# A first report on Whirlpool, NUSH, SC2000, Noekeon, Two-Track-MAC and RC6[*] NES/DOC/UIB/WP3/007/b[†]

Lars Knudsen and Håvard Raddum

7th March 2001

## 1 Whirlpool

### 1.1 Brief description of algorithm

Whirlpool is a hash function that operates on messages less than $2^{256}$ bits in length, and produces a message digest of 512 bits. Whirlpool uses the Miyaguchi-Preneel scheme with a 512 bit block cipher, and it is claimed to be collision resistant and one-way. The bit string to be hashed is always padded, and its length after padding is a multiple of 512. The message string is then divided into a sequence of 512-bit words, $m_1, m_2, \ldots, m_t$. This sequence is then used to generate a new sequence of 512-bit words, $H_1, H_2, \ldots, H_t$ in the following way. $m_i$ is encrypted with $H_{i-1}$ as key, and the resulting ciphertext is XORed with $m_i$ and $H_{i-1}$ to produce $H_i$. $H_0$ is a string of 512 0-bits, and the final word $H_t$ is the output of the algorithm.

The block cipher, W, that forms the basis of Whirlpool is very similar to the AES selection Rijndael. The main difference between W and Rijndael is that Rijndael supports blocklengths of 128, 192 and 256 bits, while W only works on 512-bit blocks. The plaintext block in W is regarded as an 8x8 array of bytes, called a *state* and each byte is viewed as an element in GF($2^8$). The polynomial that defines this field is different from the one used in Rijndael. W consists of 10 rounds, preceded by an XOR of the first round key. As in Rijndael, each round starts with each byte of the state passing through an S-box. After this, each column of the state is rotated downwards a number of positions. This is followed by a multiplication with a fixed 8x8 matrix over GF($2^8$) with maximal branch number, the state being the left factor. Finally, the round ends with a key addition.

[*]This work has been supported by the Commision of the European Union through the IST program under contract IST-1999-12324.

[†]The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

### 1.2 Security issues

Rijndael has received quite a bit of analysis in the AES process, and given the similarities between Rijndael and W, some of it may carry over to Whirlpool. The designers state three criteria that defines the level of security. Assume we take as hash result any $n$-bit substring of the output of Whirlpool, then the criteria are:

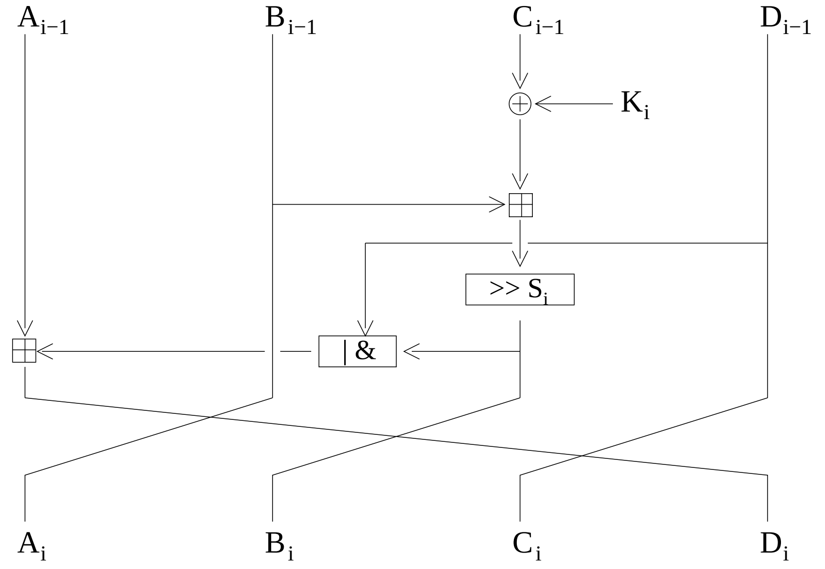
- The workload of generating a collision is expected to be of the order $2^{n/2}$ (collision resistant).
- Given an $n$-bit value, the workload of finding a message that hashes to that value is of the order $2^n$ (one-way).
- Given a message and its hash result, the workload of finding a different message that has the same hash value is of the order $2^n$.

