IPA®
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

乱数生成器に関する説明会

IPA セキュリティセンター
情報セキュリティ認証室

2010年3月25日

説明会の流れ

- 第1部
  - 暗号アルゴリズム実装試験と暗号アルゴリズム確認
  - 暗号モジュール試験と暗号モジュール認証
  - 暗号モジュールのセキュリティ要求事項
    - 暗号鍵管理/重要情報の管理
    - 事例紹介
    - 暗号鍵管理/重要情報の管理の重要性
- 第2部
  - 乱数生成器の要求事項
    - 決定論的乱数生成器(DRBG)
    - DRBGを使った暗号鍵生成
    - ミニマムエントロピー
    - NIST SP800-90とその要求事項を満たす実装例

IPA®
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

Japan Cryptographic Module Validation Program JCMVP®

乱数生成器に関する説明会

第1部：「暗号モジュール試験及び認証制度」の紹介

暗号アルゴリズム確認制度

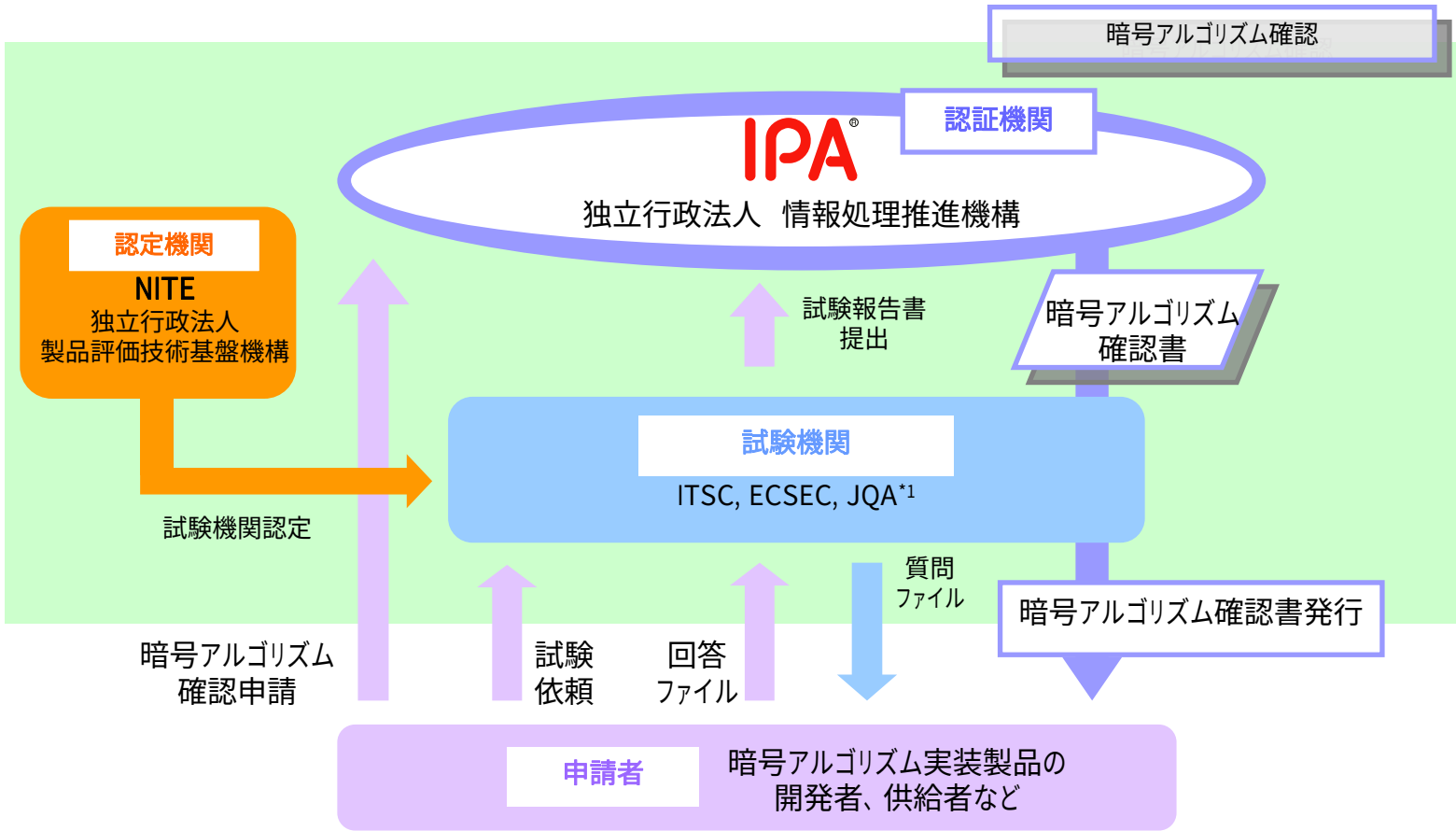
- **暗号アルゴリズム確認とは**
  JCMVP制度下で暗号モジュール試験の一部である暗号アルゴリズム実装試験を実施し、制度で承認しているセキュリティ機能が適切に実装されていることを確認することです。
  （2009年1月より実施）

- **暗号アルゴリズム実装試験には専用のツール（JCATT）を使用します。**

- 暗号アルゴリズム実装試験に合格すると「暗号アルゴリズム確認書」が発行されます。

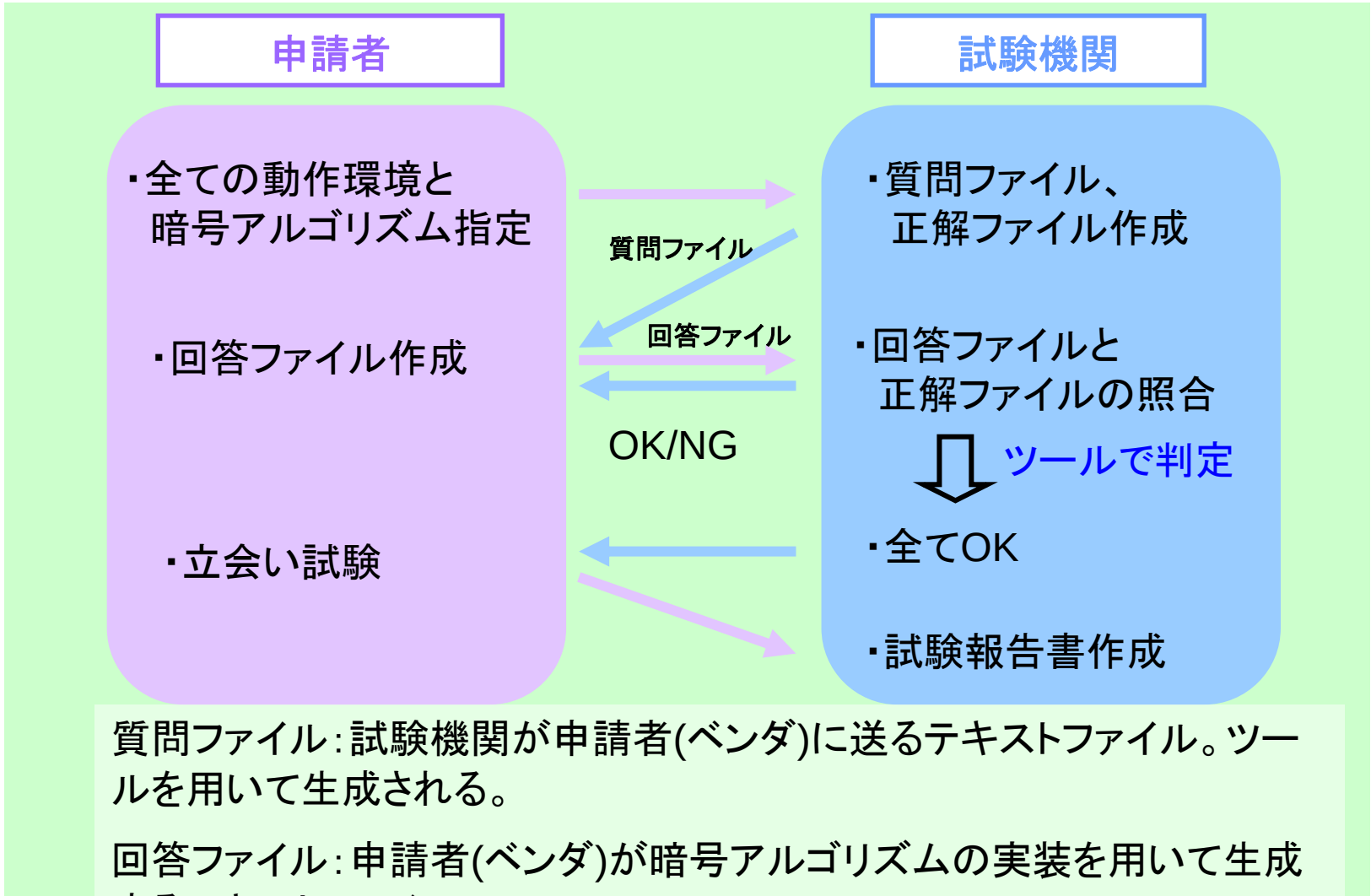
- 現在、AESアルゴリズムの確認された実装は15件。

暗号アルゴリズム確認制度の仕組み



*1 ITSC：一般社団法人 ITセキュリティセンター
ECSEC：株式会社電子商取引安全技術研究所 評価センター、
JQA：財団法人 日本品質保証機構 関西試験センター、

暗号アルゴリズム確認手順



質問ファイル：試験機関が申請者(ベンダ)に送るテキストファイル。ツールを用いて生成される。

回答ファイル：申請者(ベンダ)が暗号アルゴリズムの実装を用いて生成するテキストファイル。

暗号アルゴリズム実装試験仕様、質問ファイル、回答ファイルについて

ファイルフォーマットについては、以下の資料をセキュリティ機能別にJCMVPサイトの「公開資料」のページで公開しています。

- 暗号アルゴリズム実装試験仕様書
- JCATTファイルフォーマット仕様書
- JCATTサンプルファイル

	暗号アルゴリズム実装試験仕様書	JCATTファイルフォーマット仕様書	JCATTサンプルファイル
公開鍵	ATR-01-A	jcatt fileformat a.zip	jcatt sample a.zip
共通鍵	ATR-01-B	jcatt fileformat b.zip	jcatt sample b.zip
ハッシュ	ATR-01-C	jcatt fileformat c.zip	jcatt sample c.zip
メッセージ認証	ATR-01-D	jcatt fileformat d.zip	jcatt sample d.zip
乱数生成器	ATR-01-E	jcatt fileformat e.zip	jcatt sample e.zip
鍵確立手法	ATR-01-F	jcatt fileformat f.zip	jcatt sample f.zip

