on a 4 byte MAC, or a protocol might allow only a small number of invalid MAC attempts." One would hardly consider 2-to-4 billion MAC attempts as being "only a small number".

From this I would surmise that nobody currently knows how to predict SHA-1 HMAC message-digest values, but that there is a nagging suspicion that, in the future, it may be possible for adversaries (who do not know the secret key) to predict that certain values are more probable that other values. Therefore if the number of invalid MAC attempts cannot be limited to "only a small number" (which one would presume to be something like 100), one should "play it safe" and go with the 80-bit MAC.However I note that ISO 9797-2 (message authentication mechanisms using a hash-function), in its evaluation of "guessing the MAC" (Annex B) indicates that a random guess (if the key is sufficiently long) is the best an adversary can do. Does NIST disagree with this view? The situation regarding MAC length is confusing. Therefore could NIST be any more specific about how "only a small number" or "numerous attempts" should be interpreted? It obviously makes a difference in those environments that currently use a 32-bit MAC (e.g. virtually all retail-banking environments), because it means that these environments cannot evolve to SHA-1 HMAC but instead must evolve to triple-DEA. (We presume that with triple-DEA it is impossible for adversaries to predict that some MAC values are more probable that others.) The MAC length is the primary issue about which I am concerned. A lesser issue is the requirement, in the NIST "HMAC" document, is that "Keys … shall be changed periodically". Manual distribution of new keys provides opportunities for physical compromise, and thus is undesirable. Automated key distribution is much simpler, but is of no value against physical compromise of the device, which would disclose the key-encrypting key as well as any data keys. Thus periodic key changes are primarily of value to deter analytical attacks on the key. We have always assumed with DEA that, as long as the key was double-length, periodic key changes were unnecessary, since key-exhaustion is the only possible analytical attack and this is non-feasible with a double-length key. I assume, however, that one does not necessarily have this assurance with SHA-1 HMAC, and therefore keys should be changed periodically. Is this assumption correct? If one should change SHA-1 HMAC keys periodically, does this alter the (presumed) ability of adversaries to predict that some MACs are more probable that other MACs? Finally, assuming that periodic key changes are necessary with SHA-1 HMAC, is NIST planning to endorse the use of SHA-1 HMAC itself to produce new keys? I have noticed that the proposed revision to PKCS #5 (Password-Based Cryptography Specification), which is also RFC2898, states that SHA-1 HMAC can be used to as a pseudorandom function for key derivation. Though this specification is for password-based keys (and therefore the HMAC "key" is the perhaps-predictable password, the HMAC data is a random but non-secret "salt", and many iterations are necessary), presumably the HMAC key could be a long (e.g. 20 byte) unpredictable long-term key, the HMAC data could be a non-secret "Key Change Counter", and only one iteration would be necessary. Thus new MAC keys could be created very frequently, simply by incremented the Key-Change Counter and applying SHA-1 HMAC using the 20-byte long-term secret key. The 20-byte resulting message-digest would then serve as the new MAC key. The MAC verifier, which would hold the same long-term key, could produce the current MAC key with knowledge of the non-secret Key-Change Counter. Is NIST planning to endorse such a procedure?

In summary, we would like to know if and how we can apply SHA-1 HMAC to retail banking, as an alternative to triple-DEA, especially using the same 32-bit MAC field that is specified today. Many thanks for any light you can shed on this subject.

Carl Campbell

From: "Don Johnson" <djohnson@certicom.com>
To: HMAC@nist.gov
Date: Fri, 5 Jan 2001 14:42:00 -0500
Subject: early comments on HMAC draft FIPS by Don Johnson, Certicom

Hi,
1) I think NIST should have a forum for public comments, similar to AES. Perhaps you are already planning this. And publish all the public comments, so others can see them as this promotes synergy of thought.
2) I think the symmetric key size equivalent table should be given relating common symmetric keysizes (80, 112, 128, 192 and 256 bits) with HMAC appropriate parameters. This makes it easy to get it right.
3) I do not see a need for lots of granularity for the key size between L/2 and L and it would aid interoperability to chunk it. (In fact I see L/2 becoming dominant, see below.) I suggest at least 2 byte chunks and even chunkier is probably better.
4) I do not see a need for lots of granularity for the HMAC output size and it would aid interoperability to chunk it. I suggest at least 2 byte chunks and even chunkier is probably better.
5) The absolute minimum HMAC output size of 4 bytes is very short. This can be guessed with a 1/2**32 (one in 4 billion) chance which is far from negligible, which I would say is about 1/2**80. Use of smaller than an 80 bit HMAC should have some more security discussion. I agree that such a shortened MAC can have a valid use, but more guidance on correct use and cost/benefits is needed. For example, even if I was only signing one message with a short HMAC, if the message was for lots of money, the HMAC should not be shortened.
5) Given the emphasis of storage and bandwidth constraints, I suspect that the minimum key size of L/2 and minimum normal MAC size of L/2 will become predominant, except for special cases. Given this, why not encourage it as the "normal expected interoperable choice" to use unless there are other special factors.

