Security Analysis of DRBG Using HMAC in NIST SP 800-90

Security Analysis of DRBG Using HMAC in NIST SP 800-90

Shoichi Hirose

Graduate School of Engineering, University of Fukui
hrs_shch@u-fukui.ac.jp

Abstract. HMAC_DRBG is a deterministic random bit generator using HMAC specified in NIST SP 800-90. The document claims that HMAC_DRBG is a pseudorandom bit generator if HMAC is a pseudorandom function. However, no proof is given in the document. This article provides a security analysis of HMAC_DRBG and confirms the claim.

Key words: NIST SP 800-90, pseudorandom bit generator, HMAC, pseudorandom function

1 Introduction

Background. Random bits are indispensable to every cryptographic application. However, it is not easy to prepare sufficient amount of truly random bits in general. Thus, most applications use a cryptographic mechanism that is called a pseudorandom bit generator (PRBG). It stretches a short sequence of random bits to a long sequence of bits that are indistinguishable from truly random bits in practice.

HMAC_DRBG is a deterministic random bit generator (DRBG) specified in NIST SP 800-90 [3]. It is claimed in NIST SP 800-90 that HMAC_DRBG is a PRBG if HMAC is a pseudorandom function (PRF). However, no proof is made public as far as the authors know.

Contribution. This article presents a security analysis of HMAC_DRBG. The result supports the claim in NIST SP 800-90 mentioned above. This article does not provide new techniques and just uses well-known ones in the analysis. In spite of this fact, the contribution of this paper is still important since HMAC_DRBG is expected to be widely used in practice.

HMAC_DRBG consists of three algorithms. They are instantiate, reseed and generate algorithms. The instantiate/reseed algorithm is used to produce/refresh a secret key. The generate algorithm produces a binary sequence from a secret key given by the instantiate or reseed algorithms. This article gives a proof for the pseudorandomness of HMAC_DRBG on the assumption that the instantiate and reseed algorithms are ideal. Namely, a secret key given to the generate algorithm is selected uniformly at random by each of them.

Related Work. NIST SP 800-90 specifies four DRBG mechanisms: Hash_DRBG, HMAC_DRBG, CTR_DRBG and Dual_EC_DRBG. Hash_DRBG and HMAC_DRBG are based on hash functions. CTR_DRBG uses a block cipher in the counter mode. Dual_EC_DRBG is based on the elliptic curve discrete logarithm problem. A security analysis of these DRBGs was presented in [9]. However, the discussion was quite informal. A security analysis of CTR_DRBG was also presented in [7]. Brown and Gjøsteen [6] provided a detailed security analysis of Dual_EC_DRBG.

There also exist some other approved DRBGs in ANSI X.9.31 [2], ANSI X.9.62-1998 [1] and FIPS PUB 186-2 [11]. The security of these algorithms was discussed in [8] and [10].

HMAC was proposed by Bellare, Canetti and Krawczyk [5]. It was proved to be a PRF on the assumption that the compression function of the underlying iterated hash function is a PRF with two keying strategies[4]. Actually, HMAC is used as a PRF in many cryptographic schemes.

Organization. This article is organized as follows. Definitions of a pseudorandom bit generator and a pseudorandom function are given in Sect. 2. A description of HMAC_DRBG is presented in Sect. 3. Results on security analysis of HMAC_DRBG are shown in Sect. 4. Concluding remarks are given in Sect. 5.

2 Preliminaries

Let $a \xleftarrow{\$} A$ represent that an element a is selected uniformly at random from a set A.

2.1 A Pseudorandom Bit Generator

Let G be a function such that $G : \{0,1\}^n \to \{0,1\}^l$. Let $\mathcal{D}$ be a probabilistic algorithm which outputs 0 or 1 for a given input in $\{0,1\}^l$. The goal of $\mathcal{D}$ is to tell whether a given input is $G(k)$ for k selected uniformly at random or it is selected uniformly at random. The advantage of $\mathcal{D}$ against G is defined as follows:

$$\mathrm{Adv}_G^{\mathrm{prbg}}(\mathcal{D}) = \left|\Pr[\mathcal{D}(G(k)) = 1 \mid k \xleftarrow{\$} \{0,1\}^n] - \Pr[\mathcal{D}(s) = 1 \mid s \xleftarrow{\$} \{0,1\}^l]\right| \ ,$$

where the probabilities are taken over the coin tosses by $\mathcal{D}$ and the uniform distributions on $\{0,1\}^n$ or $\{0,1\}^l$. G is called a pseudorandom bit generator (PRBG) if $l > n$ and $\mathrm{Adv}_G^{\mathrm{prbg}}(\mathcal{D})$ is negligible for any efficient $\mathcal{D}$.

2.2 A Pseudorandom Function

Let H be a keyed function such that $H : K \times D \to R$, where K is a key space, D is a domain, and R is a range. $H(k, \cdot)$ is denoted by $H_k(\cdot)$. Let $\mathcal{A}$ be a probabilistic algorithm which has oracle access to a function from D to R. The goal of $\mathcal{A}$ is to tell whether the oracle is H_k for k selected uniformly at random or it is a

function selected uniformly at random. $\mathcal{A}$ first asks elements in D and obtains the corresponding elements in R with respect to the function, and then outputs 0 or 1. $\mathcal{A}$ makes the queries adaptively: $\mathcal{A}$ makes a new query after receiving an answer to the current query.

