Multicollisions of EDON-R hash function and other observations (preliminary version)

Vlastimil Klima[1]

Abstract
The main principle how to make n-bit EDON-R hash functions [1] resistant to generic multicollisions and multipreimages attacks ([2], [3]) is the 2n-bit width of internal chaining value. We show how to degenerate 2n-bit chaining value to n-bit chaining value (for n = 256, 512) by keeping the half of chaining value constant from the beginning. It circumvents the main principle and make EDON-R hash functions (for n = 256, 512) vulnerable to generic multicollisions and multipreimages attacks ([2], [3]) with small additional work factor.

We show several properties of EDON-R compression function, which could be interesting for the next study of collisions and preimages.

The first cryptanalysis of EDON-R was made in [4]. We present an independent research, partially overlaping with [4].

We want to note that this is preliminary version, that we present here only sketches of the proofs and that not all of the accompanied problems are completely solved.

1 Introduction

For the sake of simplicity we will deal only with the main variants of n-bit EDON-R, where n = 256, 512.

Quasigroup operation.
The compression function R of EDON-R uses quasigroup operation Q several times. Q has two n-bit arguments A, B and gives an n-bit result C = Q(A, B). These values are graphically expressed at the Fig. 1, where the arrow starts at A, goes through B and ends at C. C = Q(A, B) is defined as C = Q(A, B) = F(A) + G(B), where F: $\{0,1\}^n \rightarrow \{0,1\}^n$ and G: $\{0,1\}^n \rightarrow \{0,1\}^n$ are two special bijections (which are easily invertible). We will analyse EDON-R without using the internal structures of F and G.

Quasigroup rule (QR).
The definition of Q actually guarantees the following mathematical property:: having any two of the values (A, B, C) we can easily compute the remaining one.

[1] Independent cryptologist, Prague, Czech Republic, http://cryptography.hyperlink.cz, v.klima@volny.cz

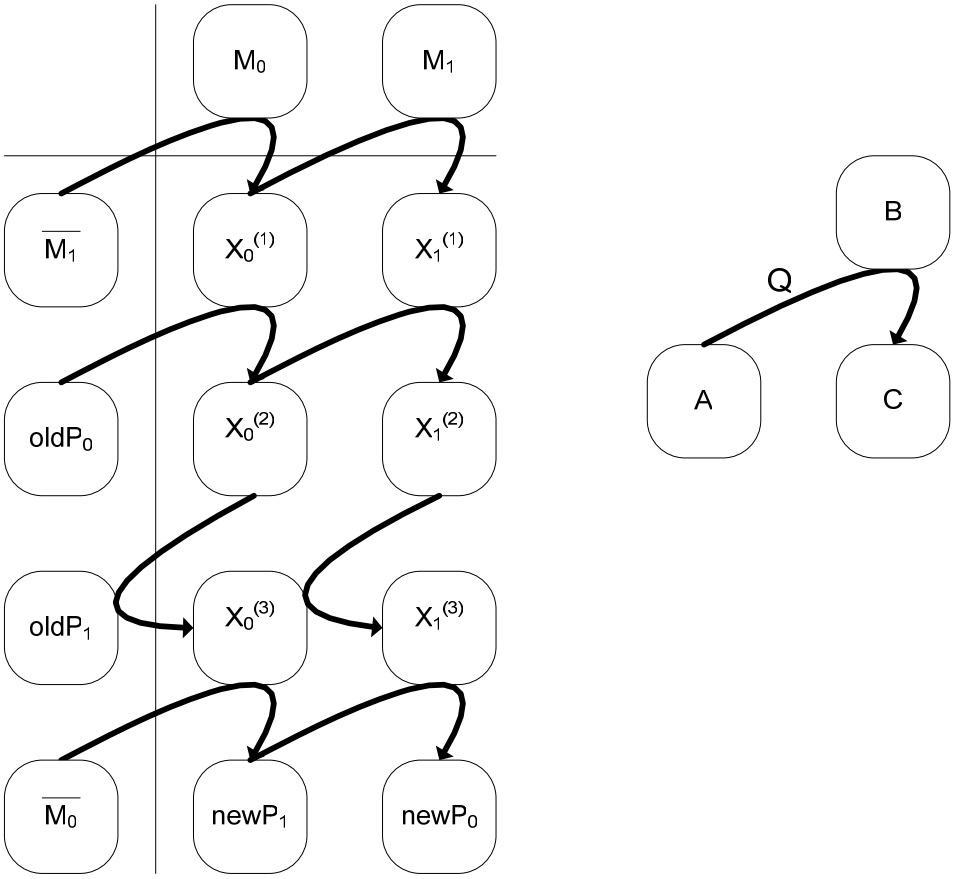

Fig. 1: The compression function R and the quasigroup operation Q

Compression function R.
EDON-R uses compression function R, which takes 2n-bit "old" chaining value oldP = ($oldP_0$, $oldP_1$) and 2n-bit block M = (M_0, M_1) and creates new 2n-bit chaining value newP = ($newP_0$, $newP_1$) = R($oldP_0$, $oldP_1$, M_0, M_1). Note that P_0, P_1, M_0 and M_1 are n-bit values (higher and lower halves of P and M).

