Organization: ESAT, K.U.Leuven, Belgium
Date: Fri, 28 Aug 1998 11:36:03 +0200 (METDST)
From: **Johan Borst** <Johan.Borst@esat.kuleuven.ac.be>
Reply-To: Johan Borst <Johan.Borst@esat.kuleuven.ac.be>
To: AESFirstRound@nist.gov
cc: Bart Preneel <preneel@esat.kuleuven.ac.be>,
Vincent Rijmen <rijmen@esat.kuleuven.ac.be>,
Bart Van Rompay <vrompay@esat.kuleuven.ac.be>
Subject: analysis CRYPTON

Hello,

Regarding the call of NIST for comments on the AES-candidates here is a report of a preliminary analysis of CRYPTON, that identifies a class of weak keys.
It is attached in gzipped postscript format.

Regards,
Johan Borst.

Johan Borst
Katholieke Universiteit Leuven
Departement Elektrotechniek-ESAT/COSIC
Kardinaal Mercierlaan 94
B-3001 Heverlee
Belgium

Tel.: ++ 32 16 321862
Fax : ++ 32 16 321986

email: Johan.Borst@esat.kuleuven.ac.be
http://www.esat.kuleuven.ac.be/~borst

Weak Keys of CRYPTON

Johan Borst

K.U. Leuven, Dept. Elektrotechniek-ESAT/COSIC
Kardinaal Mercierlaan 94, B-3001 Heverlee België
Johan.Borst@esat.kuleuven.ac.be

Abstract

The block cipher CRYPTON is a candidate proposal for the AES standard. In this report we describe a class of 2^{32} weak keys. This is mainly a consequence of the use of linear operations in the key schedule. These weak keys especially have consequences for the use of CRYPTON in certain hash function constructions.

1 Introduction

The AES candidate CRYPTON is an iterated block cipher with block size 128. Its key size is variable and can be chosen to be each whole number of bytes between 8 and 32. Hence, the key sizes 128, 192 and 256 are supported, as requested by NIST for the AES standard. The design is based on SQUARE [DKR97].

In this report we describe a class of 2^{32} weak 256-bit keys. When such a key is used for encryption plaintexts from certain subsets of the text space always have ciphertexts in a specified subset. This is mainly caused by the use of linear transformations and symmetrical constants in the key schedule. The weakness for example can be exploited to find collisions when CRYPTON with a 256-bit key is used in the Davies-Meyer hash function construction.

The rest of this report is organized as follows. In Section 2 we describe the features of CRYPTON that are relevant for our analysis. In Section 3 we describe the weak key class and in Section 4 its consequences. We conclude in Section 5.

2 The Cipher

CRYPTON is defined as a 12-round block cipher preceded by an initial transformation and followed by a final output transformation. Each round transformation is composed of four different, consecutive transformations. All transformations operate on 16-byte strings. The round transformations for odd and even rounds are slightly different, though have a same structure. They are defined as

$$\begin{aligned} \rho_{K_r}(A) &= (\sigma_{K_r} \circ \tau \circ \pi_o \circ \gamma_o)(A), \text{ odd rounds: } r = 1, 3, \ldots, \\ \rho_{K_r}(A) &= (\sigma_{K_r} \circ \tau \circ \pi_e \circ \gamma_e)(A), \text{ even rounds: } r = 2, 4, \ldots, \end{aligned}$$

for $K_i, A \in \{0,1\}^{128}$. The initial transformation is defined by σ_{K_0}. The final output transformation is defined by $\tau \circ \pi_e \circ \tau$. In the following we omit the o and e subscripts as the properties we use hold for both even and odd round transformations.

The round keys[1] K_i each consist of four 32-bit words which we denote with $(K[4i+3], K[4i+2], K[4i+1], K[4i])$. The expanded keys are derived from the user key by a key schedule as follows. The first 8 expanded keys $K[i], i = 0 \ldots 7$, are derived from the user key via an invertible transformation. After that for each $i \in \{0 \ldots 7\}$ each expanded key $K[i+8j], j = 1 \ldots 6$ only depends on $K[i]$.

For further details we refer to [Lim98].

3 The Weak Keys

The only operations that are used in the second phase of the key scheduling are rotations with a multiple of 8 and xoring with round constants of the form $aaaa$, where a are byte values. This has as consequence that if for one of the first eight expanded keys $K[i] = aaaa$ for some byte value a, then all derived expanded keys $K[i+8j] = bbbb$ for some b, not necessarily the same b for all j. Furthermore, the xoring and rotations are done in such a way that, if for a pair $(i_1, i_0) \in \{(7,5),(6,4),(3,1),(2,0)\}$ the expanded keys $K[i_1] = K[i_0] = aaaa$, then $K[i_1 + 8j] = K[i_0 + 8j] = bbbb$.

With the above we can construct 2^{32} user keys for which every round key is of the form $bbbbaaaabbbbaaaa$. These user keys can be computed by inverting the first phase of the key schedule. In this way we find 2^{32} 256-bit (or 32-byte) keys of which about $2^{27.7}$ have an equivalent 31-byte userkey. We call this set of keys W.

We define three subsets of the text space as follows.

$$\begin{aligned} V_0 &= \{x \in \{0,1\}^{128} | x = ghghefefcdcdabab\}, \\ V_1 &= \{x \in \{0,1\}^{128} | x = ghefcdabefghabcd\} \text{ and} \\ V_2 &= \{x \in \{0,1\}^{128} | x = efghabcdefghabcd\}, \end{aligned}$$

where $a \ldots h$ are byte values. Each of the subsets contains 2^{64} elements.