■公開資料

http://www.ipa.go.jp/security/jcmvp/open_documents.html

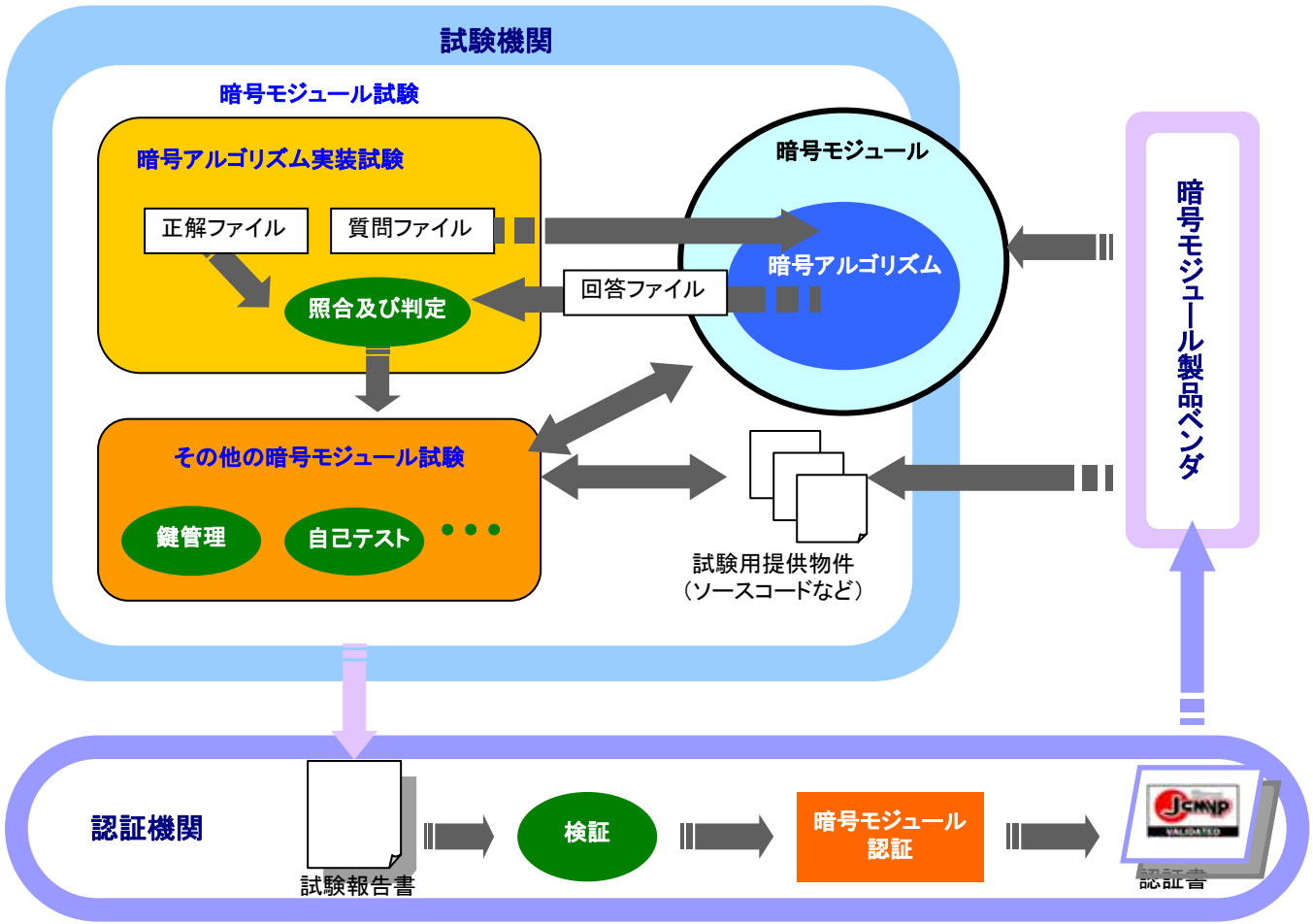
暗号アルゴリズム実装試験の有効性

- 北米での調査結果
  - 初期の332件の暗号アルゴリズム実装の調査結果(DES, Triple-DES, DSA, SHA1)
    - 88(26.5%)の実装にセキュリティ欠陥(*1)
  - 2007年7月から2009年5月期の報告では、暗号アルゴリズム実装の数%に不具合あり

■(*1)出典：CMVP FAQ
http://csrc.nist.gov/groups/STM/cmvp/documents/CMVPFAQ.pdf

暗号モジュール試験

暗号モジュール試験　＝　暗号アルゴリズム実装試験　＋　その他の暗号モジュール試験

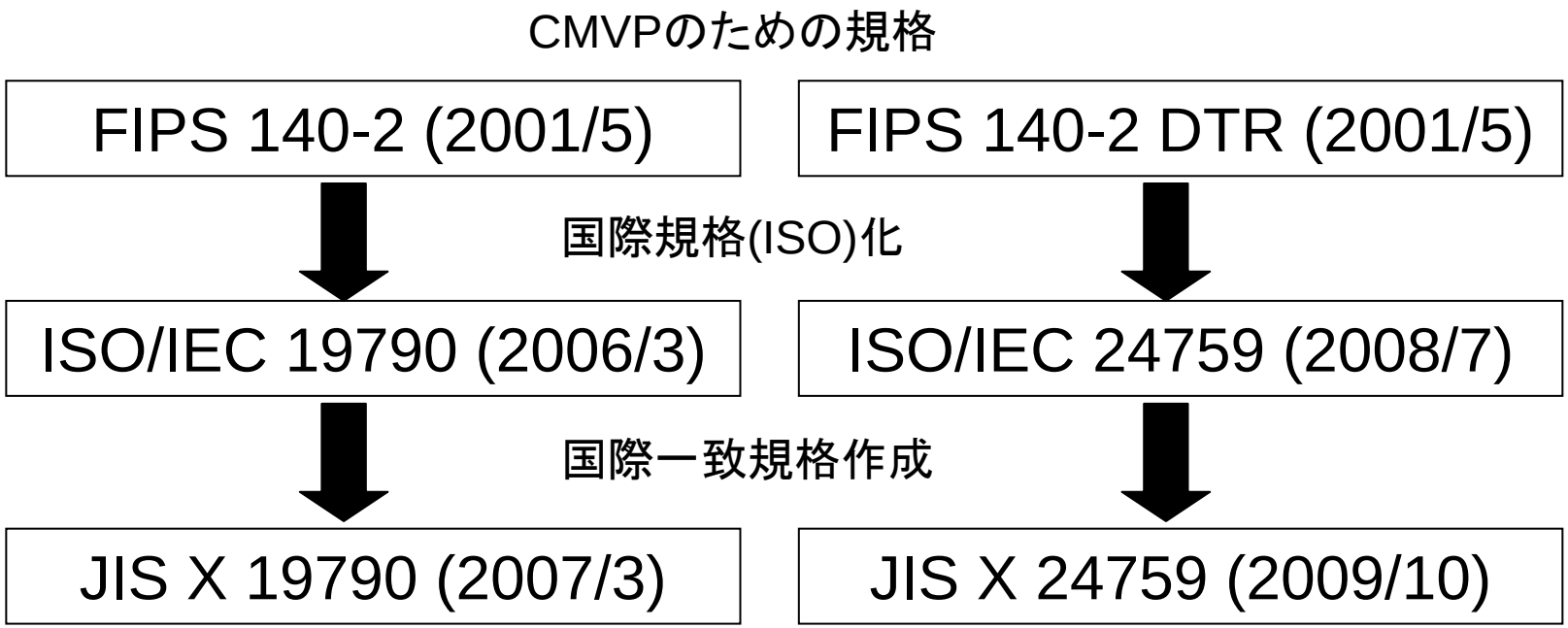


暗号モジュール試験内容

- **暗号アルゴリズム実装試験**
  - JCATTツールを用いたセキュリティ機能のブラックボックステスト
- **ドキュメントレビュー**
  - セキュリティポリシ
  - 有限状態モデル(FSM)
  - 設計資料
  - VEドキュメント(ベンダ情報要件の説明／参照先指定)
- **物理的セキュリティ試験**
  - タンパー証跡
  - 囲いの不透明性確認
  - タンパー検出・応答
- **ソースコードレビュー**
  - FSM・設計資料とソースコードの一致
- **オペレーションテスト**
  - FSMとの一致(エラー状態を含む)
  - ガイダンス文書との一致

標準化の歴史

セキュリティ要求事項　　セキュリティ試験要件



JIS X 19790
暗号モジュールセキュリティ要件要約

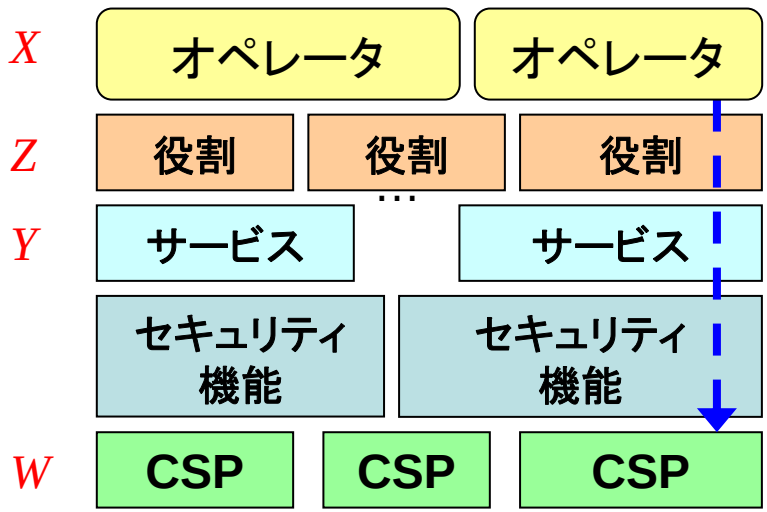
	セキュリティレベル1	セキュリティレベル2	セキュリティレベル3	セキュリティレベル4
暗号モジュールの仕様	暗号モジュール，暗号境界，承認されたアルゴリズム，承認された動作モードの仕様。すべてのハードウェア，ソフトウェア，ファームウェアの構成要素を含む暗号モジュールの記述。暗号モジュールのセキュリティポリシの宣言。			
暗号モジュールのポート及びインタフェース	必す（須）のインタフェース及び選択可能なインタフェース。すべてのインタフェースの仕様及びすべての入出力データパスの仕様。		他のデータポートから論理的に分離された，CSPのためのデータポート。	
役割，サービス，及び認証	必す（須）の役割及びサービスと選択可能な役割及びサービスとの論理的な分離。	役割ベースのオペレータ認証又はIDベースのオペレータ認証。	IDベースのオペレータ認証。	
有限状態モデル	有限状態モデルの仕様。必す（須）の状態及び選択可能な状態。状態遷移図及び状態遷移の仕様。			
物理的セキュリティ	製品グレードの装置。	錠又はタンパー証跡。	カバー及びドアに対してのタンパー検出及びタンパー応答。	タンパー検出及びタンパー応答が可能な包被。EFP又はEFT。
動作環境	単一のオペレータ。実行可能なコード。承認された完全性技術。	参照PPに適合し，EAL2の条件で評価を受けた環境。EAL2の条件で評価を受け，任意アクセス制御機構及び監査機構をもつ環境。	参照PPに加え，高信頼パスに適合し，EAL3に加え，セキュリティポリシのモデル化の条件で評価を受けた環境。	参照PPに加え，高信頼パスに適合し，EAL4の条件で評価を受けた環境。
暗号鍵管理	鍵管理機構：乱数生成及び鍵生成，鍵確立，鍵配送，鍵入出力，鍵の保管，並びに鍵のゼロ化。			
	手動の転送方法を用いて転送された秘密鍵及びプライベート鍵は，平文の形式で入力又は出力してもよい。		手動の転送方法を用いて転送された秘密鍵及びプライベート鍵は，暗号化又は知識分散処理を用いて，入力又は出力されなければならない。	
自己テスト	パワーアップ自己テスト：暗号アルゴリズムテスト，ソフトウェア/ファームウェア完全性テスト及び重要機能テスト。 条件自己テスト。			
設計保証	構成管理。セキュアな設置及び生成。設計とポリシとの対応。ガイダンス文書。	構成管理システム。セキュアな配送。機能仕様。	高級言語による実装。	セキュリティポリシのセキュリティルール、特徴を記述する形式的モデルの規定
その他の攻撃への対処	攻撃への対処の仕様。 現在は，試験可能な要求事項は用意されていない。			試験可能な要求事項を備えた，攻撃への対処の仕様。

暗号モジュール試験内容 –鍵管理–

- JIS X 19790の要求事項

  セキュリティポリシは、次の質問に十分答えられるように詳述する必要がある。

  「暗号モジュールに含まれるすべての役割，サービス及びセキュリティに関連するデータに対して，役割Zを担ってサービスYを実行しているオペレータXは，セキュリティに関連するデータ項目Wへのどのようなアクセスを行うか？」

