NIST Special Publication 800-130

A Framework for Designing Cryptographyc Key Management Systems

Elaine Barker
Miles Smid
Dennis Branstad
Santosh Chokhani

http://dx.doi.org/10.6028/NIST.SP.800-130

コンピュータ セキュリティ

National Institute of Standards and Technology
U.S. Department of Commerce

この文書は以下の団体によって翻訳監修されています

IPA 独立行政法人 情報処理推進機構
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

NIST Special Publication 800-130

A Framework for Designing Cryptographic Key Management Systems

Elaine Barker
Computer Security Division
Information Technology Laboratory

Miles Smid
Orion Security Solutions
Silver, Spring, MD

Dennis Branstad
NIST Consultant
Austin, TX

Santosh Chokhani
Cygnacom
McLean, VA

http://dx.doi.org/10.6028/NIST.SP.800-130

2013 年 8 月

米国商務省
Penny Pritzker、*長官*

米国国立標準技術研究所
Patrick D. Gallagher, Under Secretary of Commerce for Standards and Technology and Director

発行機関

本文書は、米国国立標準技術研究所（NIST：National Institute of Standards and Technology、以下、NIST と称す）によって、連邦情報セキュリティマネジメント法（FISMA：Federal Information Security Management Act)、公法（P.L.）107-347 に基づく法的責任を推進するために策定された。NIST は、連邦情報システムの最小限の要求事項を含め情報セキュリティ標準およびガイドラインを開発する責務があるが、このような標準およびガイドラインは国家安全保障に適用されてはならず、このようなシステムについての政策的権限を有する適切な連邦機関の明確な承認が必要となる。このガイドラインは、行政管理予算局（OMB：Office of Management and Budget）による通達（Circular）A-130Section 8b(3), Securing Agency Information Systems, as analyzed in Circular A-130, Appendix IV: Analysis of Key Sections 要求事項に一致している。また、補足情報は Circular A-130, Appendix III, Security of Federal Automated Information Resources で提供されている。.

本文書における一切は、商務長官が法的権威に基づき連邦政府に対して義務および拘束力を与えた標準およびガイドラインを否定するものではない。また、これらのガイドラインは、商務長官、行政管理予算局長、又は他の全ての連邦政府当局者の既存の権威に変更を加えたり、これらに取って代わるものと解釈したりしてはならない。本文書は、非政府組織が自由意思で使用することもでき、米国における著作権の制約はないが、NIST に帰属する。

National Institute of Standards and Technology Special Publication 800-130
Natl. Inst. Stand. Technol. Spec. Publ. 800-130, 112 pages (August 2013)
CODEN: NSPUE2

http://dx.doi.org/10.6028/NIST.SP.800-130

本文書中で特定される商業的組織、装置、資料は、実験手順または概念を適切に説明するためのものである。このような特定は、NIST による推奨または同意を意味するものではなく、これらの組織、資料、または装置が、その目的ために利用可能な最善のものであることを意味している訳ではない。

与えられた法的責任に従い、NIST によって現在開発中のその他の文書への参照が本文書にあるかもしれない。本文書におけるその情報は、概念および方法論を含め、このような関連文書の完成前であっても連邦政府によって利用されるかもしれない。したがって、それぞれの文書が完成されるまで、現在の要求事項、ガイドライン、および手順は存在する限り運用の効力を有する。計画および移行目的に関して、連邦政府は、NIST によるこれらの新しい文書の開発に密接に従うことを希望するかもしれない。

公開コメント期間中に各組織は全てのドラフト文書をレビューし、NIST へフィードバックを提供するよう奨励する。上記以外の全ての NIST コンピュータセキュリティ部門の文書は、http://csrc.nist.gov/publications から入手可能である。

この文書へのコメントは以下に提出してもよい：

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive（Mail Stop 8930）Gaithersburg, MD 20899-8930
Email: ckmsdesignframework@nist.gov

コンピュータシステムの技術に関する報告書

米国国立標準技術研究所（NIST：National Institute of Standards and Technology、以下、NISTと称す）の情報技術ラボラトリ（ITL：Information Technology Laboratory、以下、ITLと称す）は、国家の計測および標準に関する基盤において技術的リーダーシップを提供することにより、米国の経済および社会福祉に貢献している。ITLは、テストの開発、テスト技法の開発、参照データの作成、概念実証の実施および技術的分析を通じて、情報技術の開発と生産的利用の発展に努めている。ITLの責務は、連邦政府の情報システムにおいて、国家安全保障に関連する情報以外の情報に対する費用対効果の高いセキュリティとプライバシーを実現するための、技術面、物理面、管理面及び運用面での標準およびガイドラインを策定することを含んでいる。本Special Publication 800シリーズは、情報システムセキュリティに関するITLの調査、ガイドラインおよび公共福祉のために教育や援助を行う努力、ならびに産業界、政府機関および学術機関との共同活動について報告する。

要旨

この暗号鍵管理システム（CKMS）設計のフレームワークは、CKMS設計者がCKMS設計仕様を開発する際に考慮すべきトピックを含んでいる。それぞれのトピックごとに、設計仕様に記載する必要がある1件以上の文書か要求事項が存在する。そのため、これら全ての要求事項に対処しているいかなるCKMSも本フレームワークに適合する設計仕様を備えることになる。

キーワード

アクセスコントロール；機密性；暗号鍵管理システム；暗号鍵；フレームワーク；完全性；鍵管理ポリシー；鍵メタデータ；ソース認証（source authetication）

謝辞

米国国立標準技術研究所（NIST）は、本推奨事項に関連する多くのセキュリティ課題に関して、本フレームワークの作成、レビュー、公開に参画した全ての関係者による貢献に深く感謝の意を表する。NISTはまた、公的機関ならびに民間の方々による、本刊行物の品質および有用性を向上させる思慮深く建設的なコメントをいただいたことに感謝する。NISTで2009年、2010年及び2012年に開催されたワークショップにおいてなされた多くの有用な提案がこの文書に取り込まれている。

目次

図の一覧

表の一覧

1　序説

本フレームワークは暗号鍵管理システム（CKMS[1]）設計のためのものであり、CKMS 設計時に考慮すべきトピック及び記載する文書化要求事項（以下、「要求事項」と呼ぶ）を記述したものである。CKMS 設計者は、CKMS に組み込むポリシー、手続き、コンポーネント（ハードウェア、ソフトウェア、ファームウェア）、デバイス（コンポーネントのグループ）を選択し、これらの項目を本フレームワークの要求事項に適合するためにどのように用いるかを明記することによって要求事項を満たす。

CKMS は、暗号鍵及びある種の特有の情報（ここでは（関連付けられた）メタデータと呼ぶ）を保護、管理及び配付するために使用されるポリシー、手続き、コンポーネント及びデバイスから構成される。CKMS は、暗号化されていない鍵又はそのメタデータにアクセスできる全てのデバイス又はサブシステムを含む。暗号化された鍵とそれらの暗号学的に保護された（鍵と結び付けられた）メタデータはコンピュータで扱われ、通信システムを通して伝送され、CKMS の一部とは見なされないメディアに保管されることがある。

本 CKMS フレームワークは、あらゆる CKMS のための設計文書要求事項を提供する。言い換えれば、CKMS の設計において文書化する必要があるものを記述する。本フレームワークの目標は、CKMS 設計者が望む CKMS を構築、調達及び評価するために使用できる、完全かつ統一的な CKMS 仕様の作成をガイドすることである。

本フレームワークは、以下の利点を提供する：

a) 重要な CKMS の機能の仕様を要求することで CKMS 設計タスクを定義する助けになる。

b) CKMS 設計者に CKMS 全体に必要な要素を考慮するように促す。

c) CKMS 設計者に、適切に対処されていれば、CKMS にセキュリティを提供できる要素とメカニズムを考慮するように促す。

d) 異なる複数の適合 CKMS 及びその機能について論理的に比較するために使用できる。

e) CKMS の機能とその実装についての仕様を要求することで、セキュリティアセスメントを実行する助けになる。

f) 米国連邦 CKMS プロファイルの基礎を形成する。

本フレームワークにおいて、NIST 標準と Special Publications は例示としてのみの参照である。本フレームワークは、合理的に完全かつ優れた設計がされたあらゆる CKMS を包含するような十分に一般的なものとなることを意図している。

本フレームワークは CKMS 設計についての記載となることを意図していない。その CKMS の設計タスクは CKMS 設計者に委ねられている。むしろ、本フレームワークは、設計者がその設計に取り込むために選択し得るオプションのリストとして使用する、要求仕様を提供する。

本フレームワークは文書化要求事項を定義するが、セキュリティ要求事項は定義しない。特定のセキュリティ機能を義務づけることもしない。本フレームワークの要求事項は CKMS 設計文書に課せられる。本フレームワークは、優れた CKMS 設計の基礎を形成する必須の実装の選択肢を提供することで、CKMS 設計者の助けとなる。セキュアな CKMS であることを保証するための選択がどのようなものかは、CKMS 設計者、又は本フレームワークに基づいたセキュリティプロファイルのような他のドキュメントに委ねられる。

本フレームワークは、対象とする公共又はプライベートセクタ（例えば、米国政府、金融業界、ヘルスケアサービス）に属する情報の保護に関しての要求事項を義務づけない。セクタは自らのプロフ

[1] 本文書において、CKMS は単一システムの場合も複数システムの場合もあり得るので、そのように読むべきである。

ファイルを開発するか、自らの要求を満たす他のセクタのプロファイルを採用することが期待されている。

本フレームワークへの適合のための要求事項は“shall”文で表される。推奨事項は“should”で表されるが、それは本フレームワークに適合するための要求事項ではない。“must”や“need to”は本フレームワークが基にしている仮定を表すが、CKMS 設計文書の特定の要求事項を構成するわけではない。本フレームワークでは、“**FR:*i,j***”は *i* 節の *j* 番目のフレームワーク要求事項を示す。

FR:1.1 適合 CKMS 設計は、本フレームワークの全ての要求事項（“shall”文）を満たさなければならない。

本フレームワークの要求事項は文書化要求事項であることから、要求事項で明記されている機能が CKMS に実装されていないと記述することによってその要求事項を取り扱うことが適切であるかもしれない。多くの要求事項において、“if”、“how”、“where”、“under what circumstances”という用語が現れる。“if”は条件付き要求事項であることを示す。“if”に対する回答が“no”であれば、CKMS 設計者には、条件が適用されない理由を示すことによってその要求事項を取り扱うことが期待される。“if”に対する回答が“yes”であれば、CKMS 設計者には、要求事項によって課される情報を提供することによってその要求事項を取り扱うことが期待される。“how”への対応は、要求事項がどのように満たされているのか（すなわち、どのように実装され、実施され、使用されるのか）を取り扱うべきである。“where”への対応は、（論理的にはシステム内の）どこに実装されたメカニズムが存在するのかを取り扱うべきである。最後に、“under what circumstances”への対応は、メカニズムが使用される前に適用されなければならない条件を取り扱う。

ここに明記されている全ての要求事項を適切に取り扱い、明記し、満たしている CKMS 設計は、本フレームワークに準拠し、適合していると見なされる。適合 CKMS 設計は、それぞれの要求事項を満たす設計仕様を精査することで、他の適合 CKMS 設計と比較することができる。

1.1 本フレームワークの適用範囲

CKMS は、情報処理アプリケーションを実行する大きな情報システムの一部となる。CKMS は、暗号鍵管理処理（cryptographic key management service）を提供することでこれらのアプリケーションをサポートするが、特定のアプリケーション又は特定のクラスのアプリケーションについての記述は本フレームワークの範囲外である。

本フレームワークのトピックの記述と要求事項の正当性の論拠を示すために、いくつかの導入的な内容が記載されている；しかしながら、本フレームワークでは、読者が鍵管理の原理に関する実用的な知識を有するか、あるいはそのような情報を他（例えば、［SP 800-57-part1］内）から見つけることが可能であることを想定している。附属書 A は、暗号、暗号鍵管理、及びそれらの情報セキュリティへの適用を理解するのに有益な参考文献のリストを含んでいる。

1.2 想定読者

本フレームワークは主として CKMS 設計者向けであることを意図している。しかしながら、CKMS 設計や関連する設計仕様に興味があるあらゆる人が使用してもよい。CKMS セキュリティアナリスト、調達担当者、実装者、システムインテグレータ、オペレータ、及び管理責任者は、本フレームワークに適合する CKMS 設計仕様及び製品に興味を持つことが期待される。

CKMS 設計者は、以下の目的のために、カバーされている全てのトピックを取り扱うチェックリストとして本フレームワークを使用することが期待される：

- CKMS 全体への全ての視点を考慮する
- CKMS に含まれるポリシー、コンポーネント、及びデバイスを選択する

- 設計において下された全ての決定事項を明記する
- 詳細な仕様及び決定根拠とともに決定事項を文書化する

出来上がった設計文書は、実装者が製品を作成する、システムインテグレータが他の製品やサブシステムに製品を組み込む、及び調達担当者が類似の特性を持った製品同士を理解し評価し比較する、のに適しているものにすべきである。

1.3 構成

1 章（**序説**）は、鍵管理フレームワークの紹介とその背後のモティベーションを記載する。

2 章（**フレームワークの基本**）は、本フレームワークの基本的な概念をカバーし、フレームワークの概要を記載する。

3 章（**目標**）は、堅牢な CKMS の目標を定義する。

4 章（**セキュリティポリシー**）は、構成、典型的な内容、並びに情報管理、情報セキュリティ、CKMS セキュリティ及び他の関連するセキュリティポリシーの必要性について説明する。

5 章（**役割及び責任**）は、CKMS をサポートする役割と責任を提示する。

6 章（**暗号鍵及びメタデータ**）は、CKMS の最も重要な要素である鍵とメタデータをカバーする：そのために、利用可能な鍵タイプ、鍵のメタデータ、アクセスコントロールの考慮やセキュリティ課題及び回復メカニズムと共に鍵及びメタデータ管理機能、について列挙及び定義する。

7 章（**相互運用性及び移行**）は、相互運用性の必要性、及び将来のニーズに適応するための CKMS の機能を容易に移行するための機能について考察する。

8 章（**セキュリティコントロール**）は、典型的な CKMS に適用可能なセキュリティコントロールを記載する。

9 章（**テスト及びシステム保証**）は、セキュリティテストと保証について記載する。

10 章（**災害復旧**）は、一般的な災害復旧、及び CKMS 特有の災害復旧について扱う。

11 章（**セキュリティアセスメント**）は、CKMS のセキュリティアセスメントについて説明する。

12 章（**技術的課題**）は、暗号アルゴリズム、鍵確立プロトコル、CKMS デバイス、及び量子コンピュータに関する新しい攻撃によってもたらされる技術的課題について簡潔に説明する。

附属書 A は、有益な参考文献を列挙し記載する。

附属書 B は、本フレームワークで使用される用語集から成る。

附属書 C は、本フレームワークで使用される頭字語のリストを記載する。

2 フレームワークの基本

この章では、暗号鍵管理フレームワークについての利用する動機付け、意図、特性及び限界について説明する。

2.1 暗号鍵管理の原理

今日の情報システム及びそれに含まれる情報は、保護を必要とする重要な資産と考えられる。政府及びビジネスによって使用される情報は、インターネットのような、共有ネットワークを形成する相互接続されたコンピュータ群から成るコンピュータシステムに保持されることが多い。インターネットは、多様な、時には競合する組織と個人によって共有されているため、情報システムは自分自身と自身に含まれる情報を、認可されない開示（窃取）、改ざん及び不正使用から保護するべきである。それに加えて、正規ユーザへのサービス拒否攻撃は、多くのサービス、タイムクリティカルなアプリケーションシステム及びそれらを守るために使用される CKMS にとって重要な脅威と考えられることもある。CKMS でサポートされる通信の匿名性、連結不可能性（Unlinkability）及び観測不可能性（unobservability）を含む追加のセキュリティ要件が、個人のプライバシーを保護するための組織目標から導出されることもある。これらのシステムで使用される情報は、保管時、保護された設備内で処理されている時、そしてある場所から他の場所へ伝送されている時にも保護が必要である。

暗号は、認可されない開示（窃取）からの情報保護、認可されない改変（改ざん）検出、及びシステムエンティティの個体認証（例えば、個人、組織、デバイス、プロセス）のためによく使用される。暗号が特に有用なのは、通信ネットワークを物理的な手段（物理的セキュリティ対策）で保護するためには桁違いの費用がかかる又は実装不可能でさえあるような通信ネットワーク上で、データ通信又はエンティティ認証が行われるときである。そのため、ビジネスが行われる場合や、機微な情報がインターネットを通して伝送される場合に、暗号は広く使用される。暗号はまた、保管されたデータへの物理的又は場合によっては論理的なアクセス権を持っていてもデータの中身を知ったり改変したりする権限がないハッカーや内部関係者（例えば、メンテナンス要員や CKMS ユーザ）に対する保護のレイヤーも提供できる。

CKMS によってそのライフサイクルにわたって管理及び保護されている暗号鍵を、暗号技術は使用する。効果的に実装されている暗号は、大量の情報に対して保護が必要である状況から、鍵及びある種のメタデータ（すなわち、その鍵が使用されるアルゴリズム、その鍵を使用することで提供されるセキュリティ処理（service）などのような、鍵及び認可された使用に関する情報）だけに対しての保護が必要である状況に、情報管理の対象となる範囲を縮小することができる。

CKMS を設計するとき、CKMS によって管理される鍵を保護するために使用される暗号技術は、攻撃者（になりそうな人）が回避したり解読したりすることが実行不可能であるような、保護レベル（セキュリティ強度と呼ばれる）を提供するようにすべきである。暗号技術のセキュリティ強度は、暗号アルゴリズムを解読するのに必要な最小の操作数を、2 を底とする対数で表したものであり、しばしばビットセキュリティとして評価される。この設計原理は、金庫や保管庫の構築に使用される設計原理に相当する：設計者は、侵入の試みに対する合理的な攻撃者のやる気をそぐような標準に沿って保管庫を設計する；つまり、金庫を開ける唯一現実的な方法が、金庫のドアに対して選択可能なダイヤルの組み合わせを、正しい組み合わせが選択されるまで試みる方法であるようにする。同様に、過去に暗号化されたデータを（正しい鍵の知識なしに）復号する唯一の方法が、正しい平文を得るまで選択可能な鍵を試行し、最終的に正しい鍵を使って暗号文を復号する方法であるようにする。金庫によって提供される保護が、選択可能なダイヤルの組み合わせの数に依存するのと同様に、暗号アルゴリズムの強度は選択可能な鍵の数に依存する。

金庫の中身、あるいは暗号化された情報へのアクセスを得る他の手段が存在する可能性もある。例えば、金庫の囲いに穴を開けることや、暗号アルゴリズムを解読するショートカット手法を見つける試みを行うことができる。また、正しい組み合わせ、又は鍵を盗み取ろうとすることもできる。金庫のダイヤルの組み合わせと暗号鍵は、ともに似たような保護を必要とする。CKMS は鍵及びメタデータに対する必要な保護を提供するように設計されるべきである。

暗号は、データに対して 3 つの主要なタイプの保護を提供するために使用される。その 3 つとは、機密性、完全性、ソース認証（source authentication）である。

a) **機密性**保護：データを認可されない開示（窃取）から保護する。暗号化アルゴリズムは平文データを理解不能な暗号文に変換し、復号アルゴリズムは暗号文を元の平文に戻すために使われ

る。変換はひとつ又はそれ以上の暗号鍵によって制御され、鍵を持つ認可された当事者のみが変換を成功裏に実行することができる。

b) **完全性**保護：認可されないデータ改変（改ざん）の検出メカニズムを提供する。暗号学的認証アルゴリズムは、典型的には認証コード又はデジタル署名を計算する。それらは、保護されるデータと、アルゴリズムで使用される暗号鍵を入力とする関数である。正しい鍵を持たずに、エンティティがデータの改変及び正しい認証コード又はデジタル署名の計算を実行できる可能性は極めて低い。したがって、認可されないデータ改変（改ざん）を、改ざんされたデータが使用される前に検出できる。

c) **ソース認証（Source authentication）**：保護されたデータが認可されたエンティティから来たものであることの保証を提供する。例えば、デジタル署名がデータに対して計算され、データとともに伝送されたとする。受信者はデジタル署名を検証して、それゆえにそのデータが特定のエンティティから来たことを知る。本フレームワークにおけるソース認証は、ソースの個体認証することと、認証されたエンティティが実行される機能に参加するために認可されていることの検証することを含む。

これらの保護は、CKMS で保護されるいかなるデータ（鍵及び関連付けられるメタデータ（6.2.1 節の項目 s)と t）を参照）を含む）に対しても提供可能である。

暗号アルゴリズムは、暗号モジュール（ハードウェア、ソフトウェア、ファームウェア、又はその組み合わせ）内に存在しているべきである。暗号モジュールは、その中身（例えば、アルゴリズム、暗号鍵、及びメタデータ）を認可されない改変（改ざん）と開示（窃取）から物理的及び論理的に保護する。暗号モジュールは CKMS の一部で、鍵、メタデータ、及びユーザデータに対して暗号学的保護を提供できる。

FR:2.1 CKMS 設計は、システムによって使用される全ての暗号アルゴリズムと、それぞれのアルゴリズムでサポートされる全ての鍵長を明記しなければならない。

FR:2.2 CKMS 設計は、鍵と鍵に結び付けられたメタデータを保護するために導入されているそれぞれの暗号技術について、推定されるセキュリティ強度を明記しなければならない。

2.2 鍵、メタデータ、信頼関係（Trusted Associations）、及び結び付き（Bindings）

鍵は、特性、制約、受け入れられるユーザ、及び鍵に適用可能なパラメタを指定するメタデータと関連付けられなければならない。例えば、鍵は、鍵タイプ、どのように生成されたか、いつ生成されたか、所有者の識別子、使用するアルゴリズム、及び暗号鍵有効期間（cryptoperiod）を指定するメタデータと関連付けられているかもしれない。メタデータの各ユニットはメタデータ要素と呼ばれる。鍵と同様、メタデータは認可されない改変（改ざん）から保護される必要があり、場合によっては窃取からも保護される必要がある；メタデータはまた、そのソースが適切に認証されている必要がある。

メタデータ要素は、暗黙的に知られ、それゆえ CKMS 内のある種の鍵に対して特に記録されていないかもしれない。例えば、デバイス内の全ての鍵が AES128 ビットの鍵であれば、鍵長を記録するメタデータ要素は必要ないかもしれない。しかしながら、多くのシステムでは、ひとつかそれ以上の明示的に記録されたメタデータ要素を使用して、鍵を他の鍵から区別する必要がある。本 CKMS フレームワークは、CKMS によって明示的に記録され管理されているそれらのメタデータ要素に焦点を当てる。“メタデータ”という用語はこの文脈で使用する（すなわち、“メタデータ”という用語は明示的に記録され管理されているメタデータ要素を意味する）。

ある鍵が与えられたとき、その鍵には多くのメタデータ要素となり得るものが存在する。多くの場合、CKMS が鍵管理機能を実行するために、鍵と選択されたメタデータ要素の間の信頼関係が必要となる。例えば、静的公開鍵（static public key）と所有者の識別子の間には信頼関係があることが望ま

しい。所有者登録プロセスで共に使用される場合、信頼関係は、識別子によって指定された所有者が、対応するプライベート鍵を所有している、又は所有していたことの保証を提供する。

メタデータ要素は、鍵を生成した同じエンティティによって生成されることもあるし、あるいは信頼されるエンティティから受け取ることもある。メタデータが信頼されるエンティティからいつ受け取ったとしても（関連付けられた鍵が同時に送られたかどうかに関わらず)、メタデータと関連付けられた鍵には信頼関係がなければならない。配付時に保持される信頼関係は、鍵及びメタデータの暗号学的結び付き（cryptographic binding）（例えば、鍵とメタデータの組で計算されたデジタル署名）によって確立されるか、あるいは信頼プロセス（trusted process）（例えば、既知で信頼されるエンティティからのメタデータの対面手渡し）によって確立される。CKMS は、鍵及びメタデータの配付プロセス及び管理プロセスで使用される暗号学的結び付き（cryptogaphic binding）機能及び検証機能を提供することが多い。受信者は、鍵及びメタデータが適切に関連付けられ、特定のソースから来ていて、改ざんされておらず、かつ伝送中の認可されない開示（窃取）から保護されていることの保証を得る。メタデータの受信した場合、その後すぐに、鍵とメタデータの間の関連付けは検証されるべきである。暗号学的結び付き（cryptographic binding）は、鍵及び関連付けられたメタデータ要素に適切な暗号検証機能を適用することで検証される。暗号学的ではない信頼関係は、信頼プロセス（trusted process）（すなわち、送信エンティティと配付プロセスの信頼）を評価することによって検証される。下の図 1 を参照されたい。

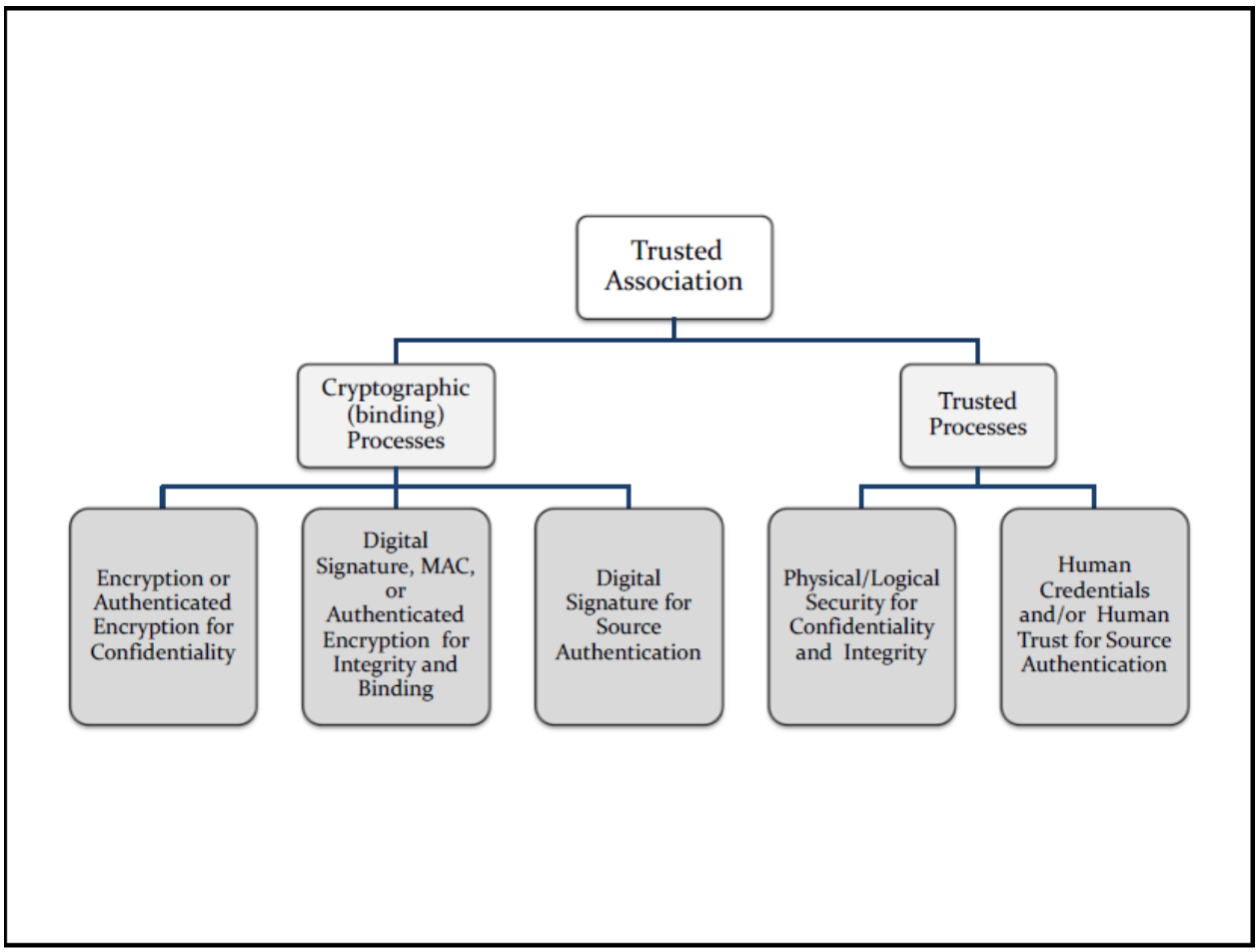

図 1 信頼関係とサポートするプロセス

受信後、メタデータはローカルで生成された他のメタデータ（もしあれば）と組み合わされることができ、鍵と全ての利用可能なメタデータとの間の新しい信頼関係を保管された情報のために確立できる。

システムに保管されたメタデータに対しては、鍵とそのメタデータの間の信頼関係も必要となる。その関連付けは、ストレージの場所と特性に依存するものの、物理的セキュリティ手段又は暗号学的手段を利用して保持され得る。物理的セキュリティ手段には、関連付け（すなわち、鍵及びそのメタデータの機密性（要求された場合）及び完全性）を保持するために信頼されるデバイス内のストレージが含まれる。信頼関係の完全性が保持されている限り、メタデータ要素は、関連付けられた鍵に属しており、認可されないエンティティに開示（窃取）されていないことが保証される。しかしながら、そのような物理的セキュリティ手段は非現実的なこともある。物理的にセキュアなストレージサイトはコストがあまりに高いか、又は利用不可能であることもある。この場合、鍵及びそのメタデータが適切に関連付けられていることの保証を提供するために、暗号学的結び付き（cryptographic binding）が要求されうる。

2.3 CKMS アプリケーション

CKMS は、単一の個人（例えば、個人のデータストレージシステム内）、組織（例えば、オフィス間通信のためのセキュアな VPN 内）、又は大規模で複雑な組織（例えば、米国政府用セキュア通信内）に対してサービスを提供するために設計されうる。CKMS は所有されることもレンタルされることもありうる。

2.4 フレームワークトピックと要求事項

本フレームワークは、フレームワークトピック（FT）（節の見出しに対応する）のリストと、それぞれのトピックに対し CKMS を設計する上で満たす必要があるフレームワーク要求事項（FR）のセットから構成される（下の図 2 を参照）。これらの要求事項は CKMS 設計に課せられる。

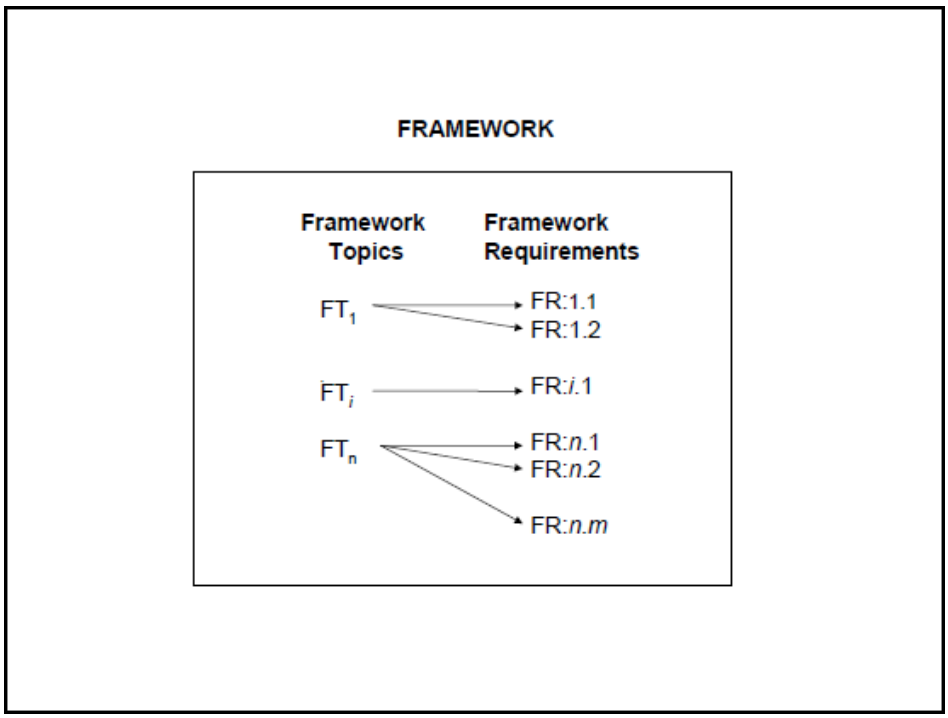

図 2 フレームワークトピックとフレームワーク要求事項

本フレームワークは、CKMS におけるいかなる特定のポリシー、手続き、セキュリティ要求事項、あるいはシステム設計制約も課さない；CKMS 設計者が理解し比較することができるように、構造的な方法で文書化されることを単に要求しているだけである。

本フレームワークは、あるセクタ（米国連邦政府、航空宇宙、ヘルスケアなどのような）のための特定の CKMS 又は CKMS のクラスを指向しているわけではない。本フレームワークは全ての CKMS に適用できることを意図している。

FR:2.3 適合 CKMS 設計は、本フレームワークの要求事項で要求されているように、設計選択について記載し、文書を提供しなければならない。

2.5 CKMS 設計

CKMS 設計の目的は、機微なデータを保護するための暗号鍵を使用するエンティティに対して、暗号鍵を提供するようなシステムを、どのように構築できるかを記述することである。CKMS の概要は、各鍵タイプの利用用途、鍵をどこでどのように生成するのか、鍵が存在するそれぞれエンティティのストレージ内及び配送中においてどのように鍵を保護するのか、及び鍵が配送されるエンティティのタイプを示すべきである。

図 3 は、CKMS 設計が本フレームワークに適合していることをどのように示すことができるかを説明している。それぞれのフレームワーク要求事項 **FR:*i,j*** に対し、適切なフレームワークへの回答 **fr:*i,j*** が CKMS 設計者によって要求事項に適合するように提供される。要求事項とその回答{**FR:*i,j*, fr:*i,j*}から成る完全なペア一式が CKMS 設計を形成する。

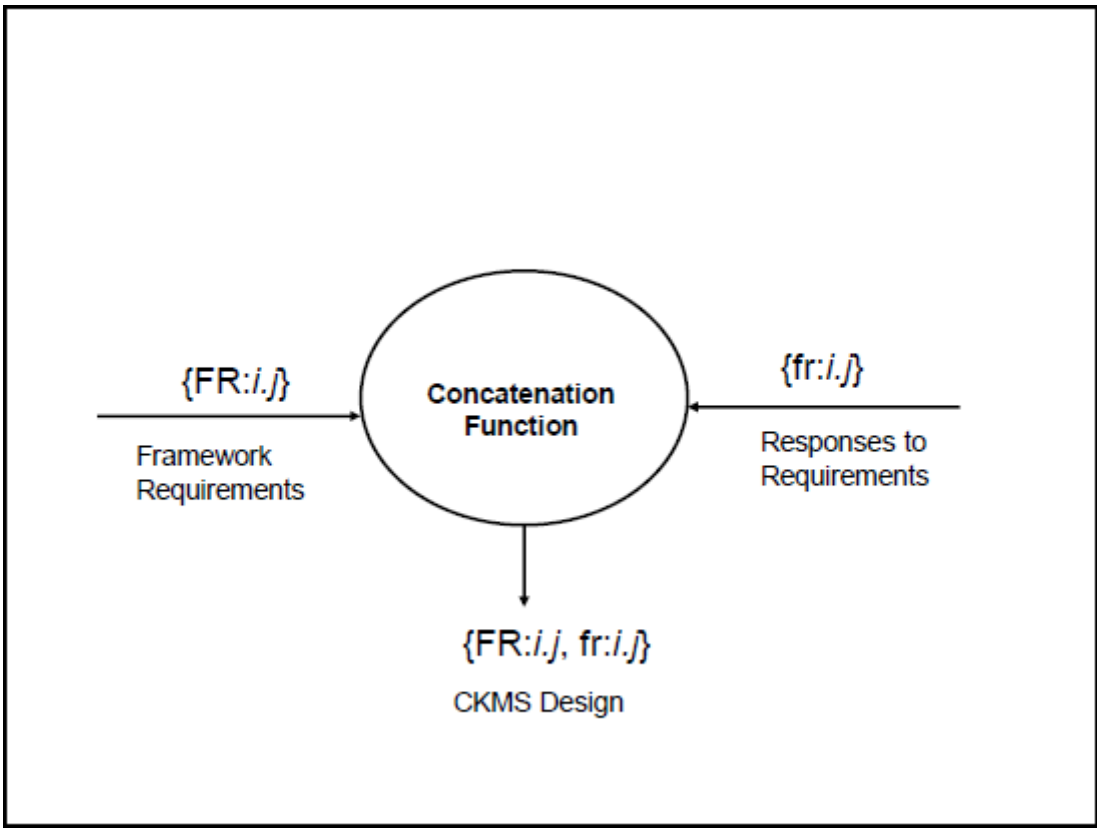

図 3 フレームワークに適合するための CKMS 設計プロセス

FR:2.4 CKMS 設計は、以下を含む CKMS システムの高レベルの概要を明記しなければならない：

a) **利用するそれぞれの鍵タイプの用途**

b) **鍵が生成される場所と手段**

c) **それぞれの鍵タイプとの信頼関係で使用されるメタデータ要素**

d) **鍵やメタデータが存在しているそれぞれのエンティティのストレージにおける、鍵やメタデータの保護方法**

e) **配送時の鍵やメタデータの保護方法**

f) **鍵やメタデータが配送され得る先となるエンティティの種類（例えば、ユーザ、ユーザデバイス、ネットワークデバイス）**

2.6 CKMS プロファイル

CKMS プロファイルは、適合する CKMS、その実装、及びその運用が、（例えば連邦政府機関のような）特定のセクタの組織において満たさなければならない要求事項を規定する。CKMS プロファイルは、CKMS がどのように設計され、実装され、テストされ、評価され、かつ運用されなければならないかを規定する。セクタとは、製品、システム、又はサービスへの共通の要求事項を持つ組織のグループである。CKMS プロファイルは、運用システムで実装されているような CKMS によって満たさなければならないセキュリティと相互運用性に関わる要求事項一式である。本フレームワークは特定のセクタのための CKMS プロファイルを導出するために使用してもよい。本フレームワークと同様に、ひとつ又はそれ以上のプロファイル要求事項がそれぞれのプロファイルトピックに対応する。

2.7 CKMS フレームワークと導出されたプロファイル

図 4 は、CKMS フレームワークと、CKMS から導出されたセクタのプロファイルとの関係を描いている。CKMS プロファイルをフレームワークから導出するとき、フレームワークの要求事項は、対象として選択されたセクタでの必要性を満たすように、増補し精緻化されることがある。例えば NIST は、本フレームワークを使用しつつ、適用可能な Feredal Information Processing Standarts（FIPS）、NIST Special Publications（SP）及び精緻化された要求事項を満たすために必要なガイドラインに準拠した標準やプロトコルを選択し、米国連邦政府機関のための連邦 CKMS プロファイルを開発することができる。

