10.1.2.2.2 Instantiation of Hash_DRBG

Notes for the instantiate function:

The instantiation of **Hash_DRBG** requires a call to the instantiate function specified in Section 9.2; step 10 of that function calls the instantiate algorithm in this section. For this DRBG, no *DRBG_specific_input_parameters* are required for the instantiate function specified in Section 9.2 (i.e., step 5 **should** be omitted).

The values of *highest_supported_security_strength* and *min_entropy_input_length* are provided in Table 3 of Section 10.1.1. The contents of the internal state are provided in Section 10.1.2.2.1.

The instantiate algorithm:

Let **Hash_df** be the hash derivation function specified in Section 9.6.2 using the selected hash function. The output block length (*outlen*), seed length (*seedlen*) and appropriate *security_strengths* for the implemented hash function are provided in Table 3 of Section 10.1.1.

The following process or its equivalent **shall** be used as the instantiate algorithm for this DRBG (see step 10 in Section 9.2).

Input:

1. *entropy_input*: The string of bits obtained from the entropy input source.
2. *personalization_string*: The personalization string received from the consuming application. If a *personalization_string* will never be used, then steps 1 and 2 may be combined as follows:

   seed = **Hash_df** (*entropy_input*, *seedlen*).

Output:

1. *working_state*: The inital values for *V*, *C* and *reseed_counter* (see Section 10.1.2.2.1).

Process:

1. *seed_material* = *entropy_input* || *personalization_string*.
2. *seed* = **Hash_df** (*seed_material*, *seedlen*).
3. *V* = *seed*.
4. *C* = **Hash_df** ((0x00 || *V*), *seedlen*). Comment: Preceed *V* with a byte of zeroes.
5. *reseed_counter* = 1.
6. Return *V*, *C* and *reseed_counter* as the *working_state*.

10.1.2.2.3 Reseeding a Hash_DRBG Instantiation

Notes for the reseed function:

The reseeding of a **Hash_DRBG** instantiation requires a call to the reseed function specified in Section 9.3; step 5 of that function calls the reseed algorithm specified in this section. The values for *min_entropy_input_length* are provided in Table 3 of Section 10.1.1.

The reseed algorithm:

Let **Hash_df** be the hash derivation function specified in Section 9.6.2 using the selected hash function. The value for *seedlen* is provided in Table 3 of Section 10.1.1.

The following process or its equivalent **shall** be used as the reseed algorithm for this DRBG (see step 6 in Section 9.3):

Input:

1. *working_state*: The current values for *V*, *C* and *reseed_counter* (see Section 10.1.2.2.1).
2. *entropy_input*: The string of bits obtained from the entropy input source.
3. *additional_input*: The additional input string received from the consuming application. If *additional_input* will never be provided, then step 1 may be modified to remove the *additional_input.*

Output:

1. *working_state*: The new values for *V*, *C* and *reseed counter.*

Process:

1. *seed_material* = 0x01 || *V* || *entropy_input* || *additional_input.*
2. *seed* = **Hash_df** (*seed_material*, *seedlen*).
3. *V* = *seed.*
4. *C* = **Hash_df** (0x00 || *V*), *seedlen*). Comment: Preceed with a byte of all zeros.
5. *reseed_counter* = 1.
6. Return *V*, *C* and *reseed_counter* as the new *working_state*.

10.1.2.2.4 Generating Pseudorandom Bits Using Hash_DRBG

Notes for the generate function:

The generation of pseudorandom bits using a **Hash_DRBG** instantiation requires a call to the generate function specified in Section 9.4; step 8 of that function calls the generate algorithm specified in this section. The values for *max_number_of_bits_per_request* and *outlen* are provided in Table 3 of Section 10.1.1.

The generate algorithm:

Let **Hash** be the selected hash function. The seed length (*seedlen*) and the maximum interval between reseeding (*reseed_interval*) are provided inTable 3 of Section 10.1.1. Note that for this DRBG, the reseed counter is used to update the value of *V* as well as to count the number of generation requests.

The following process or its equivalent **shall** be used as the generate algorithm for this DRBG (see step 8 of Section 9.4):

Input:

1. *working_state*: The current values for *V*, *C* and *reseed_counter* (see Section 10.1.2.2.1).
2. *requested_number_of_bits*: The number of pseudorandom bits to be returned to the generate function.
3. *additional_input*: The additional input string received from the consuming application. If *additional_input* will never be provided, then step 2 may be omitted.

Output:

1. *status*: The status returned from the function. The *status* will indicate **SUCCESS** or indicate that a reseed is required before the requested pseudorandom bits can be generated. In the latter case, either nothing but the reseed indication **shall** be returned as output, or a *Null* string **shall** be returned as the *returned_bits* (see below).
2. *returned_bits*: The pseudorandom bits to be returned to the generate function.
3. *working_state*: The new values for *V*, *C* and *reseed_counter*.

Process:

1. If *reseed_counter* > *reseed_interval*, then return an indication that a reseed is required.
2. If (*additional_input* ≠ *Null*), then do

   2.1 w = **Hash** (0x02 || V || *additional_input*).

   2.2 $V = (V + w) \bmod 2^{seedlen}$.
3. *returned_bits* = **Hashgen** (*requested_number_of_bits*, *V*).
4. H = **Hash** (0x03 || V).
5. $V = (V + H + C + reseed\_counter) \bmod 2^{seedlen}$.
6. *reseed_counter* = *reseed_counter* + 1.
7. Return **SUCCESS**, *returned_bits*, and the new values of *V*, *C* and *reseed_counter* for the new *working_state*.

Hashgen (...):

Input:

1. *requested_no_of_bits*: The number of bits to be returned.
2. *V*: The current value of *V*.

Output:

1. *returned_bits*: The generated bits to be returned to the generate function.

Process:

1. $m = \left\lceil \frac{requested\_no\_of\_bits}{outlen} \right\rceil$.
2. $data = V$.
3. W = the *Null* string.
4. For $i = 1$ to m

   4.1 w_i = **Hash** ($data$).

   4.2 $W = W \| w_i$.

   4.3 $data = (data + 1) \bmod 2^{seedlen}$.
5. *returned_bits* = Leftmost (*requested_no_of_bits*) bits of W.
6. Return returned_bits.