Ciphertext Forgery on HANUMAN

Damian Vizár

EPFL, Switzerland

Abstract. HANUMAN is a mode of operation of a keyless cryptographic permutation for nonce-based authenticated encryption with associated data, included among the modes bundled in the PRIMATEs candidate in the currently ongoing CAESAR competition. HANUMAN is a sponge-like mode whose design and security argument are inspired by the SpongeWrap construction. We identify a flaw in the domain separation of HANUMAN, and show how to exploit it to efficiently produce ciphertext forgeries.

Keywords: Authenticated encryption, PRIMATEs, ciphertext forgery, CAESAR competition.

1 Introduction

Authenticate encryption (AE) is a symmetric-key cryptographic primitive that ensures confidentiality and authenticity of processed messages simultaneously. The currently ongoing Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR) [2] is the major scientific effort in the field of AE. It is aimed at identifying a portfolio of new AE schemes that will surpass the previous generation of schemes (represented by the GCM mode of operation [4]) in their security guarantees, efficiency, or other features that improve their applicability in various usage scenarios.

One of the second round candidates in the CEASAR competition is the collection of schemes dubbed PRIMATEs [1]. The PRIMATEs candidate defines a new cryptographic permutation design PRIMATE, and proposes four instances separated by the use of different round constants. PRIMATEs also proposes three modes of operation of a keyless permutation for authenticated encryption: APE, HANUMAN and GIBBON.

HANUMAN is a scheme for nonce-based authenticated encryption with associated data (AEAD). It's design is inspired by the SpongeWrap [3] construction [3] and the authors refer to the security analysis of SpongeWrap when arguing about the security of HANUMAN. In this short note, we identify a flaw in the domain separation between the processing of associated data (AD) and the processing of messages in HANUMAN show how to exploit it to produce a ciphertext forgery with a single encryption query and the probability of successful verification equal to 1.
The rest of this note is organized as follows. In Section 2 we briefly recall the nonce-based AEAD and its security, in Section 3 we briefyl describe the mode HANUMAN and in Section 4 we describe the forgery attack.

2 Nonce-Based AE with associated data

A nonce-based AEAD [5] scheme Π is a triple $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ where $\mathcal{K}$ is the secret key space and $\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C} \cup \{\perp\}$ and $\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ are deterministic encryption and decryption algorithms respectively.

The encryption algorithm takes as inputs a secret key $K \in \mathcal{K}$, a nonce $N \in \mathcal{N}$, associated data (AD) $A \in \mathcal{A}$ and a plaintext (PT) $M \in \mathcal{M}$ and produces a ciphertext. The decryption algorithm

takes as inputs a secret key $K \in \mathcal{K}$, a nonce $N \in \mathcal{N}$, associated data (AD) $A \in \mathcal{A}$ and a ciphertext (CT) $C \in \mathcal{C}$ and outputs a plaintext $M \in \mathcal{M}$ or the symbol $\perp$ that indicates an authentication error.

It is required that if $C = \mathcal{E}(K, N, A, M)$ then $M = \mathcal{D}(K, N, A, C)$. It is required that for every message the $M \neq \perp$ and that length of the ciphertext $|\mathcal{E}(K, N, A, M)| = f(|A|, |M|)$ only depends on the length of the message and AD.[1]

Security. The privacy goals of an AEAD scheme Π are captured by indistinguishability of ciphertexts from random strings under chosen plaintext attack. We measure the advantage of advantage of an adversary $\mathscr{A}$ in breaking the privacy of Π as

$$\mathbf{Adv}^{\mathbf{priv}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathcal{E}_K(\cdot,\cdot,\cdot)} \Rightarrow 1] - \Pr[\mathscr{A}^{\$(\cdot,\cdot,\cdot)} \Rightarrow 1].$$

Here $\mathcal{E}_K(\cdot,\cdot,\cdot)$ denotes the encryption algorithm initialized by a random key and $\$(\cdot,\cdot,\cdot)$ denotes a dummy algorithm that returns a random string of $|E_K(N, A, M)|$ bits on query $\$(N, A, M)$. It is required that every query uses a unique nonce.

