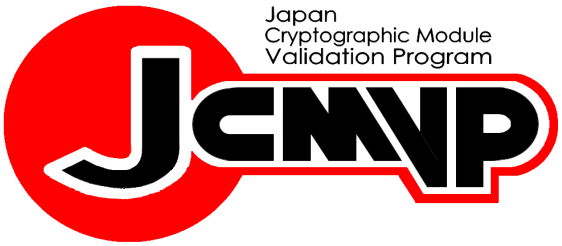


暗号アルゴリズム実装試験仕様書
– 乱数生成器 –

令和元年 7 月 11 日

ATR-01-E

Cryptographic Algorithm Implementation Testing Requirements

独立行政法人情報処理推進機構

目 次

1 目的

本書は, 暗号アルゴリズム実装試験ツール (JCATT) に実装された乱数生成器に関する暗号アルゴリズム実装試験仕様を記述するものである. 試験の対象とする暗号アルゴリズムは, 2 章に示す通りである.

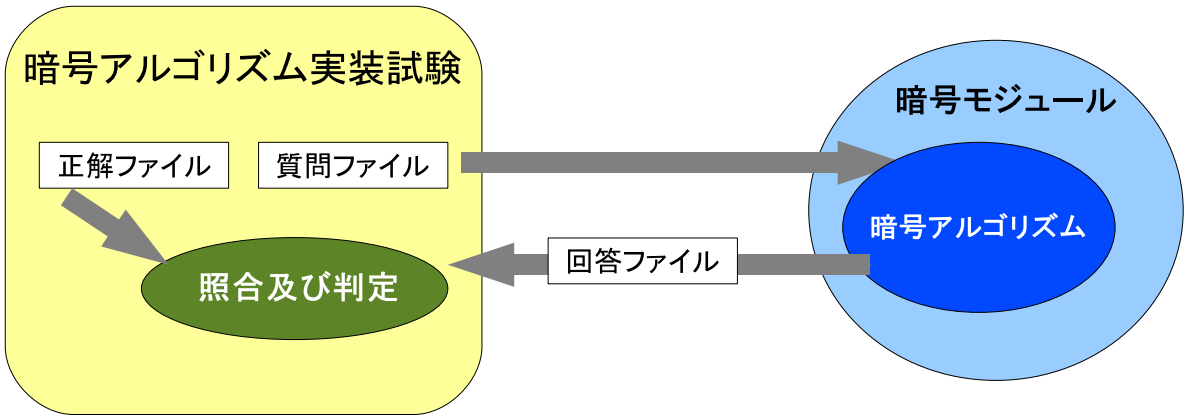
1.1 暗号アルゴリズム実装試験ツールの概要

暗号アルゴリズム実装試験ツールは次の特長を持つ.

- 試験対象の実装が暗号アルゴリズム仕様書に記述された事項に従って実装されているかどうかを試験する.
- 例えば乱数生成器の場合は擬似乱数生成など, 各暗号が有する機能ごとに試験を行う.
- 暗号アルゴリズム実装試験ツールと試験対象の実装は, 各種ファイルを介してデータの通信を行う. このことにより, 様々なプラットフォーム上の暗号実装を試験可能となる. ここで, ツールで使う各種ファイルの内訳は以下のとおりである.
  - 質問ファイル: 暗号アルゴリズム実装試験ツールが生成するファイル. 暗号アルゴリズムに対する入力データ及び制御情報が記録されている. 暗号モジュール試験機関からベンダ側へ送る.
  - 正解ファイル: 暗号アルゴリズム実装試験ツールが質問ファイルと同時に生成するファイル. 暗号アルゴリズムに対する入力データ, 制御情報及び対応する出力データが記録されている. 暗号モジュール試験機関で保存し, 回答ファイルが送られてきた際に回答ファイルと照合する.
  - 回答ファイル: ベンダ側で, 質問ファイルを元に暗号モジュールが生成したテキストファイル. ベンダから暗号モジュール試験機関側へ送る.

ファイルフォーマットは文献 [1], サンプルファイルは文献 [2] を参照.
暗号アルゴリズム実装試験の流れは, 図 1.1 の通りである.

図 1.1: 暗号アルゴリズム実装試験の流れ



1.2 本書の構成

本書の以降の構成は次の通りである.

- 2 章: 暗号アルゴリズム実装試験ツールが試験の対象とする暗号アルゴリズムを示す.

