4 Terms and Definitions

Algorithm	A clearly specified mathematical process for computation; a set of rules that, if followed, will give a prescribed result.
Approved	FIPS-**approved**, NIST-Recommended and/or validated by the Cryptographic Algorithm Validation Program (CAVP).
Approved entropy source	An entropy source that has been validated as conforming to SP 800-90B.
Backtracking Resistance	An RBG provides *backtracking resistance* relative to time *T* if it provides assurance that an adversary that has knowledge of the state of the RBG at some time(s) subsequent to time *T* (but incapable of performing work that matches the claimed security strength of the RBG) would be unable to distinguish between observations of *ideal random bitstrings* and (previously unseen) bitstrings that are output by the RBG at or prior to time *T*. In particular, an RBG whose design allows the adversary to "backtrack" from the initially-compromised RBG state(s) to obtain knowledge of prior RBG states and the corresponding outputs (including the RBG state and output at time *T*) would not provide backtracking resistance relative to time *T*. (Contrast with *prediction resistance*.)
Biased	A value that is chosen from a sample space is said to be biased if one value is more likely to be chosen than another value. Contrast with unbiased.
Bitstring	A bitstring is an ordered sequence of 0's and 1's. The leftmost bit is the most significant bit of the string and is the newest bit generated. The rightmost bit is the least significant bit of the string.
Bitwise Exclusive-Or	An operation on two bitstrings of equal length that combines corresponding bits of each bitstring using an exclusive-or operation.
Block Cipher	A symmetric-key cryptographic algorithm that transforms a block of information at a time using a cryptographic key. For a block-cipher algorithm, the length of the input block is the same as the length of the output block.
Consuming Application	The application (including middleware) that uses random numbers or bits obtained from an **approved** random bit generator.

Cryptographic Key (Key)	A parameter that determines the operation of a cryptographic function such as: 1. The transformation from plaintext to ciphertext and vice versa, 2. The generation of keying material, 3. A digital signature computation or verification.
Deterministic Algorithm	An algorithm that, given the same inputs, always produces the same outputs.
Deterministic Random Bit Generator (DRBG)	An RBG that includes a DRBG mechanism and (at least initially) has access to a source of entropy input. The DRBG produces a sequence of bits from a secret initial value called a seed, along with other possible inputs. A DRBG is often called a Pseudorandom Number (or Bit) Generator.
DRBG Mechanism	The portion of an RBG that includes the functions necessary to instantiate and uninstantiate the RBG, generate pseudorandom bits, (optionally) reseed the RBG and test the health of the the DRBG mechanism.
DRBG Mechanism Boundary	A conceptual boundary that is used to explain the operations of a DRBG mechanism and its interaction with and relation to other processes. (See min-entropy.)
Entropy	A measure of the disorder, randomness or variability in a closed system. Min-entropy is the measure used in this Recommendation.
Entropy Input	An input bitstring that provides an assessed minimum amount of unpredictability for a DRBG mechanism. (See min-entropy.)
Entropy Source	A combination of a noise source (e.g., thermal noise or hard drive seek times), health tests, and an optional conditioning component that produce random bitstrings to be used by an RBG.
Equivalent Process	Two processes are equivalent if, when the same values are input to each process, the same output is produced.
Exclusive-or	A mathematical operation; the symbol $\oplus$, defined as: $0 \oplus 0 = 0$ $\quad$ $1 \oplus 0 = 1$ $0 \oplus 1 = 1$ $\quad$ $1 \oplus 1 = 0$ Equivalent to binary addition without carry.

