Security Bounds for the NIST Codebook-based Deterministic Random Bit Generator

Matthew J. Campagna
matthew.campagna@pb.com
Secure Systems
Pitney Bowes Inc.

Abstract

The NIST codebook-based deterministic random bit generators are analyzed in the context of being indistinguishable from random. Upper and lower bounds based on the probability of distinguishing the output are proven. These bounds imply that the security of the designs are bounded by the codebook width, or more precisely on the property that the codebooks act like a random permutation, as opposed to their underlying security parameter or key length. This paper concludes that these designs fail to support security parameters larger than the codebook width.

1 Introduction

Random bit generation is an essential element to almost every cryptographic system. A fundamental aspect of cryptographic systems are not only should the key or secret be hard to determine given visible aspects of the system, but also that it be hard to guess or divine. The security of many protocols hinge on the availability of *random* values. The collection of *random* bits is typically expensive relative to the deterministic aspects of modern computing. Because of this most cryptographic systems use a *deterministic random bit generator* DRBG or *pseudo-random number generator* PRNG seeded from an entropy source.

In turn, cryptographic systems are designed around a chosen security parameter k. This security parameter determines to what extent we expect a DRBG to behave like a true random source. A DRBG for a cryptographic application should be *indistinguishable* from a true random number generator to a computationally bounded adversary, where the bound is related to the security parameter k of the overall system.

This paper constructs lower and upper bounds on the advantage function against the codebook DRBGs constructed in NIST SP800 90 [11]. It uses the previous bounds given by Bellare, Desai, Jokiph and Rogaway [4] to derive an upper bound. It also analyzes one adversarial distinguisher to compute a lower bound on the advantage function.

1.1 Background

The PRNGs defined in ANSI X9.17 [12] and FIPS 186 [1] are being deprecated in favor of new standards that account for more flexibility in terms of security parameters and to perform like a general purpose PRNG instead of an application specific PRNG. These new standards are being developed as *deterministic random bit generators* (DRBGs) both by the ANSI X9 F1 working group in X9.82 [5], and by NIST in FIPS SP800-90 [11]. These standards contain multiple designs based on different cryptographic primitives, such as a secure hash function, an HMAC construction, a codebook construction, and elliptic curve cryptography.

One such construction is the codebook based DRBG, that uses a construction similar to counter-mode encryption.

1.2 Contributions of the paper

The main goal is to analyze the codebook-based DRBG in terms of its indistinguishability from random. In so it applies the work of Bellare et al [4] to generate an upper bound on the advantage function, under an assumption of independence between the events of subsequent calls to the underlying DRBG generate function. The paper then describes a concrete distinguisher algorithm that can be used by an adversary and then computes the advantage function to the adversary for this distinguisher to arrive at a lower bound.

The paper speculates that the design was inspired by the Bellare et al[4] paper. The results of that paper indicate an upper bound on a real-or-random experiment on chosen plain text attacks on the counter-mode of a symmetric encryption algorithm. The paper expresses the advantage function based on the width of the underlying codebook, not the security parameter of the codebook. It is precisely this subtlety that leads to the bounds on an advantage function which is a non-negligible function in terms of the security parameter whenever that parameter exceeds the codebook width.

1.3 Related Work

There have been numerous papers on analyzing the output of codebook ciphers and the counter-mode of codebook ciphers, Bellare et al [4], and McGrew[9], in addition to a great number of other papers.

There has been additional work on the theory of PRNGs or DRBGs, such as the work of Blum Blum Shub [7] on the construction of PRNGs with provable properties and the work of Desai, Hevia and Yin [2] to generate a framework for security of PRNGs. There has been additional work on straightforward attacks on construction by Kelsey, Schneir, Wagner and Hall [6].

The work of Bleichenbacher pointed out a weakness of using application specific DRBGs as a general purpose DRBG. This led to some of the new design requirements on the current set of DRBGs in proposed standards and the changes in FIPS 186-2 [10].

While the above work has examined various DRBGs, constructed security frameworks for them, and have analyzed various modes of operations for block ciphers, this paper applies those to the specific constructions and requirements of the codebook-based DRBG defined in SP800 90.

1.4 Outline of the paper

The paper contains a brief introduction that contains a background and highlights the specific contributions of this paper relative to the previous work done in the area. It contains three more sections. The second section is notations and preliminaries that builds up some of the language from previous related works, and defines the general experiment that will be used to distinguish a DRBG from random.

