X-Sieve: CMU Sieve 2.2
Date: Tue, 13 Mar 2007 01:50:24 +0100 (CET)
From: Orr Dunkelman <Orr.Dunkelman@esat.kuleuven.be>
To: hash-function@nist.gov
Subject: A comment to NIST regarding the Hash function Competition
X-Virus-Scanned: by KULeuven Antivirus Cluster
X-Proofpoint-Virus-Version: vendor=fsecure engine=4.65.5502:2.3.11,1.2.37,4.0.164 definitions=2007-03-13_01:2007-03-12,2007-03-13,2007-03-12 signatures=0
X-PP-SpamDetails: rule=spampolicy2_notspam policy=spampolicy2 score=0 spamscore=0 ipscore=0 phishscore=0 adultscore=0 classifier=spam adjust=0 reason=mlx engine=3.1.0-0703060001 definitions=main-0703120085
X-PP-SpamScore: 0
X-NIST-MailScanner: Found to be clean
X-NIST-MailScanner-From: orr.dunkelman@esat.kuleuven.be

Dear NIST team,

In response to your request for comments regarding the hash function competition:

It appears that NIST is looking for the hash function with the best security/performance tradeoff. However, most hash functions are composed of two elements: a compression function, and an iteration mode (e.g., Merkle-Damgard).
A secure hash function must have both a secure compression function and a secure iteration mode. Of course, the combintion itself has to be secure as well.

It is very evident that having two competitions, the first one aimed at choosing the mode of iteration and then a second one for choosing the compression function is far from being practical (as it would cause a long delay in the selection of the new hash standard).

Thus, I believe that NIST would have to allow tweaks in the mode of iteration in a much more liberal way.

Technically, it would help the designers if NIST had announced in advance that some modes of iteration are to be favored, or even a standard API for the compression functions. Most of the recent modes require the compression function to accept additional parameters besides the chaining value and a message block. Thus, by requiring a support of additional input(s) to the compression function, NIST is likely to encourage people to use the more secure modes of iteration, while increasing the agility of the compression function designers to switch between various modes of iteration.

Orr Dunkelman.
--
Orr Dunkelman,
Katholieke Universiteit Leuven
Dept. Electrical Engineering-ESAT / COSIC
+32-16-321-049

X-Sieve: CMU Sieve 2.2
Date: Tue, 20 Mar 2007 16:06:59 -0400
From: Bill Burr <william.burr@nist.gov>
User-Agent: Thunderbird 1.5.0.9 (Windows/20061207)
To: Orr Dunkelman <Orr.Dunkelman@esat.kuleuven.be>
CC: Shu-jen Chang <shu-jen.chang@nist.gov>, Donna Dodson <ddodson@nist.gov>,
John Kelsey <john.kelsey@nist.gov>, Lily Chen <llchen@nist.gov>,
Rene Peralta <rene.peralta@nist.gov>, Quynh Dang <qdang@nist.gov>,
Jim Nechvatal <jnechvatal@nist.gov>, Tim Polk <wpolk@nist.gov>
Subject: Re: A comment to NIST regarding the Hash function Competition
X-NIST-MailScanner: Found to be clean
X-NIST-MailScanner-From: william.burr@nist.gov

Orr,

Thanks for your comment.
I presume that we have to be looking for some "best" security/performance trade off. Otherwise we, if we look only for security, then we get a hugely inefficient winner. But what is "best?"

So what are the parameters that you think we ought to have, and do they feed into the initial state or to the compression function? For example, they might include an IV and perhaps a round count, that feed into the initial state or iteration mode, and perhaps also a salt that feeds into the compression function. Do you have some more specific kinds of iteration modes or API in mind?

I assume that we would want to ask for default values, because the feedback last August was that a lot of folks who implement systems would never use these things (and probably didn't even want them to exist as settable parameters). Also we would have to have some understanding of which values we were actually going to test.

Would it make sense to ask for possibly different round counts for collision resistance and 2nd preimage resistance?

What is a "secure compression function?" That is does the compression function in isolation necessarily have testable security properties?
For an MD hash I suppose that we want a "collision free" compression function, but if we allow other iteration structures I'm not sure if that is the right condition at all.