Let F be the set of all functions from D to R. The advantage of $\mathcal{A}$ for H is defined as follows:

$$\mathrm{Adv}_H^{\mathrm{prf}}(\mathcal{A}) = \left|\Pr[\mathcal{A}^{H_k} = 1 \mid k \xleftarrow{\$} K] - \Pr[\mathcal{A}^{\rho} = 1 \mid \rho \xleftarrow{\$} F]\right| \ ,$$

where the probabilities are taken over the coin tosses by $\mathcal{A}$ and the uniform distributions on K or F. H is called a pseudorandom function (PRF) if $\mathrm{Adv}_H^{\mathrm{prf}}(\mathcal{A})$ is negligible for any efficient $\mathcal{A}$.

3 HMAC_DRBG

HMAC_DRBG is a DRBG using HMAC in NIST SP 800-90 [3]. It consists of three algorithms: an instantiate algorithm, a reseed algorithm and a generate algorithm. The instantiate algorithm is used to produce a secret key. The reseed algorithm is used to refresh it. These algorithms are out of the scope of the article. They also use HMAC to produce the secret keys. However, the security of the outputs is not based on the secrecy and randomness of the keys given to HMAC but the data given to HMAC via message input. From this viewpoint, we may say that they abuse HMAC.

We only assume that the keys produced by the instantiate and reseed algorithms are ideal. Namely, we assume that a secret key given to the generate algorithm is selected uniformly at random.

The generate algorithm produces a binary sequence for a given secret key. A description of this algorithm is given in the following part. The instantiate and reseed algorithms are given in Appendix A for reference.

Notation. HMAC is simply denoted by H. Let n denote the output length of H. Let null denote an empty sequence. Concatenation of binary sequences x and y is denoted by $x\|y$. The symbol $\|$ is sometimes omitted.

3.1 Internal State

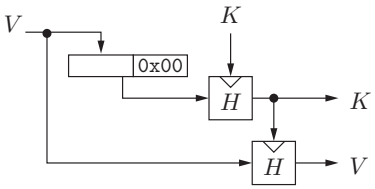
The internal state of HMAC_DRBG includes $K \in \{0,1\}^n$, $V \in \{0,1\}^n$, and a reseed counter $d \geq 1$. K and V are assumed to be secret. The reseed counter d is an integer variable indicating the number of requests for pseudorandom bits since instantiation or reseeding.

3.2 The Function Update

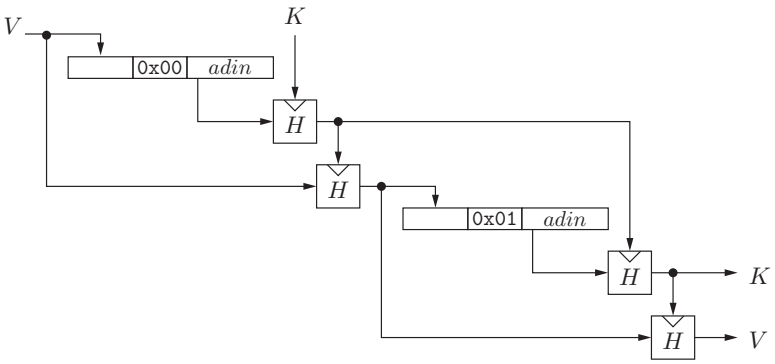
The function Update is used in the generate algorithm to produce a secret key (K, V) for the next invocation of the generate algorithm. It is described as follows.

Update(K, V, $adin$):
1. $K = H(K, V \| \texttt{0x00} \| adin)$
2. $V = H(K, V)$
3. If $adin = \mathsf{null}$, then return (K, V)
4. $K = H(K, V \| \texttt{0x01} \| adin)$
5. $V = H(K, V)$
6. Return (K, V)

$adin$ is an optional additional input. Update is shown in Fig. 1.



(a) If $adin$ is an empty sequence



(b) If $adin$ is not an empty sequence

Fig. 1. The function Update

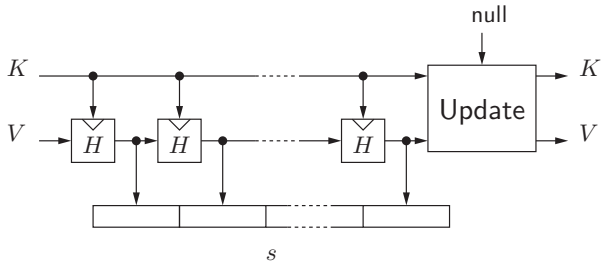
3.3 The Algorithm Generate

The generate algorithm Generate produces a binary sequence s for given secret (K, V) and an optional additional input $adin$. It is described as follows.