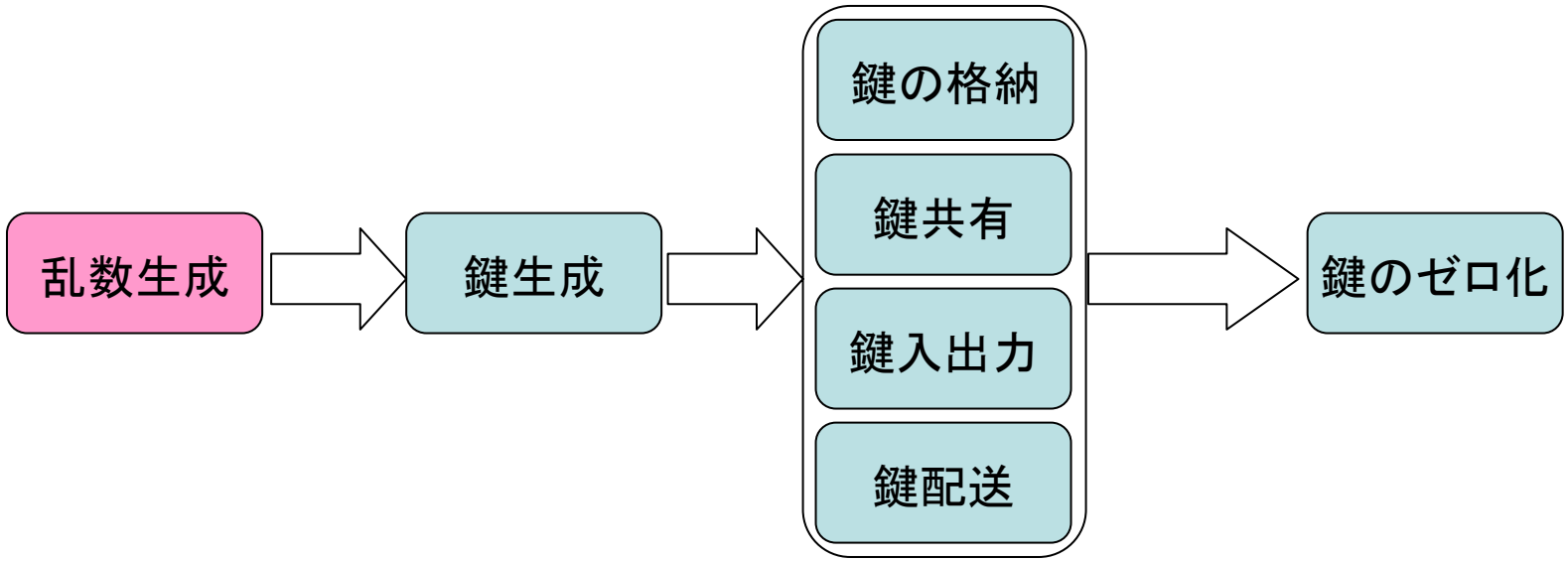


CSP (Critical Security Parameter) :

秘密又はプライベートセキュリティに関する情報であって，その開示又は変更が，暗号モジュールのセキュリティを危殆化し得るもの。

例 秘密鍵，プライベート鍵，認証データ

暗号モジュール試験内容 –鍵管理–



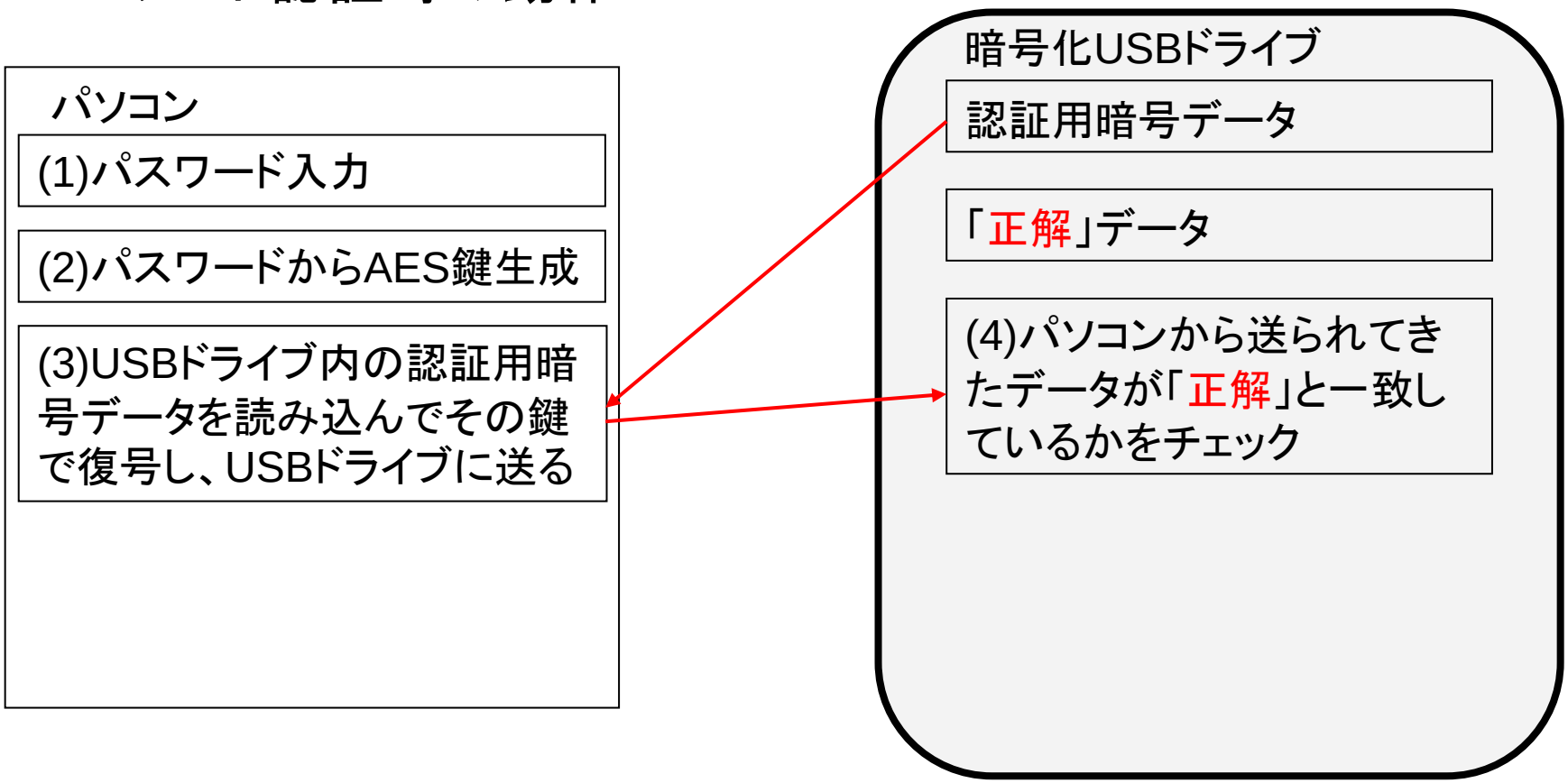
- 乱数生成は、暗号鍵管理の起点!

事例紹介1
暗号化USBドライブがクラックされた!

- この暗号化USBドライブでは、パスワードによる認証で暗号化USBドライブへのアクセスを制御している。
- ただし、認証方法は、パソコン上の管理ソフトウェアに依存している

事例紹介1
暗号化USBドライブがクラックされた!

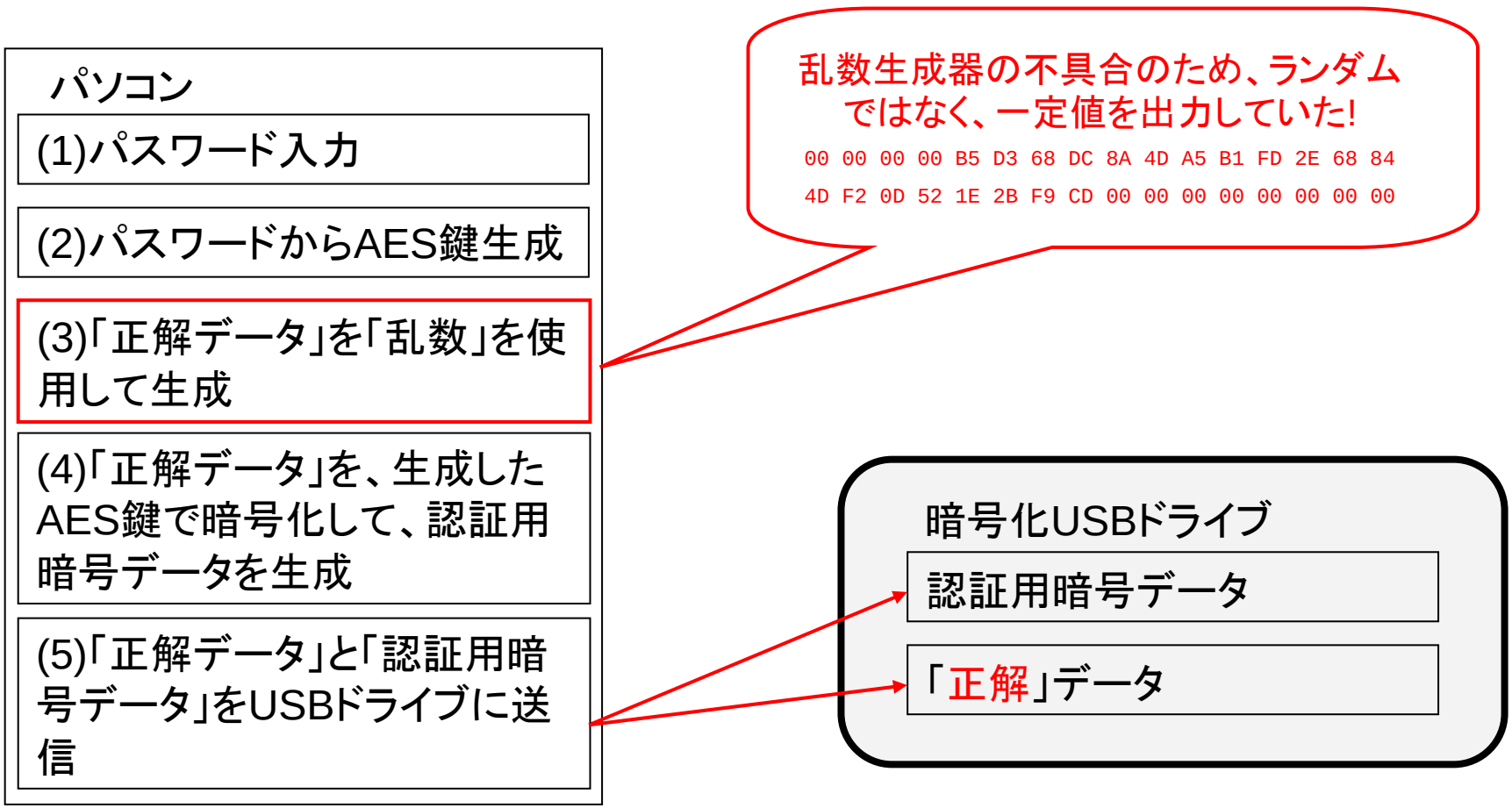
パスワード認証時の動作



出展:
http://www.syss.de/fileadmin/ressources/040_veroeffentlichungen/dokumente/SySS_Cracks_Kingston_USB_Flash_Drive.pdf

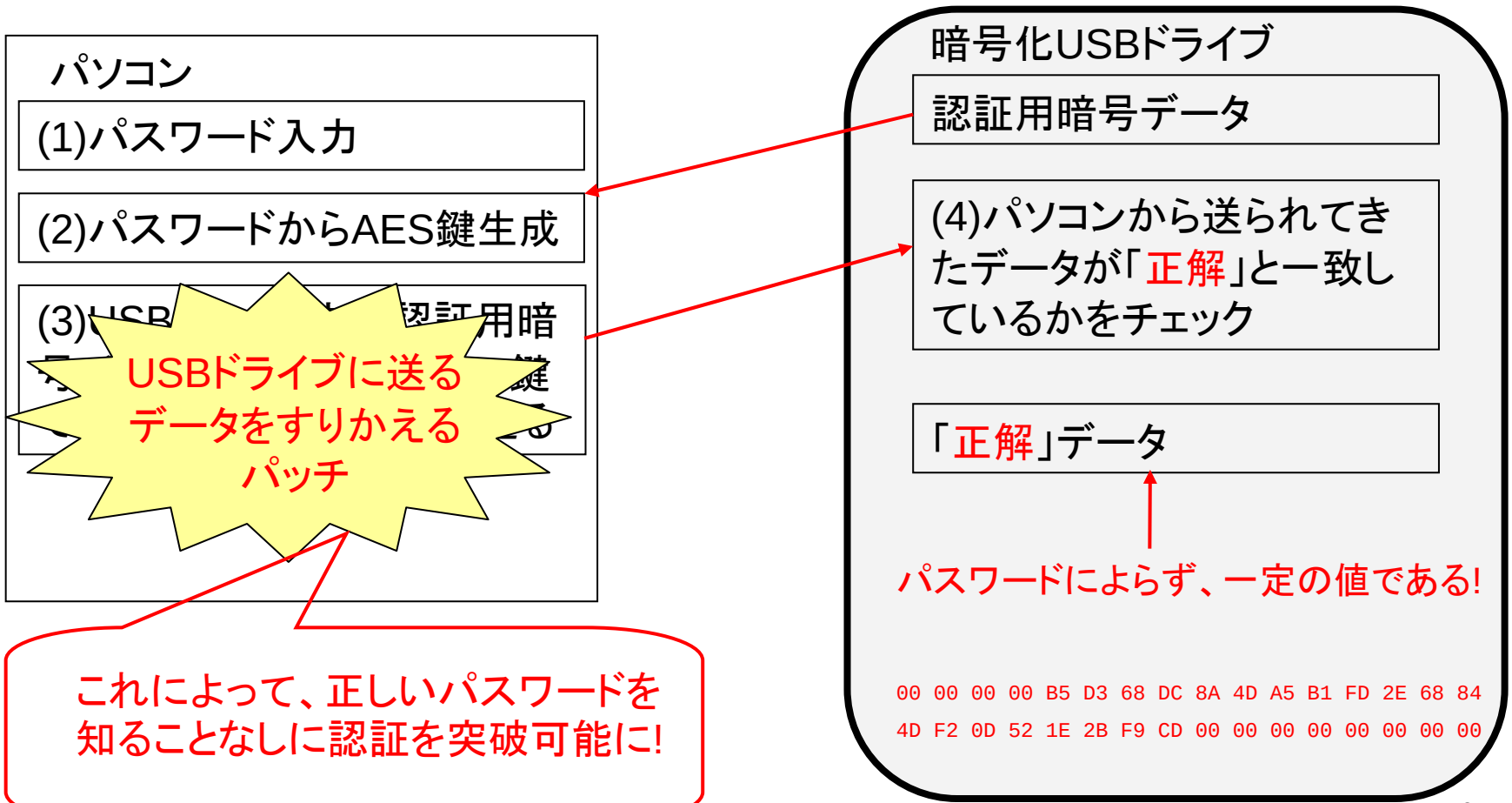
事例紹介1
暗号化USBドライブがクラックされた!

