CRYPTREC Report 2005

平成 18 年 3 月

独立行政法人情報通信研究機構
独立行政法人情報処理推進機構

「暗号技術監視委員会報告」

目次

はじめに

現在、CRYPTREC 活動は平成 15 年度に発足した「暗号技術監視委員会」と「暗号モジュール委員会」を中心に行われている。両委員会とも総務省及び経済産業省が主催している暗号技術検討会の下で活動をしており、前者は電子政府推奨暗号の安全性の監視等、後者は電子政府推奨暗号を実装する暗号モジュールの評価基準・試験基準の作成等を行っている。本書は、“**暗号技術監視委員会の平成 17 年度の活動報告書**”である。

暗号技術監視委員会の前身とも言える暗号技術評価委員会では平成 12 年度から平成 14 年度の 3 カ年をかけて我が国の電子政府(e-Government)で利用可能な暗号技術のリストアップを目的とした暗号技術評価活動(暗号アルゴリズムの安全性評価)を推進してきた。

その結果、平成 14 年度末に、暗号技術検討会を主催する総務省、経済産業省が電子政府推奨暗号リストを公表する運びとなり、暗号技術評価活動も一区切りを迎えた。

平成 15 年度からは、暗号技術の安全性に係わる研究開発動向の監視活動を担うために暗号技術監視委員会が設置された。さらに暗号技術関連の学会、国際会議、関係団体の Web サイト等から、電子政府推奨暗号の安全性に影響を与えかねない情報を収集し、分析するための監視要員が事務局内に配置された。

暗号技術監視委員会は、独立行政法人情報通信研究機構及び独立行政法人情報処理推進機構が共同で運営しており、技術面を中心とした活動を担当している。一方、ユーザの立場でかつ政策的な判断を加えて結論を出しているのが暗号技術検討会であり、相互に協調して電子政府の安全性及び信頼性を確保する活動を推進している。

“監視活動元年”の平成 15 年度は、暗号技術監視委員会、暗号技術調査ワーキンググループの設置、監視要員の配置等、監視体制の樹立に始まり、監視活動方針・手順の確立とそれに従った監視活動及び関連調査活動等を行った。昨年（平成 16 年）度は、引き続き監視活動及び関連調査活動等を実施した。平成 16 年 8 月にはハッシュ関数 SHA-0、SHA-1 に対する衝突（collision）発見方法について注目すべき学会発表があり、監視活動の一環として SHA-1 の危殆化に関する調査を強化した。今年（平成 17 年）度は、ハッシュ関数の危殆化が危惧されることからハッシュ関数の安全性について再検討するとともに、電子署名法の指針の改訂および電子政府推奨暗号リストの見直しに向けた活動を重点に実施した。

電子政府推奨暗号の監視は、暗号が使われ続ける限り継続していかなければならない活動である。また、この活動は、暗号モジュール委員会との連携を保ちつつ、暗号技術の研究者、実装技術者等の多くの関係者の協力を得て成り立っているものであることを改めて強調しておきたい。

末筆ではあるが、本活動に様々な形でご協力下さった関係者の皆様に謝意を表する次第である。

暗号技術監視委員会　委員長　今井 秀樹

本報告書の利用にあたって

本報告書の想定読者は、一般的な情報セキュリティの基礎知識を有している方である。たとえば、電子政府において電子署名や GPKI システム等暗号関連の電子政府関連システムに関係する業務についている方などを想定している。しかしながら、個別テーマの調査報告等については、ある程度の暗号技術の知識を備えていることが望まれる。

本報告書の第 1 章は暗号技術監視委員会及び監視活動等について説明してある。第 2 章は今年度の監視活動、調査等の活動概要の報告である。第 3 章は暗号技術監視委員会の下で活動している暗号技術調査ワーキンググループの活動報告である。

本報告書の内容は、我が国最高水準の暗号専門家で構成される「暗号技術監視委員会」及びそのもとに設置された「暗号技術調査ワーキンググループ」において審議された結果であるが、暗号技術の特性から、その内容とりわけ安全性に関する事項は将来にわたって保障されたものではなく、今後とも継続して評価・監視活動を実施していくことが必要なものである。

本報告書ならびにこれまでに発行された CRYPTREC 報告書、技術報告書、電子政府推奨暗号の仕様書は、CRYPTREC 事務局（総務省、経済産業省、独立行政法人情報通信研究機構、及び独立行政法人情報処理推進機構）が共同で運営する下記の Web サイトで参照することができる。

http://www.cryptrec.jp/

本報告書ならびに上記 Web サイトから入手した CRYPTREC 活動に関する情報の利用に起因して生じた不利益や問題について、本委員会及び事務局は一切責任をもっていない。

本報告書に対するご意見、お問い合わせは、CRYPTREC 事務局までご連絡いただけると幸いである。

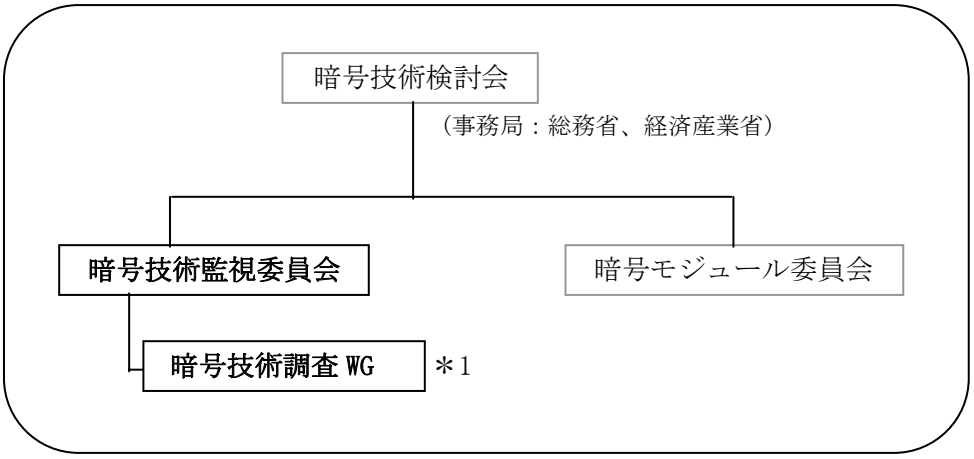
【問合せ先】　info@cryptrec.jp

委員会構成

暗号技術監視委員会(以下「監視委員会」)は、総務省と経済産業省が共同で主催する暗号技術検討会の下に設置され、独立行政法人情報通信研究機構（NICT）と独立行政法人情報処理推進機構(IPA)が共同で運営する。監視委員会は、暗号技術の安全性及び信頼性確保の観点から、電子政府推奨暗号の監視、電子政府推奨暗号に関する暗号アルゴリズムを主な対象とする調査・検討を適宜行う等、主として技術的活動を担い､暗号技術検討会に助言を行う。また、将来的には、電子政府推奨暗号リストの改訂に関する調査・検討を行う予定である。なお、日常的な監視業務を行う監視要員を NICT 及び IPA に配置し、暗号技術関連学会や国際会議等を通じての暗号技術に関する情報収集、関係団体の Web サイトの監視等を行う。

暗号技術調査ワーキンググループ(以下「調査 WG」)は、監視委員会の下に設置され、NICT と IPA が共同で運営する。調査 WG は、監視委員会活動に関連して必要な項目について、監視委員会の指示のもとに調査・検討活動を担当する作業グループである。監視委員会の委員長は、実施する調査・検討項目に適する主査及びメンバーを、監視委員会及び調査 WG の委員の中から選出し、調査・検討活動を指示する。主査は、その調査・検討結果を監視委員会に報告する。現在、監視委員会の指示に基づき実施されている調査項目は、「署名・認証技術調査」、「ハッシュ関数・暗号利用モード調査」、「擬似乱数生成系調査」である。

監視委員会と連携して活動する「暗号モジュール委員会」も、監視委員会と同様、暗号技術検討会の下に設置され、NICT と IPA が共同で運営している。



＊1 今年度実施されている調査項目

1) 署名・認証技術の調査

2) ハッシュ関数・暗号利用モードの調査

3) 擬似乱数生成系の調査

図 1　CRYPTREC 体制図

委員名簿

暗号技術監視委員会

委員長	今井 秀樹	国立大学法人東京大学 教授
顧問	辻井 重男	情報セキュリティ大学院大学　学長
委員	太田 和夫	国立大学法人電気通信大学 教授
委員	金子 敏信	東京理科大学 教授
委員	佐々木 良一	東京電機大学 教授
委員	松本 勉	国立大学法人横浜国立大学 大学院 教授
委員	大塚 玲	独立行政法人情報処理推進機構 主任研究員
委員	田中 秀磨	独立行政法人情報通信研究機構 研究員
委員	山村 明弘	独立行政法人情報通信研究機構 グループリーダ
委員	渡辺 創	独立行政法人産業技術総合研究所 副研究センター長

暗号技術調査ワーキンググループ

委員	荒木 純道	国立大学法人東京工業大学 大学院 教授
委員	有田 正剛	情報セキュリティ大学院大学 教授
委員	小暮 淳	株式会社富士通研究所 主任研究員
委員	酒井 康行	三菱電機株式会社 主席研究員
委員	四方 順司	国立大学法人横浜国立大学 大学院 助教授
委員	新保 淳	株式会社東芝 主任研究員
委員	洲崎 誠一	株式会社日立製作所 主任研究員
委員	藤岡 淳	日本電信電話株式会社 主幹研究員
委員	松崎 なつめ	松下電器産業株式会社 主幹技師
委員	青木 和麻呂	日本電信電話株式会社 研究主任
委員	川村 信一	株式会社東芝 室長
委員	香田 徹	国立大学法人九州大学 大学院 教授
委員	古原 和邦	国立大学法人東京大学 助手
委員	下山 武司	株式会社富士通研究所 研究員
委員	大森 基司	松下電器産業株式会社 チームリーダ
委員	角尾 幸保	日本電気株式会社 主席研究員
委員	時田 俊雄	三菱電機株式会社　主席研究員
委員	古屋 聡一	株式会社日立製作所 研究員
委員	森井 昌克	国立大学法人神戸大学 教授
委員	栃窪 孝也	東芝ソリューション株式会社 SI 技術担当
委員	廣瀬 勝一	国立大学法人福井大学 助教授
委員	盛合 志帆	株式会社ｿﾆｰ･ｺﾝﾋﾟｭｰﾀｴﾝﾀﾃｲﾝﾒﾝﾄ ﾘｻｰﾁｻｲｴﾝﾃｨｽﾄ

オブザーバー

大貫 秀明	内閣官房 情報セキュリティセンター
齋藤 正憲	警察庁 情報通信局
山城 瑞樹	防衛庁 長官官房
加納 信生	防衛庁 陸上幕僚監部
山本 寛繁	総務省 行政管理局
岡本 成男	総務省 自治行政局
吉武 啓治	総務省 自治行政局
野崎 雅稔	総務省 情報通信政策局（平成 17 年 7 月まで）
藤田 和重	総務省 情報通信政策局
榎本 淳一	総務省 情報通信政策局（平成 17 年 7 月まで）
能登 治	総務省 情報通信政策局
黒田 崇	総務省 情報通信政策局（平成 17 年 7 月まで）
網野 尚子	総務省 情報通信政策局
石川 雅一	外務省 大臣官房（平成 17 年 7 月まで）
山本 明裕	外務省 大臣官房
勝亦 眞人	経済産業省 産業技術環境局
松井 洋二	経済産業省 商務情報政策局
柳原 聡子	経済産業省 商務情報政策局（平成 17 年 6 月まで）
太田 保光	経済産業省 商務情報政策局
滝澤　修	独立行政法人 情報通信研究機構
大蒔 和仁	独立行政法人 産業技術総合研究所

事務局

独立行政法人 情報通信研究機構

松島裕一、山村明弘、田中秀磨、外川政夫、黒川貴司、金森祥子、
吉野智明（平成 17 年 8 月から）

独立行政法人 情報処理推進機構

三角育生、西原正人、大塚玲、杉田誠、山岸篤弘、大熊建司、上野天徳、
大久保美也子（平成 17 年 9 月から）

第 1 章　活動の目的

1.1 電子政府システムの安全性確保

電子政府システムが平成 15 年度に本格的に始動した。電子政府システムの安全性の確保は緊急に対処しなければならない。内閣府高度情報通信ネットワーク社会推進戦略本部(IT 戦略本部)(http://www.kantei.go.jp/jp/singi/it2/index.html)は e-Japan 戦略 II(平成 15 年 7 月)を発行し、「新しい IT 社会基盤整備」において「安心・安全な利用環境の整備」を唱え、電子政府や電子自治体、重要インフラ等の公共的分野のサービスの情報セキュリティ対策の一層の充実が求めている。また、平成 15 年 8 月には、e-Japan 重点計画-2003、平成 16 年 6 月には、e-Japan 重点計画-2004 と計画の進展にともなってより具体的な施策が示されている。

これらの電子政府、電子自治体における情報セキュリティ対策は根幹において暗号アルゴリズムの安全性に依存している。情報セキュリティ確保のためにはネットワークセキュリティ、通信プロトコルの安全性、機械装置の物理的な安全性、セキュリティポリシー構築、個人認証システムの脆弱性、運用管理方法の不備を利用するソーシャルエンジニアリングへの対応と幅広く対処する必要があるが、暗号技術は情報セキュリティシステムにおける基盤技術であり、暗号アルゴリズムの安全性を確立することなしに情報セキュリティ対策は成り立たない。

このため、平成 17 年までに世界最先端の IT 国家になるとの目標を達成するための e-Japan 戦略Ⅱ加速化パッケージ(平成 16 年 2 月)においてもセキュリティ(安全・安心)政策の強化が政府として取り組むべき重点施策とされていて、各府省庁の情報セキュリティ確保において「攻撃の予兆や被害に関する情報収集・分析」が重要案件としてあげられている。

また、政府の IT 戦略本部が平成 17 年 5 月 30 日に設置した情報セキュリティ政策会議から出された「政府機関の情報セキュリティ対策のための統一基準（平成 17 年 12 月版（全体版初版））」（平成 17 年 12 月 13 日）においても、新規（更新を含む）に暗号化又は電子署名の付与のアルゴリズムを導入する場合には、電子政府推奨暗号リストの中から選択することが基本遵守事項として明記されている。

暗号技術の危殆化を予見し、電子政府システムで利用される暗号技術の安全性を確保するためには、最新の暗号理論の研究動向を専門家が十分に情報収集・分析することが不可欠である。

1.2 暗号技術監視委員会

電子政府システムにおいて利用可能な暗号アルゴリズムを評価・選択する活動が平成12年度から平成14年度まで暗号技術評価委員会(CRYPTREC: Cryptography Research and Evaluation Committees)において実施された。その結論を考慮して電子政府推奨暗号リスト(付録参照)が総務省・経済産業省において決定された。電子政府システムの安全性を確保するためには電子政府推奨暗号リストに掲載されている暗号の安全性を常に把握し、安全性を脅かす事態を予見することが重要課題となった。そのため平成15年度に電子政府推奨暗号の安全性に関する継続的な評価、電子政府推奨暗号リストの改訂に関する調査・検討を行うことが重要であるとの認識の下に、暗号技術評価委員会が発展的に改組され、暗号技術検討会の下に「暗号技術監視委員会」が設置された。暗号技術監視委員会の責務は電子政府推奨暗号の安全性を把握し、もし電子政府推奨暗号の安全性に問題点を有する疑いが生じた場合には緊急性に応じて必要な対応を行うことである。さらに暗号技術監視委員会は電子政府推奨暗号の監視活動のほかにも暗号理論の研究動向を把握し、将来の電子政府推奨暗号リストの改訂に技術面から支援を行うことを委ねられている。

1.3 電子政府推奨暗号リスト

平成12年度から平成14年度のCRYPTRECプロジェクトの集大成として、暗号技術評価委員会で作成された「電子政府推奨暗号リスト（案)」は、平成14年に暗号技術検討会に提出され、同検討会での審議ならびに（総務省・経済産業省による）パブリックコメント募集を経て、「電子政府推奨暗号リスト」（付録1参照）として決定された。そして、「各府省の情報システム調達における暗号の利用方針（平成15年2月28日、行政情報システム関係課長連絡会議了承)」において、可能な限り、「電子政府推奨暗号リスト」に掲載された暗号の利用を推進するものとされた。電子政府推奨暗号リストの技術的な裏付けについては、CRYPTREC Report 2002 暗号技術評価報告書（平成14年度版）に詳しく記載されている。CRYPTREC Report 2002 暗号技術評価報告書（平成14年度版）は、次のURLから入手できる。
http://www.cryptrec.jp/report.html

1.4 活動の方針

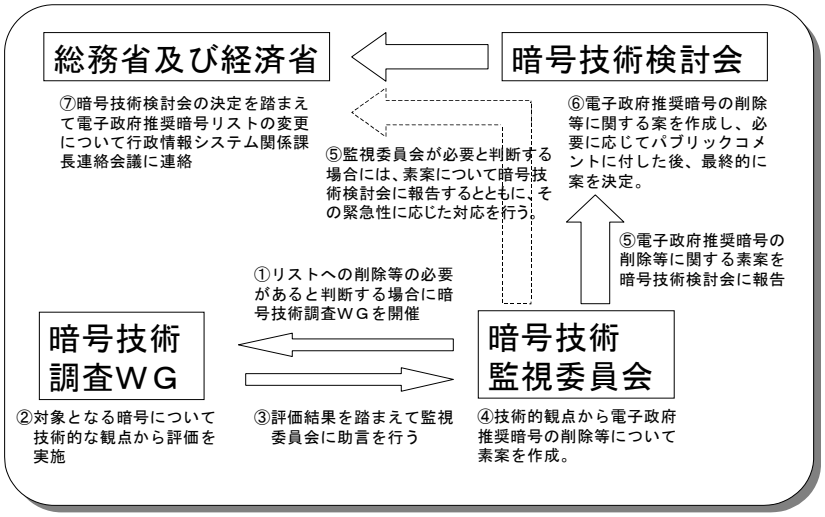
電子政府推奨暗号リスト掲載の暗号に関する研究動向を把握して、暗号技術の安全性について監視を行い、必要に応じて電子政府システムにおける暗号技術の情報収集と電子政府推奨暗号リストの改訂について暗号技術検討会(総務省・経済産業省)に対して助言を行う。また、暗号理論全体の技術動向を把握して、最新技術との比較を行い、電子政府システムにおける暗号技術の陳腐化を避けるため、将来の電子政府推奨暗号リストの改正を考慮して、電子政府推奨暗号に関する調査・検討を行う。監視活動は、情報収集、情報分析、

審議及び決定の 3 つのフェーズからなる。

暗号技術検討会における電子政府推奨暗号の監視に関する基本的考え方は以下の通りである。

(1) 実運用環境において安全性に問題が認められた電子政府推奨暗号は原則としてリストから削除する。
(2) 電子政府推奨暗号の仕様変更は認めない。
(3) 電子政府推奨暗号の仕様変更に到らないパラメータの修正等の簡易な修正を行うことにより当該暗号の安全性が維持される場合には、修正情報を周知して当該暗号をリストに残す。

電子政府推奨暗号の削除等の手順



平成17年度は、ハッシュ関数の危殆化が危惧されることからハッシュ関数の安全性について再検討し、電子署名法の指針の改訂および電子政府推奨暗号リストの見直しに向けた活動を重点に実施する。

第 2 章 監視活動

2.1 概要

平成 17 年度は、SHA-1 の危殆化が顕在化してきたため、新規に署名・認証技術調査ワーキンググループを組織した。また、平成 16 年度の暗号利用モード調査ワーキンググループは、ハッシュ関数の暗号学的な安全性評価を活動に追加し、ハッシュ関数・暗号利用モード調査ワーキンググループと名称変更した。擬似乱数生成系調査ワーキンググループは昨年度の活動を継続した。各ワーキンググループ（WG）が活動した主要活動項目は、表 2.1 の通りである。

表 2.1　平成 17 年度の主要活動項目

ワーキンググループ名	主査	主要活動項目
署名・認証技術調査 WG	松本勉	①電子署名に関する技術的意見の提出 ②ハッシュ関数の安全性評価に関する情報発信方法の検討
ハッシュ関数・暗号利用モード調査 WG	古原和邦	①Wang 等の SHA-1 への攻撃の拡張性に関する検討 ②SHA-256、SHA-384、SHA-512 等の安全性の検討 ③暗号利用モード及びメッセージ認証の評価／標準化動向に関する調査
擬似乱数生成系調査 WG	金子敏信	擬似乱数生成系検定のためのミニマムセットの確定

2.2 監視活動報告

2.2.1 ハッシュ関数の安全性評価について

平成 16 年度の報告時点では、収集した全ての情報が、「情報収集」、「情報分析」フェーズに留まり、「審議及び決定」には至らず、電子政府推奨暗号の安全性に懸念を持たせるような事態は生じていないと判断した。

平成 17 年度はハッシュ関数の安全性評価に表 2.3 に示す進展が見られた。その安全性評価成果を踏まえて、第 2 回暗号技術監視委員会（平成 18 年 3 月 8 日）において、ハッシュ関数の安全性に関する情報発信の方法についても、表 3.1 に示す電子政府推奨暗号リスト

の注釈修正についての素案及び表 2.2 に示すコメント（案）の両案について検討されたが、最終案については、関連情報が整った段階で決定することとなった。

表 2.2　SHA-1 の安全性に関するコメント（案）

SHA-1 の安全性に関する見解（案）

平成 18 年 xx 月 xx 日
暗号技術監視委員会

電子政府における情報セキュリティ確保のために、各府省の情報システム構築において暗号を利用する場合には、「各府省の情報システム調達における暗号の利用方針」（平成 15 年 2 月 28 日 行政情報システム関係課長連絡会議了承）において、必要とされる安全性・信頼性などに応じ、可能な限り、「電子政府推奨暗号リスト」に掲載された暗号の利用を推進するものとされている。

また、情報セキュリティ政策会議から出された「政府機関の情報セキュリティ対策のための統一基準（2005 年 12 月版（全体版初版））」（平成 17 年 12 月 13 日）においても、新規（更新を含む。）に暗号化又は電子署名の付与のアルゴリズムを導入する場合には、電子政府推奨暗号リストの中から選択することが基本遵守事項として明記されている。

電子政府推奨暗号リストでは、ハッシュ関数の SHA-1 は注釈において、『（注 6）新たな電子政府用システムを構築する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。』　と規定している。

SHA-1 については、最近の研究動向によれば、Wang 等により 2^{69} 回以下の SHA-1 の実行回数で同じハッシュ値を持つ 2 つのメッセージが発見できる衝突探索攻撃アルゴリズムが発表され CRYPTREC で検証した結果、2^{69} 回の SHA-1 の実行回数で衝突発見できることの妥当性は検証された。また、近い将来に 2^{63} 回以下の SHA-1 の実行回数で衝突発見できることも妥当性があるとの結論を得た。このことは、SHA-1 を長期間にわたって利用する電子署名やタイムスタンプなどは、近い将来に SHA-1 の衝突発見が現実的な問題に発展する可能性を示唆している。このようなことから、電子署名やタイムスタンプのように長期間にわたって利用するシステムでは、新規（更新を含む）に暗号化又は電子署名の付与のアルゴリズムを導入する場合には、256 ビット以上のハッシュ関数の使用を薦める。