The authenticity of Π is formalized as the inability of an adversary $\mathscr{A}$ to forge a valid ciphertext. The advantage of $\mathscr{A}$ in breaking the authenticity of Π is measured as

$$\mathbf{Adv}^{\mathbf{auth}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathcal{E}_K(\cdot,\cdot,\cdot)}\text{forges}].$$

We say that $\mathscr{A}$ forges, if it issues a decryption query N, A, C that decrypts to an $M \neq \perp$ under key K and that was not obtained through encryption queries. It is required that every encryption query uses a unique nonce.

3 HANUMAN

HANUMAN is a nonce-based AEAD scheme. It is a mode of operation for a pair of keyless permutations p_1, p_4 with $p_i : \{0,1\}^b \to \{0,1\}^b$ for $i \in \{1, 4\}$, that operates on a b-bit state in an iterative, sponge-like fashion. The state is divided into a "capacity" part of c bits and a "rate" part of r bits, such that $r + c = b$. To process a query (N, A, M), HANUMAN partitions the associated data (line 3 in Algorithm 3) into r bit blocks, except that the last blocks is possibly fractional, i.e. $|A[u]| \leq r$. The message is partitioned in the same fashion (line 10 in Algorithm 3). The processing of an encryption query (N, A, M) passes through three stages:

1. The state is initialized by applying p_1 to the secret key and the nonce N.
2. An AD A is absorbed block-by-block using the permutation p_4. The last block of A is processed using p_1. If $|A[u]| < r$, padding is applied: $A[u]\|10^{r-|A[u]|-1}$. Otherwise, the constant 10^{c-1} is xored to the capacity part of the state.
3. A message M is processed block-by-block using the permutation p_1. An authentication tag is produced.

The algorithmic description of HANUMAN is in Figure 1.

[1] We typically have $|\mathcal{E}(K, N, A, M)| = |M| + \tau$ for some positive constant τ.

Algorithm 3: $\mathcal{E}_K(N, A, M)$

Input: $K \in \mathbf{C}^{\frac{1}{2}}$, $N \in \mathbf{C}^{\frac{1}{2}}$, $A \in \{0,1\}^*$, $M \in \{0,1\}^*$
Output: $C \in \{0,1\}^*$, $T \in \mathbf{C}^{\frac{1}{2}}$

$V \leftarrow p_1(0^r \parallel K \parallel N)$
if $A \neq \varnothing$ **then**
  $A[1]A[2]\cdots A[u] \leftarrow A$
  $A[u] \leftarrow A[u] \parallel 10^*$
  for $i = 1$ **to** $u - 1$ **do**
    $V \leftarrow p_4(A[i] \oplus V_r \parallel V_c)$
  end
  $V \leftarrow p_1(A[u] \oplus V_r \parallel V_c)$
end
$M[1]M[2]\cdots M[w] \leftarrow M$
$\ell \leftarrow |M[w]|$
$M[w] \leftarrow M[w] \parallel 10^*$
for $i = 1$ **to** w **do**
  $C[i] \leftarrow M[i] \oplus V_r$
  $V \leftarrow p_1(C[i] \parallel V_c)$
end
$C \leftarrow C[1]C[2]\cdots C[w-1]\lfloor C[w] \rfloor_\ell$
$T \leftarrow \lfloor V_c \rfloor_{\frac{c}{2}} \oplus K$
return (C, T)

Algorithm 4: $\mathcal{D}_K(N, A, C, T)$

Input: $K \in \mathbf{C}^{\frac{1}{2}}$, $N \in \mathbf{C}^{\frac{1}{2}}$, $A \in \{0,1\}^*$, $C \in \{0,1\}^*$, $T \in \mathbf{C}^{\frac{1}{2}}$
Output: $M \in \{0,1\}^*$ or $\perp$

$V \leftarrow p_1(0^r \parallel K \parallel N)$
if $A \neq \varnothing$ **then**
  $A[1]A[2]\cdots A[u] \leftarrow A$
  $A[u] \leftarrow A[u] \parallel 10^*$
  for $i = 1$ **to** $u - 1$ **do**
    $V \leftarrow p_4(A[i] \oplus V_r \parallel V_c)$
  end
  $V \leftarrow p_1(A[u] \oplus V_r \parallel V_c)$
end
$C[1]C[2]\cdots C[w] \leftarrow C$
$\ell \leftarrow |C[w]|$
for $i = 1$ **to** $w - 1$ **do**
  $M[i] \leftarrow C[i] \oplus V_r$
  $V \leftarrow p_1(C[i] \parallel V_c)$
end
$M[w] \leftarrow \lfloor V_r \rfloor_\ell \oplus C[w]$
$V \leftarrow p_1((M[w] \parallel 10^* \oplus V_r) \parallel V_c)$
$M \leftarrow M[1]M[2]\cdots M[w-1]M[w]$
$T' \leftarrow \lfloor V_c \rfloor_{\frac{c}{2}} \oplus K$
return $T = T' \ ? \ M : \perp$