Fresh Entropy	A bitstring output from a source of entropy input for which there is a negligible probability that it has been previously output by the source and a negligible probability that the bitstring has been previously used by the DRBG.
Full Entropy	For the purposes of this Recommendation, an *n*-bit string is said to have full entropy if that bitstring is estimated to contain at least $(1-\varepsilon)n$ bits of entropy, where $0 \leq \varepsilon \leq 2^{-64}$. A source of full-entropy bitstrings serves as a practical approximation to a source of ideal random bitstrings of the same length (see ideal random sequence).
Hash Function	A (mathematical) function that maps values from a large (possibly very large) domain into a smaller range. The function satisfies the following properties: 1. (One-way) It is computationally infeasible to find any input that maps to any pre-specified output; 2. (Collision free) It is computationally infeasible to find any two distinct inputs that map to the same output.
Health Testing	Testing within an implementation immediately prior to or during normal operation to determine that the implementation continues to perform as implemented and as validated
Ideal Random Bitstring	See Ideal Random Sequence.
Ideal Random Sequence	Each bit of an ideal random sequence is unpredictable and unbiased, with a value that is independent of the values of the other bits in the sequence. Prior to the observation of the sequence, the value of each bit is equally likely to be 0 or 1, and, the probability that a particular bit will have a particular value is unaffected by knowledge of the values of any or all of the other bits. An ideal random sequence of *n* bits contains *n* bits of entropy.
Implementation	An implementation of an RBG is a cryptographic device or portion of a cryptographic device that is the physical embodiment of the RBG design, for example, some code running on a computing platform.
Implementation Testing for Validation	Testing by an independent and accredited party to ensure that an implemention of this Recommendation conforms to the specifications of this Recommendation.
Instantiation of an RBG	An instantiation of an RBG is a specific, logically independent, initialized RBG. One instantiation is

	distinguished from another by a "handle" (e.g., an identifying number).
Internal State	The collection of stored information about a DRBG instantiation. This can include both secret and non-secret information.
Key	See Cryptographic Key.
Min-entropy	The *min-entropy* (in bits) of a random variable X is the largest value m having the property that each observation of X provides at least m bits of information (i.e., the min-entropy of X is the greatest lower bound for the information content of potential observations of X). The min-entropy of a random variable is a lower bound on its entropy. The precise formulation for min-entropy is $-(\log_2 \max p_i)$ for a discrete distribution having probabilities $p_1,\ldots, p_n$. Min-entropy is often used as a worst-case measure of the unpredictability of a random variable. Also see SP 800-90B.
Non-Deterministic Random Bit Generator (Non-deterministic RBG) (NRBG)	An RBG that always has access to an *entropy source* and (when working properly) produces output bitstrings that have *full entropy*. Often called a True Random Number (or Bit) Generator. (Contrast with a *deterministic random bit generator* (DRBG)).
Nonce	A time-varying value that has at most a negligible chance of repeating, e.g., a random value that is generated anew for each use, a timestamp, a sequence number, or some combination of these.
Personalization String	An optional string of bits that is combined with a secret entropy input and (possibly) a nonce to produce a seed.
Prediction Resistance	An RBG provides prediction resistance relative to time T if it provides assurance that an adversary with knowledge of the state of the RBG at some time(s) prior to T (but incapable of performing work that matches the claimed *security strength* of the RBG) would be unable to distinguish between observations of ideal random bitstrings and (previously unseen) bitstrings output by the RBG at or subsequent to time T. In particular, an RBG whose design allows the adversary to step forward from the initially compromised RBG state(s) to obtain knowledge of subsequent RBG states and the corresponding outputs (including the RBG state and output at time T) would not provide prediction resistance relative to

	time *T*. (Contrast with *backtracking resistance*.)
Pseudorandom	A process (or data produced by a process) is said to be pseudorandom when the outcome is deterministic, yet also effectively random, as long as the internal action of the process is hidden from observation. For cryptographic purposes, "effectively" means "within the limits of the intended cryptographic strength."
Pseudorandom Number Generator	See Deterministic Random Bit Generator.
Random Number	For the purposes of this Recommendation, a value in a set that has an equal probability of being selected from the total population of possibilities and, hence, is unpredictable. A random number is an instance of an unbiased random variable, that is, the output produced by a uniformly distributed random process.
Random Bit Generator (RBG)	A device or algorithm that outputs a sequence of binary bits that appears to be statistically independent and unbiased. An RBG is either a DRBG or an NRBG.
Reseed	To acquire additional bits that will affect the internal state of the DRBG mechanism.
Secure Channel	A path for transferring data between two entities or components that ensures confidentiality, integrity and replay protection, as well as mutual authentication between the entities or components. The secure channel may be provided using cryptographic, physical or procedural methods, or a combination thereof.
Security Strength	A number associated with the amount of work (that is, the number of operations of some sort) that is required to break a cryptographic algorithm or system in some way. In this Recommendation, the security strength is specified in bits and is a specific value from the set {112, 128, 192, 256}. If the security strength associated with an algorithm or system is *S* bits, then it is expected that (roughly) 2^S basic operations are required to break it.
Seed	Noun : A string of bits that is used as input to a DRBG mechanism. The seed will determine a portion of the internal state of the DRBG, and its entropy must be sufficient to support the security strength of the DRBG. Verb : To acquire bits with sufficient entropy for the desired

6 Document Organization

This Recommendation is organized as follows:

— Section 7 provides a functional model for a DRBG that uses a DRBG mechanism and discusses the major components of the DRBG mechanim.

