Cheetah hash function is not resistant against length-extension attack.

The mechanism in Cheetah to protect against length-extension attack is the permutation of the chaining value before the last invocation of the compression function. However, the initial chaining value of Cheetah is a zero vector of 256 or 512 bits. That means that every hashing of short messages that have length less than 959 bits will suffer from the trivial length-extension attack because the permutation of the initial zero vector is known to the attacker.

Best regards,
Danilo Gligoroski
Hi all,

we would like to make some clarification on the status of Cheetah. Gligoroski's observation showed that the IV is one of a few fixed points of the permutation which should prevent length-extension attacks. A simple change of the IV would make a length-extension attack on even short messages impossible. Therefore, we do not consider this observation as a break.

Another option, which however does not affect neither speed nor the security of compression function, would be to add to the last-round permutation a non-zero constant, which would remove any fixed points and completely avoid length-extension attacks.

So it would be good if editors of the following web-sites which currently list Cheetah as "broken" take note:

- skein-hash.info
- wikipedia
- etc.

Note also that Cheetah, though being AES-based hash functions, runs at remarkably high speed. Our recent implementation of Cheetah-256 runs at a speed of 9.3 cpb, while Cheetah-512 runs at 13.6 cpb.

--

Best regards,
Dmitry, Alex, Ivica

University of Luxembourg,
Laboratory of Algorithmics, Cryptography and Security,
Note also that Cheetah, though being AES-based hash functions, runs at remarkably high speed. Our recent implementation of Cheetah-256 runs at a speed of 9.3 cpb, while Cheetah-512 runs at 13.6 cpb.

Is this code available somewhere?

David Bauer
Subject: Re: OFFICIAL COMMENT: Cheetah
From: Dmitry Khovratovich <khovratovich@gmail.com>
Date: Fri, 6 Feb 2009 14:08:43 -0500
To: Multiple recipients of list <hash-forum@nist.gov>

Not yet, but we will publish it soon.

On Fri, Feb 6, 2009 at 7:26 PM, David Bauer <astgtciv2009@gatech.edu> wrote:

> Note also that Cheetah, though being AES-based hash functions, runs at
> remarkably high speed. Our recent implementation of Cheetah-256 runs
> at a speed of 9.3 cpb, while Cheetah-512 runs at 13.6 cpb.

Is this code available someplace?

David Bauer

--
Best regards,
Dmitry Khovratovich

University of Luxembourg,
Laboratory of Algorithmics, Cryptography and Security,
+ 352 46 66 44 5478
Hi all,

Cheetah now has its own webpage: http://cryptolux.org/cheetah, where the specification, updates, slides and code will host.

A new 64-bit assembler implementation (9.3 / 13.6 cpb for 256/512 bit digest, resp.) is also available there.

Comments are welcome.

--
Best regards,
Dmitry Khovratovich

University of Luxembourg,
Laboratory of Algorithmics, Cryptography and Security,
+ 352 46 66 44 5478
UPD.: the certificate of our web-server is self-signed so you probably get a security warning (we will resolve it soon). Please just choose the option "accept the certificate" when open the web-site.

On Fri, Feb 20, 2009 at 7:01 PM, Dmitry Khovratovich <khovratovich@gmail.com> wrote:

Hi all,

Cheetah now has its own webpage: http://cryptolux.org/cheetah , where the specification, updates, slides and code will host.

A new 64-bit assembler implementation (9.3 / 13.6 cpb for 256/512 bit digest, resp.) is also available there.

Comments are welcome.

--
Best regards,
Dmitry Khovratovich

University of Luxembourg,
Laboratory of Algorithmics, Cryptography and Security,
+ 352 46 66 44 5478

--
Best regards,
Dmitry Khovratovich

University of Luxembourg,
Laboratory of Algorithmics, Cryptography and Security,
+ 352 46 66 44 5478
Hi,

I think I have second preimage attack on un-salted Cheetah with complexity of $O(2^{(n/2)})$ computations and negligible memory.

Cheetah uses a sort of Rijndael block cipher in Davies-Meyer mode and HAIFA framework.

Let us call the used Rijndael-like block cipher as RijndaelCheetah. More precisely RijndaelCheetah(Key, PlainText) is a block cipher where Key = (Message_Block_of_1024_bits || Block_Counter).

Similarly, let us call Inverse_RijndaelCheetah(Key, CipherText) the inverse block cipher.

We are going to define two-block second preimage attack on Cheetah (meet-in-the-middle attack).

Let Cheetah(Unknown_Message) = H1.

The goal is to find a second preimage message $M=(M0, M1)$ consisting of two blocks, such that Cheetah(M) = H1.

Note that both blocks $M0$ and $M1$ are 1024 bits long.

Step 1. Fix the last 88 bits of $M1$, according to the definition of the padding of a message long $2048 - 88 = 1960$ bits.

Step 2. Fix also the last 88 bits of $M0$ to the same padding constant value as in $M1$.