The third section contains the bulk of the contribution of this paper. It gives a brief description of the NIST SP800-90 codebook-based DRBG, both in a pictorial and algorithmic way. It redefines the experiment from Section 2 in a way more expressive of the specific underlying construction of the DRBG. It then uses this expression and relates it to previous work to construct an upper bound on the adversarial function. The construction of the lower bound is done by explicitly defining a distinguisher algorithm that can be utilized by an adversary to distinguish the output of the DRBG from a random source. This distinguisher algorithm is used to construct a lower bound on the adversary function. These two

bounds, lower and upper, are then used in analyzing the effectiveness of the DRBG design in regards to stated design goals.

The fourth section contains a brief summary and some suggestions for follow on analysis.

2 Notations and Preliminaries

We will adopt notation and system descriptions from both Ueli's [8], and Bellare et al's [4] papers. A random variable will be denoted by a capital letter X and concrete values by the lower case letter, x. For a set $\mathcal{S}$, an $\mathcal{S}$-sequence is a sequence $s = s_1, s_2, \ldots$ (finite or infinite). Let $p_{ncoll}(n,t)$ denote the probability that t independent random variables with uniform distribution from a set of size n does not contain a collision. When $t^2 < n$ we have

$$p_{ncoll}(n,t) = \prod_{i=1}^{t-1}\left(1 - \frac{i}{n}\right) > e^{-\frac{t^2}{2n}}$$

And, when $t << n$ this lower bound also serves as a good approximation. We denote $p_{coll}(n,t)$ as the analogous probability that a collision occurs, $p_{coll}(n,t) < \frac{t^2}{2n}$, and again the upper bound serves as a good approximation when $t << n$.

We will consider systems that upon request produce an output $Y_i \in \mathcal{Y}$. Any such system can be stateless or contain internal memory, be deterministic or probabilistic. Our deterministic systems will be endowed with a state space Σ, and we can think of our deterministic function $f_i : \Sigma \longrightarrow \mathcal{Y} \times \Sigma$, where $(Y_i, S_i) = f(S_{i-1})$.

Our basic goal will be to distinguish between two $\mathcal{Y}$-systems $\mathbf{G}$ and $\mathbf{R}$, by means of a computationally bounded distinguisher algorithm $\mathbf{D}$, making at most q requests for output from the system. In general the distinguisher receives output $Y_1, Y_2, \ldots Y_q$ and then produces a binary decision. For now we will assume the algorithm is possibly unbounded. In general $\mathbf{G}$ is a deterministic random bit generator system, like our DRBGs, and $\mathbf{R}$ will denote a true random source.

A DRBG Model: A DRBG $\mathcal{GE}$ can be expressed as a tuple of algorithms $\mathcal{GE} = (\mathbf{I}, \mathbf{G})$. An initialization algorithm $\mathbf{I} : \mathbb{N} \longrightarrow \Sigma$ which takes as input a security parameter $k \in \mathbb{N}$, and returns a state $S_0 \in \Sigma$, the state space. In our case the state will be a compound state of a key K_i and a digest V_i. The generation function $\mathbf{G}$ takes as input the state $S_i = (V_i, K_i)$, and returns an output Y_i, and a next state S_{i+1}, where $|Y_i| \leq ML$, for some maximum length ML. In general we will restrict our discussion to just the generate function $\mathbf{G}$, and will drop the full model notation $\mathcal{GE}$ and the security parameter. The security parameter will return in the conclusion in analyzing the suggested bounds on an advantage function.

Ideally a polynomially bounded adversary cannot distinguish a deterministic $\mathcal{Y}$-system $\mathbf{G}$, from a random $\mathcal{Y}$-system $\mathbf{R}$ by observing the output. An adversary is allowed to query the system. In the first game, each query is responded to by a deterministic random bit string $Y_i \longleftarrow \mathbf{G}$. In the second game, each query is responded to with $Y_i \longleftarrow \mathbf{R}$. We call such an experiment $\mathbf{Exp}^b_{\mathbf{G},\mathbf{D}}$, and we depict it in Figure 1.