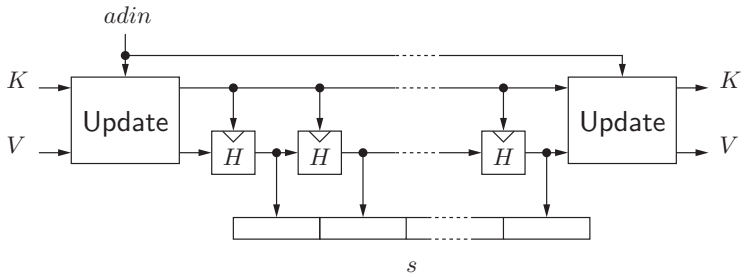
Generate(K, V, $adin$):

1. If $d > w$, then return an indication that a reseed is required.
2. $(K, V) = \mathsf{Update}(K, V, adin)$ if $adin \neq \mathsf{null}$
3. $tmp = \mathsf{null}$
4. While $|tmp| < \ell$ do:
   (a) $V = H(K, V)$
   (b) $tmp = tmp \| V$
5. $s =$ the leftmost ℓ bits of tmp
6. $(K, V) = \mathsf{Update}(K, V, adin)$
7. $d = d + 1$
8. Return s and (K, V)

The maximum sizes of parameters are given in Table 1. w is called a reseed interval. It represents the total number of requests for pseudorandom bits between reseeding. If $adin$ is not supported, then the step 6 is executed with $adin = \mathsf{null}$. Generate is given in Fig. 2.



(a) If $adin$ is not given or not supported



(b) If $adin$ is given

Fig. 2. The algorithm Generate. The update of the reseed counter d is omitted.

Table 1. The maximum sizes of parameters

parameter	maximum size
$\|adin\|$	2^{35} bits
ℓ	2^{19} bits
w	2^{48}

4 Security Analysis

For simplicity, we assume that ℓ is a multiple of n. Thus, the output length of Generate is $\ell+2n$. We analyze the security of a generator $G : \{0,1\}^{2n} \to \{0,1\}^{w\ell}$ defined as follows:

$G(K, V)$:
1. $(K_0, V_0) = (K, V)$
2. $s = \mathsf{null}$
3. for $i = 1$ to w do
   (a) $(s^i, K_i, V_i) = \mathsf{Generate}(K_{i-1}, V_{i-1}, adin_{i-1})$
   (b) $s = s \| s^i$
   (c) Return s

We make $adin_i$ implicit in the notation of $G(K, V)$, because the analysis given in the remaining parts does not depend on the value of $adin_i$. It depends only on whether $adin_i = \mathsf{null}$ or not.

4.1 If $adin = \mathsf{null}$

First, notice that we cannot prove the pseudorandomness of G directly from that of Generate. Generate is not a PRBG if $adin = \mathsf{null}$. Let $q = \ell/n$ and $\mathsf{Generate}(K, V) = s_1 \| s_2 \| \cdots \| s_q \| K' \| V'$, where $s_j \in \{0,1\}^n$ for $1 \le j \le q$. Then, $V' = H_{K'}(s_q)$. Thus, it is easy to distinguish $s_1 \| s_2 \| \cdots \| s_q \| K' \| V'$ from a truly random sequence of length $\ell + 2n$.

We introduce two generators G_{01} and G_{02}. $G_{01} : \{0,1\}^{2n} \to \{0,1\}^{\ell+2n}$ is described as follows:

$G_{01}(K, V)$:
1. $s = V$
2. $tmp = \mathsf{null}$
3. While $|tmp| < \ell$ do:
   (a) $V = H(K, V)$
   (b) $tmp = tmp \| V$
4. $s = s \| tmp$
5. $K = H(K, V \| \mathtt{0x00})$
6. $s = s \| K$
7. Return s

$G_{02} : \{0,1\}^{2n} \to \{0,1\}^{\ell+3n}$ is described as follows:

$G_{02}(K, V)$:
1. $s = V$
2. $V = H(K, V)$
3. $s = s \| G_{01}(K, V)$
4. Return s

The only difference between G_{01} and G_{02} is that G_{02} calls HMAC more than G_{01} by one time.

Let $G_0 : \{0,1\}^{2n} \to \{0,1\}^{w(\ell+n)+n}$ be a generator which, for a given (K, V), produces a sequence

$$\begin{cases} s_0^1 s_1^1 \cdots s_{q+1}^1 & \text{if } w = 1 \\ s_0^1 s_1^1 \cdots s_{q-1}^1 \| s_{-1}^2 s_0^2 \cdots s_{q-1}^2 \| \cdots \| s_{-1}^{w-1} s_0^{w-1} \cdots s_{q-1}^{w-1} \| s_{-1}^w s_0^w \cdots s_{q+1}^w & \text{if } w \geq 2 \end{cases},$$

where

1. $s_j^i \in \{0,1\}^n$,
2. $s_0^1 s_1^1 \cdots s_{q+1}^1 = G_{01}(K, V)$, and
3. $s_{-1}^i s_0^i s_1^i \cdots s_{q+1}^i = G_{02}(s_{q+1}^{i-1}, s_q^{i-1})$ for $2 \leq i \leq w$.