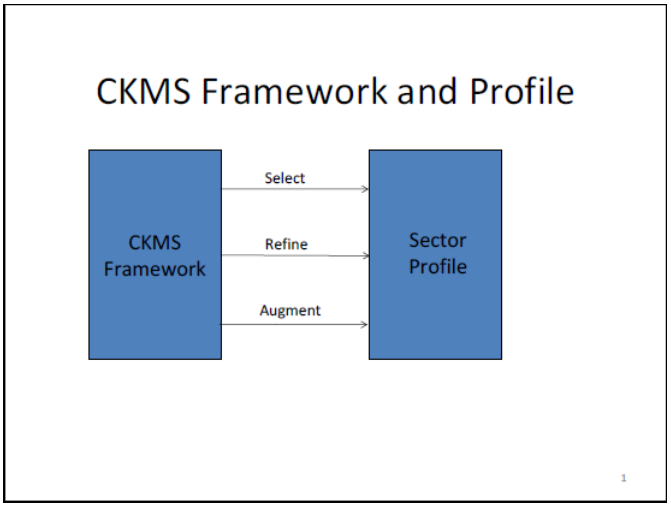

図 4 CKMS フレームワークとセクタプロファイルとの関係

2.8 フレームワークとプロファイルの違い

フレームワークは、特定のトピックが CKMS の設計で扱われていることを要求するが、これは設計そのものを判断するものではない。あらゆる CKMS は、本フレームワークに従って設計したり規定されたりし得る。一方、プロファイルは、指定された使用セクタが満足する CKMS を得るために満たさなければならない要求事項を記述する。CKMS プロファイルは判断を下す（すなわち、プロファイルに適合するために何を実装し、使用する必要があるかを指定する）。本フレームワークに適合するある CKMS が、特定のプロファイルに適合しないこともありえる。例えば、2.1 節の **FR:2.1** は CKMS 設計が CKMS によって使用される全ての暗号アルゴリズムを明記することを要求する。米国連邦政府 CKMS プロファイルは NIST 承認された暗号アルゴリズムのみを使用することを要求するかもしれない。

2.9 セキュア E メールアプリケーションをサポートする分散 CKMS の例

図 5 は、システム（図では System A、System B、System C）間でインターネットを通じて通信する分散 CKMS を描いている。CKMS は全ての CKMS モジュール（図では CKMS Module A、CKMS Module B、CKMS Module C）の和集合から成る。各 CKMS モジュールはシステム内の論理的エンティティと見なされる。CKMS 機能を実行するシステムのいかなる部分も、それらの機能が実行される時において論理的 CKMS モジュールの一部である。それに加えて、CKMS モジュールの一部（例えば、暗号化アルゴリズム）は他のアプリケーションで使用されるかもしれない（例えば、一般データを暗号化する）。

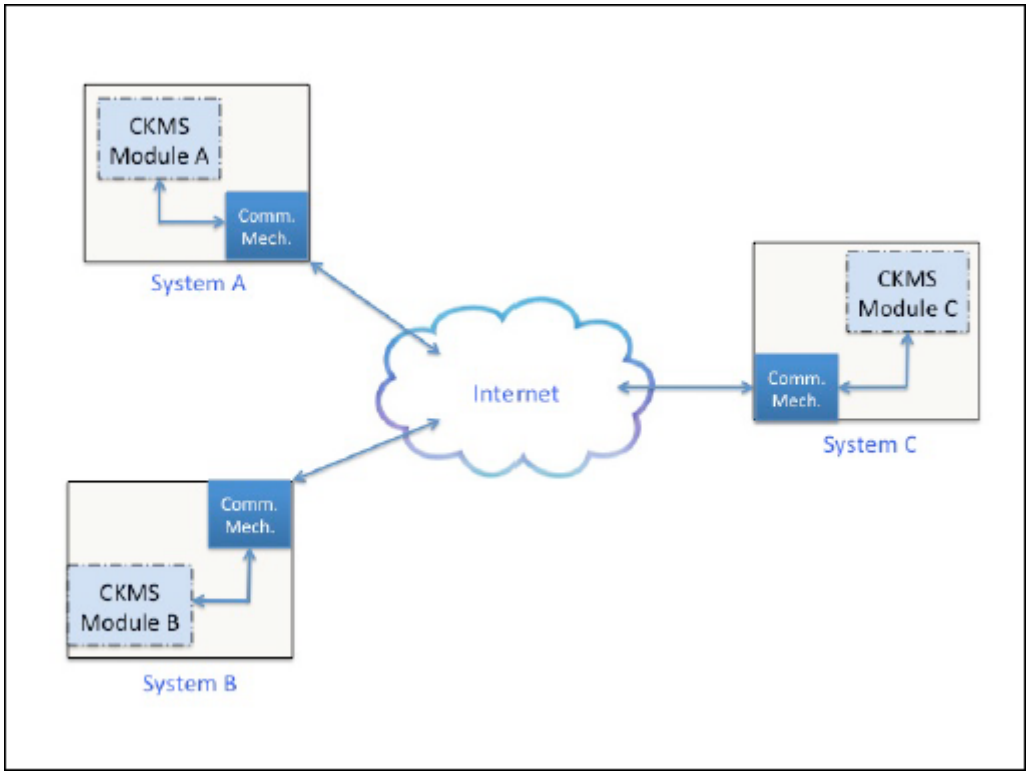

図 5 CKMS 概要例

他の（CKMS モジュールを含む）システムとの間で、インターネットを介してやり取りを行う実際の通信メカニズムは、CKMS の一部ではない。しかしながら、CKMS の機能を実行するためのプロト

コルの部分（例えば、プロトコル内で使うために鍵を生成して鍵管理情報を提供する）は CKMS の一部と見なされる。

図 6 は、分散 CKMS を使用する E メールアプリケーションの例である。送信者の E メールアプリケーションは CKMS モジュールと接続する。CKMS モジュールは、以下の目的で利用する鍵を生成する；意図する受信者宛にインターネットを介して送信する E メールデータに対して要求される暗号学的保護を適用し、さらに要求があれば、受信エンティティに伝送される鍵に暗号学的保護を適用する。その後、E メールアプリケーションは、保護された鍵と保護されたデータを伝送用の通信メカニズムに引き渡す。通信メカニズムもまた図 5 で議論されたように CKMS モジュールとやり取りし得ることに注意されたい。

送信者のシステム内の通信メカニズムは、必要に応じて、暗号学的に保護された E メールデータを E メールアプリケーションに送る前に、CKMS モジュールとやり取りする。E メールアプリケーションは、保護された鍵をローカル CKMS モジュールに送り、その鍵はその後、保護された E メールデータを処理するために使用される。

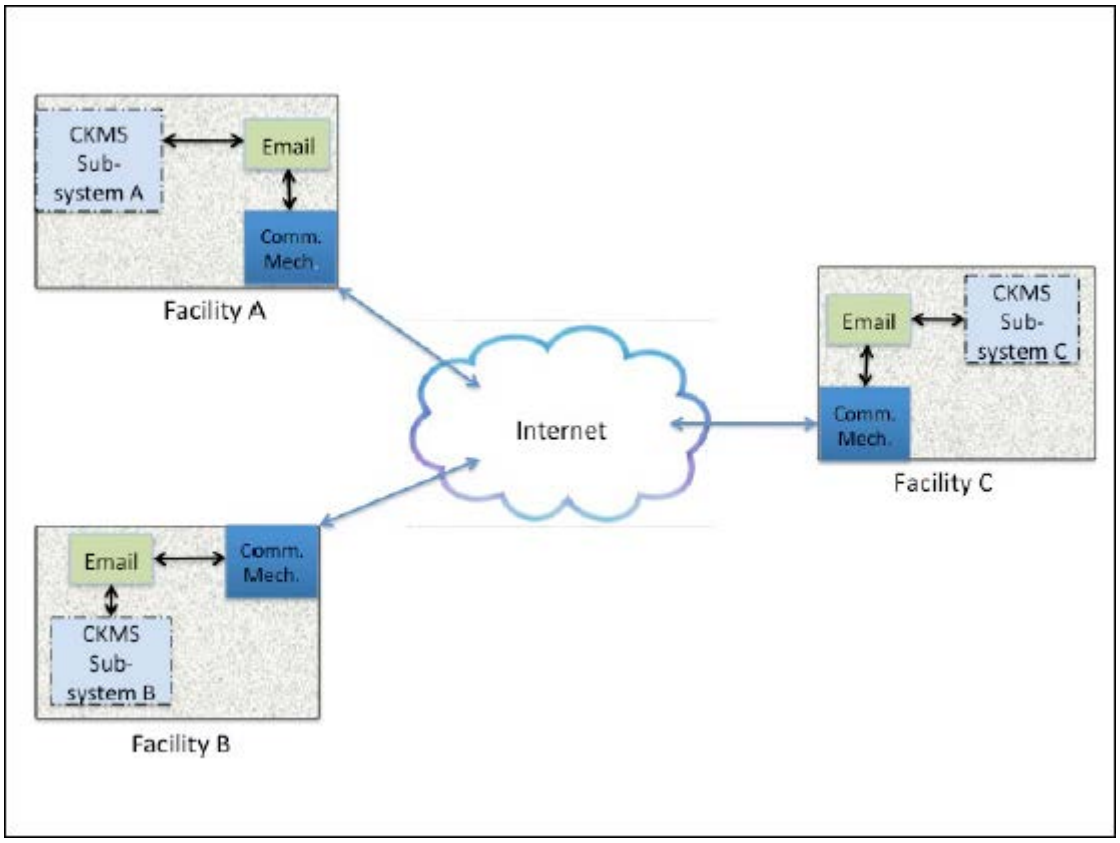

図 6 セキュア E メールアプリケーションの例

2.10 CKMS フレームワークコンポーネントとデバイス

この CKMS フレームワークでは、“コンポーネント（component)”という用語を、CKMS を構成するために必要とするハードウェアやソフトウェア、あるいはファームウェアという意味で使用する。“デバイス（device)”という用語は、特定の目的を供するコンポーネントの組み合わせを意味する。CKMS は、単一ユーザが利用するコンピュータ上で動きユーザアプリケーションをサポートするソフトウェアプログラムのように、シンプルであることもありうる。また、ネットワークでつながった膨大なユーザ及びアプリケーションに鍵管理サービスを提供するために、多くのデバイスで構成された多様なサブシステム群のような、複雑なものであることもありえる。CKMS は、単一のコンピュ

ータに実装されることも、あるいは地理的に広く分散して無数の通信ネットワークを通してつながっていることもある。本フレームワークでは、プロセッサ、通信手段、ストレージユニット等は全てデバイスと見なされる。

CKMS は、暗号鍵及びメタデータを保護、管理及び確立するために設計されたポリシー、手続き、デバイス及びコンポーネントの一式として記述することができる。CKMS は、ひとつ以上の組織とそのユーザの代理として暗号鍵管理処理（service）を実行する機能一式を提供する。これらの機能は、まとめて CKMS 設計での仕様項目として表される（6.4 節を参照）。

FR:2.5 CKMS 設計は、CKMS の全ての主要なデバイス（例えば、メーカ、モデル、バージョン）を明記しなければならない。

3 目標

CKMS は、特定の目標を達成するために設計されるべきである。この章では、いくつかの考えられる目標について説明する。

3.1 ネットワーク、アプリケーション、及びユーザへの鍵管理の提供

いくつかのセキュリティプロトコル標準（例えば、TLS、IKE、SSH、CMS）では、暗号が広範囲に使用されている。そこでは、プロトコル自身によって、静的鍵（static key）（すなわち、長期的な鍵）も一時鍵（ephemeral key）（すなわち、1 回のセッション又は鍵管理トランザクションでのみ使用される鍵）も使用される。CKMS の対象は静的鍵（static key）の生成、配付及び保管であるが、CKMS 設計では使用する一時鍵（ephemeral key）の生成、保管及び保護も含めなければならない。

CKMS の運用を支えているネットワークは CKMS の通信バックボーンを形成する。CKMS 設計者は、CKMS がネットワークへ与えうる負の影響を最小化するような設計にするために、ネットワークの効率性及び信頼性を理解する必要がある。ネットワークサイズ及びスケーラビリティは、CKMS が初期及び将来に扱う必要があるユーザ数に関してある種の指標を与える。誤り特性のようなネットワークの特性もまた、復号実行後に通信エラーの影響が拡大（悪化）する恐れがある暗号アルゴリズムや暗号利用モードの選択に影響を与え得る。

CKMS は、特定の単一アプリケーション（例えば、E メール、データ保管、ヘルスケアシステム、支払システム）を扱うように構築されることも、あるいは多くのアプリケーションを包含するエンタープライズ全体を扱うように設計されることもありえる。単一のアプリケーションのために設計された CKMS はアプリケーションに特化して設計され、緊密に統合される傾向がある一方、エンタープライズ向け CKMS は共通の鍵管理機能が可能な限り共有できるように、より汎用的であるだろう。CKMS 設計者は、設計における選択に影響する可能性が高いので、利用可能なアプリケーションをよく理解する必要がある。

CKMS 設計者は、システムの潜在的なユーザについても検討するべきである。

- 何人のユーザがどのような目的で CKMS を使用するのか？
- ユーザはモバイル利用なのか固定された場所にいるのか？
- ユーザは CKMS についてよく知っている必要があるのか、それともユーザにとっては透過的なのか？
- ユーザは、作業を遂行する上で時間が重要であるような、ストレスのかかる条件下で操作するのか？

CKMS の中には、ユーザが暗号鍵及び公開鍵証明書の目的と重要性を理解していると CKMS 設計者が仮定したために、失敗したものがある。CKMS によってユーザの作業が妨げられるならば、CKMS は使用されなくなるため、成功するセキュリティソリューションとはいえなくなる可能性が高い。

CKMS 設計者の目標は、望ましいレベルのセキュリティを提供してアプリケーションと使用する組織のニーズを満たし、手頃なコストで、かつ運用への負の影響が最小限になることを同時に満たすように機能するセキュリティメカニズム一式を規定することである。これらは、他の CKMS の目標と同様、CKMS が設計され、実装され、運用される前に考慮されるべきである。

FR:3.1 CKMS 設計は、それが機能する通信ネットワークに関しての目標を明記しなければならない。

FR:3.2 CKMS 設計は、それがサポートすることを意図しているアプリケーションを明記しなければならない。

FR:3.3 CKMS 設計は、意図するユーザ数と、それらのユーザに課する責任を一覧にしなければならない。

3.2 CKMS における商用既製品使用の最大化

顧客は一般に商用既製品（COTS）を好む。そのような製品は、入手、運用及び保守のためのコストが特定顧客用にカスタム設計、構築された製品より安いことが多い。しかしながら、多数の顧客の“最小公倍数”的な要求を満たすように設計し構築された商用既製品は、どの顧客の要求も完全には満たさないかもしれない。CKMS 設計者が、ある特定の市場セクタでの一連の要求事項を満たす製品を使用するならば、その CKMS はその市場で受け入れられる可能性がより高まる。

一般に、標準インタフェースを使用することは製品の拡張性を高める。多様な機能及び負荷に関する要求（ユーザ数、トランザクション、鍵、及びアプリケーションデータに基づくものを含む）を満たすように CKMS を設定できるようにするために、CKMS は商用既製品を採用することで拡張や拡充ができるような設計とすべきである。

FR:3.4 CKMS 設計は、CKMS で使用される商用既製品を明記しなければならない。

FR:3.5 CKMS 設計は、商用既製品によってどのセキュリティ機能が実行されるのかを明記しなければならない。

FR:3.6 CKMS 設計は、CKMS の目標を満たすために商用既製品をどのように設定し拡張するかを明記しなければならない。

3.3 標準への適合

適用可能な標準を利用する範囲を精査することで、CKMS に関して多くのことを学ぶことができる。標準に準拠する設計では、標準を開発した人の経験と知恵の恩恵を受けることができる。加えて、その標準に適合性を検査する認証プログラムがある場合、CKMS が正しく実装されていることのさらなる信頼性が得られる。各々に簡潔な説明を付した適切な標準のリストは附属書 A を参照されたい。

標準は、何かをどのように行わなければならないか、あるいは行うべきであるかを規定する。複数のベンダが同じ標準を作り上げ、それによって相互運用性と競争を促進させることができる。加えて、標準を使用することは製品又は実装における信頼性を高めることが多い。複数の当事者の共同作業によって開発されレビューされた標準にはさらなる信頼性がもたらされる。また、標準への適合は、必須の概念を再発明する必要がないため、製品の製造期間（time-to-production）又は実装の作業

時間（time-to-operation）も削減させ得る。適合性試験機関は、市場に製品が出回る前に実装の誤りを発見し除去するのに有用である。

FR:3.7 CKMS 設計は、CKMS に使用される連邦政府標準、国内標準、及び国際標準を明記しなければならない。

CKMS アーキテクチャにおいてひとつ以上の標準に準拠する商用既製品が利用可能であることは、CKMS 構築の時間とコストを大きく削減しうる。適合性試験を受けた製品のための先行投資コストは、適合しない製品を採用したり類似の製品をゼロから作り上げたりする必要がないことによって（レビューや実装に関して）節約できるコストで十分に補われることが期待できる。

FR:3.8 CKMS に使用されるそれぞれの標準に対して、CKMS 設計は、どの CKMS デバイスが標準を実装しているのかを明記しなければならない。

FR:3.9 CKMS に使用されるそれぞれの標準に対して、CKMS 設計は、標準への適合がどのように検証されるか（例えば、第三者試験プログラムによって）を明記しなければならない。

3.4 使いやすさ

CKMS の使用にあたって最も重大な制約は、習熟していないユーザに対してある種のシステムがもたらす難しさにあるかもしれない。ほとんどのユーザは暗号セキュリティエキスパートではなく、セキュリティは彼らにとって第 2 のゴールに過ぎないため、CKMS は可能な限り透過的である必要がある。ユーザの技量に適応したユーザインタフェースは、新規で習熟していないユーザをガイドすることができる一方、エキスパートには効率的なショートカットを使い、ステップバイステップのガイダンスを迂回できるようにする。

3.4.1 ユーザの技能と嗜好への適応（Accommodate User Ability and Preferences）

使いやすさは非常に主観的である。ある人にとって簡単又は自明であることが他の人にとってはそうではないことがある。CKMS 設計者は、ユーザが一般にはセキュリティエキスパートでないので、使用しているセキュリティ機能の目的を理解していないかもしれないことに留意すべきである。セキュリティは一般に製品の最優先の目的ではないので、透過的なセキュリティが望ましい。ネガティブな経験は製品の受け入れ及び使用に影響を及ぼす可能性が高い。したがって、大きなセグメントの潜在ユーザ集団が求めるセキュリティ製品の使いやすさのニーズを満たす必要がある。

FR:3.10 CKMS 設計は、システムへの全てのユーザインタフェースを明記しなければならない。

FR:3.11 CKMS 設計は、提案されたユーザインタフェースの使いやすさに関する、あらゆるユーザ受け入れテストの結果を明記しなければならない。

3.4.2 ユーザインタフェースの設計原理

使いやすさは非常に主観的で評価が困難であるが、この目標を達成するためのいくつかの設計原則が確立されている。使いやすさの設計目標は、以下のことを確実に行うようにするべきである：

a) CKMS を使用して正しいことを行うことが直感的で容易である。例えば、鍵管理機能の関数呼び出し名には直感的な名づけがなされるべきである。

b) システムを使用して誤ったことを行うことが困難である。例えば、CKMS は署名専用鍵を使用した暗号化を許可すべきでない。

c) 誤ったことを実行したときの回復が直感的で容易である。例えば、CKMS は前の操作を取り消す undo 機能を提供すべきである。

このアプローチは、ユーザサポートに係るコストを削減することで、ライフサイクル全体を通して係るコストを削減する。

FR:3.12 CKMS 設計は、ユーザインタフェースの設計原理を明記しなければならない。

FR:3.13 CKMS 設計は、システムに設計された全てのヒューマンエラー防止又はフェールセーフ機能を明記しなければならない。

3.5 パフォーマンス及びスケーラビリティ

計算処理と通信におけるパフォーマンス向上はコンピュータ業界における主要なサクセスストーリーである。パフォーマンスが向上しても、新たなアプリケーションはさらに高速な処理と通信を要求する。過去には、多くの大規模な鍵配付センタが、最大で数千人のセキュリティ加入者にサービスを提供していた。現在では、何百万もの人々が要求（セキュリティ及び暗号鍵に関する新しい要求を含む）を高めつつインターネットを定常的に使用している。セキュアな処理、データ保管、通信へのニーズは増大し続けている。ニーズの増大は増大する負荷に耐えるためのスケーラビリティを CKMS に要求する。

FR:3.14 CKMS 設計は、CKMS のパフォーマンス特性を明記しなければならない。それには、実装された機能とトランザクションのタイプによる処理可能な平均及びピーク時の負荷と、その負荷がかかったときの機能とトランザクションのタイプごとの応答時間を含む。

FR:3.15 CKMS 設計は、増大する負荷要求に応じてシステムを拡張するために、サポートされており、利用することができる技術を明記しなければならない。

FR:3.16 CKMS 設計は、増大する負荷要求に対応して CKMS を拡張できる範囲を明記しなければならない。これは、追加される負荷、その負荷に対する応答時間、及びコストの観点で表現しなければならない。

4 セキュリティポリシー

CKMS は、CKMS を使用しているそれぞれの組織の目標をサポートするやり方で設計されなければならない。いくつかのタイプのポリシーは CKMS の設計及び使用に影響を及ぼす。

組織は、異なったアプリケーション又は情報のカテゴリをカバーする異なったポリシーを持っているかもしれない。例えば、軍事関係の組織は、機密（classified）情報をカバーするポリシー一式及び人事情報をカバーする完全に異なったポリシー一式を持っているかもしれない。

組織は、情報管理レベルの課題を取り扱う高位のポリシー及びデータ保護に関する特定のルールを取り扱う低位のポリシーから成る階層構造のポリシーを作成し、それに依存することが多い。物理セキュリティポリシーがある文書に規定され、通信セキュリティポリシーが別の文書に規定されていることもある。コンピュータシステムは独自のコンピュータセキュリティポリシーに従って構築されることが多い。

ポリシーの各層（例えば、情報管理、情報セキュリティ、物理セキュリティ、コンピュータセキュリティ、通信セキュリティ、及び暗号鍵セキュリティ）は、様々な形態で相互に関係する。ポリシー階層の中間及び下の層は、より高位層のポリシーよりも実装及び実施についてさらに詳細な記述を提供すべきである。例えば、認可されない開示（窃取）から情報は保護されなければならないと規定された組織的情報管理ポリシーは、適切に識別され認可された人にのみ情報へのアクセス及び使用が制限されることを規定する情報セキュリティポリシーにつながるべきである。

組織は CKMS に対する要求事項を決定する階層構造のポリシーを使用するかもしれない。図 7 は使用されるポリシー群とそれらの間の関係の例を示す。これらのポリシーに関するさらなる説明は以降の小節に記載する。

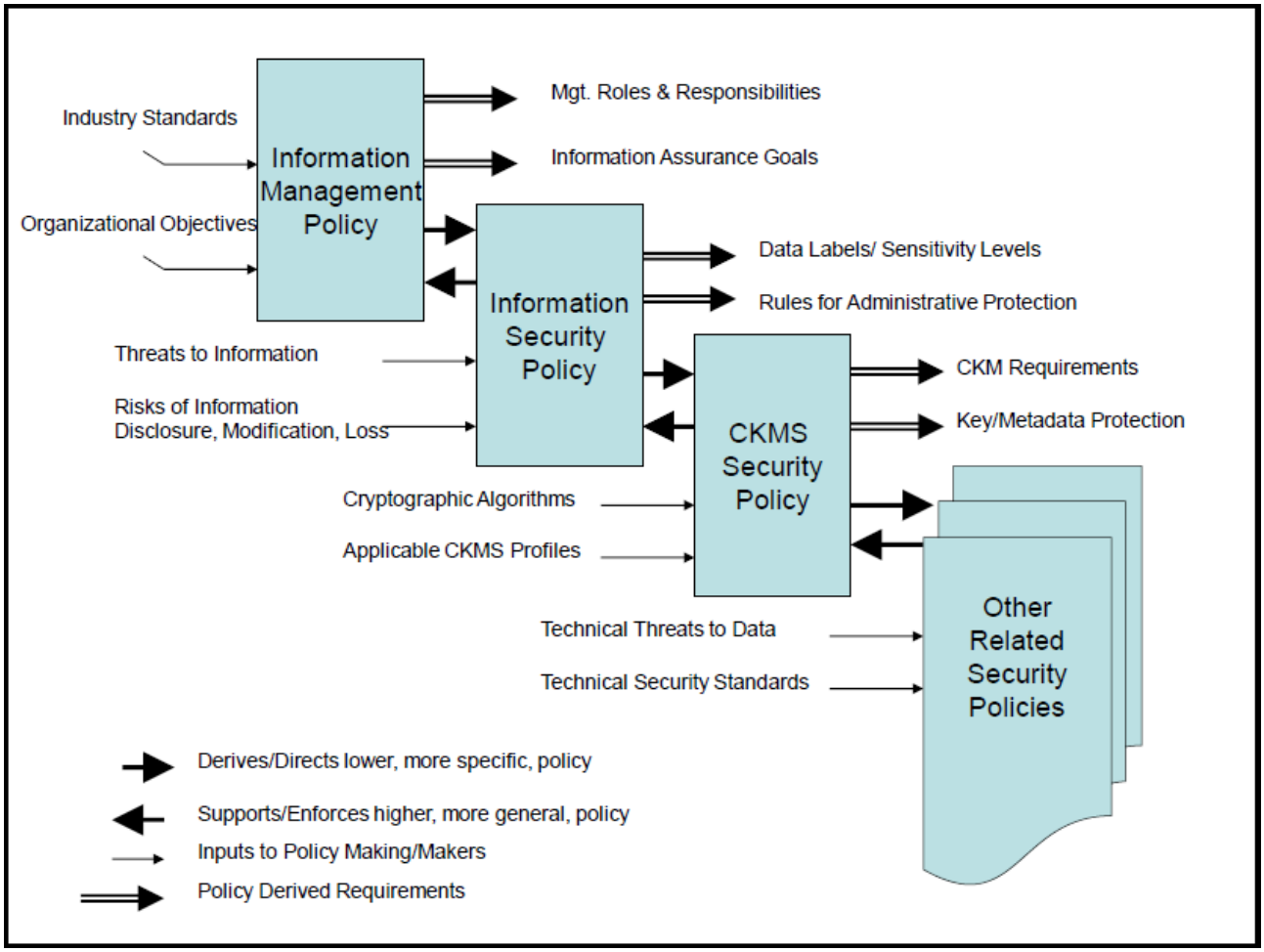

図 7 関連するセキュリティポリシー

4.1 情報管理ポリシー

組織の情報管理ポリシーは、どの情報が収集又は作成されるか、及びどのようにその情報が管理されるかを規定する。組織の経営層は、優れた実践例についての業界標準、組織の情報に関する法的要求事項、及び組織が収集及び作成する情報を使用して達成しなければならない組織の目標を利用し、情報管理ポリシーを確立する。

情報管理ポリシーは、典型的には管理上の役割及び責任を明らかにし、役員・従業員がこれらの情報管理の義務を実行するために必要な権限を確立する。また、それは、どの情報に価値があり機微であると見なされるのか、及びどのように保護されるのかも規定する。特に、ポリシー階層の最上位に位置するこのポリシーは、どのカテゴリの情報が認可されない開示（窃取）、改ざん又は破壊から保護される必要があるのかを規定する。これらの規定は、情報セキュリティポリシーの基盤を形成し、異なるカテゴリの機微で価値のある情報に対して提供されなければならない、機密性、完全性、可用性及びソース認証（source authenticataion）の保護レベルを決定する。

4.2 情報セキュリティポリシー

組織の情報セキュリティポリシーは、どの情報が想定される脅威から保護され、どのようにその保護が達成されるのかをより詳細に規定することによって、組織の情報管理ポリシーの一部をサポート及び実施するために作成される。価値があり機微である情報を、紙及び電子的方法の両方で、収集、保護及び配付するためのルールはこの階層のポリシーで規定される。情報セキュリティポリシーへの入力は、情報管理ポリシーの記載事項、組織の情報に対するセキュリティへの潜在的脅威、及び情報への認可されない開示（窃取)、改ざん、破壊又は損失に関わるリスクなどを含む（なお、これらだけに限らない)。

情報セキュリティポリシーの出力は、情報の様々なカテゴリに割り当てられた情報機密性レベル（例えば、低、中、高）及び情報を保護するための高レベルなルールを含む。また、情報セキュリティポリシーは、組織のための鍵及びメタデータの機密性及び完全性の保護を提供するための暗号鍵、アルゴリズム、メカニズムの使用及び保護を規定する CKMS セキュリティポリシーを作成するために使用されることもある。

4.3 CKMS セキュリティポリシー

CKMS セキュリティポリシーは、CKMS がサポートしなければならない鍵及びメタデータの保護のためのルールを規定する。CKMS セキュリティポリシーは、CKMS によって使用される全ての暗号鍵及びメタデータの機密性、完全性、可用性、及びソース認証（source authentication）を保護するためのルールを確立し規定する必要がある。これらのルールは鍵ライフサイクル全体（いつ鍵が運用され、保管され、伝送されるかを含む）をカバーする。CKMS セキュリティポリシーは、CKMS が選択可能な全ての暗号メカニズム及び暗号プロトコルを含むこともある。CKMS セキュリティポリシーは、組織のより高位レベルのポリシー群と整合している必要がある。例えば、情報セキュリティポリシーに、電子的に送信された情報の機密性は最大 30 年間保護すると記載している場合、CKMS セキュリティポリシー及び CKMS 設計にはそのポリシーをサポートする能力がなければならない。

CKMS 設計者は、CKMS を使用する組織のメンバーではないかも知れず、組織のポリシーへアクセスできないかもしれない。例えば組織は、外部で開発された CKMS 又は CKMS サービスを購入するかもしれない。CKMS 設計者は、CKMS を作成する対象である市場をサポートする設計の中にセキュリティ能力又は機能一式を作成すべきである。これらのセキュリティ能力又は機能群は CKMS 設計者によって文書化されるべきで、CKMS 設計者による最初の CKMS セキュリティポリシーを形成すると見なすことができる。CKMS 設計文書には、それらの機能が CKMS セキュリティポリシーをサポートするためにいつどのように使用されるのかを記述すべきである。組織は、CKMS 設計者によって開発された CKMS セキュリティポリシーの初期設定を元に、CKMS 設計者又は CKMS サービスプロバイダと共に CKMS セキュリティポリシーを修正する作業をするかもしれない。究極的には、CKMS 設計が CKMS セキュリティポリシー（修正されている場合もある）を適切にサポートしているか、又はサポートするように設定できるかどうかを保証するのは CKMS を使用する組織の責任である。

それぞれの鍵タイプ及びそのメタデータ（6 章を参照）に適用される特定の保護は、(CKMS セキュリティポリシーの一部となり得る）鍵セキュリティポリシーをサポートしていると見なされる。鍵セキュリティポリシーは、鍵ライフサイクル全体にわたる鍵及びそのメタデータの機密性、完全性及びソース認証（source authentication）のためのポリシーを記述する。その後、これらのポリシーは CKMS によってサポートされる。

鍵及びメタデータが保持される時間の長さを規定する、鍵及びメタデータの保持ポリシー（Key and Metadata Retention Policy）もまた、CKMS セキュリティポリシーの一部であるべきである。鍵及びメタデータの保持ポリシーは、鍵及びメタデータが保護する情報の機微度に基づくべきである。CKMS は鍵及びメタデータの保持ポリシーを実行すべきである。例えば、CKMS は、鍵及びメタデータをセキュリティ保護期間を通して保護すべきであり、そして鍵及びメタデータが必要なくなったときにそれらを破壊すべきである。

CKMS セキュリティポリシーは、ポリシーを守ることに責任を持つ役員・従業員が、ポリシーを容易に理解して自らの役割及び責任を正しく実施できるように書かれるべきである。セキュリティポリシーは、CKMS 内に電子的に保管され自動的に処理されるフォーム（例えば、表、形式記述言語、フローチャート）の形で規定されているかもしれないことに注意されたい。形式記述言語で記述されたポリシーは、自動的に適用するように設計された CKMS によって実行されるかもしれない。そのようなシステムは、自分自身が正常に機能しているかをチェックし、現在のあるいは潜在的な問題を診断し、問題を組織の責任のあるエンティティに報告し、場合によっては問題を自動的に修復することまで行うことができるかもしれない。

FR:4.1 CKMS 設計は、実施するために設計した設定可能なオプションとサブポリシーを含む CKMS セキュリティポリシーを明記しなければならない。

FR:4.2 CKMS 設計は、CKMS セキュリティポリシーが CKMS によってどのように実施されるのか（例えば、ポリシーが要求する保護を提供するために使用されるメカニズム）を明記しなければならない。

FR:4.3 CKMS 設計は、CKMS セキュリティポリシーのあらゆる自動化部分についてどのように曖昧さのない表形式又は形式言語（例えば XML、ASN.1）で表現されているのかを明記しなければならない。これは CKMS の自動化されたセキュリティシステム（例えば table driven 又は syntax-directed software mechanisms）がそれらを実行できるようにするためである。

4.4 その他の関連するセキュリティポリシー

CKMS セキュリティポリシーは、他のセキュリティポリシーを含む、又はそれに依存することがある。CKMS 設計は、他のどのセキュリティポリシーが、CKMS の適切でセキュアな運用を実施するために要求されているのかを記載すべきである。例えば、CKMS は、その CKMS 自身の保護を保証するために必要な全ての物理的保護及びアクセスコントロールを提供するように、設計及び実装されるかもしれない。また、CKMS の外部の物理セキュリティ及びアクセスコントロールが、CKMS が設置され運用される施設によって提供されることを仮定（及び要求）して、CKMS が設計されるかもしれない。コンピュータシステムは、独自のコンピュータセキュリティポリシーに従って構築されることも多い。組織は、適切な部門に対するポリシーを管理及び実施する役割を規定する論理的な体系の中で、ポリシーを作成すべきである。

FR:4.4 CKMS 設計は、CKMS セキュリティポリシーをサポートする他の関連するセキュリティポリシーを明記しなければならない。

4.5 ポリシー間の相互関係

CKMS 設計者は、CKMS 製品又は CKMS サービスを調達し使用する組織の様々なポリシーを意識すべきである。CKMS 設計者は、単一の組織のシンプルな鍵管理ポリシーを実施するシンプルな CKMS を設計するかもしれないし、様々なセキュリティポリシーをサポートできる複雑な CKMS を設計するかもしれない。

FR:4.5 CKMS 設計は、CKMS 設計によってサポートされるポリシーと、その設計によってどのようにサポートされるのかの要約を明記しなければならない。

4.6 個人の説明責任（Personal Accountability）

個人の説明責任（Personal accountability）についてのポリシーは、機微な情報にアクセスする各個人が自らの行動に対しての説明責任を課されていることを要求する。個人の説明責任（Personal accountability）は、CKMS の特定の機能となる情報管理ポリシーの、要求事項であるかもしれない。

CKMS 設計者は、CKMS が個人の説明責任（personal accountability）をサポートすることを意図するかどうかを決定すべきである。もし意図するのであれば、鍵及びメタデータの管理に対する説明責任をサポートするためのメカニズムを CKMS 内に提供すべきである。

FR:4.6 CKMS 設計は、個人の説明責任（personal accountability）が CKMS でサポートされるかどうか、及びどのようにサポートされるかを明記しなければならない。

4.7 匿名性、連結不可能性（Unlinkability）、観測不可能性（Unobservability）

情報管理ポリシーは、セキュアな情報処理システムのユーザに匿名性や連結不可能性（unlinkability）、観測不可能性（unobservability）を保証できると記述するかもしれない。匿名性は、パブリックなデータが所有者と関係付けることができないことを保証する。連結不可能性（unlinkability）は、情報処理システム内の 2 つ以上の関連する事象が、互いに関係付けることができないことを保証する。最後に、観測不可能性（unobservability）は、観測者がトランザクションに関係する当事者の識別子（ID）を特定又は推定することができないことを保証する。

FR:4.7 CKMS 設計は、CKMS でサポートできる匿名性、連結不可能性（unlinkability）、及び観測不可能性（unobservability）に関するポリシーを明記しなければならない。

4.7.1 匿名性

エンティティに対してプライバシーを提供するため、適用されるプライバシーに関する法律を遵守するため、又はセキュリティを強化するために、CKMS は、トランザクションに参加するエンティティの観点で、CKMS トランザクションの匿名性を要求することがある。また、プライバシー上の理由で、CKMS は、鍵やメタデータとエンティティとを関連付けるときに匿名性を要求することもある。

FR:4.8 CKMS 設計は、どの CKMS トランザクションが匿名性保護を提供している、又は提供可能であるのかを明記しなければならない。

FR:4.9 CKMS 設計は、匿名性の保証を提供する場合、CKMS トランザクションの匿名性保証をどのように達成するのかを明記しなければならない。

4.7.2 連結不可能性（Unlinkability）

エンティティに対してプライバシーを提供するため、適用されるプライバシーに関する法律を遵守するため、又は（トランザクションに誰が関連付けられているのかを推定することに対して保護することによって）セキュリティを強化するために、CKMS は、トランザクションに参加するエンティティの観点で、CKMS トランザクションに対する連結不可能性（Unlinkability）保護を提供することがある。

FR:4.10 CKMS 設計は、どの CKMS トランザクションが連結不可能性（unlinkability）保護を提供している、又は提供可能であるのかを明記しなければならない。

FR:4.11 CKMS 設計は、CKMS トランザクションの連結不可能性（unlinkability）をどのように達成するのかを明記しなければならない。

4.7.3 観測不可能性（Unobservability）

エンティティに対してプライバシーを提供するため、適用されるプライバシーに関する法律を遵守するため、又は（開示が望ましくないかもしれないあらゆる情報を推定することに対して保護するこ

とによって）セキュリティを強化するために、CKMS は、トランザクションを開始又は参加するエンティティの観点で、CKMS トランザクションの観測不可能性（unobservability）を提供することがある。

FR:4.12 CKMS 設計は、どの CKMS トランザクションが観測不可能性（unobservability）保護を提供している、又は提供可能であるのかを明記しなければならない。

FR:4.13 CKMS 設計は、CKMS トランザクションの観測不可能性（unobservability）をどのように達成するのかを明記しなければならない。

4.8 法律、ルール、規制

組織のセキュリティポリシーは、CKMS が使用される地域、州、及び国家の法律、ルール及び規制に従うべきである。CKMS が国際的に使用できるように設計される場合、各国の制限に従うことができる十分な柔軟性を持っているべきである。

FR:4.14 CKMS 設計は、CKMS が使用されることを意図する国名や地域名、及び CKMS に適用される可能性があるあらゆる法的規制を明記しなければならない。

4.9 セキュリティドメイン

セキュリティドメインとは、それぞれの CKMS が同じセキュリティポリシー（“ドメインのセキュリティポリシー”として知られる）の下で運用されるような、CKMS も含むエンティティの集合である。2 つの互いに信頼するエンティティが同じセキュリティドメインに属しているとき、それらのエンティティは、ドメインのセキュリティポリシーが要求する保護を提供しながら鍵及びメタデータを交換できる。