We can then think about the advantage to an adversary running the algorithm $\mathbf{D}$,

$$Adv_{\mathbf{G},\mathbf{D}} = \left|Pr(\mathbf{Exp}^1_{\mathbf{G},\mathbf{D}} = 1) - Pr(\mathbf{Exp}^0_{\mathbf{G},\mathbf{D}} = 1)\right|$$

We define the advantage function of the scheme $\mathbf{G}$ as,

$$Adv_{\mathbf{G}}(t,q) = \max_D\{Adv_{\mathbf{G},\mathbf{D}}\}$$

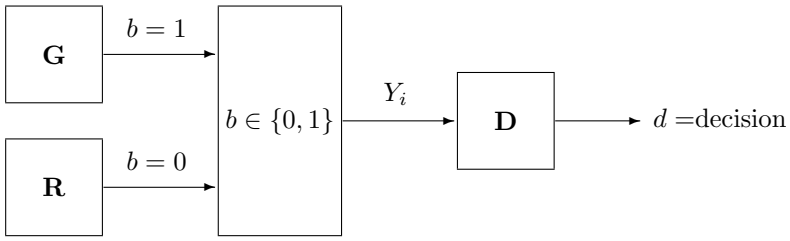

Figure 1: $\mathbf{Exp}^{b}_{\mathbf{G},\mathbf{D}}$

where the maximum is over all distinguisher algorithms $\mathbf{D}$,with storage/time complexity t, making at most q queries to the oracle. Here we accept the convention the t represents both total running time of the experiment, plus the size of storage required for the experiment.

Definition 1. ***(negligible function)*** *A function $f : \mathbb{N} \longrightarrow \mathbb{R}$ is called* negligible *if for all $c \in \mathbb{N}$, there exists an integer n_c such that $f(n) < n^{-c}$ for all $n \geq n_c$.*

We can use this notion of negligible function from [3] to define *indistinguishable from random.*

Definition 2. ***indistinguishable from random.*** *We claim a DRBG $\mathcal{GE} = (\mathbf{I}, \mathbf{G})$ is indistinguishable from random if the $Adv_{\mathbf{G}}(t, q)$ is negligible in respect to the security parameter k used to instantiate the DRBG.*

3 Analysis of NIST SP800 90 CTR DRBG

The counter-mode codebook deterministic random bit generator(CTR_DRBG), described in NIST's SP800 90 contains the two basic functions, Initialize, and Generate. We will concentrate on the generate function. The DRBG is defined around a codebook $\mathbf{E} : \mathcal{V} \times \mathcal{K} \longrightarrow \mathcal{V}$, $y = \mathbf{E}(x, K)$, where $x, y \in \mathcal{V}$ and $|x| = |y| = L$ the codebook width, and $K \in \mathcal{K}$ is the key, $|K| = k$ the security parameter, effectively. The DRBG uses the codebook function $\mathbf{E}$ to operate on a state $(V, K) \in \mathcal{V} \times \mathcal{K}$. We use a simplified version that does not accept input, and since the analysis is not dependent on the reseeding algorithm we will ignore the logic that pertains to it.

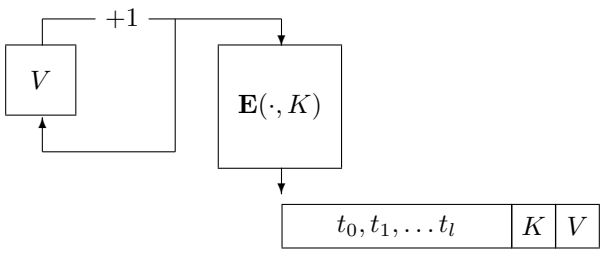

Figure 2: NIST SP800 90 CTR DRBG Generate, DRBG_E

A simplified generate function can be described as follows.

Simplified NIST CTR_DRBG Generate Function

INPUT: A requested `length` less then the maximum allowable length, current state (V, K).

OUTPUT: A `length`-many bits string, and updated state (V, K).

1. `temp`$\longleftarrow$ `NULL`
2. While (len(`temp`) <`length`
   (a) $V \longrightarrow (V + 1) mod\ 2^L$
   (b) `temp` $\longleftarrow$ `temp`$|E(V, K)$
3. $(V, K) \longrightarrow Update(V, K)$
4. `Return` Y =`length`-leftmost bits of `temp`

The Update function in the generate function behaves nearly identical to the generate function. A simplified version is presented here that takes no additional input from the calling function.

Simplified NIST CTR_DRBG Update Function

INPUT: A current state (V, K),

OUTPUT: A new state (V, K).