Note 1 (Moving the rotation into the quasigroup operation).
Function R uses values of M_0 and M_1 at the top of the scheme on Fig.1 and rotated values of M_0 and M_1, which enter the scheme from the left side. For the sake of simplicity we omit the rotated values in the following description. Namely, they can be shifted into the quasigroup operations, leaving their QR property unchanged. Thus we will use two quasigroup operations Q and Q_1, where Q_1 is defined as $Q_1(A, B) = Q(rot(A), B)$ and the concrete bit rotation "rot" is specified in [1].

2 Collisions and fix points

Lemma 1 (Forward compression with fixed half of the chaining value).
There are functions f: $\{0,1\}^n \rightarrow \{0,1\}^n$, g: $\{0,1\}^n \rightarrow \{0,1\}^n$, such that for any values of $oldP_0$, $oldP_1$ and M_0, we have ($newP_0$, $newP_1$) = R($oldP_0$, $oldP_1$, M_0, M_1), where $M_1 = g(oldP_0, oldP_1, M_0)$, $newP_0 = f(oldP_0, oldP_1, M_0)$ and **$newP_1 = oldP_1$**.

Sketch of the proof.
Let us choose any $oldP_0$, M_0, $oldP_1$ (and $newP_1 \equiv oldP_1$). We will define the functions f and g in the following manner. Starting from values $oldP_0$, M_0 and $newP_1 \equiv oldP_1$ and using (QR) we can easily compute the values, marked as 1, 2, 3, 4, 5, 6, 7, 8 (in this order in Figure 2). We can define the function g as the expression No. 4 in terms of $oldP_0$, $oldP_1$ and M_0. Similarly, we define the function f as the expression No. 8 in terms of $oldP_0$, $oldP_1$ and M_0.