- 3 章以降: 各暗号の試験項目を記述する.

なお, 本書を通して次の略語を使用する.

- JCATT: 暗号アルゴリズム実装試験ツール
- IUT: JCATT が試験の対象とする実装

2　本書で対象とする承認されたセキュリティ機能

本書が対象とする暗号アルゴリズムを次に示す.

2.1　乱数生成器

2.1.1　決定論的乱数生成器

- Hash_DRBG in NIST SP 800-90A
- HMAC_DRBG in NIST SP 800-90A
- CTR_DRBG with DF in NIST SP 800-90A
- CTR_DRBG without DF in NIST SP 800-90A

3 暗号アルゴリズム実装試験仕様 – 乱数生成器 –

3.1 NIST SP800-90A に記載された決定論的乱数生成器

3.1.1 Hash_DRBG, HMAC_DRBG, and CTR_DRBG in NIST SP 800-90A

3.1.1.1 NIST SP800-90A に記載された決定論的乱数生成器と暗号アルゴリズム実装試験

次の乱数生成器ついて, 以下に詳細を記述する.

- Hash_DRBG in NIST SP 800-90A
- HMAC_DRBG in NIST SP 800-90A
- CTR_DRBG with DF in NIST SP 800-90A
- CTR_DRBG without DF in NIST SP 800-90A

暗号アルゴリズム実装試験ツール (JCATT) では, NIST SP800-90A[3] の Section 9 及び Section 10 に記載された決定論的乱数生成器の仕様の内, 正常系の動作が正しいことのみを確認する. JCATT による暗号アルゴリズム実装試験で検証されない要求事項については, 対応する試験要件が明らかになったときに検証される.

NIST SP800-90A では, 決定論的乱数生成器 (以下 DRBG と言う) の内部状態を初期化するための乱数シードを, DRBG の利用アプリケーションから入力することを禁止しているが, 暗号アルゴリズム実装試験用インタフェースに限って, 乱数シードを入力することが許可されている.

そこで, まず 3.1.1.2 節で, NIST SP800-90A の Section 9 に記載の DRBG 機能を拡張した暗号アルゴリズム実装試験用のインタフェースを定義する. 次に 3.1.1.3 節では, 3.1.1.2 で定義した暗号アルゴリズム実装試験用インタフェースを用いて, DRBG に対する試験 1∼ 試験 3 を定義する. これらの試験は, NIST DRBGVS[4] に準拠している.

なお, 3.1.1.3 で述べる DRBG の試験の記述は, 上記 4 種類の DRBG に共通である.

NIST SP800-90A で規定された DRBG は, 次の暗号アルゴリズムを組み合わせて使用する.

- Hash_DRBG は, FIPS 180-4 で規定されたハッシュ関数
- HMAC_DRBG は, FIPS 198-1 で規定された HMAC
- CTR_DRBG は, FIPS 197 で規定された AES 又は NIST SP800-67 で規定された 3-key Triple DES

DRBG の暗号アルゴリズム実装試験に先立って, 対応する暗号アルゴリズムの暗号アルゴリズム実装試験に合格している必要がある.

3.1.1.2 DRBG 機能の試験用インタフェース

3.1.1.2.1 InstantiateTestIF 関数

機能概要

暗号アルゴリズム実装試験用に提供されたパラメータを用いて DRBG の初期化を行う.

関数定義

$$(status, state\_handle) = \texttt{InstatiateTestIF}(requested\_instantiation\_security\_strength,\ prediction\_resistance\_flag,\ personalization\_string,\ entropy\_input,\ nonce)$$

戻り値

変数名	型	説明
status	-	InstantiateTestIF 関数の戻り値の 1 つ. NIST SP800-90A の 9.1 で規定されている. SUCCESS 又は何らかの ERROR を示す.
state_handle	整数	InstantiateTestIF 関数の戻り値の 1 つで, 初期化された DRBG インスタンスへのポインタ. NIST SP800-90A の 9.1 で規定されている.

パラメータ

変数名	型	説明
requested_instantiation_security_strength	整数	DRBG インスタンスがサポートするセキュリティ強度をビット数で表したもの. NIST SP800-90A の 9.1 で規定されている.
prediction_resistance_flag	整数	DRBG の利用アプリケーションが, DRBG インスタンスに prediction resistance を要求するかどうかを表す変数. NIST SP800-90A の 9.1 で規定されている.
personalization_string	ビット列	DRBG インスタンスを他の DRBG インスタンスと差別化するために入力されるビット列. NIST SP800-90A の 9.1 で規定されている.