There have not been reported any attacks on Whirlpool, falsifying any of the statements above.

## 2 NUSH

### 2.1 Brief description

NUSH is a block cipher that accepts 128-, 192- or 256-bit keys, and with block lengths that can be 64, 128 or 256 bits. Each block consists of four registers, which are mixed with key material and transformed through a number of rounds. For the 64-, 128- and 256-bit block versions, the number of rounds are 9, 17 and 33 respectively. Whitening keys are added to the block before the first round, and after the last round. Each round consists of four iterations where the registers are mixed with each other and with key material. In each iteration two of the registers are changed in a non-linear way, and the registers are rotated as in a type-3 Feistel network. The details of one round are given in the following figure.



The number of bit rotations, $S_i$, follows a schedule, and is different in each round. The box with '—&' in it denotes the logical OR or AND operations, a schedule defines which one is used in the different rounds.

## 2.2 Security issues

NUSH does not make use of any S-boxes, the only operations used are AND, OR, XOR, addition mod $2^n$, $n$=16, 32 or 64, and bit rotations. Like IDEA, it depends on the mixing of non-commutative operations for confusion and diffusion. The submitters have not given any rationale for their design choices, or done any security analysis of the cipher. Nessie has received a paper by Wu Wenling that describes a linear attack with complexities less than that of exhaustive key search for the different variants of NUSH. After a first look at the paper, the attack seems valid, in which case NUSH must be considered broken.

The important finding in the attack is a linear approximation between some of the bits in a block before and after one iteration that holds with probability 0.25 or 0.75 depending on whether AND ($\wedge$) or OR ($\vee$) is chosen in the iteration. Either way, the bias of the approximation is $2^{-2}$. The probabilities come from the fact that for two bits $x$ and $y$, $x \wedge y = x \oplus y$ with probability 0.25, and that $x \vee y = x \oplus y$ with probability 0.75. Let $X[0]$ denote the least significant bit of $X$. Referring to the figure, it is then obvious that $B_{i-1}[0] = A_i[0]$ (*). When two bitstrings are added together mod $2^n$, the addition in the least significant bits are always a simple XOR. By tracing backwards from $B_i$, through the '—&'-box and the addition with $A_{i-1}$, it's not difficult to see that the following relation holds with probability 0.25 or 0.75: $D_i[0] = A_{i-1}[0] \oplus D_{i-1}[0] \oplus B_i[0]$. Rearranging the terms, and combining with (*) we get

$$A_i[0] \oplus B_i[0] \oplus D_i[0] = A_{i-1}[0] \oplus B_{i-1}[0] \oplus D_{i-1}[0]$$

with bias $2^{-2}$. This approximation can be iterated through the cipher to create an approximation that has bias far enough from 0.5, so that it can be used in a linear attack.

# 3 SC2000

## 3.1 Brief description of algorithm

SC2000 is a 128 bit block cipher, with key lengths of 128, 192 or 256 bits. The cipher works on four 32-bit words, and has 6.5 rounds when the key length is 128 bits, and 7.5 rounds for 192-, and 256-bit keys. Each round starts with a key addition, followed by 32 parallel S-box look-ups where the bits in the same position in the four words form the input to the S-box. After this there is another key addition. Then the words are sent through an unkeyed two round Feistel network to create diffusion. The round function in the Feistel part takes two words as input. Each of these words is sent through a layer of S-boxes, made up of four instances of a 5-bit S-box and two instances of a 6-bit S-box. Each word is then multiplied with a fixed 32x32 binary matrix. Finally the words are mixed with each other, using the AND operation with a constant and XOR, to produce two words of output from the function in the Feistel network. This two-round Feistel structure is the last operation in one round of SC2000.

### 3.2 Security issues

A differential attack on a 4.5-round version of the cipher has been reported by the authors. This attack finds 28 bits in the first and last round key. The key schedule in SC2000 appears to be very strong. The knowledge of one round key doesn't seem to leak any information about any of the other round keys or the key selected by the user, so the key schedule prevents the attacker from searching exhaustively for the remaining key bits. The success of this attack raises some questions about the design of SC2000.