A diagram of G_0 is given in Fig. 3. Notice that $G(K, V)$ is a part of $G_0(K, V)$. It is obvious if $w = 1$. For $w \geq 2$,

$$\begin{aligned} & G(K, V) \\ &= s_1^1 s_2^1 \cdots s_{q-1}^1 \| s_{-1}^2 s_1^2 s_2^2 \cdots s_{q-1}^2 \| \cdots \| s_{-1}^{w-1} s_1^{w-1} s_2^{w-1} \cdots s_{q-1}^{w-1} \| s_{-1}^w s_1^w s_2^w \cdots s_q^w \\ &= s_1^1 s_2^1 \cdots s_q^1 \| s_1^2 s_2^2 \cdots s_q^2 \| \cdots \| s_1^{w-1} s_2^{w-1} \cdots s_q^{w-1} \| s_1^w s_2^w \cdots s_q^w \end{aligned},$$

where $s_{-1}^{i+1} = s_q^i$ for $1 \leq i \leq w-1$. Thus, G is a PRBG if G_0 is a PRBG. We will discuss the security of G_0 in the remaining part.

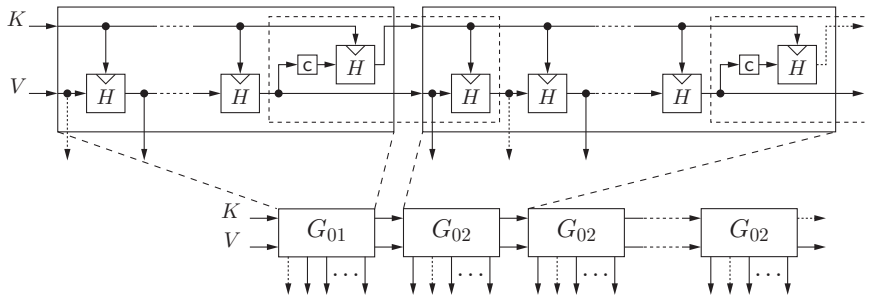


Fig. 3. A diagram of the generator G_0. c represents the concatenation with 0x00. The Update functions are surrounded by dashed rectangles.

We first show that both G_{01} and G_{02} are PRBGs if HMAC is a PRF. For an algorithm A, let t_A be the running time of A.

Lemma 1. *Let $\mathcal{D}$ be a distinguisher for G_{01} which runs in $t_{\mathcal{D}}$. Then, there exists an adversary $\mathcal{A}$ for HMAC such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{01}}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) + \frac{q(q-1)}{2^{n+1}} \ .$$

$\mathcal{A}$ runs in $t_{\mathcal{D}} + O(\ell)$ and asks at most $q+1$ queries.

Proof. Let $F_{*,n}$ be the set of all functions from $\{0,1\}^*$ to $\{0,1\}^n$. Let $\hat{G}_{01}(\rho,\cdot)$ be a generator obtained from G_{01} by replacing HMAC with a function $\rho \in F_{*,n}$. Let

$$\begin{aligned}
P_0 &= \Pr[\mathcal{D}(s) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{01}(K,V)] \ , \\
P_1 &= \Pr[\mathcal{D}(s) = 1 \,|\, \rho \xleftarrow{\$} F_{*,n} \wedge V \xleftarrow{\$} \{0,1\}^{n} \wedge s \leftarrow \hat{G}_{01}(\rho,V)] \ , \\
P_2 &= \Pr[\mathcal{D}(s) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{\ell+2n}] \ .
\end{aligned}$$

Then,

$$\mathrm{Adv}^{\mathrm{prbg}}_{G}(\mathcal{D}) = |P_0 - P_2| \leq |P_0 - P_1| + |P_1 - P_2| \ .$$

Let $\hat{G}_{01}(\rho,V) = \hat{s}_0\hat{s}_1\cdots\hat{s}_{q+1}$, where $\hat{s}_j \in \{0,1\}^n$ for $0 \leq j \leq q+1$. If $\rho \xleftarrow{\$} F_{*,n}$, then $\hat{G}_{01}(\rho,V)$ and a random sequence of length $\ell + 2\,n$ is completely indistinguishable as far as $\hat{s}_{j_1} \neq \hat{s}_{j_2}$ for every j_1 and j_2 such that $0 \leq j_1 < j_2 \leq q-1$. Notice that $\hat{s}_j \neq \hat{s}_q \| \mathtt{0x00}$ for every $0 \leq j \leq q-1$. Thus,

$$|P_1 - P_2| \leq \frac{q(q-1)}{2^{n+1}} \ .$$

On the other hand, for $|P_0 - P_1|$, it is easy to see that we can construct an adversary $\mathcal{A}$ for HMAC, using $\mathcal{D}$ as a subroutine, such that

$$\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) \geq |P_0 - P_1| \ ,$$

where $\mathcal{A}$ runs in $t_{\mathcal{D}} + O(\ell)$ and asks at most $q+1$ queries. □

Lemma 2. *Let $\mathcal{D}$ be a distinguisher for G_{02} which runs in $t_{\mathcal{D}}$. Then, there exists an adversary $\mathcal{A}$ such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) + \frac{q(q+1)}{2^{n+1}} \ .$$

$\mathcal{A}$ runs in $t_{\mathcal{D}} + O(\ell)$ and asks at most $q+2$ queries.