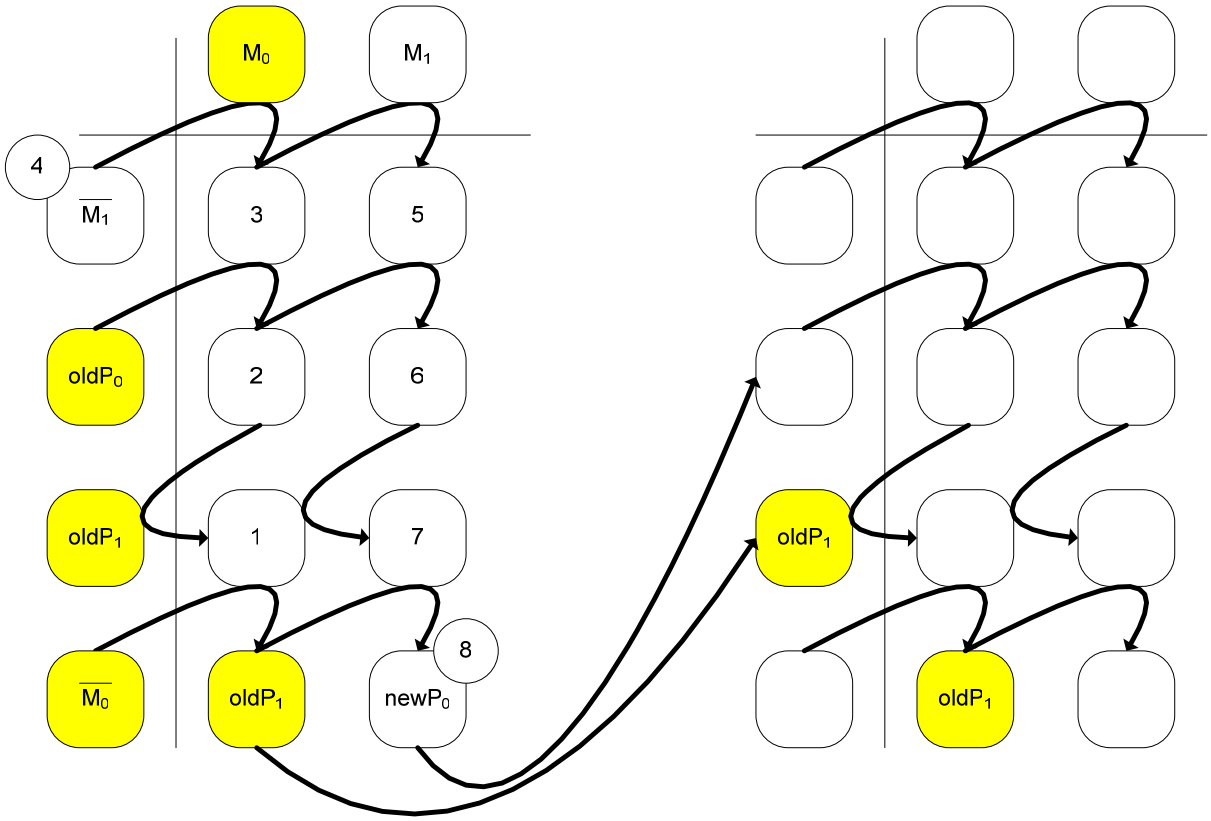

Fig.2: Forward compression, leaving the half of the chaining value constant

Theorem 1 (collisions and fix points of R).
(1) We can find a collision of 2n-bit compression function R, starting from the chaining value (IV_0, IV_1), with the same complexity as finding a collision of a n-bit hash function.
(2) We can find a fix point of 2n-bit compression function R, starting from the chaining value (IV_0, IV_1), with the same complexity as finding a collision of a n-bit hash function.
(3) For any value of (P_0, P_1) we can find a collision of 2n-bit compression function R, starting from the chaining value (P_0, P_1), with the same complexity as finding a collision of a n-bit hash function.
(4) For any value of (P_0, P_1) we can find a fix point of 2n-bit compression function R, starting from the chaining value (P_0, P_1), with the same complexity as finding a collision of a n-bit hash function.

Sketch of the proof.
(1)
First, we will describe how to obtain a collision of R, starting from the chaining value (IV), see Figure 3. Let us set oldP = IV. Choose $2^{n/2}$ different blocks $M_0[i]$ and using Lemma 1 compute the appropriate $M_1[i]$ and the new chaining value $newP = (newP_0, IV_1)$. In the set of $2^{n/2}$ values of $newP_0$ we will find an n-bit collision P_0. Let us denote appropriate blocks $M_0[i]$

and $M_0[j]$. The complexity of finding $M_0[i]$ and $M_0[j]$ is the same as finding a collision of a n-bit hash function (later we will describe that hash function as the projection of R). Then we have $R(M_0[i], M_1[i], IV_0, IV_1) = R(M_0[j], M_1[j], IV_0, IV_1) = (P_0, IV_1)$, what gives a 2n-bit collision for the compression function R (pseudo-collision of the hash function EDON-R).

(3)

From Lemma 1 it follows that we can start the attack (1) from any starting value (P_0, P_1), not only (IV_0, IV_1).

(2)

Now we will describe how to obtain a fix point of R (see the Fig. 4), starting the attack from (IV_0, IV_1). Let us choose $2^{n/2}$ different blocks $M_0[i]$. We can build one or several lines (starting from (IV_0, IV_1)) of various lengths as depicted on the Fig.4. In any point of the lines we are able to compute appropriate $M_1[i]$ (and new chaining variable $newP = (newP_0, IV_1)$ using Lemma 1. Thus the half of the chaining value is everywhere fixed to IV_1. Now, in the set of all $2^{n/2}$ values of $newP_0$ (in the first line or in the whole set of lines) we can find an n-bit collision P_0. Then we have fix point of R, starting from initializing value (IV_0, IV_1) and ending in the chaining value (P_0, IV_1) using two different ways (two different messages), possibly with different lengths.

(4)

From Lemma 1 it follows that we can start the attack (1) from any starting value (P_0, P_1), not only (IV_0, IV_1).