＊参照：CRYPTREC Report 2005「暗号技術監視委員会報告」
http://cryptrec.jp/

以下に、ハッシュ関数の安全性評価の検討内容および見解を示す。

2.2.1.1 SHA-1 の安全性評価について

SHA-1 に対する攻撃については、Wang の 2^{69} 回の SHA-1 実行回数の計算量による攻撃アルゴリズムの概略が CRYPTO 2005 に先駆けて Eurocrypt 2005 のランプセッションと ECRYPT on Hash Function において発表された。これ以外にも Biham, Joux なども SHA-1 の攻撃結果を発表している。CRYPTO 2005 では、Wang の攻撃アルゴリズムが正式に発表されたが、同時にランプセッションで計算量が 2^{63} まで削減できるという発表があった。

平成 17 年 10 月 31 日-11 月 1 日に開催された NIST(National Institute of Standards and Technology：(米国)国立標準技術研究所) の第 1 回ハッシュワークショップでは SHA-1、SHA-256 に関する技術面での現状把握があった。そこで、SHA-1 については平成 22 年に FIPS (Federal Information Processing Standards：(米国)連邦情報処理規格) から外すことが公表され、当面の間は SHA-256 は安全であるという認識が確認された。次世代ハッシュ関数(SHA-256 の後継)については、その選定方法から議論がされているが、公募/選定の実施を含めて結論に至っていない。NIST は第 2 回ハッシュワークショップを平成 18 年 8 月 24、25 日に予定している。

このことから、平成 17 年度は、ハッシュ関数・暗号利用モード調査ワーキンググループにおいて、電子政府推奨ハッシュ関数の安全性について分析・評価を実施した。表 2.3 に評価結果を示す。

表 2.3　平成 17 年度のハッシュ関数評価結果

ハッシュ関数	安全性評価
SHA-1	衝突発見困難性に対して、2^{69} 回以下の SHA-1 の実行回数で攻撃できる手法が発見された。ただし、公開された攻撃アルゴリズムには一部不明な点があり、第三者によって実装して検証されたわけではない。しかし、この攻撃アルゴリズムの不明な点は近い将来に明らかになり第三者による実装が可能になると予想されるので、本攻撃アルゴリズムは極めて大きな脅威になると考えられる。 第二原像計算困難性に対しては、2^{60} バイトのメッセージに対して 2^{106} の SHA-1 の実行回数で攻撃される手法が公開されたが、平成 18 年 2 月の時点で脅威とは言えない。
RIPEMD-160	RIPEMD-160 は異なる二つのブロック暗号 L、R で構成され、そのうちブロック暗号 L については SHA-1 と同程度の差分パ

	スの存在が予想できることが報告されている。これはデータ撹拌においてメッセージ置換とステップ依存のビットシフトの採用など SHA-1 と類似した関数を採用しているためであり、具体的なパスの発見など安全性に関する報告はないが、今後の研究の進展を考え研究動向について非常に注意する必要がある。
SHA-256/-384/-512	実用的な安全性を脅かす攻撃方法が報告されていないため、これらのハッシュ関数は暗号の応用分野で使うのに十分安全であると考えられる。
Whirlpool	Whirlpool 全体では差分の拡散が十分であり、近年の攻撃手法を適用しても衝突発見は困難である。

情報発信について

署名・認証技術調査ワーキンググループでは、表 2.3 に示す評価結果を受けて、ハッシュ関数の安全性に関する技術的な情報を正式コメントとして公表する方法を検討し、表 3.1、表 3.2 に示す提案を行い、暗号技術監視委員会に報告された。

なお、NIST は平成 18 年 3 月 15 日に Web サイトに SHA-1 の利用について、表 2.4 に示す声明を発表した。(http://csrc.nist.gov/CryptoToolkit/tkhash.html)

表 2.4　NIST のハッシュ関数に関する声明文

March 15, 2006:
The SHA-2 family of hash functions (i.e., SHA-224, SHA-256, SHA-384 and SHA-512) may be used by Federal agencies for all applications using secure hash algorithms.
Federal agencies should stop using SHA-1 for digital signatures, digital time stamping and other applications that require collision resistance as soon as practical, and must use the SHA-2 family of hash functions for these applications after 2010.
After 2010, Federal agencies may use SHA-1 only for the following applications: hash-based message authentication codes (HMACs); 0key derivation functions (KDFs);and random number generators (RNGs).
Regardless of use, NIST encourages application and protocol designers to use the SHA-2 family of hash functions for all new applications and protocols.

March 15, 2006:

安全なハッシュアルゴリズムを使う全てのアプリケーションに対して SHA-2 ファミリー（SHA-224, SHA-256, SHA-384, 及び SHA-512）のハッシュ関数を連邦政府機関が利用することは許される。連邦政府機関は、電子署名、タイムスタンプおよびハッシュ関数の衝突発見困難性を安全性の基礎とするその他の用途においては、実施上できるだけ早期に SHA-1 を使うことを止めるべきであり、2010 年以降はそれらのアプリケーションに対して SHA-2 ファミリーのハッシュ関数を使わなければならない。

2010 年以降は、連邦政府機関は次のアプリケーションに対してだけ SHA-1 を使うことが許される。

・ハッシュベースメッセージ認証コード（HMACs）
・鍵導出関数（KDFs）
・擬似乱数生成系（RNGs）

利用目的に係わらず、NIST はアプリケーションやプロトコルの設計者に全ての新しいアプリケーションとプロトコルに対して SHA-2 ファミリーのハッシュ関数を使うよう勧める。

2.2.1.2 MD5 に対する見解について

MD5 に対する攻撃については実時間で衝突の発見が可能な状況にあり、それを使った X.509 電子証明書における不正やポストスクリプトなどのページ記述言語における問題など、実システムやアプリケーションのリスクが明らかになってきている。これらの情報収集分析の結果、MD5 に関する対応を暗号技術監視委員会で議論し平成 17 年 8 月 5 日に表 2.5 に示す「MD5 等に対する見解」として了承された。これは、平成 17 年 8 月 29 日に暗号技術検討会事務局より各府省に通知された。

表 2.5 MD5 等に関する見解

MD5 等に関する見解

平成 17 年 8 月 5 日
暗号技術監視委員会

電子政府における情報セキュリティ確保のために、各府省の情報システム構築において暗号を利用する場合には、「各府省の情報システム調達における暗号の利用方針」（平成 15 年 2 月 28 日 行政情報システム関係課長連絡会議了承）において、必要とされる安全性・信頼性などに応じ、可能な限り、「電子政府推奨暗号リスト」に掲載された暗号の利用を推進するものとされている。

ハッシュ関数の MD5 並びに共通鍵暗号の DES(鍵長 56 ビット)、RC4 及び RC2（鍵長 128 ビット未満）は、暗号技術評価報告書（2002 年度版）＊に記述されている評価基準を満たしていない。このため、電子政府推奨暗号リストに掲載されていないこれら暗号の使用は薦められない。

なお、MD5 については、最近の研究動向によれば、MD5 に対する攻撃アルゴリズムが発表され、現実的な時間内に衝突が発見できる状況になった。さらに、MD5 を発行者署名に利用した場合に、公開鍵情報だけ（例えば RSA 暗号におけるモジュラス）が異なる 2 つの X.509 電子証明書を作成する不正行為が報告されており、衝突発見が現実的な問題に発展することを示唆している。このような不正行為以外にも現実的な問題に発展する危険性が指摘されていることから、認証局の自己署名証明書のフィンガープリントにおける MD5 の使用は薦められない。

＊参照：暗号技術評価報告書（2002 年度）
http://www2.nict.go.jp/y/y213/cryptrec_publicity/c02_report.pdf

2.2.2 電子署名に関する技術的意見について

現在の電子署名法に基づく特定認証業務に係る電子署名の基準に記されている公開鍵暗号技術ではハッシュ関数として SHA-1 のみが規定されているため、SHA-1 以外のハッシュ関数を利用した電子署名が電子署名法では認められていない。

http://www.soumu.go.jp/joho_tsusin/top/ninshou-law/1-6.pdf
http://www.moj.go.jp/MINJI/minji32-3.html
http://www.meti.go.jp/policy/netsecurity/digitsign_sisin.pdf

そこで、SHA-256、SHA-384、SHA-512、RIPEMD-160 を利用した署名技術の是非について検討し、SHA-256、SHA-384、SHA-512 の 3 つのハッシュ関数を新たに追加すべきとの合意を得た。その内容は表 2.6 に示す新たな改定案を記載した「電子署名法の指針の改訂に係わる意見の提出」として、暗号技術監視委員会承認（平成 17 年 7 月 15 日)、暗号技術検討会承認（平成 17 年 10 月 12 日）を経て政府に提言した。

表 2.6　電子認証業務認定指針第三条及び第十条の改訂案

(1) 告示第三条（特定認証業務に係る電子署名の基準）

一 RSA 方式（オブジェクト識別子 1.2.840.113549.1.1.5 又 1.2.840.113549.1.1.11 又は 1.2.840.113549.1.1.12 又は 1.2.840.113549.1.1.13）であって、モジュラスとなる合成数が 1024 ビット以上のもの
二　RSA-PSS 方式（オブジェクト識別子 1.2.840.113549.1.1.10）であって、ハッシュ関数

として SHA 方式（オブジェクト識別子 1.3.14.3.2.26 又は 2.16.840.1.101.3.4.2.1 又は 2.16.840.1.101.3.4.2.2 又は 2.16.840.1.101.3.4.2.3）を使用し、モジュラスとなる合成数が 1024 ビット以上のもの

三　ECDSA 方式（オブジェクト識別子 1.2.840.10045.4.1）であって、楕円曲線の定義体及び位数が 160 ビット以上のもの

四　DSA 方式（オブジェクト識別子 1.2.840.10040.4.3）であって、モジュラスとなる素数が 1024 ビットのもの

(2) 告示第十条第二号（認定認証業務と他の業務との誤認を防止するための措置）

二　発行者署名検証符号に係る電子証明書の値を SHA-1（オブジェクト識別子 1.3.14.3.2.26）又は SHA-256（オブジェクト識別子 2.16.840.1.101.3.4.2.1）又は SHA-384（オブジェクト識別子 2.16.840.1.101.3.4.2.2）又は SHA-512（オブジェクト識別子 2.16.840.1.101.3.4.2.3）で変換した値によって認定認証業務を特定すること。

2.2.3　NIST FIPS46-3 の廃止に伴う T-DES の扱いについて

NIST は平成 17 年 5 月 19 日付けで T-DES を規定した FIPS46-3 を廃止した。これを受けて、それが記載されている電子政府推奨暗号リストの注釈の取扱いについて検討し、「3-key Triple DES に係わる電子政府推奨暗号リストの注釈の一部修正について」を暗号技術監視委員会承認（平成 17 年 9 月 1 日）、暗号技術検討会承認（平成 17 年 10 月 12 日）を経て、行政情報システム関係課長連絡会議の配下の共通システム専門部会(平成 17 年 11 月 17 日)で各府省に周知するとともに、CRYPTERC の Web サイトに表 2.7 に示す内容で公開した。

表 2.7　電子政府推奨暗号リストの注釈の一部修正

3-Key Triple DES に係わる電子政府推奨暗号リストの注釈の一部修正について

平成 17 年 11 月 30 日

暗号技術監視委員会

NIST は2005年5月19日付けで、連邦政府が取り扱う情報の秘匿には Data Encryption Standard (DES) では十分な安全性をもたなくなったとして、Data Encryption Standard (DES) の規定を含んでいる NIST FIPS 46-3 を廃止した[1]。FIPS 46-3 の廃止に伴い, NIST SP 800-67を新たに発行して[2]、Triple Data Encryption Algorithm (TDEA) の規定を FIPS 46-3 からこれに移した。また、NISTは、FIPS 46-3 が再確認された1999年10月25日以来 Single DES (= DES) から Triple DES (= TDEA) (及び AES)への移行を推奨してきている。

一方、電子政府推奨暗号リスト（平成15年2月20日）では、3-key Triple DES[3]に対して

次のように定めている。

（注3）新たな電子政府用システムを構築する場合は、より長いブロック長の暗号が使用できるのであれば、128ビットブロック暗号を選択することが望ましい。

（注4）3-key Triple DESは、以下の条件を考慮し、当面の使用を認める。

1）FIPS 46-3として規定されていること

2）デファクトスタンダードとしての位置を保っていること

今回のNISTの対応は、電子政府推奨暗号リストの考え方とは基本的に矛盾していないことから、暗号技術監視委員会（平成17年6月20日）および暗号技術検討会（平成17年10月12日）の意見にもとづき、リストそのものの変更は行わず、注釈の注4）において、

（修正前）1）FIPS 46-3として規定されていること

（修正後）1）SP800-67として規定されていること

の修正のみとして、下表を電子政府推奨暗号リストの末尾に添付する。

電子政府推奨暗号リストに関する修正情報				
修正日付	修正箇所	修正前	修正後	修正理由
平成17年10月12日	注釈の注 4)の1)	FIPS 46-3 として規定されていること	SP800-67 として規定されていること	仕様変更を伴わない、仕様書の指定先の変更

[1] http://csrc.nist.gov/publications/fips/05-9945-DES-Withdrawl.pdf

[2] http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf

[3] Triple Data Encryption Algorithm（TDEA）の規定には最初の2つの鍵を独立に取り、3つ目の鍵を1つ目の鍵と等しく取るオプションも含まれているが、CRYPTRECにおいては3つの鍵すべてを独立に取るオプションのみを3-key Triple DES と呼んでいる。

2.2.4 擬似乱数検定に関するミニマムセットの作成について

擬似乱数検定のためのミニマムセットとして、NIST の SP800-22 の 16 種類の検定法の中から表 2.8 に示す 14 種類をミニマムセットとして採択し、仕様書（付録 6）を作成し、暗号モジュール委員会事務局に送付した。

表 2.8　擬似乱数検定のミニマムセット

項番	検定の名称
1	頻度検定
2	ブロック単位の頻度検定

3	連検定
4	ブロック単位の最長連検定
5	2 値行列ランク検定
6	重なりの無いテンプレート適合検定
7	重なりのあるテンプレート適合検定
8	Maurer の「ユニバーサル統計量」
9	線形複雑度検定
10	系列頻度検定
11	累積和検定
12	ランダム回遊検定
13	変形ランダム回遊検定
14	近似エントロピー検定

2.2.5 NIST の暗号技術標準化動向

平成 17 年から平成 18 年 3 月にかけて NIST が発刊した暗号技術標準化に関連するドキュメントを表 2.9 に示す。

表 2.9 最近の NIST の暗号技術標準化活動

分類	時期	ドキュメント名
Draft Publications	March 13, 2006	Draft FIPS186-3 : Digital Signature Standard (DSS)
	March 13, 2006	Draft NIST SP800-89 : Recommendation for Obtaining Assurances for Digital Signature Applications
	December 16, 2005	Draft SP800-90 Recommendation for Random Number Generation Using Deterministic Random Bit Generators
Special Publications	March 2006	SP800-73 Revision 1 : Interfaces for Personal Identity Verification
	March 2006	SP800-56A : Recommendation for Pair-wise Key Establishment Schemes Using Discrete Logarithm cryptography
	December 2005	SP800-21-1 : Second Edition, Guideline for Implemeting Cryptography in the Federal Government

	August 2005	SP800-57 : Recommendation on Key Management
	May 2005	SP800-38B : Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
	April 2005	SP800-78 : Cryptographic Algorithms and Key Sizes for Personal Identity Verification
FIPS Pubs	March 2006	FIPS 201-1 : Personal Identity Verification (PIV) of Federal Employees and Contractors
	March 2006	FIPS 200 : Minimum Security for Federal Information and Information Systems
	Withdrawn May 19, 2005	FIPS 46-3 : Data Encryption Standard (DES), specifies the use of Triple DES

これらの活動の中で、特記すべき事項は、以下の通りである。

1) NIST は平成17年5月19日付けで、米国連邦政府が取り扱う情報の秘匿には Data Encryption Standard (DES) では十分な安全性をもたなくなったとして、DESの規定を含んでいる FIPS 46-3 を廃止した。FIPS 46-3 の廃止に伴い, Triple DES(TDES)の規定は、FIPS 46-3 から NIST SP 800-67 (Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, May 2004) に移した。
2) 平成17年10月31日-11月1日、NIST主催の第1回ハッシュワークショップが開催され、ハッシュ関数に関する今後の取り扱い方が議論された。第2回ハッシュワークショップは、CRYPTO 2006 の直後(平成18年8月24－25日)に開催される予定で CFP も公開され、一般に投稿論文を募集している。
3) 平成18年3月13日付けで、Draft FIPS 186-3、Draft NIST SP800-89がパブリッコメント発表された。これまでのFIPS 186-2で規定していたDSA, RSA, ECRSAの暗号アルゴリズムの安全性強化のための改定版である。パブリックコメント期限はNIST SP800-89が平成18年4月28日、FIPS 186-3が平成18年6月12日である。
4) 平成18年3月、FIPS 201-1 (Personal Identity Verification (PIV) of Federal Employees and Contractors) が発刊され、連邦政府関係者のPIVの暗号が規定された。

2.2.6 ECRYPT の暗号技術標準化動向

ECRYT では、現在、ストリーム暗号について評価を実施している。以下に状況を示す。

平成16年11月　募集開始
平成17年 4月　募集締め切り
平成17年 5月　ワークショップSKEW開催
平成18年 2月　フェーズ1終了 （ SASC 2006 Workshop（平成18年2月2―3 日）終了）
平成18年 7月　フェーズ2開始
平成19年 9月　フェーズ2終了
平成20年 1月　ファイナルレポート

表 2.10　提案暗号（発表済み分）

暗号名	提案者	提案国
Salsa20	Daniel J. Bernstein	米
Rabbit	Martin Boesgaard, Mette Vesterager, Thomas Christensen, and Erik Zenner	デンマーク
SOSEMANUK	. Berbain, O. Billet, A. Canteaut, N. Courtois, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin, and H. Sibert	仏
Mir-1	Alexander Maximov	ロシア、現独
Phelix	Doug Whiting, Bruce Schneier, Stefan Lucks, and Frederic Muller	米、独、仏
TSC-3	Jin Hong, Dong Hoon Lee, Yongjin Yeom, Daewan Han, and Seongtaek Chee	韓国
F-FCSR-8, F-FCSR-H	F. Arnault, T. Berger, and C. Lauradoux	仏
Achterbahn	Berndt M. Gammel, Rainer Göttfert, and Oliver Kniffler	独
SFINKS	An Braeken, Joseph Lano, Nele Mentens, Bart Preneel, and Ingrid Verbauwhede	ベルギー
WG	Yassir Nawaz and Guang Gong	カナダ、シンガポール
Py (Roo)	Eli Biham and Jennifer Seberry	イスラエル、オーストラリア
Mosquito	Joan Daemen and Paris Kitsos	ベルギー
Polar Bear	Johan Håstad and Mats Näslund	スウェーデン
Edon80	D. Gligoroski, S. Markovski,L. Kocarev, and M. Gusev	マケドニア
CJCSG	Cees J.A. Jansen, Tor Helleseth, and Alexander Kholosha	ノルウェー

DECIM	C. Berbain, O. Billet, A. Canteaut, N. Courtois, B. Debraize, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin, and H. Sibert	仏
MICKEY, MICKEY-128	Steve Babbage and Matthew Dodd	英国
YAMB	Anatoly Lebedev, Alexander Ivanov, Sergey Starodubtzev, and Alexey Volchkov	ロシア
LEX	Alex Biryukov	イスラエル、現ベルギー
Fubuki	Makoto Matsumoto, Takuji Nishimura, Mariko Hagita, and Mutsuo Saito	日本、広島大学、慶應大学、御茶ノ水女子大学
ABC	Vladimir Anashin, Andrey Bogdanov, Ilya Kizhvatov, and Andrey Bogdanov	ロシア

2.2.7 IETF の暗号技術標準化動向

IETF ではセキュリティ関連技術として、公開鍵暗号アルゴリズム、共通鍵暗号アルゴリズム、ハッシュ関数、HMAC、IPSec,DNSSec、Kerberos、OpenPGP、PKI X.509、Secure Shell、暗号プロトコルなど幅広い範囲の標準化を行っている。ハッシュ関数を利用したプロトコルに関する標準化は、The IEEE Security Area 配下に以下のワーキンググループを配置して行っている。

- Open Security Area Drectorate (saag)
- IKEv2 Mobility and Multihoming (mobike)
- Profiling Use of PKI in IPSEC (pki4ipsec)
- Public-Key Infrastructure (X.509) (pkix)
- S/MIME Mail Security (smime)
- Transport Layer Security (tls)
- One-way Hash Function (hash)

※ One-way Hash Function ワーキンググループは、63rd Meeting 以降に活動。

1) 62nd IETF Meeting (平成 17 年 3 月 6－11 日)

Saag ワーキンググループにおいて、Eric Rescorla により“Current Status of MD5 and SHA-1”という発表が行われ、ハッシュ関数に関して安全性に影響があるものとして、電子証明書、タイムスタンプ、否認防止 (non-repudiation) が議論された。

2) 63rd IETF Meeting (平成 17 年 7 月 31 日－8 月 5 日)

hash ワーキンググループにおいて、SHA-1 の危殆化に関する発表が行われ、攻撃に対する耐性、影響度、延命策（ハッシュのランダム化、ハッシュ対象メッセージに対する Pre-processing、SHA-256 の 160bit への Truncation など)、および新ハッシュ関数などが議論された。

3) 64th IETF Meeting（平成 17 年 11 月 6－11 日）

saag, pkix, smime, および tls の各ワーキンググループにおいて、ハッシュの危殆化に関する発表が行われ、直前に開催された NIST のハッシュワークショップを受けて、SHA-256 への対応の必要性、SHA-1 をマイナーチェンジした SHA1-IME の使用可能性、ECDSA や DSA といった SHA-2 ファミリーへの未対応のものへの対応方法について議論された。

平成 17 年 11 月に、RFC4270 として、”Attacks on Cryptographic Hashes in Internet Protocols”（インターネットプロトコルにおける暗号技術的ハッシュ関数についての攻撃）というメモが発表された。この RFC では、MD5 や SHA-1 といったハッシュ関数への攻撃の概要、インターネットにおけるハッシュ関数の利用、ハッシュ値の衝突が電子署名や公開鍵証明書の安全性に与える影響について、まとめられている。

2.2.8 ISO/IEC JTC 1/SC 27 の暗号技術標準化動向

1) 暗号アルゴリズム国際規格(18033)