Proof. The proof is similar to that of Lemma 1. □

For $w \geq 2$, let $G_{02}^{(w-1)}$ be a generator which calls G_{02} $(w-1)$ times successively. Namely,

$$G_{02}^{(w-1)}(s^1_{q+1}, s^1_q) = s^2_{-1}s^2_0\cdots s^2_{q-1} \| \cdots \| s^{w-1}_{-1}s^{w-1}_0\cdots s^{w-1}_{q-1} \| s^{w}_{-1}s^{w}_0\cdots s^{w}_{q+1} \ .$$

Lemma 3. *Let $\mathcal{D}$ be a distinguisher for $G_{02}^{(w-1)}$ which runs in $t_{\mathcal{D}}$. Then, there exists a distinguisher $\mathcal{D}'$ of G_{02} such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}^{(w-1)}}(\mathcal{D}) \leq (w-1)\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}}(\mathcal{D}') \ .$$

$\mathcal{D}'$ runs in $t_{\mathcal{D}} + (w-2)\, t_{G_{02}} + O(w\ell)$.

Proof. For a given input $s \in \{0,1\}^{\ell+3n}$, the distinguisher $\mathcal{D}'$ behaves as follows:

1. Select $2 \leq r \leq w$ uniformly at random.
2. If $r \geq 3$, then select $s_{-1}^2 s_0^2 \cdots s_{q-1}^2, \ldots, s_{-1}^{r-1} s_0^{r-1} \cdots s_{q-1}^{r-1}$ uniformly at random.
3. Let $s_{-1}^r s_0^r \cdots s_{q+1}^r = s$.
4. If $r < w$, $s_{-1}^i s_0^i \cdots s_{q+1}^i = G_{02}(s_{q+1}^{i-1}, s_q^{i-1})$ for $r+1 \leq i \leq w$.
5. Call $\mathcal{D}$ with
   $\boldsymbol{s} = s_{-1}^2 s_0^2 \cdots s_{q-1}^2 \| \cdots \| s_{-1}^{w-1} s_0^{w-1} \cdots s_{q-1}^{w-1} \| s_{-1}^w s_0^w \cdots s_{q+1}^w$.
6. Output $\mathcal{D}$'s output.

Then,

$$\begin{aligned}
&\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}}(\mathcal{D}') \\
&= \left| \Pr[\mathcal{D}'(G_{02}(K,V)) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n}] - \Pr[\mathcal{D}'(s) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{\ell+3n}] \right| \\
&= \Big| \Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{02}(K,V)] \\
&\qquad - \Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{\ell+3n}] \Big| \ .
\end{aligned}$$

$$\begin{aligned}
&\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{02}(K,V)] \\
&= \sum_{u=2}^{w} \Pr[r = u \wedge \mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{02}(K,V)] \\
&= \sum_{u=2}^{w} \frac{\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{02}(K,V) \wedge r = u]}{w-1} \\
&= \frac{\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge \boldsymbol{s} \leftarrow G_{02}^{(w-1)}(K,V)]}{w-1} + \\
&\qquad \sum_{u=3}^{w} \frac{\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_{02}(K,V) \wedge r = u]}{w-1} \ .
\end{aligned}$$

$$\begin{aligned}
\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{\ell+3n}] = &\sum_{u=2}^{w-1} \frac{\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{\ell+3n} \wedge r = u]}{w-1} \\
&+ \frac{\Pr[\mathcal{D}(\boldsymbol{s}) = 1 \,|\, \boldsymbol{s} \xleftarrow{\$} \{0,1\}^{(w-1)(\ell+n)+2n}]}{w-1} \ .
\end{aligned}$$

There may exist a better distinguisher for G_{02} than $\mathcal{D}'$ with the same running time. Thus, we have

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}}(\mathcal{D}') \geq \frac{1}{w-1}\mathrm{Adv}^{\mathrm{prbg}}_{G_{02}^{(w-1)}}(\mathcal{D}) \ .$$

The running time of $\mathcal{D}'$ is at most $t_{\mathcal{D}} + (w-2)\, t_{G_{02}} + O(w\ell)$. □

Lemma 4. *Let $\mathcal{D}$ be a distinguisher for G_0 which runs in $t_{\mathcal{D}}$. Then, there exist distinguishers $\mathcal{D}'$ for G_{01} and $\mathcal{D}''$ for $G_{02}^{(w-1)}$ such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_0}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prbg}}_{G_{01}}(\mathcal{D}') + \mathrm{Adv}^{\mathrm{prbg}}_{G_{02}^{(w-1)}}(\mathcal{D}'') \ .$$

$\mathcal{D}'$ runs in $t_{\mathcal{D}} + t_{G_{02}^{(w-1)}} + O(w\ell)$, and $\mathcal{D}''$ runs in $t_{\mathcal{D}} + O(w\ell)$.