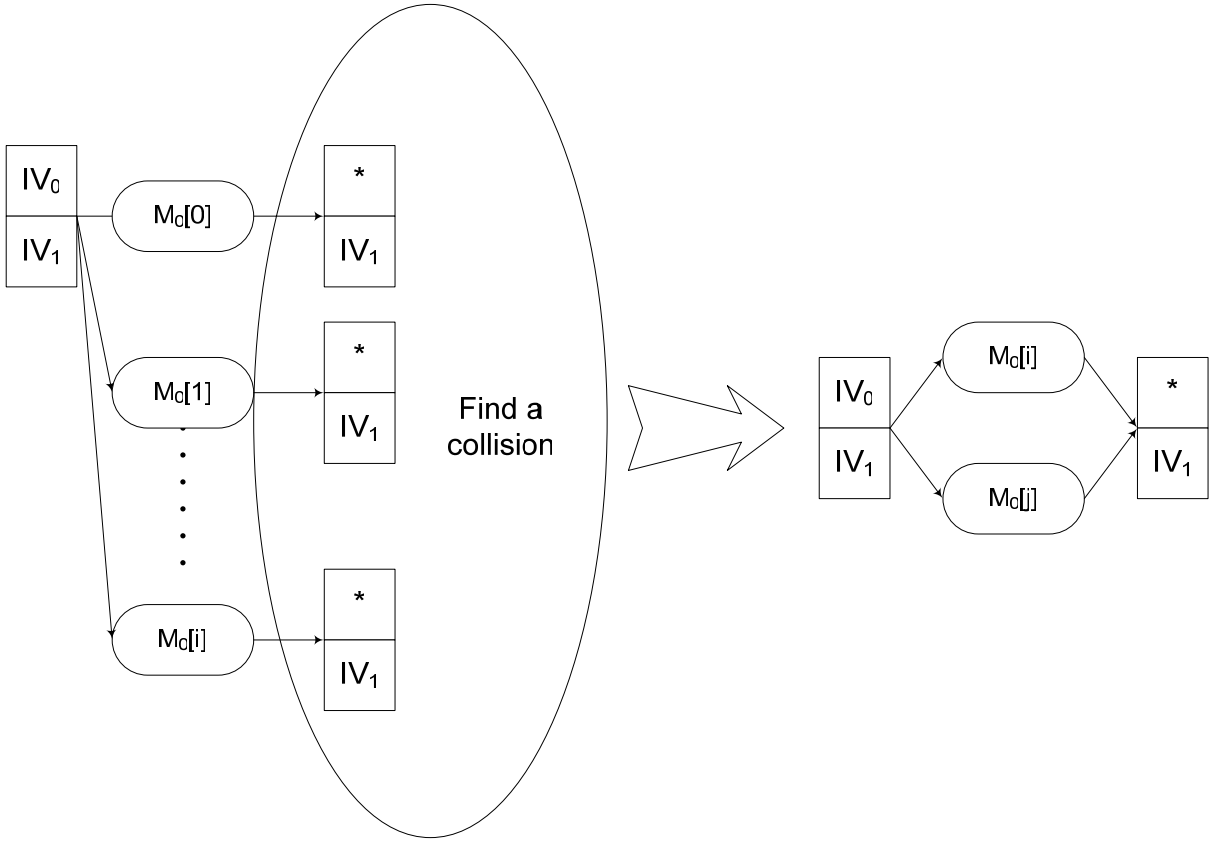

Fig. 3: Finding two different message blocks with the same input and output chaining value.

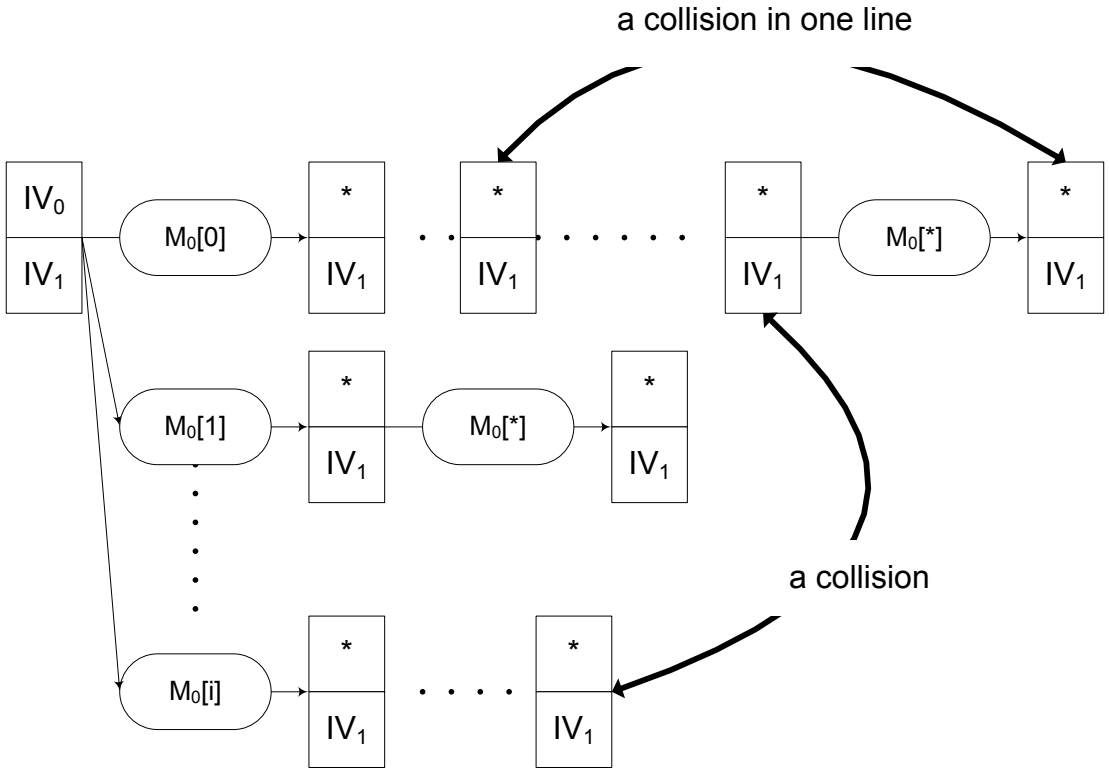

Fig. 4: Finding a fix point of the compressio function R and the hash function EDON-R

3 Multicollisions and multipreimages of EDON-R

Note 2 (multicollisions and multipreimages of EDON-R).
Theorem 1 gives us a tool for building multicollisions and multipreimages using directly generic attacks from [2] and [3] for n-bit hash function with n-bit chaining value. Both types of attacks have a common base - the underlying hash function and also a common problem with the last block.

The last block problem.
We can build multicollisions in the known way directly using Theorem 1. We start from the IV and find two different blocks, which lead to the same chaining value. Then we start from this chaining value and find two different blocks, leading to the same chaining value etc. (or we can start hashing a prefix message, leading to common starting value (as it was IV)). In any case we end in the same chaining value. It remains to ensure that the last block will contain valid padding. The problem is in the fact, that the message blocks have the form [M_0, M_1] =[M_0, g(oldP_0, oldP_1, M_0)] and padding is in the place, where the value g(oldP_0, oldP_1, M_0) is stored. So the value g(oldP_0, oldP_1, M_0) has to end with correct padding value.

