Preimage attack on CubeHash512-r/4 and CubeHash512-r/8

Dmitry Khovratovich, Ivica Nikolić, and Ralf-Philipp Weinmann

University of Luxembourg

1 Description of CubeHash512-r/b

Internally, CubeHash512-r/b maintains a 128-byte state S. The input message M is divided into message blocks $M_i, i = 1, \ldots, k$ of b-bytes. Then, sequentially, for each message block M_i of b-bytes the following is done:

1. xor M_i to the first b bytes of the state S
2. transform the state S through r identical rounds

The exact definition of the state transformation is irrelevant for our attack. It is only important that this transformation is easily invertible.
The final output transformation is also invertible.

2 Preimage attack on CubeHash512-r/4 and CubeHash512-r/8

For finding preimages in these variants we will use the meet-in-the-middle (MITM) attack. Since the size of the intermediate state is 128 bytes = 1024 bits, we have to somehow decrease the attack space for MITM technique. This is done trivially. Going forward from the initial hash value, we create a set S_1 of $2^{\frac{1024-8b}{2}}$ different intermediate hash values. Similarly, going backwards from the target hash value, we create a set S_2 of the same amount of different hash values. Note that in the backward direction, we create the set S_2 without application of the last xor of the message word M_t. Then, if S_1 and S_2 have two elements, s_1 and s_2, respectively, such that s_1 and s_2 coincide on all, but the first b bytes, then by changing the message word M_t that is xor-ed to the elements of S_2, we can get another s_2' such that $s_1 = s_2'$ and therefore produce a preimage.

3 Complexity and memory

If we apply the classical MITM attack then we will need 2^{512-4b} computations and memory to attack CubeHash512-r/b. In order to reduce the memory requirement to negligible amount, we will use the memoryless version of the MITM attack described in [2]. The memoryless version, similarly like the memoryless version of the collision search attack [3], uses the cycle finding algorithm. Recall

that this algorithm only deals with one function. In the memoryless MITM attack we have two functions: 1)$f(x)$ - forwards from the initial value, and 2)$g(x)$ - backwards from the target hash value. Hence, in order to launch the memoryless birthday attack, a switching function $h(x)$ is introduced. It is defined as:

$$h(x) = \begin{cases} f(x), \text{if } cr(x) = 0 \\ g(x), \text{if } cr(x) = 1 \end{cases}$$

where $cr(x)$ is some function with a random behavior that outputs 0 and 1 with probability 1/2. Then, by straightforward application of the memoryless collision search algorithm, we can easily find a collision among the sets S_1 and S_2 and therefore a preimage for the hash value. The main requirement for the functions $f(x)$ and $g(x)$ is that have the same range and domain. Now, let us specify the functions $f(x)$ and $g(x)$ for our case. Let x in $f(x)$ (forward direction) be the first $1024 - 8b$ input words. Then $f(x)$ is the value of the last $1024 - 8b$ bits of the state obtained by hashing the input x. Similarly, let x in $g(x)$ (backward direction) be the last $1024 - 8b$ input words. So $g(x)$ is the value of the state (again the last $1024 - 8b$ bits) obtained when going backwards and taking x as the input. Hence the domain and the range for both $f(x)$ and $g(x)$ are the same, and the attack can be launched. The preimage attacks on CubeHash512-r/4 and CubeHash512-r/8 require 2^{496} and 2^{480} computations respectively and negligible memory.

4 Final remarks

The brute-force preimage search for CubeHash512-r/1 requires on average 2^{511} calls of the hash function on a 64-byte message. In our attack we need to prepare two sets of size 2^{508} each. Each state is the result of the hash function query on a 64-byte message. Thus our (memory-cost) attack needs about 4 times less computations than the brute-force. The memoryless modification is about three times more expensive [3] thus giving an attack of roughly the same complexity of the brute-force. As a result, we do not claim that we broke CubeHash512-r/1 though these versions actually have the same weakness.

References

1. D. J. Bernstein - CubeHash Specification (2.B.1) `http://cubehash.cr.yp.to/submission/spec.pdf`
2. H. Morita, K. Ohta, S. Miyaguchi: A switching closure test to analyze cryptosystems Advances in Cryptology CRYPTO 1991, LNCS 576, Springer-Verlag, 1992, p. 183-193.
3. J.-J. Quisquater and J.-P. Delescaille:How easy is collision search. New results and applications to DES. Advances in Cryptology - CRYPTO 1989, LNCS 435, G. Brassard, Ed., Springer-Verlag, 1990, p. 408-413.