Proof. Let

$$P_0 = \Pr[\mathcal{D}(s) = 1 \,|\, (K,V) \xleftarrow{\$} \{0,1\}^{2n} \wedge s \leftarrow G_0(K,V)] \ ,$$
$$P_1 = \Pr[\mathcal{D}(s) = 1 \,|\, s_0^1 \cdots s_{q+1}^1 \xleftarrow{\$} \{0,1\}^{\ell+2n} \wedge s \leftarrow s_0^1 \cdots s_{q-1}^1 \| G_{02}^{(w-1)}(s_{q+1}^1, s_q^1)] \ ,$$
$$P_2 = \Pr[\mathcal{D}(s) = 1 \,|\, s \xleftarrow{\$} \{0,1\}^{w(\ell+n)+n}] \ .$$

Then, there exist $\mathcal{D}'$ and $\mathcal{D}''$ such that

$$\begin{aligned}\mathrm{Adv}^{\mathrm{prbg}}_{G_0}(\mathcal{D}) = |P_0 - P_2| &\leq |P_0 - P_1| + |P_1 - P_2| \\ &\leq \mathrm{Adv}^{\mathrm{prbg}}_{G_{01}}(\mathcal{D}') + \mathrm{Adv}^{\mathrm{prbg}}_{G_{02}^{(w-1)}}(\mathcal{D}'') \ .\end{aligned}$$

The running time of $\mathcal{D}'$ is $t_{\mathcal{D}} + t_{G_{02}^{(w-1)}} + O(w\ell)$. The running time of $\mathcal{D}''$ is $t_{\mathcal{D}} + O(w\ell)$. □

The following theorem directly follows from Lemmas 1, 2, 3 and 4. It implies that G_0 is a PRBG if HMAC is a PRF.

Theorem 1. *Let $\mathcal{D}$ be a distinguisher for G_0 which runs in $t_{\mathcal{D}}$. Then, there exists an adversary $\mathcal{A}$ for HMAC such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_0}(\mathcal{D}) \leq w\,\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) + \frac{wq(q+1)}{2^{n+1}} \ .$$

$\mathcal{A}$ runs in $t_{\mathcal{D}} + w(q+2)\, t_{\mathrm{HMAC}} + O(w\ell)$ and asks at most $q+2$ queries, where the length of each query is at most $n+8$.

Remark 1. Suppose that SHA-1 is the underlying hash function of HMAC. Then, $n = 160$. Suppose that $w = 2^{48}$ and $\ell = 2^{11} \times 160$ ($\leq 2^{19}$). Then,

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_0}(\mathcal{D}) \leq 2^{48}\,\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) + \frac{1}{2^{90}} \ ,$$

where $\mathcal{A}$ runs in time $t_{\mathcal{D}} + 2^{60}\, t_{\mathrm{HMAC}} + O(2^{66.3})$ and makes at most 2050 queries. The big-O notation is abused here. $O(2^{66.3})$ is upper bounded by $c \times 2^{66.3}$ for some positive constant c.

Remark 2. Suppose that SHA-256 is the underlying hash function of HMAC. Then, $n = 256$. Suppose that $w = 2^{48}$ and $\ell = 2^{19}$. Then,

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_0}(\mathcal{D}) \leq 2^{48}\,\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) + \frac{1}{2^{186}} \ ,$$

where $\mathcal{A}$ runs in time $t_{\mathcal{D}} + 2^{60}t_{\mathrm{HMAC}} + O(2^{67})$ and makes at most 2050 queries.

4.2 If $adin \neq$ null

If $adin \neq \mathsf{null}$, then the analysis is similar but tedious. We first define several generators.

Let $g_{10} : \{0,1\}^{2n} \to \{0,1\}^{2n}$ be a generator such that

$$g_{10}(K, V) = V \| H(K, V \| \mathtt{0x00} \| adin) \ .$$

Let $g_{11} : \{0,1\}^{2n} \to \{0,1\}^{3n}$ be a generator such that $g_{11}(K, V) = s$, where s is obtained as follows:

1. $V_1 = H(K, V)$
2. $V_2 = H(K, V_1 \| \mathtt{0x01} \| adin)$
3. $s = V \| V_1 \| V_2$

Let $g_{12} : \{0,1\}^{2n} \to \{0,1\}^{3n}$ be a generator such that $g_{12}(K, V) = s$, where s is obtained as follows:

1. $V_1 = H(K, V)$
2. $V_2 = H(K, V_1 \| \mathtt{0x00} \| adin)$
3. $s = V \| V_1 \| V_2$

The generators g_{10}, g_{11} and g_{12} are depicted in Fig. 4.

Let $G_{10} : \{0,1\}^{2n} \to \{0,1\}^{\ell+3n}$ be a generator equivalent to G_{02} defined in the previous subsection.

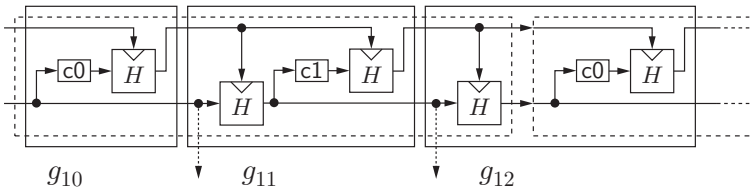


Fig. 4. A diagram of the generators g_{10}, g_{11} and g_{12}. The Update functions are surrounded by dashed rectangles. c0 represents the concatenation with $\mathtt{0x00} \| adin$, and c1 represents the concatenation with $\mathtt{0x01} \| adin$.