Theorem 2 (The projection of EDON-R to dR).
We can project n-bit hash function EDON-R with 2n-bit chaining value to a n-bit hash function with n-bit chaining value. It enables the attacker to use generic multicollisions and multipreimages attacks ([2], [3]) against it with small additional work factor.

Sketch of the proof.
Let R is 2n-bit compression function of n-bit hash function EDON-R. It works with 2n-bit chaining value $P = (P_0, P_1)$ and 2n-bit message block $M = (M_0, M_1)$. The chaining value starts from 2n-bit constant $IV = (IV_0, IV_1)$ and the hash value is n-bit half (P_0) of the final 2n-bit chaining value $P = (P_0, P_1)$.

We can define n-bit compression function dR (degenerated R), working with n-bit message block M_0 and n-bit chaining value P_0, starting from the initialization value IV_0. This function leads to the same n-bit hash value as EDON-R, but has no precautions against multicollisions and multipreimages attacks (because the chaining variable has the same width as the hash value).

We can define the degenerated version of R by choosing special value of M_1 in the message block $M = (M_0, M_1)$ and setting the variable P_1 constant ($= IV_1$) in any step of R.
Now, when $P_1 = IV_1$, we can define the second part of the message block as a function of the first part and actual chaining value: $M_1 = g(P_0, IV_1, M_0)$. Now from the chaining value $P = (P_0, P_1)$ and this message block $M = (M_0, M_1)$ we obtain new chaining value
$R(P_0, IV_1, M_0, g(P_0, IV_1, M_0))$, what is according to Lemma 1, equal to $(newP_0, IV_1)$. Also we have $newP_0 = f(oldP_0, oldP_1, M_0)$, but it is not important. The fact that the next step will start from the value $(*, IV_1)$ enables us to define the "projection" correctly.
We define the projection of the function R as $dR(P_0, M_0) \equiv R(P_0, IV_1, M_0, g(P_0, IV_1, M_0))$, for any values of (P_0, M_0). dR is in fact a new compression function which has n-bit chaining value P_0, n-bit message block M_0 and starts from n-bit initializing value IV_0. The compression function dR thus creates an appropriate hash function (dEDON-R), which has n-bit hash value equal to its last n-bit chaining value. Thus we have obtained the new hash function, which has the same hash value as EDON-R. Because its chaining variable has the same length as the hash value itself, there are no precautions of dEDON-R against multicollisions or multipreimages attacks.

After finding multicollisions (or multipreimages) of dR, using generic attacks ([2] and [3]), we need to project them back to the function R.
From any value of the chaining variable P_0 and message block M_0 of dR we can create the chaining variable (P_0, IV_1) and the message block $(M_0, g(P_0, IV_1, M_0))$ of R. According to Lemma 1, we have valid state of the function R, excepting the final block, going to the compression function.

Note that the final block is the only point, when we cannot set the second part of the message block arbitrarily (we need to set it to the special value $g(P_0, IV_1, M_0)$). The reason is that (at minimum) the last 64 +1 bits of this block have to be set to specified number, the padding. This number is the smallest padding string, consisting of bit 1 and 64-bit number of hashed bits. So, the last 65 bits of $g(P_0, IV_1, M_0)$ should be equal to the specified number. This will happen with probability 2^{-65}. The "brute solution" is to exclude any invalid block from the set of obtained multicollisions (or multipreimages), what means to divide the number of them by 2^{65}. Even if 2^{65} is a big number, it is negligible when comparing it to $2^{n/2}$ (n = 256, 512).
There are other ways how to bypass this "last block problem", but in this sketch of the proof we won´t deal with them.

Note 3 (Mounting an attack from any starting value).
Note that we can start the attack (the projection) from Theorem 2 from any starting point. For instance, we can use a prefix message, which ends with some value of the chaining variable (P_0, P_1). From this point we can project R to dR.

Note 4 (Length extension attack using fixpoints).
The last block problem is that we need to have the last 65 bits of $g(P_0, IV_1, M_0)$ equal to bit 1 and the 64-bit number of hashed bits. Using a fix point in the center of the message, we can lengthen the message by a lot of message blocks. In this way we can correct the length of the message such that the bits 64, 63,..., 10, 9 (9 for n =256, 10 for n =512) from the end of $g(P_0, IV_1, M_0)$ - the number of blocks - are correct.

Note 5 (Complexity of multicollision attack).
Using Theorem 1 and techniques from [2] we can create 2^K multicollisions of EDON-R. The complexity of the attack is roughly $K*2^{n/2}$ hash computations and $2^{n/2}$ memory (n = 256, 512). We will create one couple of colliding message blocks as is depicted in the Fig. 3 with the complexity of $2^{n/2}$ hash computations and $2^{n/2}$ memory. K such couples will take the complexity of $K*2^{n/2}$ hash computations and $2^{n/2}$ memory (we reuse the memory). This sequence of dR we project back to R using Theorem 2. Thus we have now the sequence of K couples of blocks, ending in the same final chaining value (of the function R). We simply add one 2n-bit message block with the correct padding and process it with the function R. This will create the 2^K multicollisions of EDON-R.

