Improved Beer-Recovery Attack against APE

Brice Minaud

brice.minaud@gmail.com

Abstract. At the rump session of Eurocrypt 2009, Aumasson and Khovratovich introduced beer-recovery attacks against Keccak [4]. In this note, we improve on their results by attempting to recover a higher-quality beer (according to [11]) through attacking PRIMATEs [2]. Our target cipher is APE [3], one of the PRIMATE designs entered in the CAESAR authenticated cipher competition [1]. We mount a cube attack to recover the key of the APE-80 (resp. APE-120) cipher, with the core permutation round-reduced to 8 instead of 12 rounds, using 2^{33} chosen plaintexts and time complexity 2^{61} (resp. 2^{71}). As a side note, we also point out that the full 12-round permutation can be distinguished from a perfectly random permutation using a zero-sum distinguisher with a zero-sum partition of size 2^{130}; however this distinguisher does not yield an actual attack.

Key words: APE, PRIMATEs, Beer-Recovery Attack

Introduction

One of the lesser known outcomes of the SHA-3 hash function competition is that tempting cryptanalysts to study a given algorithm using beers and chocolates appears to work. This strategy was successfully applied by the Keccak team over the course of four consecutive contests, inspired by Daniel Bernstein's earlier monthly CubeHash cash prizes [5]. Following these fruitful experiments, the design team behind the PRIMATEs entry [2] in the CAESAR authenticated cipher competition [1] is offering beers and chocolates as prizes for the best cryptanalysis of PRIMATEs (it is relevant to note that most of the PRIMATE team, like the Keccak team, is based in Belgium).

Previous Work. During the Keccak cryptanalysis competitions, Aumasson and Khovratovich introduced beer-recovery attacks at the rump session of Eurocrypt 2009 [4]. However we target a different cipher and a different beer. Furthermore their beer is probably long gone by now, so new attacks are quite relevant.

Our contribution. The PRIMATE designs are permutation-based authenticated ciphers relying on a core permutation named PRIMATE. This permutation comes in two main variants, PRIMATE-80 and PRIMATE-120, with different state sizes to accomodate security parameters of 80 bits and 120 bits respectively. In all cases the permutation follows an AES-like design with a 5-bit S-box.

Our contribution simply relies on the fact that, although this is not mentioned in the design document, this S-box has algebraic degree only two. If the core

permutation is round-reduced to 8 out of 12 rounds, using this low algebraic degree, we can mount a cube attack againt APE-80 (resp. APE-120) using 2^{33} chosen plaintexts, to recover the full encryption key in time complexity 2^{61} (resp. 2^{71}).

As a side note, we also point out that both full 12-round PRIMATE permutations can be distinguished from random permutations using a zero-sum distinguisher with 2^{130} data. This is somewhat relevant because e.g. HANUMAN follows a hermetic sponge strategy, i.e. its security proof assumes a perfectly random core permutation (as does that of APE). However this is a purely theoretic concern, as this distinguisher does not lead to an actual attack, even disregarding the vast data requirements (for comparison, a zero-sum distinguisher also exists on the full Keccak permutation [8], which also follows the hermetic sponge strategy; and it is not remotely considered a threat).

Structure of the paper

We begin by briefly recalling the relevant aspects of the PRIMATE permutation, as well as the design of APE. In Section 2 we describe our cube attack against round-reduced APE. In Section 3 we point out the existence of a zero-sum distinguisher on the PRIMATE permutation.

1 Brief Description of PRIMATE and APE

	APE-80	APE-120
Security parameter	80	120
Capacity and key size	160	240
Rate	40	40
State size	200	280

APE comes in two variants, APE-80 and APE-120, offering 80 and 120 bits of security respectively. Each variant relies on its own core permutation, PRIMATE-80 and PRIMATE-120. These two permutations differ slightly in order to accomodate 200-bit and 280-bit states respectively.

1.1 PRIMATE-80

PRIMATE-80 is a permutation based on an AES-like design, following the wide trail strategy. The 200-bit inner state is viewed as a 5×8 matrix of 5-bit cells, called elements. The permutation consists in the repeated application of a round function over 12 rounds[1]. The round function is composed of the following steps:

SubElements. A fixed 5-bit S-box is applied to each 5-bit element in the state. The S-box is given in Fig. 1.