試験用インタフェース拡張パラメータ

変数名	型	説明
entropy_input	ビット列	エントロピーを含む入力ビット列. そのビット長については, NIST SP800-90A の Section 10 で選択された DRBG のメカニズム毎に規定されている. 暗号アルゴリズム実装試験用に, 入力パラメータとして設定している. 本来, *entropy_input* は, NIST SP800-90A の 9.1 Step 6 で Get_entropy_input 関数の戻り値として得られるが, DRBG の試験用インタフェースでは, 暗号アルゴリズム実装試験のために外部から入力できるよう拡張している.
nonce	ビット列	NIST SP800-90A の 8.6.7 で規定されているビット列. 暗号アルゴリズム実装試験用に, 入力パラメータとして設定している. 本来, NIST SP800-90A の Section 9 に記載されているとおり, 内部で生成するものであるが, DRBG の試験用インタフェースでは, 暗号アルゴリズム実装試験のために外部から入力できるよう拡張している.

3.1.1.2.2 ReseedTestIF 関数

機能概要

暗号アルゴリズム実装試験用に提供されたパラメータを用いて, DRBG インスタンスに reseed を行う.

関数定義

status = `ReseedTestIF`(*state_handle*,
additional_input,
entropy_input)

戻り値

変数名	型	説明
status	-	`ReseedTestIF` 関数の戻り値の 1 つ. NIST SP800-90A の 9.2 で規定されている. SUCCESS 又は何らかの ERROR を示す.

パラメータ

変数名	型	説明
state_handle	整数	`ReseedTestIF` 関数の戻り値の 1 つで, reseed される DRBG の内部状態へのポインタ. NIST SP800-90A の 9.2 で規定されている. `ReseedTestIF` 関数の呼び出しによって, DRBG の内部状態は更新される.
additional_input	ビット列	オプションの入力ビット列. そのビット長については, NIST SP800-90A の Section 10 で選択された DRBG のメカニズム毎に規定されている.

試験用インタフェース拡張パラメータ

変数名	型	説明
entropy_input	ビット列	エントロピーを含む入力ビット列. そのビット長については, NIST SP800-90A の Section 10 で選択された DRBG のメカニズム毎に規定されている. 暗号アルゴリズム実装試験用に, 入力パラメータとして設定している. 本来, *entropy_input* は, NIST SP800-90A の 9.2 Step 4 で Get_entropy_input 関数の戻り値として得られるが, DRBG の試験用インタフェースでは, 暗号アルゴリズム実装試験のために外部から入力できるよう拡張している.

3.1.1.2.3 GenerateTestIF 関数

機能概要

指定された DRBG インスタンスと, 暗号アルゴリズム実装試験用に提供されたパラメータを用いて, 擬似乱数ビット列の生成を行う.

関数定義

(*status*, *pseudorandom_bits*) = `GenerateTestIF`(*state_handle*,
requested_number_of_bits,
requested_security_strength,
prediction_resistance_request,
additional_input,
entropy_input)

戻り値

変数名	型	説明
status	-	`GenerateTestIF` 関数の戻り値の 1 つ. NIST SP800-90A の 9.3.1 で規定されている.
pseudorandom_bits	ビット列	`GenerateTestIF` 関数で生成された擬似乱数ビット列. このビット列の長さは *requested_number_of_bits* である.

パラメータ

変数名	型	説明
state_handle	整数	擬似乱数ビット列生成のために使用される DRBG の内部状態へのポインタ. NIST SP800-90A の 9.3.1 で規定されている. `GenerateTestIF` 関数の呼び出しによって, DRBG の内部状態は更新される.
requested_number_of_bits	整数	生成したい擬似乱数ビット列の長さ. NIST SP800-90A の 9.3.1 で規定されている.
requested_security_strength	整数	*pseudorandom_bits* と関連付けられるセキュリティ強度. NIST SP800-90A の 9.3.1 で規定されている.
prediction_resistance_request	-	prediction resistance を要求するかどうかを表す変数. NIST SP800-90A の Section 9.3.1 で規定されている.
additional_input	ビット列	オプションの入力ビット列. そのビット長については, NIST SP800-90A の Section 10 で選択された DRBG のメカニズム毎に規定されている.