For the key addition σ with a round key of the form $K = bbbbaaaabbbbaaaa$ as well as for γ it holds that if $x \in V_i$ then also $\sigma_K(x) \in V_i$ and $\gamma(x) \in V_i$, for all i. For the other two transformations the following holds. π maps an element of V_0 to an element of V_1, V_1 to V_0 and V_2 to V_2. This is clarified in Appendix A. Finally, τ maps V_0 to V_2, V_1 to V_1 and V_2 to V_0.

From this it can be seen that a CRYPTON round with a round key of the form $bbbbaaaabbbbaaaa$ maps an element of V_0, V_1 and V_2 to an element of respectively V_1, V_2 and V_0. Furthermore the final output transformation maps V_0, V_1 and V_2 to respectively V_0, V_2 and V_1.

Hence, for each of the keys in W CRYPTON (with 12 rounds) will encrypt an element of V_0, V_1 and V_2 to an element of respectively V_0, V_2 and V_1.

[1] In [Lim98] a distinction is made between round keys for encryption and decryption, given by K_e^i and K_d^i. As the properties described here hold for both, we omit the subscript and lower the superscript.

4 Consequences

There is only a small probability that a weak key is used when a user key is generated randomly. On the other hand with a few chosen plaintexts it can be easily tested if a weak key of W is used for encryption. However, the weak key class certainly influences security if an attacker can choose the key. An example of that is the following.

In [Lim98] it was mentioned that CRYPTON can be used as underlying block cipher in the Davies-Meyer hash function construction. Here the hash H_m on a message $x_0 x_1 \dots x_m$ is defined recursively by

$$H_i = H_{i-1} \oplus E_{x_i}(H_{i-1}),$$

where H_0 is an initial vector. If $H_{i-1} \in V_0$ for some i and $x_i \in W$, then $E_{x_i}(H_{i-1}) \in V_0$ and also $H_i \in V_0$. In accordance with the birthday paradox by varying $H_0 \in V_0$ and $x_0 \in W$ one finds a collision for H_1 with probability ≈ 0.39 in 2^{16} hash calculations. We generated 1000 sets of 2^{16} according to the above method generated hash calculations. 427 sets contained at least one collision, which is slightly higher than expected. Note that to mount a practical attack we should also have to allow an attacker to be able to influence the initial vector as the probability that an $H_i \in V_0$ is again very small.

Remark: Due to the non-linear first phase of the key schedule, there is no obvious way in which the algebraic properties of this section might be used to mount an attack on the Matyas-Meyer-Oseas or the Miyaguchi-Preneel hash function constructions.

5 Conclusion

We conclude that CRYPTON has a class of 2^{32} weak 256-bit keys. Hence care has to be taken when CRYPTON with 256-bit keys is used in applications where an attacker can choose the key, such as the Davies-Meyer hash function construction. A way to resolve this weakness is to strengthen the key schedule. This can be done by introducing more non-linearity in the key schedule, as for example was done in Rijndael [DR98], whose design also was based on SQUARE.

References

[DKR97] J. Daemen, L.R. Knudsen, V. Rijmen, "The block cipher SQUARE," *Fast Software Encryption, Proc. Fourth International Workshop, LNCS 1267*, Springer-Verlag, 1997, pp. 149–165.

[DR98] J. Daemen and V. Rijmen, *"AES Proposal: Rijndael,"*, AES Proposal, 1998, available via http://www.esat.kuleuven.ac.be/˜rijmen/rijndael/index.ht ml.

[Lim98] C. H. Lim, *"CRYPTON: A New 128-bit Block Cipher,"*, AES Proposal, 1998, available via http://crypt.future.co.kr/˜chlim.

A The Transformation π

In this appendix we do not distinguish between the π transformations of even and odd rounds as the properties we use hold for both. The transformations of CRYPTON can be viewed upon as operating on a 4×4 byte array. The transformation π consists of four separate transformations, which are transformations on the separate columns of the array. We can thus write $\pi(x_3, x_2, x_1, x_0) = (\pi_3(x_3), \pi_2(x_2), \pi_1(x_1), \pi_0(x_0))$, where we notate the 4×4 array by its four columns. Now the following properties hold.

- For all i: $\pi_i(abab)^T = (cdcd)^T$, where $a \ldots d$ are byte values.
- If $\pi_i(abcd)^T = (efgh)^T$ then $\pi_{(i+2) \bmod 4}(abcd)^T = (ghef)^T$ for all i.

With this properties we can deduce the following.
For $ghghefefcdcdabab \in V_0$ it holds that

$$\begin{pmatrix} a & b & a & b \\ c & d & c & d \\ e & f & e & f \\ g & h & g & h \end{pmatrix} \xrightarrow{\pi} \begin{pmatrix} i & m & k & o \\ j & n & l & p \\ k & o & i & m \\ l & p & j & n \end{pmatrix} \in V_1.$$

For $ghefcdabefghabcd \in V_1$ it holds that

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ c & d & a & b \\ g & h & e & f \end{pmatrix} \xrightarrow{\pi} \begin{pmatrix} i & m & i & m \\ j & n & j & n \\ k & o & k & o \\ l & p & l & p \end{pmatrix} \in V_0.$$

For $efghabcdefghabcd \in V_2$ it holds that

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ a & b & c & d \\ e & f & g & h \end{pmatrix} \xrightarrow{\pi} \begin{pmatrix} i & k & m & o \\ j & l & n & p \\ i & k & m & o \\ j & l & n & p \end{pmatrix} \in V_2.$$