[1] In this paper, we only consider the 12-round variants of the permutations; and not the 6-round variants used in the inner steps of GIBBON.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	1	0	19	1a	11	1d	15	1b	14	5	4	17	e	12	2	1c
x	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f
$S(x)$	f	8	6	3	d	7	18	10	1e	9	1f	a	16	c	b	13

Fig. 1. PRIMATE S-box.

ShiftRows. Row 0, 1, 2, 3, 4 is shifted to the left by 0, 1, 2, 4, 7 positions respectively.

MixColumns. A fixed MDS matrix is applied to each column of the state. The exact matrix is irrelevant for our attack.

ConstantAddition. A round-dependent constant is added into the state at position $(1, 1)$. The value of the constant is irrelevant for our attack.

1.2 PRIMATE-120

PRIMATE-120 is identical to PRIMATE-80, except the state is 280-bit long, and viewed as a 7×8 instead of 5×8 matrix of 5-bit elements. The same operations are applied by the round function, except the indexes in ShiftRows, the MDS matrix in MixColumns, and the value of round constants in ConstantAddition are different. These values are irrelevant for our attack.

1.3 Finalization of APE

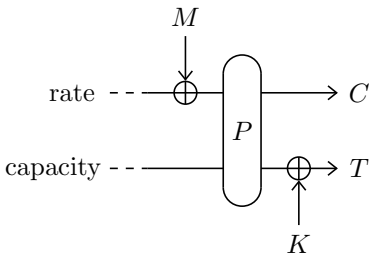

Fig. 2. Finalization of APE.

APE follows a sponge-like design. The inner state is divided into a rate part, corresponding to the first row (hence the rate is $8 \cdot 5 = 40$ bits); the rest is the capacity. Observe that the capacity is twice the security parameter (160 bits for APE-80 and 240 bits for APE-120). Moreover the size of the encryption key K is also twice the security parameter, matching the capacity.

For our purpose, it is enough to know how the finalization step of APE-80 (resp. APE-120) operates, i.e. how the last block of the message is processed,

and the authentication tag is output. Let us denote by M the last message block of 40 bits. The finalization of APE is depicted in Fig. 2, where C is the last ciphertext block and T is the authentication tag.

2 Cube Attack against APE

The PRIMATE permutation reuses the 5-bit FIDES S-box [6]. This is an almost bent permutation, offering optimal linear and differential properties. On the other hand, its algebraic degree is only 2 (although this is not mentioned in the design document). After the fact, this seems to be due to the constructions in [9].

2.1 Cube Attack

Cube attacks were introduced by Dinur and Shamir at EUROCRYPT 2009 [10]. What we propose is a simple form of this attack. Since APE does not require a nonce, we can encrypt a fixed message while changing only the last message block; and thus the last block will be added into a fixed inner state. The simplest way to achieve this is to encrypt a single-block chosen message M with no associated data. Since a single plaintext block is 40-bit long, we can use this message to cover a cube of size up to 40.

Now observe that since the PRIMATE S-box is of algebraic degree 2, after 5 rounds of the permutation, the polynomials expressing the value of each bit of the state in terms of the bits at the input of the permutation are of degree at most $2^5 = 32$. When we perform a cube attack with a cube of size 33 at the input of the permutation, and add up the outputs, we are differentiating these polynomials 33 times, and hence we will get zero.

In short, we can encrypt single-block messages M that take every possible value on their first 33 bits, and are fixed to zero on the 7 remaining bits. If we add up the inner state of the cipher after 5 rounds of the final permutation for each message, we will get zero over the whole state. Actually we can gain one additional round at the cost of a little more data, because the PRIMATE permutation starts with SubElements, i.e. the S-box layer. There are two ways to achieve this.

The first is to cover a cube of size 35 instead of 33, along the first 7 elements of the state. Then the messages M take every possible value on their first 35 bits. Because the message is added into the inner state, after message addition the state will take every possible value on its first 35 bits, i.e. the first 7 elements of the first row. Since S-boxes are permutations, after the S-box layer, the inner state still takes every possible value on its first 35 bits, and some fixed value on the remaining bits. As the rest of the first round is linear, we get the first round for free. Another way to view this is that we are starting the cube attack after the first S-box layer.