2 つのエンティティが異なるセキュリティドメインに属しているとき、それらのエンティティは異なるドメインのセキュリティポリシーの下で運用されているため、交換した鍵及びメタデータに対して同等の保護を提供することができないかもしれない。しかしながら、ドメインのセキュリティポリシーが完全に同一でない場合でも、あるドメインに属するエンティティが別ドメインに属する他のエンティティに鍵及びメタデータを送ることができる状況もある。

セキュリティドメインの例は、公開鍵証明書（[X.509]を参照）を発行する公開鍵基盤（PKI）である。PKI はひとつ以上の文書化された証明書ポリシーの下で運用され、それぞれの公開鍵証明書はその証明書が有効であるための証明書ポリシーを含んでいる。依拠当事者（証明書のユーザ）は、証明書を検査し、その証明書が許容可能なセキュリティを提供しているか判断することができる。しかしながら、異なる PKI ドメインのエンティティが通信しようとして互いの証明書を使用する場合、証明書が使用される前に、互いの PKI ドメインの証明書ポリシーを検査し、同等のセキュリティを提供しているかどうかを検証すべきである。

4.9.1 データ交換のための条件

エンティティがセキュアに鍵やメタデータを他のエンティティに送信したいとき、いくつかの条件を満たさなければならない：

a) 情報を送信及び受信する手段（通信チャネルと呼ばれる）がなければならない。

b) 2 つのエンティティは相互運用可能な暗号機能を有していなければならない（例えば、同じ鍵長を用いる機能的に同一な暗号化／復号アルゴリズム）。

c) 2 つのエンティティは同等（ただし、同一ではなく異なる場合もある）のセキュリティポリシーであると承認しなければならない。

d) 2 つのエンティティは、自身のセキュリティポリシーを実施するために互いに（ネットワーク上の他のエンティティも含めての場合もある）信頼しなければならない[2]。

エンティティが同じセキュリティドメインに属している場合、これらの各条件が満たされている可能性は高い。しかし、エンティティが同じセキュリティドメインに属していない場合、これらの条件が満たされている可能性は低くなる。この節以降では、条件 a)、b)、d）は満たされているものと仮定し、以下の説明は条件 c）に焦点を当てる。

FR:4.15 CKMS 設計は、同等だが異なるセキュリティ保護を提供するとみなせる他のセキュリティドメインに属するエンティティ間での鍵及びメタデータの交換を許可する機能の設計を明記しなければならない。

4.9.2 保護の保証（Assurance of Protection）

保護の保証は、鍵やメタデータを認可されない開示（窃取）から保護すること、鍵やメタデータの認可されない改変（改ざん）から保護すること、及びアプリケーションに要求された際の鍵やメタデータのソース（送信者）及びディスティネーション（受信者）を知ることを含む。セキュリティドメイン A に属するエンティティ A がセキュリティドメイン B に属するエンティティ B に暗号鍵やメタデータを送信しようとしており、かつ上記の条件 a)、b）及び d）を満たしていると仮定する。また、エンティティ B も鍵やメタデータを受信し、その鍵やメタデータを、B 自身の鍵やメタデータを扱うのと全く同様に扱うつもりであると仮定する。これは、エンティティ B が、B 自身の鍵やメタデータに提供する保護と受信した鍵やメタデータに提供する保護を全く区別しないということである。ここで、エンティティ A が鍵やメタデータを送信する前に、ドメイン B のセキュリティポリシーにおける保護の要求事項が少なくともドメイン A のセキュリティポリシーと同等であることの確証を得る必要がある。また、エンティティ B も、ドメイン A のセキュリティポリシーにおける保護の要求事項が少なくともドメイン B のセキュリティポリシーと同等であることの確証を持ちたいと思うであろう。本質的には、2 つのドメインは同等のドメインのセキュリティポリシーを持っていなければならない。

エンティティ A からエンティティ B に送信されるデータに要求される保護の保証を図 8 に示す。

[2] 前もって 1 つ以上のエンティティによって保護されたデータを受信したエンティティは、相手エンティティが自身のセキュリティポリシーを適切に実施していると信頼しなければならない。

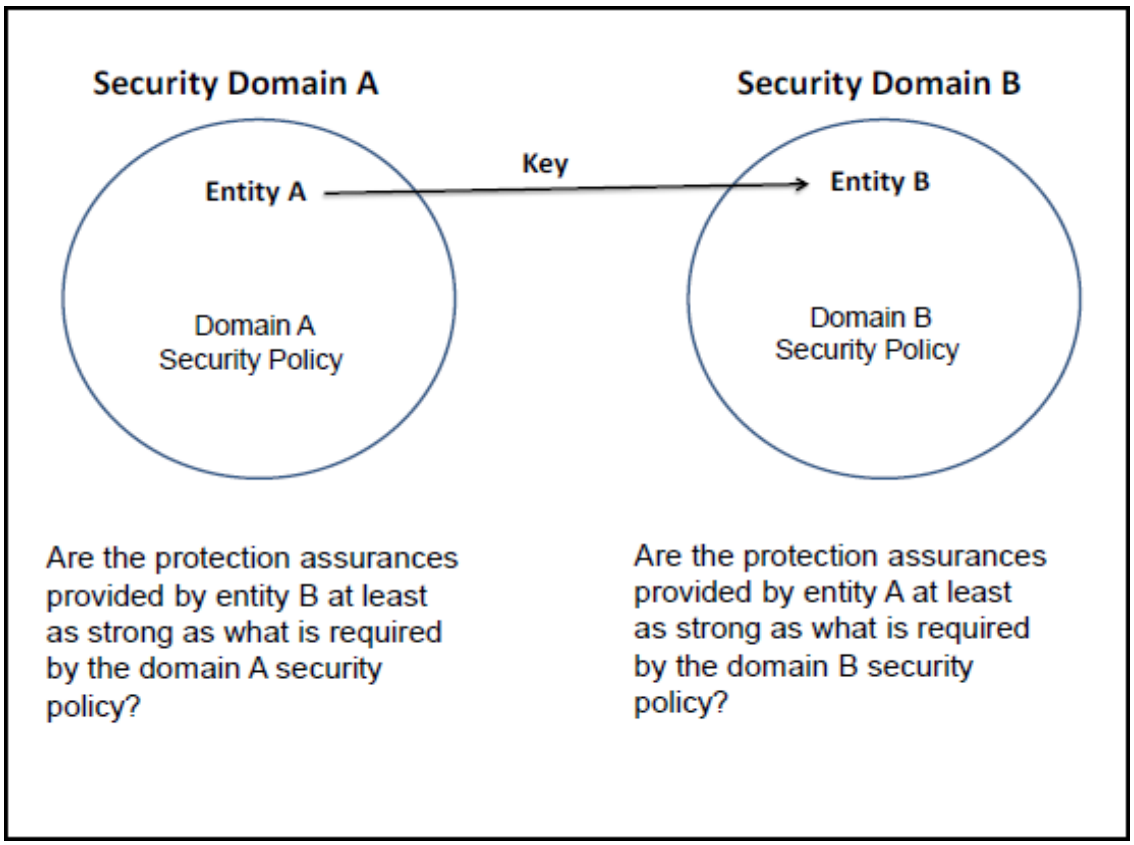

図 8 セキュリティドメイン間の保護の保証

FR:4.16 CKMS 設計は、鍵やメタデータを異なるセキュリティドメインに属するエンティティ間で共有するときに実施されるソース認証ポリシー（source authentication policy）とディスティネーション認証ポリシー（destination authentication policy）を明記しなければならない。

FR:4.17 CKMS 設計は、鍵やメタデータを異なるセキュリティドメインに属するエンティティ間で共有するときに実施される機密性と完全性のポリシーを明記しなければならない。

FR:4.18 CKMS 設計は、他のセキュリティドメインのエンティティと通信するときに要求される保証要件を明記しなければならない。

4.9.3 ドメインのセキュリティポリシーの同等性

提供されるセキュリティ保護に関して、それぞれのセキュリティドメインに責任を持つ 2 つのオーソリティ（authority）が、他方のセキュリティポリシーを自分自身のポリシーと同等であると受け入れることに同意できれば、2 つのセキュリティドメインは同等のセキュリティポリシーを持っているといえる。各ドメインに責任を持つオーソリティは、ドメインのセキュリティポリシーを受け入れる前に、慎重に調査しなければならない[3]。オーソリティが保護の同等性に同意できなければ、このプロセスは不可能であるかもしれない。セキュリティドメインに責任を持つオーソリティは、あらゆる潜在的な危殆化の影響を軽減するために、他のドメインと共有しようとしている鍵やメタデータ、ひいてはデータのセキュリティレベルを制限するかもしれない。エンティティ A とエンティティ B が鍵やメタデータを共有しようとしていて、セキュリティドメイン B がセキュリティドメイン A より弱いセキ

[3] セキュリティポリシーの同等性を決定するプロセスは、公開鍵基盤における認証局の相互認証プロセスと類似している。

ュリティポリシーを持っている場合、高度な CKMS は、少なくとも、エンティティ A にセキュリティに関して起こり得る影響を通知すべきである。

2 つのドメインのポリシーが同等であると判定された場合、適宜、一方のドメインのエンティティは他の同等のドメインに属するあらゆるエンティティとデータを共有してよい。

FR:4.19 CKMS 設計は、ドメイン間通信が許可される前に通信相手のドメインのセキュリティポリシーのレビューと検証をサポートするかどうか、またどのようにサポートするのかを明記しなければならない。

FR:4.20 CKMS 設計は、弱いポリシーを持つセキュリティドメインのエンティティとの通信がもたらす潜在的なセキュリティに関する影響をどのように検知、防止、又はエンティティに警告するのかを明記しなければならない。

4.9.4 第三者共有（Third-Party Sharing）

セキュリティドメイン A のエンティティ A とセキュリティドメイン B のエンティティ B が同等のドメインのセキュリティポリシーを持っていると仮定する。その場合、エンティティ A 及びエンティティ B は、ドメイン A あるいはドメイン B のいずれのドメインに属するあらゆる他のメンバーと鍵やメタデータを共有しても良いかもしれない。なぜなら、各ドメインは他方のドメインのセキュリティポリシーを受け入れているからである。しかしながら、ここでエンティティ B が第 3 エンティティであるドメイン C のエンティティ C とも鍵及びメタデータを共有すると仮定する。この場合、エンティティ A とエンティティ B はそれぞれのドメインのセキュリティポリシーが同等であることの保証があり、エンティティ B とエンティティ C にもそれぞれのドメインのセキュリティポリシーが同等であることの保証があることになる。エンティティ B がエンティティ A から受領した鍵やメタデータを自分自身の鍵やメタデータと同様に扱う場合、エンティティ A はエンティティ B と共有した鍵やメタデータが他の同等のセキュリティドメインとも共有され得ると認識するべきである。2 つのエンティティが互いのドメインのセキュリティポリシーの同等性を検査するとき、鍵、メタデータ及び他の情報を他のセキュリティドメインのエンティティと共有することに関する互いのポリシーに細心の注意を払うべきである。

4.9.5 マルチレベルのセキュリティドメイン

セキュリティドメインは、それぞれが同じマルチレベルのドメインのセキュリティポリシーをサポートしているエンティティを含んでいることがある。例えば、ドメインのセキュリティポリシーが、処理する鍵やメタデータに対して高レベルの保護と低レベルの保護を提供していることがある。この場合、セキュリティドメインは 2 つの分離したセキュリティドメインのような振る舞いをする。これは 2 つのレベルの保護を区別しなければならないためである。それぞれのエンティティは、高レベルのポリシーで保護された鍵やメタデータ（高レベル保護の鍵やメタデータ）が常に高レベルの保護を受け、低レベルのポリシーで保護された鍵やメタデータ（低レベル保護の鍵やメタデータ）が高レベル保護の鍵やメタデータと混同されることがなく、高レベル保護の鍵やメタデータが低レベル保護の鍵やメタデータと混同されることがないことを保証しなければならない。このような形態は、典型的にはマルチレベル OS が関与して行われる。図 9 を参照されたい。物理エンティティ B が 2 つの論理エンティティ（高レベル保護のエンティティ B_{HL} と低レベル保護のエンティティ B_{LL}）に分割されている。B_{HL} の鍵と B_{LL} の鍵の分離は OS によって論理的に保持されている（図において破線で示されている）。マルチレベルのセキュリティドメインの利点は、異なったセキュリティレベルで運用しているエンティティからの鍵やメタデータを処理できることである。

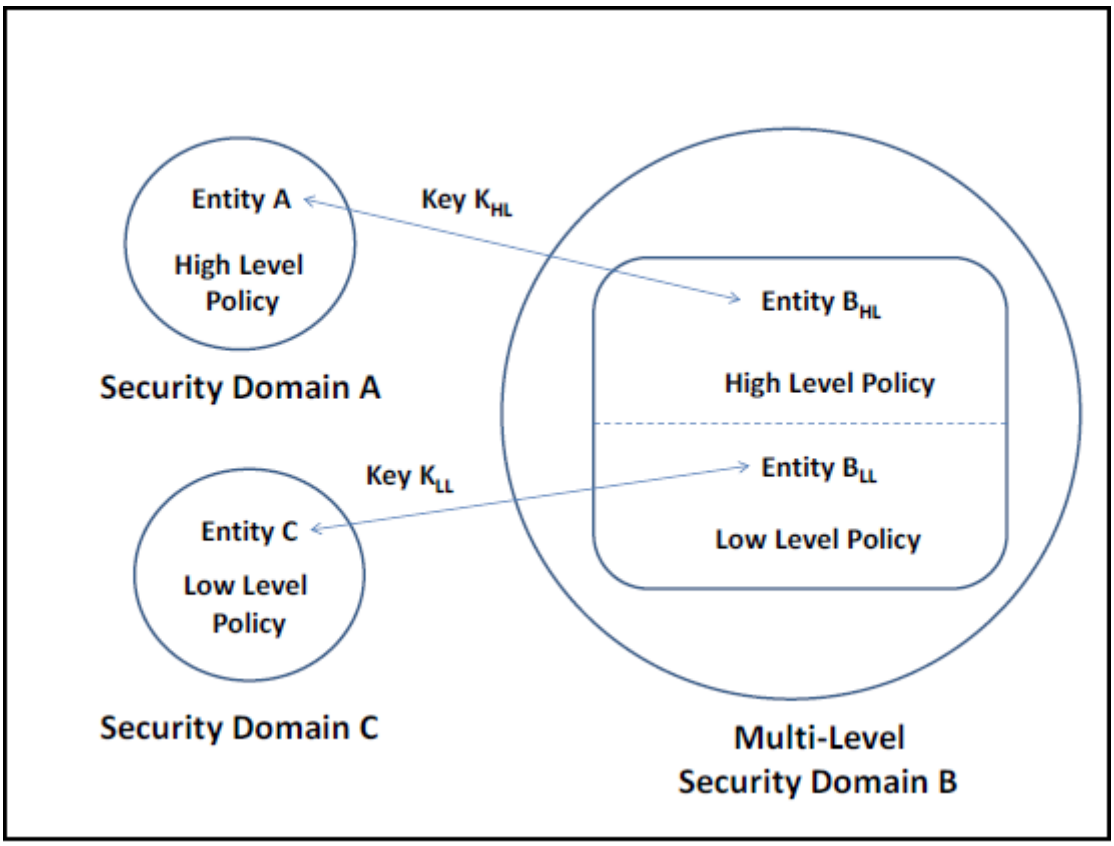

図 9 マルチレベルのセキュリティドメイン

FR:4.21 CKMS 設計は、マルチレベルのセキュリティドメインをサポートするかどうかを明記しなければならない。

FR:4.22 CKMS 設計は、サポートするセキュリティドメインのそれぞれのレベルを明記しなければならない。

FR:4.23 マルチレベルのセキュリティドメインをサポートしている場合、CKMS 設計は、それぞれのセキュリティレベルに属する鍵及びメタデータの分離をどのように保持しているのかを明記しなければならない。

4.9.6 アップグレードとダウングレード

ある条件の下で、ドメインオーソリティは、低レベルセキュリティドメイン（より低い保護しか提供しないドメイン）のエンティティからの鍵やメタデータを受け入れ、その後、自分自身のドメインのセキュリティポリシーで要求される高いレベルの保護を行うことができると決定するかもしれない。このプロセスをアップグレードと呼ぶ。アップグレードはリスクなしではできず、高レベルセキュリティドメインに責任を持つオーソリティが、低レベルドメインからの鍵やメタデータのソース及び信頼性（authenticity）に信頼と確信を持っている場合にのみ行うべきである。ドメインオーソリティによる判断の誤りは、そのドメインに属するエンティティへのセキュリティ上の危殆化を招きかねない。同様に、ある条件の下で、高レベルセキュリティドメインのドメインオーソリティが、低レベルドメインに属するエンティティに鍵やメタデータを伝えようとする（すなわち、ダウングレードする）かもしれない。この場合、高レベルドメインのドメインオーソリティは、ダウングレードして伝えられる鍵やメタデータが受信者によって提供される低レベルのセキュリティしか必要としないとの確信を持つべきである。

FR:4.24 CKMS 設計は、鍵及びメタデータのアップグレード又はダウングレードをサポートするかどうか、及びどのようにサポートするのかを明記しなければならない。

FR:4.25 CKMS 設計は、アップグレード又はダウングレード機能をどのようにドメインオーソリティ（domain authority）に制限しているかを明記しなければならない。

4.9.7 ドメインのセキュリティポリシーの変更

ドメインのセキュリティポリシーは、時々、改訂又は更新されることが望ましい。鍵及びメタデータ要素に提供する保護をアップグレードするとの経営層の決定による場合、別のセキュリティドメインと同等にしようとした場合、又は新しいアプリケーションをサポートする場合といった契機で更新は行われる。

ある CKMS では、異なるドメインのエンティティとの通信を許可するためのドメインのセキュリティポリシーを設定できるように設計されているものがあるかもしれない。例えば、様々なアプリケーションをサポートするために使用される鍵やメタデータの管理機能を選択することを、特定の管理職員に許可しているセキュリティドメインがあるかもしれない。このようなドメインは設定可能（configurable）であるという。特定のドメインのセキュリティポリシーの変更が設定可能なシステムの機能の範囲内であっても、あらゆるポリシーの変更は、変更が実行される前にドメイン管理職員が必ず承認するべきである。

FR:4.26 CKMS 設計は、異なるドメインのセキュリティポリシー及び異なるアプリケーションをサポートするように、鍵やメタデータの管理機能を設定することができるかどうか、及びどのように設定するのかを明記しなければならない。

FR:4.27 CKMS 設計は、異なるセキュリティドメイン間のエンティティ同士との通信に適応するために、再設定によるドメインのセキュリティポリシーの変更をサポートしているかどうか、及びどのようにサポートできるかを明記しなければならない。

5 役割及び責任

CKMS には、特定の管理者やユーザや、運用者の役割を実行する人間に対するインタフェースが必要になる場合がある。それぞれの役割はそのために定義された特定の認可が必要で、その役割を実行する人間には、その役割の責任を果たすために必要な鍵及びメタデータを管理する一連の機能へのアクセス権限が提供されているべきである。CKMS の役割として考えられる例には、以下のものが含まれる（なお、これらだけに限らない）：

a) **システムオーソリティ（System Authority）**：システムオーソリティは、CKMS の全体運用及びセキュリティについて、経営幹部層（例えば、最高情報責任者（CIO））に対して責任を負う。システムオーソリティは CKMS 運用に関する全ての役割を管理する。運用に関する役割とは、CKMS を直接運用する役割である。

b) **システム管理者**：システム管理者は、鍵以外の CKMS に対する役員・従業員、日々の運用、訓練、メンテナンス、及び関連する管理について責任を負う。システム管理者は、CKMS の運用及び使用に関係する全ての人員について、個人識別子（ID）の初期検証とその後の適切な識別子（ID）の確立に責任を負う。CKMS の運用及び使用に関係する人員には、ユーザ、セキュリティ監査者、暗号責任者（cryptographic officer）、鍵保管者（key custodian）、オペレータ、メンテナンス要員、及びエージェントを含み、これらのシステムのデータへのアクセス又は CKMS の使用を求める人々のクレデンシャルは入念に検査する必要がある。

c) **暗号責任者（Cryptographic Officer）**：暗号責任者には、CKMS 及びその暗号モジュールにある暗号の初期化及び管理機能を実行する権限が与えられている。

d) **ドメインオーソリティ（Domain Authority）**：ドメインオーソリティは、ドメインのセキュリティポリシーの定義及び承認、その後の他のセキュリティドメインとの通信に必要な条件の決定、及びその条件が満たされていることの保証に関して責任を負う。

e) **鍵保管者（Key Custodian）**：鍵保管者は、鍵又は分割鍵（key splits）を暗号モジュール内に配付や装填するために指名される。複数の鍵保管者が、マルチパーティコントロール及び鍵分割を行うために使われることがある（6.7.4 節及び 6.7.5 節を参照）。

f) **鍵所有者**：鍵所有者とは、暗号鍵又は鍵ペアを使用する権限を与えられ、暗号鍵又は鍵ペアと関連付けられた識別子（identifier）を持つエンティティ（例えば、個人、グループ、組織、デバイス、又は暗号モジュール）のことである。公開鍵とプライベート鍵の鍵ペアでは、関連付けの確立は典型的には登録プロセスを通して行われる。対称鍵は、特定の単一所有者が持つことも、複数の所有者が同じ鍵を共有していることもある。

g) **CKMS ユーザ**：CKMS ユーザは、アプリケーションをサポートする際に鍵管理機能が必要な場合に CKMS を利用する。CKMS ユーザは、多くの場合、鍵所有者である。

h) **監査責任者（Audit Administrator）**：監査責任者は、CKMS のセキュリティと権限付けられた運用について全ての観点で CKMS を監査することに責任を負う。監査責任者は、あくまでイベントログを管理及びレビューするだけで、CKMS に関して何ら運用上の責任を負うべきでない。監査責任者は、自分自身の鍵以外のあらゆる運用鍵に対してアクセスの権限を持つべきでない。

i) **登録エージェント**：登録エージェントは、新しいエンティティの登録、並びに彼らの鍵と識別子（identifier）及び（もしあれば）他の選択されたメタデータとの関連付けに責任を負う。また、登録エージェントは、エンティティ情報、鍵、及びメタデータを CKMS が使用するデータベースに入力することもある。

j) **鍵復元エージェント（Key-Recovery Agent）**：鍵復元エージェントは、CKMS セキュリティポリシーに従って要求エンティティの ID 認証及び認可を行った上で、バックアップ又はアーカイブストレージから鍵を復元することを許可されている（6.4.15 節及び 6.4.17 節を参照）。

k) **CKMS オペレータ（CKMS Operator）**： CKMS オペレータは、システム管理者の指示に従って CKMS を操作（例えば、CKMS の起動、パフォーマンス監視、及びバックアップ実行）することを認可されている。

それぞれの役割に複数の個人が割り当てられることも、1 人の個人に複数の役割が割り当てられることもある。しかしながら、ある役割に関しては、同一の人間に両方の役割が同時に割り当てられることがないように分離されるべきである。例えば、管理上の誤用又は不正使用を検知するため、監査ログはシステム管理者とは異なる人によって管理されるべきである。加えて、長期の不正使用の可能性を最小化するために、役割を交代で割り当てることも賢明である。

FR:5.1 CKMS 設計は、CKMS に用いられているそれぞれの役割と責任、及びそれぞれの役割にどのようにエンティティが割り当てられるのかを明記しなければならない。

FR:5.2 CKMS 設計は、CKMS に用いられているそれぞれの役割を満たしているエンティティが使用できる鍵及びメタデータの管理機能（6.4 節を参照）を明記しなければならない。

FR:5.3 CKMS 設計は、どの役割が役割分離を必要とするのかを明記しなければならない。

FR:5.4 CKMS 設計は、役割分離を必要とする役割に対してその分離がどのように保持されるのかを明記しなければならない。

FR:5.5 CKMS 設計は、セキュリティ違反が認可された役割を実行する個人（内部者）によるのか、認可された役割がない人（外部者）によるのかを、特定するための全ての自動化された対策を明記しなければならない。

6　暗号鍵及びメタデータ

6.1　　鍵のタイプ

一般に、暗号鍵は特性と用途に応じて分類される。鍵の特性には、公開、プライベート、対称がある[4]。また、静的（すなわち、長期的）又は一時的（1 つのセッション又は鍵管理トランザクションでのみ使用される）という鍵の特性を持つこともある。鍵の用途には、署名、認証、暗号化／復号、鍵ラッピング、RNG（乱数生成）、マスタ鍵、鍵配送、鍵合意、及び認可が含まれる。［SP 800-57-part1］は 20 個の異なった鍵タイプを挙げている。21 個の鍵タイプを以下の表 1 に示す（この表では SP 800-57 Part1 にある 1 つの複合的な鍵タイプを 2 つのシンプルな鍵タイプに分割している）。本段落でのイタリック体で表記している項目は、表中の鍵タイプ名を構成する実際の用語であることに注意されたい。CKMS では、これら又は他の鍵タイプを設計において使用するかもしれない。

鍵タイプ
1）署名プライベート鍵（Private Signature Key）
2）署名公開鍵（Public Signature Key）
3）認証対称鍵（Symmetric Authentication Key）
4）認証プライベート鍵（Private Authentication Key）
5）認証公開鍵（Public Authentication Key）
6）データ暗号化／復号対称鍵（Symmetric Data Encryption/Decryption Key）
7）鍵ラッピング対称鍵（Symmetric Key Wrapping Key）
8）乱数生成対称鍵（Symmetric RNG Key）
9）乱数生成プライベート鍵（Private RNG Key）
10）乱数生成公開鍵（Public RNG Key）
11）マスタ対称鍵（Symmetric Master Key）
12）鍵配送プライベート鍵（Private Key Transport Key）
13）鍵配送公開鍵（Public Key Transport Key）
14）鍵合意対称鍵（Symmetric Key Agreement Key）
15）鍵合意静的プライベート鍵（Private Static Key Agreement Key）
16）鍵合意静的公開鍵（Public Static Key Agreement Key）
17）鍵合意一時的プライベート鍵（Private Ephemeral Key Agreement Key）

[4] 本文書で鍵が非対称か対称かが明示されていない場合は、非対称か対称かのどちらかであることが暗黙に想定されている。

18）鍵合意一時的公開鍵（Public Ephemeral Key Agreement Key）
19）認可対称鍵（Symmetric Authorization Key）
20）認可プライベート鍵（Private Authorization Key）
21）認可公開鍵（Public Authorization Key）

表 1 鍵タイプ

FR:6.1 CKMS 設計は、使用されているそれぞれの鍵タイプを明記及び定義しなければならない。

6.2 鍵メタデータ

本節では、鍵と関連付けられ得るメタデータの一覧を作り、記述する。鍵のメタデータは、CKMS によって明示的に記録され管理されている特定の鍵に関連付けられている情報として定義される。本フレームワークでは、特定のメタデータ要素のセットが関連付けられている鍵を“the key”で表す。

鍵との信頼関係に適切であろうメタデータは、多くの要素（鍵タイプ、鍵ライフサイクル状態及び CKMS セキュリティポリシーを含む）を基に、CKMS 設計者によって選択されるべきである。CKMS は、与えられた鍵に適用可能な全てのメタデータを関連付ける必要はなく、また一部又は全ての鍵にいかなるメタデータも関連付けないかもしれない。6.2.1 節の項目 u）を参照されたい。

6.2.1 メタデータ要素

以下に挙げるものが典型的なメタデータ要素及びその説明である：

a) **鍵ラベル（Key Label）**：鍵ラベルはテキスト文字列であり、人間が読解可能で、かつ機械が解釈可能であるかもしれないような、鍵の記述子のセットを提供する。鍵ラベルの例には、“Root CA Private Key 2009-29”や“Maintenance Secret Key 2005.”などがある。

b) **鍵識別子（Key Identifier）**：本要素は、CKMS が鍵の集合から特定の鍵を選択するために使用される。鍵識別子は一般的にセキュリティドメインにおいて固有である。公開鍵及びプライベート鍵では、鍵識別子は、公開鍵のハッシュ値又はハッシュ値の一部であることも、あるいは CKMS によって割り当てられることもある。

c) **所有者識別子（Owner Identifier）**：本要素は、鍵を所有するエンティティの識別子を明記する。

d) **鍵ライフサイクル状態（Key Lifecycle State）**：鍵ライフサイクル状態は、暗号鍵が現状で許可されている条件を表した有限個の状態の集合のひとつである（6.3 節を参照）。

e) **鍵フォーマット指定子（Key Format Specifier）**：本要素は鍵のフォーマットを指定するために使用される。これはオブジェクト識別子を使用し、構造を参照することで達成される。例えば、RSA 公開鍵は法（modulus）と公開指数（public exponent）から構成される。フォーマット指定子は、これらの 2 つの値が格納される順序及びそれぞれの値のコード化フォーマットを指定すべきである。Internet Engineering Task Force（IETF）は、DSA、DH、RSA、EC、RSAPSS 及び RSAOAEP の鍵といった、異なる形式の公開鍵を格納するためのオブジェクト識別子を定義している。オブジェクト識別子及び関連する公開鍵の構造は、以下のインターネット RFC に定義されている：[RFC 3279]、[RFC 4055]、及び [RFC 5480]。

f) **鍵生成に使用した製品（Product used to create the Key）**：本要素は、どの暗号製品が鍵生成に使用されたかを明記する。

g) **鍵を使用する暗号アルゴリズム（Cryptographic Algorithm using the Key）**：本要素は、鍵を使用することを意図されている暗号アルゴリズムを指定する。例には、DSA、ECDSA、RSA、AES、TDEA 及び HMAC-SHA1 が含まれる。

h) **スキーム又は暗号利用モード（Scheme or Modes of Operation）**：本要素は、鍵を使用する暗号機能を実行するための適用可能なスキーム又は暗号利用モードを定義する。非対称アルゴリズムでは、有限体、binary field（標数 2 の体）又は楕円曲線（EC）上の離散対数問題の演算を指定するかもしれない。対称アルゴリズムでは、本フィールドは、鍵を使用するときのブロック暗号で使用される暗号利用モードを定義するかもしれない。暗号利用モードの例は、Electronic Code Book（ECB）、Cipher Block Chaining（CBC）、Output Feedback Mode（OFB）、及び Counter with Cipher Block Chaining-Message Authentication Mode（CCM）である。さらなる情報は［SP 800-38A］から［SP 800-38F］を参照されたい。

i) **鍵パラメタ（Parameters for the Key）**：本要素は、もしあれば、鍵のパラメタを指定する。例えば、DSA 鍵には以下のドメインパラメタがある：大きい素数（p）、 小さい素数（q）及び生成元（g）。

j) **鍵長（Length of the Key）**：本要素は鍵の長さをビット（又はバイト）で指定する。例としては、RSA の法の 2048 ビットや楕円曲線暗号の鍵の 256 ビットがある。

k) **鍵／アルゴリズム組のセキュリティ強度（Security Strength of the Key/Algorithm Pair）**：本要素は、暗号アルゴリズムを破る（すなわち、暗号解読する）ために必要な計算量（すなわち、操作数の 2 を底とする対数）を表す数値である。例えば、168 ビット（パリティビットを含まない）の TDEA 鍵では、セキュリティ強度は 112 ビットと規定されている。2048 ビット RSA の法では、セキュリティ強度は 112 ビットと規定されている。鍵／アルゴリズム組のセキュリティ強度は、以前に知られていなかった攻撃が発見されると低下することがある。

l) **鍵タイプ（Key Type）**[5]：本要素は鍵のタイプを特定する。鍵タイプは 6.1 節で説明する。

m) **鍵に対する適切なアプリケーション（Appropriate Applications for the Key）**：本要素は、鍵を使用してよいアプリケーションを指定する。例には、Kerberos、署名付き E メール、信頼されるタイムスタンプ（Trusted Time Stamp）、コード署名、ファイル暗号化、及び IPSEC がある。

n) **鍵セキュリティポリシー識別子（Key Security Policy Identifier）**：本要素は、鍵又は鍵タイプに適用できるセキュリティポリシーを特定する。鍵セキュリティポリシーは、生成から破棄までの鍵ライフサイクル全体にわたって鍵又は鍵タイプを保護するために使用するセキュリティコントロール一式である（6.7 節及び［RFC3647］を参照）。鍵セキュリティポリシーは、典型的には CKMS 組織によって登録されたオブジェクト識別子で表現される。個々の鍵又は鍵タイプに対する鍵セキュリティポリシーは、CKMS セキュリティポリシーの一部分であり、CKMS セキュリティポリシーと整合しているべきである。

o) **鍵アクセスコントロールリスト（Key Access Control List（ACL））**[6]：アクセスコントロールリストは、鍵及びメタデータの管理機能で制限された通りに、鍵へのアクセスや使用ができるエンティティを特定する（6.7 節を参照）。本フレームワークは、アクセスコントロールの構造を指定しない。アクセスコントロールリストの構造の例として、0 個以上のアクセスコントロールエントリで構成される Microsoft Windows ファイル／フォルダアクセスコントロールリスト、Sun ファイルシステムアクセスコントロールリスト、及びリストの形ではない UNIX 保護ビットがある。相互運用性が望ましい場合、以下の項目の標準化が必要であると考えられる：アクセスコントロールエントリ間のセパレータのシンタックス及びセマンティクス、アクセス

[5] 用途は鍵タイプを定義する 2 つの要素の内の 1 つであるので、鍵タイプは鍵の用途も暗黙的に示している。したがって、鍵タイプによって暗黙的に示される鍵の用途は、鍵の使途と一致すべきである。

[6] ACL には、認可された当事者の識別子、（作成、初期化、使用、入力、出力、更新、置換、失効、削除等の）アクセスモード又は許可又は認可、各アクセスモードの委任権限、及び各アクセスモードの有効期間が含まれる。

コントロールエントリ内でのエンティティと“アクセスモード”の順序、エンティティ識別子、及び異なる“アクセスモード”のビット指定。相互運用性が要求される場合、これらの項目は適切に詳細設計仕様に含まれるべきである。

p) **鍵使用カウント（Key Usage Count）**：本要素は、鍵が使用された回数を示す。

q) **親鍵（Parent Key）**：本要素は、メタデータに関連付けられた鍵の導出元の鍵を指す。例えば、新しい鍵（即ち、子鍵）は TLS マスタ秘密鍵（すなわち、親鍵）とそのメタデータから導出されることがある。

本要素は 2 つの下位要素を持つことがある。

i. 鍵識別子（Key Identifier）：親鍵の識別子（上記の項目 b）を参照)。

ii. 関係特性（Nature of the Relationship）：本要素は、親鍵がどのように子鍵と関連しているかを特定する。関係の例は、入力のひとつとして親鍵を使って子鍵を生成するときに使用された算術関数である。その関係は、算術関数の識別子によって示されるかもしれない。

r) **鍵機微性（Key Sensitivity）**：本要素は、鍵の機微度又は重要度を明記する。これは、リスクレベル（例えば、低、中、高）又は機密区分レベル（例えば、Confidential、Secret、Top Secret）に関係し得る。

s) **鍵保護（Key Protections[7]）**：本要素は、鍵に対する完全性、機密性及びソース認証（source authentication）の保護を明記する。公開鍵証明書は、CA のデジタル署名が完全性保護とソース認証の両方を提供する鍵保護の例である（[X.509] を参照)。対称鍵及びそのハッシュ値を共に暗号化したものは機密性と完全性の保護の例である。鍵及びそのメタデータを外部エンティティから受信した場合は、鍵及びメタデータが運用で使用される前にそれらが保護されているかどうかを検証するべきである。一般に、1 つの暗号機能（例：HMAC 又はデジタル署名）が完全性保護とソース認証の両方を提供するために使用される。

本要素は、いくつかの下位要素を持つことがある：

i. 完全性保護に使用されるメカニズム（例：ハッシュ値、MAC、又はデジタル署名）

ii. 機密性保護に使用されるメカニズム（例：鍵ラッピング、又は鍵配送）

iii. ソース認証（source authentication）に使用されるメカニズム（例：MAC、又はデジタル署名）

iv. 特定の非暗号学的な信頼プロセスによって実施される保護の表示

t) **メタデータ保護（Metadata Protections）（鍵保護のサブセットであることも、鍵保護と異なることもある）**：本要素は、関連付けられたメタデータの完全性、機密性及びソース認証（source authentication）を保護するために使用されるメカニズムを明記する。一般に（特に、一緒に伝送又は格納される場合には)、同じメカニズムを使用して鍵及びそのメタデータを保護する。

本要素は、いくつかの下位要素を持つことがある：

i. 完全性保護に使用されるメカニズム（例：ハッシュ値、MAC、又はデジタル署名）

ii. 機密性保護に使用されるメカニズム（例：暗号化）

iii. ソース認証（source authentication）に使用されるメカニズム

iv. 特定の非暗号学的な信頼プロセスによって実行される保護の表示

---

[7] 鍵は、複数のタイプの保護（例えば、完全性及び機密性）を有することができる。本フレームワークは、同じセキュリティ処理（service）のために複数の暗号化メカニズム（例えば、完全性のためのデジタル署名及び MAC）の使用を認めている。

u) **信頼関係保護（Trusted Association Protections）（すなわち、鍵とメタデータの信頼関係がどのように保護されるか）（上記の項目 s）の鍵保護の一部になり得る）**： 上記の項目 s）で挙げられている保護によって、鍵及びメタデータがひとつの集約された項目として保護されている場合、この情報が暗黙的に提供されている。そうでなければ、それぞれの信頼関係保護に対して、以下の項目が提供されるべきである：

i. 完全性保護に使用されるメカニズム（例：ハッシュ値、MAC、デジタル署名、又は信頼プロセス）

ii. ソース認証（source authentication）に使用されるメカニズム（例：暗号学的メカニズム又は非暗号学的な信頼プロセス）

v) **日時（Date Times）：**鍵のライフサイクル状態遷移のためのいくつかの重要な日時が存在する：

i. 生成日（Generation date）：鍵が生成された日時

ii. 関連付け日（Association date）：初めて鍵がメタデータと関連付けられた日時

iii. 活性化日（Activation date）：鍵が最初に使用された日時

iv. 活性化予定日（Future activation date）：鍵が最初に使用される予定の日時

v. 更新日（Renewal date）：公開鍵が更新され、より長い時間使用することを許可された日時。例えば、古い証明書で提供されていた公開鍵と同じ鍵に対して新しい証明書が生成されたことによって更新される（6.4.7 節を参照）。

vi. 更新予定日（Future renewal date）：公開鍵が更新され、より長い時間使用することを許可される予定の日時（例えば、古い証明書で提供されていた公開鍵と同じ鍵に対して新しい証明書が生成されることによって更新される）

vii. 前回鍵再設定日（Date of the last rekey）：置き換え対象の鍵を、その鍵とは完全に独立して生成された新しい鍵に置き換えた日時

viii. 鍵再設定予定日（Future rekey date）：置き換え対象の鍵を、その鍵とは完全に独立するように生成される新しい鍵に置き換える予定の日時

ix. 鍵最終使用日（Date of the last usage of the key）：鍵が最後に使用された日時

x. 非活性化日（Deactivation date）：鍵が非活性化された日時

xi. 非活性化予定日（Future deactivation date）：鍵が非活性化される予定の日時

xii. 有効期限（Expiration date）：鍵の使用可能なライフタイムが永久に終了した日時

xiii. 失効日（Revocation date）：鍵がこの時以降は有効でないと見なされた日時

xiv. 危殆化日（Compromise date）：鍵が危殆化したと判明又は疑われ、更新ではなく置き換えると記録された日時

xv. 破壊日（Destruction date）：鍵が破壊された日時

xvi. 破壊予定日（Future destruction date）：鍵が破壊される予定の日時

w) **失効理由（Revocation Reason）**：鍵が失効される場合、本要素は失効理由を明記する。例としては、敵対者が鍵を所持していることによる危殆化、敵対者が鍵を含む暗号モジュールを所持していることによる危殆化、鍵の喪失、鍵を含む暗号モジュールの喪失、鍵危殆化の疑い、鍵所有者の支援組織離脱、及び所有者による鍵の誤使用がある。