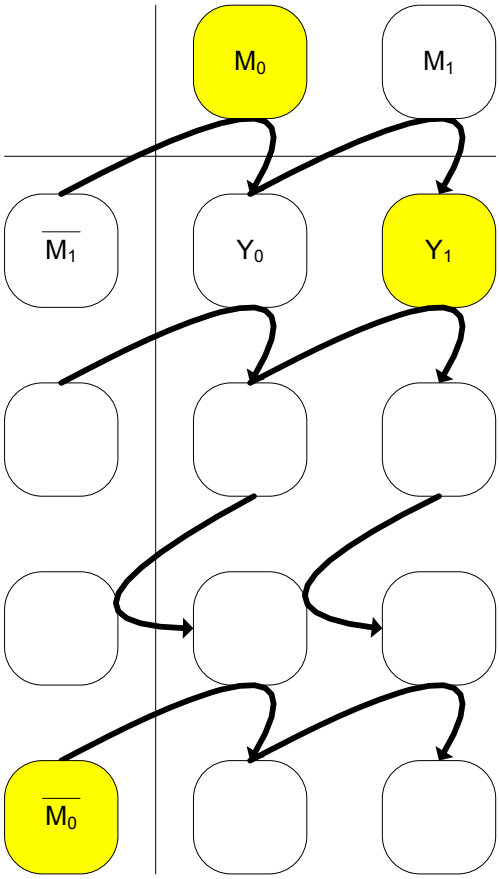

Fig. 5: A tiny internal differential invariability in the compression function

4 Random walk backwards

Theorem 3 (A tiny internal differential invariability in the compression function R).
For any M_0, there exist with high probability two different $M_1^* \neq M_1^{\sim}$, one Y_1 and two different $Y_0^* \neq Y_0^{\sim}$, such that
$Y_0^* = Q(\text{rot } M_1^*, M_0)$ and $Y_1 = Q(Y_0^*, M_1^*)$,
$Y_0^{\sim} = Q(\text{rot } M_1^{\sim}, M_0)$ and $Y_1 = Q(Y_0^{\sim}, M_1^{\sim})$,
where "rot" is the rotation, described in Note 1.
We can rewrite the theorem in this form: For a given fixed M_0 we can find two different inputs M_1^* and $M_1^{\sim}$ in the scheme of R (Fig. 5) such that there will be two different values of Y_0 (Y_0^* and $Y_0^{\sim}$), but only one value of Y_1. So, the differences in M_1 can´t be canceled in Y_0, but will be canceled in Y_1.

Sketch of the proof.
We substitute $Y_0^* = Q(\text{rot } M_1^*, M_0)$ into $Y_1 = Q(Y_0^*, M_1^*)$ and obtain
$Y_1 = Q(Q(\text{rot } M_1^*, M_0), M_1^*) = F(Q(\text{rot } M_1^*, M_0)) + G(M_1^*) = F(F(\text{rot } M_1^*) + G(M_0)) + G(M_1^*)$. We also substitute $Y_0^{\sim} = Q(\text{rot } M_1^{\sim}, M_0)$ into $Y_1 = Q(Y_0^{\sim}, M_1^{\sim})$ and obtain similarly $Y_1 = F(F(\text{rot } M_1^{\sim}) + G(M_0)) + G(M_1^{\sim})$. So we want to find two different solution M_1^* and $M_1^{\sim}$ of the equation $Y_1 = F(F(\text{rot } M_1) + G(M_0)) + G(M_1)$, where $G(M_0)$ is a constant. Because F and "rot" are bijections on $\{0,1\}^n$, we have a new bijection F_1 on $\{0,1\}^n$, defined by the relation $F_1(M_1) = F(F(\text{rot } M_1^*) + \text{const})$. We will find the two solutions of the equation $Y_1 = F_1(M_1) + G(M_1)$. It is possible to expect that the expression $F_1(M_1) + G(M_1)$, for M_1 going through all values of $\{0,1\}^n$, determines random function $f:\{0,1\}^n \rightarrow \{0,1\}^n : M_1 \rightarrow F_1(M_1) + G(M_1)$. So, in the range $f(\{0,1\}^n)$ there will be many collisions. We can find one of them by birthday paradox. We generate $2^{n/2}$ of different random values M_1 and then find a collision in the f-image of them.
Complexity of this solution is the same as finding a collision of a n-bit hash function.

Note 6 (Using the pair M_0, M_1^* and M_0, $M_1^{\sim}$ only once).
In the following text it suffices to use only one pair of **M_0, M_1^* and M_0, $M_1^{\sim}$,** computed according to the proof of Theorem 3. But for variability or for any other purposes we can generate several such pairs for different M_0.