## 4 Noekeon

### 4.1 Brief description

Noekeon is a 128-bit block cipher that uses a 128-bit key, and operates on four 32-bit words, say $a_0, a_1, a_2, a_3$. It consists of a round function that is repeated 16 times. Each round starts with the addition of a round constant to $a_0$. Then $a_0$ and $a_2$ are XORed together to make the word $w$, and two copies of $w$ are made. One of the copies are rotated 8 bits to the left, and the other is rotated 8 bits to the right. These rotated copies are then XORed back onto $w$, and $w$ is XORed onto $a_1$ and $a_3$. After this the 128-bit working key is added to the four words. Then $a_1$ and $a_3$ are used to create a temporary $w$ in the same way as described above, and this word is XORed onto $a_0$ and $a_2$. The words $a_1, a_2$ and $a_3$ are then rotated 1, 5 and 2 bits, respectively, to the left. Then, for all 32 positions in a word, the bits that are in the same position in the different words are passed through an S-box. Finally, the round ends with the words $a_1, a_2, a_3$ being rotated 1, 5 and 2 bits, respectively, to the right.

After the last round, the linear operations described before the rotations of $a_1, a_2, a_3$ is repeated yet another time.

### 4.2 Security issues

The key schedule in Noekeon consists of taking the user selected key, and encrypting it once with the all zero key. The ciphertext is called the working key, and is the key used in the cipher. The fact that the same key is used in all rounds has not revealed any weaknesses yet. Another part of the cipher that might be vulnerable to attack is the S-box, which is the identity function on four of the sixteen different inputs.

## 5 Two-Track-MAC

### 5.1 Brief description

Two-Track-MAC is based on the unkeyed hash function RIPEMD-160, and has a key length of 160 bits. First, the message to be authenticated is padded with a 1-bit, and then 0-bits until its length is -64 mod 512. Then the binary representation of the length of the original message (mod $2^{64}$) is appended, so the length of the message becomes a multiple of 512. Each 512-bit block is split into a set of sixteen 32-bit words, $M_1, \ldots, M_{16}$. The secret key is a set of five 32-bit words, $K_1, \ldots, K_5$. The algorithm works by iterating a round function as follows. Two sets of five 32-bit words $X_1, \ldots, X_5$ and $Y_1, \ldots, Y_5$ and a set of 16 message words are input to two different functions $f_L$ and $f_R$ that outputs five 32-bit words each.

$$A_1, \ldots, A_5 = f_L(X_1, \ldots, X_5, M_1, \ldots, M_{16})$$

$$B_1, \ldots, B_5 = f_R(Y_1, \ldots, Y_5, M_1, \ldots, M_{16})$$

These two functions are identical to the ones used in RIPEMD-160. Then we compute $C_i = A_i - X_i$ and $D_i = B_i - Y_i \bmod 2^{32}$, for $1 \leq i \leq 5$. To finish the round, the ten words $C_i$ and $D_i$ are mixed in two linear transformations $g_L$ and $g_R$.

$$E_1, \ldots, E_5 = g_L(C_1, \ldots, C_5, D_1, \ldots, D_5)$$

$$F_1, \ldots, F_5 = g_R(C_1, \ldots, C_5, D_1, \ldots, D_5)$$

$E_i$ and $F_i$, together with the next set of message words, form the inputs to the next round. In the first round, the five secret keywords are input as both $X_i$ and $Y_i$, $1 \leq i \leq 5$. In the final round, where the last message words are input, $f_L$ and $f_R$ swap places. After the subtraction of the input words, instead of executing $g_l$ and $g_R$ at the end of the round, we compute $E_i = C_i - D_i \bmod 2^{32}$, $1 \leq i \leq 5$. The results from these subtractions form the output of Two-Track-MAC.