上記に挙げたメタデータ要素で使用された日時は、様々な CKMS トランザクションの日時と同様、正確で、かつ Network Time Protocol（NTP）サーバのように権威ある情報源を元にすることが要求

されるかもしれない。さらに、いくつかのトランザクションは信頼される第三者機関によるタイムスタンプを必要とするかもしれない。そのようなタイムスタンプは［RFC 3161］及び［SP 800-102］に記載されている。

FR:6.2 システムで使用されているそれぞれの鍵タイプに対して、CKMS 設計は、信頼関係のために選択される全てのメタデータ要素、メタデータ要素が作成され鍵との関連付けが満たされている状況、及び関連付けの手段（すなわち、暗号メカニズム又は信頼プロセス）を明記しなければならない。

FR:6.3 メタデータ要素の鍵保護（Key Protections）（上記の項目 s）で使用されるそれぞれの暗号メカニズムに対して、CKMS 設計は、以下を明記しなければならない：

i. 暗号アルゴリズム：上記の項目 g）を参照
ii. 鍵パラメタ：上記の項目 i）を参照
iii. 鍵識別子：上記の項目 b）を参照
iv. 保護値（protection value）：この要素は、完全性保護、機密性保護、又はソース認証（source authentication）の保護値（protection value）を含む。例えば、適切に実装された MAC 又はデジタル署名技術は、完全性保護やソース認証（source authentication）を提供し得る。
v. 保護が適用された時期
vi. 保護が検証された時期

FR:6.4 メタデータ要素の鍵保護（Key Protections）（上記の項目 s）で使用される暗号学的ではないそれぞれの信頼プロセスに対して、CKMS 設計は、以下を明記しなければならない：

i. 他のプロセスと区別するために使用されるプロセス識別子
ii. プロセスの説明又はプロセスの説明へのポインタ

FR:6.5 メタデータ要素のメタデータ保護（Metadata Protections）（上記の項目 t）で使用されるそれぞれの暗号メカニズムに対して、CKMS 設計は、以下を明記しなければならない：

i. 暗号アルゴリズム
ii. 鍵パラメタ
iii. 鍵識別子
iv. 保護値（protection value）（例：MAC、デジタル署名）
v. 保護が適用された時期
vi. 保護が検証された時期

一般に、特に鍵とメタデータがひとまとめにされる場合、鍵とメタデータに対して同じメカニズムが使用される。

FR:6.6 メタデータ要素のメタデータ保護（Metadata Protections）（上記の項目 t）で使用される暗号学的ではないそれぞれの信頼プロセスに対して、CKMS 設計は、以下を明記しなければならない：

i. このプロセスを他のプロセスから区別するために使用される識別子
ii. プロセスの説明又はプロセスの説明へのポインタ

FR:6.7 メタデータ要素の信頼関係保護（上記の項目 u）で使用されるそれぞれの暗号メカニズムに対して、CKMS 設計は、以下を明記しなければならない：

i. 暗号アルゴリズム

ii. 鍵パラメタ

iii. 鍵識別子

iv. 保護値（protection value）（例：MAC、デジタル署名）

v. 保護が適用された時期

vi. 保護が検証された時期

FR:6.8 メタデータ要素の信頼関係保護（上記の項目 u）で使用される暗号学的ではないそれぞれの信頼プロセスに対して、CKMS 設計は、以下を明記しなければならない：

i. このプロセスを他のプロセスから区別するために使用される識別子

ii. プロセスの説明又はプロセスの説明へのポインタ

FR:6.9 CKMS 設計は、システムで使用される日時に要求される正確さと精度を明記しなければならない。

FR:6.10 CKMS 設計は、要求される正確さを達成するためにどの権威時刻ソース（authoritative time source）を使用するかを明記しなければならない。

FR:6.11 CKMS 設計は、要求される正確さを達成するためにどのように権威時刻ソース（authoritative time source）を使用するかを明記しなければならない[8]。

FR:6.12 CKMS 設計は、どの日付、時刻、及び機能が信頼される第三者タイムスタンプ（trusted third-party time stamp）を要求するかを明記しなければならない。

6.2.2 鍵とメタデータ情報への要求

CKMS 設計はどのように鍵とメタデータが管理されるかに関する特定の情報を明確にする必要がある。

FR:6.13 それぞれの鍵タイプに対して、CKMS 設計は、鍵及びメタデータ要素に関する以下の情報を明記しなければならない：

a) 鍵タイプ

b) 暗号鍵有効期間（cryptoperiod）（静的鍵（static key）に対して）

c) 生成手段

i. 使用した乱数生成器（RNG）

ii. 鍵生成の仕様（例えば、署名鍵については［FIPS 186］、Diffie-Hellman 鍵確立鍵（key establishment key）については［SP800-56A］）

d) それぞれのメタデータ要素に対して、以下を含める

i. メタデータのソース

ii. メタデータの検証方法

e) 鍵確立（key establishment）の手段

i. 鍵配送スキーム（使用されている場合）

ii. 鍵合意スキーム（使用されている場合）

---

[8] 例えば、権威時刻ソース（authoritative time source）と同期させるための NTP サーバ及び NTP プロトコルの使用。

iii. プロトコル名（名称があるプロトコルが使用されている場合）

f) 暴露に対する保護（例えば、鍵の機密性、物理セキュリティ）

g) 改ざんに対する保護（例えば、MAC 又はデジタル署名）

h) 鍵を使用し得るアプリケーション（例えば、TLS、EFS、S/MIME、IPSec、PKINIT、SSH、等）

i) 鍵の使用が許可されないアプリケーション

j) 鍵保証（key assurances）

i. 対称鍵保証（Symmetric key assurances）（例えば、フォーマットチェック）

- 誰が保証を得るか
- 保証が得られる状況
- どのように保証を得るか

ii. 非対称鍵保証（Asymmetric key assurances）（例えば、所有と有効性の保証）

- 誰が保証を得るか
- 保証が得られる状況
- どのように保証を得るか

iii. ドメインパラメタ有効性チェック

- 誰が有効性チェックを実行するか
- チェックが実行される状況
- どのようにドメインパラメタの有効性の保証を得るか

FR:6.14 CKMS 設計は、CKMS によって生成、保管、伝送、処理、及びその他管理される全ての鍵タイプ及びメタデータについて、全てのシンタックス、セマンティクス、及びフォーマットを明記しなければならない。

6.3 鍵のライフサイクル状態及び遷移

鍵はその生成から破壊までの間にいくつかの状態を経ることがある。本節は、[SP 800-57-part1]の 7 節、鍵状態と遷移（Key States and Transitions）に基づいている。鍵が取り得る状態には以下を含む：活性化前（Pre-Activation）、活性化（Active）、非活性化（Deactivated）、危殆化（Compromised）、破壊（Destroyed）、危殆化鍵破壊（Destroyed Compromised）、及び失効（Revoked）。CKMS 設計者は、CKMS とそのアプリケーションに適切な鍵状態と遷移を選択し定義することに留意されたい。

FR:6.15 CKMS 設計は、CKMS の鍵が取り得る全ての状態を明記しなければならない。

FR:6.16 CKMS 設計は、全ての CKMS 鍵状態間の遷移、及び遷移を起こすことに関係するデータ（入力と出力）を明記しなければならない。

6.4 鍵及びメタデータの管理機能

本節で記載されている鍵及びメタデータの管理機能は、管理目的のために鍵又はメタデータに対して CKMS によって実行される。呼び出しエンティティの認証（authentication）と認可

（authorization）は、6.7 節に記載されたように、アクセスコントロールシステム（ACS）によって実行される。

CKMS は、鍵及びそのメタデータの生成、変更、置き換え、及び破壊の機能を提供するべきである。機能にもよるが、入力や出力には、適用される完全性やソース認証（source authentication）、あるいは機密性についての処理（service）があるかもしれない。

機能への入力の場合、その機能は、入力に対して他のエンティティによって行われた保護を、処理する必要があるかもしれない。例えば、鍵入力機能において、鍵を供給するエンティティ（すなわち、鍵ソース [9]）は、保護されてない状態の鍵に対しデジタル署名を施し、その署名結果を暗号化しているかもしれない。したがって、この例では、鍵入力機能は、入力を復号してデジタル署名検証を実行することで、鍵ソースを認証して鍵の完全性を検証する必要がある。

機能からの出力の場合、その機能はセキュリティ処理（service）を適用する必要があるかもしれない。例えば、鍵出力機能において、その機能の実行者は、暗号化後にデジタル署名が施された鍵が出力されることを期待するかもしれない。そこで、鍵出力関数は、意図する受信者に適切であるように鍵に暗号化及びデジタル署名生成を適用することになる。

FR:6.17 CKMS 設計は、実装されサポートされる鍵及びメタデータの管理機能を明記しなければならない。

FR:6.18 CKMS 設計は、CKMS に実装されるそれぞれの鍵及びメタデータの管理機能のパラメタに適用される完全性、機密性、及びソース認証（source-authentication）の処理（service）を特定しなければならない。

6.4.1 鍵生成

ユーザが鍵を必要とするとき、ユーザは CKMS による鍵生成を要求するべきである。ユーザは、本機能を要求するときに鍵と関連付けられる必要があるメタデータも含めて、鍵タイプ及び他の必要なパラメタ（例えば、鍵生成技術の名称）を指定する必要があるかもしれない。本機能は、鍵及び場合によってはそのメタデータへのポインタである鍵識別子を返すかもしれない。ユーザが鍵の実際の値を知りたい場合、ある状況下での鍵出力機能（6.4.20 節を参照）が使用されるかもしれない。

鍵生成技術は、典型的には鍵と対になる暗号アルゴリズムの仕様に依存する（[FIPS 186] を参照）。異なった暗号アルゴリズムには、異なった仕様（例えば、鍵長とフォーマット）に準拠する鍵を使うことになる。非対称暗号アルゴリズムに対する鍵生成は、鍵ペアでの生成を行うことになる。鍵生成は、暗号目的として設計された乱数生成器の使用を要求する。例えば、NIST はいくつかの承認された乱数生成器（[SP800-90A] を参照）及び鍵生成手順（[FIPS 186] を参照）を公開している。

鍵生成機能は、生成された鍵に関連付けられるメタデータの選択又は入力についても準備されていることがある。

FR:6.19 CKMS 設計は、それぞれの鍵タイプに対して、CKMS で使用される鍵生成手段を明記しなければならない。

FR:6.20 CKMS 設計は、対称鍵及びプライベート鍵を生成するのに使用される元となる乱数生成器を明記しなければならない。

[9] 鍵ソースは、鍵入力機能を使用するエンティティである場合もそうでない場合もある。

6.4.2 所有者登録

セキュリティエンティティ（すなわち、個人（人）、組織、デバイス、又はプロセス）及びメタデータを伴う暗号鍵の最初の登録は、全ての CKMS の基本的な要求事項である。この要求事項を、セキュリティを保ちながら（すなわち、なりすましの脅威から保護する）完全に自動化することは困難であり、そのため通常は人間とのやり取りを必要とする。典型的には、それぞれのエンティティの対称鍵、公開鍵、又はプライベート鍵の初期セットと、エンティティ識別子及びおそらく他のメタデータとも結び付ける CKMS の登録プロセスが存在する。所有者の識別子と鍵を結び付けるプロセスは、所有者の初めての本人証明（ID 証明）プロセスか、CKMS に前から存在している所有者の識別子（ID）に依拠したプロセスを含む。

FR:6.21 CKMS 設計は、鍵と所有者の識別子を結び付けるプロセスを含めて、所有者登録に関わる全てのプロセスを明記しなければならない。

6.4.3 鍵活性化

鍵活性化機能は、暗号鍵の活性化前状態から活性化状態への遷移を提供する。本機能により、鍵の生成後直ちに、その鍵を自動的に活性化するかもしれない。別の方法として、本機能が、鍵がいつ活性化して使用可能になるかを示す日時のメタデータ値を生成するかもしれない。非活性化日時もまた本機能を使用して確立されるかもしれない。

FR:6.22 CKMS 設計は、それぞれの鍵タイプがどのように活性化されるか、及び鍵が活性化される状況を明記しなければならない。

FR:6.23 それぞれの鍵タイプに対して、CKMS 設計は、鍵活性化の通知の要求事項を明記しなければならない。それには、どの当事者に通知されるか、どのように通知されるか、どのセキュリティ処理（services）が通知に適用されるか、及び通知の期間が含まれる [10]。

6.4.4 鍵非活性化

本機能は、鍵を非活性化状態に遷移させる。暗号鍵は、一般的には自らが生成され配付される時に、非活性化される日時が与えられる。場合によっては、使用回数又は保護されたデータの量を基に非活性化することもある。このような非活性化情報はメタデータとして鍵に関連付けられるかもしれない。活性化と非活性化の間の時間は、一般に暗号鍵有効期間（cryptoperiod）と見なされる。この時間は、保護するデータの機微度及び CKMS にもたらされるかもしれない脅威（さらなる説明は［SP 800-57- Part1］を参照）にある程度基づく最大値を持つことが一般的である。暗号鍵有効期間は、鍵及びデータを管理している暗号責任者（cryptographic officer）の懸念に基づいて短縮可能である。CKMS セキュリティポリシーは、ポリシーでカバーされるデータを保護するために使用されるあらゆる鍵タイプの最大許容暗号鍵有効期間を記載すべきである。

FR:6.24 CKMS 設計は、各鍵タイプに対して、鍵の非活性化がどのように決定されるのか（例えば、暗号鍵有効期間（cryptoperiod）による、使用回数による、又はデータ量による）を明記しなければならない。

FR:6.25 CKMS 設計は、それぞれの鍵タイプがどのように非活性化されるか（例えば、非活性化日時、使用回数、又は保護されたデータの量に基づいて、手動で行われるのか自動で行われるのか）を明記しなければならない。

---

[10] 例えば、通知には、鍵活性化の直前に 1 回実行する、事前のある時点から開始して鍵活性化まで単位時間ごとに n 回実行する、又は鍵活性化の時間が近づくにつれて頻度を上げながら実行する、がありうる。

FR:6.26 CKMS 設計は、それぞれの鍵タイプの非活性化日時がどのように変更できるかを明記しなければならない[11]。

FR:6.27 それぞれの鍵タイプに対して、CKMS 設計は、鍵タイプの非活性化の事前通知の要求事項を明記しなければならない。それには、CKMS がサポートするどの役割に通知されるか、どのように通知されるか、どのセキュリティ処理（services）が通知に適用されるか、及び通知の期間が含まれる。

6.4.5 鍵失効

鍵失効は、確立された鍵の暗号鍵有効期間（cryptoperiod）より前に、その鍵の認可された使用を終了させる必要が生じた場合に行われる。暗号鍵は、もはや使用が認可されない状態になった（例えば、鍵が危殆化した）後、可能な限り速やかに失効すべきである。鍵失効には、暗号学的保護への適用のため又はすでに保護された情報の処理のための使用に対してもはや認可されないと鍵に記録することを含む。過去、現在又は未来に鍵を使用するセキュリティエンティティ（すなわち、依拠する当事者）には、鍵が失効されたことが通知される必要がある。これには、失効された鍵を特定する失効リストの公開があるかもしれない。鍵管理システムによって、失効通知の他の形式がサポートされるかもしれない。

FR:6.28 CKMS 設計は、いつ、どのように、どのような状況で失効が実行され、失効情報を依拠する当事者が利用可能になるかを明記しなければならない。

6.4.6 鍵の一時停止及び再活性化

鍵は一時的に停止され後に再活性化されるかもしれない[12]。不可逆的な失効ではなく、一時停止の正当な理由となり得る状況の例には以下のようなものがある：利用延長された時間内に所有者が利用可能な状況にいない、鍵が誤使用された、危殆化の可能性が調査されている、あるいは鍵を含むトークンが誤った場所に配置された。また、失効に関連するセキュリティ上の課題に加え、一時停止された鍵の再活性化におけるセキュリティ上の課題（一時停止は可逆ではあるが失効に他ならないため）も重大（critical）な課題である。

一時停止が、ローカルに呼び出されるエンティティと同様、鍵を保持する遠隔のエンティティにも適用される場合、他のエンティティへの一時停止及び再活性化の通知手段が用意されなければならない。

FR:6.29 CKMS 設計は、どのように、どのような状況で鍵が一時停止されるかを明記しなければならない。

FR:6.30 CKMS 設計は、どのように一時停止情報を依拠又は通信する当事者が利用可能になるかを明記しなければならない。

FR:6.31 CKMS 設計は、どのように、どのような状況で一時停止された鍵が再活性化されるかを明記しなければならない。

FR:6.32 CKMS 設計は、どのように一時停止された鍵によるセキュリティ処理（services）の実行を防止するのかを明記しなければならない。

---

[11] 例えば、時間の経過とともに、鍵全数探索技術の進歩が予想以上の速さで向上したり、鍵及びそのアルゴリズムによって提供されるセキュリティ強度を低下させる新しい攻撃法が発見されたりするかもしれない。そのため、鍵の非活性化日時の変更が必要となるかもしれない。

[12] 一時停止は一時的な非活性化である。言い換えれば、非活性化は一般に不可逆的であるが、一時停止は、鍵を再活性化することで元に戻すことができる。

FR:6.33 CKMS 設計は、どのように再活性化情報を依拠又は通信する当事者が利用可能になるのかを明記しなければならない。

6.4.7 公開鍵の更新

公開鍵証明書は、非対称鍵ペアの公開鍵（すなわち、サブジェクト鍵（subject key））及びその証明書の有効期間を含んでいる。公開鍵証明書の有効期間は、サブジェクト鍵の暗号鍵有効期間（cryptoperiod）より短いことが望ましいかもしれない。これは、失効リスト及び失効情報のサイズを減少させるが、証明書をより頻繁に発行する必要が生じる。公開鍵証明書の更新では、新しい有効期間を持った同じ公開鍵を含む新しい証明書を発行することで、以前の有効期間を超えて既存のサブジェクト公開鍵に対する新しい有効期間を確立する。与えられた公開鍵の更新期間の合計は、その鍵の暗号鍵有効期間（cryptoperiod）を超えてはならない。

事前通知は運用及びミッションの継続性のために有益であり、新しい鍵及び関連するメタデータの適切なセットを適切な当事者に発行できるようになる。例えば、エンティティの現在の公開鍵証明書の期限切れ時に、エンティティは既存の公開鍵の更新か新しい公開鍵の確立かのいずれかを要求する必要があるかもしれない。

FR:6.34 CKMS 設計は、どのように、どのような条件で公開鍵が更新できるかを明記しなければならない。

FR:6.35 それぞれの鍵タイプに対して、CKMS 設計は、鍵タイプの更新の事前通知の要求事項を明記しなければならない。それには、どの当事者に通知されるか、どのように通知されるか、どのセキュリティ処理（services）が通知に適用されるか、及び通知の期間が含まれる。

6.4.8 鍵導出及び鍵更新

一部が秘密であるような他の情報から不可逆な形で鍵が導出されるとき、このプロセスは鍵導出と呼ばれる。鍵導出は、互いの共有秘密（shared secret）から共有鍵を導出する鍵確立プロトコルで使用されることが多い（[SP 800-56A]、[SP 800-56B]、[SP 800-56C]、及び［SP 800-135］を参照）。

鍵導出は、他の鍵（[SP800-108］を参照）又はパスワード（[SP800-132］を参照）から鍵を導出するために使用されることもある。鍵（例えば、K_1）が別の鍵（K_2）を導出するために使用され、導出された鍵（K_2）が元の鍵（すなわち、K_1）を“置き換える”ために使用される場合、このプロセスは鍵更新と呼ばれる。過去には、ただ単に、新しい鍵を確立するために鍵確立プロトコルを使用しなければならないことを避けるために鍵更新が行われていた；つまり、鍵を共有する全てのエンティティが、他の秘密データを何ひとつ使用することなく新しい鍵を生成するために単に鍵更新をしていた。このような鍵更新のプロセスは、鍵を（危殆化又は暗号解読によって）取得し更新方法を知った敵対者が取得した鍵を鍵更新することにより将来のあらゆる時期の鍵を得る、というセキュリティリスクにさらされる可能性がある。

FR:6.36 CKMS 設計は、鍵を導出又は更新するために使用される全てのプロセス、及び鍵が導出又は更新される状況を明記しなければならない。

FR:6.37 それぞれの鍵タイプに対して、CKMS 設計は、鍵の導出又は更新の事前通知の要求事項を明記しなければならない。それには、どの当事者に通知されるか、どのように通知されるか、どのセキュリティ処理（services）が通知に適用されるか、及び通知の期間が含まれる。

6.4.9 鍵及びメタデータの破壊

鍵及びそのメタデータの一部は、使用されることがなくなったときに、復元できないように破壊されるべきである。高セキュリティのアプリケーションにおける鍵の破壊は複雑なプロセスであり、鍵を格納するメディア及び鍵のコピーの配付範囲に依存する。歴史的には、予め規定された方法によって、紙の鍵素材（紙テープ、パンチカード、又は印刷された鍵のリスト）のセキュアな焼却が行われていた。電子的保管媒体に記録された鍵は 0 と 1 のランダムパターンで上書きされるかもしれない。低レベルの磁気を保持する傾向のある磁気メディアは、物理的な破壊、消磁、又は様々なビットパターンで何度も上書きが行われるかもしれない。バックアップストレージメディアが使用されている場合、CKMS 設計者はそのようなメディア内の鍵を破壊する手段を用意すべきである。

FR:6.38 CKMS 設計は、どのように、どのような条件で鍵が意図して破壊されるか、及び破壊がコンポーネントへの局所的（local）なものであるか CKMS 全体への共通的（universal）なものであるかを明記しなければならない。

FR:6.39 それぞれの鍵タイプに対して、CKMS 設計は、鍵破壊の事前通知の要求事項を明記しなければならない。それには、どの当事者に通知されるか、どのように通知されるか、どのセキュリティ処理（services）が通知に適用されるか、及び通知の時期が含まれる。

6.4.10 鍵とメタデータの関連付け

暗号鍵は、関連付けられているいくつかのメタデータ要素を持っていることがある。CKMS 設計者は、どのメタデータが鍵と関連付けなければならないか、又は関連付けることができるかを決定しなければならず、また関連付けを提供する保護メカニズムも決定しなければならない。メタデータ要素に格納されている情報の特性に応じて、メタデータ要素は機密性保護、完全性保護、及びソース認証（source authentication）を要求するかもしれない。関連付け機能は、このような保護を提供するために暗号学的又は信頼プロセス（trusted process）を使用する。

FR:6.40 使用されているそれぞれの鍵タイプに対して、CKMS 設計は、何のメタデータが鍵と関連付けられているか、どのようにメタデータが鍵と関連付けられているか、及びメタデータが鍵と関連付けえられる状況を明記しなければならない。

FR:6.41 使用されているそれぞれの鍵タイプに対して、CKMS 設計は、どのように次のセキュリティ処理（services）（保護）が関連付けられたメタデータに適用されるかを明記しなければならない：ソース認証（source authentication）、完全性、及び機密性。

6.4.11 メタデータの変更

メタデータ変更機能は、鍵と関連付けられている既存の書き込み可能なメタデータを変更するために使用することができる。鍵と関連付けられているメタデータは、認可されていないエンティティによって変更可能であるべきではない。例えば、鍵の所有者の識別子がメタデータに含まれている場合、認可されていないエンティティが鍵所有者の識別子を変更すること又は新たな所有者を追加することは許可されるべきではない。鍵とメタデータとの紐付けは、MAC 又はデジタル署名で実現することもできる。鍵及びそのメタデータの完全性は、MAC 又はデジタル署名の検証によって判断できるかもしれない。

FR:6.42 CKMS 設計は、関連付けられたメタデータが変更される状況を明記しなければならない。

6.4.12　メタデータの削除

本機能は、鍵に関連付けられたメタデータ（削除権限が付与されているもの）を削除する。複数のメタデータ要素は、完全なグループ全体として、個々の要素として、又は複数の要素のうちの特定のサブセットとして削除されることがある。

FR:6.43 CKMS 設計は、鍵と関連付けられたメタデータが削除される状況を明記しなければならない。

FR:6.44 CKMS 設計は、関連付けられたメタデータを削除するために使われる手法を明記しなければならない。

6.4.13　鍵メタデータのリスト化

本機能は、エンティティに認可されている鍵のメタデータをリスト化することをそのエンティティが実行できるようにするものである。エンティティは、ストレージ内にメタデータが関連付けられた複数の鍵を持っているかもしれない。これらの鍵は、デジタル署名生成及び署名検証、認証、暗号化／復号、データ完全性、鍵確立、及び鍵格納のための鍵であるかもしれない。鍵を使用するための認可が鍵に関連付けられたそれぞれのメタデータ要素へのアクセスを自動的に意味するわけではない一方、鍵に関連付けられたそれぞれのメタデータ要素の全ての値を記憶しておくことは非現実的かもしれない。したがって、メタデータのリスト化機能は非常に有用であるかもしれない。

FR:6.45 それぞれの鍵タイプに対して、CKMS 設計は、どのメタデータが認可されたエンティティによってリスト化が可能かどうかを明記しなければならない。

6.4.14　運用中の鍵及びメタデータの保管

運用中の鍵及びメタデータの保管には、鍵やメタデータをメディアへ移すことを含み、後にそのメディア（に保管された鍵やメタデータ）から保管されたデータが復元されるかもしれない。鍵及びメタデータが暗号モジュールの外部に保管されるときに、それらは物理的又は暗号学的に保護されるべきである（[SP 800-57-part1]を参照）。

FR:6.46 それぞれの鍵タイプに対して、CKMS 設計は、以下のことを明記しなければならない：それぞれの鍵タイプとそのメタデータが保管される状況、鍵とメタデータの保管場所、及び鍵とメタデータの保護方法。

6.4.15　鍵及びメタデータのバックアップ

鍵及びメタデータのバックアップは、鍵やメタデータを安全な設備へコピーすることを含み、オリジナルの（運用中の）コピーが喪失、改変、又はその他の理由で利用不能状態になったときにそのコピーを復元できるようにする。自然災害又は人災の後であっても必要なときに鍵及びメタデータが復元できることを保証するために、鍵及びメタデータのバックアップコピーは、運用中の鍵／メタデータと同じ設備にも異なる設備にも配置されるかもしれない。鍵／メタデータは、所有者又は信頼されるエンティティによってバックアップされることがある。

FR:6.47 CKMS 設計は、どのように、どこで、どのような状況において鍵及びそのメタデータがバックアップされるかを明記しなければならない。

FR:6.48 CKMS 設計は、バックアップされた鍵及びメタデータの保護のためのセキュリティポリシーを明記しなければならない[13]。

FR:6.49 CKMS 設計は、鍵及びメタデータのバックアップ中のセキュリティポリシーがどのように実装されるかを明記しなければならない。例として、バックアップされた鍵及びメタデータの配送及び保管中における、機密性とマルチパーティコントロールの要求事項の実装方法が挙げられる。

6.4.16 鍵やメタデータのアーカイブ

鍵やメタデータのアーカイブには、鍵やメタデータを必要なときに復元できるように、それらを長期保管用のストレージ設備に安全に配置することを含む。アーカイブは、鍵及びメタデータの保全ポリシー（4.3 節を参照）をサポートする。アーカイブされた鍵やメタデータは物理的又は暗号学的に保護されなければならない。アーカイブ内の鍵やメタデータを保護するために使用される鍵はアーカイブ鍵と呼ばれる。これらのアーカイブ鍵もまた暗号鍵有効期間（cryptoperiod）を持ち、アーカイブされた鍵やメタデータに提供される保護の継続性はアーカイブ鍵の暗号鍵有効期間が期限切れになるときに考慮される必要がある。これは、物理的保護、あるいは同等以上の強度の暗号アルゴリズムのための新しいアーカイブ鍵の生成及びその新しいアーカイブ鍵によるアーカイブされた鍵やメタデータの再暗号化を伴うかもしれない。

鍵及びメタデータをアーカイブすることは、通常、アーカイブされた鍵やメタデータを新しい保管メディアに移動するための手段を要求する。この移動は、メディア及びメディアリーダの老朽化又は技術的な変化のために、古いメディアがもはや読み込み不能になるときに行われる。アーカイブされた鍵やメタデータは古い保管メディアから復元され、新しい保管メディアに保管されるべきである；転送後、古い保管メディアからその鍵は破壊されるべきである。鍵やメタデータのアーカイブ又は破壊を実行するとき、鍵やメタデータが法律や規則で要求される期間利用可能であるように、適用される法律や規則を考慮しなければならない。

FR:6.50 CKMS 設計は、どのように、どこで、どのような状況で鍵やメタデータがアーカイブされるかを明記しなければならない。

FR:6.51 CKMS 設計は、鍵やメタデータのセキュアな破壊、又は新しい保管メディアに書き込まれた後の古い保管メディアのセキュアな破壊のための手法を明記しなければならない。

FR:6.52 CKMS 設計は、アーカイブ鍵の暗号鍵有効期間（cryptoperiod）の期限切れ後に、鍵やメタデータがどのように保護されるかを明記しなければならない。

6.4.17 鍵やメタデータの復元

鍵やメタデータの復元には、前もってバックアップ、アーカイブ又は保管された鍵やそのメタデータのコピーを取得することを含む。鍵やメタデータは、復元のための全てのルールが満たされ検証された後に、認可されたエンティティ（例えば、その所有者又は信頼されるエンティティ（trusted entity））によって復元することができる。CKMS セキュリティポリシーは、鍵やメタデータが復元できる条件を記述すべきである。

FR:6.53 CKMS 設計は、鍵やメタデータの CKMS 復元ポリシーを明記しなければならない。

FR:6.54 CKMS 設計は、鍵やメタデータの復元ポリシーを実装及び実施するために使用されるメカニズムを明記しなければならない。

---

[13] 例えば、2 人によるコントロールが要求されるかもしれない。

FR:6.55 CKMS 設計は、どのように、どのような状況で鍵やメタデータがそれぞれの鍵データベース又はメタデータ保管設備から復元されるかを明記しなければならない。

FR:6.56 CKMS 設計は、鍵やメタデータが復元中にどのように保護されるかを明記しなければならない。

6.4.18 鍵確立

鍵確立は、2 つ又はそれ以上のエンティティ間で鍵をセキュアに共有するプロセスである。あるエンティティから他のエンティティに鍵が伝送される場合（鍵配送）も、エンティティによって共有されている情報から鍵が導出される場合（鍵合意）もある。鍵を伝送する手段あるいは情報を共有する手段は、手動（例えば、配達人による送付）であることも、自動（例えば、インターネット上での送信）であることもある。

FR:6.57 CKMS 設計は、どのように、どのような状況で鍵及びそのメタデータが確立されるかを明記しなければならない。

6.4.19 鍵及び関連付けられたメタデータの暗号モジュールへの入力

鍵入力機能は、ひとつ又はそれ以上の鍵及び関連付けられたメタデータを、実際に使用する前の準備として暗号モジュールに入力するために使用される。鍵及びメタデータは、平文形式、暗号化された形式、鍵分割、完全性保護が行われた形式（例えば、署名された証明書で）、又はそれらの組み合わせで入力され得る。

FR:6.58 CKMS 設計は、どのように、どのような状況で鍵及びメタデータが暗号モジュールに入力されるか、入力される形式、及び入力に用いられる手段[14]を明記しなければならない。

FR:6.59 CKMS 設計は、（必要ならば）どのように入力された鍵とメタデータの完全性及び機密性が入力時に保護され検証されるかを明記しなければならない。

6.4.20 鍵及び関連付けられたメタデータの暗号モジュールからの出力

鍵出力機能は、ひとつ又はそれ以上の鍵及び関連付けられたメタデータを、外部での使用又は保管のために暗号モジュールから出力する。出力するのは、アーカイブ、バックアップ、又は通常の運用目的のためかもしれない。鍵生成設備として動作するモジュールは、その後に行われる鍵の配付のために鍵を出力するかもしれない。鍵及びメタデータは、平文形式、暗号化された形式、鍵分割、完全性保護が行われた形式、又はそれらの組み合わせで出力され得る。

FR:6.60 CKMS 設計は、どのように、どのような状況で鍵及びメタデータが暗号モジュールから出力されるか、及び出力される形式を明記しなければならない。

FR:6.61 CKMS 設計は、どのように出力された鍵とメタデータの機密性及び完全性が暗号モジュールの外部で保護されるかを明記しなければならない。

FR:6.62 プライベート鍵、対称鍵、又は機密のメタデータが暗号モジュールから平文形式で出力される場合、CKMS 設計は、鍵及びメタデータが提供される前に、呼び出しエンティティを認証するかどうか、及びどのように認証するかを明記しなければならない。

---

[14] 例えば、キーボード入力、鍵ローダ、又は自動化されたプロトコル経由。

6.4.21　公開鍵ドメインパラメタの検証

本機能は、ある種の公開鍵暗号アルゴリズムの公開ドメインパラメタについてある種の正当性検証を実行する。これらのテストに合格することは、ドメインパラメタが数学的に正しいことの保証を提供する（[SP 800-89］及び［SP 800-56A］を参照)。

FR:6.63 CKMS 設計は、どのように、どこで、どのような状況で、公開鍵ドメインパラメタが検証されるかを明記しなければならない。

6.4.22　公開鍵の検証

本機能は、公開鍵についてある種の正当性チェックを実行して、公開鍵が数学的に正しいことについてある程度の保証を提供する。これらのテストは、典型的には公開鍵が対象とする公開鍵暗号アルゴリズムに依存するが、プライベート鍵の知識には依存しない（[SP 800-89]、[SP 800-56A]、及び［SP 800-56B］を参照)。6.4.22 節、6.4.23 節及び 6.4.28 節が、公開鍵検証のための全体的な信頼シナリオの提供に関係していることに留意されたい。

FR:6.64 CKMS 設計は、どのように、どこで、どのような状況で、公開鍵が検証されるかを明記しなければならない。

6.4.23　公開鍵証明書パスの検証

本機能は、証明書パス（証明書チェーンとしても知られる）を検証するー依拠するエンティティのトラストアンカーから依拠するエンティティが信頼を確立する必要がある公開鍵（つまり、トランザクションでの他のエンティティの公開鍵）へのパス。証明書パスの検証によって、証明書で与えられるサブジェクト ID（subject identity）が、静的な公開鍵の所有者識別子（ID）であり、対応する静的なプライベート鍵の保有者であることの実際の保証が提供される（プライベート鍵所有の証明を、認証局又は依拠するエンティティによって信頼されている他のエンティティが保証していると仮定する)。

FR:6.65 CKMS 設計は、どのように、どこで、どのような状況で公開鍵証明書パスが検証されるかを明記しなければならない。

6.4.24　対称鍵の検証

本機能は、対称鍵及びそのメタデータに対するテストを実行する。例えば、このテストでは、鍵長及び鍵フォーマットが正しいかのチェックを含むかもしれない。また、本機能は、鍵やそのメタデータに付与されたあらゆるエラー検出／訂正コード又は完全性チェックも検証するかもしれない。

FR:6.66 CKMS 設計は、どのように、どこで、どのような状況で、対称鍵やそのメタデータが検証されるかを明記しなければならない。

6.4.25　プライベート鍵（又は鍵ペア）の検証

本機能は、プライベート鍵に対してある種のテストを実行し、その仕様を満たすことの保証を提供する。このテストは、プライベート鍵の所有者又はプライベート鍵の所有者の代理として振舞う信頼される第三者のみが実行できる。このテストには、公開鍵に対する復号関数をプライベート鍵が実行することを検証する鍵ペア整合性テストも含むかもしれない。例えば、RSA 鍵ペアの場合、与えられた入力ブロックにプライベート鍵を適用し、続いてその結果に公開鍵を適用したとき、常に最初に与えられた入力ブロックが得られるはずである（さらなる情報は［SP 800-56B］の 6.4.1 節を参照)。

FR:6.67 CKMS 設計は、どのように、どこで、どのような状況で、プライベート鍵又は鍵ペア、あるいはそのメタデータが検証されるかを明記しなければならない。

6.4.26　プライベート鍵の所持の検証

本機能は、公開鍵の所有者であると主張する者が対応するプライベート鍵を所持しており、したがって鍵ペアの所有者であることの保証を得ることを望む、公開鍵を受領したエンティティによって使用される。典型的には、他のエンティティがプライベート鍵の所持を検証するために公開鍵を使う暗号トランザクションにおいて、鍵ペアの所有者はプライベート鍵を使用することを要求される。例えば、受信者に送信する前に、所有者はプライベート鍵を使用してデータ（例えば、公開鍵及び他の情報）に署名するかもしれない。受信者は受信した公開鍵を使用して受信したデータの署名を検証する（[SP 800-56A]、[SP 800-56B]、及び［SP 800-89］を参照)。また、本機能は、プライベート鍵の所有者が自身のプライベート鍵の所持を検証する機能を含むこともある。

FR:6.68 CKMS 設計は、どのように、どこで、どのような状況で、プライベート鍵とそのメタデータの所持が検証されるかを明記しなければならない。

6.4.27　鍵を使用した暗号機能の実行

主に使用される機能は、実際にデータへの暗号学的保護を提供する機能である。これらの機能は、署名生成、署名検証、暗号化、復号、鍵ラッピング、鍵アンラッピング、MAC 生成、及び MAC 検証を含み得る。それらは暗号モジュールの内部で実行されるべきである。

FR:6.69 CKMS 設計は、サポートされている全ての暗号機能、及びそれらの暗号機能が CKMS のどこで実行されるか（例えば、CA、ホスト、又はエンドユーザシステム）を明記しなければならない。