Step 3. (Forward step) Generate $2^{(n/2)}$ different messages \{M0_i | i=1, ..., 2^{(n/2)} \} (with the fixed last 88 bits as defined in Step 2.) and compute $H0_i$ = LastBlockPermutation( RijndaelCheetah(M0_i, Block_Counter0, IV) XOR IV ), i=1, ..., $2^{(n/2)}$,

where Block_Counter0=0, and IV is any IV defined by the designers of Cheetah. In the current documentation IV=0, but in one OFFICIAL COMMENT the designers mentioned possibility to use a different IV. This attack works well no matter what IV was chosen.

Step 4. (Backward step) Generate $2^{(n/2)}$ different messages \{M1_i | i=1, ..., 2^{(n/2)} \} (with the fixed last 88 bits as defined in Step 1.) and compute $H1_i$ = Inverse_RijndaelCheetah(M1_i, Block_Counter1, H1), i=1, ..., $2^{(n/2)}$,

where Block_Counter1=1.

Step 5. With high probability, there is a matching pair (M0_i, M1_j) such that the corresponding $H0_i = H1_j$ i.e. Cheetah(M) = H1 where M = (M0_i, M1_j).

Remark: Since the domain for message blocks $M0_i$ and $M1_i$ is the same, we can launch a memoryless version of this attack described in memoryless birthday attack of van Oorschot and Wiener paper [1], and the total complexity of this attack is $O(2^{(n/2)})$ computations and negligible memory.

Regards,
Danilo Gligoroski
Clarification:

The described attack was based on the Figure 1 in the official Cheetah documentation where there is no last feed-forward. If there is a feed-forward, the attack as described is not possible.

Regards,
Danilo!

Hi,

I think I have second preimage attack on un-salted Cheetah with complexity of \(O(2^{(n/2)})\) computations and negligible memory.

Cheetah uses a sort of Rijndael block cipher in Davies-Meyer mode and HAIFA framework.

Let us call the used Rijndael-like block cipher as RijndaelCheetah. More precisely RijndaelCheetah(Key, PlainText) is a block cipher where Key = (Message_Block_of_1024_bits || Block.Counter).

Similarly, let us call Inverse_RijndaelCheetah(Key, CipherText) the inverse block cipher.

We are going to define two-block second preimage attack on Cheetah (meet-in-the-middle attack).

Let Cheetah(Unknown_Message) = \(H_1\).

The goal is to find a second preimage message \(M=(M_0, M_1)\) consisting of two blocks, such that Cheetah(M) = \(H_1\).

Note that both blocks \(M_0\) and \(M_1\) are 1024 bits long.

Step 1. Fix the last 88 bits of \(M_1\), according to the definition of the padding of a message long 2048 - 88 = 1960 bits.

Step 2. Fix also the last 88 bits of \(M_0\) to the same padding constant value as in \(M_1\).

Step 3. (Forward step) Generate \(2^{(n/2)}\) different messages \(\{M_0_i | i=1, ..., 2^{(n/2)}\}\) (with the fixed last 88 bits as defined in Step 2.) and compute \(H_0_i = \text{LastBlockPermutation( RijndaelCheetah}(M_0_i, \text{Block.Counter}_0, \text{IV}) \oplus \text{IV} \), \(i=1, ..., 2^{(n/2)},\)

where \(\text{Block.Counter}_0=0\), and IV is any IV defined by the designers of Cheetah. In the current documentation IV=0, but in one OFFICIAL COMMENT the designers mentioned possibility to use a different IV. This attack works well no matter what IV was chosen.

Step 4. (Backward step) Generate \(2^{(n/2)}\) different messages \(\{M_1_i | i=1, ..., 2^{(n/2)}\}\) (with the fixed last 88 bits as defined in Step 1.) and compute \(H_1_i = \text{Inverse_RijndaelCheetah}(M_1_i, \text{Block.Counter}_1, H_1), i=1, ..., 2^{(n/2)},\)
Hi, you are right, Figure 1 is incorrect.

There is a feed-forward, of course. See, e.g., the reference code, the conference slides, or the pseudocode (page 2).

On Tue, Apr 21, 2009 at 3:42 AM, Danilo Gligoroski <danilo.gligoroski@gmail.com> wrote:

Clarification:

The described attack was based on the Figure 1 in the official Cheetah documentation where
there is no last feed-forward. If there is a feed-forward, the attack as described is not possible.

Regards,

Danilo!

I think I have second preimage attack on un-salted Cheetah with complexity of
\( O(2^{n/2}) \) computations and negligible memory.

Cheetah uses a sort of Rijndael block cipher in Davies-Meyer mode and HAIFA framework.

Let us call the used Rijndael-like block cipher as RijndaelCheetah.
More precisely RijndaelCheetah(Key, PlainText) is a block cipher
where Key = (Message_Block_of_1024_bits || Block_Counter).

Similarly, let us call Inverse_RijndaelCheetah(Key, CipherText) the inverse block cipher.