1. `length` $\longleftarrow |K| + |V|$.
2. (temp) $\longleftarrow$ `NULL`
3. While (len(`temp`) <`length`
   (a) $V \longrightarrow (V + 1) mod\ 2^L$
   (b) `temp` $\longleftarrow$ `temp`$|E(V, K)$
4. `temp` $\longleftarrow$ `length`-leftmost bits of `temp`
5. $K \longleftarrow$ leftmost $|K|$ bits of `temp`
6. $V \longleftarrow$ rightmost $|V|$ bits of `temp`
7. `return` (V, K).

We will assume that the generate function produces an output of Y_i where $|Y_i| \leq ML$ where ML is the maximum output length in bits of a single call to the generate function. Since this analysis is confined to the codebook counter mode DRBG detailed in NIST SP800 90, $ML = 2^l$ where $l = 13$ for 3DES and 19 for AES implementations. We will denote the maximum output in blocks as M.

We provide the following reference table of lengths here to support their use throughout the remainder of the paper.

L	-	The codebook width in bits of **E**
M	-	The maximum number of blocks the generate algorithm **DRBG_E** will return
ML	-	The maximum length in bits that the generate algorithm **DRBG_E** will return
q	-	The number of queries made to the oracle in an experiment.

3.1 An Upper Bound

We will restrict ourselves to NIST codebook-based DRBG models $\mathcal{DRBG\_E} = (\mathbf{DRBG\_IE}, \mathbf{DRBG\_E})$. Where **E** represents the underlying codebook algorithm, **DRBG_IE** the initialization algorithm, and **DRBG_E** the generate algorithm. The maximum output length of **DRBG_E** will generate on a single call is $ML = M \cdot L$ where L is the block bit-length and M is the maximum number of blocks. Let $b \in \{0,1\}$, and **D** be a distinguisher algorithm.

Consider the experiment $\mathbf{Exp}^{b}_{\mathbf{DRBG\_E,D}}$, which can now be expressed as,

- A current state of the **DRBG_E** is given (V, K).
- for $j = 1, ...$ we have $Y_j^0 \xleftarrow{R} \{0,1\}^{ML}$.
- for $j = 1, ...$ we have $Y_j^1 \in \{0,1\}^{ML} \leftarrow \mathbf{DRBG\_E}$.
- $d \leftarrow D(Y_1^b, Y_2^b, \ldots)$.

The advantage of the distinguisher is

$$Adv_{\mathbf{DRBG\_E,D}} = |Pr(Exp^1_{\mathbf{DRBG\_E,D}} = 1) - Pr(Exp^0_{\mathbf{DRBG\_E,D}} = 1)|$$

We can then re-define the advantage of the family for any integers t, q as,

$$Adv_{\mathbf{DRBG\_E}}(t,q) = \max_D \{Adv_{\mathbf{DRBG\_E,D}}(q)\}$$

where the maximum is over all distinguishing algorithms **D** with time complexity t each making at most q queries. At times when t is not germane we will drop it from the notation.

The output of the DRBG **DRBG_E** can be viewed as keystream of the underlying codebook **E** in counter mode. Under this perspective the previous work of Bellare, et al we can apply Proposition 8 and Theorem 13.

Proposition 8 [PRPs are PRFs] For any permutation family P with length L,

$$Adv^{prf}_P(t, M) \leq Adv^{prp}_P(t, M) + M^2 2^{-L-1}$$

Proposition 8 provides bounds by which a pseudorandom permutation, (prp) can be viewed as a pseudorandom function, (prf). Now theorem 13 originally proven for left-or-right experiments of chosen

plaintext attacks, also applies to the real-or-random experiments and chosen plaintext attacks. Here we state a simplified version of the theorem that allows more seamless application to our experiments.

Theorem 13 [Security of CTR using a pseudorandom function] Suppose E is a pseudorandom family with input and output length L, then for any t and M,

$$Adv^{ror-cpa}_{\mathbf{CTR}[\mathbf{E}]}(t, M) \leq 2 \cdot Adv^{prf}_{\mathbf{E}}(t, M)$$

This theorem shows that the using the codebook algorithm **E** in counter mode **CTR**[**E**], the advantage of a real-or-random *ror* experiment using chosen plaintext attack *cpa* is not greater than twice the advantage of distinguishing the function **E** from a pseudorandom function.