パスワード変更時の動作



事例紹介1
暗号化USBドライブがクラックされた!

認証回避!



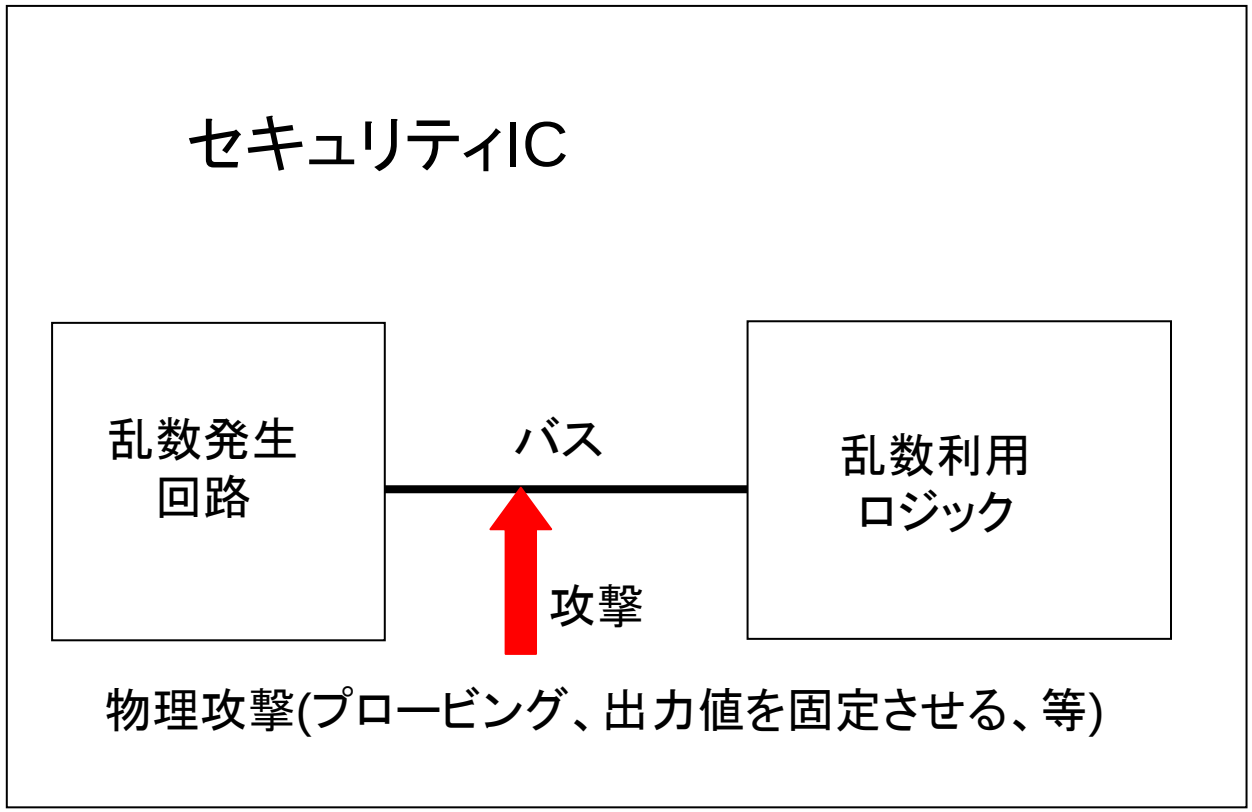
事例紹介2
Debian GNU/Linux の OpenSSL で発生した脆弱性

- OpenSSLをDebian用パッケージにする際、Valgrind(未初期化エリアへのメモリアクセス等を検出するソースコードの静的チェッカ)の警告を消すためにソースコードを修正した
- その修正のために、乱数生成器のシードへのエントロピープールからの入力がなくなった
- そのため、乱数が予測可能になり、その結果、生成される鍵も予測可能になってしまった

参考: CVE-2008-0166

(http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0166)

事例紹介3
セキュリティICへの物理攻撃



まとめ

- 乱数生成器は、重要なセキュリティ機能のひとつである
- 乱数生成器が出力する乱数の「ランダム性」は、セキュリティ強度に直接影響する
- 乱数生成器は、クラッカーにとっての有力な攻撃対象のひとつ
- JCMVPでの要求事項を満たす実装例は…第2部へ

乱数生成器に関する説明会

第2部：乱数生成器の要求事項

- 決定論的乱数生成器(DRBG)
- DRBGを使った暗号鍵生成
- ミニマムエントロピー
- NIST SP800-90とその要求事項を満たす実装例

決定論的乱数生成器(DRBG)とは

- 推定困難な初期値(乱数シード)を用いて、一様な分布でかつ推定困難なビット列を生成する機能ブロック
- 用途:暗号鍵生成、Initial Vector生成等
- DRBG : Deterministic Random Bit Generators

DRBGを使って、暗号鍵を作りたい(1)

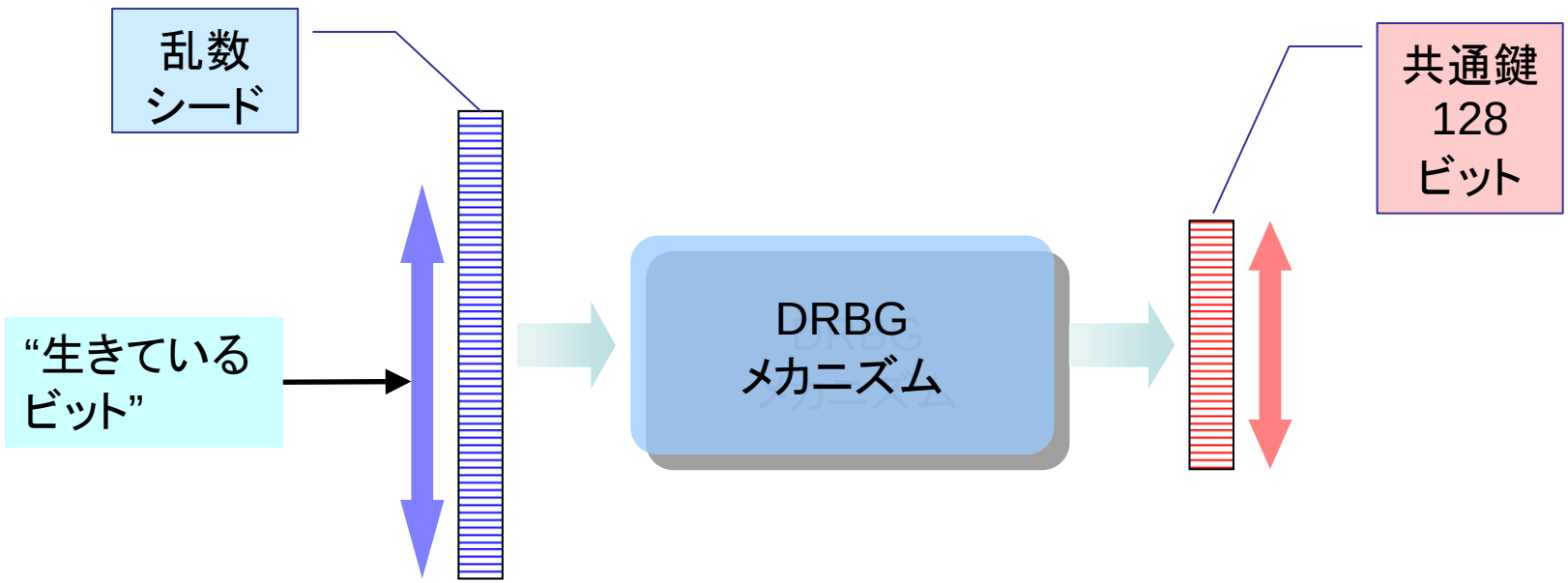
- 目的:
  - 128-bit ブロック暗号の128-bitの暗号鍵を生成したい。
- 考察:
  - 生成される暗号鍵が攻撃者に対して、推定困難でないと意味がない。
  - 生成される暗号鍵に偏りが無ければ、この暗号を危殆化するのに、およそ2^{128}回の計算が必要。(必要な計算量をビット数で表したものがセキュリティ強度)
  - もし、偏りがあったら、攻撃者としては、その偏りを利用した方が計算量が少なくて、簡単。

DRBGを使って、暗号鍵を作りたい(2)

- 目的を実現するために:
  - 暗号鍵が推定困難であるためには、
    - DRBGの内部状態、乱数シードは、機密情報として保護される必要がある。→DRBG機能境界
  - 暗号鍵が128-bit の共通鍵として意味があるためには、
    - DRBGの初期値(乱数シード)が取り得る値の数は、2^{128}以上でないと、意味がない。
    - DRBGの初期値(乱数シード)が取り得る値の出現頻度に偏りがあるならば、それも加味する必要がある。

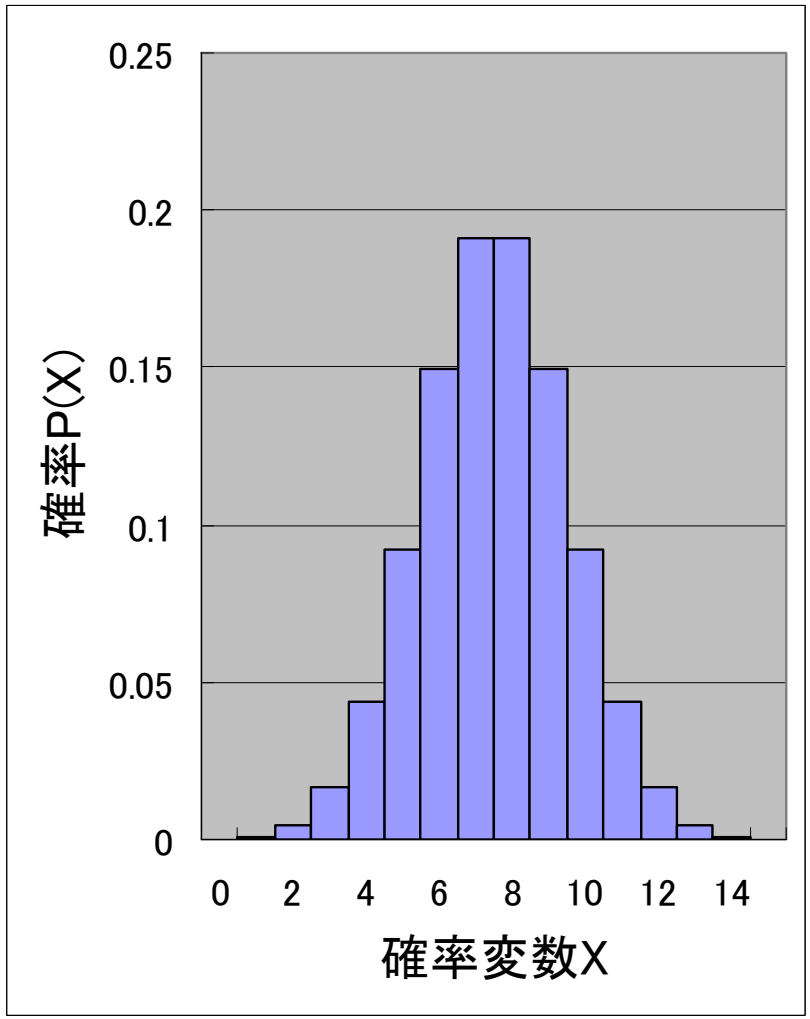
DRBGを使って、暗号鍵を作りたい(3)

- 乱数シードの素性を調べて、共通鍵の長さ以上の情報量を持っている乱数シードを持ってくる。→十分なエントロピー
- NIST SP800-90では、共通鍵の長さの1.5倍以上の情報量を推奨



乱数シードの統計分析(1)

- 4-bitの確率変数の出現頻度の例
  - 確率変数Xが取り得る値の総数は16。
  - 出現頻度には偏りがあり、X としては、7とか8が出現しやすい
  - 出現頻度を加味したら、どれくらいのビットが”生きているの”?