— Section 8 provides concepts and general requirements for the implementation and use of a DRBG mechanism.

— Section 9 specifies the functions of a DRBG mechanism that are introduced in Section 8. These functions use the DRBG algorithms specified in Section 10.

— Section 10 specifies **approved** DRBG algorithms. Algorithms have been specified that are based on the hash functions specified in [FIPS 180], and block cipher algorithms specified in [FIPS197] and [SP 800-67] (AES and TDEA, respectively).

— Section 11 addresses assurance issues for DRBG mechanisms, including documentation requirements, implementation validation and health testing.

This Recommendation also includes the following appendices:

— Appendix A provides conversion routines.

— Appendix B provides example pseudocode for each DRBG mechanism. Examples of the values computed for the DRBGs using each **approved** cryptographic algorithm and key size are available at http://csrc.nist.gov/groups/ST/toolkit/examples.html under the entries for SP 800-90A.

— Appendix C provides a discussion on DRBG mechanism selection.

— Appendix D provides references.

— Appendix E provides a list of modifications to SP 800-90A since it was first published.

3. The reseed function acquires new entropy input and combines it with the current internal state and any additional input that is provided to create a new seed and a new internal state.
4. The uninstantiate function zeroizes (i.e., erases) the internal state.
5. The health test function determines that the DRBG mechanism continues to function correctly.

8. DRBG Mechanism Concepts and General Requirements

8.1 DRBG Mechanism Functions

A DRBG mechanism requires instantiate, uninstantiate, generate, and health testing functions. A DRBG mechanism includes an optional reseed function. A DRBG **shall** be instantiated prior to the generation of output by the DRBG. These functions are specified in Section 9.

8.2 DRBG Instantiations

A DRBG may be used to obtain pseudorandom bits for different purposes (e.g., DSA private keys and AES keys) and may be separately instantiated for each purpose, thus effectively creating two DRBGs.

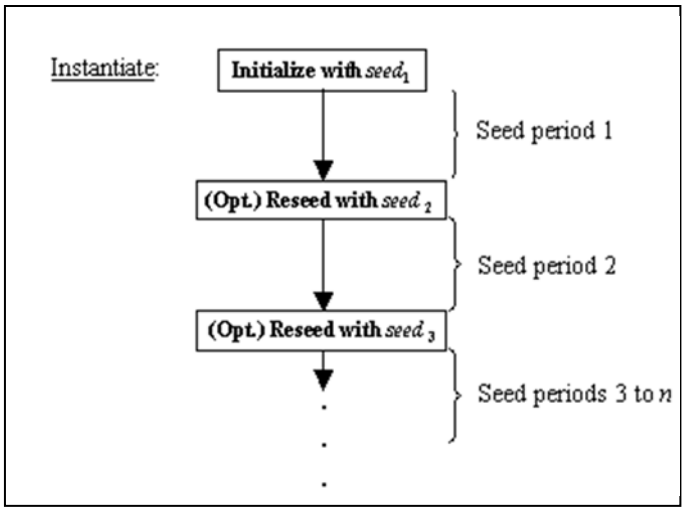

Figure 2: DRBG Instantiation

A DRBG is instantiated using a seed and may be reseeded; when reseeded, the seed **shall** be different than the seed used for instantiation. Each seed defines a *seed period* for the DRBG instantiation; an instantiation consists of one or more seed periods that begin when a new seed is acquired and end when the next seed is obtained or the DRBG is no longer used (see Figure 2).

8.3 Internal States

During instantiation, an initial internal state is derived from the seed. The internal state for an instantiation includes:

1. The working state:
   a. One or more values that are derived from the seed and become part of the internal state; these values **shall** remain secret, and
   b. A count of the number of requests produced since the instantiation was seeded or reseeded.
2. Administrative information (e.g., security strength and prediction resistance flag).

The internal state **shall** be protected at least as well as the intended use of the pseudorandom output bits requested by the consuming application. A DRBG mechanism implementation may be designed to handle multiple instantiations. Each DRBG

instantiation **shall** have its own internal state. The internal state for one DRBG instantiation **shall not** be used as the internal state for a different instantiation.