ブロック暗号(18033-3)およびストリーム暗号(18033-4)は、FDIS 投票で賛成が多数となり、7 月 15 日に IS 化された。また、公開鍵暗号(18033-2)は、FDIS 投票の結果が 1 月 31 日付けで発表され、賛成が圧倒的多数であったため IS 化が確実な状況になった。

2) デジタル署名(14888)

総論(14888-1)、素因数分解に基づく機構(14888-2)、離散対数問題に基づく機構（14888-3）の 3 パートから構成されている。デジタル署名規格の再編成が問題になっていたが、ウィーン会合でほぼ解消された。14888-2 は CD（2ndCD）に留まり、14888-3 は FCD に進むことが決議された。また、クアラルンプール会合では、SHA-1 の衝突発見困難性が危うくなったことが議題となり、比較的長い時間を掛けて議論した結果、メカニズム毎にハッシュ関数の要件を注釈（Note）の形で参考情報として記載することで合意した。

3) ハッシュ関数（10118-3）

最近の SHA-1 の安全性低下を示す研究結果に対する SC27 としての対応策を協議し、SHA-1 の安全性に関する情報の募集と SHA-1 に関する SC27 からの声明をホームページに掲載することで合意した。

4) ECRYPT とのリエゾン

Ecrypt (Bart Preneel)からの依頼に基づき、SC27 との間にリエゾンを置くことが承認された。

2.3 学会等参加記録

平成 17 年度は、国内・国外の学会に参加し、暗号解読技術に関する情報収集を実施した。

情報収集の結果、ハッシュ関数に関する安全性の評価の進展を除いては、電子政府推奨暗号の安全性に影響を与える発表等は見られなかった。

監視要員等を派遣した国際会議は、表 2.11 に示すとおりである。

表 2.11 国際会議への参加状況

学会名・会議名		開催国・都市	期間
ISO/IEC	ISO/IEC JTC 1/SC 27/WG 2	Vienna, Austria	2005/4/11 ～ 2005/4/19
EUROCRYPT 2005	Eurocrypt	Aarhus, Denmark	2005/5/23 ～ 2005/5/26
ECRYPT	STVL Workshop	Aarhus, Denmark	2005/5/26 ～ 2005/5/27
ECRYPT	ECRYPT on Hash Function	Krakow, Poland	2005/6/21 ～ 2005/6/22
SAC 2005	Selected Area in Cryptography	Kingston, CANADA	2005/8/11 ～ 2005/8/12
CRYPTO 2005	CRYPTO	Santa Barbara USA	2005/8/14 ～ 2005/8/18
HashWorkshop	NIST Hash Workshop	Washington, USA	2005/10/31～ 2005/11/1
ISO/IEC	ISO/IEC JTC 1/SC 27/WG 2	Kuala Lumpur, Malaysia	2005/11/7 ～ 2005/11/11
Asiacrypt	Asiacrypt	Chennai, India	2005/12/4～ 2005/12/8
Indocrypt	Indocrypt	Bangalole, India	2005/12/10～ 2005/12/12
CANS	The 4th International Conference on Cryptology and Network Security	Xian, China	2005/12/14～ 2005/12/16
CT-RSA	RSA Conference 2006, Cryptographers' Track	SanJose, USA	2006/2/13～ 2006/2/16

また、情報収集を行なった学会等で発表された主要論文を付録 3 に示す。以下に、国際学会等に発表された論文を中心に、暗号解読技術の最新動向について述べる。

2.3.1 ハッシュ関数の解読技術

MD4 に関しては、2^{-2} から 2^{-6} 確率の間の(トータルで 2^{8} operations を超えない)計算量で MD4 の衝突を発見したという発表がなされた[Cryptanalysis of the Hash Functions MD4 and RIPEMD, Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, Xiuyuan Yu, EUROCRYOPT2005]。

また、MD4 に対する第二原像攻撃 (Second-preimage Attack) として、ターゲットとなるメッセージを修正 (modify) してその修正したメッセージの第二原像 (Second-preimage) を求めるという不完全な形での攻撃が発表された [The Second-Preimage Attack on MD4,Hongbo Yu and Gaoli Wang and Guoyan Zhang and Xiaoyun Wang, CANS2005]

MD5 についても衝突攻撃が発表された。この衝突攻撃は、IBM P690 で MD5 の衝突を発見するのに必要な時間が、(M0, M0′)の発見に約 1 時間(最速で 15 分)、(M1, M1′)の発見に 15 秒から 5 分というものである [How to break MD5 and Other Hash Functions, Xiaoyun Wang, Hongbo Yu, EUROCRYPT2005]

SHA-0 については、4 ブロック SHA-0 の衝突攻撃により、80,000 CPU hours で計算でき、具体的に衝突を示された [Collisions of SHA-0 and Reduced SHA-1, Eli Biham, Rafi Chen, Antoione Joux, Patrick Carribault, Christophe Lemuet, William Jalby, EUROCRYPT2005]。その後 この計算量は 2^{39} に改良された [Efficient Collision Search Attacks on SHA-0, Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin, CRYPTO2005]。

SHA-1 については、フルラウンド SHA-1 のディスターバンスベクトルを利用し、24 ステップから 80 ステップまでの近似衝突 (near collision) 差分パスで確率 2^{-68} のものを発見し、その後 1-23 ステップについてディスターバンスベクトルに対応する不可能な差分パスを可能な差分パスに変換する攻撃が発表された。衝突を発見するために必要な計算量は 2^{69} と見積もられている[Finding Collisions in the Full SHA-1, Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu, CRYPTO2005]。

その後、この計算量は 2^{63} にまで改良された[New Collision Search for SHA-1, Xiaoyun Wang, Andrew Yao and Frances Yao, CRYPTO2005]。

SHA-1 について、これらの結果はすぐに安全性に重大な影響を及ぼすということまでには至っていない。

2.3.2 ストリーム暗号の解読技術

ストリーム暗号については、無線システム Bluetooth (ブルートゥース) に用いられている E0 というストリーム暗号の解読について鍵の最初の 24 ビットを $2^{23.8}$ フレームかつ 2^{38} の計算量で推定が可能であるという発表があった[The Conditional Correlation Attack: A Practical Attack on Bluetooth Encryption, Yi Lu, Willi Meier, and Serge Vaudenay, CRYPTO2005]。

また、RC4 を用いた WEP という無線システムのセキュリティ規格に対して、鍵回復攻撃(key recovery attack) が提案され、入力する IV の生成の仕方(master key との関係)により攻撃しやすい場合があり、鍵の長さが伸びたとしてもそれを探索するのにかかる計算量は線形にしか増えず、実用的な攻撃だという発表がなされた。ただし、この攻撃が適用できる場合には制限があり、IV が攻撃に適した生成方法により作られた場合に限られる [A Practical Attack on the Fixed RC4 in the WEP Mode, Itsik Mantin, ASIACRYPT2005]。

他は、目新しいものは発表されていない。

2.3.3 ブロック暗号の解読技術

電子政府推奨暗号の Camellia の解読に関する論文発表があったが、極めて少ない段数での評価であり、Camellia の安全性に直接影響するものではなかった。〔New Observation of Camellia, Duo Lei, Li Chao and Feng Keqin, SAC2005〕

2003 年に AES に対して XSL と呼ばれる代数的攻撃(Algebraic attack) が適用できるのではないかと言う発表があったが、それに対する否定的な結果として、XSL において中間値の消去を考えた場合に、効率的な攻撃が困難であることを示された。[An Analysis of the XSL Algorithm, Carlos Cid, and Gaetan Leurent, ASIACRYPT2005]。

3GPP の標準アルゴリズムとなっている KASUMI に対して、関連鍵矩形攻撃 (related key rectangle attack) を適用することにより解読可能であるとする発表があったが、攻撃手法の前提に強い仮定があるため、実質的な利用については本攻撃の影響はほとんど無いと思われる。[A Related-Key Rectangle Attack on the Full KASUMI, Eli Biham, Orr Dunkelman, Nathan Keller, ASIACRYPT2005]

2.3.4 公開鍵暗号の解読技術

RSA 暗号アルゴリズムに対して、Wiener 攻撃(秘密鍵 d のサイズが $N^{0.25}$ 以下の場合に多項式時間で d を求めるアルゴリズムが存在する)と 98 年に Boneh-Durfee-Frankel により提案されたサイドチャネル攻撃の手法とを組み合わせた攻撃の論文が発表された。[Partial Key Exposure Attacks on RSA up to Full Size Exponents, Matthias Ernst, Ellen Jochemsz, Alexander May, Benne de Weger, EUROCRYPT 2005]

また、RSA 暗号アルゴリズムに関する解析結果を示した論文が発表された。$ed \equiv 1(\mathrm{mod}\, lcm(p-1, q-1))$ なる d が小さい場合に想定される攻撃として、Continue fraction attack, lattice-based attack, factoring attack 等の様々な攻撃について解析し、現実的な範囲でのパラメータ評価を行った結果が示された。[Another Look at Small RSA Exponents, M. Jason Hinek, CT-RSA 2006]

他、現在の電子政府推奨暗号リスト公開鍵暗号の安全性を急激に減少させる程の新しい解読技術は発表されていない。

また、最近の発表論文の傾向としては、従来一般的に安全性の証明モデルとして用いら

れていたランダムオラクルモデルとは異なる新しい安全性の証明モデルに関する研究をはじめとし、ランダムオラクルモデルに基づかない安全性証明による方式の提案・既にランダムオラクルモデルで安全性証明が示されている方式についてランダムオラクルの仮定を取り除いた場合の安全性解析等、”ハッシュ関数の出力≠ランダムオラクルの出力”の流れにのる発表が目立ちはじめている。

その他、アルゴリズムが仮定としている問題の解析等の研究も進展している。

国内では、ID ベース暗号やそれを利用したプロトコル・安全性モデル・困難性を仮定している問題の解析・実装技術及び実装解析・運用なども加味した上位アプリケーションなど研究のターゲットが多岐に渡っている。

2.4 委員会開催記録

平成 17 年度、暗号技術監視委員会は、表 2.12 の通り 2 回開催された。暗号技術調査ワーキンググループは、表 2.13 の通り計 12 回開催された。各会合の開催日及び主な議題は以下の通りである。

(1) 暗号技術監視委員会

表 2.12　暗号技術監視委員会の開催

回	年月日	議題
第 1 回	平成 17 年 6 月 20 日	活動方針確認、監視状況報告
第 2 回	平成 18 年 3 月 8 日	監視状況報告、CRYPTRYC report 2005 審議

(2) 暗号技術調査ワーキンググループ

表 2.13　暗号技術調査ワーキンググループの開催

回	年月日	議題
第 1 回	平成 17 年 4 月 25 日	第 1 回署名・認証技術調査WG
第 2 回	平成 17 年 6 月 8 日	第 2 回署名・認証技術調査WG
第 3 回	平成 17 年 7 月 6 日	第 3 回署名・認証技術調査WG
第 4 回	平成 17 年 7 月 28 日	第 1 回ハッシュ関数・暗号利用モード調査WG
第 5 回	平成 17 年 8 月 11 日	第 1 回擬似乱数生成系調査WG
第 6 回	平成 17 年 9 月 13 日	第 2 回ハッシュ関数・暗号利用モード調査WG
第 7 回	平成 17 年 9 月 22 日	第 2 回擬似乱数生成系調査WG
第 8 回	平成 17 年 11 月 25 日	第 3 回ハッシュ関数・暗号利用モード調査WG
第 9 回	平成 17 年 12 月 27 日	第 4 回署名・認証技術調査WG
第 10 回	平成 18 年 2 月 2 日	第 4 回ハッシュ関数・暗号利用モード調査WG
第 11 回	平成 18 年 2 月 24 日	第 5 回ハッシュ関数・暗号利用モード調査WG
第 12 回	平成 18 年 2 月 24 日	第 5 回署名・認証技術調査WG

第 3 章 暗号技術調査ワーキンググループ

3.1 署名・認証技術調査ワーキンググループ

3.1.1 活動目的

ハッシュ関数に関する最近の研究結果が、電子政府及びシステム等におけるハッシュ関数の利用に及ぼす影響について調査する。特に、電子署名において、SHA-256、SHA-384、SHA-512 を利用できるように、これらのハッシュ関数の利用について調査する。主な検討項目は以下である。

1) 電子署名法に基づく特定認証業務に係る電子署名の基準に記されている署名技術に関するハッシュ関数の利用についての調査と技術的意見の提出
2) MD5 の利用についての問題点の提出
3) ハッシュ関数の安全性評価に関する情報発信の方法についての検討

3.1.2 委員構成

主査： 松本 勉　　横浜国立大学　大学院　環境情報研究院　教授
委員： 太田和夫　　電気通信大学　電気通信学部　情報通信工学科　教授
委員： 小暮 淳　　株式会社富士通研究所　IT コア研究所　主任研究員
委員： 洲崎誠一　　株式会社日立製作所　システム開発研究所　主任研究員
委員： 渡辺 創　　産業技術総合研究所　副研究センター長

3.1.3 活動概要

3.1.3.1 電子署名に関する技術的意見の提出

電子署名法に基づく特定認証業務に係る電子署名の基準に記されている公開鍵暗号技術で利用されているハッシュ関数として SHA-1 のみが規定されているため、SHA-1 以外のハッシュ関数を利用した電子署名が電子署名法では認められていない。そこで、SHA-256、SHA-384、SHA-512、RIPEMD-160 を利用した署名技術について検討し、電子署名に関する技術的意見を提言することが必要となっている。

SHA-256、SHA-384、SHA-512、RIPEMD-160 を利用した電子署名についての考え方を整理した上で、電子署名の指針の改訂に係わる意見を、「電子署名法の指針の改訂に係わる意見の提出」としてまとめた。

主な検討項目は以下であった。

1) 電子署名の安全性が依存するアルゴリズム問題のパラメータサイズについては、このまま維持する（規則第二条の第一,二,三号、告示第三条)。
2) 電子政府推奨暗号リスト（平成 15 年 2 月 20 日）の注釈（注 6）において明記されているように、256 ビット以上のハッシュ関数を推奨しているので、RIPEMD-160 と SHA-224 の 2 つは追加しない。
3) RSASSA-PKCS1-v1_5 方式及び RSASSA-PSS 方式[1]に関しては、ハッシュ関数 SHA-256、SHA-384、SHA-512 の 3 つを追加する。
4) ECDSA 方式に関しては、平成 17 年 2 月 18 日時点でドラフト版である署名アルゴリズムの仕様書（SEC 1 v1.5 Working draft や Draft ANSI X9.62-2005）が正式版になり次第、ハッシュ関数 SHA-256、SHA-384、SHA-512 の 3 つの追加の検討をするが、セキュリティパラメータとの整合性を確認する必要はある。
5) DSA 方式に関しては変更なし。
6) 告示第十条第二号（認定認証業務と他の業務との誤認を防止するための措置）に関しては、ハッシュ関数 SHA-256、SHA-384、SHA-512 の 3 つを追加する。

平成 17 年 7 月 15 日付けで暗号技術監視委員会にて承認され、平成 17 年 10 月 12 日暗号技術検討会へ提出されている。

3.1.3.2 MD5 の利用についての問題点の提出

MD5 の衝突を発見することが容易な状況、及び、MD5の危殆化がシステムに及ぼす影響について、これまでに明らかにされてきている問題点をまとめた。

調査によって、不正行為以外にも現実的な問題に発展する危険性が示されていることから、暗号技術監視委員会の了承のもと、「MD5等に対する見解」を平成17年8月29日に暗号技術検討会事務局より各府省に通知した。

MD5については、以下のような問題点が知られている。

1) X.509 電子証明書における衝突

A. Lenstra, X. Wang and B. de Weger, “Colliding X.509 Certificates,” Cryptology ePrint Archive, Report 2005/067, Available at http://eprint.iacr.org/2005/067, March 1, 2005.

2) ポストスクリプトなどのページ記述言語における不正

S.Lucks, M.Daum,

http://th.informatik.uni-mannheim.de/people/lucks/HashCollisions/

[1] IETF RFC4055 において、SHA-1 以外の SHA の利用に係わる RSASSA-PSS に関する識別子が明確になった(平成 17 年 6 月 21 日)。

3.1.3.3 ハッシュ関数の安全性評価と情報発信方法の検討

平成 17 年度、ハッシュ関数・暗号利用モード調査ワーキンググループでは、表 3.1 のようにハッシュ関数に関する安全性評価を実施した。

署名・認証技術調査ワーキンググループではこの評価結果を受けて、ハッシュ関数の安全性に関する技術的な情報を正式コメントとして公表する方法として、電子政府推奨暗号リスト（平成 15 年 2 月 20 日）において、SHA-1 及び RIPEMD-160 に付属している注釈を表 3.1 のように修正すべきであると判断した。主な理由は以下の通り。

1) 電子政府推奨暗号リスト及びその注釈が、CRYPTREC と外部との間の統一したインターフェイスとしての正式な文書であるから、注釈の修正をすることが適当である。
2) 存の注釈においても、256 ビット以上が望ましいとあり、問題意識は既に表明されているものの、ハッシュ関数の危殆化が進んでいるという事実を新たに、外部に公表する必要性がある。

表 3.1 注釈修正についての素案

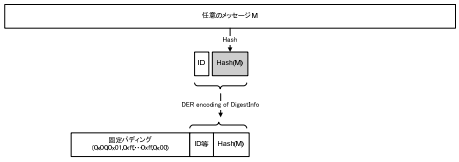
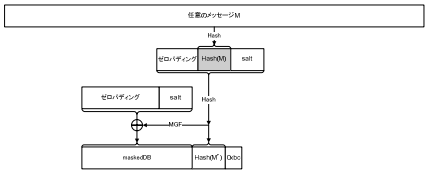
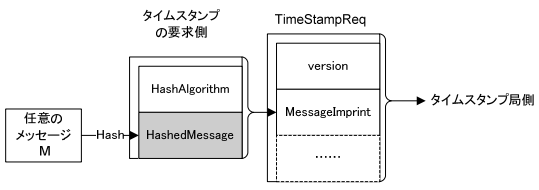
<table>
<tr><td colspan="3">電子政府推奨暗号リスト（総務省・経済省、平成 15 年 2 月 20 日）の抜粋</td></tr>
<tr><td rowspan="5">ハッシュ関数</td><td colspan="2">RIPEMD-160(注 6)</td></tr>
<tr><td colspan="2">SHA-1(注 6)</td></tr>
<tr><td colspan="2">SHA-256</td></tr>
<tr><td colspan="2">SHA-384</td></tr>
<tr><td colspan="2">SHA-512</td></tr>
<tr><td>現在</td><td colspan="2">（注 6）新たな電子政府用システムを構築する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。</td></tr>
<tr><td>素案※</td><td colspan="2">（注 6）新たに電子政府用システムを構築または更改する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。特に、ハッシュ関数の衝突発見困難性[2]を安全性の基礎とする用途においては、256 ビット以上のハッシュ関数を選択するべきである。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。</td></tr>
</table>

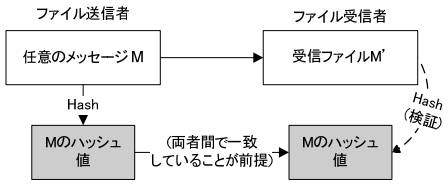
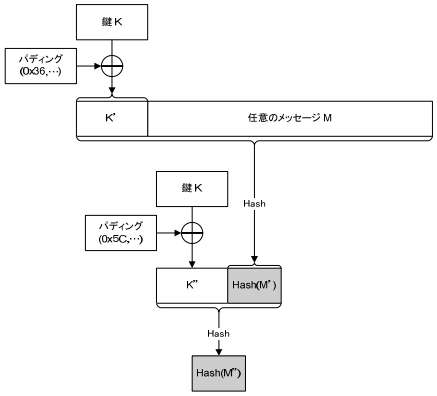
※ 表中の下線部が修正部分に相当する。

[2] 衝突発見困難性とは、ハッシュ値が一致するような異なる 2 つのメッセージを見つけることが計算量的に困難であることをいう(ハッシュ値は事前に与えられていない)。第二原像計算困難性とは、ある既知のメッセージとそのハッシュ値が与えられた時、ハッシュ値が与えられた値と一致するような別のメッセージを見つけることが計算量的に困難であることをいう。原像計算困難性とは、ある未知のメッセージのハッシュ値が与えられた時、ハッシュ値が与えられた値と一致するようなメッセージを見つけることが計算量的に困難であることをいう。

また、どのようなシステムであるとハッシュ関数の衝突発見からの影響を受けるのかについて、表 3.2 に示すようにハッシュ関数の利用形態別で分類した。

表 3.2　ハッシュ関数の利用に係わる類型化

分類	ハッシュ関数の安全性（擬似ランダム性を除く）	SHA-1 の利用に関する評価（2005 年度）	具体例
電子署名における入力メッセージのハッシュ値： 以下は、その例示である(署名生成における最初の段階)。 ・RSASSA-PKCS1-v1_5 の EMSA-PKCS1-v1_5  ・RSASSA-PSS の EMSA-PSS  	衝突発見困難性	衝突発見困難性については、2^{69} 回以下の SHA-1 の実行回数で攻撃できる手法が発見されている。	X.509 電子証明書(tbsCertificate 部分が入力メッセージにあたる)、 その他、X.509 形式を用いない電子署名全般
データやファイルのフィンガープリント (システム側にはハッシュ関数(のアルゴリズム識別子)とハッシュ値のみが明らかにされ、ユーザー側にのみハッシュ値の元となるメッセージが保持されている場合)： 以下は、その例示である。 ・タイムスタンプ要求における要求側のメッセージフォーマット  	衝突発見困難性	衝突発見困難性については、2^{69} 回以下の SHA-1 の実行回数で攻撃できる手法が発見されている。	タイムスタンプサービスにおけるタイムスタンプ要求(メッセージのハッシュ値が MessageImprint フィールドに格納される)
データやファイルのフィンガープリント (ハッシュ値の元となるメッセージ及びハッシュ値の両方が明らかにされている場合)： 以下は、その例示である。	第二原像計算困難性	第二原像計算困難性については、2^{60} バイトのメッセージに対して	データやファイルの暗号学的チェックサム

・通信における受信ファイルの検証  		2106 の SHA-1 の実行回数で攻撃できる手法が公開されたが、2006 年 2 月の時点で脅威とは言えない。	
HMAC(IETF における RFC 2104 等で定められているもの) :  	衝突発見困難性	初期値または入力の一部を秘密かつランダムとした場合の擬似ランダム性、衝突発見困難性が保証されていれば、SHA-1 を使った HMAC は安全である。	データやファイルの暗号学的チェックサム、暗号プロトコル(SSL3.0/TLS1.0, IPsec 等)の暗号学的チェックサム
暗号プロトコルや暗号アルゴリズムにおける補助関数など(鍵導出関数、マスク生成関数、擬似乱数生成系)	原像計算困難性、衝突発見困難性	特に問題は見つかっていない。	電子署名におけるエンコーディングメソッドの補助関数、公開鍵暗号や秘密鍵暗号、MAC のための鍵や初期値、salt 等の生成時の補助関数