乱数シードの統計分析(2)

- 次の計算式に従って、Shannon エントロピー H_Sを計算する。

$$H_S = -\sum_{x \in \Omega_X} P(x) \log_2 (P(x))$$

  - Ω_X : 確率変数Xがとり得る値xの全体からなる集合
- 前のページの確率分布を使うと、H_S ≈3.1
  →平均的には、約3ビットの情報量。

※Shannon エントロピーは、情報量の平均

乱数シードの統計分析(3)

- 次の計算式に従って、ミニマム エントロピーH_{min}を計算する。

$$H_{\min} = -\log_2 P_{\max}$$

  - P_{max}：確率の最大値
- 2ページ前の確率分布を使うと、H_{min} ≈2.3
  →平均的には、約2ビットの情報量。

- ※ミニマムエントロピーは、名前の通り、最小の情報量。
  H_{min} <= H_Sである。
  (ISO/IEC 18031:2005, Annex G参照)

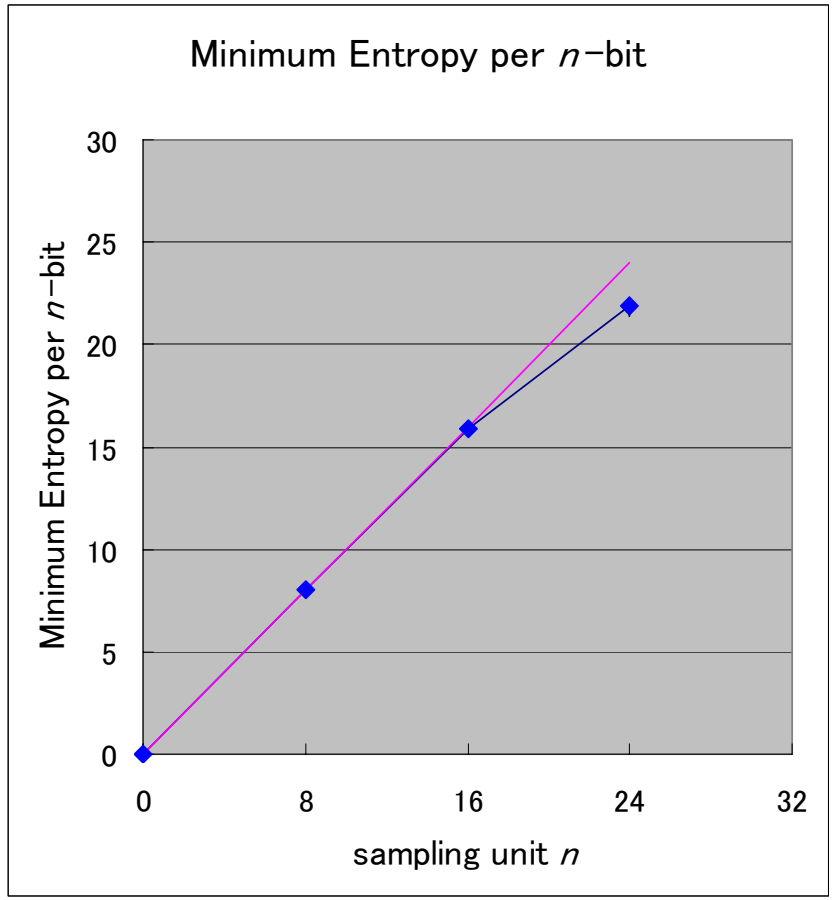
ミニマムエントロピーの推定(1)

- 1個の乱数シードの長さ(ビット)は、4に限定されない。
  →取得した乱数シードを連結して、そこからn-bitずつ取り出す
- ミニマムエントロピーを計算する際の n：8,16,…

- 有効数字について
  - 連結した乱数シード：256メガバイト(=2^{28}バイト)
  - n=8とすると、サンプル数Nは、2^{28}。
    - ヒストグラムの平均値は、$2^{28}/2^8=2^{20}=1024^2$。
      →確率の最大値P_{max}の有効数字は3桁以上。
  - n=16とすると、サンプル数Nは、2^{27}。
    - ヒストグラムの平均値は、$2^{27}/2^{16}=2^{11}=2048$。
      →確率の最大値P_{max}の有効数字は3桁。

ミニマムエントロピーの推定(2)

- 乱数シードに必要な長さ
  - 最小エントロピーを*n*についてプロットして外挿

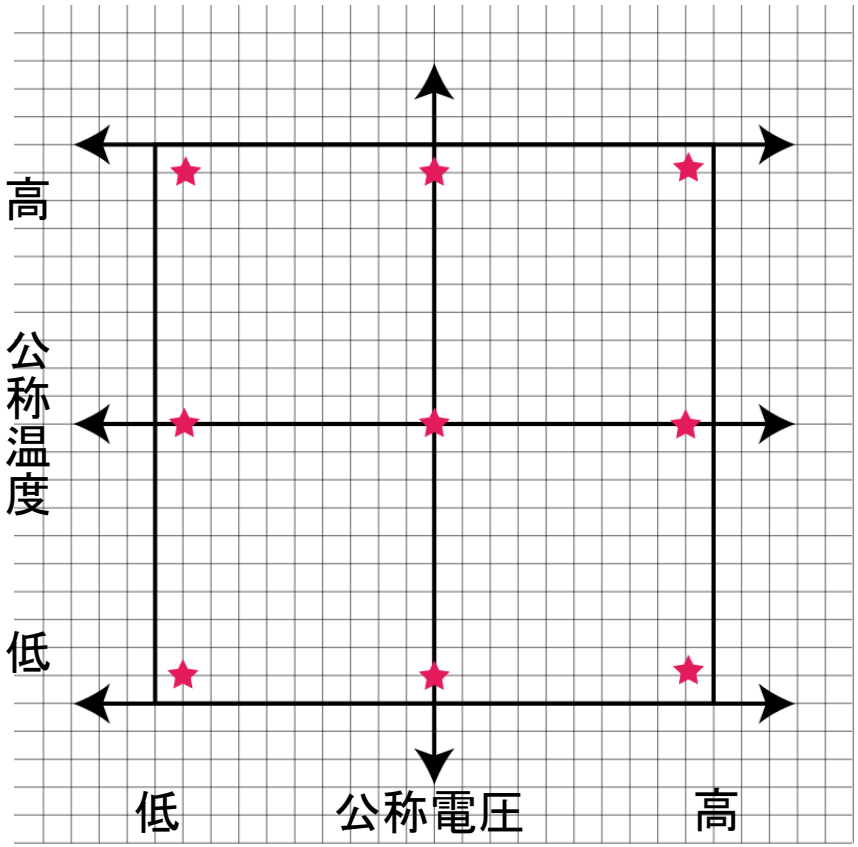


とあるエントロピー源の
ミニマムエントロピーの分析結果

ミニマムエントロピーの測定にあたって

- 動作範囲内で、エントロピー源に影響を与える環境条件を変化させて、ミニマムエントロピーを測定。
- 環境条件の例
  - 電源電圧
  - 温度等

- ※電源電圧、温度を変化させて得られた値の最小値が、ミニマムエントロピー

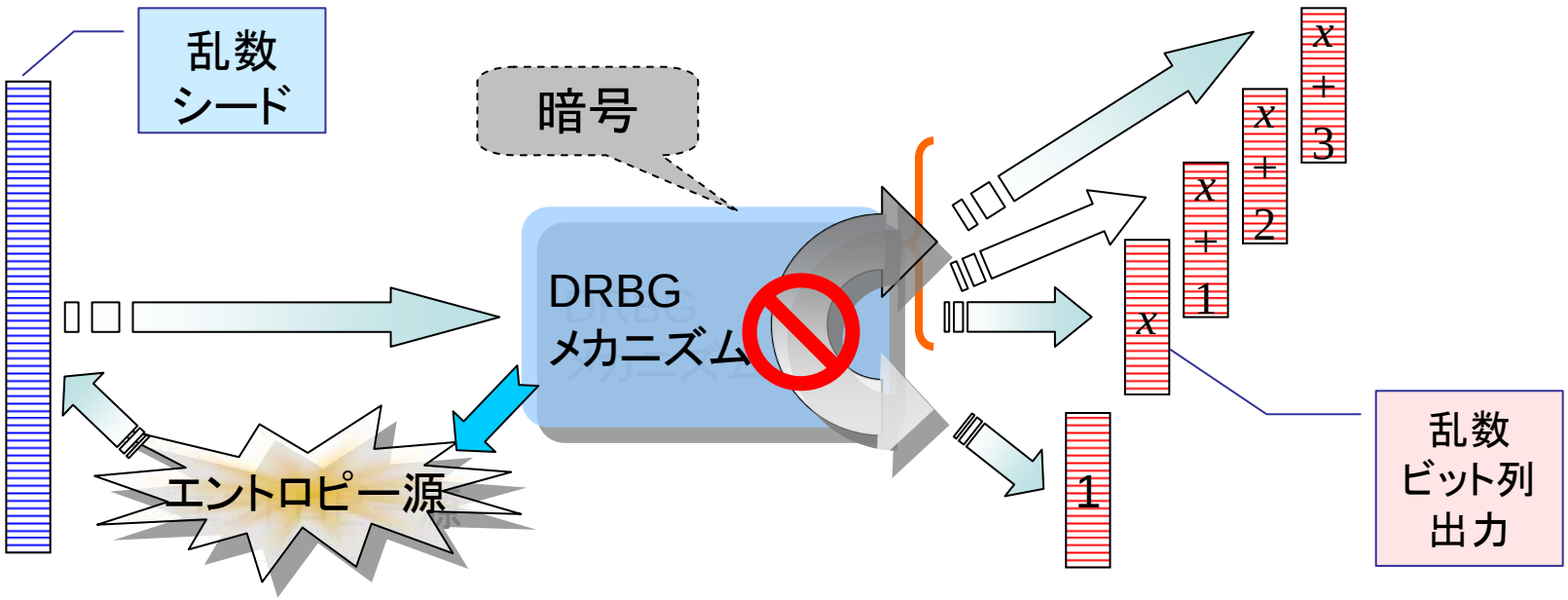


ミニマムエントロピーの推定(3)

- 参考資料
  - *Entropy and Entropy Sources in X9.82*, John Kelsey, NIST, July 2004 http://csrc.nist.gov/groups/ST/toolkit/documents/rng/EntropySources.pdf
  - *Testing Issues with OS-based Entropy Sources*, Peter Gutmann, University of Auckland http://csrc.nist.gov/groups/ST/toolkit/documents/rng/TestingOSSources.pdf
  - NIST Special Publication 800-22rev1, *A Statistical Test Suite for Random and Pseudorandom Number Generators* http://csrc.nist.gov/publications/nistpubs/800-22-rev1/SP800-22rev1.pdf

RBGと求められるセキュリティ特性

- 一度初期化したRBGを何度も使う
  - ある時点で乱数シードが危殆化しても、次に出力される乱数ビット列が予測できないようにしたい→prediction resistance
  - 定期的に、または必要に応じて、新しい情報を乱数シードに取り込む→reseeding
  - ある時点で乱数シードが危殆化しても、以前の乱数ビット列が推定できないようにしたい→backtracking resistance(→一方向性関数)



乱数生成器(RBG)の規格(1)