試験用インタフェース拡張パラメータ

変数名	型	説明

entropy_input	ビット列	エントロピーを含む入力ビット列. そのビット長については, NIST SP800-90A の Section 10 で選択された DRBG のメカニズム毎に規定されている. 暗号アルゴリズム実装試験用に, 入力パラメータとして設定している. 本来, *entropy_input* は, *prediction_resistance_request* が設定されていた場合, 又は NIST SP800-90A の Section 10 に記載の *reseed_counter* が *reseed_interval* に一致するか超える場合, NIST SP800-90A の 9.3.1 Step 7.1 の Reseed_function の呼び出しにより, NIST SP800-90A の 9.2 Step 4 の Get_entropy_input 関数の戻り値として得られる. DRBG の試験用インタフェースでは, 暗号アルゴリズム実装試験のために外部から入力できるよう拡張し, 最終的に *entropy_input* を処理する `ReseedTestIF` 関数に渡され, 処理される.

3.1.1.2.4 UninstantiateTestIF 関数

機能概要

指定された DRBG インスタンスのゼロ化を行う.

関数定義

status = `UninstantiateTestIF`(*state_handle*)

戻り値

変数名	型	説明
status	-	`UninstantiateTestIF` 関数の戻り値. NIST SP800-90A の 9.4 で規定されている.

パラメータ

変数名	型	説明
state_handle	整数	ゼロ化される DRBG の内部状態へのポインタ. NIST SP800-90A の 9.4 で規定されている. `UninstantiateTestIF` 関数の呼び出しによって, DRBG の内部状態はゼロ化される.

3.1.1.3 全機能試験

3.1.1.3.1 試験 1

試験 1 の概要

試験 1 は, prediction resistance を有効化した実装に対する既定の試験である. 質問ファイルで指定される次の入力ビット列を用いて, 後述の試験 1 のアルゴリズムに従って擬似乱数ビット列を生成する.

- *additional_input*
- *entropy_input*

- *nonce*
- *personalization_string*

質問ファイルに指定されているこれらのビット列は, 試験 1 では, ランダムな値が設定されている.

記号の定義

擬似コード上の表記	型	説明
$additional\_input_{i,j}$	ビット列	*addtional_input* の組の (i, j) 番目の要素.
$entropy\_input\_PR_{i,j}$	ビット列	GenerateTestIF 関数の入力となる *entropy_input* の組の (i, j) 番目の要素. 試験 1 では, ランダムな値が質問ファイルで提供される.
$entropy\_input_i$	ビット列	InstantiateTestIF 関数の入力となる *entropy_input* の組の i 番目の要素. 試験 1 では, ランダムな値が質問ファイルで提供される.
i	整数	0 から $number\_of\_trials - 1$ までの値をとる整数.
j	整数	0 又は 1.
$nonce_i$	ビット列	*nonce* の組の i 番目の要素. 試験 1 では, ランダムな値が質問ファイルで提供される.
$number\_of\_bits$	整数	GenerateTestIF 関数の呼出に使用する変数.
$number\_of\_trials$	整数	DRBG インスタンスを生成する回数.
$outlen$	整数	使用する暗号アルゴリズムの出力ビット長. 例えば, HMAC-SHA-256 を用いる HMAC_DRBG の場合には, 256 になる.
$personalization\_string_i$	ビット列	personalization string の組の i 番目の要素
$prediction\_resistance\_flag$	整数	DRBG の利用アプリケーションが, DRBG インスタンスに prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する.
$PR\_request$	整数	prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する. 試験 1 では 1 が設定される.
$pseudorandom\_bits_{i,j}$	ビット列	GenerateTestIF 関数を呼び出すことにより生成された擬似乱数ビット列の組の (i, j) 番目の要素. 回答ファイルに出力される.
$requested\_number\_of\_bits$	整数	乱数ビット列のビット長を保持する変数.
$state\_handle$	整数	DRBG インスタンスへのポインタ.
$status$	-	試験用インタフェースからの戻り値. JCATT は, 何らかの ERROR を引き起こす入力パラメータの組み合わせも生成しないため, *status* のチェックを行わない.
$strength$	整数	DRBG がサポートするセキュリティ強度.

注

1. 網掛け ( ) された項目については, 質問ファイルで提供される.