And so we can combine Theorem 13 and Proposition 8 to get the following inequality and adopted notation.

$$Adv_{\mathbf{CTR}[\mathbf{E}]}(t, M) \leq 2 \cdot Adv^{prf}_{\mathbf{E}}(t, M) \leq 2 \cdot (Adv^{prp}_{\mathbf{E}}(t, M) + M^2 2^{-L-1}).$$

These advantage functions inequalities are based on experiments of chosen plaintext attacks on distinguishing output based on real or random. However, the observation that the output from the NIST CTR_DRBG is simply keystream is analogous to being able to launch a chosen plaintext attack. Further if we assume our codebook **E** cannot be distinguished from a *prp*, then $Adv^{prp}_{\mathbf{E}}(t, M) = 0$.

Using these results, the DRBG constructs **DRBG_E**, the above observation about their output streams, and the fair assumption that the underlying codebooks cannot be distinguished from a random permutation we get

$$Adv_{\mathbf{DRBG\_E}}(t, 1) \leq Adv_{\mathbf{CTR}[\mathbf{E}]}(t, M) \leq 2 \cdot (M^2 2^{-L-1}) = M^2 2^{-L}.$$

We can run multiple experiments of this variety, and while the underlying key will be changed subject to the update function, the individual experiments will not be effected. In this way we can get an upper bound for $Adv_{\mathbf{DRBG\_E}}(t, q)$. Under this case we can ignore the t values and consider only how many times we make a request for to the generate function in the experiment, when the experiment variable b does not change. We will express this advantage as $Adv_{\mathbf{DRBG\_E}}(q)$, and look to express an upper bound in terms of $Adv_{\mathbf{DRBG\_E}}(\cdot, 1)$, by considering the worse case, that the events are independent. Then

$$Adv_{\mathbf{DRBG\_E}}(\cdot, q) \leq 1 - (1 - \epsilon)^q$$

where ϵ is $Adv_{\mathbf{DRBG\_E}}(\cdot, 1)$. However, since ϵ is very small relative to $\frac{1}{q}$ we can approximate this as

$$Adv_{\mathbf{DRBG\_E}}(\cdot, q) \leq q \cdot \epsilon = q \cdot Adv_{\mathbf{DRBG\_E}}(\cdot, 1)$$

Thus whenever we are dealing with a specific NIST codebook DRBG we have

$$Adv_{\mathbf{DRBG\_E}}(q) \leq q \cdot Adv_{\mathbf{CTR}[\mathbf{E}]}(\ \cdot, M) \leq q \cdot (M^2 2^{-L})$$

3.2 A Lower Bound

First we will construct a distinguisher algorithm **D** on a generation function **G**. Then we will apply this distinguisher to the specific case when **G** is a **DRBG_E**, a NIST CTR_DRBG construction with codebook **E**.

Our distinguisher **D** will operate on output from a $\mathcal{Y}$-system independently searching for an $L - bit$ collision. If no collision is found the algorithm will output 1, if a collision is found the algorithm will output 0.

Distinguisher D

INPUT: A q-tuple of values $\{Y_1, Y_2, \ldots Y_q\}$, each of bit length ML.

OUTPUT: A `result` $\in \{0, 1\}$

1. Set $i \longleftarrow 1$
2. Express $Y_i = t_1^i | t_2^i | \ldots | t_M^i$, M-blocks of bit length L
3. If $t_j^i = t_l^i$ for any $0 < j < l \leq M$ `return result`$= 0$
4. Set $i \longleftarrow i + 1$
5. If $i \leq q$ `GOTO` 2
6. `return result`$= 1$.

Now we can think of the how this distinguisher can be used on a NIST CTR_DRBG, **DRBG_E** with output length ML where L is the block length and M is the maximum number of output blocks that **DRBG_E** will generate on a single call, and let $b \in \{0, 1\}$, and **D** be the above distinguisher algorithm.

In the case $b = 1$,

$$\begin{aligned} Y_i^1 =& E(V + 1(mod\ 2^L), K) | E(V + 2(mod\ 2^L), K) | ... \\ & ... | E(V + M(mod\ 2^L), K) \end{aligned}$$

and since E is a proper ECB, we are guaranteed not to repeat whenever $M < 2^{128}$. In this case the distinguisher will never see an L-bit block collision within and ML-bit output string Y_i^1, and will always output 1, therefore $Pr(Exp^1_{\mathbf{DRBG\_E,D}}(q) = 1) = 1$.