Thanks for your consideration,
Don Johnson, Certicom

From: FRousseau@chrysalis-its.com
To: HMAC@nist.gov
Subject: Comments on Draft FIPS for HMAC
Date: Thu, 8 Feb 2001 10:29:55 -0500

This message is in response to your request for comments on this draft Federal Information Processing Standard (FIPS) for the Keyed-Hash Message Authentication Code (HMAC).
Although the HMAC algorithm is similar in this draft FIPS for HMAC, RFC 2104, ANSI X9.71 and ISO/IEC FDIS 9797-2, there are still few minor differences that have been identified below between these four documents, which may cause some incompatibilities when validating products against all these standards:
a. Truncated Output: This draft FIPS for HMAC first indicates that a truncated output length shall be no less than four bytes and than no less than half the length of the hash output unless an application or the protocol makes numerous trials impractical. RFC 2104 and X9.71 both recommend that it be no less than half the length of the hash output and not less than 80 bits. 9797-2 specifies that it shall be less or equal to the length of the hash output and than indicates that the selection of the truncated output length is beyond the scope of this standard. This seems to imply that the FIPS for HMAC could allow 4 byte truncated HMAC in low bandwidth channel that prevent numerous trials whereas RFC 2104 and X9.71 would not allow truncated outputs smaller than 80 bits.
b. Minimum Key Size: This draft FIPS for HMAC indicates that the key shall be equal to or greater than half the length of the hash function output (e.g. 80 bits for SHA-1). RFC 2104 and X9.71 only recommend whereas 9797-2 specifies that the length for the key be at least the block size of the output to the hash function (e.g. 160 bits for SHA-1). RFC 2104 also indicates that smaller keys are strongly discouraged, as it would decrease the security strength of the function. X9.71 also indicates after discussing truncated outputs, that the key length must always be at least as great as the HMAC length. This seems to imply when using SHA-1 for X9.71 that 80 bit keys can only be used with a truncated output of 80 bits (e.g. HMAC-SHA1-80) whereas 9797-2 would only accept 160 bit keys and the FIPS for HMAC would allow 80 bit keys even with non truncated output (e.g. HMAC-SHA1).
c. Maximum Key Size: This draft FIPS for HMAC, RFC 2104 and X9.71 all require to first hash the key if the key itself is longer than the block size of the input to the hash function. 9797-2 instead specifies that the key shall be at most the length of the block size of the input to the hash function.
Here are other technical and editorial comments on this draft FIPS for HMAC:
a. Announcement, Section 8 on Implementation should be referring to FIPS 140-2 instead of FIPS 140-1.
b. Announcement, Section 9 on Other Approved Security Functions, the text from this section should probably match the similar section from FIPS 140-2. You would need to add "c. specified in the list of Approved security functions."
c. Section 1 and Appendix B should probably be referring to the original paper from Bellare, Canetti and Krawczyk titled "Keying Hash Functions for Message Authentication" from the Proceedings of Crypto'96 for the details on the rationale and security analysis of HMAC.
d. Section 1, first paragraph, in the last sentence, the word "specified" is misspelled and the reference to ANSI X9.71 is incorrect since it currently refers to ANSI X9.17.
e. Section 2.1, the definition for "FIPS-Approved" and "Secret Key" are slightly different than those from FIPS 140-2. The definition for "FIPS-Approved" is also terminated with two periods. Finally, the terms in the glossary are not appearing in alphabetical order.
f. Section 2.2, the acronyms should also address SHA-256, SHA-384 and SHA-512.
g. Section 2.3, the example for the block size of the input to the hash function should also address SHA-256, SHA-384 and SHA-512 (i.e. B=64 for SHA-1 and SHA-256 whereas B=128 for SHA-384 and SHA-512).
h. Section 2.3, the example for the FIPS-approved hash function should also address SHA-256, SHA-384 and SHA-512.
i. Section 2.3, the example for the block size of the output to the hash function should also address SHA-256, SHA-384 and SHA-512 (i.e. L=20 for SHA-1, L=32 for SHA-256, L=48 for SHA-384 and L=64 for SHA-512).
j. Section 2.3, for the "text" value, only X9.71 mentions, based on the underlying hash algorithm, the formula for the maximum length of the information that can be authenticated with the HMAC algorithm. A similar formula should probably be included in the FIPS for HMAC.
k. Section 3 and Appendix B, when discussing the fact that keys shall be generated at random, the document should probably be referring to FIPS 140-2 and the NIST Special Pub 800-22, A Statistical Test Suite for the Validation of RNGs and PRNGs for Cryptographic Applications, Dec 2000.