6.4.28　トラストアンカーストアの管理

CKMS は、あるエンティティがひとつ以上の信頼される公開鍵を持っていることを要求することがある。これらの公開鍵はトラストアンカーとしても見なされる。トラストアンカーは、それなしでは信頼されない他の公開鍵についての信頼を確立するために使用される。トラストアンカーなしでは信頼されない公開鍵についての信頼は、依拠するエンティティが信頼するトラストアンカーから始まる公開鍵証明書のチェーン（6.4.23 節で“証明書パス”と名付けられた）の中にある全ての署名を検証することによって確立される。そのため、トラストアンカーの完全性は CKMS のセキュリティにとって極めて重要である。典型的には、トラストアンカーの追加、削除及び保管を行うようなトラストアンカー管理機能を CKMS はサポートする。トラストアンカーのフォーマットは［RFC 5914］で規定されている。Secure Trust Anchor Management Protocol（TAMP）は［RFC 5934］で定義されている。

FR:6.70 CKMS 設計は、サポートされている全てのトラストアンカー管理機能を明記しなければならない（[RFC6024］を参照)。

FR:6.71 CKMS 設計は、依拠するエンティティがトラストアンカーについてのソース認証（source authentication）及び完全性検証を実行できるように、どのようにそれらのトラストアンカーがセキュアに配付されるかを明記しなければならない。

FR:6.72 CKMS 設計は、依拠するエンティティのシステムのトラストアンカーストアに対して、認可された追加、変更、削除のみが行えることを保証するために、どのように依拠するエンティティのシステムでトラストアンカーが管理されるかを明記しなければならない。

6.5 暗号鍵やメタデータのセキュリティ：保管中

暗号鍵をストレージに入力するとき、その鍵は典型的にはメタデータと共に入力される。メタデータは、所有者識別子又はユーザアクセスコントロールリストを含むかもしれない。どのメタデータが誤っていても、誤った情報は CKMS システムによって長く記録される。したがって、CKMS ストレージシステムは、あらゆるデータが格納されるより前に、入力エンティティの認可と入力されるデータの完全性を検証すべきである [15]。

暗号鍵が保管されるとき、その鍵は保護を必要とする。対称鍵及びプライベート鍵は、機密性保護及びアクセスコントロールを必要とする。全ての鍵は完全性保護を必要とする。機密性保護のためには、暗号やコンピュータセキュリティ、物理セキュリティが導入可能である。対称鍵暗号が鍵の機密性保護のために使用される場合、保管された鍵及び守秘メタデータを暗号化及び復号するために使用される鍵ラッピング対称鍵が存在することが多い。鍵暗号化鍵の階層のトップレベルでは、典型的には物理的に保護されなければならない鍵が存在する。

非対称鍵暗号が鍵の機密性保護のために使用される場合、保管される鍵を暗号化するために公開鍵が使用されるかもしれない。鍵を復号するために使用される対応するプライベート鍵は、ある種の方法で保護されなければならない。例えば、（通常は暗号化を伴わない）物理セキュリティや鍵分割（6.7.5 節を参照）を使用する。

破損した鍵は意図する機能を正しく実行せず、ある状況においては他の鍵の危殆化を引き起こすかもしれないため、全ての保管される鍵は完全性保護を必要とする。物理セキュリティは鍵に完全性保護を提供することができるが、追加の手段を用いることが多い。エラー検出コードは鍵への意図しない破損を検出することができ、エラー訂正コードはある種の破損を訂正することができる。しかしながら、故意に鍵が破損（改ざん）されうるならば、エラー検出のために MAC やデジタル署名のような暗号学的完全性チェックが実装されるべきである。鍵に訂正不能な破損が検出された場合、その鍵は使用すべきでない。公開鍵が証明書に含まれている場合、証明書上のデジタル署名の手段によって完全性保護が提供されている。公開鍵が証明書の外部に保管されている場合、その完全性は何か他の手段で保護される必要がある。

CKMS は、認可されたユーザのみが保管された鍵にアクセスできるようにすべきである。そのため、CKMS はある種のアクセスコントロールシステム（ACS）を備えているべきである（6.7.1 節を参照）。ACS は、パスワード又は暗号鍵を持つ認可されたユーザからの鍵を要求するようなシンプルなものであることも、生体認証技術を使用していることもある。

鍵は、エラー訂正コードでも再構築できない程度にまで破損する、喪失する又は破壊されるかもしれない。その鍵が対称鍵又はプライベート鍵である場合、その鍵によって保護されたデータの喪失という結果になり得る。CKMS は、価値のあるデータの回復を提供するのに必要な鍵のバックアップ、アーカイブ、及び回復のための手段を搭載すべきである。例えば、［SP 800-57-Part1］の附属書 B は、様々な鍵タイプに対する回復手続きのガイダンスを提供している。

鍵メタデータへの破損は、鍵の誤使用又はサービス拒否という結果をもたらし得る。したがって、メタデータもまたバックアップ、アーカイブ、及び回復を必要とするかもしれない。

FR:6.73 CKMS 設計は、鍵やメタデータをストレージに入れるエンティティの ID 認証及び認可検証に使用される手段を明記しなければならない。

FR:6.74 CKMS 設計は、ストレージに入力する鍵やメタデータの完全性検証に使用される手段を明記しなければならない。

FR:6.75 CKMS 設計は、保管された対称鍵、プライベート鍵及びメタデータの機密性保護に使用される手段を明記しなければならない。

---

[15] アクセス直後及び運用上の使用前に、鍵及びメタデータの完全性を検証することも優れた実践である。

FR:6.76 鍵ラッピング鍵（又は鍵ペア）が保管された鍵を保護するために使用される場合、CKMS 設計は、鍵ラッピング鍵（又は鍵ペア）を保護し、その使用を制御するために使用される手段を明記しなければならない。

FR:6.77 CKMS 設計は、保管された鍵及びメタデータの完全性保護に使用される手段を明記しなければならない。

FR:6.78 CKMS 設計は、保管された鍵へのアクセスがどのように制御されるかを明記しなければならない。

FR:6.79 CKMS 設計は、全ての保管された鍵を訂正又は復元するために使用される手法を明記しなければならない。

6.6 暗号鍵及びメタデータのセキュリティ：鍵確立の期間中

鍵及びメタデータは、鍵配送又は鍵確立の手段を使用し、通信したいエンティティの間で確立されうる。これらの手段は典型的には電子通信ネットワークを通して鍵を確立するために使用されるが、鍵が手動で配付されるときに（物理的保護に加えて）追加のセキュリティを提供するために使用されることもある。鍵配送のときは、一方のエンティティが共有する鍵を生成し、その鍵及び（あれば）メタデータを他方のエンティティに配付する。鍵合意のときは、共有鍵を導出するために使用する情報を両方のエンティティが拠出する。メタデータは共有鍵による保護下で配送されるかもしれない。［SP 800-56A］及び［SP 800-56B］が、鍵確立の暗号スキームを規定する。

6.6.1 鍵配送

一方のエンティティ（送信者）から他方のエンティティ（意図する受信者）に暗号鍵及びメタデータが配送（配付）されるとき、それらは保護されるべきである。対称鍵及びプライベート鍵は、機密性保護とアクセスコントロールを必要とする。機密性保護のためには、物理的保護又は暗号が使用される。手動で配付される鍵は、信頼される配達人によって物理的に保護されるか、物理的に保護されたチャネルが使用されるかもしれない。データ盗聴及び改ざんを受けやすいネットワークを通して、鍵が送信されることが非常に多い。配送中の対称鍵及びプライベート鍵の機密性を保護するために暗号が使用される場合、鍵ラッピング対称鍵（symmetric key wrapping key）又はひとつ以上の非対称配送鍵ペア（asymmetric transport key pair）が関わる鍵確立技術が使用される。これらのラッピング鍵及び配送鍵は、配送に関わるエンドエンティティによっても保護されるべきである。

破損した鍵は意図する機能を正しく実行せず、攻撃者にコントロールされた攻撃による鍵への破損（改ざん）はなりすまし又は暗号解読攻撃につながりかねないため、全ての配送鍵は完全性保護を必要とする。そのため、使用前に破損した鍵を検出することは、システムのセキュリティ及び信頼性を向上させる。物理セキュリティは鍵の完全性保護を提供しうるが、典型的なネットワークにおいては電子データの物理的保護が欠如しているため、他の手段が使用されることが多い。エラー検出コードは鍵への意図しない破損を検出することができ、エラー訂正コードはある種の破損を訂正することができる。しかしながら、故意に鍵が破損（改ざん）されるならば、エラー検出のために MAC やデジタル署名のような暗号学的完全性チェックが使用されるべきである。鍵に訂正不能な破損が検出された場合、使用前に新しい又は訂正された鍵を確立すべきである。公開鍵が証明書に含まれている場合、証明書上のデジタル署名の手段によって完全性保護が提供されている。

配送された鍵の受信者は、期待する認可された鍵送信者からその鍵が来たことの保証を求める。自動化された手段が配送時に使用される場合、この保証は、典型的には受信者に対して送信者の ID 認証をする暗号メカニズムを使用することによって提供される。鍵が手動で配送される場合、この保証は鍵を配送する信頼される配達人の認証によって提供されるかもしれない。

FR:6.80 CKMS 設計は、配送中の対称鍵及びプライベート鍵の機密性保護に使用される手段を明記しなければならない。

FR:6.81 CKMS 設計は、配送された鍵の完全性保護に使用される手段、及びエラー検出後にどのように鍵が再構築又は置き換えられるのかを明記しなければならない。

FR:6.82 CKMS 設計は、配送される鍵素材（keying material）の受信者に、どのように鍵送信者の識別子（ID）が認証されるかを明記しなければならない。

6.6.2 鍵合意

2 つのエンティティは、一方から他方に鍵を配送することなく、協調して、暗号鍵を生成して合意することができる。それぞれのエンティティは、共通鍵を導出するために使われるある種の情報を提供し合うが、セキュアな鍵合意スキームが使用されるかぎり、これら提供された情報を取得した傍受者が合意済みの鍵を求めることはできない。（鍵合意鍵を使用する）暗号アルゴリズムは、それぞれのエンティティによって実行される。

鍵合意プロセスに参加する各エンティティは、典型的には他方のエンティティ識別子（ID）の保証を必要とする。この保証は鍵合意プロトコルによって提供されることが多い。

FR:6.83 CKMS 設計は、CKMS にサポートされるそれぞれの鍵合意スキームを明記しなければならない。

FR:6.84 CKMS 設計は、鍵合意に参加するそれぞれのエンティティがどのように認証されるかを明記しなければならない。

6.6.3 鍵確認

鍵が 2 つのエンティティの間で確立されるとき、それぞれのエンティティは、実際に、他方のエンティティが正しい鍵を確立したことの確認を望むかもしれない。鍵確認スキームは、この機能を提供するために使用される。［SP 800-56A］及び［SP 800-56B］に、連邦 CKMS で使用される鍵確認スキームを規定する。他の手段も適切であるかもしれない。

FR:6.85 CKMS 設計は、他方のエンティティと正しい鍵を確立したことを確認するために使用されるそれぞれの鍵確認手段を明記しなければならない。

FR:6.86 CKMS 設計は、それぞれの鍵確認が実行される状況を明記しなければならない。

6.6.4 鍵確立プロトコル

保管及び配送の両方における暗号鍵の調達方法について、いくつかの自動化プロトコルが開発されている。これらのプロトコルは特定のアプリケーション又はアプリケーション一式のために設計されていることが多い。以下に有名な鍵確立プロトコルをいくつか挙げる：

a) Internet Key Exchange（IKE）

b) Transport Layer Security（TLS）

c) Secure/Multipart Internet Mail Extensions（S/MIME）

d) Kerberos

e) Over-The-Air-Rekeying（OTAR）Key Management Messages

f) Domain Name System Security Extensions（DNSSEC）

g） Secure Shell（SSH）

項目 a）から f）までの高レベルの概要は、米国政府での使用に当たってどの暗号オプションが推奨されるかのガイダンスとともに［SP 800-57-part3］に記載されている。Secure Shell の情報については、［RFC 4251］を参照されたい。

FR:6.87 CKMS 設計は、鍵確立と保管の目的のために CKMS によって採用されている全てのプロトコルを明記しなければならない。

6.7 鍵及びメタデータの管理機能へのアクセス制限

本節は、鍵及びメタデータの管理機能へのアクセスがどのようにコントロールされるかを記載する。アクセスを要求するエンティティは認証されるかも知れず、鍵及び他の機微なメタデータの人間への公開は防止されるか又は厳しく制限されるかもしれない。

6.7.1 アクセスコントロールシステム

CKMS のセキュリティは、6.4 節に記載された鍵及びメタデータの管理機能の適切な順序での実行に依存する。これらの機能は、時刻、イベント、エンティティによる要求、又はこれらの選択肢の組み合わせによって実行されるかもしれない。鍵及びメタデータの管理機能が認可されたエンティティ[16]による要求（呼び出し）への応答としてのみ実行されること、及び全ての適用可能な制限が満たされている[17]ことを保証するために、アクセスコントロールシステム（ACS）が必要である。例えば、鍵復元機能（6.4.17 節を参照）が暗号責任者（cryptographic officer）の役割に限定され、かつ入力パラメタが特定の境界内にあり、特定のフォーマットに従っていることが検証されるかもしれない。

アクセスコントロールシステムは、暗号モジュールと連動して、機微な鍵及びメタデータへのアクセスをコントロールするために動作する。アクセスコントロールシステムは、認可されたエンティティのみが鍵及びメタデータの管理機能を実行することを許可されていると保証することによって、鍵を保護する。加えて、典型的には、管理上のアクセスログが記録され、かつ個人の説明責任（personal accountability）のために監査を受ける。ACS は非常にシンプルであるかもしれない；例えば、適切な ID 及びパスワードを入力したユーザは誰でもあらゆる鍵に対してあらゆる鍵管理機能の実行を認可されるかもしれない。あるいは、ACS はもっとはるかに複雑であるかもしれない。

図 10 は、呼び出しエンティティ、アクセスコントロールシステム、保護されたメモリ、及び暗号モジュールの間の関係を図示している。これらのデバイスは、隣接配置されている場合も、図に示したようにセキュアチャネルによって接続されている場合もあり得る。呼び出しエンティティは、ACS によって提供される CKMS 機能の呼び出しを行う。ACS は、保護されたメモリ及び暗号モジュールを使用して、呼び出しエンティティを認証する。認証が成功し、エンティティが適切に認可されれば、暗号モジュールに対して暗号処理（cryptographic service）要求を生成することによって該当機能が実行される。最後に、呼び出しエンティティに対してその応答が返される。

鍵管理機能の操作の見本として追加の詳細を図 11 に示す。機能呼び出しは、呼び出しエンティティ識別子（ID）、呼び出しエンティティ認証符号（authenticator）、機能名及び鍵識別子から構成され、その機能呼び出しが ACS に送られる。最初にエンティティの認証が行われる。次に、エンティティ識別子（ID）が鍵及びその機能に対するアクセスコントロールリスト（鍵メタデータに存在する）に入っていることをチェックすることによって、コマンド実行のためのエンティティの認可が検証され

---

[16] エンティティの認可は、エンティティの識別子（ID）（又は役割）が認証された後に決定される。識別子（ID）（又は役割）は、機能を実行するための承認手続きとして検証される。

[17] 正しく安全な操作を確実にすることに役立つために、機能への入力及び使用に課される制限のことである。

る。ACS がその機能を許可すべきでないと判断すれば、機能拒否インジケータを返す。しかしながら、鍵及びメタデータを使用する認証されたエンティティにその機能が許可されれば、ACS は機能ロジックに要求された操作を実行するように通知する。機能ロジックは、必要に応じて、暗号化、復号、署名、署名検証又は MAC 計算のために暗号モジュールを呼び出すかもしれない。最後に、機能呼び出しの応答が呼び出しエンティティに提供される。

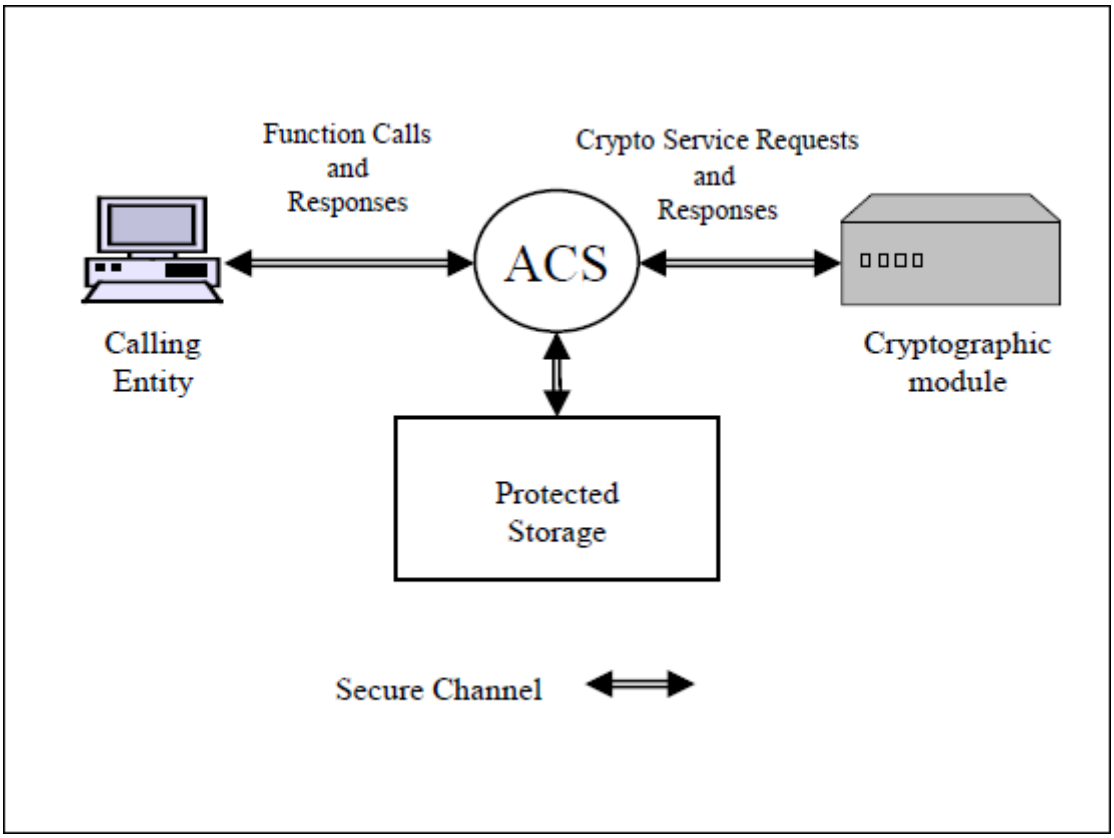

図 10 管理機能アクセスコントロール

ACS は、要求された機能を実行するか否かを決定する。この決定は、主として呼び出しエンティティの認証された識別子（ID）（又は役割）、そのエンティティの認可、CKMS のセキュリティポリシー、機能、鍵及びそのメタデータに基づく。鍵のメタデータは、実施されるコントロールを決定する上でクリティカルな役割を果たすことがある。例えば、ある組織は、複数のユーザが共有鍵を使用して特定のファイルを暗号化及び復号することを許可するが、他のファイルについては 1 人のユーザのみが復号できるようにすると決定するかもしれない。CKMS ポリシーは、管理組織の情報管理ポリシーをサポート及び実施すべきである。したがって、CKMS アクセスコントロールシステムは、CKMS セキュリティポリシーの要求事項に適応できるように十分な柔軟性を持つことが非常に望まれる。

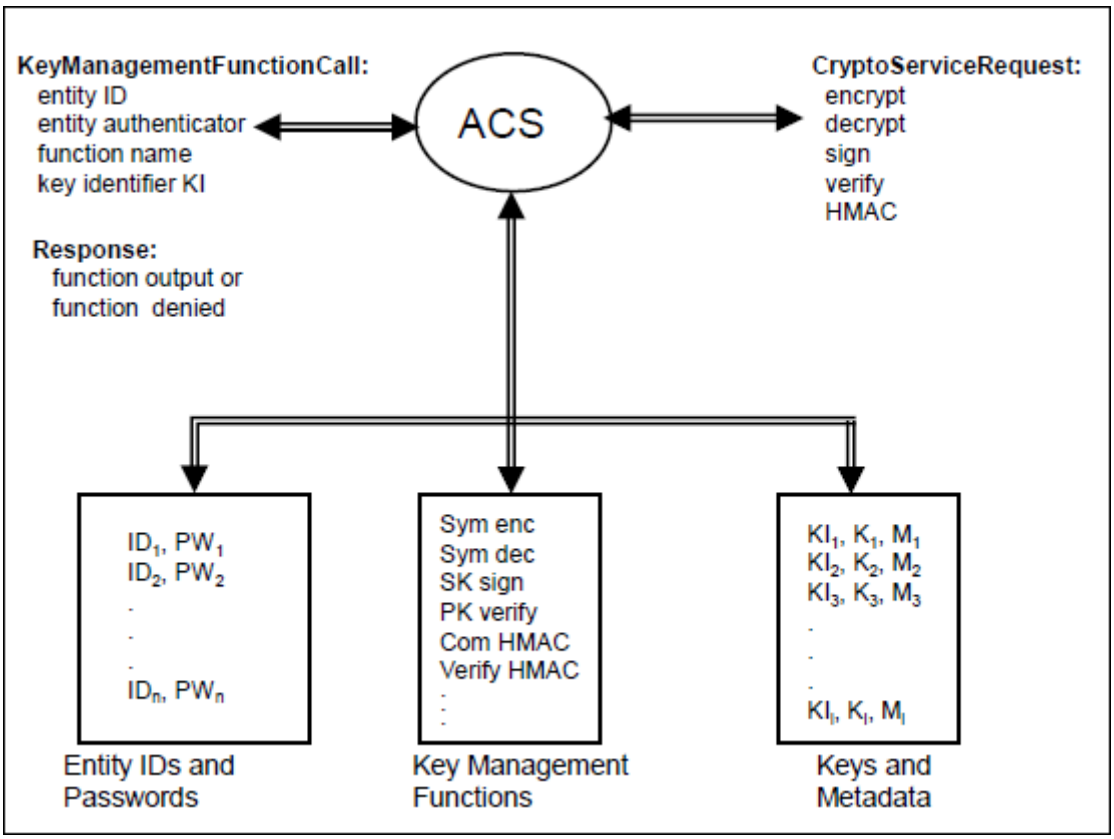

図 11 鍵管理機能のコントロールロジックの見本

FR:6.88 CKMS 設計は、エンティティ、ACS（アクセスコントロールシステム）、機能ロジック、及びそれらの間の接続の配置を示すことで CKMS のトポロジーを明記しなければならない。

FR:6.89 CKMS 設計は、適切な操作を保証するために実装されている鍵管理機能に対する制限を明記しなければならない。

FR:6.90 CKMS 設計は、鍵管理機能へのアクセスが、どのように認可されたエンティティに制限されているかを明記しなければならない。

FR:6.91 CKMS 設計は、鍵管理機能へのアクセスを制御するための ACS とそのポリシーについて明記しなければならない。

FR:6.92 CKMS 設計は、少なくとも以下を明記しなければならない：

a) エンティティの粒度（例：人、デバイス、組織）

b) エンティティが識別されているかどうか、及びその方法

c) エンティティが認証されているかどうか、及びその方法

d) エンティティの認可が検証されているか、及びその方法

e) それぞれの鍵管理機能のアクセスコントロール

FR:6.93 CKMS 設計は、CKMS セキュリティポリシーを適応、実装、実施するための ACS の能力を明記しなければならない。

6.7.2 平文鍵の暗号モジュールへの入力及び出力の制限

よく設計された CKMS は、人間による平文鍵へのアクセスを最小化する。鍵が平文であることが必要である主な場面は、暗号モジュール内で暗号機能を実行するときである。これらの暗号モジュールは、通常平文鍵への物理的保護を提供し、平文鍵が露出しないようにする。暗号モジュールが人間の代理として鍵を生成して暗号機能を実行することがあり、人間は平文の対称鍵又はプライベート鍵を見る必要は全くない。この特長は、そのような暗号モジュールを使用する CKMS をより透過的でよりセキュアにする。例えば、プライベート鍵配送鍵（private key transport key）が暗号モジュール内で生成され、モジュール外に出ることが決して許可されないということがあり得る。暗号モジュールから出力される鍵は、鍵配送スキームを使用して（暗号化された形で）配送され得る。暗号化／復号対称鍵は、受信エンティティの公開鍵を使用して暗号化され配送され得る。公開鍵保管鍵（public key storage key）又は鍵ラッピング対称鍵（symmetric key wrapping key）で暗号化されたときに、鍵は暗号モジュールの外部でセキュアに保管されるかもしれない。時には、レガシーシステムをサポートするために平文鍵の出力が許可される。そのようなケースでは、以下の 6.7.4 節で説明されるような、マルチパーティコントロール（multi-party control）を検討すべきである。

暗号モジュールへの鍵の入力及び出力のための要求事項は、それぞれ 6.4.19 節及び 6.4.20 節に記載されている。

FR:6.94 CKMS 設計は、平文の対称鍵又は平文のプライベート鍵が暗号モジュールに入力又は出力される状況を明記しなければならない。

FR:6.95 いかなる暗号モジュールにおいても、平文の対称鍵又は平文のプライベート鍵が入力又は出力される場合には、CKMS 設計は、平文の鍵がどのように暗号モジュールの外部で保護され、制御されるかを明記しなければならない。

FR:6.96 いかなる暗号モジュールにおいても、平文の対称鍵又は平文のプライベート鍵が入力又は出力される場合には、CKMS 設計は、そのような動作がどのように監査されるかを明記しなければならない。

6.7.3 人間の入力に対するコントロール

鍵管理機能が、鍵又は機微なメタデータの入力を人間に求める場合、正確さと、おそらくは入力のセキュリティも、人間に依存することになる。加えて、適切な時間又は適切なイベント発生時に入力が行われるかも、人間に依存するかもしれない。この場合、人間による入力が行われないときにシステムがどのようなアクションを起こすべきかという問題が発生する。必要なときに CKMS がそのような機能を自動的に実行できるのであれば、そのシステムはユーザにとってより透過的になり、よりセキュアになる可能性がある。

FR:6.97 それぞれの鍵とメタデータの管理機能に対し、CKMS 設計は、全ての人間による入力パラメタ、そのフォーマット、及び入力が行われないときに CKMS が取るアクションを明記しなければならない。

6.7.4 マルチパーティコントロール（Multiparty Control）

ある種の鍵管理機能は、その機能を実行するために複数の協力するエンティティを必要とするかもしれない。このマルチパーティコントロール（multiparty control）は、機能を実行する前にその機能のアクセスコントロールシステムによって、n 個中 k 個のエンティティの認証が要求され、その認証後に認可が行われるかもしれない。マルチパーティコントロールは、高度に機微な機能のために利用されるべきである。例えば、高度に機微な機能は、その機能を実行するために 2 人以上の個人がログオンして認証されていることを要求すべきである。

FR:6.98 CKMS 設計は、マルチパーティコントロール（multiparty control）を要求する全ての機能を明記し、それぞれの機能に対して *k* と *n* を規定しなければならない。

FR:6.99 それぞれのマルチパーティ機能に対して、CKMS 設計は、なぜ *n* 個中任意の *k* 個のエンティティで望む機能を有効にできるが *k-1* 個のエンティティでは有効にできないのかを示すあらゆる既知の論拠（論理、数学）を引用又は明記しなければならない。

6.7.5 鍵分割（Key Splitting）

鍵分割（Key splitting）は、マルチパーティコントロール（multiparty control）のための技術である。高度に機微な鍵が必要なとき、*n* 個の分割鍵が生成され、任意の *k* 個の分割鍵を使用することでその鍵を構成できるが、いかなる *k-1* 個の分割鍵からはその高度に機微な鍵に関する何らの知識も得られない。*n* 個の分割鍵のそれぞれが *n* 個の信頼されるエンティティの一つに割り当てられ、そのうち *k* 個のエンティティが参加することに同意しない限り鍵が構成できないようにする。いかなる *k-1* 個のエンティティが所有する分割鍵が危殆化して、たとえその *k-1* 個全ての分割鍵を持つ攻撃者であっても鍵を再構成することはできない。そのため、鍵のセキュリティは分散される。多くの他の鍵を保護し、その危殆化が深刻な惨事をもたらすようなルート鍵又はマスタ鍵を確立するために、知識分散手続きが使用されてきた。これらの分割鍵（分割鍵を合成することで得られる平文鍵ではなく）は、バックアップのために、平文形式で CKMS に入力又は CKMS から出力されることが多い。

FR:6.100 CKMS 設計は、鍵分割技術を使用して管理される全ての鍵を明記しなければならず、またそれぞれの技術に対して *n* と *k* を明記しなければならない。

FR:6.101 使用しているそれぞれの（*k, n*）鍵分割技術に対して、CKMS 設計は、鍵分割がどのように行われ、なぜ *n* 個中任意の *k* 個の分割鍵で鍵を構成できるが *k-1* 個の分割鍵では鍵に関する情報を何ら提供しないのかを示すあらゆる既知の論拠（論理、数学）を明記しなければならない。

6.8 危殆化からの回復（Compromise Recovery）

理想的な状況下では、CKMS は全ての鍵及び機微なメタデータを保護しており、それらは決して危殆化することも、認可されない当事者によって改変（改ざん）されることもない。しかしながら、全ての潜在的なセキュリティ問題を防止する完璧な CKMS を設計することは困難あるいは不可能ですらある。このため、CKMS は危殆化及び認可されない改変（改ざん）を検知するように設計すべきであり、望ましくない影響を軽減し、適切な当事者に危殆化を警告し、かつ危殆化又は認可されない改変（改ざん）が発見された際にセキュアな状態に回復する（又は回復を手助けする）ように設計すべきである。本節は、危殆化からの回復がどのように行われるべきかを取り扱う。

CKMS での危殆化が検出されたとき、

a) その原因及び範囲を探し出すために危殆化を評価すべきであり、

b) 鍵やメタデータの露出を最小化するために危殆化軽減手段を行うべきであり、

c) 危殆化の再発を防止するために適切な是正手段を行うべきであり、

d) CKMS をセキュアな運用状態に復帰させるべきである。

6.8.1 鍵の危殆化

鍵タイプ及び鍵の用途に依存し、鍵の危殆化は以下のような結果をもたらし得る：

a) 機密性の喪失

b) 完全性の喪失

c) 認証の喪失

d) 否認防止の喪失

e) これらの喪失の組み合わせ

鍵に提供されるセキュリティ処理（service）の喪失は、その鍵が保護対象とするデータに対する当該セキュリティ処理（service）のみでなく、他のセキュリティ処理（service）の喪失をももたらす可能性が高い。例えば、公開鍵配送鍵の完全性の喪失は、その公開鍵で保護されたデータ暗号化鍵の機密性に影響し、その結果、そのデータ暗号化鍵で保護されたデータの機密性の危殆化をもたらし得る（さらに具体的には、RSA 公開鍵が法 n で値 1 を持つように改ざんされた場合、改ざんされた鍵で暗号化されたあらゆるデータが危殆化し得る）。

鍵の危殆化は、検出されないことも検出されることも又は疑われることもあり得る。CKMS は、使用するそれぞれの鍵に対して暗号鍵有効期間（cryptoperiod）又は利用制限（usage limit）を設定することで、検出されない鍵の危殆化の露出を制限すべきである[18]。それぞれの暗号鍵有効期間（cryptoperiod）の終了時に、新しい鍵が確立され、古い鍵を置き換えるかもしれない。新しい鍵が確立されて新しいデータを保護するために活性化されたとき、古い鍵はもはや新しいデータを保護するために使用すべきでない。これにより、新しい鍵に対して危殆化が再発しない限り、新しいデータは保護される。もちろん、古い鍵で保護された古いデータは危殆化してしまったかもしれないが、古い鍵が新しい鍵を保護するために使用されていなかった限り（例えば、鍵配送において新しい鍵を保護するために古い鍵が使用されなかった）、危殆化の影響範囲は限定される。鍵の危殆化が検出された場合、危殆化した鍵及びその危殆化した鍵のセキュリティに安全性が依存する他の鍵は、可能な限り速やかに置き換えられるべきである。鍵の危殆化は、その鍵が保護する他の多くの鍵の危殆化をもたらす可能性があるため、鍵の危殆化の影響を最小化するように CKMS を設計することが重要である。［SP 800-57-part1］は様々な鍵タイプに対する適切な暗号鍵有効期間（cryptoperiod）のガイダンスを提供する。

鍵ラッピング対称鍵（symmetric key wrapping key）、プライベート鍵配送鍵（private key transport key）、又はプライベート鍵合意鍵（private key agreement key）が危殆化した場合、配送された又は合意された鍵も同様に危殆化しているかもしれない。危殆化が検出されない場合、さらなる鍵の危殆化が永遠に起こり続けるかもしれない。幾つかのプロトコルは、そのような攻撃を防止又は軽減するように設計されている。しかしながら、一般的には、鍵ラッピング対称鍵、鍵配送鍵、及び鍵合意鍵の暗号鍵有効期間（cryptoperiod）を現実的な最短期間にしておくことがよい考えであると見なされる。

鍵導出鍵又はマスタ鍵が危殆化した場合、その鍵導出鍵又はマスタ鍵から導出されたあらゆる鍵がまた危殆化した可能性がある。したがって、鍵導出鍵とマスタ鍵もまた定期的に変更すべきである。

FR:6.102 CKMS 設計は、システムによって使用されているそれぞれの鍵タイプの受け入れ可能な暗号鍵有効期間（cryptoperiod）又は利用制限（usage limit）の範囲を明記しなければならない。

FR:6.103 それぞれの鍵に対し、CKMS 設計は、セキュリティをその鍵に依存する他の鍵タイプを明記しなければならず、また初期鍵の危殆化が発生した時に、その初期鍵に依存する鍵がどのように置き換えられるかを明記しなければならない。

FR:6.104 CKMS 設計は、鍵が危殆化したときに他の危殆化した鍵を特定できるための手段を明記しなければならない。例えば、鍵導出鍵が危殆化したとき、導出された鍵をどのように特定するのか？

---

[18] 鍵の使用は、その鍵を使用して処理されたデータ量、又はその鍵を使用してアルゴリズムが初期化された回数などの基準に基づいて制限されることがある。

6.8.2 メタデータの危殆化

メタデータ要素及びその使われ方にもよるが、メタデータ要素の危殆化は、鍵の危殆化又は鍵によって保護されるデータの危殆化につながるかもしれない。例えば、暗号化／復号対称鍵のメタデータ要素は、その鍵の正規のユーザに対応する識別子（ID）のリストであるかもしれない。アクセスコントロールシステムは、メタデータ要素に対するユーザの認証された識別子（ID）を検証し、そのユーザが復号機能を実行して平文データを取得することが許可されているかどうかを判定する。メタデータ要素に対する認可されたユーザリストに、認可されていないユーザを加えるという改ざんが可能であった場合、暗号化されたデータは危殆化している可能性がある。異なった鍵が共通のメタデータ要素を持っている場合、一方のメタデータ要素の危殆化がそれぞれの鍵で保護されたデータの危殆化をもたらすかもしれない。認可されない改変（改ざん）の影響を受けやすいメタデータ要素は、そのメタデータの完全性を容易に検証できるように、そのメタデータの関連付けられる鍵と暗号学的に結びつけられるべきである。

FR:6.105 導入されたそれぞれの鍵タイプに対して、CKMS 設計は、どのメタデータ要素が危殆化（機密性、完全性、又はソース）の影響を受けやすいかを明記しなければならない。

FR:6.106 CKMS 設計は、鍵のそれぞれの危殆化の影響を受けやすいメタデータ要素に危殆化（機密性、完全性、又はソース）が起こったときに、起こり得るセキュリティ上の結果を明記しなければならない。

FR:6.107 CKMS 設計は、それぞれの危殆化の影響を受けやすいメタデータ要素の危殆化からどのように回復できるかを明記しなければならない。

6.8.3 鍵及びメタデータの失効

鍵は多くの理由によって失効させられる。その理由には、鍵の危殆化、及び組織内での役員・従業員の退職又は役割の終了が含まれる。CKMS は、鍵（非対称鍵と対称鍵の両方）を速やかに置き換える機能、及び依拠する当事者（その鍵を使用する者）に危殆化／失効を通知する機能を備えているべきである。

危殆化鍵リスト（CKLs)、証明書失効リスト（CRLs)（[RFC 5280］を参照)、ホワイトリスト、クエリホワイトリスト（Query White List)、及び OCSP（Online Certificate Status Protocol)（[RFC 6960］を参照）は、依拠する当事者に鍵の失効情報を周知するために使用されるメカニズムの例である。それぞれのメカニズムに長所と短所がある。例えば、CRL と CKL は膨大な量に膨れ上がり、すぐに最新でなくなる（つまり、古くなる）という問題がある。肥大化は、通信、計算及びストレージの要求に不利な影響を及ぼす。肥大化がエンドエンティティに与える問題は、失効情報を小さな塊に分割し、それぞれの塊がより少ない数の鍵を扱うようにすることで軽減できる。最新でなくなることは完全には防ぐことはできないが、より頻繁にリストを発行することで影響を軽減することができる。セキュリティ要求と依拠する当事者の制限を満たすために複数の失効メカニズムが使用できる実例もあることに留意されたい。

鍵失効メカニズムは、以下を考慮すべきである：

a) 失効情報の適時性に対する依拠する当事者の要求事項

b) 依拠する当事者の計算及び通信に対する限界

c) インフラコストの考慮

FR:6.108 CKMS 設計は、使用される又は使用できる鍵失効メカニズム及び関連付けられた依拠するエンティティへの通知メカニズムを明記しなければならない。

6.8.4　暗号モジュールの危殆化

暗号モジュールはある時点において平文鍵を保持しているため、暗号モジュールの危殆化はそのモジュールに保持されている対称鍵及びプライベート鍵の危殆化の可能性を伴う（8.4 節を参照）。これは、上記 6.8.1 節に記載されているように、機密性の喪失、完全性の喪失、又は認証能力の喪失につながり得る。暗号モジュールは、物理的に（すなわち、そのモジュール内の鍵への直接アクセスが得られる）、又は非侵襲的手段（そのモジュール内の鍵についての知識が何らかの外部からの操作によって得られる）によって危殆化され得る。物理的保護を提供するために、暗号モジュールが運用される場所は、認可されないアクセスが許可されない場所、又は深刻な危殆化が起こる前に認可されないアクセスが速やかに検出されるような場所であるべきである。ある種の暗号モジュールでは、この保護がそのモジュールの暗号境界において提供されるが、より大きな境界が関わることもある。暗号モジュール内部に対する物理的保護についてのさらなる情報については［FIPS 140］を参照されたい。暗号モジュール内部へのアクセスを許可する場合、認可された当事者のみがアクセスに成功することを保証するために、アクセスコントロールシステムが必要であるかもしれない。

暗号モジュールの実際の危殆化又は危殆化の疑いがあった後には、そのモジュールはセキュア状態に再確立されるべきである。この状態へ戻すために必要な行動は総称的に回復と呼ばれる。回復は、時には暗号モジュール内部のハードウェアやソフトウェアの置き換えを必要とすることがある。暗号モジュールは、通常運用に戻る前に、セキュア状態に戻すべきである。暗号モジュールは、修理又は交換の後に、セキュリティステータスとともに運用機能のテストを行わなければならない。