Now when $b = 0$, we want to compute $Pr(Exp^0_{\mathbf{DRBG\_E,D}}(q) = 1)$. Well our distinguisher will output 1 whenever it fails to detect an L-bit block collision in any of the q output strings. In the random case we have a small probability of a collision. If we consider just one such Y_j^0, then we first compute the probability that each L-bit block is unique $P_{ncoll,j}(2^L, M)$

$$P_{ncoll,j}(2^L, M) = \prod_{i=1}^{M-1} (1 - \frac{i}{2^L}) \approx e^{-\frac{M^2}{2^{L+1}}}$$

And so the probability of a collision $P_{coll,j}(2^L, M) \approx \frac{M^2}{2^{L+1}}$. So, after q queries of ML-bit blocks, the probability of observing a collision within a single query response, P_{coll}

$$P_{coll} = 1 - (1 - P_{coll,j})^q \approx q \cdot P_{coll,j} \approx q \cdot \left(\frac{M^2}{2^{L+1}}\right)$$

So, the distinguisher has a probability of output 1 after q such queries

$$Pr(Exp^0_{\mathbf{DRBG\_E,D}}(q) = 1) = 1 - P_{coll} \approx 1 - q \cdot \left(\frac{M^2}{2^{L+1}}\right)$$

So we can compute the advantage function,

$$\begin{aligned} Adv_{\mathbf{DRBG\_E,D}}(q) &= |Pr(Exp^1_{\mathbf{DRBG\_E,D}}(q) = 1) - Pr(Exp^0_{\mathbf{DRBG\_E,D}}(q) = 1)| \\ &\approx 1 - \left(1 - q \cdot \left(\frac{M^2}{2^{L+1}}\right)\right) \\ &= q \cdot \left(\frac{M^2}{2^{L+1}}\right) \end{aligned}$$

So this distinguisher algorithm gives us a lower bound for the advantage function,

$$Adv_{\mathbf{DRBG\_E,D}}(q) \leq Adv_{\mathbf{DRBG\_E}}(q)$$

This gives us a fairly tight bound on the advantage function

$$q \cdot \left(\frac{M^2}{2^{L+1}}\right) \leq Adv_{\mathbf{DRBG\_E}}(q) \leq q \cdot \left(\frac{M^2}{2^L}\right)$$

3.3 Examining the bounds

So, now that we have some bounds on the advantage functions of distinguishing output from the codebook DRBG constructions from random, we can evaluate what this might mean to the specific instantiations defined in NIST SP800-90. In particular NIST defines four models $\mathcal{DRBG\_3DES}$, $\mathcal{DRBG\_AES}128$, $\mathcal{DRBG\_AES}192$, and $\mathcal{DRBG\_AES}$ defined to accept the following security parameters for instantiation.

Model/Security	112	128	192	256
$\mathcal{DRBG\_3DES}$	✓	∅	∅	∅
$\mathcal{DRBG\_AES}128$	✓	✓	∅	∅
$\mathcal{DRBG\_AES}192$	✓	✓	✓	∅
$\mathcal{DRBG\_AES}256$	✓	✓	✓	✓

Table 1: NIST CTR_DRBG Instantiation Function

In Table 1, ✓symbolizes the instantiation function is defined and ∅ is interpreted as undefined. So for majority of the matrix we can think of the security bounds on the **DRBG_E** the generate function when $\mathbf{E} \in \{\mathbf{3DES}, \mathbf{AES128}, \mathbf{AES192}, \mathbf{AES256}\}$, regardless of the security parameter. In all defined cases we look for the advantage function to be negligible in the security parameter $k \in \{112, 128, 192, 256\}$.

In the case we are dealing with 3DES $ML = 2^{13}$ and since $L = 64 = 2^6$, $M = 2^7$. We can consider q in terms of 2^m for some m. Then we have the following more concrete bounds

$$\frac{1}{2^{51-m}} \leq Adv_{\mathbf{DRBG\_3DES}}(2^m) \leq \frac{1}{2^{50-m}}$$

And similarly for AES128, AES192, and AES256 we have $ML = 2^{19}$ and $L = 128 = 2^7$, $M = 2^{12}$, and so

$$\frac{1}{2^{105-m}} \leq Adv_{\mathbf{DRBG\_AES}}(2^m) \leq \frac{1}{2^{104-m}}$$

The advantage function is negligible in these constructs in the case where the codebook width is the security parameter. However, since the indistinguishable aspects of the security is based on the underlying primitive being a pseudorandom permutation on the codebook width, the overall security seems to be diminished from the advertised security level whenever the keylength is greater than the codebook width. We can think of how this value might change based on the value of q, a summary of results are given.