Using the generators defined above, we further define two generators G_{11} and G_{12}. $G_{11} : \{0,1\}^{2n} \to \{0,1\}^{\ell+4n}$ is described as follows: $G_{11}(K, V) = s_{-2}s_{-1}\cdots s_{q+1}$, where

1. $V_1K_1 = g_{10}(K, V)$
2. $s_{-2}V_2K_2 = g_{11}(K_1, V_1)$
3. $s_{-1}s_0 \cdots s_q s_{q+1} = G_{10}(K_2, V_2)$

$G_{12} : \{0,1\}^{2n} \to \{0,1\}^{\ell+6n}$ is described as follows: $G_{12}(K, V) = s_{-4}s_{-3} \cdots s_{q+1}$, where

1. $s_{-4}V_1K_1 = g_{11}(K, V)$
2. $s_{-3}V_2K_2 = g_{12}(K_1, V_1)$
3. $s_{-2}V_3K_3 = g_{11}(K_2, V_2)$
4. $s_{-1}s_0 \cdots s_q s_{q+1} = G_{10}(K_3, V_3)$

Now, we are ready to discuss the pseudorandomness of $G(K, V)$. Let $G_1 : \{0,1\}^{2n} \to \{0,1\}^{w(\ell+4n)}$ be a generator which, for a given (K, V), produces a sequence

$$\begin{cases} s^1_{-2}s^1_{-1} \cdots s^1_{q+1} & \text{if } w = 1 \\ s^1_{-2}s^1_{-1} \cdots s^1_{q-1} \| s^2_{-4} \cdots s^2_{q-1} \| \cdots \| s^{w-1}_{-4} \cdots s^{w-1}_{q-1} \| s^w_{-4} \cdots s^w_{q+1} & \text{if } w \geq 2 \end{cases} ,$$

where

1. $s^i_j \in \{0,1\}^n$,
2. $s^1_{-2}s^1_{-1} \cdots s^1_{q+1} = G_{11}(K, V)$, and
3. $s^i_{-4} \cdots s^i_{q+1} = G_{12}(s^{i-1}_{q+1}, s^{i-1}_q)$ for $2 \leq i \leq w$.

Notice that $G(K, V)$ is a part of $G_1(K, V)$. It is easy to see if $w = 1$. For $w \geq 2$,

$$\begin{aligned} &G(K, V) \\ &= s^1_1 s^1_2 \cdots s^1_{q-1} \| s^2_{-4} s^2_1 s^2_2 \cdots s^2_{q-1} \| \cdots \| s^{w-1}_{-4} s^{w-1}_1 s^{w-1}_2 \cdots s^{w-1}_{q-1} \| s^w_{-4} s^w_1 s^w_2 \cdots s^w_{q+1} \\ &= s^1_1 s^1_2 \cdots s^1_q \| s^2_1 s^2_2 \cdots s^2_q \| \cdots \| s^{w-1}_1 s^{w-1}_2 \cdots s^{w-1}_q \| s^w_1 s^w_2 \cdots s^w_q \ , \end{aligned}$$

where $s^{i+1}_{-4} = s^i_q$ for $1 \leq i \leq w-1$. Thus, we discuss the pseudorandomness of G_1 in the remaining part. We only present the results since the proofs are similar.

Lemma 5. *Let $\mathcal{D}$ be a distinguisher for g_{10} which runs in $t_{\mathcal{D}}$. Then, there exists an adversary $\mathcal{A}$ for HMAC such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{g_{10}}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) \ .$$

$\mathcal{A}$ runs in $t_{\mathcal{D}} + O(n)$ and asks 1 query.

Lemma 6. *Let g be g_{11} or g_{12}. Let $\mathcal{D}$ be a distinguisher for g which runs in $t_{\mathcal{D}}$. Then, there exists an adversary $\mathcal{A}$ for HMAC such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{g}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}) \ .$$

$\mathcal{A}$ runs in $t_{\mathcal{D}} + O(n)$ and asks at most 2 queries.

Lemma 7. *Let $\mathcal{D}$ be a distinguisher for G_{11} which runs in $t_{\mathcal{D}}$. Then, there exist $\mathcal{D}_0$, $\mathcal{D}_1$ and $\mathcal{D}'$ such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{11}}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prbg}}_{g_{10}}(\mathcal{D}_0) + \mathrm{Adv}^{\mathrm{prbg}}_{g_{11}}(\mathcal{D}_1) + \mathrm{Adv}^{\mathrm{prbg}}_{G_{10}}(\mathcal{D}') \ .$$

$\mathcal{D}_0$ runs in $t_{\mathcal{D}} + t_{g_{11}} + t_{G_{10}} + O(n)$. $\mathcal{D}_1$ runs in $t_{\mathcal{D}} + t_{G_{10}} + O(n)$. $\mathcal{D}'$ runs in $t_{\mathcal{D}} + O(n)$.