8.4 Security Strengths Supported by an Instantiation

The DRBG mechanisms specified in this Recommendation support four security strengths: 112, 128, 192 or 256 bits. The security strength for the instantiation is requested during DRBG instantiation, and the instantiate function obtains the appropriate amount of entropy for the requested security strength. Each DRBG mechanism has restrictions on the security strength it can support, based on its design (see Section 10).

The actual security strength supported by a given instantiation depends on the DRBG implementation and on the amount of entropy provided to the instantiate function. Note that the security strength actually supported by a particular instantiation could be less than the maximum security strength possible for that DRBG implementation (see Table 1). For example, a DRBG that is designed to support a maximum security strength of 256 bits could, instead, be instantiated to support only a 128-bit security strength if the additional security provided by the 256-bit security strength is not required (i.e., by requesting only 128 bits of entropy during instantiation, rather than 256 bits of entropy).

Table 1: Possible Instantiated Security Strengths

Maximum Designed Security Strength	**112**	**128**	**192**	**256**
Possible Instantiated Security Strengths	112	112, 128	112, 128, 192	112, 128, 192, 256

Following instantiation, requests can be made to the generate function of that instantiation for pseudorandom bits. These pseudorandom bits **shall not** be used for any application that requires a higher security strength than the DRBG is instantiated to support. For example, a DRBG instantiated to support a security strength of 112 bits must not be used to generate a key intended to support a security strength of 256 bits.

For each generate request, the security strength to be provided for the bits is requested. Any security strength can be requested during a call to the generate function, up to the security strength of the instantiation, e.g., an instantiation could be instantiated at the 128-bit security strength, but a request for pseudorandom bits could indicate that a lesser security strength is actually required for the bits to be generated. Assuming that the request is valid, the requested number of bits is returned.

When an instantiation is used for multiple purposes, the minimum entropy requirement for each purpose must be considered. The DRBG needs to be instantiated for the highest security strength required. For example, if one purpose requires a security strength of 112 bits, and another purpose requires a security strength of 256 bits, then the DRBG needs to be instantiated to support the 256-bit security strength.

8.5 DRBG Mechanism Boundaries

As a convenience, this Recommendation uses the notion of a "DRBG mechanism boundary" to explain the operations of a DRBG mechanism and its interaction with and relation to other processes; a DRBG mechanism boundary contains all DRBG mechanism functions and internal states required for a DRBG. Data enters a DRBG mechanism boundary via the DRBG's public interfaces, which are made available to consuming applications. The DRBG mechanism boundary should not be confused with a cryptographic module boundary, as specified in [FIPS 140]; the relationship between a cryptographic module boundary and a DRBG boundary is discussed below and in [SP 800-90C].

Within a DRBG mechanism boundary,

1. The DRBG internal state and the operation of the DRBG mechanism functions **shall** only be affected according to the DRBG mechanism specification.
2. The DRBG internal state **shall** exist solely within the DRBG mechanism boundary. The internal state **shall not** be accessible by non-DRBG functions or other instantiations of that or other DRBGs.
3. Information about secret parts of the DRBG internal state and intermediate values in computations involving these secret parts **shall not** affect any information that leaves the DRBG mechanism boundary, except as specified for the DRBG pseudorandom bit outputs.

Each DRBG mechanism includes one or more cryptographic primitives (i.e., a hash function or block-cipher algortihm). Other applications may use the same cryptographic primitive, but the DRBG's internal state and the DRBG mechanism functions **shall not** be affected by these other applications. For example, a DRBG mechanism may use the same hash-function code as a digital-signature application.

A DRBG mechanism's functions may be contained within a single device, or may be distributed across multiple devices (see Figures 3 and 4). Figure 3 depicts a DRBG for which all functions are contained within the same device. In this case, the DRBG mechanism boundary is the same as or is fully contained within a cryptographic module boundary.