Fig. 1: The HANUMAN encryption $\mathcal{E}_K(N, A, M)$ and decryption $\mathcal{D}_K(N, A, C, T)$ algorithms for fractional messages. Here $\mathbf{C} = \{0,1\}^c$ and $\mathbf{C}^{\frac{1}{2}} = \{0,1\}^{c/2}$. The Figure comes from [1].

Security. According to the PRIMATEs submission, HANUMAN follows the main design principles of SpongeWrap, and thus the security of HANUMAN follows (more or less directly) from the security analysis of SpongeWrap. In SpongeWrap, the AD and the message are also processed block by block with a split state, however each block is concatenated with a single bit that ensures domain separation before it is "absorbed" into the sponge state. In HANUMAN, the usage of this so-called "frame bit" got replaced by the application of two independent permutations in a certain pattern.

4 Ciphertext Forgery on HANUMAN

The design of HANUMAN mode contains a flaw thanks to which it is possible to create a forgery with a single encryption query. The problems arise from the fact that if the associated data is empty, the 2nd stage of processing is simply omitted. In particular, both the last block of AD and the first block of a message are processed by p_1, and there is no mechanism that would *always* distinguish these two calls. Indeed, if $|A| < r$, no calls to p_4 are made and no constant is xored to the capacity.

Thus, if the AD consists of a single block of $< r$ bits $A = A[1]$, then the two encryption queries $C_1, T_1 \leftarrow \mathcal{E}_K(N, A, M)$ and $C_2, T_2 \leftarrow \mathcal{E}_K(N, \varepsilon, A[1]\|10^*\|M)$[2] will pass through the same sequence of the internal state's values when being processed and will have the same tag. Moreover, the ciphertext C_1 will be a suffix of C_2. This is valid for any values of K, N and M. This allows us to mount the forgery attack described in Figure 2. The forgery is always successful, as A gets padded to form

[2] ε denotes an empty string.

$M'[1] = A\|10^*$ in the decryption query and the decryptions of (N, ε, C', T') and (N, A, C, T) are processed identically.

1: $M' = A\|10^{r-|A|-1}\|M$
2: $C', T' \leftarrow \mathrm{Enc}(N, \varepsilon, M')$
3: $C \leftarrow \mathsf{right}_{|M|}(C')$; $T \leftarrow T'$
4: **return** $N, A, (C, T)$

Fig. 2: **Ciphertext forgery** for HANUMAN mode with associated data A, message M and nonce N with $|A| < r$. Here $\mathsf{right}_m(X)$ returns the m rightmost bits of a string X.

4.1 (Non-)Applicability to SpongeWrap and Other PRIMATEs

The attack we presented is **not** applicable to SpongeWrap or other PRIMATEs. In case of SpongeWrap, the attack will fail because a single permutation is made in the phase of AD processing even if AD is empty. This mitigates the ciphertext truncation attack that works on HANUMAN.

Our attack works neither with APE nor with GIBBON. The former mode follows a different design philosophy. GIBBON is however similar to HANUMAN in the overall design structure, and in particular it is similar in that it uses a specific pattern of calls to several independent permutations to ensure the domain separation. Unlike HANUMAN, GIBBON makes a call to a permutation in the AD processing phase even if AD is empty, and thus mitigates the attack in a similar fashion as SpongeWrap.

References

1. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mendel, F., Mennink, B., Mouha, N., Wang, Q., Yasuda, K.: Primates. https://competitions.cr.yp.to/round2/primatesv102.pdf
2. Bernstein, D.J.: Cryptographic competitions: CAESAR. http://competitions.cr.yp.to
3. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Duplexing the sponge: Single-pass authenticated encryption and other applications. In: Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2012)
4. McGrew, D.A., Viega, J.: The security and performance of the galois/counter mode (GCM) of operation. In: INDOCRYPT 2004. pp. 343–355 (2004)
5. Rogaway, P.: Authenticated-Encryption with Associated-Data. In: ACM CCS 2002. pp. 98–107 (2002)