- ISO/IEC 18031, *Information technology — Security techniques — Random bit generation*
  - ANS X9.82のDraftをもとに、AIS20及びAIS31の内容を取り込んで作成
  - 対象とする乱数生成器
    - 非決定論的乱数生成器
      (NRBG : Non-deterministic Random Bit Generators)
    - 決定論的乱数生成器(DRBG)
  - JIS X 19790から参照
  - →ANS X9.82, NIST SP800-90との整合を取るべく改訂作業中

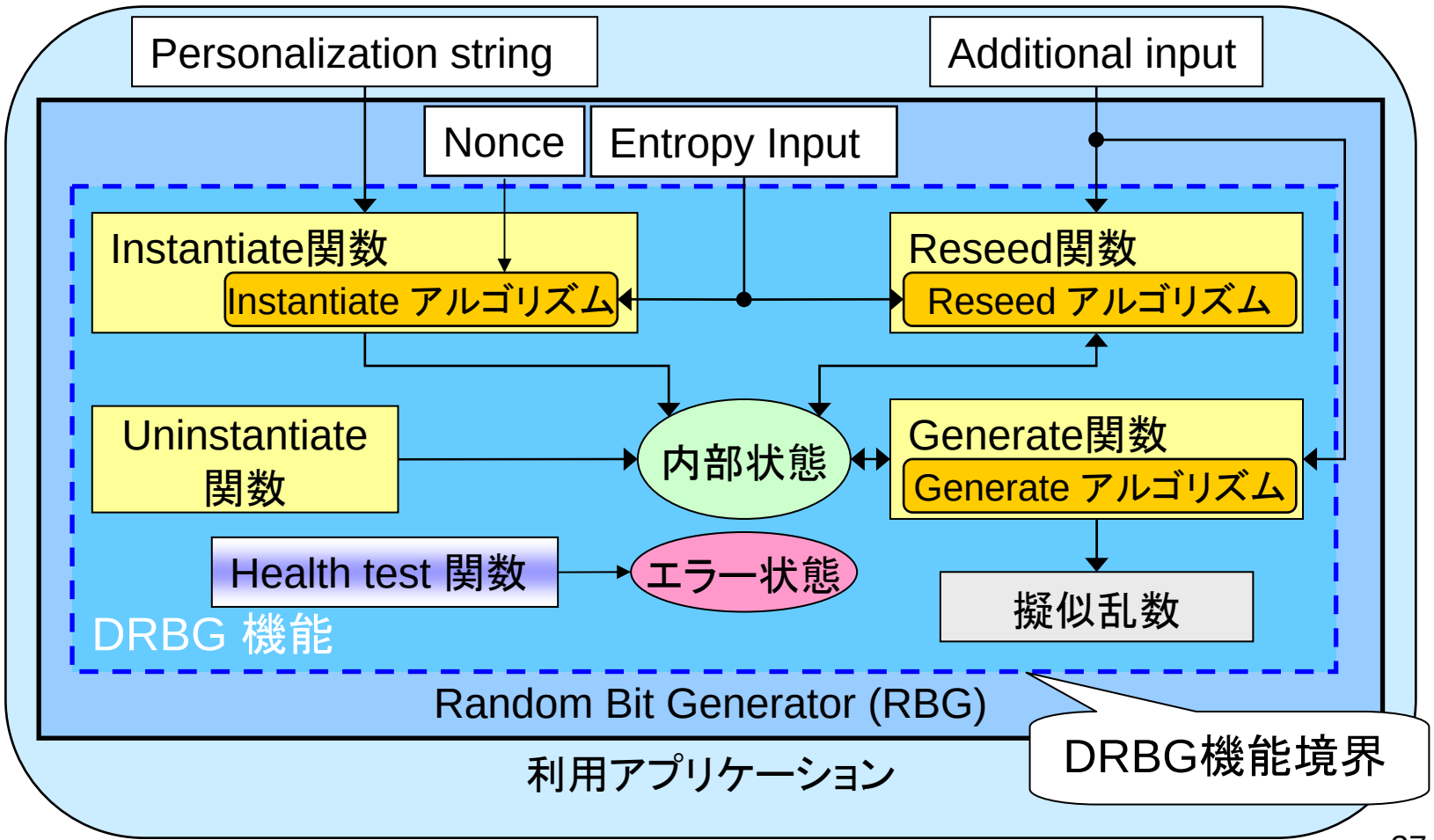
- ANS X9.82, *Random Number Generation*
  →Part1:*Overview and Basic Principles*, Part3:*Deterministic Random Bit Generators*は公開されたが、Part2:*Entropy Sources*が未発行
- AIS20, *Functionality classes and evaluation methodology for deterministic random number generators*
- AIS31, *Functionality classes and evaluation methodology for true (physical) random number generators*
  →統計テストは、FIPS 140-1から継承

乱数生成器(RBG)の規格(2)

- NIST SP800-90
  - 対象とする乱数生成器
    - 決定論的乱数生成器(DRBG)
  - ISO/IEC 18031の要求事項を満たす一例
  - ANS X9.82:Part3のDraftをもとに作成。
    内容は、ANS X9.82:Part3とほとんど同じ。
    - ※Hash_DRBGは、NIST SP800-90には掲載されているが、
      ANS X9.82:Part3には掲載されていない。
      Hash_DRBGは、ANSI-approved な乱数生成器が必要な場合(
      例えば、ANS X9.62:2005に基づくECDSA)には、使えない。
- http://csrc.nist.gov/publications/nistpubs/800-90/SP800-90revised_March2007.pdf

NIST SP800-90に基づくDRBG(1)

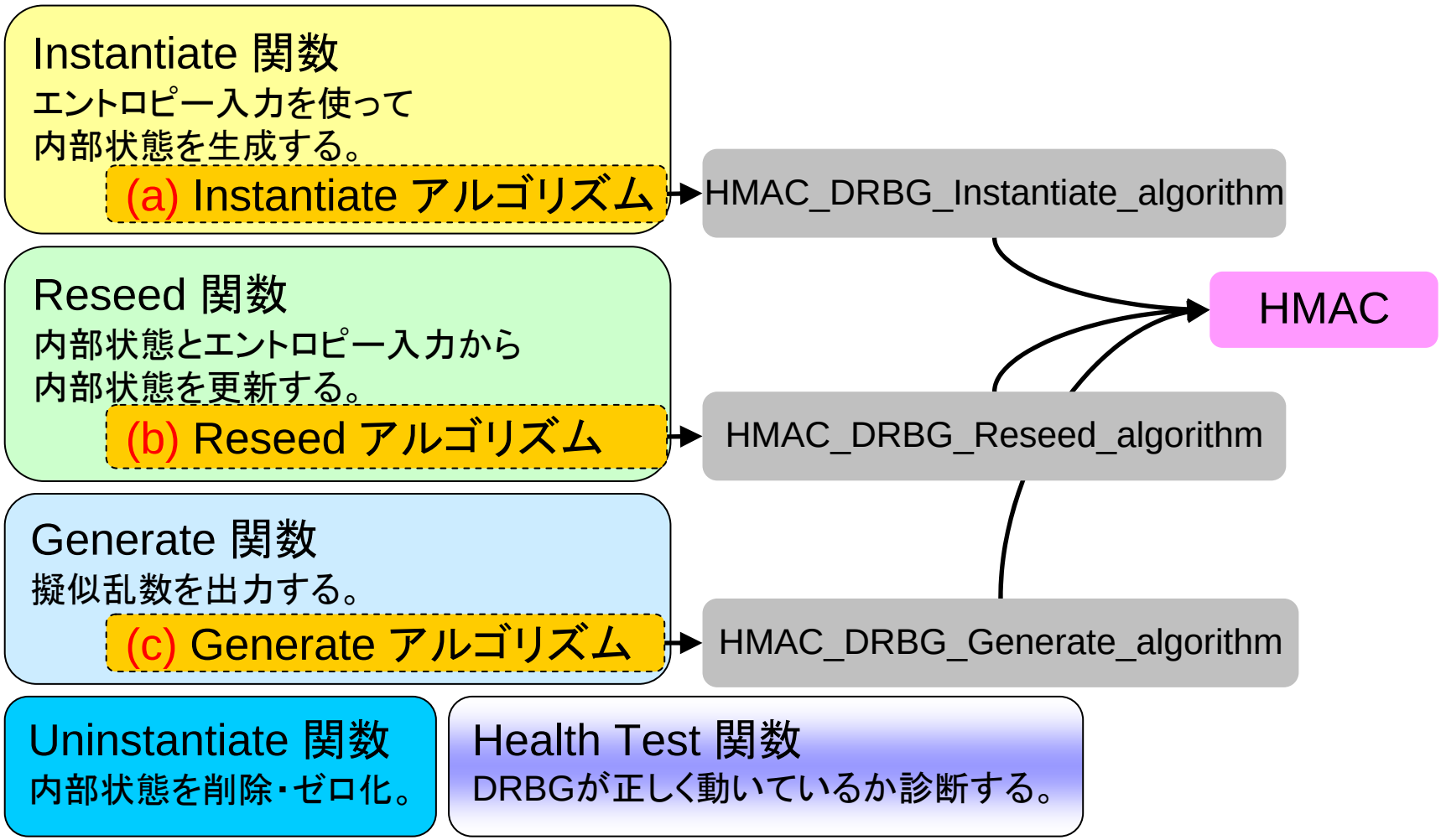
- 基本構成



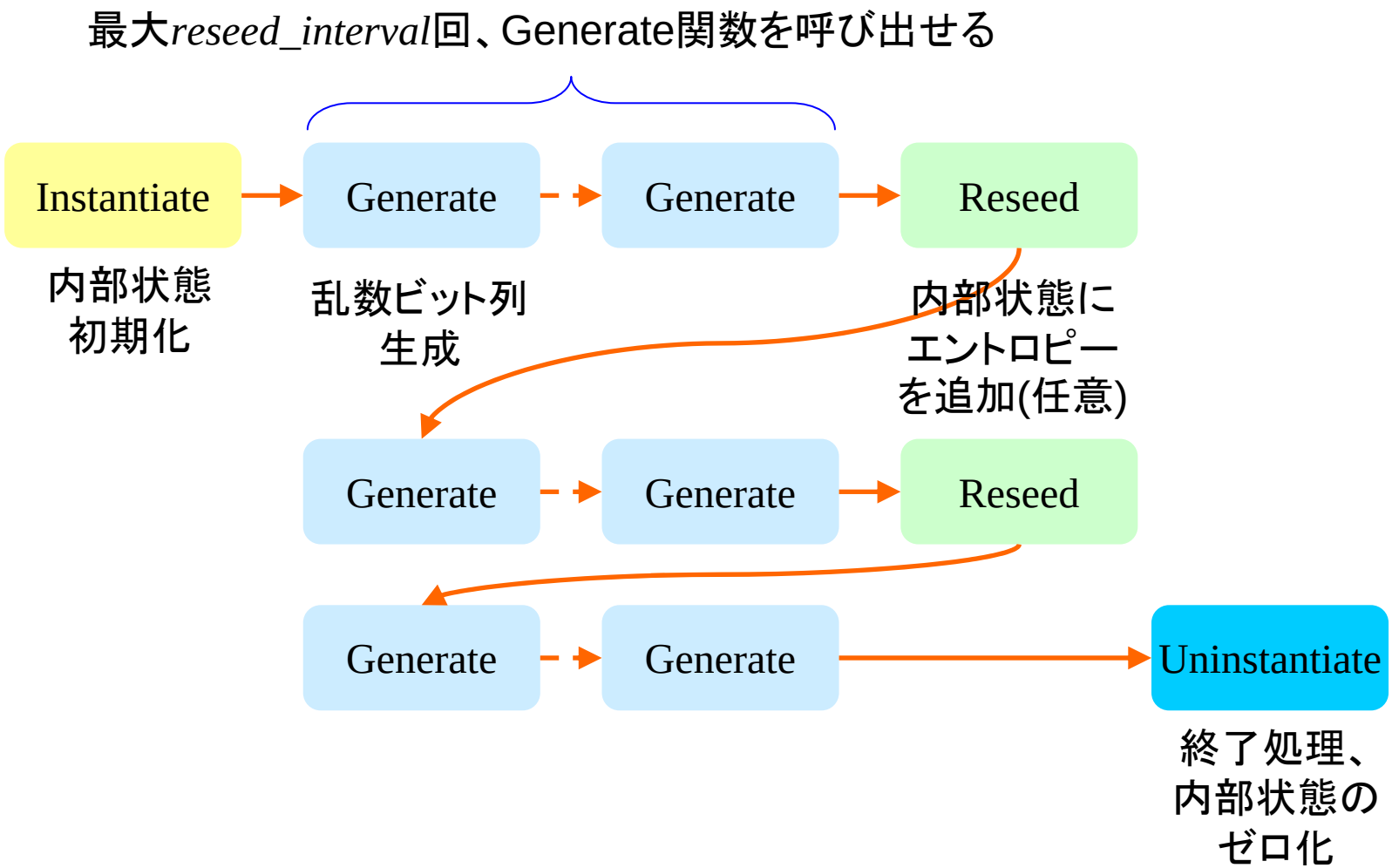
NIST SP800-90に基づくDRBG(2)