3.2 ハッシュ関数・暗号利用モード調査ワーキンググループ

3.2.1 調査背景

平成 16 年度、暗号利用モード調査ワーキンググループでは、米国を中心とする暗号利用モード標準の見直しの流れのもと、ブロック暗号を用いた暗号利用モードの調査研究に注力した。平成 17 年度は、暗号利用モード標準化の動きが一段落したこと、ハッシュ関数の衝突解析に進展があり、現在利用されているハッシュ関数に対して脅威が生じる可能性が出てきたことから、電子政府推奨暗号の監視という観点より、ハッシュ関数の安全性に関

する調査を行うこととした。平成17年度の目的は、電子政府推奨暗号リストに掲載されているハッシュ関数について最近提案されている攻撃方法の適用可能性を調査し安全性評価の再検討を行うと共に、主に米国で標準化が進んでいる暗号利用モードやメッセージ認証について調査を行い暗号技術監視委員会に報告することである。ハッシュ関数の安全性の検討に関しては署名・認証技術調査ワーキンググループと連携して評価項目や視点について吟味する。

ハッシュ関数の構成法としては、ブロック暗号に基づくもの(ISO/IEC 10118-2)、専用ハッシュ関数と呼ばれるもの(ISO/IEC 10118-3, FIPS180-2)、剰余演算を用いるもの(ISO/IEC 10118-4)などがあるが、電子政府推奨暗号にも含まれ、解析方法の進展が著しい、専用ハッシュ関数に注力して調査を行う必要がある。

3.2.2 活動目的

本ワーキンググループの目的は、電子政府推奨暗号リストに掲載されているハッシュ関数について最近提案されている攻撃方法の適用可能性を調査し安全性評価の再検討を行うと共に、主に米国で標準化が進んでいる暗号利用モードやメッセージ認証について調査を行い暗号技術監視委員会に報告するものである。ハッシュ関数の安全性の検討に関しては署名・認証技術調査ワーキンググループと連携して評価項目や視点について吟味する。

3.2.3 委員構成

主査：古原和邦　東京大学生産技術研究所　情報・システム系部門　助手
委員：廣瀬勝一　福井大学工学部　電気・電子工学科　助教授
委員：川村信一　株式会社東芝 研究開発センター
　　コンピュータ・ネットワークラボラトリー　室長
委員：古屋聡一　株式会社日立製作所 システム開発研究所　研究員
委員：盛合志帆　株式会社ソニー・コンピュータエンタテインメント
　　開発研究本部　リサーチサイエンティスト

3.2.4 活動内容

3.2.4.1 SHA-1の安全性評価

(1) 調査概要

近年提案されたSHA-1の衝突攻撃(以下、Wangの攻撃手法と呼ぶ)について調査を行い、SHA-1の安全性について検討した。

[W1] Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu, "Finding Collisions in the Full SHA-1", Advances in Cryptology - CRYPTO2005, Lecture Notes in Computer Science Vol.3621, pp.17--36, Springer-Verlag, 2005

[W2] XiaoyunWang, Andrew C Yao and Frances Yao ," Cryptanalysis on SHA-1" http://www.csrc.nist.gov/pki/HashWorkshop/2005/Oct31_Presentations/Wang_SHA1-New-Result.pdf

Wang の攻撃手法の概要は表 3.3 の通りである。

表 3.3 Wang の攻撃手法の概要

Wang の攻撃手法の位置づけ	暗号用ハッシュ関数が持つ衝突発見困難性と原像計算困難性および第二原像計算困難性のうち衝突発見困難性を誕生日攻撃よりも効率良く無効にしている。
攻撃条件	512 ビット入力に対して、1024 ビット（2 ブロック）を利用する
攻撃によって実現される脅威	ハッシュ値が同じになる 1024 ビットのメッセージ対を作成できる
攻撃コスト	誕生日攻撃が約 2^{80} 回の SHA-1 の実行を必要としているのに対して、2^{69} 回程度の SHA-1 の実行で攻撃が実行可能である。[W2]には 2^{63} 回の実行回数による手法が述べられている。

SHA-1 の安全性の検討には、以下の技術的な検討と考察が必要である。

［検討 1］ Wang らの結果の追試(正しいのか、真偽判定に不明点はないか)

［検討 2］ 最終的な攻撃の実現可能性(改良の余地、攻撃計算量、実装容易性)

さらに Wang の手法には下記の技術的不明点がある。

a. 局所衝突の探索法とディスターバンスベクトルの最適性

b. 内部変数差分の決定法

c. 計算量見積もりの妥当性

そこで、これらを検討するために外部に評価依頼を行った。

以下、本年度の検討結果をまとめる。

（ｉ）技術的不明点 a. 局所衝突の探索法とディスターバンスベクトルの最適性

この課題について、局所衝突の見直しと組み合わせ探索及びその最適性について検討し、この最適性を裏付ける評価結果を得た。審議の結果、本質的な差分パスとそれから計算される充足確率の最適性は極めて高く、この観点からの劇的な攻撃の改良は困難と考えられると結論した。

（ⅱ）技術的不明点 b. 内部変数差分の決定法

[W1]及び[W2]では内部変数差分の導出結果が例示され、その決定手法の一部が示されていない。しかしながら導出結果があれば、その後に続く処理の検証には障害にならず、また必要な計算量見積もりには影響を与えないことが確認された。従って、1 メッセージブロック目と 2 メッセージブロック目における 1 ラウンド目内部変数に対する差分の決定手法が明らかになっていないが、攻撃に必要な計算量の見積もりやアルゴリズムの検証において障害にならない、と結論した。

（ⅲ）技術的不明点 c. 計算量見積もりの妥当性

技術的不明点 b.が明らかでないものの計算量見積もりに影響を与えず、さらに必要な計算量は 2^{69} 回の SHA-1 実行であることが確認された。従って、Wang の攻撃手法[W1]に必要な計算量見積もりは 2^{69} 回の SHA-1 実行で妥当であると判断した。

上記の議論を踏まえて攻撃の実現性について検討した。Wang の攻撃手法[W1]は、計算量見積もりが確認できる程度に明らかになっているが、平成 18 年 2 月の時点では第三者が実装可能な状況にはない。しかし、攻撃アルゴリズムの大筋については確認できており、不明な部分も近い将来明らかになると予想する。

また、衝突探索に必要な攻撃アルゴリズムは、

・高い並列処理度

・極めて小さい必要メモリ量

という二つの特徴から、計算量単体評価の実現性と、攻撃全体の実現性のギャップは極めて小さいと考え、Wang らが発表した SHA-1 の計算量が 2^{63} の攻撃手法[W2]は、近い将来に第三者による実装が可能になり、極めて大きな脅威となると考えられると結論した。

(2) SHA-1 の安全性評価のまとめ

衝突発見困難性に対して、2^{69} 回以下の SHA-1 の実行回数で攻撃できる手法が発見された。ただし、公開された攻撃アルゴリズムには一部不明な点があり、第三者によって実装して検証されたわけではない。しかし、アルゴリズムの不明な点は近い将来に明らかになり第三者による実装が可能になると予想されるので、本攻撃アルゴリズムは極めて大きな脅威になると考えられる。

第二原像計算困難性に対しては、2^{60} バイトのメッセージに対して 2^{106} の SHA-1 の実行回数で攻撃できる手法が公開されたが、平成 18 年 2 月の時点では脅威と言えない。

3.2.4.2 SHA-256/-384/-512 の安全性評価

電子政府推奨ハッシュ関数である、SHA-256、SHA-384、SHA-512 に関する安全性を検討するために、外部に評価依頼を行った。

評価の結果、現時点では SHA-1 の攻撃手法がそのまま適用できるわけではなく、SHA-2 の安全性を脅かすものではないとの結論が確認された。また、平成 15 年から平成 18 年 1 月末までに発表された SHA-256、SHA-384、SHA-512 の安全性に関する論文を調査した結果、局所衝突に関しては、9 ステップで確率 2^{-66} で成立する局所衝突の発見が報告されているが、SHA-256 については 3 回以上の組み合わせが必要であり、誕生日攻撃を下回る計算量での攻撃には至っていないことが報告された。さらに、都合の良いメッセージ変更法が存在すると仮定した場合の安全性評価やメッセージ拡張関数の効果の解析について報告されているが、SHA-256、SHA-384、SHA-512 の安全性を脅かすレベルには達していないと結論された。

以上より、SHA-256、SHA-384、SHA-512 の安全性については、「実用的な安全性を脅かす攻撃方法が報告されていないため、これらのハッシュ関数は暗号の応用分野で使うのに十分安全であると考えられる」と結論した。

3.2.4.3 RIPEMD-160 の安全性評価及び Whirlpool の調査

RIPEMD-160及びWhirlpoolの安全性を検討するために外部に評価依頼を行った。

評価の結果、RIPEMD-160はSHA-1と構造が異なるものの、メッセージ置換やステップ依存のビットシフトなど安全性について本質的な部分であるデータ撹拌の構造についてはSHA-1と同様であり、RIPEMD-160はSHA-1と同程度の衝突探索のための有効性を持つ差分パスを持つと予想されることが確認された。ただし、このような差分パスの探索は非常に困難であり、現時点では有効な結果は得られていないことも示された。

しかしながら、研究の進展によってはSHA-1と同じ程度に危殆化する可能性があることから、「RIPEMD-160は異なる二つのブロック暗号L、Rで構成され、そのうちブロック暗号Lについ てはSHA-1と同程度の差分パスの存在が予想できることが報告されている。これはデータ撹拌においてメッセージ置換とステップ依存のビットシフトの採用などSHA-1と類似した関数を採用しているためであり、具体的なパスの発見など安全性に関する報告はないが、今後の研究の進展を考え研究動向について非常に注意する必要がある」と結論した。

Whirlpoolは平成12年にRijmenとBarettoによってNESSIEに提案され、その後、内部関数の仕様変更を経て平成17年にISO10118-3としてISO/IECの標準ハッシュ関数の一つに選ばれている。これまで安全性について述べられた文献はいくつかあるが、部分的な評価であ

り全体に関する安全性について言及されたものはない。

そこで、ビット単位の差分パス探索を行い差分特性確率の評価を行った結果、「Whirlpool 全体では差分の拡散が十分であり、近年の攻撃手法を適用しても衝突発見は困難である」と結論した。

3.2.4.4 暗号利用モード及びメッセージ認証の技術動向調査

暗号利用モード及びメッセージ認証の技術動向について平成 15 年度に引き続き調査を行った。新たなメッセージ認証方式として OMAC を追記し、NIST が OMAC1 を平成 17 年 5 月に SP800-38B として推奨方式に採用している情報を含めた。なお、NIST は OMAC1 を CMAC という名称で呼んでいる。

さらに HMAC の技術的詳細を追記し、その安全性について議論を行った。その結果、SHA-1 についてその初期値または入力の一部を秘密かつランダムとした場合の擬似ランダム性、衝突発見困難性が保証されていれば、SHA-1 を使った HMAC は安全である。Wang の攻撃手法は衝突攻撃であるが、上記の性質を脅かす結果は報告されておらず、HMAC の安全性に影響を与えないことを確認した。

3.3 擬似乱数生成系調査ワーキンググループ

3.3.1 調査背景

電子政府推奨暗号リストにおける擬似乱数生成系では、SHA-1 を使った擬似乱数生成系が例示されている。しかし、用途によっては、例示された擬似乱数生成系以外でも、暗号学的に安全性が確認できれば、用途によっては利用できる場合もある。そこで電子政府で使用される擬似乱数生成系が、少なくとも高い乱数性を持つことを検証するためのツールが必要と考えられている。擬似乱数の検定法には様々な観点からの検定法が存在しており、それらを複数集めて検定ツールとしてまとめられたものもいくつか存在する。代表的なものとしては、NIST Special Publication (SP) 800-22、DIEHARD、”The Art of Computer programming 準数値算法” D.Knuth 著に記載されたものなどが知られている。しかし、これらの検定ツールを比較検討すると、

1) 検定ツールごと採用されている検定法が異なり、検定法の選択基準が明確になっていない
2) 同じ検定手法でも検定ツール毎に閾値等の設定値が異なる場合がある

など問題点が存在する。特に、NIST SP800-22 に関しては、検定仕様のみならず対応する検定プログラムが公開されている。しかし、公開されている検定プログラムには、いくつかの検定法に不具合があることを指摘した学術論文がある。DIEHARD については、検定プログ

ラムについては公開されているものの判断基準が示されていないので参考資料としての位置づけとした。

他方、暗号モジュール委員会で検討中の暗号モジュール評価においても擬似乱数検定が必要になるという背景があり、CRYPTREC としての擬似乱数検定ミニマムセットの策定を目標に、擬似乱数生成系の調査および検定法の調査を行っている。

3.3.2 活動内容

乱数の検定法は一般的な乱数を対象としているが、CRYPTREC が対象としているのは暗号利用用途の擬似乱数生成系であり、このような観点から CRYPTREC が推奨する擬似乱数検定法をまとめた擬似乱数検定ツール(以下 CRYPTREC 擬似乱数検定ミニマムセット)を作成することを最終的な目標とする。

この CRYPTREC 擬似乱数検定ミニマムセットの導出にあたり各擬似乱数検定法の理論的根拠を確認し、どのような観点からの検定が CRYPTREC の方針に適切かを整理する。また、閾値等のパラメータを適切な値に修正する。

3.3.3 委員構成

主査：金子　敏信　東京理科大学理工学部　教授
委員：荒木　純道　東京工業大学工学部　教授
委員：栃窪　孝也　東芝ソリューション株式会社 SI 技術開発センター
委員：廣瀬　勝一　福井大学工学部　電気・電子工学科　助教授
委員：森井　昌克　神戸大学工学部　教授

3.3.4 活動概要

上期（～平成 17 年 9 月）は、CRYPTREC 擬似乱数検定ミニマムセットの導出にあたり各擬似乱数検定法の理論的根拠を確認し、どのような観点からの検定が CRYPTREC の方針に適切かを整理し、ミニマムセットを確定した。ミニマムセットに対応する仕様書は付録 6 に示す。下期は、ミニマムセットに基づいた擬似乱数検定ツールの作成を行った。

(1) 乱数検定のためのミニマムセットの作成

擬似乱数検定のためのミニマムセットとしては、NIST の SP800-22 の 16 種類の検定法の中から 14 種類をミニマムセットとして採択し、検定項目を定めた。表 3.4 に採択した 14 種類の検定項目を示す。

表 3.4　擬似乱数検定のミニマムセット

1.　頻度検定
2.　ブロック単位の頻度検定
3.　連検定
4.　ブロック単位の最長連検定
5.　2 値行列ランク検定
6.　重なりの無いテンプレート適合検定
7.　重なりのあるテンプレート適合検定
8.　Maurer の「ユニバーサル統計量」
9.　線形複雑度検定
10.　系列頻度検定
11.　累積和検定
12.　ランダム回遊検定
13.　変形ランダム回遊検定
14.　近似エントロピー検定

(2)　検定に対する可否判断基準の作成

採択した 14 種類の検定項目に対して、NIST の SP 800-22 の検定法をベースに可否判断基準をリストアップし、表 3.5 に示す検定に対する可否判断基準案を作成した。

表 3.5　可否判断基準

検定項目名	可否判断基準	検定に用いるデータ-量
1.　頻度検定	p-value ≧ 0.01	1,000,000bit×1,000 本 (総量 10^9bit)
2.　ブロック単位の頻度検定	p-value ≧ 0.01	
3.　連検定	p-value ≧ 0.01	
4.　ブロック単位の最長連検定	p-value ≧ 0.01	
5.　2 値行列ランク検定	p-value ≧ 0.01	
6.　重なりの無いテンプレート適合検定	p-value ≧ 0.01	
7.　重なりのあるテンプレート適合検定	p-value ≧ 0.01	
8.　Maurer の「ユニバーサル統計量」	p-value ≧ 0.01	
9.　線形複雑度検定	p-value ≧ 0.01	
10.　系列頻度検定	p-value ≧ 0.01	
11.　累積和検定	p-value ≧ 0.01	
12.　ランダム回遊検定	p-value ≧ 0.01	
13.　変形ランダム回遊検定	p-value ≧ 0.01	
14.　近似エントロピー検定 (3)参照)	p-value ≧ 0.01	

(3) 検定に用いる閾値等の修正

近似エントロピー検定においては、昨年度の調査結果からパラメータをせばめることによって利用できると考えられる。伏見らによる先行研究[伏見正則，”乱数”，UP 応用数学選書，東京大学出版会，1989 年]では、$m < \log_2(n-6)$ で良いとあるが、昨年度実験した結果を考えると、具体的には、$m < \log_2(n-7)$ とパラメータの範囲を狭めることにより、正確な値をとりえると考えられる。以上のことから、近似エントロピー検定については範囲を狭めて（各ボックスに 100 個以上）ミニマムセットに含めることとした。

(4) ミニマムセット仕様書および擬似乱数検定ツールの作成

上記の結果に基づき、擬似乱数生成系調査WG事務局で、NIST SP 800-22 “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”を参考に、「擬似乱数検定のための CRYPTREC ミニマムセット仕様書」（付録 6）を作成した。この「擬似乱数検定のための CRYPTREC ミニマムセット仕様書」に基づき、暗号モジュール委員会事務局が「擬似乱数検定ツール」の作成を実施している。

3.3.5 まとめ

擬似乱数生成系調査ワーキンググループとしては、ワーキンググループ設置時点の目標である CRYPTREC 擬似乱数検定ミニマムセットの策定を完了した。一方、ISO/IEC JTC1 SC27 WG2 では、電子政府推奨暗号リスト中に例示された擬似乱数生成系以外の擬似乱数系を評価する際に利用する。ミニマムセットによる評価で不合格となる擬似乱数生成系は、採用しないように勧告する。

付録 1

電子政府推奨暗号リスト

平成 15 年 2 月 20 日
総　　務　　省
経　済　産　業　省

技術分類		名称
公開鍵暗号	署名	DSA
		ECDSA
		RSASSA-PKCS1-v1_5
		RSA-PSS
	守秘	RSA-OAEP
		RSAES-PKCS1-v1_5(注 1)
	鍵共有	DH
		ECDH
		PSEC-KEM(注 2)
共通鍵暗号	64 ビットブロック暗号(注 3)	CIPHERUNICORN-E
		Hierocrypt-L1
		MISTY1
		3-key Triple DES(注 4)
	128 ビットブロック暗号	AES
		Camellia
		CIPHERUNICORN-A
		Hierocrypt-3
		SC2000
	ストリーム暗号	MUGI
		MULTI-S01
		128-bit RC4(注 5)
その他	ハッシュ関数	RIPEMD-160(注 6)
		SHA-1(注 6)
		SHA-256
		SHA-384
		SHA-512
	擬似乱数生成系(注 7)	PRNG based on SHA-1 in ANSI X9.42-2001 Annex C.1
		PRNG based on SHA-1 for general purpose in FIPS 186-2 (+ change notice 1) Appendix 3.1
		PRNG based on SHA-1 for general purpose in FIPS 186-2 (+ change notice 1) revised Appendix 3.1

注釈：

(注 1)　SSL3.0/TLS1.0 で使用実績があることから当面の使用を認める。

(注 2)　KEM (Key Encapsulation Mechanism) -DEM(Data Encapsulation Mechanism)構成における利用を前提とする。

(注 3)　新たな電子政府用システムを構築する場合、より長いブロック長の暗号が使用できるのであれば、128 ビットブロック暗号を選択することが望ましい。

(注 4)　3-key Triple DES は、以下の条件を考慮し、当面の使用を認める。

1) FIPS46-3 として規定されていること

2) デファクトスタンダードとしての位置を保っていること

(注 5)　128-bit RC4 は、SSL3.0/TLS1.0 以上に限定して利用することを想定している。なお、リストに掲載されている別の暗号が利用できるのであれば、そちらを使用することが望ましい。

(注6)　新たな電子政府用システムを構築する場合､より長いハッシュ値のものが使用できるのであれば､256ビット以上のハッシュ関数を選択することが望ましい｡ただし、公開鍵暗号での仕様上､利用すべきハッシュ関数が指定されている場合には、この限りではない。

(注7)　擬似乱数生成系は､その利用特性上､インタオペラビリティを確保する必要性がないため､暗号学的に安全な擬似乱数生成アルゴリズムであれば、どれを利用しても基本的に問題が生じない。したがって､ここに掲載する擬似乱数生成アルゴリズムは「例示」である。

別添

電子政府推奨暗号リストに関する修正情報

修正日付	修正箇所	修正前	修正後	修正理由
平成 17 年 10 月 12 日	注釈の注 4)の 1)	FIPS46-3 として規定されていること	SP800-67 として規定されていること	仕様変更を伴わない､仕様書の指定先の変更

付録 2

電子政府推奨暗号リスト掲載暗号の問い合わせ先一覧

1.1　公開鍵暗号技術

暗号名	DSA
関連情報	仕様 ・ANSI X9.30:1-1997, Public Key Cryptography for The Financial Services Industry: Part 1: The Digital Signature Algorithm (DSA) で規程されたもの。 ・参照 URL <http://www.x9.org/>　なお、同規格書は日本規格協会(http://www.jsa.or.jp/)から入手可能である。

暗号名	ECDSA (Elliptic Curve Digital Signature Algorithm)
関連情報	公開ホームページ 和文：http://jp.fujitsu.com/group/labs/techinfo/technote/crypto/ecc.html 英文：http://jp.fujitsu.com/group/labs/en/techinfo/technote/crypto/ecc.html
問い合わせ先	富士通株式会社 電子政府推奨暗号 問合わせ窓口 E-MAIL：crypto-ml@ml.soft.fujitsu.com

暗号名	RSA Public-Key Cryptosystem with Probabilistic Signature Scheme (RSA-PSS)
関連情報	仕様　公開ホームページ ・PKCS#1 RSA Cryptography Standard (Ver.2.1) ・参照 URL <http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/> 和文：　なし 英文：　http://www.rsasecurity.com/rsalabs/submissions/index.html
問い合わせ先	〒100-0005 東京都千代田区丸の内 1-3-1 東京銀行協会ビルヂング 13F RSA セキュリティ株式会社 デベロッパ営業本部 部長　齊藤賢一 TEL：03-5222-5210, FAX：03-5222-5270, E-MAIL：ksaito@rsasecurity.com

暗号名	RSASSA-PKCS1-v1_5
関連情報	仕様　公開ホームページ ・PKCS#1 RSA Cryptography Standard (Ver.2.1) ・参照 URL <http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/index.html> 和文：　なし 英文：　http://www.rsasecurity.com/rsalabs/submissions/index.html
問い合わせ先	〒100-0005 東京都千代田区丸の内 1-3-1 東京銀行協会ビルヂング 13F RSA セキュリティ株式会社 デベロッパ営業本部 部長　齊藤賢一 TEL：03-5222-5210, FAX：03-5222-5270, E-MAIL：ksaito@rsasecurity.com