Lemma 8. *Let $\mathcal{D}$ be a distinguisher for G_{12} which runs in $t_{\mathcal{D}}$. Then, there exist $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}'$ such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_{12}}(\mathcal{D}) \leq 2\,\mathrm{Adv}^{\mathrm{prbg}}_{g_{11}}(\mathcal{D}_1) + \mathrm{Adv}^{\mathrm{prbg}}_{g_{12}}(\mathcal{D}_2) + \mathrm{Adv}^{\mathrm{prbg}}_{G_{10}}(\mathcal{D}') \ .$$

$\mathcal{D}_1$ runs in $t_{\mathcal{D}} + t_{g_{11}} + t_{g_{12}} + t_{G_{10}} + O(n)$. $\mathcal{D}_2$ runs in $t_{\mathcal{D}} + t_{g_{11}} + t_{G_{10}} + O(n)$. $\mathcal{D}'$ runs in $t_{\mathcal{D}} + O(n)$.

The following theorem directly follows from Lemmas 5, 6, 7 and 8. It implies that G_1 is a PRBG if HMAC is a pseudorandom function.

Theorem 2. *Let $\mathcal{D}$ be a distinguisher for G_1 which runs in $t_{\mathcal{D}}$. Then, there exist adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$ such that*

$$\mathrm{Adv}^{\mathrm{prbg}}_{G_1}(\mathcal{D}) \leq w\,\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}_1) + 3\,w\,\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{HMAC}}(\mathcal{A}_2) + \frac{wq(q-1)}{2^{n+1}} \ .$$

$\mathcal{A}_1$ runs in $t_{\mathcal{D}} + w(q+8)\,t_{\mathrm{HMAC}} + O(w\ell)$ and asks at most $q+1$ queries, and $\mathcal{A}_2$ runs in $t_{\mathcal{D}} + (w+1)(q+8)\,t_{\mathrm{HMAC}} + O(w\ell)$ and asks at most 2 queries.

5 Conclusion

We have shown that the binary sequence generation algorithm of HMAC_DRBG is a PRBG if HMAC is a PRF. Future work includes analysis of the instantiate and reseed algorithms of HMAC_DRBG.

Acknowledgements

The author would like to thank anonymous reviewers for their valuable comments. This research was supported in part by the National Institute of Information and Communications Technology, Japan.

References

1. American National Standards Institute. Public key cryptography for the financial services industry: The elliptic curve digital signature algorithm (ECDSA). ANSI X9.62-1998 (1998)
2. American National Standards Institute. Digital signatures using reversible public key cryptography for the financial services industry (rDSA). ANSI X9.31-1998 (1998)

3. Barker, E., Kelsey, J.: Recommendation for random number generation using deterministic random bit generators (revised). NIST Special Publication 800-90 (2007)
4. Bellare, M.: New proofs for NMAC and HMAC: Security without collision-resistance. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 602–619. Springer (2006). The full version is "Cryptology ePrint Archive: Report 2006/043" at `http://eprint.iacr.org/`.
5. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Koblitz, N. (ed.) CRYPTO '96. LNCS, vol. 1109, pp. 1–15. Springer (1996)
6. Brown, D.R., Gjøsteen, K.: A security analysis of the NIST SP 800-90 elliptic curve random number generator. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 466–481. Springer (2007)
7. Campagna, M.J.: Security bounds for the NIST codebook-based deterministic random bit generator. Cryptology ePrint Archive: Report 2006/379, `http://eprint.iacr.org/`
8. Desai, A., Hevia, A., Yin, Y.L.: A practice-oriented treatment of pseudorandom number generators. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 368–383. Springer (2002)
9. Kan, W.: Analysis of underlying assumptions in NIST DRBGs. Cryptology ePrint Archive: Report 2007/345, `http://eprint.iacr.org/`
10. Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Cryptanalytic attacks on pseudorandom number generators. In: Vaudenay, S. (ed.) FSE '98. LNCS, vol. 1372, pp. 168–188. Springer (1998)
11. U.S. Department of Commerce/National Institute of Standards and Technology. Digital signature standard (DSS). Federal Information Processing Standards Publication 186-2 (+Change Notice) (2000)

A The Instantiate and Reseed Algorithms of HMAC_DRBG

The internal state of HMAC_DRBG includes $K \in \{0,1\}^n$, $V \in \{0,1\}^n$, and a reseed counter d. *data* is the entropy source. *adin* is an optional additional input.

The Instantiate Algorithm. The instantiate algorithm Instantiate is described as follows:

Instantiate$(data, nonce, adin)$:
1. $seed = data \| nonce \| adin$
2. $K = \texttt{0x0000} \cdots \texttt{00}$
3. $V = \texttt{0x0101} \cdots \texttt{01}$
4. $(K, V) = \mathsf{Update}(seed, K, V)$
5. $d = 1$
6. Return (K, V) and d.

If *adin* is not supported, then the first step of the procedure is replaced by

$$seed = data \| nonce \ .$$

The Reseed Algorithm. The reseed algorithm Reseed is described as follows:

Reseed$(K, V, d, data, adin)$:
1. $seed = data \| adin$
2. $(K, V) =$ Update$(seed, K, V)$
3. $d = 1$
4. Return (K, V) and d.

The input (K, V) to Reseed is given by the latest Generate. If $adin$ is not supported, then the first step of the procedure is replaced by

$$seed = data \ .$$