- DRBG機能境界
  DRBG機能とそれ以外を分ける境界。境界内に必ずHealth Test関数を含む。
- Health Test関数
  DRBG機能関数群(Instantiate, Reseed, Generate, Uninstantiate)が正しく動いているか診断する。テストに失敗した場合は、DRBG機能境界からの出力抑止。
- 内部状態
  擬似乱数を生成するための元データ(Seed等)を格納する。
- エントロピー入力
  擬似乱数を生成するためのSeedを作る元になる。
- nonce
  エントロピー入力とあわせてSeedを作る元になる。
- additional input
  Reseed関数及びGenerate関数が呼ばれる場合に使用される。
- personalization string
  DRBGのインスタンスを特徴付ける文字列。

NIST SP800-90に基づくDRBG
DRBG機能の階層構造(例:HMAC_DRBG)



NIST SP800-90
DRBGインスタンスのライフサイクル

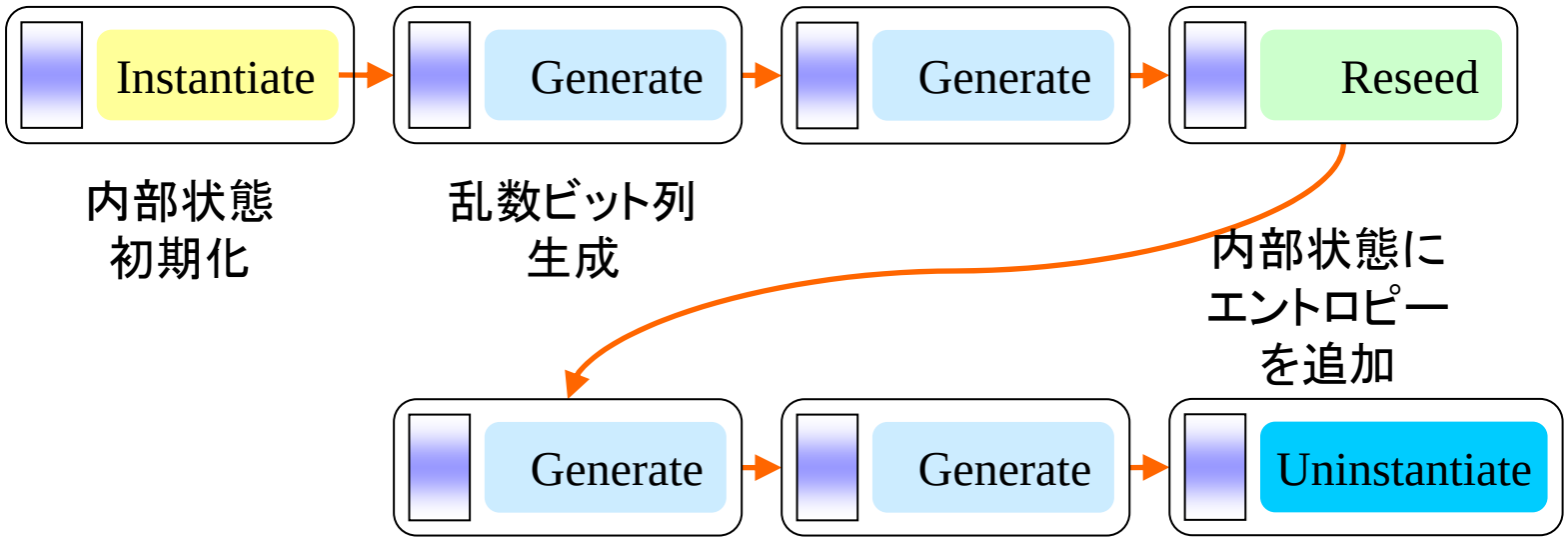


暗号モジュールのセキュリティ要求事項との関係

- JIS X 19790
  - ISO/IEC 18031準拠の乱数生成器のみが使用可能。JCMVPでは、NIST SP800-90記載の次の乱数生成器のみ。
    Hash_DRBG、HMAC_DRBG、CTR_DRBG
- 米国で策定中のFIPS 140-3
  - 特徴:
    - 乱数生成器:NIST SP800-90のみを参照。
    - 暗号アルゴリズム実装の既知解テストを、その暗号を使う直前までに行えば良い(”条件自己テスト”)
      → NIST SP800-90のHealth Testに関する要求事項とも整合。
    - FIPS 140-2に記載されていた連続RBGテストは、削除されている。
- http://csrc.nist.gov/publications/drafts/fips140-3/revised-draft-fips140-3_PDF-zip_document-annexA-to-annexG.zip

NIST SP800-90
DRBGインスタンスのライフサイクルとヘルステスト

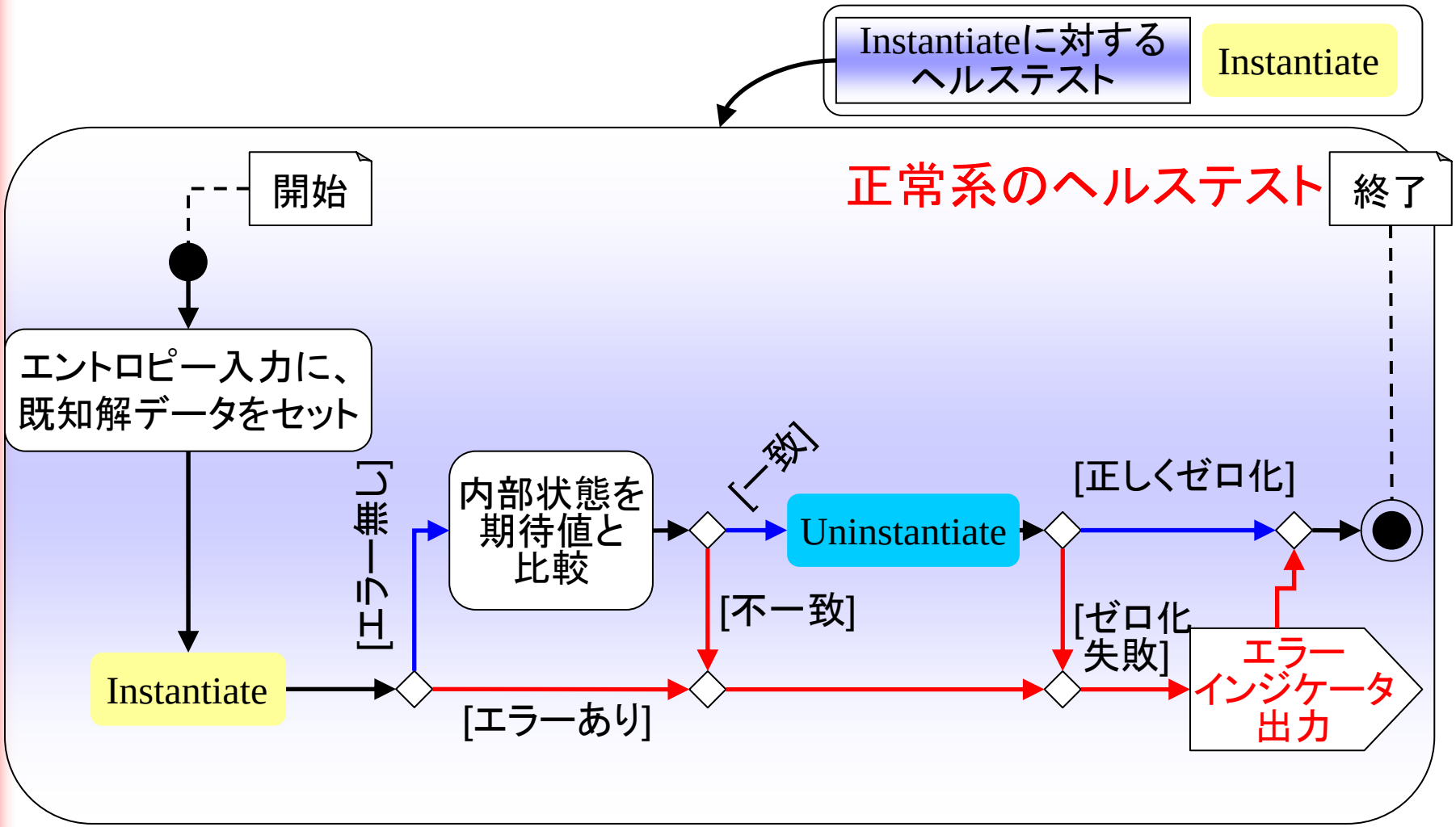
- 要求事項を満たすDRBGの実装例



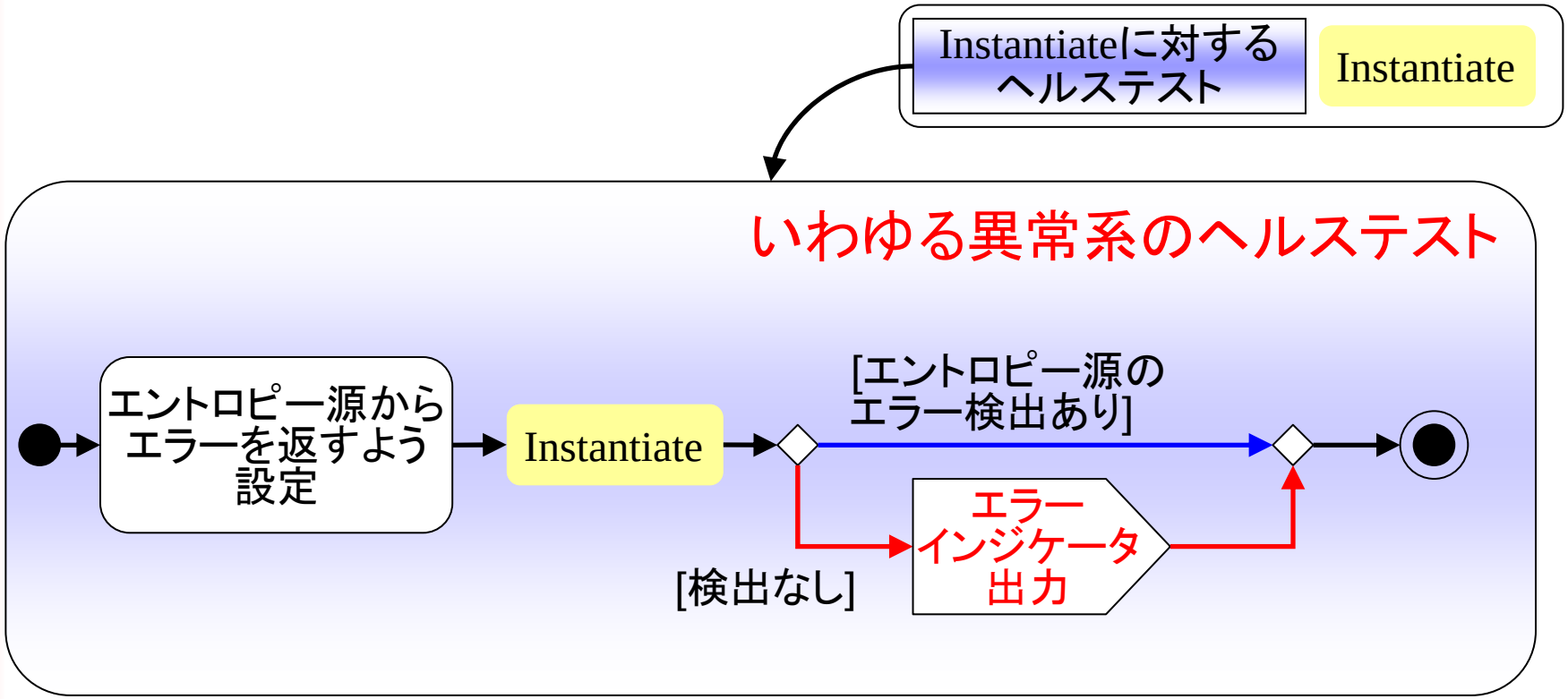
※DRBG機能関数を使う前に、
ヘルステストで確認する。

: ヘルステスト

NIST SP800-90
Instantiate関数に対するヘルステスト(1)