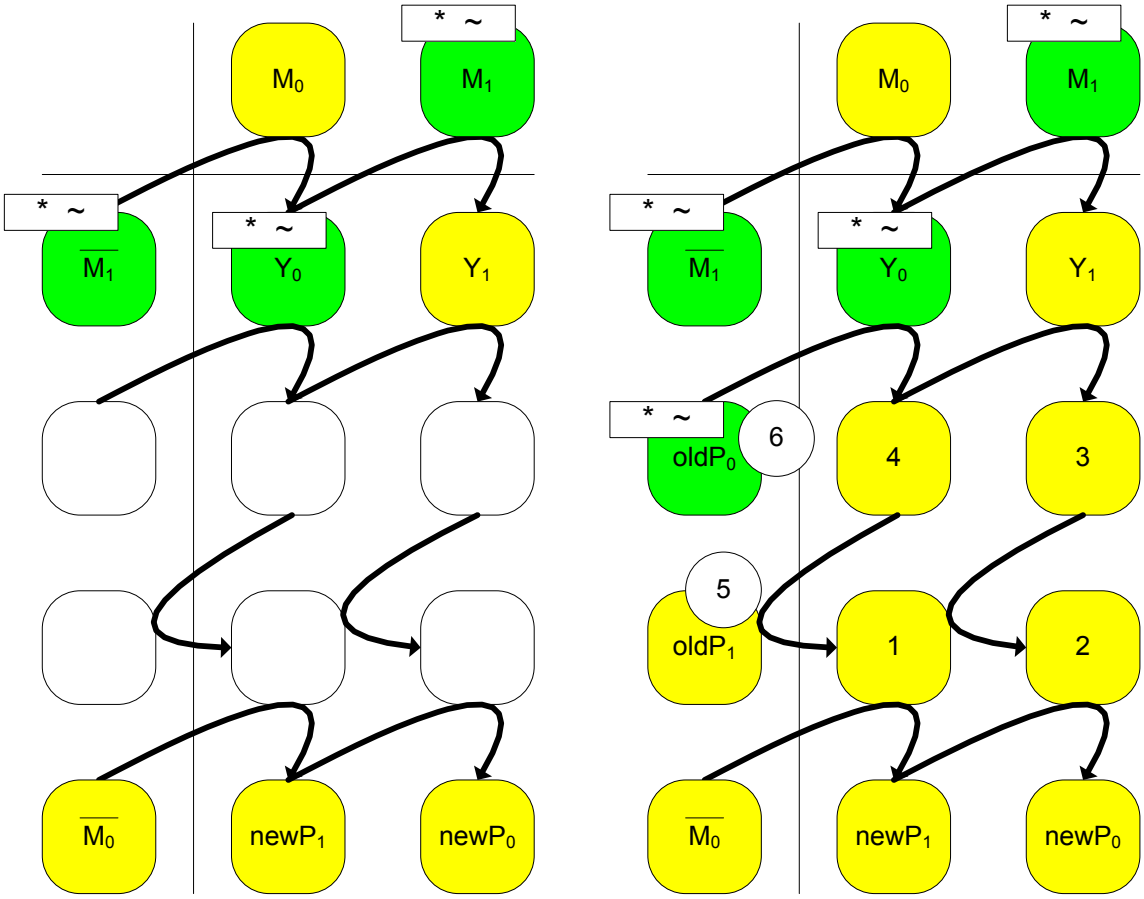

Fig. 6: Using the differential in half of the message block M_1

Theorem 4 (Random walk backwards).
Let us choose one of the solutions of Theorem 3 (M_0, M_1^* and M_0, $M_1^\sim$). Let us set the chaining variable of R to any value ($newP_0$, $newP_1$). Then we can easily compute two different preimages of R, ($oldP_0^*$, $oldP_1$) and ($oldP_0^\sim$, $oldP_1$) such that
$R(oldP_0^*, oldP_1, M_0, M_1^*) = R(oldP_0^\sim, oldP_1, M_0, M_1^\sim) = (newP_0, newP_1)$. These preimages have half of the chaining variable ($oldP_1$) the same, whereas the remaining part is different.

Sketch of the proof.
The computation of ($oldP_0^*$, $oldP_1$) and ($oldP_0^\sim$, $oldP_1$) can be seen on the Fig.6. Let us first compute the values Y_0^* and $Y_0^\sim$ and Y_1 according to the Theorem 3. The values Y_0^* and $Y_0^\sim$ are different, but leading to the same value of Y_1. The same values are yellow, the different values are green and they are marked with * and ~. Now we will compute consequently values marked 1, 2, 3, 4, 5, 6 in this order. We will compute them using the (QR). We can see that in computing values 1 - 5, we always use the two arguments of the (QR) constant (yellow). Only when we compute $oldP_0$ in the step 6, we are using two different values of Y_0. This gives two different values of $oldP_0$ ($oldP_0^*$ and $oldP_0^\sim$).