Another option is to encrypt 2-block messages instead of 1-block messages, with the first block fixed arbitrarily, and the second block covering a cube of

size 33. The point is that before the second plaintext block is added into the state, the first ciphertext block is output, and it is equal the rate part of the state, where the second plaintext block will be added. This allows the adversary to essentially invert the first S-box layer: she can choose M so that the rate part of the state takes a value of her choice after the first S-box layer, and so she can choose messages M so as to cover a cube of size 33 after the first S-box layer. Note that the first ciphertext block is fixed, so one chosen encryption is enough to know it. In this way we also get the first round for free, using 2^{33} messages of 2 blocks each.

For simplicity, in the remainder, we assume this second strategy is followed, so we use 2^{33} messages of 2 blocks each.

2.2 Key Recovery

Assume the core permutation of APE is reduced from 12 rounds to 8. We begin by considering APE-80; APE-120 will be very similar. In the previous section, we showed how to choose a set of 2^{33} messages such that when they are encrypted, after 6 rounds of the final permutation, the bitwise sum of the inner states over all messages is zero everywhere. Now we show how we can use this to recover the key two rounds later.

To do this, observe that after the last S-box layer of the permutation, the remaining operations are linear. Then the last ciphertext and authentication tag provide access to the whole inner state, xor-ed with the secret key. Because key addition is a linear operation, we can reverse all operations in the last round up to the S-box layer, and this amounts to replacing the key by a linear equivalent representation of it.

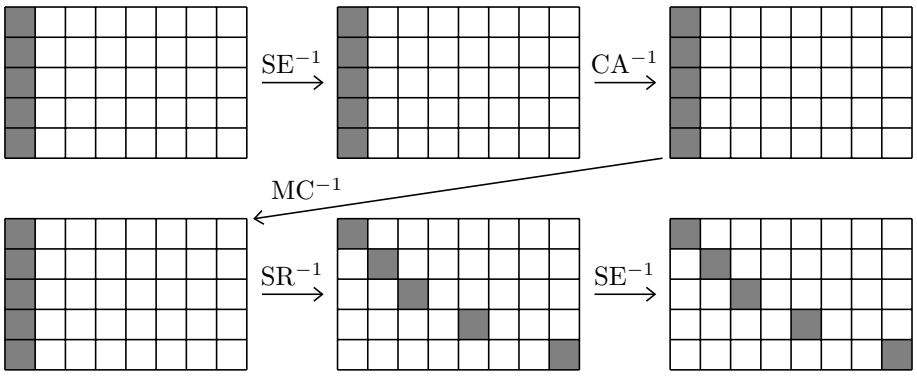

Fig. 3. Key recovery for APE-80, starting after the last S-box layer. Grey cells are known elements.

After we have peeled off the final linear layer in this manner, we guess $5 \times 5 = 25$ bits of (a linear equivalent representation of) the key along one column of

the state. This allows us to know the exact value of the elements in this column, and hence reverse the final S-box layer. Because we know the whole column, we can reverse the last MixColumns operation on this column, then we can follow the 5 elements along the inverse ShiftRows operation, and finally we can reverse the S-box layer on these five known elements (see Fig. 3).

Thus we are able to reverse 2 rounds in total: at the cost of guessing 25 bits of the secret key, we are able compute 25 bits of the inner state 2 rounds before the final state. This allows us to check, for each of these 25 bits, that the sum of this bit computed for each message adds up to zero. This requires 2^{25} 2-round backwards permutation computation on each encrypted message, hence time complexity is $2^{33+25} = 2^{58}$ partial backwards permutation computation.

Since we guessed 25 bits, and we are validating our guess by checking that 25 bits are equal to zero, the false positive rate is very low; namely one false positive on average. In the end, we recover 25 key bits using 2^{33} data and 2^{58} time complexity. We can repeat this process 8 times (essentially once along each column) to recover the full 160-bit key[2]. This increases the time complexity to $2^{58+3} = 2^{61}$ overall.

For APE-120, the same strategy applies. The only difference is that we guess $7 \cdot 5 = 35$ key bits and accordingly check the sum over 35 bits instead of 25. We still use 2^{33} data, and recover 35 key bits in time complexity $2^{33+35} = 2^{68}$. We repeat this process 8 times to recover the full 240-bit key. Thus the overall time complexity is $2^{68+3} = 2^{71}$.

3 Zero-Sum Distinguisher on PRIMATE permutations

The zero-sum distinguisher is built in exactly the same way for PRIMATE-80 and PRIMATE-120. Build 2^{130} inner states by taking every possible value along 26 5-bit elements, and fixing the rest of the state to some arbitrary value. In short, these states will be our inner states after 4 permutation rounds.