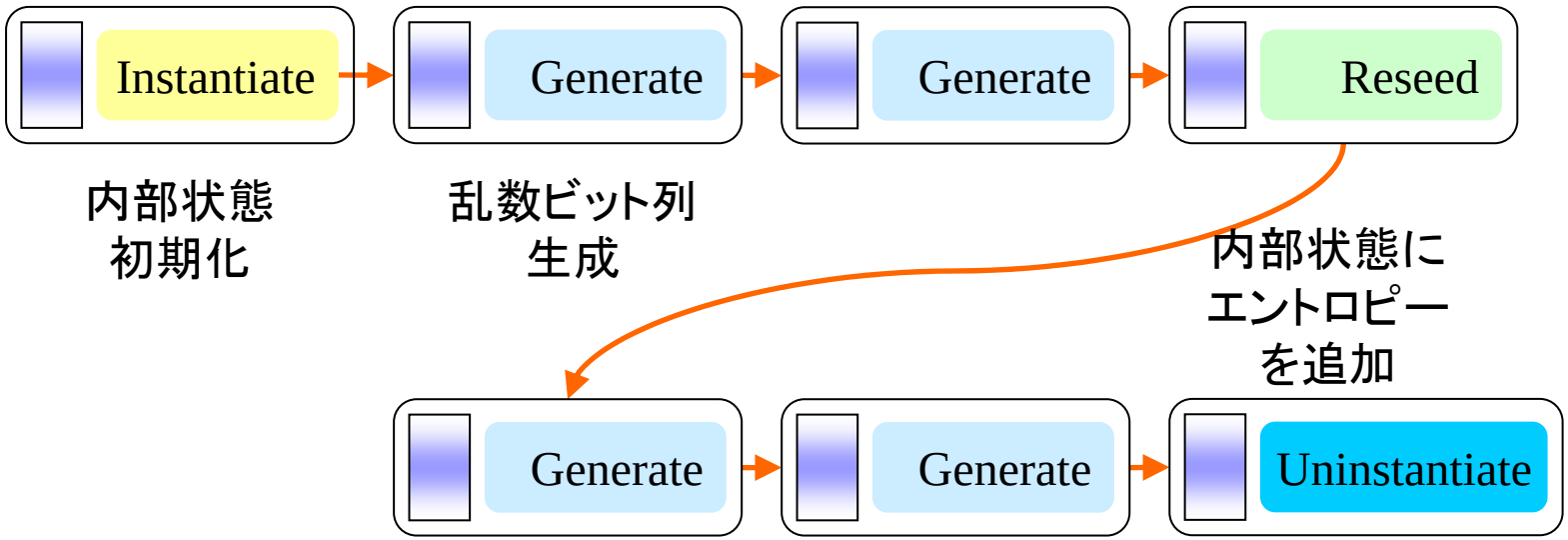


NIST SP800-90
Instantiate関数に対するヘルステスト(2)



NIST SP800-90
DRBGインスタンスのライフサイクルとヘルステスト

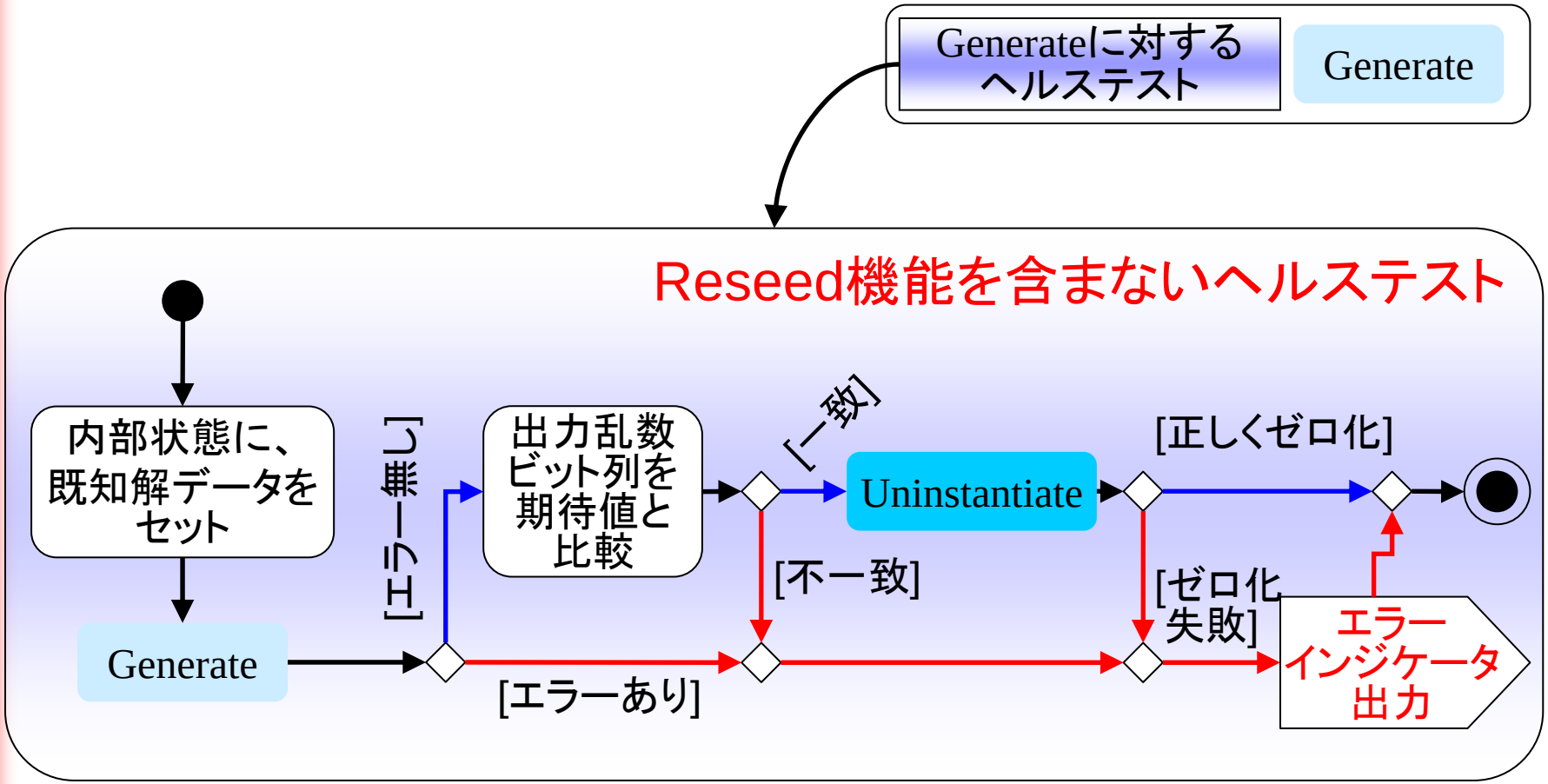
- 要求事項を満たすDRBGの実装例



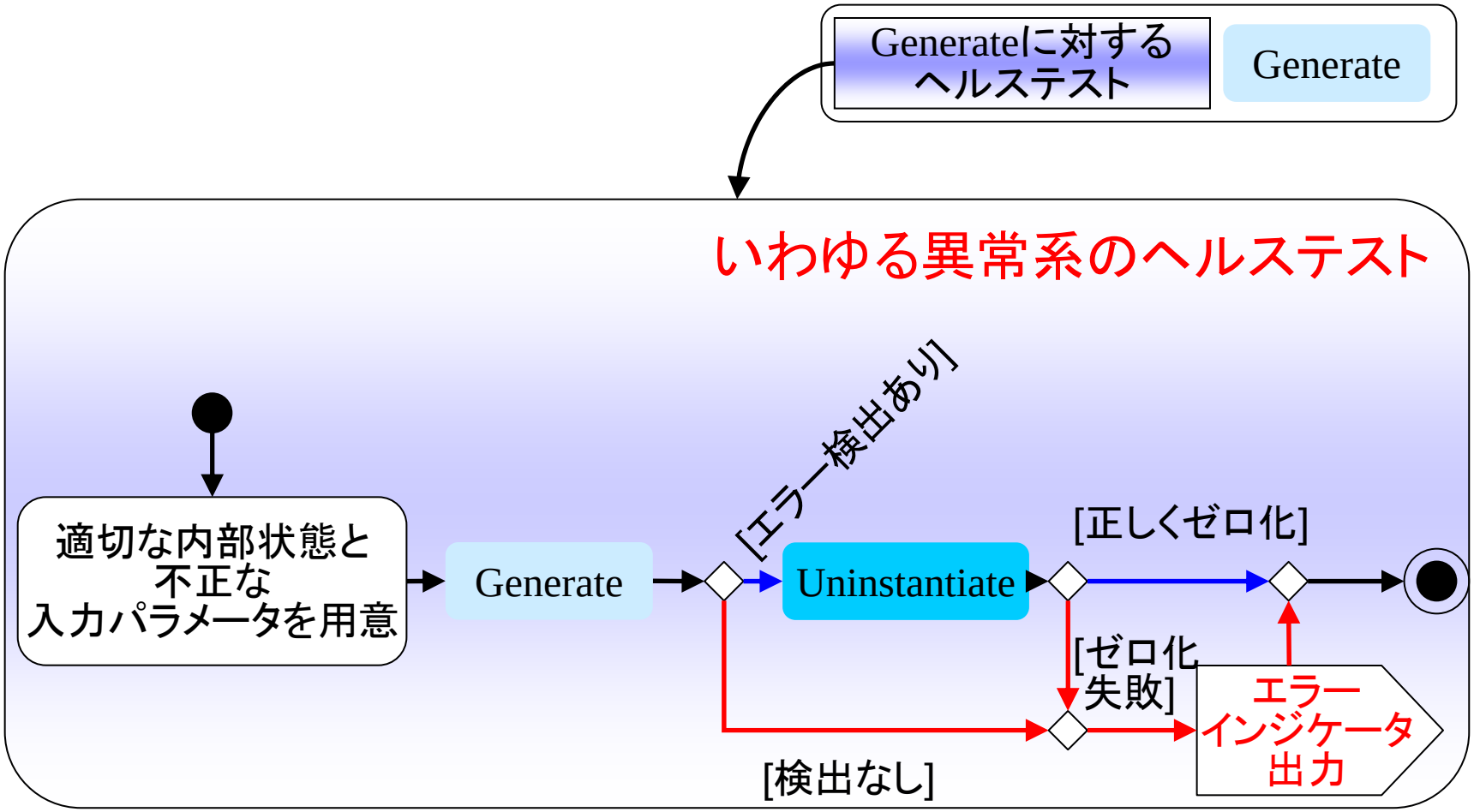
※DRBG機能関数を使う前に、
ヘルステストで確認する。

: ヘルステスト

NIST SP800-90
Generate関数に対するヘルステスト(1)



NIST SP800-90
Generate関数に対するヘルステスト(2)



NIST SP800-90
ヘルステストのエラー時の動作

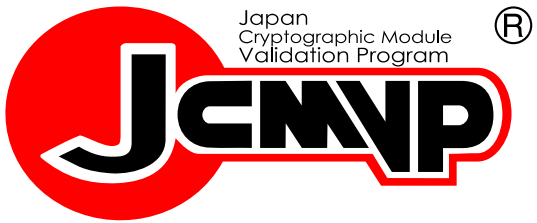
- ヘルステストのエラー:
  - 正しいDRBG機能の動作を保証できない
    →内部状態等を誤って出力してしまうかもしれない
  - DRBGがエラー状態にある時は、次の事項が求められる
    。
    - DRBG機能の実行の抑止
    - DRBG機能境界からの「乱数ビット列出力の禁止」

  - DRBG自体にFIPS 140-2, JIS X 19790と同様の要求事項

NIST SP800-90の試験内容のまとめ

- ツールによる正常系の動作確認
- ソースコードレビューによる実装確認
  - ヘルステスト…条件自己テスト
    - 正常系
    - 異常系
  - エラー状態
    - 実行の抑止
    - 出力の禁止
- 文書レビュー
  - 乱数シードのエントロピーが十分である事の文書化
    要求事項の数は、122個(7章から11章まで)

ご清聴有難うございました。



JCMVPホームページ

http://www.ipa.go.jp/security/jcmvp/

お問い合わせ先

jcmvp-info@ipa.go.jp