暗号モジュールへの非侵襲攻撃に対する保護を提供するため、そのモジュールの使用を信頼されるユーザに制限するか、又はこのような特定の種類の攻撃を防止するようにそのモジュールを設計するかを行うべきである。第一のケースでは、暗号モジュールが紛失又は盗難に遭うか、あるいは信頼されていたユーザが裏切る脅威が常に存在する。第二のケースでは、全ての可能なタイプの非侵襲攻撃に対して保護することは極めてコストがかかることがある。攻撃者は、暗号処理中の暗号モジュールの詳細な電力消費パターンを分析することで、そのモジュールが使用している暗号鍵の情報を推定することができるかもしれない。他の潜在的な非侵襲攻撃として、ある種の暗号機能の実行にかかる時間、あるいは通常動作中の暗号モジュールが出す電磁放射を注意深く分析することに基づくものもある。

FR:6.109 CKMS 設計は、暗号モジュール内部への物理的及び論理的アクセスがどのように認可されたエンティティに制限されるかを明記しなければならない。

FR:6.110 CKMS 設計は、暗号モジュールを危殆化から回復するために使用されるアプローチを明記しなければならない。

FR:6.111 CKMS 設計は、どの非侵襲攻撃が、そのシステムで使用される暗号モジュールによって軽減されるかを記載し、どのように軽減が実行されるかの記載を提供しなければならない。

FR:6.112 CKMS 設計は、非侵襲攻撃に脆弱であるあらゆる暗号モジュールを明記しなければならない。

FR:6.113 CKMS 設計は、可能性のある非侵襲攻撃によって起こりうる脆弱性を、受け入れる場合の原則を明記しなければならない。

6.8.5　コンピュータシステムの危殆化からの回復

分離されたセキュアなプラットフォーム上で動き、ファイルへのあらゆる改変、ファイル内容のハッシュ値の変化、又はファイル属性（例えば、誰が所有者であるか、誰が ACL に存在するか）の変化を監視するツールを使用することで、CKMS ソフトウェア又はコンピュータ OS の主要な部分に対する認可されない改変（改ざん）を検出することができる（8.2.4 節を参照）。代替として、多層の保護システムが CKMS に構築されることも多い。保護メカニズムがシステムに構築されるとき、そのメカニズムは、システムそのものへ起こりうる脅威に対して、システムそのものと同様に保護される必要

がある。重要なファイルに対する認可されない改変（改ざん）が、監視ユーティリティによって検出又はイベントログに表示されたとき、これらのファイルは、セキュアなストレージに保管された、正当でセキュアであると分かっているファイルを使って置き換えられるべきである。

認可されない改変（改ざん）が広範囲にソフトウェアになされた場合、そのソフトウェアは 10.5 節に記載されたように回復すべきである。

FR:6.114 CKMS 設計は、CKMS システムハードウェア、ソフトウェア、及びデータに対する認可されない改変を検出するために利用されるメカニズムを明記しなければならない。

FR:6.115 CKMS 設計は、CKMS システムハードウェア、ソフトウェア、及びデータに対する認可されない改変からどのように CKMS が回復するのかを明記しなければならない。

6.8.6 ネットワークセキュリティコントロール及び危殆化からの回復

CKMS へ保護を提供するネットワークセキュリティコントロールの危殆化は CKMS そのものの危殆化につながり得る。ネットワークセキュリティコントロールのスコープは、ファイアウォール、VPN、侵入検知システム及び侵入保護システムのような境界デバイス（boundary device）を含む。ネットワークセキュリティコントロールのスコープには、前述のネットワークセキュリティコントロールデバイスの運用のために使用される場合を除いて、暗号機能、暗号プロトコル及び暗号処理（service）は含まれない。

以下がネットワークセキュリティコントロールの危殆化のいくつかの例である：

a) ネットワークセキュリティコントロールデバイスの物理的危殆化

b) ネットワークセキュリティコントロールデバイスで使用されるひとつ以上の暗号鍵の危殆化

c) ネットワークセキュリティコントロールデバイスを管理するために使用されるひとつ以上の鍵の危殆化

d) 危殆化につながるネットワークアーキテクチャの変更（例えば、誰かが VPN 接続されたワークステーションをセキュアでないネットワークに接続し、VPN ワークステーションがイントラネットを攻撃するために使用される）

e) 特権ユーザのパスワード（例えば、システム管理者のパスワード）の危殆化

f) プラットフォーム OS の危殆化

g) ネットワークセキュリティアプリケーション（例えば、ファイアウォール、IDS 等）の危殆化

h) プロトコルへの新しい攻撃による危殆化

物理セキュリティが危殆化した場合、デバイスは新しいデバイスに交換し、かつ物理セキュリティコントロールを適切に見直し、修理し、強化すべきである。

デバイス又は管理鍵が危殆化した場合、鍵は置き換えるべきである。危殆化の原因、被害範囲及び取るべき是正措置を確定するためのアセスメントを実行するべきである。低確率ではあるが、鍵のセキュリティ強度が問題になるようなイベントにおいては、鍵長を延ばすことが必要になったり、よりセキュアな暗号アルゴリズムの使用が必要になったりするかもしれない。

ネットワークアーキテクチャの前提に違反した場合、違反の原因をレビューし、適切な行動を取るべきである。

危殆化したネットワークデバイスは、通常動作を続ける前にセキュア状態に回復すべきである。

パスワードが危殆化した場合、パスワードを置き換えるべきである。ユーザには、パスワードの選択、パスワードエントロピーの理解、パスワードの頻繁な変更及び紙に書かれたパスワードの機密性

の維持についてのさらなる研修を必要とするかもしれない。また、パスワードスニッフィング（傍受）、オンライン辞書攻撃又はオフライン辞書攻撃が現実的かどうかを見極めるために、認証プロトコルの検査も実施すべきである。

プラットフォーム OS が危殆化した場合、以下のひとつ以上の行動を検討し、適切な是正手段を取るべきである：

a) 全ての最新の OS セキュリティパッチがインストールされていることを確認する

b) OS ベンダに危殆化に対するパッチが存在するかどうかを問い合わせる

c) デバイス設定変更又はある種のプロトコルのブロッキングが、危殆化を引き起こしたものと同じ特性を持つ将来の攻撃を防止できるかどうか見極める

ネットワークセキュリティアプリケーションが危殆化した場合、以下のひとつ以上の行動を検討し、適切な是正手段を取るべきである：

a) 全ての最新のネットワークセキュリティパッチがインストールされていることを確認する

b) アプリケーションベンダに危殆化に対するパッチが存在するかどうかを問い合わせる

c) デバイス変更、アプリケーション設定変更、又はある種のプロトコルのブロッキングが、危殆化を許す又は引き起こす攻撃を将来にわたって防止できるかどうかを見極める

危殆化が不適切なネットワークセキュリティプロトコルに起因する場合、以下のひとつ以上の行動を検討し、適切な是正手段を取るべきである：

a) ネットワークセキュリティアプリケーションベンダに危殆化に対するパッチが存在するかどうかを問い合わせる

b) デバイスの設定変更又はある種のプロトコルのブロッキングが、危殆化を引き起こしたものと同じ特性を持つ将来の攻撃を防止できるかどうか見極める

これらの全ての状況において、インシデントを完全に調査し、ネットワークセキュリティコントロールの危殆化に起因して他のシステム及び鍵のどれが危殆化した可能性があるのかを特定するべきである。このような被害のアセスメントは、追加の危殆化の申告及び追加の危殆化回復手続きにつながることがある。

FR:6.116 CKMS 設計は、システムによって使用されるネットワークセキュリティコントロールの危殆化からどのように回復するかを明記しなければならない。特に、

a) CKMS 設計は、それぞれのネットワークセキュリティコントロールデバイスに対して考えられる危殆化シナリオを明記しなければならない。

b) CKMS 設計は、それぞれの想定される危殆化シナリオに対して、この節に記載されたどの軽減技術が採用されるかを明記しなければならない。

c) CKMS 設計は、採用されるあらゆる追加又は代替の軽減技術を明記しなければならない。

6.8.7 役員・従業員によるセキュリティ危殆化からの回復（Personnel Security Compromise Recovery）

CKMS の正しくセキュアな運用に責任のある人間は、そのセキュリティを危殆化させる能力を持っていることが多い。しかしながら、CKMS は、人間による危殆化の可能性を最小化し、危殆化を検知

し、危殆化のネガティブな結果を最小化し、かつ危殆化から効率的に回復するための自己機能を持つように設計することができる。

CKMS は以下のように設計すべきである：

a) 人間の、セキュリティ障害を起こす能力を最小化する

b) セキュリティ障害を起こした行動に対する、人間の隠蔽する能力を最小化する

c) 誰が、又は何がセキュリティ障害を起こしたのかを特定することを補助する（例えば、監査記録を保持することによって）

d) 障害のネガティブな結果を軽減する

検知されたあらゆるセキュリティ障害は、情報セキュリティポリシー及び CKMS 機能に基づいた回復手続きの開始につながるべきである。

典型的な対応は、以下を含む：

e) システムの完全なシャットダウン

f) 新しい鍵による、バックアップ設備及びシステムの活性化

g) 起こり得るセキュリティ障害についての、現在の及び潜在的なユーザへの通知

h) 危殆化した鍵へのフラグ付け

上記の対応に加えて、役員・従業員による危殆化が関わる障害は、管理上の懲戒から、役割又は地位からの解任、及び民事又は刑事裁判に関わる法的措置まで多岐に亘りうる。

FR:6.117 CKMS 設計は、それぞれのサポートされる役割に提供される、あらゆる役員・従業員による危殆化の検知機能を明記しなければならない。

FR:6.118 CKMS 設計は、それぞれのサポートされる役割に提供される、あらゆる役員・従業員による危殆化を最小化する機能を明記しなければならない。

FR:6.119 CKMS 設計は、それぞれのサポートされる役割に提供される、CKMS 危殆化からの回復能力を明記しなければならない。

6.8.8 物理セキュリティ危殆化からの回復

暗号モジュールの物理セキュリティは 6.8.4 節で説明され、鍵及びメタデータの一般的な危殆化はそれぞれ 6.8.1 節及び 6.8.2 節で説明されている。しかしながら、CKMS の物理セキュリティ侵害は、鍵又は暗号モジュールの危殆化とは別の危殆化をもたらすかもしれない。セキュリティ関連ロジックが CKMS 暗号モジュールの外部に存在する場合、そのロジックの完全性もまた保護されるべきである。典型的には、暗号モジュールによって使用される技術と類似の技術が採用される。物理的保護により、潜在的な攻撃者がコンポーネント及びデバイスへの物理アクセスを得ることを防ぐことができる。代替として、認可されないアクセス（不正アクセス）を検知しその後にある種の防御行動を起こすために、検知メカニズムを使用することもあり得る。例えば、認可されないアクセス（不正アクセス）が検知されたとき、警告音を発する、あるいはセキュリティ管理者に警告を送信することがあり得る。物理アクセス防止技術と検知メカニズムを組み合わせた手段が使用されることも多い。

一旦セキュリティが侵害されると、侵害された領域全体の完全性が疑わしくなるはずである。CKMS が侵害を検知した場合、CKMS セキュリティポリシーに規定された通りに、適切なエンティティに侵害について通知し、これらのエンティティが軽減行動を取れるようにすべきである。さらに、侵害された領域内の全ての機微なデータを置き換えるだけでは十分でないかもしれない。なぜなら、

攻撃者は、将来においても新しい鍵及び機微な情報を危殆化させられるように、領域内のロジックを改ざん又は追加しているかもしれないためである。

FR:6.120 CKMS 設計は、全ての CKMS コンポーネント及びデバイスがどのように認可されない（不正な）物理アクセスから保護されるかを明記しなければならない。

FR:6.121 CKMS 設計は、CKMS がどのように認可されない（不正な）物理アクセスを検知するかを明記しなければならない。

FR:6.122 CKMS 設計は、CKMS がどのように暗号モジュール以外のコンポーネント及びデバイスへの認可されない（不正な）物理アクセスから回復するかを明記しなければならない。

FR:6.123 CKMS 設計は、あらゆる CKMS のコンポーネント又はデバイスへの物理セキュリティ侵害が CKMS によって検知されたときに、自動的に通知されるエンティティを明記しなければならない。

FR:6.124 CKMS 設計は、侵害された領域がどのようにセキュアな状態に再確立できるかを明記しなければならない。

7　相互運用性及び移行

相互運用性とは、多様なシステムが通信し共に動作する（すなわち、相互運用する）能力である[19]。CKMS は、アプリケーション又は対向の CKMS と相互運用するかもしれない。相互運用性は、CKMS が相互運用しようとするシステムへのインタフェースの詳細な仕様を有することでのみ達成できる。これは本質的に以下を含む：

a) 共通のインタフェース及びプロトコル：すなわち、ある CKMS エンティティから別の CKMS エンティティの機能及びサービスを呼び出すためのインタフェースの、シンタックスとセマンティクスが、相互運用するシステムにおいて同一である

b) 鍵、メタデータ、及びその他の交換するデータのフォーマットが、相互運用するシステムにおいて同一であるか変換可能である

c) セキュリティメカニズムを含むデータ交換メカニズムが、相互運用するシステム間で同一であるか同等である

使用中の暗号アルゴリズムは、必要なときに拡張又は置き換えができるように実装されるべきである。［SP 800-57-part1］及び［SP 800-131A］は、政府承認暗号アルゴリズムの NIST 推奨ライフタイムを規定している。CKMS は、CKMS 及びそれが保護する情報の、予定するライフタイムをカバーするような、セキュリティライフタイムを持つアルゴリズムのみ使用すべきである。CKMS がその暗号アルゴリズム及び鍵長のセキュリティライフタイムを超えてサービスを続けることを意図するのであれば、将来的により強力なアルゴリズム及び鍵長に移行するための移行戦略を用意するべきである。

ある暗号アルゴリズムから別のアルゴリズムへ移行するとき、円滑な移行のためには、少なくとも 2 つのアルゴリズム（異なった鍵長であるかもしれない）の利用を同時にサポートする機能が要求されることが多い。これは、全ての参加者が新しいアルゴリズムで運用する能力を持つようになるまで相互運用性を維持できるようにするためである。この場合、暗号プロトコルは、特定の鍵確立トランザクションでどのアルゴリズムを使用するかを特定及びネゴシエーションするように設計されるべきである。また、異なった時期に異なったアルゴリズムによって保護されるデータのセキュリティは、最

[19] 相互運用性の能力と使用についての詳細は、http://en.wikipedia.org/wiki/interoperability を参照されたい。

も弱いアルゴリズムのセキュリティを上回らないことにも留意すべきである。それゆえ、可能な限り素早く移行することが最善であると考えられる。

FR:7.1 CKMS 設計は、デバイスのインタフェース間の相互運用性の要求事項がどのように満たされるかを明記しなければならない。

FR:7.2 CKMS 設計は、サポートすることを意図しているアプリケーションとの相互運用に必要な標準、プロトコル、インタフェース、サポートする処理（service)、コマンド、及びデータフォーマットを明記しなければならない。

FR:7.3 CKMS 設計は、相互運用性を意図している他の CKMS との相互運用に必要な標準、プロトコル、インタフェース、サポートする処理（service)、コマンド、及びデータフォーマットを明記しなければならない。

FR:7.4 CKMS 設計は、アプリケーション及び他の CKMS に対する全ての外部インタフェースを明記しなければならない。

FR:7.5 CKMS 設計は、新規の、相互運用可能な、移行先となるデバイスへの移行のための全ての規定を明記しなければならない。

FR:7.6 CKMS 設計は、暗号アルゴリズムのアップグレード又は置き換えのために提供されるあらゆる規定を明記しなければならない。

FR:7.7 CKMS 設計は、暗号アルゴリズムの移行期間中に、どのように相互運用性をサポートするかを明記しなければならない。

FR:7.8 CKMS 設計は、暗号アルゴリズムと鍵長の使用をネゴシエーションするプロトコルを明記しなければならない。

8　セキュリティコントロール

CKMS は、CKMS のコンポーネント及びデバイスを、それらに含まれるデータとともに、保護するためのセキュリティコントロールを要求する。例えば：

a) CKMS は、コンポーネント、デバイス、及び CKMS 内に含まれる機微なデータが認可されない開示（窃取）及び改ざんから保護されるように、物理的に保護されるべきである。

b) CKMS は、鍵生成、鍵保管、鍵復元、鍵配付、暗号モジュールコントロール、及びメタデータ管理のような機能を実行するためにコンピュータシステムを必要とする可能性が高い。これらの機能が正しく実行されることを保証するためのコントロールが存在すべきである。

c) CKMS は、鍵及びメタデータをユーザ及び他の末端エンティティに配付するためにネットワーク接続されている可能性が高い。そのような状況では、CKMS はネットワークセキュリティコントロールデバイスを使用して保護されるべきである。

d) CKMS は、暗号モジュールを使用して暗号鍵を生成、保管、使用、及び保護すべきである。

e) CKMS は、暗号モジュールから出力される前に、鍵に必要な暗号学的保護を適用すべきである。

この章の以下の小節では、それぞれの種類のセキュリティコントロールに対する要求事項を記載する。

8.1 物理セキュリティコントロール

CKMS のコンポーネント及びデバイスは、情報セキュリティを保証するために物理的に保護されるべきである。良好な物理セキュリティなしでは、コンポーネント及びデバイスが細工され、ハードウェアやソフトウェアの改ざんによりセキュリティが回避されうる。

CKMS は、認証局、鍵配付センタ、登録局又は認証ディレクトリ（Certificate Directory）のような第三者鍵管理サービスを提供する設備、さらには PC、携帯情報端末（PDA）、スマートフォン及びインテリジェントセンシングデバイス（intelligent sensing device）のようなエンド間通信デバイス（end-to-end communication device）を提供する設備を含むかもしれない。

CKMS は、1 つ以上の基幹設備と 1 つ以上の災害復旧機能を提供するバックアップ設備を含んでいることがある。これらの設備はそれぞれに、設備全体へのアクセスコントロール、若しくは設備内の機密性の高いコンポーネントやデバイスへのアクセスコントロールを行うことによる、物理的保護が施されているべきである。災害復旧のためのバックアップ設備の使用については 10.4 節で説明する。

CKMS のコンポーネントやデバイスのセキュリティの重要性に応じ、CKMS 設備を物理的に保護するために下記のメカニズムのうち 1 つ以上が選択されるべきである。下記は物理セキュリティメカニズムの例である。下記のメカニズム一覧の中には、適切な防止メカニズムによる補強が別途必要となる検出メカニズムもある。

a) フェンス

b) 門、ドア、及びカバー

c) 警備員

d) 錠（物理鍵又は組み合わせ数字（combination））

e) 細工検出及び防御

f) パスワード

g) 社章、社員証

h) カード及びトークンシステム

i) 生体認証デバイス

j) 警報システム

k) 監視カメラ

l) 監査システム

m) 入退室ログ

FR:8.1 CKMS 設計は、それぞれの CKMS デバイスと意図する目的を明記しなければならない。

FR:8.2 CKMS 設計は、CKMS コンポーネントを内包する各デバイスを保護するための物理セキュリティコントロールを明記しなければならない。

8.2 OS 及びデバイスのセキュリティコントロール

本節では、OS 及び CKMS デバイスの、コンピュータセキュリティコントロールについて取り扱う。汎用 OS を組み込んだ CKMS デバイスもまたコンピュータセキュリティコントロールを有するべきであることに留意されたい。

8.2.1 OS セキュリティ

セキュアな OS はセキュアなコンピュータシステムの基礎である。セキュアな OS なしでは、コンピュータシステム上のプログラム及びデータのセキュリティを保証することはできない。セキュアな OS は以下のセキュリティ機能を持つ：

i. 起動時の OS の正しいインスタンス化を保証するための BIOS 保護機能（[SP800-147] 参照）

ii. ユーザ及びユーザプロセスによる認可されない改変（改ざん）から OS を保護するための自己保護機能

iii. ユーザ及びユーザプロセスに対して分離された実行ドメインを提供及びメンテナンスすることで、それらが互いに干渉せず、それによってデータ分離のためのセキュリティポリシー要求事項を侵害しないようにするための隔離機能

iv. ユーザ、グループ又はその他のメタデータ要素に基づいて、ユーザにデータを共有することを許可するための、アクセスコントロール及び OS 機能

v. 個人の説明責任（personal accountability）をサポートし、また異常調査を行うための、イベントログ機能

vi. エンティティ識別及び認証を含む、CKMS アカウントのユーザ管理

セキュアな OS（のセキュリティ）は、セキュアなコードが動いている信頼されるハードウェアプラットフォームに依存する。信頼されるハードウェアプラットフォームは、メモリ及び I/O 管理のような特権操作を提供するために 2 つ以上の状態を実行することが多い。

ある状況では、セキュアな OS は隔離カーネル（ハイパーバイザとしても知られる）であり、それはゲスト OS 及びその上で動く CKMS アプリケーション用の仮想マシンを提供する。このアーキテクチャにおいて、隔離カーネルはゲスト OS をアプリケーションと見なす。

専用のセキュリティ機能を実行し汎用機能を持たないように構築された CKMS デバイスは、縮退した又は最小限の OS 要求事項を持てばよい。例えば、侵入検知機能を実行するためのファームウェアやソフトウェアを搭載した特定目的のデバイスを考える。このデバイスは OS を持たず、したがって OS セキュリティ要求事項は必要ないかもしれない。別の例は、追加のコードをロードする機能が利用できないように“locked-down”（すなわち、変更不可）な OS 上に構築されたファイアウォール又は侵入検知システムである。

FR:8.3 CKMS 設計は、それぞれの CKMS デバイスに対して、セキュアな OS への全ての要求事項（必要な OS 設定も含む）を明記しなければならない。

FR:8.4 CKMS 設計は、下記のどの堅牢化[20]機能が CKMS によって実施されているかを明記しなければならない：

a) 全ての必須でないソフトウェアプログラムとユーティリティをコンピュータから削除する

b) 機微なシステム機能及びアプリケーションに対するアクセスコントロールに最小権限の原則を適用する

c) 機微なシステム及びアプリケーションのファイルとデータに対するアクセスコントロールに最小権限の原則を適用する

d) ユーザアカウントを正規の運用に必要なだけに制限する、すなわち、もはや必要のないアカウントは無効化又は削除する

[20] 堅牢化については 11.4 節でさらに説明する。

e) 最小権限の原則でアプリケーションを動作させる

f) 全てのデフォルトパスワード及びデフォルト鍵をそれぞれ強力なパスワード及びランダムに生成された鍵で置き換える

g) システムの運用に必要でないネットワークサービスを無効化又は削除する

h) システムの運用に必要でない全ての他の処理（service）を無効化又は削除する

i) リムーバブルメディアを無効化する、又はリムーバブルメディアにおける自動実行機能を無効化しメディア挿入時の自動マルウェアチェック機能を有効にする

j) システム運用に必要でないネットワークポートを無効化する

k) オプションのセキュリティ機能を適切に有効化する

l) 他の設定オプションの選択の結果、セキュアであるようにする

FR:8.5 CKMS 設計は、OS の正しいインスタンス化を保証する BIOS 保護機能を明記しなければならない。

8.2.2 個々の CKMS デバイスセキュリティ

CKMS は様々なデバイスから構成されることがある。それぞれのデバイスが、認可されない使用から自らを保護するように設計されていることが好ましい。そうでなければ、外部から適用される保護が必要である。システム設計及び機能要求事項にも依存するが、CKMS デバイスは、より精緻なアクセスコントロール、及びホスト OS に取り込まれないデバイス特有のイベントログを提供するかもしれない。例えば、暗号モジュールは独自のアクセスコントロールシステムを有していることもある。そのため、よく設計された CKMS デバイスは以下のセキュリティ機能を備えるべきである：

a) 他の CKMS デバイスからの自己保護（例えば、OS のプロセス分離機能の使用）

b) CKMS デバイスユーザからの自己保護

c) ユーザ及びユーザプロセスに対して分離したセッションを提供し保全することで、それらが互いに干渉しないようにし、それによってデータ分離のセキュリティポリシーに違反しないようにするための隔離機能

d) ユーザ、グループ又はその他のメタデータ要素に基づいて、ユーザにデータを共有することを許可するための、CKMS デバイスレベルのオブジェクト（例えば、鍵及びメタデータ、又はデータベース管理システムの行とテーブル）への精緻なアクセスコントロール

e) 個人の説明責任（Personal accountability）をサポートし、また異常調査を行うための CKMS デバイスレベルのイベントログ

f) CKMS のためのユーザアカウント管理

FR:8.6 CKMS 設計は、それぞれの CKMS デバイスに必要なセキュリティコントロールを明記しなければならない。

FR:8.7 CKMS 設計は、堅牢化の基となるデバイス／CKMS のセキュリティ設定要求事項及びガイドラインを明記しなければならない。

8.2.3 マルウェアからの保護

通信、データ、ファイル等を保護されていないネットワークを通して受信する CKMS デバイスは、受信した情報にマルウェアに対するスキャンをすべきである。保護されていないネットワークを通して情報を何も受け取らない、又は全ての情報が強力に（例えば暗号学的に）認証されている場合には、マルウェア対策の重要性は下がるかもしれない。マルウェア対策は以下の 3 つの一般的なカテゴリに分けられる：

a) 意図しない動作を行い、セキュリティ上の危殆化を引き起こす可能性があるプログラムが無意識にインストールされ実行されることから、CKMS デバイスを保護するウイルス対策ソフトウェア

b) システム管理者ステータス又は認可されたユーザのステータスを、認可されない当事者が取得することから CKMS デバイスを保護し、スパイウェアが認可されたデバイスのように動作することを防止するスパイウェア対策ソフトウェア

c) システムコードを置き換えて、ウイルス対策ソフトウェアやスパイウェア対策ソフトウェアからプロセス及びファイル（ルートキットコード自体を含む）を隠蔽するために、OS のコンフィグ設定を変更するようなルートキットマルウェアから CKMS デバイスを保護するルートキット検出防御ソフトウェア

OS 及び CKMS アプリケーションソフトウェアの完全性は、インストール時及びその後定期的にチェックすべきである。インストール時のソフトウェア完全性の検証の例には、ソフトウェアの製造・流通過程の管理、及びソフトウェアが改ざんされていないことを保証するために使用される完全性コード（例えば、ハッシュ値、メッセージ認証コード、及びデジタル署名）の検証がある。定期的な検証の例には、インストールされたソフトウェアのハッシュ値、メッセージ認証コード、デジタル署名、及び更新日の検証等を日々行うことが含まれる。

効果的にするため、マルウェア対策は以下を実行するように設定されるべきである：

a) インストールされたソフトウェアの日々のスキャン

b) CKMS に、そのリムーバブルメディアが最初に挿入された時のスキャン

c) 新たにインストールされたソフトウェア及びデータファイルのスキャン

d) マルウェア対策ソフトウェアの週ごとの更新

e) マルウェアシグネチャデータベースの週ごとの更新

FR:8.8 CKMS 設計は、CKMS デバイスに対する以下のマルウェア防御能力を明記しなければならない：

a) ウイルス対策ソフトウェア。アンチウイルススキャン、ソフトウェア更新、及びウイルスシグネチャデータベース更新を開始する時間周期及びイベントの指定を含む。

b) スパイウェア対策ソフトウェア。アンチスパイウェアスキャン、ソフトウェア更新、及びウイルスシグネチャ更新を開始する時間周期及びイベントの指定を含む。

c) ルートキット検出及び防御ソフトウェア。ルートキット検出、ソフトウェア更新、及びシグネチャ更新を開始する時間周期及びイベントの指定を含む。

FR:8.9 CKMS 設計は、OS 及び CKMS アプリケーションソフトウェアに対する、以下のソフトウェア完全性チェックの情報を明記しなければならない：

a) ソフトウェア完全性がインストール時に検証される場合、検証がどのように実行されるかを記載する

b) ソフトウェア完全性が定期的に検証される場合、検証が実行される頻度を記載する

8.2.4 監査及びリモート監視

CKMS は、イベント、イベントの発生日時、及びイベントを発生させたエンティティの識別子（ID）又は役割を検知及び記録することによって、セキュリティ関連イベントを監査すべきである。監査ログは、実行された関連するセキュリティ機能の記録を提供すべきである。監査機能は、いくつかの CKMS デバイス及び場所に分散しているかもしれない。また、監査機能は、監査管理者に対し可能な限り速やかに調査すべきあらゆる異常なイベントを、検知し報告する機能も備えるべきである。監査機能及び監査ログは、監査システムの完全性が保証できるように、認可されない改変（改ざん）から保護されるべきである。

セキュリティ設定共通化手順（Security Content Automation Protocol；SCAP）に規定されているような自動評価ツールは、現在のステータス及びコンピュータシステムの完全性を評価するために、ますます有用になってきている。これらのツールは、その状態をリアルタイムに測定するために、OS に対して問い合わせることができる（[SP800-126] を参照）。ソフトウェアのバージョンナンバーが最新であることをチェックでき、データファイルの完全性及び機密性を検証できる。監視ツールは、監視されるプラットフォーム上で実行されることも、他のホストを監視することに専念する別のプラットフォーム上で実行されることもある。また、これらの監視ツールは、システムファイル又はそれらのアクセスコントロール属性の改変を検知し、警告及び監査イベントを発することができる（6.8.5 節を参照）。

FR:8.10 CKMS 設計は、サポートされている監査可能イベントを明記し、それぞれのイベントは固定されているか選択可能であるかを示さなければならない。

FR:8.11 CKMS 設計は、それぞれの選択可能な監査可能イベントに対し、どの役割であれば、それらのイベントを選択する能力を持つかを明記しなければならない。

FR:8.12 それぞれの監査可能イベントに対し、CKMS 設計は、記録されるデータ[21]を明記しなければならない。

FR:8.13 CKMS 設計は、CKMS の正しい運用及びセキュリティを評価するために、どの自動化ツールが提供されているかを明記しなければならない。

FR:8.14 CKMS 設計は、システムファイルの改変又は（アクセスコントロールリストのような）セキュリティ属性のあらゆる改変について検知や防止をするため、機微なシステムファイルに対するシステム監視要求事項を明記しなければならない。

8.3 ネットワークセキュリティコントロールメカニズム

本節は、CKMS に関わる各コンピュータシステムのネットワークセキュリティコントロールメカニズムについて取り扱う。ネットワークセキュリティコントロールメカニズムの例には、以下のものが含まれる：

a) ファイアウォール

b) フィルタリングルータ

c) 仮想プライベートネットワーク（VPN）

---

[21] 記録されるデータの例としては、固有のイベント識別子、イベントの日時、イベントを引き起こす主体（例えば、ユーザ、役割、又はソフトウェアプロセス）、イベントの成功又は失敗、及びイベント特有のデータが含まれる。

d) 侵入検知システム（IDS）

e) 侵入防止システム（IPS）

f) 適応型ネットワークセキュリティコントロール

i. 適応型フィルタリングメカニズム

ii. 適応型検出メカニズム

iii. 適応型防御メカニズム

ネットワーク接続された CKMS デバイスは、ファイアウォール及び侵入検知・防止システムの組み合わせを使用して保護されるべきである。ファイアウォールは、不要で潜在的に危険なプロトコルを除外し、許可されたプロトコルデータを検査して攻撃の成功確率を減らすことによって保護を提供する。一方、侵入検知・防止システムは、ホスト及びネットワークの活動を調べてシステムが攻撃されているかどうかを判定し、検出された攻撃を防止することによってファイアウォールを補完する。したがって、ファイアウォールと侵入検知・防止システムの両方を使用すべきである。

境界コントロールデバイス（ファイアウォール、フィルタリングルータ、VPN、IDS、及び IPS など）は、コンピュータシステム上に設置されるか（8.2 節を参照）、専用ハードウェア上で実装される。これらのデバイスは、物理的にセキュアな場所に配置されるべきであり（物理セキュリティコントロールは 8.1 節を参照）、セキュアな操作に必要なユーザアカウントとネットワークサービスのみを設定するべきである。多層防御を提供するために、境界コントロール機能も CKMS デバイスに直接実装されるべきである。そのようなコントロールは、「ホストベースの」ファイアウォールと呼ばれる。

FR:8.15 CKMS 設計は、CKMS によって採用される境界保護メカニズムを明記しなければならない。

FR:8.16 CKMS 設計は、以下を明記しなければならない：

a) 使用されるファイアウォールのタイプとファイアウォールを介して許可されるプロトコル。それぞれのプロトコルタイプの発信元（source）と宛先（destination）を含む

b) 使用される侵入検知・防止システムのタイプ。ログ及びセキュリティ侵害対応の機能を含む

FR:8.17 CKMS 設計は、CKMS デバイスをサービス拒否（DoS）攻撃から保護するために使用される方法を明記しなければならない。

FR:8.18 CKMS 設計は、使用されるそれぞれの方法がどのようにサービス拒否攻撃から保護するかを明記しなければならない。

8.4 暗号モジュールコントロール

暗号モジュールは、暗号ベースのセキュリティ機能（例えば、暗号アルゴリズム及び鍵確立スキーム）を実装するハードウェアやソフトウェア、ファームウェアの集合である。暗号モジュールは、暗号境界[22]内の全てを包含し、境界自体を含む。各暗号モジュールは、暗号モジュールセキュリティポリシー（例えば、[FIPS 140] 参照）に従い、実施するように作られるべきである。

暗号モジュール内部のセキュリティに関して、2 つの主要なセキュリティ上の問題を取り扱うべきである：セキュリティ機能の完全性と、暗号鍵及びメタデータの保護である。例えば、[FIPS 140] は、モジュール内の鍵の保護及びモジュールのセキュリティ機能の完全性の維持についての暗号モジュールの要件を規定している。ソフトウェア／ファームウェア完全性テスト（software/firmware integrity test）や既知解テスト（known-answer test）などの技術は、認可されていないアクセス（不正アクセ

[22] 暗号境界とは、暗号モジュールの全てのコンポーネントの境界を規定する明示的に定義された境界線のことである。

ス）や改ざんからの物理的保護と併せて、FIPS で規定されている。暗号鍵はモジュール内において一定期間平文形式で存在するため、認可されていない開示（窃取）、改ざん及び不正置き換えから鍵を保護するための物理的なセキュリティ対策を必要とする。暗号モジュールは、暗号モジュールを保護するために必要な物理的保護を提供することがある。それ以外の場合は、モジュールを含むより広範な物理的に保護された空間を必要とする。

ハードウェア暗号製品又はモジュールのベンダは、強固なメタルケース、錠、アラーム、及び暗号鍵破壊メカニズムを使用して、デバイスに物理的なセキュリティ保護手段を構築することが多い。しかしながら、汎用コンピュータ上で動作するソフトウェア暗号アプリケーションでは追加のリスクに直面する。なぜなら、コンピュータは暗号鍵への十分な保護を提供するようには設計及び実装されていないためである。実際、暗号化が実行される多くのコンピュータには、通常、個人が作成したセキュリティが検証されていないソフトウェアが含まれている。したがって、汎用コンピュータで実行されている暗号ソフトウェアでは、物理的に保護されている（つまり、汎用コンピュータが安全な環境に置かれている）こと、及び信頼できない（敵対的であるかもしれない）コードによる攻撃から論理的に保護されていることが重要となる。［FIPS 140］は、これらの保護に関する指針を提供している。

FR:8.19 CKMS 設計は、以下を含む、使用する暗号モジュール及びそれぞれのセキュリティポリシーを特定しなければならない：

a) それぞれのモジュールの実装形態（ソフトウェア、ファームウェア、ハードウェア、又はハイブリッド）

b) それぞれのモジュールの完全性を保護するために使用されるメカニズム

c) それぞれのモジュールの暗号鍵を保護するために使用される物理的及び論理的メカニズム

d) それぞれのモジュール（セキュリティ機能を含む）で実行された第三者試験と検証、及びそれぞれのモジュールで使用される保護措置

9 テスト及びシステム保証

CKMS デバイスは、以下のことを保証するために、いくつかの種類のテストを受けるべきである：

- CKMS 設計に適合するように構築されていること
- 所定の標準に適合していること
- CKMS 設計に従って運用を継続すること
- セキュリティを危殆化させる可能性がある機能を実行しないこと
- 他の CKMS デバイスと相互運用できること
- セキュリティを維持する合理的な保証を意図したより大きなシステムにおいて使用できること

テストは、典型的には全ての可能性の組み合わせ数よりはるかに少ない有限個のケースに限定されるため、デバイス又はシステムが全てのケースにおいて正しい又はセキュアであることを保証しない。そのため、テストスイートに合格することの価値は、選択したテストケースの包括性及び代表性に直接関連する。

CKMS デバイスは以下に一覧を挙げるカテゴリのテストを受けることがある。

9.1 ベンダテスト

デバイスベンダは、誤りを検知して除去し、期待通りに動くことを検証するためにデバイスをテストする。このカテゴリのテストの技術及び仕様は、ベンダによってプロプライエタリ情報と見なされることが多く、一般に公開されない。

FR:9.1 CKMS 設計は、システムで実行され合格した非プロプライエタリベンダテストを明記しなければならない。

9.2 第三者テスト

ベンダ又は顧客は、特定の標準への適合性について第三者が CKMS デバイスをテストするように求めるかもしれない。第三者テストは、ベンダが自身のテスト手順のなかで欠陥を見逃していないことの信頼性を提供する。例えば、NIST（National Institute of Standards and Technology）は、暗号標準及び推奨事項への製品適合を検証するいくつかのプログラムを確立している。

FR:9.2 CKMS 設計は、CKMS 又はデバイスが今までに合格した全ての第三者テストプログラムを明記しなければならない。

9.3 相互運用性テスト

相互運用性テストは、最も一般的な形式においては、ただ単に 2 つ以上のデバイスが相互接続し、互いに運用することができるかどうかをテストする。これは、デバイス間で交換されるデータがそれぞれのデバイスで処理できる形式になっていることを意味する。相互運用可能なデバイスは相互接続されてシステムを形成することがあり、相互運用可能なシステムは相互接続されてネットワークを形成することがある。この種のテストは、必ずしも個々のデバイスの内部機能をテストするわけではないことに留意されたい。デバイスが固有の機能を実行したとしても、相互運用性テストはその機能が正しく動作することを検証しているとは限らない。

別の形式の相互運用性テストは、テスト対象デバイス（device-under-test）が正しく動いているように見えるかどうかを検証するために利用される。同一又は逆の機能を実行する別のデバイス（保証ベースラインデバイス（assured-baseline device））が正しく動作することをテストして検証されていれば、テスト対象デバイス（device-under-test）は保証ベースラインデバイス（assured-baseline device）と相互運用するかどうかを検証するためのテストをするかもしれない；これは、テスト対象デバイス（device-under-test）が正しく動作することのある程度の保証を提供する。例えば、鍵確立を実行するデバイスは、正しく動作すると信じられている別の同種のデバイスによってテストされることがある。2 つのデバイスが確立された鍵に同意すれば、テストは合格である。テスト対象デバイス（device-under-test）と保証ベースラインデバイス（assured-baseline device）が、異なる組織によって独立に設計され実装されているか、テスト対象デバイス（device-under-test）の設計及び実装の関係者と独立して働く個人によって独立に設計され実装されていれば、このテストはより信用できる結果をもたらす。これは、同じグループによって実装された 2 つのデバイスは、互いに矛盾はしないものの（実際には）正しくない相互運用をするかもしれないためである。NIST 暗号アルゴリズム認証プログラム（Cryptographic Algorithm Validation Program：CAVP）では、保証ベースライン（assured-baseline）実装として NIST によって開発された実装を使用して、NIST 承認暗号アルゴリズムの実装の相互運用テストを実行する。