Thus, from these initial states, compute 4 permutation rounds backwards to obtain the starting states. Then start over from the initial states, and compute 8 rounds forward to obtain our end states. Observe that the end states are the image of the initial states through the full 12-round permutation (as long as we chose round constants appropriately). Now we claim that the starting states sum up to zero; and so do the end states.

Indeed, the PRIMATE S-box has degree 2 in the forward direction, and degree 3 in the backwards direction. After 4 backwards rounds, the maximal algebraic degree is $3^4 = 81$; since we add up over a cube of size 130 (i.e. order 130 derivative), the states will add up to zero.

Now in the forward direction, observe that the first S-box layer has essentially no effect. Indeed, we chose our 130 bits of the cube so that they saturate $130/5 = 26$ different elements; hence after the S-box layer, these 26 elements still take every possible value (while the rest have some fixed value). As a result, the first

[2] Some information we are guessing is redundant due to linear dependencies. Incidentally this is enough to discard false positives.

S-box layer is for free; as is the first round, since the rest of the round is linear. Then we have 7 rounds of forward permutation, which increases the algebraic degree to at most $2^7 = 128$ (computing the degree starting after the first S-box layer). Since our cube is of size 130, the states add up to zero.

Note that even if the number of rounds was increased by a moderate amount, a zero-sum distinguisher would still exist due to the results in [8]; however, the reason why would be much more elaborate. Again, we stress that while our zero-sum distinguisher shows that the core permutation of APE and HANUMAN is not perfectly random (and hence the hermetic sponge strategy is not strictly followed), this type of distinguisher is of purely theoretical interest; it is generally considered to have no effect on the actual security of the cipher, at least with the type of data complexity required for the PRIMATE permutation.

Open problems

The Keccak team is no longer offering beer to attack their hash function; and the PRIMATE drunken monkey competition is drawing to an end. However, challenging open problems still remain in this area, as the PRINCE team is also offering chocolates and beer for the cryptanalysis of PRINCE [7]. A related open problem is whether Danish beer and chocolates are as good as Belgian ones[3].

References

1. CAESAR– Competition for Authenticated Encryption: Security, Applicability, and Robustness. General secretary Daniel J. Bernstein, information available at `http://competitions.cr.yp.to/caesar.html`, 2013.
2. Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Florian Mendel, Bart Mennink, Nicky Mouha, Qingju Wang, and Kan Yasuda. PRIMATEs. Entry in the CAESAR competition [1], official website `http://primates.ae`, 2014.
3. Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. APE: Authenticated permutation-based encryption for lightweight cryptography. To appear in the proceedings of FSE 2014, 2014.
4. Jean-Philippe Aumasson and Dmitry Khovratovich. Beer-recovery attack. Rump session of Eurocrypt 2009, available from `http://eurocrypt2009rump.cr.yp.to`, 2009.
5. Daniel Bernstein. CubeHash competition. `http://cubehash.cr.yp.to/prizes.html`, 2008.
6. Begül Bilgin, Andrey Bogdanov, Miroslav Knežević, Florian Mendel, and Qingju Wang. Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware. In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013*, volume 8086 of *Lecture Notes in Computer Science*, pages 142–158. Springer Berlin Heidelberg, 2013.

[3] DISCLAIMER: WE DO NOT MAKE ANY CLAIM WHATSOEVER IN EITHER DIRECTION.

7. Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçın. The PRINCE challenge. Presented at the FSE 2014 and CRYPTO 2014 rump sessions, 2014.
8. Christina Boura, Anne Canteaut, and Christophe De Cannire. Higher-order differential properties of keccak and luffa. In Antoine Joux, editor, *Fast Software Encryption*, volume 6733 of *Lecture Notes in Computer Science*, pages 252–269. Springer Berlin Heidelberg, 2011.
9. Claude Carlet, Pascale Charpin, and Victor Zinoviev. Codes, bent functions and permutations suitable for des-like cryptosystems. *Designs, Codes and Cryptography*, 15(2):125–156, 1998.
10. Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 278–299. Springer Berlin Heidelberg, 2009.
11. Anonymous Voters. Rating for westvleteren xii. `http://www.ratebeer.com/beer/westvleteren-12-xii/4934/`, 2009.