試験 **1** のアルゴリズム

Algorithm 1

1: $prediction\_resistance\_flag = 1$
▷ prediction resistance をサポートする DRBG インスタンスを生成する
2: $PR\_request = 1$ ▷ prediction resistance を要求する
3: **for** $i = 0$ to $number\_of\_trials - 1$ **do**
4: $(status, state\_handle) =$ `InstantiateTestIF`$(strength,$
$prediction\_resistance\_flag,$
$personalization\_string_i,$
$entropy\_input_i,$
$nonce_i)$ ▷ 内部状態の初期化を行う
5: **for** $j = 0$ to 1 **do**
6: $number\_of\_bits = requested\_number\_of\_bits$
▷ 通常は, 質問ファイルで指定された長さの乱数ビット列を生成
7: $(status, pseudorandom\_bits_{i,j}) =$ `GenerateTestIF`$(state\_handle,$
$number\_of\_bits,$
$strength,$
$PR\_request,$
$additional\_input_{i,j},$
$entropy\_input\_PR_{i,j})$
▷ 乱数ビット列生成を行う
8: **if** $j \neq 0$ **then**
9: **Output**$(pseudorandom\_bits_{i,j})$
▷ 2 回目以降に生成された乱数ビット列を回答ファイルに記録する
10: **end if**
11: **end for**
12: $status =$ `UninstantiateTestIF`$(state\_handle)$ ▷ 内部状態をゼロ化する
13: **end for**

3.1.1.3.2 試験 2

試験の概要

試験 2 は, reseed 機能を有するが, prediction resistance を有効化しない実装に対する既定の試験である. 後述の試験 2 のアルゴリズムに従って擬似乱数ビット列を生成する.

記号の定義

擬似コード上の表記	型	説明
$additional\_input_{i,j}$	ビット列	GenerateTestIF 関数の入力パラメータとして使用される, *addtional_input* の組の (i, j) 番目の要素.
$additional\_input\_on\_reseed_{i,j}$	ビット列	ReseedTestIF 関数の入力パラメータとして使用される, *addtional_input* の組の (i, j) 番目の要素.
$entropy\_input_i$	ビット列	InstantiateTestIF 関数の入力となる, *entropy_input* の組の i 番目の要素.
$entropy\_input\_on\_reseed_{i,j}$	ビット列	ReseedTestIF 関数の入力となる, *entropy_input* の組の (i, j) 番目の要素.
i	整数	0 から $number\_of\_trials - 1$ までの値をとる整数.
j	整数	0, 1, 又は 2.
$nonce_i$	ビット列	*nonce* の組の i 番目の要素.
$number\_of\_bits$	整数	GenerateTestIF 関数の呼出に使用する変数.
$Null$	ビット列	空のビット列.
$number\_of\_trials$	整数	DRBG インスタンスを生成する回数.
$outlen$	整数	使用する暗号アルゴリズムの出力ビット長. 例えば, HMAC-SHA-256 をつかった HMAC_DRBG の場合には, 256 になる.
$personalization\_string_i$	ビット列	personalization string の組の i 番目の要素
$prediction\_resistance\_flag$	整数	DRBG の利用アプリケーションが, DRBG インスタンスに prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する.
$PR\_request$	整数	prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する. 試験 2 では 0 が設定される.
$pseudorandom\_bits_{i,j}$	ビット列	GenerateTestIF 関数を呼び出すことにより生成された擬似乱数ビット列の組の (i, j) 番目の要素. 回答ファイルに出力される.
$requested\_number\_of\_bits$	整数	擬似乱数ビット列のビット長を保持する変数.
$state\_handle$	整数	DRBG インスタンスへのポインタ.
$status$	-	試験用インタフェースからの戻り値. JCATT は, どんな ERROR を引き起こす入力パラメータの組み合わせも生成しない.

strength	整数	DRBG がサポートするセキュリティ強度.

注

1. 網掛け ( ) された項目については, 質問ファイルで提供される.