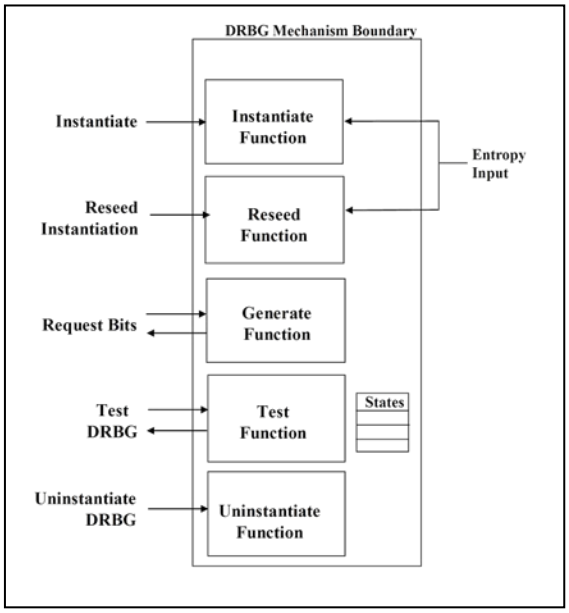

Figure 3: DRBG Mechanism Functions within a Single Device

Figure 4 provides an example of DRBG mechanism functions that are distributed across multiple devices. In this case, each device has a DRBG mechanism sub-boundary that contains the DRBG

mechanism functions implemented on that device. The boundary around the entire DRBG mechanism includes the aggregation of sub-boundaries providing the DRBG mechanism functionality. The use of distributed DRBG-mechanism functions may be convenient for restricted environments (e.g., smart card applications) in which the primary use of the DRBG does not require repeated use of the instantiate or reseed functions.

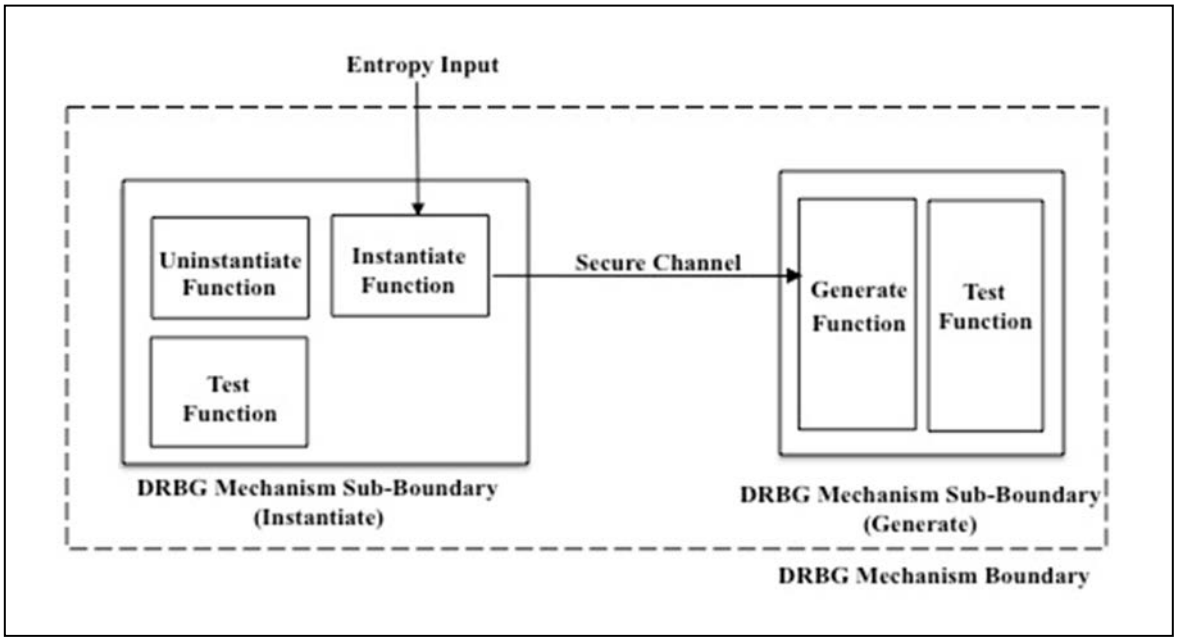

Figure 4: Distributed DRBG Mechanism Functions

Each DRBG mechanism boundary or sub-boundary **shall** contain a test function to test the "health" of other DRBG-mechanism functions within that boundary. In addition, a boundary or sub-boundary that contains an instantiate function **shall** contain an uninstantiate function in order to perform and/or react to health testing.

When DRBG mechanism functions are distributed, a physically or cryptographically secure channel **shall** be used to protect the confidentiality and integrity of the internal state or parts of the internal state that are transferred between the distributed DRBG mechanism sub-boundaries. The security provided by the secure channel **shall** be consistent with the security required by the consuming application.

For distributed DRBGs, each sub-boundary is the same as or is fully contained within a separate cryptographic module boundary.

8.6 Seeds

When a DRBG is used to generate pseudorandom bits, a seed **shall** be acquired prior to the generation of output bits by the DRBG. The seed is used to instantiate the DRBG and