l. Section 5, for the Step 2 of HMAC algorithm, the result should be Ko = H(K) and not K = H(K).
m. Section 6 should mention the different OIDs applicable with this HMAC mechanism for the various underlying hash functions (i.e. for SHA-1, SHA-256, SHA-384 and SHA-512) including various truncated variants.
n. Appendix A only includes the same three test cases for HMAC-SHA1 and one truncated test case for HMAC-SHA1-96 as in ANSI X9.71, which originally came from RFC 2202, however some other interesting test cases from RFC 2202 with longer keys and/or longer text were not included. It should also include many other test cases with SHA-256, SHA-384 and SHA-512 including some with truncated variants, longer keys and/or longer text. Although it is acknowledged that it might be more appropriate to include all these test cases in a separate testing document, to help debugging HMAC implementations, it might be more useful to instead include in the FIPS for HMAC a complex example per hash function that provides "intermediate" computation values.
o. Appendix B, many of the listed references (e.g. FIPS113, FIPS171, ISO9797-2 and RFC2404) are not even referred in the document.
I thank you for the opportunity to review this draft FIPS for HMAC. If you would like to discuss any of these comments, please feel free to contact me.
Best regards,
Francois Rousseau
Director of Standards and Conformance
Chrysalis-ITS
One Chrysalis Way
Ottawa, Ontario, CANADA, K2G 6P9
frousseau@chrysalis-its.com Tel. (613) 723-5076 ext. 3419
http://www.chrysalis-its.com Fax. (613) 723-5078

From: Mike.Chawrun@CSE-CST.GC.CA
To: ebarker@nist.gov, foti@csmes.ncsl.nist.gov
Subject: FIPS - HMAC
Date: Fri, 19 Jan 2001 15:47:35 -0500

CSE completed cursory review of the proposed FIPS-HMAC and forward, for your consideration, 3 minor comments.
1. Section 2.3, Page 3
There may be some confusion as whether x'N' represents a four bit or 16 bit number.
CSE suggests the addition of the following statement as a note. Note: x'N' denotes that N is a hexadecimal (base 16) number.
2. Section 4, Page 3; 3rd last line
There is a mention of "trials." Does this refer to "attacks? If so CSE suggests to replace "trials' with "attack attempts." In the next line "trials" could be replaced with "attacks."
3. Appendix A, page 7
Suggest that Test Case 4 be changed to read "(20 byte key; 20 byte HMAC; 12 byte MAC)."

Date: Mon, 08 Jan 2001 11:03:12 +0100
From: Simon Josefsson <sjosefsson@rsasecurity.com>
To: HMAC@nist.gov
Subject: "HMAC Examples" reference

I believe the test cases in "Appendix A: HMAC examples" are from RFC 2202 "Test Cases for HMAC-MD5 and HMAC-SHA-1" (dated 1997). X9.71 (dated 2000) might have chosen the same ones, though. Perhaps the earlier reference should be used, or both.
/Simon

Date: Wed, 4 Apr 2001 14:45:11 -0500
From: Carl Campbell <Carl_Campbell@compuserve.com>
Subject: Comments on HMAC FIPS PUB
To: "HMAC@nist.gov" <HMAC@nist.gov>

Comments on the proposed FIPS PUB #HMAC, The Keyed-Hash Message Authentication Code (HMAC):
Although it is a Federal Standard only, any FIPS PUB is considered authoritative for commercial use. There is therefore some concern about the discrepancies between this FIPS PUB and current ISO "MAC" standards for use in retail banking. The latest ISO MAC standard, ISO/DIS 16609 specifies a minimum MAC length only for triple-DEA-based MAC algorithms, where the minimum length is 32 bits. There are no caveats as to when this length applies. Though this standard discusses HMAC, it does not specify any minimum MAC length but instead defers to ISO/IEC 9797-2. This latter standard also does not specify a minimum MAC length, but rather indicates that the MAC length must be sufficiently long to deter "guessing the MAC", which 9797-2 indicates "has a success probability max (1/2^m, 1/2^k)." (In this document "m" is the MAC length in bits, and "k" is the key length in bits.) Therefore the writers of this ISO/IEC standard believed that adversaries could not predict that certain MAC values were more probably than other MAC values.