暗号名	RSA Public-Key Cryptosystem with Optimal Asymmetric Encryption Padding (RSA-OAEP)
関連情報	仕様　公開ホームページ ・PKCS#1 RSA Cryptography Standard (Ver.2.1) ・参照 URL <http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/index.html> 和文：　なし 英文：　http://www.rsasecurity.com/rsalabs/submissions/index.html
問い合わせ先	〒100-0005 東京都千代田区丸の内 1-3-1 東京銀行協会ビルヂング 13F RSA セキュリティ株式会社 デベロッパ営業本部 部長　齊藤賢一 TEL：03-5222-5210, FAX：03-5222-5270, E-MAIL：ksaito@rsasecurity.com

暗号名	RSAES-PKCS1-v1_5
関連情報	仕様 ・PKCS#1 RSA Cryptography Standard (Ver.2.1) ・参照 URL <http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/index.html>
問い合わせ先	〒100-0005 東京都千代田区丸の内 1-3-1 東京銀行協会ビルヂング 13F RSA セキュリティ株式会社 デベロッパ営業本部 部長　齊藤賢一 TEL：03-5222-5210, FAX：03-5222-5270, E-MAIL：ksaito@rsasecurity.com

暗号名	DH
関連情報	仕様 ・ANSI X9.42-2001, Public Key Cryptography for The Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography で規定されたもの。 ・参照 URL <http://www.x9.org/>　なお、同規格書は日本規格協会(http://www.jsa.or.jp/)から入手可能である。

暗号名	ECDH (Elliptic Curve Diffie-Hellman Scheme)
関連情報	公開ホームページ 和文:http://jp.fujitsu.com/group/labs/techinfo/technote/crypto/ecc.html 英文:http://jp.fujitsu.com/group/labs/en/techinfo/technote/crypto/ecc.html
問い合わせ先	富士通株式会社 電子政府推奨暗号 問合わせ窓口 E-MAIL：crypto-ml@ml.soft.fujitsu.com

暗号名	PSEC-KEM Key agreement
関連情報	公開ホームページ 和文：http://info.isl.ntt.co.jp/crypt/psec/index.html 英文：http://info.isl.ntt.co.jp/crypt/eng/psec/index.html
問い合わせ先	〒239-0847 神奈川県横須賀市光の丘 1-1-609A 日本電信電話株式会社　NTT 情報流通プラットフォーム研究所 セキュリティプラットフォームグループ　主任研究員　神田雅透 TEL：046-859-2437, FAX：046-855-1533, E-MAIL：kanda@isl.ntt.co.jp

1.2　共通鍵暗号技術

暗号名	CIPHERUNICORN-E
関連情報	公開ホームページ 和文：http://www.sw.nec.co.jp/middle/SecureWare/advancedpack/
問い合わせ先	〒108-8558 東京都港区芝浦 4-14-22 日本電気株式会社　第一システムソフトウェア事業部 TEL：03-3456-7075, FAX：03-3456-4289 E-MAIL:soft@security.jp.nec.com

暗号名	Hierocrypt-L1
関連情報	公開ホームページ 和文：http://www.toshiba.co.jp/rdc/security/hierocrypt/ 英文：http://www.toshiba.co.jp/rdc/security/hierocrypt/
問い合わせ先	〒212-8582 神奈川県川崎市幸区小向東芝町 1 (株) 東芝 研究開発センターコンピュータ・ネットワークラボラトリー 主任研究員 秋山浩一郎 TEL：044-549-2156, FAX：044-520-1841 E-MAIL:crypt-info@isl.rdc.toshiba.co.jp

暗号名	MISTY1
関連情報	公開ホームページ http://www.mitsubishielectric.co.jp/corporate/randd/information_technology/security/code/misty01_b.html
問い合わせ先	〒100-8310　東京都千代田区丸の内 2-7-3（東京ビル） 三菱電機株式会社　インフォメーションシステム事業推進本部 情報セキュリティ推進センター　担当課長　羽山哲雄 TEL:03-3218-4116　FAX:03-3218-3638 E-MAIL:Hayama.Tetsuo@aj.MitsubishiElectric.co.jp

暗号名	Triple DES
関連情報	仕様 ・NIST SP 800-67 (Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, May 2004) ・参照 URL <http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf>

暗号名	AES
関連情報	仕様 ・FIPS PUB 197, Advanced Encryption Standard (AES) ・参照 URL <http://csrc.nist.gov/CryptoToolkit/tkencryption.html>

暗号名	Camellia
関連情報	公開ホームページ 和文： http://info.isl.ntt.co.jp/camellia/ 英文： http://info.isl.ntt.co.jp/camellia/
問い合わせ先	・〒239-0847 神奈川県横須賀市光の丘 1-1-609A 日本電信電話株式会社 NTT 情報流通プラットフォーム研究所 セキュリティプラットフォームグループ 主任研究員 神田雅透 TEL： 046-859-2437, FAX：046-855-1533, E-MAIL：kanda@isl.ntt.co.jp ・〒104-6212 東京都中央区晴海 1-8-12 ﾄﾘﾄﾝｽｸｴｱｵﾌｨｽﾀﾜｰ Z13 階 三菱電機株式会社 通信システム事業本部 NTT 事業部 NTT 第一部第一課 課長 富田文隆 TEL:03-6221-2634, FAX:03-6221-2771 E-MAIL: fumitaka.tomita@hq.melco.co.jp

暗号名	CIPHERUNICORN-A
関連情報	公開ホームページ 和文：http://www.sw.nec.co.jp/middle/SecureWare/advancedpack/
問い合わせ先	〒108-8558 東京都港区芝浦 4-14-22 日本電気株式会社 第一システムソフトウェア事業部 TEL：03-3456-7075, FAX：03-3456-4289 E-MAIL:soft@security.jp.nec.com

暗号名	Hierocrypt-3
関連情報	公開ホームページ 和文： http://www.toshiba.co.jp/rdc/security/hierocrypt/ 英文： http://www.toshiba.co.jp/rdc/security/hierocrypt/
問い合わせ先	〒212-8582 神奈川県川崎市幸区小向東芝町 1 (株) 東芝 研究開発センターコンピュータ・ネットワークラボラトリー 主任研究員 秋山浩一郎 TEL：044-549-2156, FAX：044-520-1841 E-MAIL:crypt-info@isl.rdc.toshiba.co.jp

暗号名	SC2000
関連情報	公開ホームページ 和文： http://jp.fujitsu.com/group/labs/techinfo/technote/crypto/sc2000.html 英文： http://jp.fujitsu.com/group/labs/en/techinfo/technote/crypto/sc2000.html
問い合わせ先	富士通株式会社 電子政府推奨暗号 問合わせ窓口 E-MAIL：crypto-ml@ml.soft.fujitsu.com

暗号名	MUGI
関連情報	公開ホームページ 和文： http://www.sdl.hitachi.co.jp/crypto/mugi/ 英文： http://www.sdl.hitachi.co.jp/crypto/mugi/index-e.html
問い合わせ先	〒244-8555 神奈川県横浜市戸塚区戸塚町 5030 番地 (株) 日立製作所 ソフトウェア事業部 ネットワークソフトウェア本部 担当本部長 松永和男 TEL： 045-862-8498, FAX： 045-865-9055 E-MAIL： matsun_k@itg.hitachi.co.jp

暗号名	MULTI-S01
関連情報	公開ホームページ 和文： http://www.sdl.hitachi.co.jp/crypto/s01/index-j.html 英文： http://www.sdl.hitachi.co.jp/crypto/s01/index.html
問い合わせ先	〒244-8555 神奈川県横浜市戸塚区戸塚町 5030 番地 (株) 日立製作所 ソフトウェア事業部ネットワークソフトウェア本部 担当本部長 松永和男 TEL：045-862-8498, FAX： 045-865-9055 E-MAIL： matsun_k@itg.hitachi.co.jp

暗号名	RC4
関連情報	仕様 ・問い合わせ先 RSA セキュリティ社(http://www.rsasecurity.co.jp/) ・仕様 RC4 のアルゴリズムについては、RSA Laboratories が発行した CryptoBytes 誌(Volume5, No.2, Summer/Fall 2002) に掲載された次の論文に記載されているもの。Fluhrer, Scott, Itsik Mantin, and Adi Shamir, "Attacks On RC4 and WEP", CryptoBytes, Volume 5, No.2, Summer/Fall 2002 ・参照 URL <http://www.rsasecurity.com/rsalabs/cryptobytes/index.html>

1.3 ハッシュ関数

暗号名	RIPEMD-160
関連情報	仕様 ・参照 URL <http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html>

暗号名	SHA-1, SHA-256, SHA-384, SHA-512
関連情報	仕様 ・FIPS PUB 186-2, Secure Hash Standard (SHS) ・参照 URL <http://csrc.nist.gov/CryptoToolkit/tkhash.html>

1.4 擬似乱数生成系

暗号名	PRNG in ANSI
関連情報	仕様 ・ANSI X9.42-2001, Public Key Cryptography for The Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography ・参照 URL <http://www.x9.org/> なお、同規格書は日本規格協会(http://www.jsa.or.jp/) から入手可能である。

暗号名	PRNG in ANSI X9.62-1998 Annex A.4
関連情報	仕様 ・ANSI X9.62-1998, Public Key Cryptography for The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA) ・参照 URL <http://www.x9.org/> なお、同規格書は日本規格協会(http://www.jsa.or.jp/) から入手可能である。

暗号名	PRNG in ANSI X9.63-2001 Annex A.4
関連情報	仕様 ・ANSI X9.63-2001, Public Key Cryptography for The Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography ・参照 URL <http://www.x9.org/> なお、同規格書は日本規格協会 (http://www.jsa.or.jp/) から入手可能である。

暗号名	PRNG for DSA in FIPS PUB 186-2 Appendix 3
関連情報	仕様 ・FIPS PUB 186-2, Digital Signature Standard (DSS) ・参照 URL <http://csrc.nist.gov/CryptoToolkit/tkrng.html>

暗号名	PRNG for general purpose in FIPS PUB 186-2 (+ change notice 1) Appendix 3.1
関連情報	仕様 ・FIPS PUB 186-2, Digital Signature Standard (DSS) ・参照 URL <http://csrc.nist.gov/CryptoToolkit/tkrng.html>

暗号名	PRNG in FIPS PUB 186-2 (+ change notice 1) revised Appendix 3.1/3.2
関連情報	仕様 ・FIPS PUB 186-2, Digital Signature Standard (DSS) ・参照 URL <http://csrc.nist.gov/CryptoToolkit/tkrng.html>

付録 3

学会等での主要発表論文一覧

1. ハッシュ関数の解読技術

1.1 SHA-1 MD5、MD4、RIPEMD の解読技術

[1]Cryptanalysis of the Hash Functions MD4 and RIPEMD

Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen and Xiuyuan Yu

MD4 の collision アタックとして 2^{-2} から 2^{-6} 確率の間の(トータルで 2^8 operations を超えない)の MD4 の collision を発見し、RIPEMD の collision アタックとしておよそ 2^{19} 回の RIPEMD 計算の collision を発見した。 また MD4 の second pre-image attack として、weak message と呼ばれるいくつかの message に対してそれらの第 2 原像と呼ばれる同じハッシュ値を持つ message を(一時的な計算のみによって)より効率良く求めた。このような message は weak message と呼ばれ、任意の message に対して第 2 原像を求めるためには理想的安全性を持つ場合は 2^{128} 回の MD4 の計算が必要であったのに対し、今回のアタックでははるかに容易に第 2 原像を計算可能な message を発見した。また選択 message を用いた第 2 原像アタックとして、与えられた message M に対して、message を数ビット変更することにより、変更した message の第 2 原像を容易に求めることに成功した。さらに MD4 の選択 message による原像アタックとして、ランダムな message が確率 2^{-122}($2^{512-122}=2^{490}$ が 1 ブロック message)の weak message 確率を持つのに対し、Hongbo Yu はその結果を 2^{-72} に改良する weak message を与え、与えられた message M に対して、約 110 ビットを変更することにより、変更された message の第 2 原像を求めることが可能であることを示した。RIPEMD の collision アタックとしては確率 2^{-17} で計算量が約 2^{19} 回の RIPEMD 演算の collision を発見した。

[2]How to break MD5 and Other Hash Functions

Xiaoyun Wang and Hongbo Yu

MD5 は 1992 年に MD4 の強化版として提案されたハッシュ関数である。MD5 は、今日セキュリティアルゴリズムとして最低でも Wang らが 2004 年に結果を発表するまで広く使われていた。

MD5 解読の関連研究としてアルゴリズム発表以来、数々の弱点が報告されている。1993 年に Boer と Bosselaers が同じ message で異なる初期値を持つ MD5 のある種の擬似 collision を発見し、最上位ビットの弱アバランシュ性を発見した。Eurocrypt'96 で Dobbertin が 2 つの異なる 512 message と選択初期値からなるセミフリースタートの collision を与えて

いた。
今回の結果は IBM P690 で MD5 の collision を発見するのに必要な時間が、(M0, M0')の発見に約 1 時間(最速で 15 分)、(M1, M1')の発見に 15 秒から 5 分というものである（注：SHA-1 で用いた新しいテクニックによれば、MD5 の collision の発見に約 2^{32} 回の計算にまで計算量を削減可能である)。

[3]Collisions of SHA-0 and Reduced SHA-1

Eli Biham, Rafi Chen, Antoione Joux, Patrick Carribault, Christophe Lemuet and William Jalby

SHA-0 の collision 発見テクニックの改良版と、40 段までの SHA-1 の collision および 53-58 段の変形版で理論的にバースデー攻撃よりも効率的に collision を発見できることを示した。改良法は、near Collision と pseudo collision（および near-pseudo collision）を組み合わせてマルチブロックで collision を発見している。これを用いて、4 ブロック SHA-0 の collision 攻撃では、CJ/BC 法の計算量から 2^{51} に削減し 80,000 CPU hours で計算でき、具体的に collision を示している。また、40 段 SHA-1 の collision は PC により 2 秒で求められると述べた。
注) near collision:
同じ IV、異なる M≠M’に対して dH(h(IV, M), h(IV, M’)) < ε （dH はハミング距離）
pseudo collision: 異なる IV≠IV’, M≠M’に対して h(IV,M) = h(IV’, M’)

[4]Strategies and Techniques of the Full SHA-1 Collision Attack

Xiaoyun Wang

Eurocrypt2005 のランプセッションにおいて、Wang が CRYPTO2005 で報告する予定の SHA-1 の collision 発見方法についての講演があった。これによると、1997 年に Wang が代数的な方法に基づく SHA-0 の最初のアタックを中国内で発表しており、collision が 2^{58} 回 SHA-0 の計算で発見可能であることを示していた。1998 年に Chabaud と Joux は差分解読により独立に同じ差分パスを発見し、1998 年に Wang が message modification と呼ばれる方法により確率を 2^{-58} から 2^{-45} に改良し、さらに collision 差分パスがいくつかの local collision（中間段での Collisiion）によって構成されていることを指摘していた。2004 年に Joux が near collision を利用した 4 ブロックの collision を発表し、同じく 2004 年に Wang が 2 ブロックの MD5 の Collision を発表し、Biham と Chen が 40 ラウンドの SHA-1 について最初のアタックを発表していた。
これらに対し SHA-1 についての今回の結果は、フルランド SHA-1 の disturbance vector を利用し、24 ステップから 80 ステップまでの near collision 差分パスで確率 2^{-68} のものを発見し、その後 1-23 ステップについて disturbance vector に対応する不可能な差分パス

を可能な差分パスに変換するアタックで、フルラウンド SHA-1 の最初のブロックの near collision を与える条件として、58 ラウンド SHA-1 の collision 差分、58 ラウンド SHA-1 の collision の例をそれぞれ与えている。collision を発見するために必要な計算量は 2^{68} と見積もられている。

[5]What is the potential danger behind the collisions of hash functions?

Xiaoyun Wang

昨夏以来ハッシュ関数 MD5、SHA-1 などの衝突を具体的に計算して、現在最も重要な話題になっている Wang（王）による講演が予定されていたが、都合により Wang のスライドをイスラエルの Biham（ビハム）が代理で講演した。今回の講演は、解読の技術的側面だけでなく、PKI などの認証への影響に関する実システムに対する影響についても詳しく解説されていた。

認証への影響の実際の例として、Stepha Luck（リュック、独）らによるポストスクリプトファイルの偽造が可能であるという結果や、Lenstra（レンストラ）らによる X.509 認証の偽造が可能であるという結果を解説した。

解読技術については Wang の示した衝突パターンが正しいことを Biham は技術的に確認したものの、如何にしてその衝突パターンを発見したかについては Wang が全ての情報を公開していないため、Biham 自身まだ解明出来ていないと発言。

[6]Recent advances in hash functions and the way to go

Eli Biham

MD5, SHA-1 にはメッセージの diffusion と呼ばれるメッセージを少しだけ変えたときにその影響がハッシュ値に及ぶ度合いが小さいという問題点があり、最近の衝突の発見はこの欠点に基づいている。

ハッシュ関数とブロック暗号のアタックには強い類似性があり、ハッシュ関数はブロック暗号と同じ原理、技術で設計されなければならない。ブロック暗号と同様の設計方針で設計されたものは現在 Tiger（Biham 自身が提案）, Whirlpool しかなく、SHA-256, -512 などは SHA-1 よりも安全性は高いと思われるが、SHA-1 などと類似の設計方針で信頼度は低く、長期的な視点で SHA-256, 512 などをやめてブロック暗号と同様の設計方針のハッシュ関数を新しい標準にすべきだという見解を述べた。

[7]Finding Differential Patterns for the Wang Attack

Magnus Daum

この分野の先駆者である Dobbertin（独）の PhD の学生である Daum（独）氏による Wang の

アタックに関する分析結果に関する講演。通常と異なる算術の足し算に関する差分を用いていることの理論的な説明、衝突を発見するための差分パターンを探索する方法についての考察についての説明がなされたが、まだ完全解明には至っていない。

[8]Efficient Collision Search Attacks on SHA-0

Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin

[9]Finding Collisions in the Full SHA-1

Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu

昨年夏の発表以来、暗号学会のみならず、PKI などの認証においてもその影響が深刻な問題となっている SHA-0、SHA-1 の衝突発見に関する講演である。衝突発見に必要とされる計算量はそれぞれ 2^{39}、2^{69} と見積もられている。この結果は理論の範囲に留まらず、SHA-0 については実際に衝突する例を発見しており、また SHA-1 についてもフルラウンド 80 段中 58 段までは実際に衝突が発見されている。SHA-1 の衝突については、1 ブロックに対して 2^{68} の計算量で近衝突が発見できることを利用し、2 ブロックで 2^{69} の計算量で衝突を発見可能であるという理論的な結果が報告された。これに対し、電子政府推奨暗号リストに入っている SHA-256, 512 に関してはローカルコリジョンの存在確率が小さいこと、メッセージ拡大部がより複雑であることから現状では衝突発見は困難であるが、今後急速に解析が進展する可能性もあり、更なる解析が必要であると述べた。

[10]New Collision Search for SHA-1

Xiaoyun Wang, Andrew Yao and Frances Yao

SHA-1 の衝突発見に必要な計算量を 2^{63} にまで削減可能であるという発表であり、従来必要であった 2^{69} の演算量を削減することに成功したと述べている。ただし技術的詳細についてはほとんど述べられてはおらず、その真偽については不明である。今後の国際会議等でその詳細が発表されるものと思われるため今後の動向を注視する必要がある。

[11]The Second-Preimage Attack on MD4

Hongbo Yu, Gaoli Wang, Guoyan Zhang and Xiaoyun Wang

MD4 に対する Second-preimage Attack に成功したとの主張であったが、一般的に Second-pre-imageattack といわれる攻撃の成功にはいたっておらず、ターゲットとなるメッセージそのものに対する Second-preimage は求められたという攻撃ではない。到達できているのは、ターゲットとなるメッセージを modify してその modify したメッセージの Second-preimage を求めるという攻撃。ターゲットであるメッセージそのものに対する Second-pre-image を求める攻撃への到達はまだ先と思われるため、緊急の脅威に結びつく

ものではないが、今後この研究の動向は逐次追って把握しておく必要がある。

[12]Cryptanalysis for Hash functions and Some Potential Dangers
Xiaoyun Wang

SHA-1の攻撃について、以前CRYPTO 2004のRumpセッション及び11月に開かれたNISTのHash Workshop で公開した差分パスとは別の新しいバスを見つけた。
Message modification を行なう前にビット調整を行なう処理が必要とされるが、今回新たに発見したパスを利用することで、その調整できるビット選択に対する自由度を高めることができ、その結果として下位ラウンドのHWを小さくすることができ、メッセージ探索空間を広げることができきた。そのため、最終的な衝突を見つけるまでの計算量を削減することができると主張している。具体的には、従前のパスを用いた場合、Full-Roundでの衝突を見つけるために必要となる計算量が2^{63}と見積もられていたが、新たなパスを利用した場合の衝突探索に必要となる計算量は2^{61}～2^{62}にまで軽減できると主張している。しかし、この計算量の見積もりは、計算機実験で確認された値ではなく、机上見積もりに留まる。発表では、新たなパスの具体的な値を示した表なども紹介された。本発表内容の論文は、まだ未公開であり、プレゼン資料のみが公開されている情報である。

1.2　新しいハッシュ関数の構成等に関する提案

[13]Hash functions – present state of art
Bart Preneel

ハッシュ関数の様々な安全性について理論的に解説した。安全性については完璧な定義があるものの、安全性が証明されているような構成のハッシュ関数は無いこと、Merkle-Damgard と呼ばれる代表的なハッシュ関数の構成についても第 2 原像アタックが存在することを解説した。Preneel は、我々はハッシュ関数の安全性について実はまだほとんど理解しておらず、設計者はあまりに楽観的であるとの見解を述べた。
衝突に対して耐性のあるサイズの小さなハッシュ関数が必要だがその設計法はまだ確立しておらず、第 2 原像攻撃に対する耐性、部分原像攻撃に対する耐性、擬似乱数性、同じハッシュ関数を繰り返し使った時の安全性など、他のセキュリティ特性についての研究も必要であると述べた。

[14]Some Attacks Against a Double Length Hash Proposal
Lars R. Knudsen and Frederic Muller

FES 2005などで提案されているDouble Length Hash Function(提案者：Mridul Nandi) の

脆弱性の指摘。彼らの提案する方法では、collisionの探索攻撃に対して安全ではないことを示した。更に、このようにDouble length の入力による構成法による、collision attackにより耐えうる構成方法を提示し、この構成法をセキュアに構成する知見が蓄積された。発表の最後では、効率的なセキュアなハッシュ関数の構成にはこの構成方法は好ましくない、と結論付けていた。
Double Hash Function の提案者であるNandi氏は、FSE 2005での提案後Webサイトに改善版の論文を出していた(Indocrypt 2006で発表、以下で報告)が、その方式に対しても同様の脆弱性があることを指摘した。さらに、この発表の後に発表されたLucksらの方式についても、同様の脆弱性を持つとしている。