試験 **2** のアルゴリズム

Algorithm 2

1: $prediction\_resistance\_flag = 0$
▷ prediction resistance をサポートしない DRBG インスタンスを生成する
2: $PR\_request = 0$ ▷ prediction resistance を要求しない
3: **for** $i = 0$ to $number\_of\_trials - 1$ **do**
4: $(status, state\_handle) =$ `InstantiateTestIF`$(strength,$
$prediction\_resistance\_flag,$
$personalization\_string_i,$
$entropy\_input_i,$
$nonce_i)$ ▷ 内部状態の初期化を行う
5: **for** $j = 0$ to 2 **do**
6: $number\_of\_bits = requested\_number\_of\_bits$
▷ 通常は, 質問ファイルで指定された長さの乱数ビット列を生成
7: **if** $j = 0$ **then**
8: $status =$ `ReseedTestIF`$(state\_handle,$
$additional\_input\_on\_reseed_{i,j},$
$entropy\_input\_on\_reseed_{i,j})$ ▷ Reseed を行う
9: **else**
10: $(status, pseudorandom\_bits_{i,j}) =$ `GeneterateTestIF`$(state\_handle,$
$number\_of\_bits,$
$strength,$
$PR\_request,$
$additional\_input_{i,j},$
$Null)$
▷ 乱数ビット列生成を行う
11: **end if**
12: **if** $j = 2$ **then**
13: **Output**$(pseudorandom\_bits_{i,j})$
▷ 2 回目以降に生成された乱数ビット列を回答ファイルに記録する
14: **end if**
15: **end for**
16: $status =$ `UninstantiateTestIF`$(state\_handle)$ ▷ 内部状態をゼロ化する
17: **end for**

3.1.1.3.3 試験 3

試験の概要

試験 3 は, reseed 機能を有しない実装に対する既定の試験である. 後述の試験 3 のアルゴリズムに従って擬似乱数ビット列を生成する.

記号の定義

擬似コード上の表記	型	説明
$additional\_input_{i,j}$	ビット列	GenerateTestIF 関数の入力パラメータとして使用される, *addtional_input* の組の (i, j) 番目の要素.
$entropy\_input_i$	ビット列	InstantiateTestIF 関数の入力となる, *entropy_input* の組の i 番目の要素.
i	整数	0 から $number\_of\_trials - 1$ までの値をとる整数.
j	整数	0 又は 1.
$nonce_i$	ビット列	*nonce* の組の i 番目の要素.
Null	ビット列	空のビット列.
number_of_bits	整数	GenerateTestIF 関数の呼出に使用する変数.
number_of_trials	整数	DRBG インスタンスを生成する回数.
outlen	整数	使用する暗号アルゴリズムの出力ビット長. 例えば, HMAC-SHA-256 をつかった HMAC_DRBG の場合には, 256 になる.
$personalization\_string_i$	ビット列	personalization string の組の i 番目の要素
prediction_resistance_flag	整数	DRBG の利用アプリケーションが, DRBG インスタンスに prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する. 試験 3 では 0 が設定される.
PR_request	整数	prediction resistance を要求するかどうかを表す変数. 擬似コード上, 要求する場合, 1, 要求しない場合, 0 を設定する. 試験 3 では 0 が設定される.
$pseudorandom\_bits_{i,j}$	ビット列	GenerateTestIF 関数を呼び出すことにより生成された擬似乱数ビット列の組の (i, j) 番目の要素. 回答ファイルに出力される.
requested_number_of_bits	整数	擬似乱数ビット列のビット長を保持する変数.
state_handle	整数	DRBG インスタンスへのポインタ.
status	-	試験用インタフェースからの戻り値. JCATT は, どんな ERROR を引き起こす入力パラメータの組み合わせも生成しない.
strength	整数	DRBG がサポートするセキュリティ強度.

注

1. 網掛け ( ) された項目については, 質問ファイルで提供される.

試験 **3** のアルゴリズム

Algorithm 3

1: $prediction\_resistance\_flag = 0$
▷ prediction resistance をサポートしない DRBG インスタンスを生成する
2: $PR\_request = 0$ ▷ prediction resistance を要求しない
3: **for** $i = 0$ to $number\_of\_trials - 1$ **do**
4: $(status, state\_handle) = \texttt{InstantiateTestIF}(strength,$
$prediction\_resistance\_flag,$
$personalization\_string_i,$
$entropy\_input_i,$
$nonce_i)$ ▷ 内部状態の初期化を行う
5: **for** $j = 0$ to 1 **do**
6: $number\_of\_bits = requested\_number\_of\_bits$
▷ 通常は, 質問ファイルで指定された長さの乱数ビット列を生成
7: $(status, pseudorandom\_bits_{i,j}) = \texttt{GeneterateTestIF}(state\_handle,$
$number\_of\_bits,$
$strength,$
$PR\_request,$
$additional\_input_{i,j},$
$Null)$
▷ 乱数ビット列生成を行う
8: **if** $j \neq 0$ **then**
9: **Output**$(pseudorandom\_bits_{i,j})$
▷ 2 回目以降に生成された乱数ビット列を回答ファイルに記録する
10: **end if**
11: **end for**
12: $status = \texttt{UninstantiateTestIF}(state\_handle)$ ▷ 内部状態をゼロ化する
13: **end for**