Model	**Supported Security** k	**Queries** q	**Lower Bound**	**Upper Bound**
$\mathcal{DRBG\_3DES}$	112	2^{32}	2^{-19}	2^{-18}
$\mathcal{DRBG\_AES}128$	112, 128	2^{48}	2^{-57}	2^{-56}
$\mathcal{DRBG\_AES}192$	112, 128, 192	2^{64}	2^{-41}	2^{-40}
$\mathcal{DRBG\_AES}256$	112, 128, 192, 256	2^{80}	2^{-25}	2^{-24}

Table 2: Security Bounds on NIST CTR_DRBG Designs

4 Conclusion

It appears as though these designs were geared towards taking advantage of some of the provable security properties of codebooks in counter-mode. However, the security proofs for counter-mode symmetric encryption defined advantage functions that were negligible on the width of the codebook as opposed to the size of the security parameter or keysize. Therefore the security of the codebook-based DRBGs in NIST SP800-90 provide less than the advertised security whenever the security parameter is greater than the codebook width. This would effectively yield the following table summary of instantiation.

Model	112	128	192	256
$\mathcal{DRBG\_3DES}$	-	$\emptyset$	$\emptyset$	$\emptyset$
$\mathcal{DRBG\_AES}128$	+	$+^*$	$\emptyset$	$\emptyset$
$\mathcal{DRBG\_AES}192$	+	$+^*$	-	$\emptyset$
$\mathcal{DRBG\_AES}256$	+	$+^*$	-	-

Figure 3: Summary of results based on NIST codebook DRBGs

The $-$ represents an allowable codebook and security parameter, but where the security is not delivered. A $+$ represents an allowable codebook and security parameter and where the security appears to be preserved. And the $\emptyset$ represent an undefined instantiation of the DRBG. The * should be noted since in these cases their does seem to be some natural advantage given to the adversary by the virtue of the fact that a 128-bit block cipher in counter-mode can be distinguish from a random function.

The upper bound used an assumption, which while fair, could use more investigation. It was assumed that the update process changing the state during calls to the DRBG generate function results in independence between consecutive queries, regardless of whether or not a reseed operation occurs. This has the potential of raising the upper bound.

References

[1] FIPS 186-2. *Digital Signature Standard (DSS).* National Institute of Standards and Technology, 1994.

[2] A. Hevia A. Desai and Y.L. Yin. A practice-oriented treatment of pseudorandom number generators. In *Advances in Cryptology - EUROCRYPT 2002*, pages 368 – 384. LNCS, Springer-Verlag, 2002.

[3] M. Bellare. A note on negligible functions. *Journal of Cryptology*, 15(4):271 – 284, 2002.

[4] Mihir Bellare, Anand Desai, Eron Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *FOCS '97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97)*, page 394, Washington, DC, USA, 1997. IEEE Computer Society.

[5] ANSI X9.82 Part 3 2006 (Draft). *American National Standards for Financial Services - Random Number Generation Part 3: Deterministic Random Bit Generators.* Accredited Standards Committee X9, Incorporated, June 2006.

[6] John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. Cryptanalytic attacks on pseudorandom number generators. *Lecture Notes in Computer Science*, 1372:168–188, 1998.

[7] Blum L. Blum, M and M. Shub. A simple unpredictable pseudorandom number generator. *SIAM J. Computing*, 15(2), May 1986.

[8] U. Maurer. Indistinguishability of random systems. In *Advances in Cryptology - EUROCRYPT 2002*, pages 110 – 133. LNCS, Springer-Verlag, 2002.

[9] David A. McGrew. Counter mode security: Analysis and recommendations, 2002.

[10] FIPS 186-2 +Change Notice. *Digital Signature Standard (DSS).* National Institute of Standards and Technology, 2000.

[11] NIST SP800-90. *Recommendations for Random Number Generation using Deterministric Random Bit Generators.* National Institute of Standards and Technology, June 2006.

[12] ANSI X9.17. *American National Standards for Financial Services - Financial Institution Key Management (Wholesale).* America Bankers Association, 1995.