[15]A Failure-Friendly Design Principle for Hash Functions
Stefan Lucks

Markle-Damgard ベースのハッシュ関数の構造を元に、Wide-pipe Hash Functionという2入力1出力の構造を持つ方式を提案。上記の指摘にあるようにこの構造では、安全はハッシュ関数の構成は難しいことが指摘されている。

[16]How to Construct Universal One-Way Hash Functions of Order r .
Deukjo Hong, Jaechul Sung, Seokhie Hong and Sa ngjin Lee

ランダムオラクルをベースに解析を行い、adaptiveなアクセスを許したとしても安全なハッシュ関数の構成方法を提案。やや理論よりの話であり効率などの点で現実性にかける。

[17]Towards Optimal Double-Length Hash Functions.
Mridul Nandi

FES05の提案後、その安全性解析の進展結果を発表。安全性と効率性をパラメータ評価できるような指標を提案。さらにAsiacrypt 2005で提案されているLucksらの方法が安全でないことを例として挙げた。しかしながら、本提案内容については、先日行われたAsiacrypt 2005のKnudsenら論文の中で、ここで提案されている方法であったとしても安全な構成とはならないことが指摘されている。本論文やLucksらの論文は事前にWebサイトに公開されていた為、早い段階での解析が進んだものと思われ、ハッシュ関数の研究動向の急速な展開が感じ取られる。

[18]Collision Resistant Usage of SHA-1 via Message Pre-processing

Michael Szydlo

ハッシュ関数の延命策のひとつとして、ハッシュ関数の入力値に対して事前計算を行った結果をハッシュ関数の入力とする案があるが、本発表はその構成方式と耐 collision の性質に関する解析結果を発表。word 単位のメッセージ間に 0 列を入れる方法と word 単位のメッセージを 2 重に入れていく方法とを提案。少なくとも Wang 氏らによる message midification のテクニックは回避することができる。ただし、メッセージ長は、33%-200%伸び、処理時間は、33%-200%伸びる。他の延命措置として代表的な、Truncated 方法と Randomized 方法との効率性の比較を提示。他方式に比べ、flexibility がある方式といえる。ただし efficiency の点では最適とはいえない方式と思われる。

1.3 ハッシュ関数関連に関する政治的動向

Hash functions - perspective from the United States

Rich Schroeppel

NIST 関係者で Sandia National Labs の Schroeppel 氏による Hash 関数に関しての現在までに至る状況説明、最近のアタックによる影響、短期的な対応、新アルゴリズム提案などの長期的な対応、10 月 31 日の NIST によるワークショップに関する説明があった。現在までの経緯として 2004 年夏以来の Wang らによる MD5, SHA-0, SHA-1 のアタックとその技術的詳細に関する報告、Joux, Biham, Chen による, SHA-0, SHA-1 のアタックについて説明した。これらのアタックの影響としては、NIST が推奨したことが無いにも関わらずいまだに広く用いられている MD5 に関しては早急に移行する必要があること、SHA-1 のアタックについてはそれほど実際的ではない（2^{69} 回の SHA-1 の演算が必要である）こと、緊急ではないもののより安全なものに移行する必要性があること、SHA-1 の寿命はほとんど尽きてしまったという認識を持っていること、SHA-1 に対する最近の結果は同じ組織が設計、標準化し同様の構造を有する SHA-256 の安全性に関する疑問にも繋がってくることを説明した。短期的な対応として SHA-256 と truncated SHA-256 への移行、SHA-256, -512 のセキュリティを再評価する必要性について説明し、問題点としては SHA-256 の安全性についての疑問、2 度は移行出来ないこと、MD5 と SHA-1 では状況が異なることが挙げられた。長期的な対応については、SHA-family の代替が現存するかどうかの検討、新しいアルゴリズムの開発の必要性があるか否かの検討の必要性があるという説明がなされた。

Panel discussion with Eli Biham, Nicolas T. Courtois, Bart Preneel, Ron Rivest, Rich Schroeppel,

Jerzy Urbanowicz

ハッシュ関数についての技術的側面、政治的側面について、質疑応答という形でパネルディスカッションが行われた。以下は要約である。

- 欧州においてはECRYPTが主に企業からの献金により成り立っているためアルゴリズムを一つに決めるという形の標準化は予定していない、とのPreneelによる発言があった。Preneelは、「本当にコンテストを行う予定は無いのか？」という追求に対して最後まで行うという発言は出なかった。
- NISTに関してはSchroeppel氏に対する「新たな標準化の予定は無いのか？」という質問に対し、本来必要であるが、予算の問題、人的稼動の問題から現状困難であるという回答がなされた。

2. ストリーム暗号の解読技術

2.1 各種アルゴリズムへの解読技術に関する新知見

[19]The Conditional Correlation Attack: A Practical Attack on Bluetooth Encryption

Yi Lu, Willi Meier and Serge Vaudenay

無線システムBluetooth（ブルートゥース）に用いられているE0というストリーム暗号の解読についての発表で、鍵の最初の24ビットを$2^{23.8}$フレームかつ2^{38}の計算量で推定が可能であるというものである。

[20]Linear Cryptanalysis of the TSC Family of Stream Ciphers

Frederic Muller and Thomas Peyrin

ShamirがT-functionをベースとしたT-functionを用いた特徴的な構造を持つstream cipherであるTSCに対する線形攻撃の会席を行った。TSC-1; $2^{21.4}$ data and $2^{25.4}$ time, TSC-2; $2^{44.1}$ data and $2^{48.1}$ time, TSC-3; processing 2^{42} output words and 2^{66} computation steps and 2^{34} known output words を必要とする。

[21]A Practical Attack on the Fixed RC4 in the WEP Mode

Itsik Mantin

RC4に対する、key recovery attackの提案。入力するIVの生成の仕方(master keyとの関係)

により攻撃しやすい場合がある。新たに提案した方法では、鍵の長さが伸びたとしてもそれを探索するのにかかる計算量はlinearにしか増えず、実用的な攻撃主張だと主張している。ただし、この攻撃が適用できる場合には制限があり、IVが攻撃に適した生成方法により作られた場合に限られる。IVを生成する際にmodeをセキュアなIVが生成できるものを用いればこの攻撃は避けることができる。

[22]Attack the Dragon.
Haa kan Englund and Alexander Maximov

ECRYPTで提案されたstream cipher のひとつであるDragon への安全性解析の結果を発表。Distinguishing attack を施し、2種攻撃手法を提案。1つめの方式では、$O(2^{187})$complexity and $O(2^{32})$memoryで実現できる。2つめの方式では、$O(2^{155})$complexityand $O(2^{96})$memoryで実現できる。

[23]Two algebraic attacks against the F-FCSRs using the IV mode.
Thierry P. Berger and Marine Minier

F-FCSR-SF1 mode, F-FCRS-DF1 mode, F-FCSR-SF8 mode, F-FCRS-DF8 mode に対して2種類のalgebraic attack解析を行った。

[24]Cryptanalysis of Keystream Generator by Decimated Sample Based Algebraic and Fast Correlation Attacks.
Miodrag J. Mihaljevic, Marc P.C. Fossorier and Hideki Imai

Algebraic attack, Correlation attackいずれにも演算量を効果に減少できる解析手法を提案。聴者の関心を集め、活発な意見交換が行われた。来年の会議中に本発表者によるLectureが企画されることとなった。

3. ブロック暗号 Camellia の安全性に関する新知見

3.1 AES の安全性に関する新知見

[25]An Analysis of the XSL Algorithm
Carlos Cid, and Gaetan Leurent

以前、AESに対してAlgebraic attack(代数的攻撃) が適用できるのではないかと言う見方があったが、本論文はそれに対する否定的な結果を示したものである。具体的には、この

手法による中間値の消去を考えた場合に、AESに対しては効率的な攻撃が不可能であることを示した。

[26]Cache attacks and countermeasures: the case of AES
Dag Arne Osvik, Adi Shamir and Eran Tromer

Software side-channel attack に分類される攻撃手法。１つの CPU 上にプロセスを２つ並行に走らせ、予め入力に対して秘密鍵を用いて暗号化が行われる際のメモリアクセスの分布と、実際の問題文を入力したときのメモリアクセスの分布の違いから秘密鍵情報を推定していく。攻撃には特別の装置を必要としない。また、暗号文・平文いずれか一方のみで実行できる攻撃手法である。実験例としては次のようなものがある。Open SSL 上で 300 暗号化・13ms で AES の Fullkey を recover。効率かつ攻撃に特別の機器などを要さない点は強力であるが想定する攻撃環境が現実で起こる場面はまだ少ないと考えられる。

[27]Higher Order Masking of the AES
Kai Schramm and Christof Paar

AES に対するサイドチャネル攻撃に関して、攻撃防止の為に行なわれるマスキングの効果の解析結果の発表。オーダーd の DPA 攻撃に対しては、d-1 のマスキングで防御することができる。ただし、その際のマスキングには注意を払う必要がある。オーダ 4 の DPA に対しては、オーダ 3 のマスキングでは解読されてしまう可能性があるが、解析を成功させるためには、数百万回のテストが必要となる。例として、8bitAVRsmartcard at 5MHz に対しては、368byte の RAM と 3164bytes の ROM とを用いて、0.50ms の解析時間を要する。High order の DPA に対するマスキングの効果に関する知見が得られた。

3.2　Camellia の安全性に関する新知見

[28]New Observation of Camellia
Duo Lei, Li Chao and Feng Keqin

電子政府推奨暗号のCamelliaの解読に関する講演で、Squareアタックを用いることで128ビット鍵の場合でFL関数の有り無しに関わらずフルラウンド18段中9段、256ビット鍵の場合でフルラウンド24段中11段（ＦＬ無しの場合)、10段（ＦＬ関数有りの場合)、の解読がそれぞれ可能であるというものである。なおこの結果はフルラウンドに比べてはるか少ない段数の場合の結果であり、電子政府推奨暗号Camelliaの安全性についての直接の影響は無い。

3.3　KASUMI の安全性に関する新知見

[29]A Related-Key Rectangle Attack on the Full KASUMI
Eli Biham, Orr Dunkelman and Nathan Keller

3GPPの標準アルゴリズムとなっているKASUMIに対して、related key attackを適用。攻撃手法の前提に強い仮定があるため、実質的な利用については本攻撃の影響はほとんど無いと思われるにもかかわらず、次世代アルゴリズムへの検討・移行が必要であろうと結論付けていたが、やや強引な結論であると思われる。

3.4　その他

[30]New Improvements of Davies-Murphy Cryptanalysis
Sebastien Kunz-Jacques and Frederic Muller

以前に提案されてあったDavis-Murphyの攻撃手法(DESに対するdedicated attack)の改良を行った。
Bi-linear attack としては最も効率のよい手法ではないが、DES の持つ構造的な特徴を解析し攻撃に用いたという点で意義があると主張している。2^{45} chosen plaintexts　を必要とする。

4. 公開鍵暗号の解読技術

[31]Partial Key Exposure Attacks on RSA up to Full Size Exponents
Matthias Ernst, Ellen Jochemsz, Alexander May and Benne de Weger

RSA に対する Wiener 攻撃（秘密鍵 d のサイズが $N^{0.25}$ 以下の場合に多項式時間で d を求めるアルゴリズムが存在する）と Boneh-Durfee-Frankel98 等のサイドチャネル攻撃によって秘密鍵の一部が漏れた場合の攻撃の組み合わせにより、秘密鍵 d を求めるために必要な e, d のサイズと d の漏洩ビット数の関係を示している。d が小さく、d の漏洩ビットが 0 の時は既知の Wiener 攻撃の結果と等しくなる。

[32]A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers
Johannes Blomer and Alexander May

Coppersmith が Eurocrypt96 で発表した２変数多項式 p(x,y)の整数解を求めるアルゴリズムに対し、Coppersmith98 で求められる解 $p(x_0, y_0)=0$ の大きさに関するバウンド $x_0 < X$ か

つ y_0<Y を組み合わせ論的に再考察し、バウンド(X,Y)を緩和する方法を示した。また、p^rq 型の合成数の素因数分解への適用として、全数探索型のアルゴリズムを提案している。ある推測値 p’ を選び、p(x,y) = (p’+x)^{r}y-N とおくと、(p-p’, q)が p(x,y)の解となることを利用して N が分解できるとしている。この際のバウンドは 以下のようになると述べている。

$$|p - p'| < N^{r/(r+1)^2}$$

[33]Another look at Small RSA Exponents

M. Jason Hinek

$ed \equiv 1 (\mathrm{mod}\ lcm(p-1, q-1))$ なる d が小さい場合に実行しうる攻撃について解析を行った。Continue fraction attack, lattice-based attack, factoring attack それぞれに関する解析し、現実的な範囲でのパラメータ評価を行った結果を示した。

5. ディジタル署名に関する新しい知見

[34]Efficient Designated Confirmer Signatures Without Random Oracles or General Zero-Knowledge Proofs

Craig Gentry, David Molnar and Zulfikar Ramzan

一般の署名構成からdesignated confirmer signature を構成する方法を提案。安全性の証明を行う際に新たにランダムオラクルやゼロ知識証明等の付加が必要でない。Paillier’s cryptosystemのCamenisch-Shoup 版 と Pederson commitmentsを活用して構成。

[35]Universally Convertible Directed Signatures

Fabien Laguillaumie, Pascal Paillier and Damien Vergnaud

新しいDirected signature (Confirmer signature と異なり署名の検証時にconfirmerではなくsignerが検証に関わる証明方式)の提案。Bilinear map の持つ特性を活用。さらに、不正な署名であった場合に追跡可能である一般の署名への変換可能なdirected signature の構成も提案。

[36]Short Undeniable Signatures Without Random Oracles: the Missing Link.

Fabien Laguillaumie and Damien Vergnaud

Boneh-Boyenの提案したshort signatureをベースに、新たなundeniable signature を提案。安全性証明をランダムオラクルを用いずに行っている。近年盛り上がっているBilinearの

構造の利用およびランダムとオラクルを用いない安全性証明の流れにのった論文。

[37]Short Threshold Signature Schemes without Random Oracles
Hong Wang, Yuqing Zhang and Dengguo Feng

Boneh-Boyenの提案したshort signatureを元に、Pedersonの提案したVSS(Verifiable Secret Sharing)を組み込んで、Threshold なshort signature を提案。安全性証明をランダムオラクルを用いずに行っている。

6. その他

[38]Errors in Computational Complexity Proofs for Protocols
Kim-Kwang Raymond Choo, Colin Boyd and Yvonne Hitchcock

一般に公開鍵系のプロトコルなどの安全性を証明する際に、計算量的安全性に基づき、攻撃を行なう際に必要となる計算量を用いて、対象となるプロトコルの安全性を議論する場合がある。本発表では、Key Agreement protocol において従来いくつか提案されているcomputational にprovable secureだとされている方式に対して、証明上の不備を指摘。Key Agreement Protocol の安全性を検討する上で配慮すべき点を提案。この結果から、これまで probable secure だとされている方法についても、安全性の証明などを見直す必要があると考えられる。

７．国内学会

7.1 ハッシュ関数、ストリーム暗号に関する研究動向

[39] MD4 Collision Attackの差分パスおよびSufficient Conditionについて
岩崎 輝星、下山 武司

MD4 Collision Attackの差分パスの探索とSufficient Conditionの決定方法を示し、その結果Round 2 およびRound 1の差分パスを変更することでCondition数の削減に成功し、実際に衝突を発見している。

[40] Improved Collision Attack on MD4
Yusuke Naito, Yu Sasaki, Noboru Kunihiro and Kazuo Ohta
(IACR Cryptology ePrint Archive 2005/151)
[41] Improved Collision Attack on MD5

Yu Sasaki, Yusuke Naito, Noboru Kunihiro and Kazuo Ohta
(IACR Cryptology ePrint Archive 2005/400)

[42] How to Construct Sufficient Condition in Searching Collisions of MD5
Yu Sasaki, Yusuke Naito, Jun Yajima, Takeshi Shimoyama, Noboru Kunihiro and Kazuo Ohta

差分パスが与えられている場合の Sufficient Condition の構成方法を与え、Wang らが与えた Sufficient Condition が実は余分な条件を含んでいることを示した。さらに実際に衝突を発見することにも成功した。

[43] MD5 のコリジョン探索における差分パスの構築法について-Wang の差分パスは最適か-
矢嶋 純、下山 武司、佐々木 悠、内藤 祐介、國廣 昇、太田 和夫

[44] How to Construct Sufficient Condition in Searching Collisions of MD5
Yu Sasaki, Yusuke Naito, Jun Yajima, Takeshi Shimoyama, Noboru Kunihiro and Kazuo Ohta
(IACR Cryptology ePrint Archive 2006/074)

MD5 のコリジョン探索で使用する差分パスについて検討を行い、Wang の差分パスが最適かどうか検討し、著者らの方法により得られた 26 個の差分パスのうちで Wang のパスは適切なものであることを示した。

[45] SHA-0 に対する Message Modification の考察
内藤 祐介、佐々木 悠、下山 武司、矢嶋 純、國廣 昇、太田 和夫

[46] Message Modification for Step 21-23 on SHA-0
Yusuke Naito, Yu Sasaki, Takeshi Shimoyama, Jun Yajima, Noboru Kunihiro and Kazuo Ohta
(IACR Cryptology ePrint Archive 2006/016)

Wang らの方式はステップ 20 までの全ての sufficient condition に対しては message modification を用いることにより全て確率 1 で満たすことが出来るが、21 ステップ以降の sufficient condition に対する message modification の検討は一切行われていなかった。この論文ではステップ 21 以降の message modification についての検討を行い、ステップ 21、ステップ 22 の sufficient condition をほぼ 1 で満たす message modification を用いることによりコリジョン探索の計算量が 2^{39} から 2^{37} に減少した。

[47] How to Find Wang's Differential Pattern for Cryptanalysis of SHA-1?
Makoto SUGITA and Hideki IMAI
[48] Advanced Message Modification Technique of SHA-1
Makoto SUGITA and Hideki IMAI
[49] Gröbner Basis Based Cryptanalysis of SHA-1
Makoto Sugita, Mitsuru Kawazoe and Hideki Imai
(IACR Cryptology ePrint Archive 2006/098)

Wang らによってアタックの詳細が完全には公開されていなかった 58 段 SHA-1 について、Wang のアタックが本質的に非線形符号の復号問題に帰着されていることを示し、誤り訂正符号の復号アルゴリズムとグレブナー基底アルゴリズムを適用することで、コリジョン探索の計算量を 2^{34} から 2^{29} に減少させることに成功し、実験的に 58 段 SHA-1 の衝突を発見することに成功した。

[50] MUGI の再同期攻撃に対する耐性評価(II)
山岡 孝二、金子 敏信

MUGI の再同期攻撃に対する耐性評価を行った。32bit truncate 線形特性を考察した結果、MUGI の最大線形特性確率の上界は 2^{-114} となり、2^{114} 組の初期ベクトルと出力列から鍵に関する内部状態の一部を導出できる可能性がある。ただし truncation による評価であるため、直ちに MUGI の安全性を脅かすものではない。

7.2 公開鍵系のアルゴリズム等に関する研究動向

[51]Universally Composable Blind Signatures
Seiju Doi, Yoshifumi Manabe and Tatsuaki Okamoto

Universal Composable Model で安全性証明可能なブラインド署名の新たな方式の提案。ブラインド署名を実現する為の基本的構成のアイディアは様々な応用が考えられる。本論文を元に展開した論文は TCC2006 で採録された。

[52]Parallel Key-Insulated Public Key Encryption
Goichiro Hanaoka, Yumiko Hanaoka and Hideki Imai

ID ベース暗号について、優位な機能を持つ暗号方式を提案。ID ベース暗号において、セキュアな処理を行う為に適当な期間ごとの秘密鍵のアップデートは有効的である。そこで本論文では、秘密鍵のアップデートを行なう仕組みについて、従来法に比べ適切な間隔でのセキュアなアップデートが可能な方式を提案している。
特に秘密鍵の更新の仕組みが安全性証明可能に構成できている点は新たな知見として意味深い。本論文を元に更

に展開した論文は、PKC2006 に採録された。

[53]Block Lanczos Algorithm Improved by Splitting the Matrix
Soh Takehide, Hidenori Kuwakado, Masakatu Mori and Hatsukazu Tanaka

素因数分解を行なう手法として数体ふるい法は有効的な手段として知られているが、数体ふるい法を行なう場合、巨大な行列演算を処理する必要がある。本論文は、数体ふるい法で必要となる行列演算部分の高速化に関する考察を行った。

[54]Design and Implementation of an ECM-based Integer Fanctorization Tool with GMP-ECM on Windows Platforms
Bin-Hui Chou, Chung-Huang Yang and Kouichi Sakurai

ウィンドウズプラットフォーム上での楕円曲線法による素因数分解の為のツールについてのデザイン及び実装に関する結果を発表。

[55]Implementing a Sieving Algorithm on a Dynamic Reconfigurable
Takeshi Shimoyama, Tetsuya Izu, Jun Kogure, Satoshi Nishimura and Kiyomitsu Kao

素因数分解を効率的に解く手法として知られる一般数体ふるい法について、ふるい部分の高速化を専用ハードウェアを用いて試み、実装を行なった結果に関する発表。

[56]A factoring algorithm using error functions
Kunikatsu Kobayashi and Susumu Fuji

素因数分解を効率的に解く手法として知られる 2 次ふるい法と同様に効率的に誤差関数を利用して素因数を分解する方法に関する考察結果に関する発表。

[57]An Evaluation of a Routing-based Dedicated Factoring Device for 1024-bit Integers
Naoyuki Hirota, Noboru Kunishiro, Tetsuya Izu and Kazuo Ohta

専用ハードウェアを用いて素因数分解を効率的に解く試みについて、従来提案さ

れている方法として TWIRL がある。しかし、提案論文では 768bit に対してのみ評価が行われており、1024-bit については評価されていなかった。そこで本論文では上記の方法による 1024-bit の場合に関する評価を行った結果を発表。

[58]On the choice of iteration function of the Pollard-rho factorization

Mika Uemura, Mitsuru Kawazoe and Tetsuya Takahashi

素因数分解を効率的に行なう手法として、ρ法が知られている。本論文ではρ法で用いる iteration function の選び方に関する考察結果を発表。

[59]Efficient and Secure Group Signatures in the Concurrent Joining Setting (Japanese Abstract Version)

Isamu Teranishi and Jun Furukawa

従来のグループ署名では、署名グループのメンバの追加は逐次的に行なわれることが仮定として想定されていた。より現実に近い現象を捉えるためには、署名グループメンバの追加が逐次的に行なわれなかったとしてもグループ署名の安全性が損なわれないことが望ましい。本論文では、グループのメンバをセキュアに同時並行的に行なうことができる方式の構成を提案。基本構成は、RSA とペアリングを利用している。
本論文を元に展開した論文は、FC06 の Short speech 部門に採録された。

[参照文献]