4 確認書発行条件

4.1 パラメータについて

4.1.1 NIST SP800-90A に記載された決定論的乱数生成器

NIST SP800-90A に記載された決定論的乱数生成器において, 暗号アルゴリズム確認書を発行するための条件は, 網掛けされた試験対象機能を少なくとも 1 個実装し, 暗号アルゴリズム実装試験に合格することである. 暗号アルゴリズム実装試験に使用するパラメータの入力条件およびその既定値は, 表 4.1～ 表 4.5 に記載する値とする. 暗号アルゴリズム確認書には, 次の特記事項が記載される.

「本暗号アルゴリズム確認書で識別される DRBG の暗号アルゴリズム実装が, NIST SP800-90A の Section 9, DRBG Mechanism Functions に記載されたアルゴリズムの正常系に関する暗号アルゴリズム実装試験に合格したことを証する.」

表 4.1: 決定論的乱数生成器の試験で選択可能な暗号アルゴリズムとセキュリティ強度 (Hash_DRBG in NIST SP 800-90A)

試験対象機能	入力欄		既定値	入力条件
全機能	試験 1～3 共通	使用する暗号アルゴリズム	SHA-256	以下の暗号アルゴリズムから選択可能 • SHA-1 • SHA-224 • SHA-256 • SHA-384 • SHA-512 • SHA-512/224 • SHA-512/256
		strength	256	SHA-1 を選択時, 次の値から選択可能 • 112 • 128 SHA-224 , SHA-512/224 を選択時, 次の値から選択可能 • 112 • 128 • 192 SHA-256, SHA-384, SHA-512 , SHA-512/256 を選択時, 次の値から選択可能 • 112 • 128 • 192 • 256

表 4.2: 決定論的乱数生成器の試験で選択可能な暗号アルゴリズムとセキュリティ強度 (HMAC_DRBG in NIST SP 800-90A)

試験対象機能	入力欄		既定値	入力条件
全機能	試験1～3共通	使用する暗号アルゴリズム	HMAC-SHA-256	以下の暗号アルゴリズムから選択可能 ● HMAC-SHA-1 ● HMAC-SHA-224 ● HMAC-SHA-256 ● HMAC-SHA-384 ● HMAC-SHA-512 ● HMAC-SHA-512/224 ● HMAC-SHA-512/256
		strength	256	HMAC-SHA-1 を選択時, 次の値から選択可能 ● 112 ● 128 HMAC-SHA-224 , HMAC-SHA-512/224 を選択時, 次の値から選択可能 ● 112 ● 128 ● 192 HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 , HMAC-SHA-512/256 を選択時, 次の値から選択可能 ● 112 ● 128 ● 192 ● 256

表 4.3: 決定論的乱数生成器の試験で選択可能な暗号アルゴリズム, セキュリティ強度及び *ctr_len*
(CTR_DRBG with DF, CTR_DRBG without DF in NIST SP 800-90A)

試験対象機能	入力欄		既定値	入力条件
全機能	試験1～3共通	使用する暗号アルゴリズム	AES-256	以下の暗号アルゴリズムから選択可能 • 3-key Triple DES • AES-128 • AES-192 • AES-256
		strength	256	3-key Triple DES を選択時, 112 が設定される. AES-128 を選択時, 次の値から選択可能 • 112 • 128 AES-192 を選択時, 次の値から選択可能 • 112 • 128 • 192 AES-256 を選択時, 次の値から選択可能 • 112 • 128 • 192 • 256
		ctr_len	128	• 3-key Triple DES を選択時, 4 以上, 64 以下. • AES-128, AES-192, または AES-256 を選択時, 4 以上, 128 以下.