### 5.2 Security issues

The functions $f_L$ and $f_R$ consist of 80 iterations of a base function, and use the different bitslice operations AND, OR, XOR and bit complementation. The base functions also include bit rotations and addition mod $2^{32}$, and iterated 80 times it seems very hard to trace unknown key bits through it. No weaknesses in Two-Track-MAC have been reported.

It is also worth noting that the only place where key material is used is in the first round. The 160-bit output of the algorithm is the difference between two 160-bit quantities, so the knowledge of this difference still gives 160 bits of uncertainty about which two values produced it. In other words, guessing what these two values are, and computing backwards to find the input, i.e. the key, is no faster than guessing on the key directly.

## 6 RC6

### 6.1 A brief description

RC6 is a secret-key block cipher which was a candidate for the Advanced Encryption Standard (AES). RC6 is an evolutionary development of RC5. Like RC5, RC6 makes essential use of data-dependent rotations. New features of RC6 include the use of four working registers instead of two, and the inclusion of integer multiplication (squaring) as

```
Input:      Plaintext stored in four $w$-bit registers $A, B, C, D$
            Number $r$ of rounds
            $w$-bit round keys $S[0], ..., S[2r + 3]$

Output:     Ciphertext stored in $A, B, C, D$

Procedure:  $B = B + S[0]$
            $D = D + S[1]$
            for $i = 1$ to $r$ do
               {
                  $t = (B \times (2B + 1)) << \lg w$
                  $u = (D \times (2D + 1)) << \lg w$
                  $A = ((A \oplus t) << u) + S[2i]$
                  $C = ((C \oplus u) << t) + S[2i + 1]$
                  $(A, B, C, D) = (B, C, D, A)$
               }
            $A = A + S[2r + 2]$
            $C = C + S[2r + 3]$
```

Figure 1: Encryption with RC6-$w/r/b$.

an additional primitive operation. RC6 is a parameterized family of encryption algorithms, where RC6-$w/r/b$ is the version with word size $w$ in bits, with $r$ rounds and with an encryption key of $b$ bytes.

In the following we briefly recall the description of RC6, see Figure 1. The user-key has length $b$ bytes and the $4w$-bit plaintext block is loaded into words $A, B, C, D$. These four $w$-bit words also contain the ciphertext at the end. The key-schedule expands the user-key into subkeys $S[0], S[1], ..., S[2r + 3]$ by using the encryption routine of RC6. Here we shall not give a detailed description of the key-schedule. To describe the encryption algorithm the following notation is used: $(A, B, C, D) = (B, C, D, A)$ means the parallel assignment of values on the right to registers on the left. Moreover, $a \times b$ denotes integer multiplication modulo $2^w$, $a << \lg w$ means fixed rotation of the $w$-bit word $a$ by $\lg w$, the base-two logarithm of $w$, and $a << b$ denotes rotation of $a$ to the left by the amount given by the least significant $\lg w$ bits of $b$.

## 6.2 Known results

The AES submission is the version with $w = 32$, $r = 20$, and RC6 is a short hand notation for this version, whereby the key length can be $b = 16, 24$, and 32 bytes, respectively. The designers of RC6 has evaluated the algorithm with respect to differential and linear cryptanalysis. It was concluded that RC6 is secure with respect to differential cryptanalysis for 12 or more rounds. For linear cryptanalysis, some variants were considered. It was found that a two-round iterative linear approximation leads to the most effective basic linear attack applicable up to 13 rounds. However, no specific method for key-recovery was given. Furthermore, the designers sketched some potential enhancements of linear attacks using multiple approximations and linear hulls, and it was estimated that 16 rounds of RC6 can be attacked using about $2^{119}$ known plaintexts. These additional considerations on linear cryptanalysis were used to set a suitable number of rounds for RC6 to be $r = 20$.

Knudsen and Meier devised a correlation attack based on the linear approximations found by the designers which is faster than an exhautive key search for the 128-bit version of RC6 with up to 12 rounds, and for the 192-bit and 256-bit versions of RC6 with up to 14 and 15 rounds. Similar results were reported by Gilbert, Joux, Handschuh and Vaudenay. These results are in no contradiction with the designers estimations but do provide more concrete evidence of the strength of RC6.