[1]Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, Xiuyuan Yu: Cryptanalysis of the Hash Functions MD4 and RIPEMD. EUROCRYPT 2005: 1-18
[2] Xiaoyun Wang, Hongbo Yu: How to Break MD5 and Other Hash Functions. EUROCRYPT 2005: 19-35
[3] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, William Jalby: Collisions of SHA-0 and Reduced SHA-1. EUROCRYPT 2005: 36-57
[4] Xiaoyun Wang: Strategies and Techniques of the Full SHA-1 Collision Attack. EUROCRYPT 2005: Rump session
[5] Xiaoyun Wang: What is the potential danger behind the collisions of hash functions? Conference on Hash Functions
[6] Eli Biham: Recent advances in hash functions and the way to go. RSA CONFERENCE 2005 JAPAN
[7] Magnus Daum: Finding Differential Patterns for the Wang Attack. Conference on Hash Functions
[8] Xiaoyun Wang, Hongbo Yu, Yiqun Lisa Yin: Efficient Collision Search Attacks on SHA-0. CRYPTO 2005: 1-16

[9] Xiaoyun Wang, Yiqun Lisa Yin, Hongbo Yu: Finding Collisions in the Full SHA-1. CRYPTO 2005: 17-36
[10] Xiaoyun Wang, Andrew Yao and Frances Yao: New Collision Search for SHA-1. CRYPTO 2005: Rump session
[11] Hongbo Yu, Gaoli Wang, Guoyan Zhang, Xiaoyun Wang: The Second-Preimage Attack on MD4. CANS 2005: 1-12
[12] Xiaoyun Wang: Cryptanalysis for Hash functions and Some Potential Dangers. CT-RSA 2006: Invited Talk
[13] Bart Preneel: Hash functions - present state of art. Conference on Hash Functions
[14] Lars R. Knudsen, Frédéric Muller: Some Attacks Against a Double Length Hash Proposal. ASIACRYPT 2005: 462-473
[15] Stefan Lucks: A Failure-Friendly Design Principle for Hash Functions. ASIACRYPT 2005: 474-494
[16] Deukjo Hong, Jaechul Sung, Seokhie Hong, Sangjin Lee: How to Construct Universal One-Way Hash Functions of Order *r*. INDOCRYPT 2005: 63-76
[17] Mridul Nandi: Towards Optimal Double-Length Hash Functions. INDOCRYPT 2005: 77-89
[18] Michael Szydlo, Yiqun Lisa Yin: Collision-Resistant Usage of MD5 and SHA-1 Via Message Preprocessing. CT-RSA 2006: 99-114
[19] Yi Lu, Willi Meier, Serge Vaudenay: The Conditional Correlation Attack: A Practical Attack on Bluetooth Encryption. CRYPTO 2005: 97-117
[20] Frédéric Muller, Thomas Peyrin: Linear Cryptanalysis of the TSC Family of Stream Ciphers. ASIACRYPT 2005: 373-394
[21] Itsik Mantin: A Practical Attack on the Fixed RC4 in the WEP Mode. ASIACRYPT 2005: 395-411
[22] Håkan Englund, Alexander Maximov: Attack the Dragon. INDOCRYPT 2005: 130-142
[23] Thierry P. Berger, Marine Minier: Two Algebraic Attacks Against the F-FCSRs Using the IV Mode. INDOCRYPT 2005: 143-154
[24] Miodrag J. Mihaljevic, Marc P. C. Fossorier, Hideki Imai: Cryptanalysis of Keystream Generator by Decimated Sample Based Algebraic and Fast Correlation Attacks. INDOCRYPT 2005: 155-168
[25] Carlos Cid, Gaëtan Leurent: An Analysis of the XSL Algorithm. ASIACRYPT 2005: 333-352
[26] Dag Arne Osvik, Adi Shamir, Eran Tromer: Cache Attacks and Countermeasures: The Case of AES. CT-RSA 2006: 1-20
[27] Kai Schramm, Christof Paar: Higher Order Masking of the AES. CT-RSA 2006:

208-225
[28] Duo Lei, Li Chao, Keqin Feng: New Observation on Camellia. Selected Areas in Cryptography 2005: 51-64
[29] Eli Biham, Orr Dunkelman, Nathan Keller: A Related-Key Rectangle Attack on the Full KASUMI. ASIACRYPT 2005: 443-461
[30] Sébastien Kunz-Jacques, Frédéric Muller: New Improvements of Davies-Murphy Cryptanalysis. ASIACRYPT 2005: 425-442
[31] Matthias Ernst, Ellen Jochemsz, Alexander May, Benne de Weger: Partial Key Exposure Attacks on RSA up to Full Size Exponents. EUROCRYPT 2005: 371-386
[32] Johannes Blömer, Alexander May: A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers. EUROCRYPT 2005: 251-267
[33] M. Jason Hinek: Another Look at Small RSA Exponents. CT-RSA 2006: 82-98
[34] Craig Gentry, David Molnar, Zulfikar Ramzan: Efficient Designated Confirmer Signatures Without Random Oracles or General Zero-Knowledge Proofs. ASIACRYPT 2005: 662-681
[35] Fabien Laguillaumie, Pascal Paillier, Damien Vergnaud: Universally Convertible Directed Signatures. ASIACRYPT 2005: 682-701
[36] Fabien Laguillaumie, Damien Vergnaud: Short Undeniable Signatures Without Random Oracles: The Missing Link. INDOCRYPT 2005: 283-296
[37] Hong Wang, Yuqing Zhang, Dengguo Feng: Short Threshold Signature Schemes Without Random Oracles. INDOCRYPT 2005: 297-310
[38] Kim-Kwang Raymond Choo, Colin Boyd, Yvonne Hitchcock: Errors in Computational Complexity Proofs for Protocols. ASIACRYPT 2005: 624-643
[39] 岩崎 輝星、下山 武司: MD4 Collision Attackの差分パスおよびSufficient Conditionについて. SCIS2006(国内会議)
[40] Yusuke Naito, Yu Sasaki, Noboru Kunihiro and Kazuo Ohta: Improved Collision Attack on MD4. IACR Cryptology ePrint Archive 2005/151
[41] Yu Sasaki, Yusuke Naito, Noboru Kunihiro and Kazuo Ohta: Improved Collision Attack on MD5. IACR Cryptology ePrint Archive 2005/400
[42] Yu Sasaki, Yusuke Naito, Jun Yajima, Takeshi Shimoyama, Noboru Kunihiro and Kazuo Ohta: How to Construct Sufficient Condition in Searching Collisions of MD5. SCIS2006(国内会議)
[43] 矢嶋 純、下山 武司、佐々木 悠、内藤 祐介、國廣 昇、太田 和夫: MD5のコリジョン探索における差分パスの構築法について - Wangの差分パスは最適か - SCIS2006(国内会議)
[44] Yu Sasaki, Yusuke Naito, Jun Yajima, Takeshi Shimoyama, Noboru Kunihiro and Kazuo Ohta: How to Construct Sufficient Condition in Searching Collisions of MD5. IACR Cryptology ePrint Archive 2006/074

[45] 内藤 祐介、佐々木 悠、下山 武司、矢嶋 純、國廣 昇、太田 和夫: SHA-0に対するMessage Modificationの考察. SCIS2006(国内会議)
[46] Yusuke Naito, Yu Sasaki, Takeshi Shimoyama, Jun Yajima, Noboru Kunihiro and Kazuo Ohta: Message Modification for Step 21-23 on SHA-0. IACR Cryptology ePrint Archive 2006/016
[47] Makoto SUGITA and Hideki IMAI: How to Find Wang's Differential Pattern for Cryptanalysis of SHA-1? SCIS2006(国内会議)
[48] Makoto SUGITA and Hideki IMAI: Advanced Message Modification Technique of SHA-1. SCIS2006(国内会議)
[49] Makoto Sugita, Mitsuru Kawazoe and Hideki Imai: Gröbner Basis Based Cryptanalysis of SHA-1. IACR Cryptology ePrint Archive 2006/098
[50] 山岡 孝二、金子 敏信: MUGIの再同期攻撃に対する耐性評価(II). SCIS2006(国内会議)
[51] Seiju Doi, Yoshifumi Manabe and Tatsuaki Okamoto: Universally Composable Blind Signatures. SCIS2006(国内会議)
[52] Goichiro Hanaoka, Yumiko Hanaoka and Hideki Imai: Parallel Key-Insulated Public Key Encryption. SCIS2006(国内会議)
[53] Soh Takehide, Hidenori Kuwakado, Masakatu Mori and Hatsukazu Tanaka: Block Lanczos Algorithm Improved by Splitting the Matrix. SCIS2006(国内会議)
[54] Bin-Hui Chou, Chung-Huang Yang and Kouichi Sakurai: Design and Implementation of an ECM-based Integer Fanctorization Tool with GMP-ECM on Windows Platforms. SCIS2006(国内会議)
[55] Takeshi Shimoyama, Tetsuya Izu, Jun Kogure, Satoshi Nishimura and Kiyomitsu Kao: Implementing a Sieving Algorithm on a Dynamic Reconfigurable. SCIS2006(国内会議)
[56] Kunikatsu Kobayashi and Susumu Fuji: A factoring algorithm using error functions. SCIS2006(国内会議)
[57] Naoyuki Hirota, Noboru Kunishiro, Tetsuya Izu and Kazuo Ohta: An Evaluation of a Routing-based Dedicated Factoring Device for 1024-bit Integers. SCIS2006(国内会議)
[58] Mika Uemura, Mitsuru Kawazoe and Tetsuya Takahashi: On the choice of iteration function of the Pollard-rho factorization. SCIS2006(国内会議)
[59] Isamu Teranishi and Jun Furukawa: Efficient and Secure Group Signatures in the Concurrent Joining Setting(Japanese Abstract Version). SCIS2006(国内会議)

付録４

ハッシュ関数の安全性に関する技術調査報告

平成18年 2月 (平成18年 5月改訂)

ハッシュ関数・暗号利用モード調査WG

ハッシュ関数・暗号利用モード調査 WG 委員構成

主査：古原 和邦　東京大学生産技術研究所

委員：廣瀬 勝一　福井大学工学部

委員：川村 信一　株式会社東芝

委員：古屋 聡一　株式会社日立製作所

委員：盛合 志帆　株式会社ソニー・コンピュータエンタテインメント

目 次

第1章　はじめに

ハッシュ関数は、任意長の入力メッセージを固定長のメッセージダイジェスト (ハッシュ値) に圧縮する関数である。特に、一方向性 (one-wayness) と衝突発見困難性 (collision-resistance) を満たすハッシュ関数のことを暗号学的ハッシュ関数と呼ぶ。

ハッシュ関数には、鍵を入力として与えない「鍵無しハッシュ関数」(unkeyed hash function) と、鍵を入力として与える「鍵付きハッシュ関数」(keyed hash function) があるが、本報告書では鍵無しハッシュ関数のみを扱う。鍵付きハッシュ関数の代表例としてメッセージ認証コード (message authentication code) があるが、これについては、2003 年度 CRYPTREC 活動にて暗号技術監視委員会 暗号技術調査 WG で調査報告済みである [C03a]。

ハッシュ関数の代表的な構成法として、ブロック暗号を用いる構成法、ハッシュ関数専用の構成法 (dedicated design) 剰余演算 (modular arithmetic) を用いた構成法などがあるが [ISO/IEC10118-1]、本報告書では、世の中で最も広く使われている、ハッシュ関数専用の構成法に基づいて構成された「専用ハッシュ関数」を中心に技術調査を行う。

本報告書の構成　本報告書では、まず、2 章「ハッシュ関数の基礎」において、ハッシュ関数の分類や満たすべき性質、代表的な構成法、ハッシュ関数に対する攻撃法、及びハッシュ関数の用途について述べる。

次に、3 章「代表的なハッシュ関数」において、世の中で広く使われているハッシュ関数の概要、技術的特徴、仕様及び採用・標準化動向について述べる。ここで挙げるハッシュ関数はインターネット標準である RFC、国際標準規格 ISO/IEC 10118-3、米国連邦政府標準規格 FIPS180-2、そして電子政府推奨暗号リストに掲載されているアルゴリズムをカバーしている。

4 章「安全性解析」では、3 章で挙げたハッシュ関数についてこれまでに知られている解析結果をまとめる。

5 章「ソフトウェア実装性能」では、3 章で挙げたハッシュ関数について、これまでに報告されているいくつかのソフトウェア実装性能数値を示し、6 章でまとめる。

第2章　ハッシュ関数の基礎

ハッシュ関数は、任意長の入力メッセージを固定長のメッセージダイジェスト (ハッシュ値) に圧縮する関数である。ハッシュ関数は、ハッシュ値の計算は容易であるが、もとに戻すのは困難であるという特徴をもつ。

特に、**一方向性** (one-wayness) と**衝突発見困難性** (collision-resistance) を満たすハッシュ関数を暗号学的ハッシュ関数 (cryptographic hash function) と呼ぶ。以降、本報告書では、暗号学的ハッシュ関数を単に「ハッシュ関数」と呼ぶ。

2.1　ハッシュ関数の分類

ハッシュ関数には、鍵を入力として与えない**鍵無しハッシュ関数** (unkeyed hash function) と鍵を入力として与える**鍵付きハッシュ関数** (keyed hash function) がある (図 2.1 参照)。

鍵無しハッシュ関数の代表的な目的は、メッセージの改竄を検知することであり、改竄検知コード (MDC: Modification/Manipulation Detection Code) 又はメッセージ完全性コード (MIC: Message Integrity Code) と呼ばれる関数が代表的である。

このうち、特に**原像計算困難性** (pre-image resistance)、**第二原像計算困難性** (second pre-image resistance) (2.2 章参照) を満たす**一方向性ハッシュ関数** (one-way hash function)、**衝突発見困難性** (collision resistance)、**弱衝突発見困難性** (weak collision resistance) (2.2 章参照) を満たす**衝突困難ハッシュ関数** (collision resistant hash function) が重要である。

一方、鍵付きハッシュ関数の代表的な目的は、メッセージのソースを認証し、かつ改竄を検知することであり、**メッセージ認証コード** (MAC: Message Authentication Code) が代表的である。これについては 2003 年度 CRYPTREC 活動にて暗号技術監視委員会 暗号技術調査 WG で調査報告を行っている [C03a]。

本報告書では鍵無しハッシュ関数を対象に調査を行う。

2.2　ハッシュ関数の満たすべき性質

(暗号学的) ハッシュ関数 H は、以下の 3 つの性質を満たしていることが望まれる。

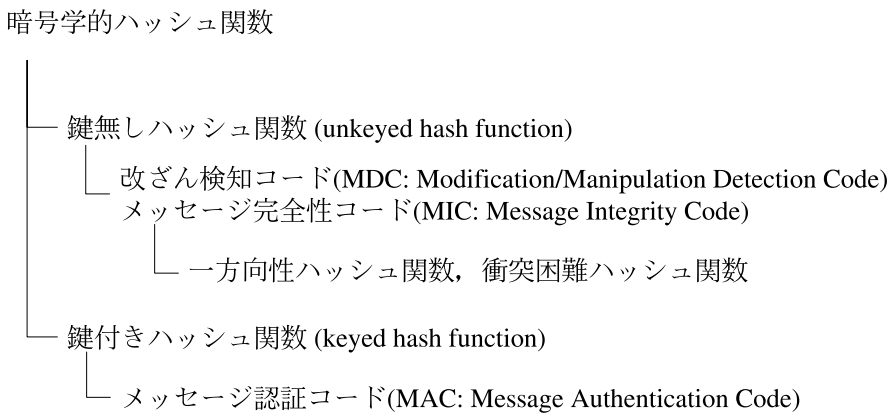


図 2.1: ハッシュ関数の分類

衝突発見困難性 (Collision Resistance)　ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすような異なる 2 つのメッセージ M と M' を見つけることが計算量的に困難である。

特に、ハッシュ値 $H(M)$ と $H(M')$ の値が、わずかのビット数 (例えば 1, 2 ビット) のみしか異ならない 2 つのメッセージ M と M' を見つけることが計算量的に困難である性質のことを**近似衝突発見困難性** (near-collision resistance) [MOV97] と呼んでいる。

原像計算困難性 (Pre-image Resistance)　ある未知のメッセージ M に対するハッシュ値が与えられた時、ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすようなメッセージ M' を見つけることが計算量的に困難である。この性質のことを**一方向性** (one-wayness) ともいう。

第二原像計算困難性 (Second Pre-image Resistance)　ある既知のメッセージ M と M に対するハッシュ値が与えられた時、ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすような別のメッセージ $M'(\neq M)$ を見つけることが計算量的に困難である。この性質のことを**弱衝突発見困難性** (weak collision resistance) ともいう。

2.3　ハッシュ関数の構成

2.3.1　繰り返し型ハッシュ関数

ハッシュ関数 $H : \{0,1\}^* \to \{0,1\}^n$ は通常、入出力が固定長の**圧縮関数** (compression function) $f : \{0,1\}^{n+m} \to \{0,1\}^n$ を繰り返し適用することで計算される。こ

のようなハッシュ関数は、**繰り返し型ハッシュ関数** (iterated hash function) または Merkle-Damgård 法 (MD 法)[M89, D89] と呼ばれる。

繰り返し型ハッシュ関数では

(1) パディング

(2) 固定長への分割

(3) 圧縮関数による繰り返し演算

(4) 出力

の順序で実行される。

(1) パディング　パディング処理では、メッセージ長が m ビットの倍数になるように入力メッセージ M の末尾に下記のようにデータが付加される方法が代表的である。

$$M||1||0^k||l$$

但し、l は M のメッセージ長をバイナリ表現した時のビット数である。このように、末尾にメッセージ長を付加するパディング法を Merkle-Damgård strengthening (MD-strengthening) と呼ぶ。

(2) 固定長への分割　次にメッセージは m ビットの固定長に分割される。m ビットの固定長に分割されたメッセージを $M = (M^{(1)}||M^{(2)}||\cdots||M^{(N)}) \in (\{0,1\}^m)^*$ とする。但し $|M^{(i)}| = m$ である。

(3) 圧縮関数による繰り返し演算　m ビットの固定長に分割されたメッセージ $M^{(i)}$ が順次、圧縮関数へ入力される。

(4) 出力　最後の圧縮関数からの出力に対し、出力変換関数 g を施した結果をハッシュ値 H とする。出力変換関数 g がない (恒等変換である) 場合もある。

圧縮関数 f と出力変換関数 g を用いたハッシュ関数 $H^{fg}(H_0,\cdot) : (\{0,1\}^m)^* \to \{0,1\}^n$ は以下のように定義される (図 2.2 参照)。

定義 2.3.1 (繰り返し型ハッシュ関数)**.**

$$H^{fg}(H_0,\cdot) : (\{0,1\}^m)^* \to \{0,1\}^n$$

For $i = 1$ to N
　$H_i = f(H_{i-1}, M^{(i)})$　　f : 圧縮関数
$H = g(H_N)$　　g : 出力変換関数
Return H.

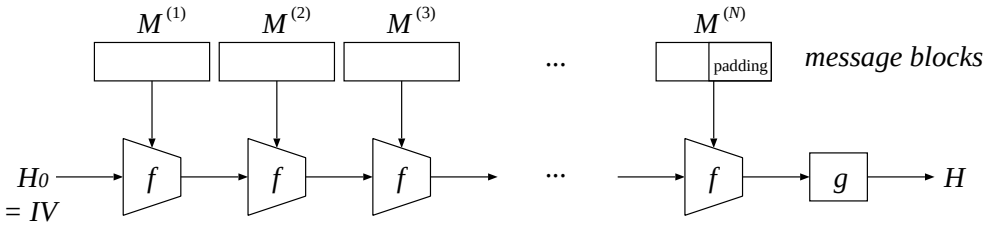


図 2.2: 繰り返し型ハッシュ関数

圧縮関数の構成法は下記のように分類することができる [ISO/IEC10118-2, ISO/IEC10118-3, ISO/IEC10118-4]。

- ブロック暗号に基づく構成法
- ハッシュ関数専用の構成法 (dedicated design)
- 剰余演算 (modular arithmetic) に基づく構成法

2.3.2 ブロック暗号に基づく構成法

ブロック暗号を圧縮関数の構成要素としてハッシュ関数を構成する場合、次のような利点が考えられる。

- 設計、評価、実装にかかるコストを削減できる
- コンパクトに実装可能

ブロック暗号に基づくハッシュ関数には、ハッシュ関数の出力長 (ハッシュ長) がブロック暗号のブロック長と同じ長さのものと、ブロック暗号のブロック長の 2 倍の長さのものがある。これらをそれぞれ、**単ブロック長ハッシュ関数** (single block length hash function)、**倍ブロック長ハッシュ関数** (double block length hash function) と呼ぶ。

単ブロック長ハッシュ関数 (Single Block Length Hash Function) ブロック暗号 1 回当たりでハッシュされるメッセージのブロック数をレート (rate) と定義する。単ブロック長ハッシュ関数は、レート 1 の安全性が証明されたスキームが 12 種類存在することが知られており、そのうち 1 つが ISO/IEC 10118-2 に規定されている (図 2.3 参照)。単ブロック長ハッシュ関数で用いられるブロック暗号のブロック長を m とすると、この方式に対する衝突攻撃 (collision attack) の計算量は $\Omega(2^{m/2})$、(別) 原像探索攻撃 ((2nd) pre-image attack) の計算量は $\Omega(2^m)$ である。

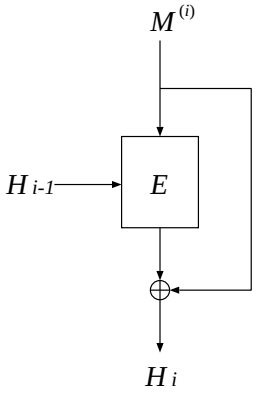


図 2.3: 単ブロック長ハッシュ関数の例

倍ブロック長ハッシュ関数 (Double Block Length Hash Function) 倍ブロック長ハッシュ関数には多くのスキームが知られており、その中で、DES ベースの 3 種類の倍ブロック長ハッシュ関数がブラックボックスモデルで衝突攻撃に対して最良の安全性をもつ (攻撃計算量が $\Omega(2^{m/2})$ である) ことが示されている [M89]。しかしながら、これらのスキームのレートは 0.276 であり、あまり効率的ではない。

衝突攻撃に対して最良の安全性をもち、よりレートの高い倍ブロック長ハッシュ関数が課題となっているが、レート 0.5 で、衝突攻撃に対して最良の安全性をもつ方式が廣瀬 [H04] らにより示されている。この方式は、鍵長がブロック長の 2 倍のブロック暗号を用いたものである。

一方、廣瀬、服部により [HH05]、レート 1 で、鍵長がブロック長の 2 倍のブロック暗号を用いた方式では、圧縮関数に計算量 $O(2^{m/2})$ の (free-start) 衝突攻撃が存在することが示された。衝突攻撃に対する最良の安全性 (計算量が $\Omega(2^{m/2})$) をもつレート 1 の倍ブロック長ハッシュ関数が存在するかはまだ未解決問題である。