FR:9.3 CKMS が他のシステムとの相互運用性を主張する場合、CKMS 設計は、その主張を検証するために実行し合格したテストを明記しなければならない。

FR:9.4 CKMS が他のシステムとの相互運用性を主張する場合、CKMS 設計は、相互運用性に必要な、あらゆる構成設定（configuration settings）を明記しなければならない。

9.4 自己テスト

デバイスは、設計され、実装され、最初に配置されたときに正しく動作するが、しばらくすると障害が起こるかもしれない。デバイスで障害が検知されればデバイスを修理又は交換することができるが、検知されない障害は重大なセキュリティへの影響をもたらし得る。CKMS は、完全性及びセキュリティ障害に対して自分自身をテストするデバイスを使用すべきである。例えば、[FIPS 140]は、全てのセキュリティ機能を含めて、暗号モジュールの正しい動作の検証を助けるいくつかの自己テストを定義している。

FR:9.5 CKMS 設計は、設計者によって作成及び実装された全ての自己テスト、及びそれが正しい動作を検証する対象の CKMS 機能を明記しなければならない。

9.5 スケーラビリティテスト

スケーラビリティとは、プロセスが増大する負荷に応じて円滑に正しく処理するためのシステム、ネットワーク又はプロセスの能力、あるいはそのようなプロセスの増大分を収容するために拡張できる能力のことである。スケーラビリティテストでは、与えられた時間内で処理するトランザクション数又は取り扱う参加者数が劇的に増加したときにデバイス又はシステムがどのように反応するかを見極めるために、デバイス又はシステムをテストすることを含む。それぞれのデバイスは限界があるが、あるデバイスの設計は他のデバイスよりスケーラビリティがよいことがある。システムがモジュール式のスケーラビリティを持つように設計されていなければ、追加のデバイスを組み入れることは現実的でないかもしれない。さらに、単に装置（equipment）を追加購入することでは解決できないような微妙な問題が発生することも多い。スケーラビリティテストは、完全に運用される前にこれらの問題を認識して軽減するために、デバイス及びシステムにストレスを与えるために使用される。

FR:9.6 CKMS 設計は、今までにシステムで実行された全てのスケーラビリティ分析及びテストを明記しなければならない。

9.6 機能テスト及びセキュリティテスト

これまでに記載したテストのタイプは、特定のテスト目的を満たすように実行できる。機能テストは、ある機能の実装が正しく動作することを検証するために試行する。機能テストでは、暗号アルゴリズムの実装が与えられた鍵を使用して平文から暗号文を正しく計算すると判定することがある。セキュリティテストは、実装がセキュアに機能することを検証するために試行する。セキュリティテストでは、暗号アルゴリズム実装が正しく機能する（すなわち、正しい結果を出力する）一方で、暗号処理中の電力消費の変動が鍵の危殆化につながり得ると判定することがある。つまり、暗号アルゴリズム実装が、機能テストには合格するが、セキュリティテストには不合格になることがある。

ペネトレーションテストは特別な種類のセキュリティテストである；それは、ペネトレーションテストのエキスパートのチームが、全体としてのシステムに対するペネトレーションシナリオを開発して、ペネトレーション成功のリスクを評価する。個々の製品／デバイスのペネトレーションテストが、CKMS セキュリティアセスメント（11 章を参照）の一部として実行されることがあることに留意されたい。ペネトレーションテストのスコープには、人的、設備及び手続きを含むべきである。ペネトレーションテストチームは、CKMS セキュリティを破ることを目的としてセキュリティ防御を回避することを試みる。ペネトレーションテストチームによるあらゆる発見に対して、最初の配備の前に対処すべきである。

FR:9.7 CKMS 設計は、システムで実行された機能テスト及びセキュリティテスト、並びにそのテスト結果を明記しなければならない。

9.7 環境テスト

CKMS 設計は、デバイス又はシステムに対する特定の環境（例えば、温度範囲及び電圧範囲）を仮定することが多い。そこで、CKMS デバイス又は CKMS システムはその環境用に構築され、その環境の範囲内でテストされる。CKMS デバイス又は CKMS システムが異なる環境で使用されると、セキュアな運用が失われる可能性がある。軍事システムは、それが使用される可能性がある極端な条件に対応するために高耐久化されることが多い。このような追加の保護は追加コストをもたらすことが多い。

FR:9.8 CKMS 設計は、CKMS が使用される設計上の環境条件を明記しなければならない。

FR:9.9 CKMS 設計は、CKMS デバイスで実行された環境テストの結果を、設計上の条件を超えたストレスをデバイスに与えた時の全てのテストの結果も含めて、明記しなければならない。

9.8 開発、配送、及びメンテナンス保証（Maintenance Assurances）

CKMS 製品のセキュアな開発、配付、及びメンテナンスは、CKMS のセキュリティにおいて重要な役割を果たし得る。以下の領域が考慮されるべきである：

a) 構成管理

b) セキュアな配送

c) 開発環境及びメンテナンス環境でのセキュリティ

d) 欠陥修正

これらの領域はそれぞれ以下の小節に記載される。

9.8.1 構成管理

CKMS は、製品への認可されていない又は意図しない変更によってセキュリティが低下せず、かつ機能的欠陥が取り込まれることがないことを保証するために、適切な構成管理システムの下で開発され維持されている製品を組み込むべきである。

FR:9.10 CKMS 設計は、以下を明記しなければならない：

a) 構成制御の下に置かれているデバイス（ソースコード、スクリプト実装、実行コード、ファームウェア、ハードウェア、文書、及びテストコードを含む）

b) 構成制御の下でコンポーネント及びデバイスへの認可された変更だけが行われたことを保証するための保護要求事項（例えば、正規の認可及び適切な記録保持）

9.8.2 セキュアな配送

CKMS で使用される製品が配送されるとき、セキュアな配送の保証（すなわち、受領した製品が間違いなく注文した製品である）が必要である。

FR:9.11 CKMS 設計は、以下を含む、CKMS で使用される製品のセキュアな配送の要求事項を明記しなければならない：

a) 配送プロセス中に製品が細工されていない、又は細工されたことが検知されることを保証するための保護要求事項

b) 配送プロセス中に製品が交換されていない、又は交換されたことが検知されることを保証するための保護要求事項

c) 要求されていない配送が検知されることを保証するための保護要求事項

d) 製品の配送が差し止め又は遅延していない、及び差し止め又は遅延が検知されることを保証するための保護要求事項

9.8.3 開発環境及びメンテナンス環境におけるセキュリティ

CKMS 開発環境及びメンテナンス環境は、物理的、人的、及び IT ハッキングの脅威から適切に保護されなければならない。コンパイラ、ソフトウェアリンカ及びテキストエディタのようなツールは自動的に信頼すべきではない。

FR:9.12 CKMS 設計は、以下を含む、CKMS の開発環境及びメンテナンス環境におけるセキュリティ要求事項を明記しなければならない：

a) 物理セキュリティ要求事項

b) 開発者、試験者、及び保守員に対する身分照会及びバックグラウンドチェックのような人的セキュリティ要求事項

c) 複数人員（multi-person）による制御、及び職掌分散（separation of duties）のような手続き的セキュリティ

d) 開発環境及びメンテナンス環境の保護、及び認可されたユーザにアクセスを許可するアクセスコントロールの提供のためのコンピュータセキュリティコントロール

e) ハッキングの試みから開発環境及びメンテナンス環境を保護するためのネットワークセキュリティコントロール

f) 開発下のソフトウェア及びその制御データの完全性を保護するための暗号学的セキュリティコントロール

g) ツール（例えば、エディタ、コンパイラ、ソフトウェアリンカ、ローダ等）が信頼でき、マルウェアのソースでないことを保証するために利用する手段

9.8.4 欠陥修正能力

CKMS は、迅速かつセキュアな方法で欠陥を検知、報告及び修正する能力を持つべきである。自動化された技術を備えた CKMS が非常に望ましい。なぜなら、継続的に自身のセキュリティステータスを監視し、潜在的な問題を適切な CKMS の役割を持つ認可された要員に報告し、かつ稀にしか起こらないイベントへの人間による監視の依存性を最小化することができるためである。

FR:9.13 CKMS 設計は、以下を含む、システムの欠陥を検知する CKMS の能力を明記しなければならない：

a) 既知解テスト

b) エラー訂正コード

c) 異常故障診断技術

d) 機能テスト

FR:9.14 CKMS 設計は、以下を含む、欠陥を報告する CKMS の能力を明記しなければならない：ステータスレポートメッセージを機密性、完全性、及びソース認証保護付きで作成する能力、及び認可されない遅延を検知する能力。

FR:9.15 CKMS 設計は、欠陥を分析し、かつ起こりやすい又はよく知られている欠陥に対する修正を作成／取得する CKMS の能力を明記しなければならない。

FR:9.16 CKMS 設計は、機密性、完全性、及びソース認証保護付きで修正を送信し、かつ認可されない遅延を検知する CKMS の能力を明記しなければならない。

FR:9.17 CKMS 設計は、時宜を得て修正を実装する CKMS の能力を明記しなければならない。

10 災害復旧

情報を保護するために使用される暗号鍵及びメタデータを管理するための CKMS の使用は、CKMS の障害がそれら情報へのアクセスの妨害又は停止につながるかもしれないという追加のリスクを伴う。例えば、復号機能の障害は暗号化されたデータの使用を遅延又は停止させるかもしれない。本節は、コンポーネント障害や鍵及びメタデータの破損の事象発生時にどのように運用継続性を達成できるかを記載する。

10.1 設備への損害

CKMS は、物理的にセキュアで環境的に保護された施設に配置されるべきである。さらに、CKMS 管理は、CKMS への損害発生時におけるバックアップ及び回復のために備えるべきである。バックアップ及び回復設備は、保護されるデータ及び運用の価値と機微度にふさわしいレベルで設計、実装及び運用されるべきである。CKMS 設備が損害を受けたとき、運用はバックアップ設備に移転すべきであり、偶発的に開示されたかもしれない鍵は、適宜、直ちに危殆化鍵リスト又は証明書失効リストに記載の上、置き換えるべきである。風水害はどちらも環境リスクであり、火災は環境リスク及び設備設計に依存したリスクの両方である。

FR:10.1 CKMS 設計は、必要な環境的、火災、及び物理的なアクセスコントロール保護メカニズム、及び損害からの基幹及び全てのバックアップ設備への回復手続きを明記しなければならない。

10.2 公共サービス（Utility Service）の停止

CKMS は、CKMS の継続的な可用性を保証するため、電気、水道、下水道、空調、熱源、及び清浄な空気を含む信頼できる公共サービスを必要とする。通常運用時及び非常時において、人間に安全及び快適を提供するのと同様に、全ての電子デバイスの要求を満たすのに十分な電力が基幹及び全てのバックアップ CKMS 設備で利用可能であるべきである。

FR:10.2 CKMS 設計は、基幹及び全てのバックアップ設備に対する、電気、水道、衛生、暖房、冷房、及び空気清浄に関する推奨要求値だけでなく最小要求値についても明記しなければならない。

10.3 通信及び計算機能（Computaiton）の停止

CKMS は、必要な機能を実行しユーザが要求するサービスを提供するために、十分な通信及び計算能力を必要とする。長距離通信能力は典型的には商用ベンダによって提供され、多くのコンピュータベンダが大規模な鍵管理アプリケーションに十分なコンピュータを供給できる。高可用性を保証するために、CKMS に冗長な通信設備が設置されることも多い。遠隔オンラインバックアップ設備は、特に潜在的な環境リスク（例えば、天候）に対して、さらなる高いサービス可用性を提供するために使用することができる。代替通信サービスに素早くアクセスする能力は、通信サービスの障害発生時に大いに望ましい。

FR:10.3 CKMS 設計は、ユーザ、エンタープライズ、及び CKMS アプリケーションによる予測されるニーズに見合うサービスの、運用継続を保証するために、設計内に存在し、かつ運用中に利用可能であることを要求される通信及び計算機能の冗長性を明記しなければならない。

10.4 システムハードウェア障害

CKMS は、サポートする情報管理システムのセキュアな運用にとって極めて重要であるため、CKMS コンポーネント及びデバイスのハードウェア障害の影響を最小限に抑えることが望ましい。ハードウェア障害から回復するためのいくつかの方法が存在する。これらの方法には、回復の容易さ及びスピードと、コストとの間でトレードオフが存在する傾向がある。バックアップシステムの冗長性は、単一のハードウェア障害の運用への影響が速やかに検知され、完全に機能的にセキュアな状態が速やかに得られることの保証を提供できる。バックアップシステムを最も効果的にするため、基幹システムとの同期を保持すべきである。基幹システムとの同期を継続的に保持しているバックアップシステムは「ホット」バックアップと呼ばれる。これらのシステムは基幹システムの責任を即座に引き継ぐ能力がある。あるバックアップシステムは、定期的に基幹システムと同期を行い、そのバックアップシステムを基幹システムが故障発生直前になっていた状態にするキャッチアップ手続きを有している。

基幹システムの故障がバックアップシステムにも同じ故障を引き起こすことがないようにするため、可能な限り、バックアップシステムが基幹システムからの独立性を持っていることが不可欠である。例えば、共通の電力線での急激な電圧変動は、基幹システムとバックアップシステムの両方の故障を引き起こすかもしれない。独立性を最大化するため、バックアップシステムを基幹システムと同じ場所に配置しないことが最善である。複数のバックアップシステムは、エラー検知能力を提供するために使用することができる。例えば、3 つのシステム全てが同じ機能を実行している場合、3 つのシステムの多数決を正しい結果として採用することで、どれかひとつのシステムの故障を検知でき、かつ訂正することができる。冗長性はサービス提供コストを増大させるため、システムベンダ及び CKMS 所有者は、冗長性とコストとの間の最適なトレードオフを見出すことを目指す。

FR:10.4 CKMS 設計は、バックアップの方策、及びハードウェアコンポーネント及びデバイスの障害からの回復のための方策を明記しなければならない。

10.5 システムソフトウェア障害

ソフトウェア障害には、典型的には以下の 2 つのタイプがある：

a. ソフトウェアが正しく書かれておらず、特定の条件が成立した時に望むように動かない

b. ソフトウェアは正しく書かれているが、実行する前にメモリ上に展開されている時に破損した

多くのソフトウェア障害は、良好な確立されたプログラミング実践を使用してコードを書くことで防ぐことができる。良好なプログラミング手続き、境界条件の扱い方法、メモリオーバーフローに対する保護方法、コード解析手法、及び正しいソフトウェアテスト技法の利用に関する本トピックについて、いくつかの書籍が書かれている。

コードを破損する障害は可能な限り速やかに検知されるべきである。これは、ソフトウェアを実行する前に、ソフトウェアに対するエラー訂正コードの検証又は既知解テストで達成できるかもしれない。エラーが検出された場合、プログラムはエラー状態に入りエラーインジケータを出力するように実装することができる。これは、コードが運用に使われる前に、エラーを検知し修復することを可能にする。これらのテストは、要望に応じて、定期的に実行することもできる。10.4 節で上述した、冗長システムもまたこの種の障害から回復するために使用することができる。

基幹設備の CKMS が既知のセキュア状態に回復したとき、最後のセキュア状態の時点以降に生成されたいくつかのデータは失われているかもしれない。CKMS は、いずれは大惨事が起こるとの仮定の下で実装し運用すべきである。したがって、完全なセキュア状態のシステムバックアップが定期的に作成され、最新の CKMS のセキュア状態がリロードされて CKMS が修復され運用可能な状態にできるようにすることが推奨される。

FR:10.5 CKMS 設計は、システムソフトウェアの正しさを検証するために、CKMS によって提供されている全ての技術を明記しなければならない。

FR:10.6 CKMS 設計は、ソフトウェアがメモリにロードされたときにソフトウェアの改変又は破損を検知するために、CKMS によって提供される全ての技術を明記しなければならない。

FR:10.7 CKMS 設計は、バックアップ及び重大なソフトウェア障害からの回復のための方策を明記しなければならない。

10.6 暗号モジュール障害

暗号モジュールは、ハードウェア、ソフトウェア又はファームウェアの障害を検知するために適切な、組み込まれたテスト機能を備えるべきである。暗号モジュールは、動作前自己テスト、条件付自己テスト、及び定期的な自己テストの機能を備えているかもしれない。例えば、障害が FIPS140-2 認証モジュールで検知されたとき、エラーインジケータが出力され、そのエラーが回復不可能なタイプ（すなわち、アフターサービス、修理又は交換が必要な障害）か回復可能なタイプ（すなわち、初期化又はリセットが必要な障害）かのどちらかを決定するエラー状態にコントロールが移行する。ほとんどのケースでは、暗号モジュールがエラー状態にある間は、機微なデータが暗号モジュールから出力されるべきではない。エラーが回復可能なものであるならば、暗号モジュールは再起動され通常処理を続行する前に全ての起動自己テストに合格するべきである。再起動を繰り返し試みてもエラー状態が再現する場合、そのモジュールを交換するべきである。

FR:10.8 CKMS 設計は、モジュールのエラー検知及び完全性検証のために、それぞれの暗号モジュールがどの自己テストを使用するかを明記しなければならない。

FR:10.9 CKMS 設計は、それぞれの暗号モジュールがどのように検知したエラーに応答するかを明記しなければならない。

FR:10.10 CKMS 設計は、障害が起こった暗号モジュールの修理又は交換の方策を明記しなければならない。

10.7 鍵及びメタデータの破損

暗号鍵及びメタデータは、伝送中又はストレージ内において破損する可能性がある。破損した鍵及びメタデータは、破損が検知されたときには可能な限り速やかに置き換え又は訂正すべきである。破損した鍵及びメタデータの置き換えは、典型的には新しい鍵及びメタデータの確立又は保管を伴う。破損した鍵又は破損したメタデータが付随した鍵がデータを保護するために使用されていた場合、機微なデータの喪失又は危殆化が結果として起こる可能性があるため、セキュリティに及ぼす余波を評価すべきである。鍵確立及び鍵保管プロトコルは、破損した鍵を検知し置き換えるために設計されることが多い。

重大な災害は、基幹ストレージによる回復能力を超えた、多数の運用中の鍵及びメタデータの喪失又は破損を引き起こす可能性がある。鍵復元、バックアップ、又はアーカイブシステムが存在する場合、鍵及びメタデータは復元可能であり、復元すべきである。しかしながら、鍵がバックアップもアーカイブもされていなかった場合、新しい鍵で置き換えなければならず、元の鍵で保護された情報は失われるかもしれない。

FR:10.11 CKMS 設計は、暗号鍵及びそのメタデータをバックアップ及びアーカイブするための手続きを明記しなければならない。

FR:10.12 CKMS 設計は、保管又は伝送された破損した鍵及びメタデータを復元又は置き換えるための手続きを明記しなければならない。

11 セキュリティアセスメント

CKMS セキュリティは、CKMS のライフタイムを通して、任意の時に評価されることがある。本章は、最初の配備時、定期的（例えば年ごと）なレビュー時、及び大きな変更後の追加的なアセスメント時における評価での考慮事項に焦点を当てる。米国政府セキュリティアセスメント実践についての追加の情報については、[SP 800-37]、[SP 800-53]、[SP 800-53A]、及び［SP 800-115］を参照されたい。

11.1 完全セキュリティアセスメント

CKMS の配備前又は配備時に、CKMS のセキュリティは完全に評価されるべきである。CKMS のセキュリティアセスメントに課すことができる実行策には以下のものが含まれる：

a) 第三者検証のレビュー

b) システム設計のアーキテクチャレビュー

c) CKMS の機能テスト及びセキュリティテスト

d) CKMS のペネトレーションテスト

これらの実行策はそれぞれ以下の小節で記載される。

FR:11.1 CKMS 設計は、完全な CKMS セキュリティアセスメントの前又は同時に行われる、必要な保証実行策を明記しなければならない。

FR:11.2 CKMS 設計は、完全なセキュリティアセスメントが繰り返される状況を明記しなければならない。

11.1.1 第三者認証のレビュー

現在のところ CKMS のセキュリティのための正式な認証プログラムは存在しないが、ある種の CKMS デバイス用には下記の認証プログラムが存在する：

a) NIST 承認暗号アルゴリズムの実装は、NIST 暗号アルゴリズム認証プログラム（Cryptographic Algorithm Validation Program：CAVP）の下で認証することができる

b) 暗号モジュールは、FIPS 140-2 への適合性を NIST 暗号モジュール認証プログラム（Cryptographic Module Validation Program：CMVP）の下で認証することができる

c) 非暗号セキュリティ及びハードウェア（例えば、OS、DBMS、又はファイアウォール）は、コモンクライテリア標準（[ISO/IEC 15408 Parts 1-3］を参照）を使用して National Information Assurance Partnership（NIAP）の下で認証できる

d) CKMS、又はその一部は、ベンダ又はスポンサーに雇われた民間のエンティティによって検証することもできるかもしれない

これらの認証プログラムはセキュリティを保証しないが、CKMS のセキュリティ及び完全性に関する信頼性を著しく向上させることができる。

FR:11.3 CKMS 設計は、あらゆる CKMS デバイスについて、認証を受けた全ての認証プログラムを明記しなければならない。

FR:11.4 CKMS 設計は、認証済みデバイスに対する全ての認証番号を明記しなければならない。

11.1.2 システム設計のアーキテクチャレビュー

本実行策の下で、CKMS アーキテクチャの評価のためにエキスパートのチームが召集される。アーキテクチャレビューチームは、CKMS 設計情報、第三者検証情報、及び全ての利用可能な CKMS テスト結果にアクセスできるべきである。アーキテクチャレビューチームから提供される推奨事項は設計者によってレビューされるべきであり、推奨事項のうち CKMS に取り込むことが選択されたものは文書化され、実装される設計変更に反映されるべきである。アーキテクチャレビューチームは、CKMS 管理者がレビューするペネトレーションテストシナリオへの推奨事項も作成すべきである。実行するように選択されたペネトレーションテストは、CKMS 管理者の指示に従って適切な時期に実行すべきである。

アーキテクチャレビューチームは、暗号、暗号プロトコル、セキュアシステム設計、ネットワークセキュリティ、コンピュータセキュリティ、ヒューマンユーザビリティ／アクセシビリティ、セーフティ機能、分散システム設計、及び適用される情報システムに関する法律と規則についての専門知識を備えているべきである。

FR:11.5 CKMS 設計は、完全なセキュリティアセスメントの一部として、アーキテクチャレビューを必要とするかどうかを明記しなければならない。

FR:11.6 アーキテクチャレビューが必要である場合、CKMS 設計は、アーキテクチャレビューチームに必要なスキルセットを明記しなければならない。

11.1.3 機能テスト及びセキュリティテスト

機能テスト及びセキュリティテストは、典型的には完全なセキュリティアセスメント、定期的なセキュリティレビュー、及び追加のセキュリティアセスメントの一部として実行される。様々な機能テスト及びセキュリティテストが、ベンダ、情報所有者（information owner)、又は信頼される第三者によって実行されることがある（9 章を参照)。

FR:11.7 CKMS 設計は、必要な全ての CKMS の機能テスト及びセキュリティテストを明記しなければならない。

FR:11.8 CKMS 設計は、今までに実行された全ての機能テスト及びセキュリティテストの結果を報告しなければならない。

11.1.4 ペネトレーションテスト

ペネトレーションテストでは、CKMS のセキュリティを危殆化させるアクティブな試みへの抵抗度合いを検証するために CKMS をテストする。この種類のテストは、典型的な攻撃技術及びシステムの弱点についての知識があり、かつ新しい又は予想外の攻撃手法を開発し試行する能力も持つセキュリティエキスパートを必要とする。攻撃グループには、CKMS 設計チームの一員ではない人、及びその

CKMS のセキュリティに関する先入観を持たない人を複数含むべきである。成功する攻撃手法では、システムを意図しない又は予想しない方法で使用することを含むことが多い。

FR:11.9 CKMS 設計は、今までに実行されたあらゆる完了したペネトレーションテストの結果を明記しなければならない。

11.2 定期的セキュリティレビュー

本レビューは、システムコントロール、物理コントロール、手続き的コントロール及び人間によるコントロールが主張どおりに整備され運用していることを保証するために、それらのコントロールを検査することから構成される。製品／デバイスがセキュアな設定で最新のアップデート及びセキュリティパッチを適用して運用され、その製品の第三者によるセキュリティレーティングが維持され続けていることを保証するために、前回のセキュリティレビューからのシステム変更箇所を検査すべきである。レビューの結果として特定された問題点には対応すべきである。加えて、定期的な機能テスト及びセキュリティテストを実行すべきである（9.6 節を参照）。

FR:11.10 CKMS 設計は、セキュリティレビューの周期を明記しなければならない。

FR:11.11 CKMS 設計は、CKMS デバイスの観点から、セキュリティレビューのスコープを明記しなければならない。

FR:11.12 CKMS 設計は、レビュー対象のそれぞれの CKMS デバイスに対して行われる実行策に関して、定期的なセキュリティレビューのスコープを明記しなければならない。

FR:11.13 CKMS 設計は、定期的なセキュリティレビューの一部として実行される機能テスト及びセキュリティテストを明記しなければならない。

11.3 セキュリティアセスメントの追加

CKMS システムに著しい変更が加えられたとき、11.1 節に記載された以下の範囲での変更箇所への追加のセキュリティアセスメントを実行すべきである：

a) 前回のセキュリティアセスメント以降の第三者認証されたデバイスへの変更

b) システム設計変更後のアーキテクチャレビュー

c) CKMS の機能テスト及びセキュリティテスト

累積的な CKMS のシステム変更が著しい場合、11.1 節に記載された完全な CKMS セキュリティアセスメントを実施すべきである。

FR:11.14 CKMS 設計は、追加のセキュリティアセスメントが実施されるべき状況を明記しなければならない。

FR:11.15 CKMS 設計は、追加のセキュリティアセスメントのスコープを明記しなければならない。

11.4 セキュリティメンテナンス

特定の保護レベル（例えば、低、中、高）を提供するために CKMS が設計され、開発され、配備された場合でも、設定が変更されたり新しい脅威が発見されたりすることで、提供される保護レベルが低下することがある。CKMS システムのセキュリティを維持し強化するために、CKMS は適切に更新され、堅牢化ガイドラインに従ってレビューされテストされるべきである。堅牢化の実行策の例に

は、CKMS を最新のセキュリティパッチで更新すること、堅牢化ガイドラインに従ってシステム設定を定期的にレビューすること、堅牢化ガイドラインに従って CKMS を定期的にテストすること、更新された堅牢化ガイドラインを適用すること、及び定期的なペネトレーションテストを行うことが含まれる。

FR:11.16 CKMS 設計は、セキュリティを維持するために、実行することが必要な堅牢化アクティビティをリスト化しなければならない。

12 技術的課題

CKMS は、長期にわたるセキュリティライフタイムを持つように設計され実装されるべきである。したがって、設計者は、CKMS がセキュアでなくなるかもしれない技術の進歩に起因する潜在的な脅威を考慮すべきである。以下にいくつかの例を挙げる。

a) 暗号アルゴリズムに対する新しい攻撃

暗号アルゴリズムには想定されるセキュリティ寿命がある。しかしながら、時間が経過するにつれ、そのセキュリティ寿命を短縮する新しい攻撃が発見されるかもしれない。CKMS のセキュリティは鍵及びメタデータを保護する暗号アルゴリズムに依存しており、暗号アルゴリズムに対する新しい攻撃の発見は CKMS のセキュリティ寿命を縮める可能性が高い。最終的には、暗号アルゴリズムを完全に更新又は置き換える必要があるかもしれない。

暗号アルゴリズムは、（当該アルゴリズム以外の）残りの実装への著しい影響なしで置き換え又は更新ができるような方法で、暗号モジュール内に実装すべきである。例えば、鍵長及びブロック長は、必要になった時に容易に延ばせるように可変にすべきである。

b) 鍵確立プロトコルに対する新しい攻撃

プロトコルが数年間使用された後に弱点が発見されることも少なくない。プロトコルは、暗号アルゴリズムに対して行われるのと同じ程度の評価がされることはめったになく、一旦広く使用されるようになるとプロトコルをアップグレードすることは困難であることが多い。したがって、評価されテストされた鍵確立用のプロトコルを CKMS が使用することは重要である。

c) CKMS デバイスに対する新しい攻撃

コンピュータベースのシステムへ論理的な攻撃を行ったり打ち破ったりするための新しい手段が発見され続けている。CKMS 設計者は、認可されない当事者が外部から CKMS デバイスへアクセスすることを、現実的な範囲で最大限防止すべきである。CKMS のセキュリティが依存するアクセスコントロールメカニズムは、要求に応じて、最新の攻撃を実行したりアップグレードしたりして定期的にレビューされるべきである。

d) 新しい計算機技術

新しい計算機技術が CKMS のセキュリティを脅かすかもしれない。現状の脅威で最も高い関心が払われているものは、暗号鍵を復元するのに十分な能力を持つ量子コンピュータの発展である。実用的な量子コンピュータの実装は、暗号学的セキュリティ技術に重大な変化をもたらすかもしれない。例えば、大きなキュービットの量子コンピュータが構築されれば、素因数分解型及び離散対数ベース型の公開鍵暗号アルゴリズムのセキュリティが脅かされるかもしれない。このことは、これらのアルゴリズムに暗号鍵の確立を依存する CKMS に対して重大な影響を与えるかもしれない。量子コンピュータに耐性がある公開鍵暗号アルゴリズム（例えば、格子ベース型公開鍵暗号）を発明する研究は現在進行中であるが、広く受け入れられている解はまだ見出されていない。また、現時点で量子コンピュータによる攻撃により耐性があると考えられている対称鍵暗号アルゴリズム（例えば、AES-256）を使用できるような、スケーラブルな対称鍵配付アーキテクチャを発明する研究も現在進行中である。

FR:12.1 CKMS 設計は、システムに実装されたそれぞれの暗号アルゴリズムの想定されるセキュリティライフタイムを明記しなければならない。

FR:12.2 CKMS 設計は、CKMS の運用に悪影響を与えることなしに、暗号アルゴリズムのどの副関数（例えば、HMAC の副のハッシュ関数）が、類似だが暗号学的に改良されている副関数にアップグレード又は置き換えを行うことができるかを明記しなければならない。

FR:12.3 CKMS 設計は、どの鍵確立プロトコルがシステムによって実装されているかを明記しなければならない。

FR:12.4 CKMS 設計は、採用されている暗号アルゴリズムの想定されるセキュリティライフタイムの観点から、システムに実装されているそれぞれの鍵確立プロトコルの想定されるセキュリティライフタイムを明記しなければならない。

FR:12.5 CKMS 設計は、CKMS デバイスへの外部からのアクセスが許容されている範囲を明記しなければならない。

FR:12.6 CKMS 設計は、CKMS デバイスへの全ての許可された外部アクセスがどのようにコントロールされるかを明記しなければならない。

FR:12.7 CKMS 設計は、量子コンピュータによる CKMS の暗号アルゴリズムに対する攻撃のような、新しい技術の発展がもたらす影響に、抵抗又は軽減するために採用している機能を明記しなければならない。

FR:12.8 CKMS 設計は、CKMS の暗号に対する量子コンピュータによる攻撃の、現在知られている影響を明記しなければならない。

附属書 A：参考文献

読者が、その文献がニーズに合うかどうかをすぐに見極められるように、各文献に短い要約を付している。

1. [FIPS 140]

   FIPS 140-2：暗号モジュールのセキュリティ要求事項、2001 年 5 月、

   www.csrc.nist.gov/publications/PubsFIPS.html

   FIPS 140-2 は、米国政府の情報を保護する暗号モジュール（以下、モジュール）が満たされなければならない 11 の分野の要求事項を規定している。この要求事項は、ハードウェア、ソフトウェア、ファームウェア、およびハイブリッドモジュールに適用される。この標準は、潜在的な応用分野および広範な動作環境を網羅するために、4 つのセキュリティレベルを提供している。セキュリティ要求事項は、暗号モジュールの安全な設計と実装に関連する分野もカバーしている。これらの領域には、暗号モジュール仕様、すなわち、暗号モジュールのポートとインタフェース、 役割、サービス、および認証メカニズムと有限状態モデル、モジュールの物理的なセキュリティ、運用環境、暗号鍵管理、電磁干渉/電磁両立性（EMI / EMC）、自己テスト、設計保証および他の攻撃への対処などが含まれる。この規格への準拠は、NIST 暗号モジュール試験及び認証プログラム（CMVP）で検証される。

2. [FIPS 180]

   FIPS 180-4：Secure Hash Standard（SHS）、2012 年 3 月、

   www.csrc.nist.gov/publications/PubsFIPS.html

   FIPS 180-4 は、メッセージのダイジェストを生成するために使用できるハッシュアルゴリズムを規定している。ダイジェストは、ダイジェストが生成された後に、対象となるメッセージが変更されたかどうかを検出するために使用されうる。ダイジェストは、デジタル署名の生成と検証、乱数の生成、メッセージ認証コードの生成に使用されうる。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

3. [FIPS 186]

   FIPS 186-4：デジタル署名標準（DSS）、2013 年 7 月、

   www.csrc.nist.gov/publications/PubsFIPS.html

   FIPS 186-4 は、デジタル署名が必要な応用分野での（署名）アルゴリズムを規定する。この規格では、DSA、RSA、および ECDSA 署名技術と、FIPS 180-4 の適切なハッシュ関数を使用してデジタル署名を生成する（計算する）。FIPS 186-4 には、鍵（署名鍵）とドメイン・パラメタの生成に関する要求事項とアルゴリズムも含まれている。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

4. [FIPS 197]

   FIPS 197：Advanced Encryption Standard（AES）、2001 年 11 月、

   www.csrc.nist.gov/publications/PubsFIPS.html

FIPS 197 は、対称鍵ブロック暗号の暗号化/復号アルゴリズムを規定している。この規格は 128、192、および 256 ビットの鍵長と 128 ビットのブロック長をサポートしている。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

5. [FIPS 198]

FIPS 198-1：鍵付きハッシュメッセージ認証コード（HMAC）、2008 年 7 月、

www.csrc.nist.gov/publications/PubsFIPS.html

FIPS 198-1 には、（暗号学的）ハッシュ関数を使用したメッセージ認証方式である、鍵付き付きハッシュ関数ベース・メッセージ認証コード（HMAC）が記規定される。HMAC は、共有鍵と組み合わせて、NIST 承認の暗号ハッシュ関数として使用される。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

6. [IPSEC]

IPSEC に係わる RFC（Request for Comment）文書

http://www.ietf.org/dyn/wg/charter/ipsecme-charter.html

これらの IPSEC に係わる RFC は、認証、暗号化、および完全性というセキュリティサービスが IP パケットに対してどのように提供されるかを記述する。RFC は、パケットのセキュリティサービスペイロードの形式、セキュリティサービスの暗号スイート、およびセキュリティサービスを提供するために使用される暗号アルゴリズムの鍵管理手法をカバーしている。

7. [ISO / IEC 15408 のパート 1-3]

ISO/IEC 15408[23] *情報技術 - セキュリティ技術 - IT セキュリティの評価基準*、

パート 1：はじめにと一般的なモデル

パート 2：セキュリティ機能要求事項

パート 3：セキュリティ保証コンポーネント

http://www.iso.org/iso/catalogue

https://www.commoncriteriaportal.org/

ISO/IEC 15408-1:2009 は、IT セキュリティ評価の一般的な概念と原則を確立し、IT 製品のセキュリティ属性の評価の基礎として使用される ISO/IEC 15408,のさまざまな部分によって与えられる評価の一般的なモデルを規定している。

ISO/IEC 15408-2:2005 は、セキュリティ評価のためにセキュリティ機能コンポーネントの必要な構造と内容を定義しる。これには、多くの IT 製品およびシステムの共通のセキュリティ機能要件を満たす機能コンポーネントのカタログが含まれている。

ISO/IEC 15408-3:2008 は、評価基準の保証要件を定義している。これには、評価対象コンポーネント（TOE）の保証を測定するためのスケールを定義する評価保証レベル、統合 TOE の保証を測定するためのスケールを定義する合成保証パッケージ、保証レベルとパッケージを構成する個々の

[23] 【訳注】ISO/IEC 15408 は、Ver.2.3 であり、CCRA が公開している最新版は Ver.3.1Release5 である。いずれの版も日本語版は、https://www.ipa.go.jp/security/jisec/cc/index.html から入手できる。

保証コンポーネント、および、保護プロファイルとセキュリティターゲットの評価基準が含まれている。

8. [Kerberos（ケルベロス）]

Kerberos に係わる RFC

http://www.ietf.org/dyn/wg/charter/krb-wg-charter.html

Kerberos に係わる RFC は、Kerberos 認証、暗号化、および安全なチケット付与（Ticket-granting）サービスの提供方法が規定されている。RFC は、セキュリティサービスのペイロードの形式、セキュリティサービスの暗号スイート、およびパスワードまたは X.509 証明書を使用した認証を含んでいる。

9. [RFC 6960]

オンライン証明書ステータスプロトコル（OCSP）

http://www.ietf.org/rfc/rfc6960.txt

RFC 6960 は、公開鍵証明書の現在の状態を問い合わせるために使用されるプロトコルを規定している。

10. [RFC 3161]

X.509 公開鍵基盤タイムスタンププロトコル（TSP）

http://www.ietf.org/rfc/rfc3161.txt

RFC 3161 は、信頼できる第三者に信頼できるタイムスタンプを要求し、受信するためのプロトコルを規定している。この文書は、デジタル署名ベースのタイムスタンプトークンの構造を規定し、データの存在に関連する時間を提供するために使用される。

11. [RFC 3279]

インターネット X.509 PKI のためのアルゴリズムと識別子

http://www.ietf.org/rfc/rfc3279.txt

RFC 3279 は、RSA、DSA、DH、および EC アルゴリズムのサブジェクト公開鍵情報を格納するための OID および構造を指定している。 この RFC では、ハッシュおよび署名アルゴリズムのオブジェクト識別子と署名構造も定義されている。 RFC 3279 は、追加の署名アルゴリズムおよびスキームに対応するために、RFC 4055 および RFC 5480 によって拡充されている。

12. [RFC 3647[24]]

インターネット X.509 PKI： 証明書ポリシーと認証実施フレームワーク

http://www.ietf.org/rfc/rfc3647.txt

RFC 3647 には、証明書ポリシーまたは証明書運用規定で対処が必要となる可能性がある項目の包括的なリストが記載されている。

---

[24] 【訳注】RFC 3647 の日本語訳は、https://www.ipa.go.jp/security/rfc/RFC.html で入手できる。

13. [RFC 4055]

インターネット X.509 PKI 証明書と CRL 用 RSA 暗号技術についての追加的アルゴリズムおよび識別子,

http://www.ietf.org/rfc/rfc4055.txt

RFC 4055 は、RSA-PSS 署名アルゴリズムと RSA-OAEP 鍵配送アルゴリズムを利用するための規則を記述し、RFC 3279 を補完している。

14. [RFC 4251]

SSH プロトコルアーキテクチャ,

http://www.ietf.org/rfc/rfc4251.txt

RFC 4251 は、安全でないネットワーク上での安全なリモートログインやその他の安全なネットワークサービスのためのプロトコルを規定している。このドキュメントでは、SSH プロトコルのアーキテクチャ、ならびに SSH プロトコルドキュメントで使用される表記法と用語を解説している。また、ローカル拡張を可能にする SSH アルゴリズム命名システムについても解説している。SSH プロトコルは、次の 3 つの主要コンポーネント（プロトコル）から構成されてる。トランスポート層プロトコルは、サーバ認証、機密性、および完全な前方機密性を備えた完全性を提供する。ユーザ認証プロトコルはサーバに対してクライアント認証を提供する。接続プロトコルは暗号化されたトンネルに複数の論理チャネルに多重化する機能を提供する。これらのプロトコルの詳細は別の文書に記載されている。

15. [RFC 6402]

暗号メッセージ構文による証明書管理(CMC),

http://www.ietf.org/rfc/rfc5272.txt

以前は RFC 5272 として公開されていた RC 6402 は、PKCS＃10 または暗号メッセージ構文（CMS）を使用した（証明書）発行要求、鍵再生成、失効などの証明書管理サービスを使用するためのプロトコル標準である。

16. [RFC 5273]

CMS（CMC）による証明書管理：配送プロトコル,

http://www.ietf.org/rfc/rfc5273.txt

RFC 5273 では、CMC（CMS による証明書管理、(Cryptographic Message Syntax)）メッセージの配送に使用されるいくつかの転送メカニズムが定義されている。 定義されている配送メカニズムは、HTTP、ファイル転送、メール配信、および TCP である。

17. [RFC 5274]

CMS（CMC）による証明書管理メッセージ：コンプライアンス要件,
http://www.ietf.org/rfc/rfc5274.txt

RFC 5274 は、CMC（CMS による証明書管理）証明書発行要求プロトコルに関する一連の準拠宣言に関する情報を提供する。

18. [RFC 5280[25]]

インターネット X.509 PKI： 証明書と CRL のプロファイル,

http://www.ietf.org/rfc/rfc5280.txt

RFC 5280 は X.509 公開鍵証明書と対応する CRL のフォーマットを定義している。この RFC では、証明書、証明書パス、および CRL 処理規則も定義されている。

19. [RFC 5295]

拡張マスタ・セッション鍵（EMSK）からルート鍵を導出するための仕様,
http://www.ietf.org/rfc/rfc5295.txt.