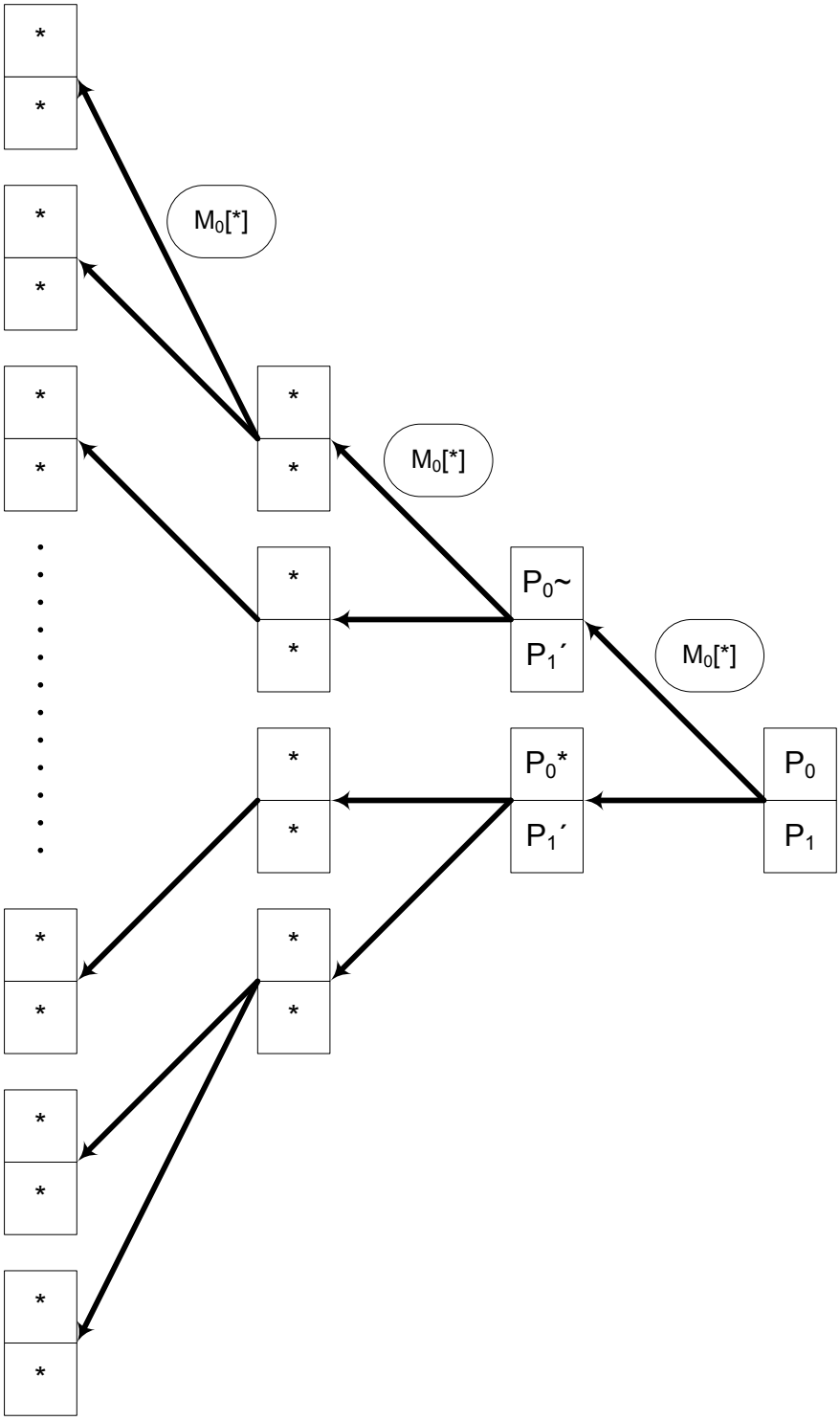

Fig. 7: Random walk backwards

Note 7 (Getting multi-preimages of R).
According to the Theorem 4 we can start from any value of chaining variable (for instance hash value, padded by n random bits) and go one step backwards. We obtain two different preimages of the chaining variable of R. In the second step backwards we obtain 2^2 preimages etc. and in the m-th step we have 2^m preimages (of the final hash and even of the final chaining value).

Conclusion

We show several properties of EDON-R compression function, which could be interesting for the study of collisions and preimages.

We show for instance how to degenerate 2n-bit chaining value of EDON-R to n-bit chaining value. It makes EDON-R hash functions (for n = 256, 512) vulnerable to generic multicollisions and multipreimages attacks ([2], [3]) with small additional work factor. The complexity of obtaining 2^K multicollisions is $K*2^{n/2}$ (hash) computations and $2^{n/2}$ memory.

Acknowledgements

We would like to thank Danilo Gligoroski for helpful comments.

References

[1] Gligoroski D., Odegard R.S., Mihova M., Knapskog S.J., Kocarev L., Drapal A.: Cryptographic Hash Function EDON-R, October 2008, SHA-3 submission, http://people.item.ntnu.no/~danilog/Hash/Edon-R/Supporting_Documentation/EdonRDocumentation.pdf

[2] Joux A.: Multicollisions in iterated hash functions. Application to cascaded constructions, Proceedings of CRYPTO 2004, Spinger-Verlag, LNCS, Vol. 3152, pp. 430 – 440, 2004.

[3] Kelsey J., Schneier B.: Second preimages on n-bit hash functions for much less than 2^n work. Proceedings of EUROCRYPT 2005, Spinger-Verlag, LNCS, Vol. 3494, pp. 474 – 490, 2005

[4] Khovratovich D., Nikolic I., Weinmann R. P.: Cryptanalysis of Edon-R, http://lj.streamclub.ru/papers/hash/edon-r.pdf