From this standard we would conclude that, with SHA-1 HMAC as with triple-DEA, MAC-determination requires a "trial-and-error" procedure, for a 32-bit MAC (with a much longer key) requiring the average of 2 billion trials if the adversaries could limit themselves to a single message, and the average of 4 billion trials if they were forced to try many different messages. This seems clearly a non-feasible number of trials in almost any retail-banking environment, so 32 bits appears more than sufficient. Retail banking systems are set up for, and use, 32-bit MACs. It is therefore considered desirable, in some retail-banking applications, to evolve from single DEA to HMAC, rather than from single DEA to triple DEA. However, this FIPS PUB would throw doubt on such an evolution. The current HMAC FIPS PUB at least permits the MAC to be as short as 32 bits (unlike ANSI X9.71, which mandates that it must be at least 80 bits), but states that the MAC "shall be at least L/2 bytes [80 bits for SHA-1] unless an application or protocol makes numerous trials impractical". The question is: How many trials are "numerous trials"? Presumably it is considerable less than 2-to-4 billion, because the HMAC FIPS PUB goes on to state: "For example, a low-bandwidth channel might prevent numerous trials on a 4 byte MAC, or a protocol might allow only a small number of invalid MAC attempts." One would hardly consider 2-to-4 billion MAC attempts as being "only a small number".

From this we would surmise that adversaries (who do not know the secret key) might be able to predict that certain MAC values are more probable that other values when SHA-1 HMAC is used. Therefore if the number of invalid MAC attempts cannot be limited to "only a small number" (which one would presume to be something like 100), one should "play it safe" and go with the 80-bit MAC. The situation regarding MAC length is confusing. From the above, we assume that "NIST knows something that the writers of 9797-2 did not know", namely that knowledgeable adversaries in the future may be able to predict that some MAC values are more probable than others. If this is not the correct interpretation of the HMAC FIPS PUB, it would be very helpful if NIST would affirm that "trial and error guessing" is the only attack that MAC-length need deter. If this is the correct interpretation, it would be very helpful if NIST could be more explicit on the risk of using less than an 80-bit MAC for SHA-1 HMAC. We presume that a 32-bit MAC is still acceptable for a triple-DEA-generated MAC, because "trial and error guessing" in the only known attack apart from exhaustive key determination.

A lesser issue is the statement in the HMAC FIPS PUB that keys "shall be changed periodically". The DEA-based ISO and ANSI retail-banking standards are based on the premise that periodic key changes are unnecessary if the key length is sufficient (e.g. 112 bits) to prevent an exhaustive attack. A manual key change exposes the key to possible physical compromise, and therefore should be avoided unless a key compromise is suspected. Automated key changes, we have believed, were pointless because they prevented only a cryptanalytical attack on the key, which is presumably impossible if the key is of sufficient length. Therefore many retail-banking systems that use triple-DEA have no provision for automatic key changes, only manual key changes in the event of a suspected key compromise.

Is there something about SHA-1 HMAC that makes it more susceptible to key compromise (assuming it uses a 160-bit key) than triple-DEA? If one should change SHA-1 HMAC keys periodically, does this alter the (presumed) ability of adversaries to predict that some MACs are more probable that other MACs? What criteria should be used to determine the key-change frequency?

Finally, assuming that periodic key changes are necessary with SHA-1 HMAC, is NIST planning to endorse the use of SHA-1 HMAC itself to produce new keys for an automated key-distribution system? It would be helpful if NIST could be somewhat more explicit in the HMAC FIPS PUB about the need for "periodic key changes", and how they can be implemented.

In summary, the issue of "periodic key changes", but even more the issue of MAC length, is of concern to those of us who are involved with the security of retail banking.

Many thanks,
Carl M. Campbell, Jr.
Consultant