拡張認証プロトコル（EAP）は、拡張マスタ・セッション鍵（EMSK）生成プロセスを定義する。RFC 5295 では、EMSK を使用してルート鍵を導出する方法が定義されてる。ルート鍵は、認証、完全性保証、暗号化など、複数のセキュリティサービスに使用できるマスタ鍵である。

20. [RFC 5480]

楕円曲線暗号によるサブジェクトの公開鍵,

http://www.ietf.org/rfc/rfc5480.txt

RFC 5480 は、X.509 証明書の楕円曲線公開鍵の形式と構造を定義している。

21. [RFC 5652]

CMS： 暗号メッセージ構文 (CMS),

http://www.ietf.org/rfc/rfc5652.txt

RFC 5652 には、暗号メッセージ構文（CMS）フォーマットが規定されている。この構文は、任意のメッセージコンテンツをデジタル署名、ダイジェスト作成/検証、認証、または暗号化するために使用される。

22. [RFC 5914]

トラストアンカーフォーマット,

http://www.ietf.org/rfc/rfc5914.txt

RFC 5914 はトラストアンカー情報を表すための構造を規定する。トラストアンカーは、公開鍵と関連データによって表される信頼できるエンティティである。公開鍵はデジタル署名を検証するために使用され、関連付けられたデータはトラストアンカーとして信頼できる情報またはトラストアンカーの目的を制限するために使用される。この文書で定義されている構造は、トラストアンカー管理要件で定義されているフォーマット関連の要件を規定する。

23. [RFC 5934]

トラストアンカー管理プロトコル (TAMP),

http://www.ietf.org/rfc/rfc5934.txt

---

[25] 【訳注】RFC 5280 の日本語訳は、https://www.ipa.go.jp/security/rfc/RFC.html で入手できる。

RFC 5934 は、デバイス、機器、およびアプリケーションが保持するトラストアンカーストアを安全に更新するためのセキュリティプロトコルを定義している。

23. [RFC 6024]

トラストアンカー管理要件,

http://www.ietf.org/rfc/rfc6024.txt

RFC 6024 では、標準的なトラストアンカー管理メカニズムの欠如が招くいくつかの問題について記述し、これらの問題に対処するために設計されたデータフォーマットとプッシュベースのプロトコルの要件を定義する。

24. [SP 800-37]

SP 800-37-rev1: 連邦情報システムにリスク管理フレームワークを適用するためのガイド - セキュリティライフサイクルアプローチ, 2010 年 2 月

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-37-rev1 は、6 段階のリスク管理フレームワーク（RMF）に従来の認証および認定プロセスを導入することによって、統一されたフレームワークへと進化している。改訂されたプロセスは以下を強調している。(i) 最先端の管理、運用上および技術上のセキュリティ管理策の適用を通じて、連邦情報システムに情報セキュリティ機能を組み込む。(ii) 強化された監視プロセスを通じて、継続的に情報システムのセキュリティ状態についての認識を維持する。(iii) 組織運営および資産、個人、他の組織、ならびに情報システムの運用および使用から生じる国家に対するリスクの容認に関する信頼できる決定を促進するために、重要な情報を上級指導者に提供すること。

25. [SP 800-38A]

SP 800-38A: ブロック暗号動作モードの推奨事項 - 方法と技法, 2001 年 12 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-38A は、対称鍵ブロック暗号アルゴリズムと組み合わせて使用する電子コードブック（ECB）モード、暗号ブロック連鎖（CBC）モード、暗号フィードバック（CFB）モード、出力フィードバック（OFB）モード、およびカウンタ（CTR）モード ）の 5 つの機密モードの動作を定義している。SP 800-38A は、承認されたブロック暗号化アルゴリズムと共に使用される。この勧告への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

26. [SP 800-38B]

SP 800-38B: ブロック暗号動作モードの推奨事項 - 認証用 CMAC モード, 2005 年 5 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-38B は、対称鍵ブロック暗号化アルゴリズムと組み合わせて使用されるメッセージ認証コード（MAC）アルゴリズムを規定している。 CMAC と呼ばれるこのブロック暗号ベースの MAC アルゴリズムは、認証の保証、したがってバイナリデータの完全性を保証するために使用される。この勧告への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）で検証される。

27. [SP 800-38C]

SP 800-38C: 推奨されるブロック暗号動作モード：認証と機密保持のための CCM モード, 2004 年 5 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-38C は、対称鍵ブロック暗号アルゴリズムのための CCM(Counter with Cipher Block Chaining-Message Authentication Code)と呼ばれる動作モードを定義している。CCM は、カウンタ（CTR）モードおよび暗号ブロック連鎖メッセージ認証コード（CBC-MAC）[26]アルゴリズムの技術を組み合わせることによって、コンピュータデータの機密性および（メッセージ）認証の保証を提供する。この勧告への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

28. [SP 800-38D]

SP 800-38D: 推奨されるブロック暗号動作モード：Galois 体上の Counter Mode（GCM）と GMAC， 2007 年 11 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-38D は、関連するデータの（メッセージ）認証と暗号化ためのガロア体上のカウンタモードアルゴリズム(GCM)、および、平文データのメッセージ認証コード（MAC）を生成するために GCM を特化した GMAC を規定している。GCM および GMAC は、承認された対称鍵ブロック暗号アルゴリズムのための動作モードである。この勧告への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

29. [SP 800-38E]

SP 800-38E:（ドラフト）推奨されるブロック暗号動作モード：ブロック志向の記憶装置における機密性のための XTS-AES モード, 2010 年 1 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-38E は、IEEE1619-2007 を参照して AES アルゴリズムの XTS-AES モードを承認する予定であり、ブロック指向のストレージデバイス上のデータの機密性を保護するためのオプションを規定する。このモードはデータまたはそのソース認証は提供しない。

30. [SP 800-38F]

SP 800-38F: 推奨するブロック暗号動作モード：鍵ラッピング方法, 2012 年 12 月,
http://csrc.nist.gov/publications/PubsSPs.html

SP800-38F は、承認された「鍵ラッピング」、すなわち暗号鍵の機密性および完全性の保護について規定している。

31. [SP 800-53]

SP 800-53: 連邦情報システムにおける推奨セキュリティ管理策, 2009 年 8 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-53 は、連邦政府の執行機関を支援する情報システムにおいて、セキュリティ管理策を選択および特定するためのガイドラインを提供している。このガイドラインは、連邦政府の情報を処理、保存、または送信する情報システムの全ての構成要素に適用される。このガイドラインは、連邦政府内でより安全な情報システムの実現を、以下の手順によって支援する。

[26] CBC-MAC は、CCM モードの構成要素として用いる場合を除き、現在、承認されている動作モードではない。

a) 情報システムのセキュリティ管理策を選択し特定するための、より一貫性があり、同等で、反復可能なアプローチを促進する。

b) 連邦情報処理標準（FIPS）199、連邦情報および情報システムのセキュリティ分類基準に従って分類された情報システムに対する最小限のセキュリティ管理策の勧告を提供する。

c) 変化する要求や技術の要求を満たすために、情報システムのためのセキュリティ管理の動的で拡張可能なカタログを促進する。

d) セキュリティ管理策の有効性を判断するための評価方法と手順を開発するための基盤を作成する。

32. [SP 800-53A]

SP 800-53A: 連邦情報システムにおけるセキュリティ管理策の評価の手引き - 効果的なセキュリティ評価計画の構築, June 2010 年 6 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-53A は、効果的なリスク管理の枠組みの中で行われるセキュリティ管理策の評価を容易にすることを目的としている。以下の評価結果が、組織の管理者層に提供される。

a) 組織の情報システムにおけるセキュリティ管理策の有効性に関する証拠。

b) 組織内で採用されているリスク管理プロセスの質の指標。

c) 洗練された脅威のある地球規模の環境において、連邦政府の重要な任務および適用を支援している情報システムの長所と短所に関する情報。

33. [SP 800-56A]

SP 800-56A: 離散対数ベースの公開鍵暗号の推奨される鍵確立方式（改訂）, 2013 年 3 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-56A は、認定規格委員会（ASC）X 9 によって開発された規格に基づいて、離散対数ベースの公開鍵暗号を使用した鍵確立方式 : ANS X 9.42（離散対数ベースの公開鍵暗号を使用した対称鍵の共有）および ANS X 9.63（楕円曲線暗号を使用した鍵共有と鍵配送）を規定している。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

34. [SP 800-56B]

SP 800-56B: 因数分解の困難性に基づく公開鍵暗号を使用した推奨推奨される鍵対確立, August 2009 年 8 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-56B は、認定規格委員会（ASC）が開発した“ANS X9.44、素因数分解問題の困難性に基づく公開鍵暗号を用いた鍵確立”に準拠した、素因数分解問題の困難性に基づく公開鍵暗号を用いた鍵対確立を規定している。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

35. [SP 800-56C]

SP 800-56C: "Extraction-then-Expansion"を用いた推奨される鍵導出, November 2011 年 11 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-56C は、［SP800-56A］または［SP800-56B］で規定された鍵確立スキームの過程で生成された共有秘密から鍵素材を導出するための“extraction-then-expansion”を用いた二段階（鍵）導出手順を規定している。この“extraction-then-expansion”手順の承認された亜種を使用するいくつかの特定用途向け導出関数は NIST SP 800-135 に記述されている。

36. [SP 800-57-part1]

SP 800-57-part 1: 鍵管理のための推奨事項 – 第一部: 全般,July 2012 年 7 月 改訂,

http://wwwcsrc.nist.gov/publications/PubsSPs.html

SP 800-57-part 1 では、暗号鍵の管理に関する問題、暗号鍵の生成、使用、そして最終的な破壊について扱う。アルゴリズムの選択と適切な鍵長、暗号化ポリシー、暗号化モジュールの選択などの関連トピックも含む。

37. [SP 800-57-part3]

SP 800-57-part 3: 鍵管理のための推奨事項– 第三部: 特定用途向け鍵管理の手引き 2009 年 12 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-57-part 3 は、主にシステム管理者やシステム構築担当が、一般的に使用されるアプリケーションを適切に保護することに対して支援を行うことを目的としている。この手引きでは、通常の使用時に管理下に置かれているアプリケーションのオプションに関する情報もエンドユーザに提供する。選択したアプリケーションのセットに対して推奨事項が示されている。

38. [SP 800-67]

SP 800-67: トリプルデータ暗号化アルゴリズム（TDEA）ブロック暗号, 2012 年 1 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-67 は、主要コンポーネント(暗号エンジン)であるデータ暗号化アルゴリズム（DEA）を含むトリプルデータ暗号化アルゴリズム（TDEA）を規定している。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

39. [SP 800-89]

SP 800-89: デジタル署名アプリケーションの保証要件, November 2006 年 11 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-89 は、有効なデジタル署名に必要な保証（ドメイン・パラメタの有効性の保証、公開鍵の有効性の保証、鍵対の所有者が実際にプライベート鍵を所有することの保証、および鍵対所有者の身元の保証）を取得する方法を規定している。

40. [SP 800-90A]

SP 800-90A: 決定的ランダムビット生成器を用いた乱数生成（改訂） , January 2012 年 1 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-90A は、決定的方法を用いてランダムビットを生成するためのメカニズムを規定している。その場合、ランダムビットは直接使用されるか、または暗号を使用するアプリケーションが要求する時点で乱数に変換される。提供される方法は、ハッシュ関数、ブロック暗号アルゴリズム、または数論的問題に基づいている。この規格への準拠は、NIST 暗号アルゴリズム検証プログラム（CAVP）の下で検証される。

80. [SP 800-102]

SP 800-102: デジタル署名の適時性に関する推奨, 2009 年 9 月,
http://www.csrc.nist.gov/publiations/PubsSPs.html

SP 800-102 は、メッセージがデジタル署名で署名された時間の証拠を提供するための技法を説明している。

81. [SP 800-108]

SP 800-108: 擬似乱数関数を使用した鍵導出. 2009 年 10 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP 800-108 では、疑似乱数関数を使用する鍵導出関数のいくつかのファミリを規定している。これらの鍵導出関数は、自動鍵確立スキームを通じて確立された鍵（たとえば[SP 800-56A]および[SP 800-56B]参照）、または（手動で配付された鍵など）事前共有鍵から追加の鍵を導出するために使用される。

82. [SP 800-115]

SP 800-115: 情報セキュリティのテストと評価に関するテクニカルガイド, 2008 年 9 月,
http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-115 は、情報セキュリティ評価を実施するための基本的な技術面のガイドである。この文書は、組織が評価の一環として使用する可能性のある技術的なテストおよび検査方法と技法を提示し、その実施およびシステムやネットワークに対する潜在的な影響について評価者に洞察を提供する。評価が成功し、システムのセキュリティ状態（そして最終的には組織全体）に良い影響を与えるためには、テストと検査の実施以外の要素が技術的プロセスをサポートしなければならない。このガイドでは、堅牢な計画立案プロセス、根本原因の分析、調整された報告など、これらの活動に関する提案も提示している。

83. [SP 800-126]

SP 800-126-r2: セキュリティ設定共通化手順(SCAP[27])技術仕様: SCAP Versin 1.2, 2011 年 9 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-126-r2 は、そのコンポーネント仕様、それらの相互関係と相互運用、および SCAP コンテンツの要件という観点から、SCAP バージョン 1.2 の技術的構成を定義している。本 SCAP の技術仕様では、SCAP に準拠したコンテンツの一貫性のある正確な交換と、SCAP に準拠した製品でのコンテンツの信頼性の高い機能を保証するために採用される要件と規則について解説している。SCAP は、ソフトウェアの欠陥とセキュリティ構成情報がマシンと人間の両方に伝達されるフォーマットと命名法を標準化する一連の仕様である。SCAP は、自動構成、脆弱性とパッチのチェック、技術管理のコンプライアンス活動、およびセキュリティ測定をサポートする仕様の多目的フレームワークである。SCAP 開発の目標には、システムセキュリティ管理の標準化、セキュリティ製品の相互運用性の促進、およびセキュリティコンテンツの標準表現の使用の促進などがある。

[27] 【訳注】我が国における SCAP の状況に関しては、https://www.ipa.go.jp/security/vuln/SCAP.html を参照されたい。

84. [SP 800-131A]

SP 800-131A: 移行：暗号化アルゴリズムと使用する鍵長の移行に関する推奨, January 2011 年 1 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-131A は、機微（機密だが未分類）な情報を保護するための、連邦政府機関による暗号の使用に関連する移行措置（暗号アルゴリズムと使用する鍵長）について、より詳細に説明している。（2012 年 6 月の FIPS140-2 および CMVP プログラムの実装ガイダンス、G14 および G15 も参照すること）。

85. [SP 800-132]

SP 800-132: *パスワードベースの鍵導出のための推奨、第 1 部：ストレージアプリケーション*, 2010 年 12 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-132 は、電子的に保存されたデータの保護またはデータ保護鍵の保護のために、パスワードまたはパスフレーズから暗号鍵を導出するための、パスワードベースの鍵導出関数（PBKDF）の族を規定している。

86. [SP 800-135]

SP 800-135 Revision 1: 既存のアプリケーション固有の鍵導出関数に関する推奨, 2011 年 12 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-135 は、いくつかの現在のセキュリティ標準における既存アプリケーション固有の鍵導出機能のためのセキュリティ要件を規定している。

87. [SP 800-147]

SP 800-147: BIOS 保護ガイドライン, 2011 年 4 月,

http://www.csrc.nist.gov/publications/PubsSPs.html

SP800-147 は、PC クライアントシステム上の BIOS（Basic Input / Output System）ファームウェアの不正な変更を防止するためのガイドラインを提供している。悪意のあるソフトウェアによる BIOS ファームウェアの許可されていない変更は、PC アーキテクチャにおける BIOS の独特で特権的な役割のために重大な脅威となる。悪意のある BIOS の変更は、組織への巧妙な標的型攻撃の一部（恒久的なサービス拒否（BIOS が破損している場合）または永続的なマルウェアの存在（BIOS にマルウェアが埋め込まれている場合））となる可能性がある。

88. [TLS]

IETF が作成したトランスポート層セキュリティ関連の RFC

http://www.ietf.org/dyn/wg/charter/tls-charter.html

これらの TLS に関わる RFC では、認証、暗号化、および完全性等のセキュリティサービスが、HTTP パケット上でどのように提供されるかについて規定している。これらの RFC は、パケット用のセキュリティサービス用ペイロードのフォーマット、セキュリティサービス用の暗号スイート、およびセキュリティサービスを提供するために使用される暗号化アルゴリズム用の鍵管理技術を網羅している。

89. [X.509]

X.509：情報技術 - 開放型システム間相互接続 - ディレクトリ：公開鍵証明書と属性証明書のフレームワーク, IEC 9594-8.

この国際標準規格は、証明書、証明書パス、および CRL 処理規則とともに、X.509 公開鍵証明書および属性証明書とそれらに関連する CRL のフォーマットを定義している。

90. [XML DSIG]

XML 署名の構文と処理（第 2 版）, W3C 勧告 2008 年 6 月 10 日,

http://www.w3.org/TR/xmldsig-core

XML DSIG は、XML 文書におけるデジタル署名のフォーマット、およびデジタル署名の検証を支援するために使用される補助情報（例えば、証明書、CRL、署名者識別子など）のフォーマットを規定している。

91. [XML ENC]

XML 暗号化の構文と処理, W3C 勧告 2002 年 12 月 10 日,

http://www.w3.org/TR/xmlenc-core

XML ENC は、暗号化された XML 文書および復号を支援するための補助情報(例えば、証明書、CRL、受信者識別子など)のフォーマットを規定する。

附属書 B：用語集

以下の用語集は本フレームワークで使用されている用語と定義を含んでいる。読者は［SP800-57-part 1］に含まれる用語集も参照するべきである。

活性化状態 Active State	鍵ライフサイクル状態における一つの状態であり、一連のアプリケーション、アルゴリズム、およびセキュリティエンティティにおいて、暗号鍵が使用可能となる状態
暗号アルゴリズムの移行 Algorithm Transition	ある暗号アルゴリズムを別の暗号アルゴリズムに置き換えるために使用されるプロセスと手順
匿名性 Anonymity	CKMSをサポートする通信において、公開データと所有者とを関連付けることができないことを保証する性質
アプリケーション Application	一連の目標を達成したり、一連のサービスを提供したりするように設計され運用されているコンピュータプログラム
アーカイブ （鍵やメタデータ） Archive (key and/or metadata)	記憶技術が変化しても維持される長期記憶媒体に電子的な暗号鍵やメタデータを格納すること。あるいは、アーカイブされた鍵やメタデータが格納される場所
関連付けられたメタデータ （メタデータともいう） Associated Metadata (also Metadata)	本フレームワーク（SP800-130）では、CKMSによって明示的に記録され、管理され、保護される暗号鍵に関連付けられたプロパティを記述するために使用されるパラメタ
関連付け機能 Association Function	SP800-130では、鍵とメタデータを不正な変更や開示から保護し、メタデータの起源を認証する機能
監査 Audit	システムのセキュリティに関して、監査責任者がシステム管理者に報告するために、必要なデータを収集、分析、要約するための手順
権威時刻ソース Authoritative Time Source	正確な時間を提供する、信頼されるネットワークエンティティ。日本国内では、NICT公開NTPサービスが該当する
バックアップ （鍵やメタデータ） Backup (key and/or metadata)	運用過程において、鍵やメタデータの元の値が失われたり、変更されたりした場合に、鍵やメタデータを復元できるように、鍵やメタデータを別の設備にコピーすること
暗号鍵管理システム CKMS	暗号鍵とメタデータを保護、管理、および、配付するために準備された方針、手順、デバイス、コンポーネント。CKMSは、1つまたは複数のエンティティに代わって暗号鍵管理機能を実行する
CKMSコンポーネント （コンポーネント） CKMS Component (Component)	CKMSを実現（実装）するために使用されるハードウェア、ソフトウェア、またはファームウェア
CMKSデバイス （デバイス） CKMS Device (Device)	特定の目的（ファイアウォール、ルータ、伝送デバイス、暗号モジュール、データストレージデバイスなど）の機能を提供するCKMSコンポーネントの任意の組み合わせ
CKMSモジュール CKMS Module	特定の場所で必要な全てのCKMS機能を実行する論理エンティティ
CKMSプロファイル CKMS Profile	利害関係にあるコミュニティ（例えば、米国政府、銀行、健康医療分野、航空宇宙など）が使用する実装に依存しないCKMSセキュリティ要件の仕様を規定する文書

商用既製品 Commercial Off-The-Shelf（COTS)	出来合いの技術やプロダクトであり、一般に販売、リース、またはライセンス供与が可能な製品
適合CKMS Compliant CKMS	本フレームワーク内で規定された各要件に対応した設計仕様を満たすCKMS
危殆化 Compromise	（例えば、鍵素材あるいは関連する情報のような）機微なデータの承認されていない開示、改変、利用
危殆化状態 Compromised State	鍵ライフサイクル状態における一つの状態であり、その鍵が侵害されたと指定され、データを暗号化して保護するために使用してはならない状態。特定の状況下では、その鍵が保護されたデータを処理するために使用されることがある
暗号解読 Cryptanalyze	暗号メカニズム、より一般的には数学的手法を応用した情報セキュリティサービスを無効にする
暗号学的結び付き（結び付き） Cryptographic Binding（Binding)	CKMSを用いて鍵と選択されたメタデータ要素との間の信頼関係を確立するに、1つまたは複数の暗号技術の使用すること
暗号境界 Cryptographic Boundary	暗号モジュールの全てのコンポーネントの境界を確立する明示的に定義された境界
暗号鍵（鍵） Cryptographic Key (Key)	暗号アルゴリズムのパラメタを構成するビット、整数、または文字列
暗号鍵管理システム Cryptographic Key Management System	暗号鍵とそのメタデータの管理システム（世代管理、流通管理、ストレージ、バックアップ、アーカイブ、リカバリ、使用、失効、破壊など）
暗号モジュール（モジュール） Cryptographic Module (Module)	セキュリティ機能（例えば、暗号アルゴリズムおよび鍵の確立）を実装した、暗号境界内に含まれるハードウェアやソフトウェア、ファームウェアの集合
暗号責任者 Cryptographic Officer	CKMSの暗号コンポーネントおよびデバイスに対して暗号初期化および管理機能を実行する権限を与えられた個人
暗号（技術） Cryptography	機密性、データの完全性、エンティティ認証、データ発信元認証などのセキュリティサービスを提供する数学の使い方
暗号鍵有効期間 Cryptoperiod	特定の鍵の使用が承認されるか、与えられたシステムやアプリケーションにおいて鍵の有効性が残存するタイムスパン
非活性化状態 Deactivated State	鍵ライフサイクル状態における一つの状態であり、鍵がデータを暗号学的に保護するために使用してはならない状態。 特定の状況下では、その鍵が保護されたデータを処理するために使用されることがある
設計者 Designer	新しいシステムを構成するあらゆるデバイスと、そのデバイスがどのように構造され、調整され、運用されるかを指定する能力、責任、権限を持つ人物または組織
破壊状態 Destroyed State	鍵ライフサイクル状態における一つの状態であり、鍵の復元または使用もできない状態
危殆化鍵破壊状態 Destroyed Compromised State	鍵ライフサイクル状態における一つの状態であり、鍵の復元も使用もできず、かつ、危殆化と識別されている状態

セキュリティドメイン （ドメイン） Security Domain (Domain)	共通の目標と持つ（人、組織、情報システムなどの）要素のグループを含む論理的なエンティティ
Entity	(SP800-57Part1)単一の信頼すべき権威機関（Trusted Authority）が認可したシステムまたはサブシステム。セキュリティドメインはより大きな領域を形成するために（例．階層的に）組織化されることもある
エンティティ Entity	人、グループ、デバイス又はプロセス。エンティティは、関連づけのために識別子を持つ。（パーティとも呼ばれることがある）
ドメインのセキュリティポリシーの同等性 Equivalent Security Domain Policies	2つの領域で提供される同等の保護の方法で、一方の領域と他の領域で暗号鍵が交換できる場合、2つのセキュリティ領域方針は、等価である
拡張性 Extensibility	システムの能力向上の容易さの尺度
ファイアウォール Firewall	望ましくないアプリケーションやリモートユーザーが、アクセスしたり操作したりするのを検出して防止するセキュリティで保護されたそのコンピュータ上のOSと統合されたプロセス
形式言語 Formal Language	規則が明白であり、言語の全ての構文的に正しい文がオートマトン（例えば、構文解析アプリケーションプログラムを実行するコンピュータ）によって正しいと認識されるように構文（適切な構造の正しい文章を作成するための規則）が定義された言語
フレームワーク Framework	CKMSの作成に使用されるポリシー、手順、コンポーネント、およびデバイスの説明
破損 Garbled	1つまたは複数の要素（例えば、ビット、数字、文字）が、変更または破壊されるデータ（例えば、暗号鍵）の変形
鍵生成 Generate Key	暗号鍵を計算または作成するために使用される鍵およびメタデータ管理機能
堅牢化 Hardening	脆弱性にパッチを当てて不必要なサービスを無効にして攻撃手段を排除するプロセス。コンピュータの耐性強化には、いくつかの段階で保護層を形成する必要がある
ハッシュ値 Hash Value	ハッシュ関数によって生成された固定長のビット列
識別子 Identifier	エンティティ（例えば、鍵管理機能を実行するもの）を示すために、またCKMSアクセス制御システムによって、鍵の集合から特定の鍵を選択するために使用されるテキスト文字列
相互運用性 Interoperability	1組のエンティティが別のエンティティのセットに物理的に接続し、論理的に通信する能力の尺度
鍵 Key	「暗号鍵」の項を参照
鍵合意 Key Agreement	結果として生じる鍵素材は、二者以上の関係者が寄与した情報の機能であり、単独の関係者では値を定めることができない鍵確立手続き
鍵確認 Key Confirmation	あるエンティティ（鍵確認受容者）に、別のエンティティ（鍵確認プロバイダ）が実際に正しい秘密の鍵素材や共通の秘密を所有していることを保証する手順

鍵配付 Key Distribution	鍵を所有しているか、または鍵を生成しているエンティティから、その鍵を使用する別のエンティティへの鍵およびその他の鍵素材を移送すること。（キートランスポートとも呼ぶ）
鍵入力 Key Entry	鍵（およびおそらくそのメタデータ）をアクティブな使用に備えて暗号モジュールに入力するプロセス
鍵確立 Key Establishment	あるエンティティから別のエンティティへ鍵を移送（鍵配送）するか、エンティティが共有する情報から鍵を導出（鍵共有）することによって、鍵が2つ以上の エンティティ 間で安全に共有されるプロセス
鍵ラベル Key Label	鍵ラベルは、人間が読み取り可能で、おそらく機械で読み取り可能な鍵の記述子のセットを提供するテキスト文字列。鍵ラベルの仮想的な例：「ルートCAプライベート鍵2009-29」、「メンテナンス共有鍵2005」
鍵ライフサイクル状態 Key Lifecycle State	鍵のライフサイクルとして活性化前、活性化（アクティブ）、一時停止中、無効化、失効、危殆化、破壊、危殆化鍵破壊などの鍵使用状態を表す有限状態の集合（モデル）
鍵出力 Key Output	鍵（およびおそらくそのメタデータ）が暗号モジュールから（通常は外部ストレージに）抽出されるプロセス
鍵所有者 Key Owner	暗号鍵または鍵ペアを使用することが許可されたエンティティ（例えば、人、グループ、組織、デバイス、またはモジュール）
鍵分割 Key Split	1つ以上の鍵分割片を適切に組み合わせることで、暗号鍵を形成する（再生する）パラメタ
鍵状態の遷移 Key State Transition	1つの鍵ライフサイクル状態から別の鍵ライフサイクル状態に移行するプロセス
鍵配送 Key Transport	ある当事者（送信者）が鍵素材を選択および暗号化し、別の当事者（受信者）に材料を配付するような鍵確立手順。典型的には、鍵配送は、鍵素材を保護するために暗号を使用するが、一部のアプリケーションでは、（暗号の）代わりに、信頼できる配達人を使用することができる。（鍵配付と呼ぶこともある）
鍵更新 Key Update	古い暗号鍵を置き換えるために、新しい暗号鍵を古い暗号鍵を利用して計算する手続き
鍵ラッピング Key Wrapping	機密性および完全性の両方の保護を提供するように、対称鍵を用いて（関連する完全情報と一緒に）鍵を暗号学的に保護する方法
鍵素材 Keying Material	鍵やメタデータ
最小権限の原則 Least Privilege	各エンティティは正当な使用に必要な情報とリソースにのみアクセスできるという（設計）原則
マルウェア Malware	（スパイウェア、ウイルスプログラム、ルートキット、トロイの木馬を含む）コンピュータのセキュリティに違反するように敵対者によって設計され、操作されるソフトウェア
メタデータ （関連付けられたメタデータ） Metadata (also Associated Metadata)	本フレームワーク（SP800-130）では、CKMSによって明示的に記録され、管理され、保護される暗号鍵に関連付けられたプロパティを記述するために使用されるパラメタ
メタデータ要素 Metadata Element	鍵に関連付けられ、CKMSによって明示的に記録および管理されるメタデータの1ユニット

暗号利用モード Mode of Operation	暗号アルゴリズムと鍵を使ってデータを操作するための一連のルール。処理される追加データの有無にかかわらず、アルゴリズムの出力の全部または一部をアルゴリズムの次の反復の入力に戻すことを含むことが多い。例としては、暗号フィードバック（CFB）、出力フィードバック（OFB）、暗号ブロック連鎖（CBC）などがある
パラメタ Parameters	特定のセキュリティ目標を達成するために有用な出力を計算するために暗号アルゴリズムで使用される特定の変数とその値
パーティ Party	エンティティを参照
活性化前状態 Pre-Activation State	鍵ライフサイクル状態における一つの状態であり、鍵の使用がまだ許可されていない状態
プライバシー Privacy	エンティティに関する特定の情報の機密性とアクセス権が保護されていることの保証
プロファイル Profile	顧客領域（例えば、連邦、私企業、または国際）の基準に適合するCKMSを作成するために使用される方針、手順、コンポーネントおよびデバイスの仕様
量子ビット Qubit	量子コンピュータでの、量子情報の単位 - 古典的なビットに対する量子的なアナロジー
回復（一般用語） Recover (General)	戻す; 回復する
（鍵やメタデータの）復元 Recover (key and/or metadata)	バックアップまたはアーカイブ（ストレージ）から鍵やメタデータを取得または再構築すること
登録 Registration	登録エージェントによって実行される、エンティティの識別と認可の検証、エンティティの識別子や他のメタデータに対するエンティティの鍵の関連付けを確立する、信頼できる手順の集合
鍵再設定 Rekey	交換される（古い）暗号鍵に依存しない方法で新しい暗号鍵が生成される手続き
更新 Renewal	公開鍵の有効期限を延長して、延長した期間に公開鍵を使用できるようにするための手続き
失効状態 Revoked State	鍵ライフサイクル状態における一つの状態であり、以前にアクティブだった暗号鍵を、データ保護の暗号化に使用することを禁止する状態
役割 Role	個人または組織が環境またはコンテキスト範囲内で実行することが認可された、受け入れ可能な機能、サービス、およびタスクの集合
ルートキット Rootkit	標準のOS機能や他のアプリケーションを損傷させて、管理者からの存在を積極的に隠したコンピュータへの権限のないアクセスを可能にするマルウェア
ルータ Router	データパケットを送受信し、多様な通信エンティティのセット間で論理接続を確立する物理的または論理的なエンティティ（通常、有線および無線の両方の通信デバイスを同時にサポートする）
スケーラビリティ Scalability	増大する負荷に応じて適切に処理する能力、あるいはそのようなプロセスの増大分を収容するために拡張できる能力
スキーム Scheme	適切に実装され、維持されている場合に（暗号化）サービスを提供できる変換のセットに関する明確な仕様。スキームは、プリミティブより上位の構造体であり、プロトコルよりも下位の構造体

セクタ Sector	共通の目標、基準、および製品、システム、またはサービスの要件を持つ組織の集まり（連邦政府機関、民間組織、国際的なコンソーシアムなど）
セキュリティドメイン Security Domain	CKMSを有するエンティティの集合。各々のCKMSは、同じセキュリティポリシー（ドメインのセキュリティポリシー）で運用される
セキュリティポリシー Security Policy	情報とサービスの許容される使用を管理し、情報の機密性、完全性、可用性を保護するための許容されるレベルと手段についての、組織により確立された規則と要件
セキュリティ強度 Security Strength	暗号アルゴリズムまたは暗号システムを暗号解読するために必要な作業量（2を底とする対数で表現された最少数）に関連付けられた数値
セマンティクス Semantics	ある言語において受理可能な文が意図する内容
標準 Standard	モデル や 例示 として、権威（例えば、国や標準化機関）、慣行、または共通的な認識によって確立されたもの
（鍵やメタデータの）格納 （鍵やメタデータの）保管 Store (key and/or metadata)	鍵やメタデータを、復元するための媒体に移動すること
一時停止状態 Suspended State	鍵ライフサイクル状態における一つの状態であり、以前に活性化状態だった鍵を、一時的にその状態から退避させるための状態。必要に応じて鍵を活性化状態に戻すことができる
シンタックス Syntax	ある言語で受理できる文章を構成するための規則
信頼 Trust	エンティティとその識別子が本物であるという保証とともに、特定の機能またはサービスを正しく、公平に、公正に実行する能力を示すエンティティの特性
トラストアンカー Trust Anchor	信頼木の基底に存在する、または信頼チェーン内の最も強いリンクとして存在する、1つ以上の信頼された公開鍵。当該公開鍵を基点にCKMSでの公開鍵基盤（PKI）が構成される
トラストアンカーストア Trust Anchor Store	トラストアンカー情報が格納される場所
信頼関係 Trusted Association	鍵とそのメタデータが適切に関連付けられ、特定の情報源から発信され、変更されておらず、不正な開示から保護されているという保証 を提供するために、選択されたメタデータ要素と鍵のリンク
連結不可能性 Unlinkability	CKMSでサポートされている通信で、情報処理システム内の2つ以上の関連イベントを関連付けることができないことの保証
観測不可能性 Unobservability	観測者がCKMSでサポートされている通信データから、関与する当事者を特定または推論することができないことの保証
ユーザ User	情報システム、1つ以上のアプリケーション、セキュリティ手順とサービス、およびサポートするCKMSを使用するために、組織およびポリシーによって承認された個人
検証 Validate	実装が使用に適切であるという保証を得るために、暗号パラメタまたはモジュールをテストし、そのテスト結果を確認すること
有効期間 Validity Period	公開鍵証明書の有効期間

附属書 C：頭字語

以下のリストは本フレームワークで使用されている頭字語を含んでいる。

ACS	Access Control System
AES	Advanced Encryption Standard
ANS	American National Standard
CBC	Cipher Block Chaining
CA	Certificate (Certification) Authority
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CKL	Compromised Key List
CKMS	Cryptographic Key Management System(s)
CMS	Certificate Management System
COTS	Commercial Off-The-Shelf
CRL	Certificate Revocation List
DNSSEC	Domain Name System Security Extensions
EAP	Extensible Authentication Protocol
E-Mail	Electronic Mail
EC	Elliptic Curve
ECB	Electronic Codebook
EFS	Electronic File System
FIP	Federal Information Processing Standard
FISMA	Federal Information Security Management Act
FR	Framework Requirement
fr	Framework Response
FT	Framework Topic
HMAC	Keyed-Hash Message Authentication Code
IDS	Intrusion Detection System
IKE	Internet Key Exchange
IP	Internet Protocol
IPSec	Internet Protocol Security
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OAEP	Optimal Asymmetric Encryption Padding
OCSP	Online Certificate Status Protocol
OFB	Output Feed Back
OID	Object Identifier
OMB	Office of Management and Budget
OTAR	Over-The-Air Rekeying

PKCS	Public Key Cryptographic Standards
PSS	Probabilistic Signature Scheme
RFC	Request For Comment
RSA	Rivest, Shamir and Adleman (Algorithm)
SCAP	Security Content Automation Protocol
S/MIME	Secure/Multipurpose Internet Mail Extensions
SP	Special Publication
SSH	Secure Shell
TDEA	Triple Data Encryption Algorithm
TLS	Transport Layer Security
VPN	Virtual Private Network
XML	Extensible Markup Language