表 4.4: 決定論的乱数生成器の既定値及び入力条件 試験 1~3 共通部分
(Hash_DRBG, HMAC_DRBG, CTR_DRBG with DF in NIST SP 800-90A)

試験対象機能	入力欄		既定値	入力条件
全機能	試験 1~3 共通	*additional_input* のビット長の最小値	0	• 8 の倍数 • 2^12 以下 • *additional_input* のビット長の最大値以下
		additional_input のビット長の最大値	1024	•8 の倍数 • 2^12 以下 • *additional_input* のビット長の最小値以上
		entropy_input のビット長	512	• 8 の倍数 • 2^12 以下 • *strength* の値以上
		nonce のビット長	128	• 8 の倍数 • 2^12 以下 • *strength* の 1/2 以上
		number_of_trials	100	• 15 以上 • 1000 以下
		personalization_string のビット長の最小値	0	• 8 の倍数 • 2^12 以下 • *personalization_string* のビット長の最大値以下
		personalization_string のビット長の最大値	1024	• 8 の倍数 • 2^12 以下 • *personalization_string* のビット長の最小値以上
		requested_number_of_bits	*outlen* の 4 倍	Hash_DRBG 又は HMAC_DRBG を選択時, • *outlen* の 1 倍以上, 256 倍以下 CTR_DRBG with DF を選択時, • *outlen* の 1 倍以上, **min**$((2^{ctr\_len} - 4), 256)$ 倍以下

表 4.5: 決定論的乱数生成器の既定値及び入力条件 試験 1~3 共通部分
(CTR_DRBG without DF in NIST SP 800-90A)

試験対象機能	入力欄		既定値	入力条件
全機能	試験 1~3 共通	*additional_input* のビット長の最小値	0	• 8 の倍数 • *additional_input* のビット長の最大値以下
		additional_input のビット長の最大値	*entropy_input* のビット長	-
			• 3-key Triple DES の場合, 232 • AES-128 の場合, 256 • AES-192 の場合, 320 • AES-256 の場合, 384	
		nonce のビット長	0	-
		number_of_trials	100	• 15 以上 • 1000 以下
		personalization_string のビット長の最小値	0	• 0 以上の 8 の倍数 • *personalization_string* のビット長の最大値以下
		personalization_string のビット長の最大値	*entropy_input* のビット長	-
		requested_number_of_bits	• 3-Key Triple DES の場合, 256 • AES の場合, 512	• *outlen* の 1 倍以上, **min**$((2^{ctr\_len}-4), 256)$ 倍以下

附 則
この手順は, 平成 21 年 1 月 23 日から施行し, 平成 21 年 1 月 8 日から適用する.

附 則
この手順は, 平成 21 年 7 月 1 日から施行し, 平成 21 年 7 月 10 日から適用する.

附 則
この手順は, 平成 24 年 2 月 29 日から施行し, 平成 24 年 6 月 1 日から適用する.

附 則
この手順は, 平成 30 年 6 月 22 日から施行し, 平成 30 年 6 月 22 日から適用する.

附 則
この手順は, 令和元年 7 月 11 日から施行し, 令和元年 7 月 11 日から適用する.

参考文献

[1] JCATT ファイルフォーマット仕様書 – 乱数生成器 –, https://www.ipa.go.jp/security/jcmvp/documents/open/jcatt/format/jcatt_fileformat_e.zip

[2] JCATT サンプルファイル – 乱数生成器 –, https://www.ipa.go.jp/security/jcmvp/documents/open/jcatt/sample/jcatt_sample_e.zip

[3] Elaine Barker and John Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, National Institute of Standards and Technology, June, 2015. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf

[4] Sharon Keller, Timothy A. Hall, *The NIST SP 800-90A Deterministic Random Bit Generator Validation System (DRBGVS)*, National Institute of Standards and Technology, October 29, 2015.

改版履歴

識別番号	ATR-01-E

改訂年月日	作成者・承認者	改訂内容
平成 21 年 1 月 23 日	橋本・仲田	新規制定
平成 21 年 7 月 1 日	櫻井・仲田	一部改正 (承認されたセキュリティ機能の改正に伴い, ISO/IEC 18031 記載の乱数生成器を削除)
平成 24 年 2 月 29 日	櫻井・仲田	一部改正 (NIST SP800-90 に記載の乱数生成器の 試験要件を追加)
平成 30 年 6 月 22 日	櫻井・江口	一部改正 (承認されたセキュリティ機能の改正に伴い, NIST SP800-90 に記載されていない乱数生成器の 試験要件を削除. NIST SP800-90 の参照先が, NIST SP800-90A Rev.1 に変更されたことに対応して, CTR_DRBG の入力パラメータに *ctr_len* を追加. DRBGVS の更新に対応.)
令和元年 7 月 11 日	櫻井・江口	一部改正 (依存関係のある暗号アルゴリズムを記載及び誤植を訂正)