Regards,

Bill
--
William. E Burr
Manager, Security Technology Group
301-975-2916

X-Sieve: CMU Sieve 2.2
Date: Sun, 1 Apr 2007 16:44:51 +0200 (CEST)
From: Orr Dunkelman <Orr.Dunkelman@esat.kuleuven.be>
To: Bill Burr <william.burr@nist.gov>
Cc: Shu-jen Chang <shu-jen.chang@nist.gov>, Donna Dodson <ddodson@nist.gov>,
John Kelsey <john.kelsey@nist.gov>, Lily Chen <llchen@nist.gov>,
Rene Peralta <rene.peralta@nist.gov>, Quynh Dang <qdang@nist.gov>,
Jim Nechvatal <jnechvatal@nist.gov>, Tim Polk <wpolk@nist.gov>
Subject: Re: A comment to NIST regarding the Hash function Competition
X-Virus-Scanned: by KULeuven Antivirus Cluster
X-Proofpoint-Virus-Version: vendor=fsecure engine=4.65.5502:2.3.11,1.2.37,4.0.164
definitions=2007-04-01_01:2007-03-30,2007-03-30,2007-04-01 signatures=0
X-PP-SpamDetails: rule=spampolicy2_notspam policy=spampolicy2 score=0 spamscore=0
ipscore=0 phishscore=0 adultscore=0 classifier=spam adjust=0 reason=mlx engine=3.1.0-
0703060001 definitions=main-0704010023
X-PP-SpamScore: 0
X-NIST-MailScanner: Found to be clean
X-NIST-MailScanner-From: orr.dunkelman@esat.kuleuven.be
Bill (and of course - all the NIST team),

[I presume that we have to be looking for some "best" security/performance trade off. Otherwise we, if we look only for security, then we get a hugely inefficient winner. But what is "best?"]

I would say that "best" is the option which ensures sufficient security margins at the best performance. Of course, defining what are sufficient security margins is not an easy task.

[So what are the parameters that you think we ought to have, and do they feed into the initial state or to the compression function? For example, they might include an IV and perhaps a round count, that feed into the initial state or iteration mode, and perhaps also a salt that feeds into the compression function. Do you have some more specific kinds of iteration modes or API in mind?]

As we suggested in the second workshop in Santa Barbara, we think that HAIFA's API is a sufficient one (in a paper submitted to ecrypt's hash function workshop we discuss instantiating HAIFA as a randomized hashing scheme, and enveloped Merkle-Damgaard scheme, and even the new ROX scheme).

I think that some salt/key/additional random input should be required even if HAIFA does not become the required API. I am aware that implementors are not too happy with "extra parameters", but I believe that having at least one of them should be sufficient to improve the security of Merkle-Damgaard significantly. Moreover, adding a salt/key/additional random input to the hash function (which can either affect the IV or be transfered to the compression function) should also be considered. In that case, the submitters have a more control on how and when to use the extra parameters.

[I assume that we would want to ask for default values, because the feedback last August was that a lot of folks who implement systems would never use these things (and probably didn't even want them to exist as settable parameters). Also we would have to have some understanding of which values we were actually going to test.]

In order to achieve security in all of the "recent" constructions the salt/key size should be something around n/4 to n/3 where n is the chaining value size. There might be people who

choose a wide pipe hash strategy, but then they can still enjoy the extra strength following the salt/key parameter.

[Would it make sense to ask for possibly different round counts for collision resistance and 2nd preimage resistance?]

I think this is a good security/implementation tradeoff. However, having too many variable parameters might prove a very annoying to the implementors. Thus, these are the guys who should be consulted in this matter.

[What is a "secure compression function?" That is does the compression function in isolation necessarily have testable security properties? For an MD hash I suppose that we want a "collision free" compression function, but if we allow other iteration structures I'm not sure if that is the right condition at all.]

Just like block ciphers, testable properties can only indicate flaws in the compression function (no matter how you define its security).

I think that assuring the three basic requirements (collision resistance, second pre-image resistance, and pre-image resistance) out of the hash function should lead the designers to pick the best strategy.

As a final note, I don't think that NIST should explicitly ask for iterated hashing, but I strongly believe that constructing secure hash functions with iterated hashing is possible, and then iterated hashing would probably be the most efficient one.

Regards,
Orr.

--
Orr Dunkelman,
Katholieke Universiteit Leuven
Dept. Electrical Engineering-ESAT / COSIC
+32-16-321-049