2.3.3 ハッシュ関数専用構成法

ハッシュ関数専用の構成法 (dedicated design) によるハッシュ関数 (以下、専用ハッシュ関数と記す) は、表 2.1 に示すように多くのアルゴリズムが存在する。

このうち、Rivest が設計した MD4、その強度を高めた MD5 は、その後の専用ハッシュ関数の設計に大きな影響を与えた (図 2.4 参照)。

RIPEMD は 1992 年にヨーロッパの RIPE (Race Integrity Primitive Evaluation) プロジェクトの成果の一つとして提案されたハッシュ関数である [RIPE92]。RIPEMD は、MD4, 及びそれを 256 ビットハッシュ値を出力するように拡張された Extended MD4[R90] をもとに設計された。RIPEMD-160 及び RIPEMD-128 は RIPEMD の流れをくみ、さらに強度を高める目的で設計された。

HAVAL も MD4, MD5 の改良アルゴリズムで、ハッシュ長 (128, 160, 192, 224, 256 ビット) 及び処理ステップ数が可変にできる特徴をもつ。

表 2.1: さまざまな専用ハッシュ関数

名称	提案者 (機関)	提案年	ブロック長 (bit)	ハッシュ長 (bit)
MD2	Rivest	1989	512	128
MD4	Rivest	1990	512	128
MD5	Rivest	1991	512	128
RIPEMD	The RIPE Consortium	1992	512	128
RIPEMD-128	Dobertin, Bosselaers, Preneel	1996	512	128
RIPEMD-160	Dobertin, Bosselaers, Preneel	1996	512	160
HAVAL	Zheng, Pieprzyk, Seberry	1992	1024	128, 160, 192 224, 256
SHA (SHA-0)	NIST/NSA	1993	512	160
SHA-1	NIST/NSA	1995	512	160
SHA-224	NIST/NSA	2004	512	224
SHA-256	NIST/NSA	2002	512	256
SHA-384	NIST/NSA	2002	1024	384
SHA-512	NIST/NSA	2002	1024	512
Tiger	Anderson, Biham	1996	512	192
Whirlpool	Barretto, Rijmen	2000	512	512

SHA (SHA-0) 及び SHA-1 も、MD4 及び MD5 をベースに設計されたハッシュ関数である。安全性を向上するために、ハッシュ長を、当時主流であった 128 ビットから 160 ビットに拡大し、メッセージスケジュール関数が導入された。SHA-1 は SHA-0 のメッセージスケジュール関数に若干の仕様変更 (1 ビットローテーションの追加) がなされたアルゴリズムで、米国政府標準のハッシュ関数、ならびにデファクトスタンダードとして広く利用されている。さらに近年、ハッシュ長の長い SHA-224, SHA-256, SHA-384, SHA-512 も提案されている。

MD4, MD5 が 32 ビットアーキテクチャ上での高速なソフトウェア実装速度を意識して設計されたのに対し、1996 年に設計された Tiger は 64 ビットアーキテクチャを意識して設計されている。

2000 年に発表された Whirlpool は、AES の設計指針の一つである Wide Trail strategy をもとに設計されたハッシュ長 512 ビットのハッシュ関数で、メッセージ長が 2^{256} ビットまでの長い入力メッセージを扱えることが特徴である。Whirlpool は 2004 年に改訂された ISO/IEC 10118-3 に採用されている。

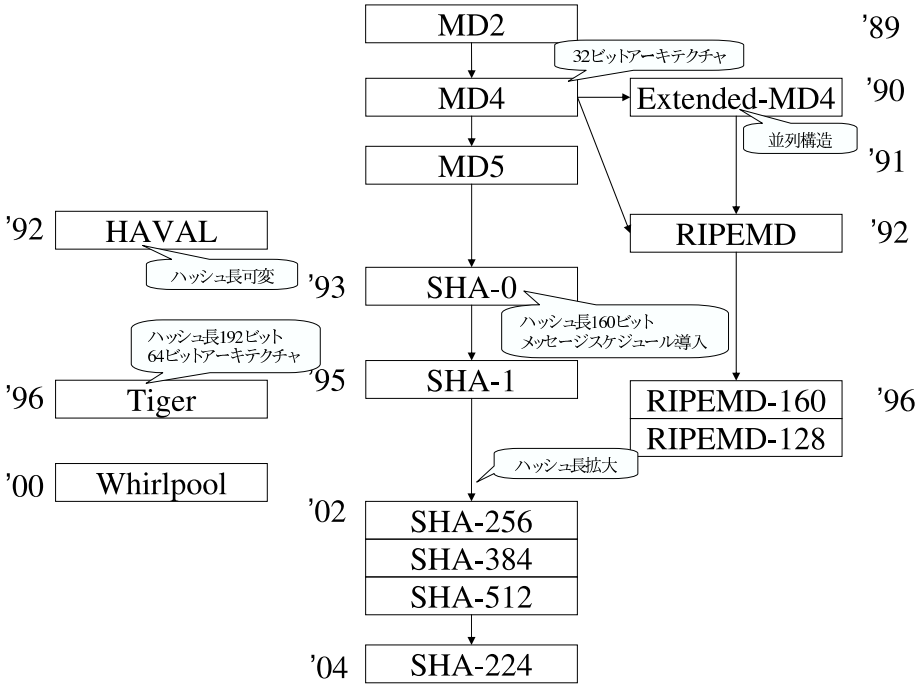


図 2.4: 専用ハッシュ関数の系譜

2.3.4 剰余演算に基づく構成法

代数構造を利用したハッシュ関数には、これまでに剰余演算 (modular arithmetic) を用いるものやナップザック問題に帰着されるものが提案されている。

加法型ナップザック問題 (additive knapsacks) に基づく構成法は、ハッシュ関数として実用的なパラメータではナップザック問題が解けてしまうのではという懸念があり、乗法型ナップザック問題 (multiplicative knapsacks) に基づく構成法は、衝突攻撃に対する安全性の証明が可能な方式もあるが、その代数構造を利用した攻撃があるのではという懸念があるなどの理由でほとんど使われていない。

剰余演算に基づくハッシュ関数としては、これまでにいろいろな方式が提案されたが、破られた方式も存在する。証明可能安全性をもつ方式もいくつか存在するが、速度が遅く、あまり実用的ではない。ここでは、ISO/IEC 10118-4 [ISO/IEC10118-4] に採用されている MASH-1, MASH-2 (Modular Arithmetic Secure Hash) について示す。

MASH-1

$$H_i = ((((m_i \oplus h_{i-1}) \vee A)^2 \pmod N) \dashv n) \oplus H_{i-1}$$

但し、$A = \texttt{F00...00}$ の定数、N は $N = pq$ (p, q はランダムに選ばれた秘密の素数) なる法である。また、$\dashv n$ は $|N|$ ビットデータの右 n ビット部分のみを左寄せする

(truncate する) 演算を示す。

MASH-2

$$H_i = ((((m_i \oplus h_{i-1}) \vee A)^{2^8+1} \pmod N) \dashv n) \oplus H_{i-1}$$

MASH-2 は MASH-1 の冪乗剰余演算の指数のみが異なる (2 が 2^8+1 となっている) アルゴリズムである。n ビットの法に対し、原像探索攻撃の計算量は $\Omega(2^{n/2})$、衝突攻撃の計算量は $\Omega(2^{n/4})$ であることが知られている。

2.4 ハッシュ関数に対する攻撃法

ハッシュ関数に対する汎用の攻撃 (generic attacks) としては、2.2 章で挙げた各項目に対応する攻撃が存在する。ハッシュ関数 $H : \{0,1\}^* \to \{0,1\}^n$ のそれぞれの攻撃に対する強度には上限が存在し、その攻撃計算量の上限はハッシュ長 n にのみ依存する。

衝突攻撃 (Collision Attack) ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすような異なる 2 つのメッセージ M と M' を探索する攻撃。攻撃計算量は n ビットデータに対する Birthday Attack の計算量 $\Omega(2^{n/2})$ となる。

また、2 つのハッシュ値 $H(M)$ 及び $H(M')$ が、わずかのビット位置を除いて一致するような衝突を、**近似衝突 (near-collision)** という。また、繰り返し型ハッシュ関数において、初期ベクトル IV を任意に選べる条件下でハッシュ値が一致する、すなわち

$$H(IV, M) = H(IV', M')$$

となるような衝突を、**擬似衝突 (pseudo-collision)** という。

原像探索攻撃 (Pre-image Attack) ある未知のメッセージ M に対するハッシュ値が与えられた時、ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすようなメッセージ M' を探索する攻撃。攻撃計算量は n ビットデータに対する全数探索の計算量 $\Omega(2^n)$ となる。

第二原像探索攻撃 (Second Pre-image Attack) ある既知のメッセージ M と M に対するハッシュ値が与えられた時、ハッシュ値が一致する、すなわち $H(M) = H(M')$ を満たすような別のメッセージ $M'(\neq M)$ を探索する攻撃。攻撃計算量は n ビットデータに対する全数探索の計算量 $\Omega(2^n)$ となる。

2.5 ハッシュ関数の利用

ハッシュ関数は、メッセージのダイジェストを計算するという目的の他に、暗号スキームまたは暗号アルゴリズムの構成要素として利用されることが多い。ハッシュ関数の用途としては下記のようなものが代表的である。

- デジタル署名 (ほぼ全てのアルゴリズム)
- 公開鍵暗号 (例: RSA-OAEP, RSAES-PKCS1-v1_5 などのスキーム)
- 擬似乱数生成器 (例: FIPS 186-2)
- メッセージ認証コード (例: HMAC)
- ブロック暗号 (例: SHACAL-2, BEAR, LION)
- ストリーム暗号 (例: SEAL)

ハッシュ関数の入力メッセージに変更を加えると、極めて高い確率で異なるメッセージダイジェスト（ハッシュ値）が出力される。ハッシュ関数を用いたメッセージ認証コードやデジタル署名ではこれを利用して認証（メッセージの改竄検出）を行うことができる。

また、電子政府推奨暗号リスト中で利用されるハッシュ関数を表 2.2 に示す。

表 2.2: 電子政府推奨暗号リスト中で利用されるハッシュ関数

技術分類		名称	ハッシュ関数
公開鍵暗号	デジタル署名	DSA	SHA-1
		ECDSA	指定なし
		RSASSA-PKCS1-v1_5	MD5 注)
		RSAPSS	指定なし
	守秘	RSA-OAEP	指定なし
	鍵共有	PSEC-KEM	SHA-1
擬似乱数生成系		PRNG based on SHA-1 in ANSI X9.42-2001 Annex C.1	SHA-1
		PRNG based on SHA-1 for general purpose in FIPS 186-2 (+change notice 1) Appendix 3.1	SHA-1
		PRNG based on SHA-1 for general purpose in FIPS 186-2 (+change notice 1) revised Appendix 3.1	SHA-1

注) RSASSA-PKCS1-v1_5(RSA 署名, 電子署名法に係る指針に記載された方式) は、PKCS#1 v1.5 で規定され、PKCS#1 v2.1 にも引き継がれている。

1. PKCS#1 v1.5 には、MD5 についての記述 (OID) はあるが、SHA-1 についての記述 (OID) はない。
2. PKCS#1 v2.0 には、EMSA-PKCS1-v1_5 エンコーディング手法として、新たに SHA-1 が利用できるように OID が記述されている。
3. PKCS#1 v2.1 には、EMSA-PKCS1-v1_5 エンコーディング手法として、新たに SHA-256, 384, 512, 224 が利用できるように OID が記述されている。

第3章　代表的なハッシュ関数

本章では、代表的なハッシュ関数として MD4, MD5, RIPEMD, RIPEMD-128, RIPEMD-160, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 について、その概要と技術仕様について示す。

代表的なハッシュ関数の特徴　表 3.1 に代表的なハッシュ関数の特徴をまとめる。

表 3.1: 代表的なハッシュ関数の特徴

名称	ハッシュ長 (bit)	メッセージ長 (bit)	ブロック長 (bit)	エンディアン	標準
MD4	128	上限なし	512	little	RFC 1320
MD5	128	上限なし	512	little	RFC 1321
RIPEMD	128	上限なし	512	little	
RIPEMD-128	128	$< 2^{64}$	512	little	ISO/IEC 10118-3
RIPEMD-160	160	$< 2^{64}$	512	little	電子政府推奨暗号 注) ISO/IEC 10118-3
SHA-1	160	$< 2^{64}$	512	big	電子政府推奨暗号 注) FIPS 180-2 ISO/IEC 10118-3
SHA-224	224	$< 2^{64}$	512	big	FIPS 180-2 Change Notice 1
SHA-256	256	$< 2^{64}$	512	big	電子政府推奨暗号 FIPS 180-2 NESSIE Portfolio ISO/IEC 10118-3
SHA-384/512	384/512	$< 2^{128}$	1024	big	電子政府推奨暗号 FIPS 180-2 NESSIE Portfolio ISO/IEC 10118-3
Whirlpool	512	$< 2^{256}$	512	neutral	NESSIE Portfolio ISO/IEC 10118-3

注) 電子政府推奨暗号リストにおいて、RIPEMD-160, SHA-1 については「新たな電子政府用システムを構築する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。」という注釈がついている。

本章で用いる記号　表 3.1 に本章で用いる記号を定義する。

表 3.2: 記号の定義

記号	定義
$+$	ワード単位毎の算術加算
$\wedge$	ビット毎の論理積
$\vee$	ビット毎の論理和
$\oplus$	ビット毎の排他的論理和
$\neg x$	x のビット反転
$\mathrm{ROTR}^n(x)$	x の n ビット右巡回シフト
$\mathrm{ROTL}^n(x)$	x の n ビット左巡回シフト
$\mathrm{SHR}^n(x)$	x の n ビット右シフト

3.1 MD4

3.1.1 概要

MD4 は 1990 年に Rivest によって提案されたハッシュ関数である [R90]。Rivest は MD2, MD4, MD5 などの一連のハッシュ関数を提案したが、その後の専用ハッシュ関数の設計に特に大きな影響を与えたのが MD4 である。

MD4 は、ビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として 128 ビットのハッシュ値を出力する。演算は 32 ビット単位で行われ、全体は 3 ラウンド ×16 ステップで構成されている。

MD4 はソフトウェア、特に 32 ビットアーキテクチャで高速に実装できるよう、32 ビットワード演算を多用して設計されている。また、Intel 80xxx プロセッサを意識し、little-endian になっている[1]。

MD4 のアルゴリズムを記述した RFC1320 が 1992 年にリリースされたが [R92a]、1996 年に Dobbertin[D96a] により容易に衝突が発見できることが示されてから、利用は控えられている。

3.1.2 技術仕様

MD4 で使用される関数

MD4 では、32 ビットワードを入力変数および出力変数とする、以下の論理関数 $f_0, f_1, \ldots, f_{47}$ が用いられる。

$$f_t(x,y,z) = \begin{cases} (x \wedge y) \vee (\neg x \wedge z) & (0 \leq t \leq 15) \\ (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) & (16 \leq t \leq 31) \\ x \oplus y \oplus z & (32 \leq t \leq 47) \end{cases}$$

$0 \leq t \leq 15$ の時、f_t は条件分岐関数 (もし x が真なら y、そうでなければ z)、$16 \leq t \leq 31$ の時、f_t は多数決関数、$32 \leq t \leq 47$ の時、f_t はパリティ関数となっており、全て x, y, z の 3 変数に関する対称関数となっている。

MD4 の前処理

1. 入力メッセージ M について、メッセージ長が 512 ビットの倍数になるように初期パディングされたメッセージ

$$M||1||0^k||l$$

[1] 当時、big-endian マシンの主流であった SUN Sparcstation は相対的に高速であったため、endian 変換のペナルティに耐えうるであろうという判断もあった [R90]。

を計算する。ただし、l は M のメッセージ長のバイナリ表現 (64 ビット[2])、k は $l+1+k \equiv 448 \pmod{512}$ を満たす正の最小値である。

2. 初期パディングされたメッセージは N 個の 512 ビット単位のブロック (block) $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 32 ビット長のワード

$$M^{(i)} = M_0^{(i)} || M_1^{(i)} || \cdots || M_{15}^{(i)}$$

からなる。

3. 初期値として

$$\begin{aligned} H_0^{(0)} &= \texttt{01234567} \\ H_1^{(0)} &= \texttt{89abcdef} \\ H_2^{(0)} &= \texttt{fedcba98} \\ H_3^{(0)} &= \texttt{76543210} \end{aligned}$$

を設定する。

MD4 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \le i \le N$ の順に以下の手続き 1.〜6. (圧縮関数) (図 3.1 参照) を実行する。

1. メッセージブロックを X_j にコピーする。

$$X_j = M_j^{(i)} \quad (0 \le j \le 15)$$

2. 4 つの 32 ビット長のワード (A, B, C, D) を $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$\begin{aligned} A_0 &= H_0^{(i-1)} \\ B_0 &= H_1^{(i-1)} \\ C_0 &= H_2^{(i-1)} \\ D_0 &= H_3^{(i-1)} \end{aligned}$$

3. ラウンド 1 $(0 \le t \le 15)$

$$\begin{cases} A_{t+1} &= D_t \\ B_{t+1} &= \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_t + C_1) \\ C_{t+1} &= B_t \\ D_{t+1} &= C_t \end{cases}$$

[2]メッセージ長が 2^{64} より大きい場合、下位 64 ビットのみが使われ、上位ビットは無視される。

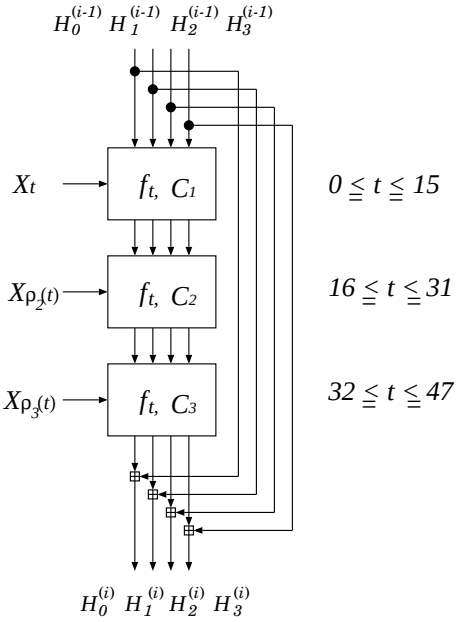


図 3.1: MD4 圧縮関数

4. ラウンド 2 $(16 \leq t \leq 31)$

$$\begin{cases} A_{t+1} & = D_t \\ B_{t+1} & = \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_{\rho_2(t)} + C_2) \\ C_{t+1} & = B_t \\ D_{t+1} & = C_t \end{cases}$$

5. ラウンド 3 $(32 \leq t \leq 47)$

$$\begin{cases} A_{t+1} & = D_t \\ B_{t+1} & = \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_{\rho_3(t)} + C_3) \\ C_{t+1} & = B_t \\ D_{t+1} & = C_t \end{cases}$$

各ラウンドにおけるローテーションビット数 $(s[t])$ は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ラウンド 1	3	7	11	19	3	7	11	19	3	7	11	19	3	7	11	19
ラウンド 2	3	5	9	13	3	5	9	13	3	5	9	13	3	5	9	13
ラウンド 3	3	9	11	15	3	9	11	15	3	9	11	15	3	9	11	15

ラウンド 2、ラウンド 3 においてメッセージ適用順序を示す関数 $\rho_2(t)$, $\rho_3(t)$ は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_2(t)$	0	4	8	12	1	5	9	13	2	6	10	14	3	7	11	15
$\rho_3(t)$	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15

C_1, C_2, C_3 はラウンド定数と呼ばれる 32 ビットワードの定数で、下記の値の整数部分である。

	C_1	C_2	C_3
	0	$2^{30} \cdot \sqrt{2}$	$2^{30} \cdot \sqrt{3}$
16 進表現	`00000000`	`5A827999`	`6ED9EBA1`

6. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_0^{(i-1)} + A_{48} \\ H_1^{(i)} &= H_1^{(i-1)} + B_{48} \\ H_2^{(i)} &= H_2^{(i-1)} + C_{48} \\ H_3^{(i)} &= H_3^{(i-1)} + D_{48} \end{aligned}$$

で計算する。

上記手続き 1.〜 6.(圧縮関数) を N 回繰り返した最終的な 128 ビットの値

$$H^{(N)} = H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)}$$

がメッセージ M のハッシュ値である。

3.2 MD5

3.2.1 概要

MD5 は MD4 の強度を高めた拡張アルゴリズムとして 1991 年に提案されたハッシュ関数である [R92b]。MD5 は、MD4 と同様に、ビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として 128 ビットのハッシュ値を出力する。演算は 32 ビット単位で行われ、全体は 4 ラウンド ×16 ステップで構成されている。MD4 と異なる点は、ラウンド数を 3 ラウンドから 4 ラウンドに増やした点、第 3 ラウンドのブール関数、メッセージワードの手順の変更、ラウンド定数の代わりに各演算に加算定数を追加した点である。

MD5 のアルゴリズムを記述した RFC1321 が 1992 年にリリースされ [R92b]、インターネット標準として広く利用されているハッシュ関数の一つであるが、2004 年に Wang ら [WFLY04] により衝突が発見されており、今後の利用は控えるべきと思われる。

3.2.2 技術仕様

MD5 で使用される関数

MD5 では、32 ビットワードを入力変数および出力変数とする、以下の論理関数 $f_0, f_1, \ldots, f_{63}$ が用いられる。

$$f_t(x,y,z) = \begin{cases} (x \wedge y) \vee (\neg x \wedge z) & (0 \leq t \leq 15) \\ (x \wedge z) \vee (y \wedge \neg z) & (16 \leq t \leq 31) \\ x \oplus y \oplus z & (32 \leq t \leq 47) \\ y \oplus (x \vee \neg z) & (48 \leq t \leq 63) \end{cases}$$

MD5 の前処理

1. 入力メッセージ M について、メッセージ長が 512 ビットの倍数になるように初期パディングされたメッセージ

   $$M||1||0^k||l$$

   を計算する。ただし、l は M のメッセージ長のバイナリ表現 (64 ビット[3])、k は $l + 1 + k \equiv 448 \pmod{512}$ を満たす正の最小値である。

[3] メッセージ長が 2^{64} より大きい場合、下位 64 ビットのみが使われ、上位ビットは無視される。

2. 初期パディングされたメッセージは N 個の 512 ビット単位のブロック (block) $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 32 ビット長のワード

$$M^{(i)} = M_0^{(i)}||M_1^{(i)}||\cdots||M_{15}^{(i)}$$

からなる。

3. 初期値として

$$\begin{aligned} H_0^{(0)} &= \texttt{01234567} \\ H_1^{(0)} &= \texttt{89abcdef} \\ H_2^{(0)} &= \texttt{fedcba98} \\ H_3^{(0)} &= \texttt{76543210} \end{aligned}$$

を設定する。

MD5 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \leq i \leq N$ の順に以下の手続き 1.〜7. (圧縮関数) (図 3.2 参照) を実行する。

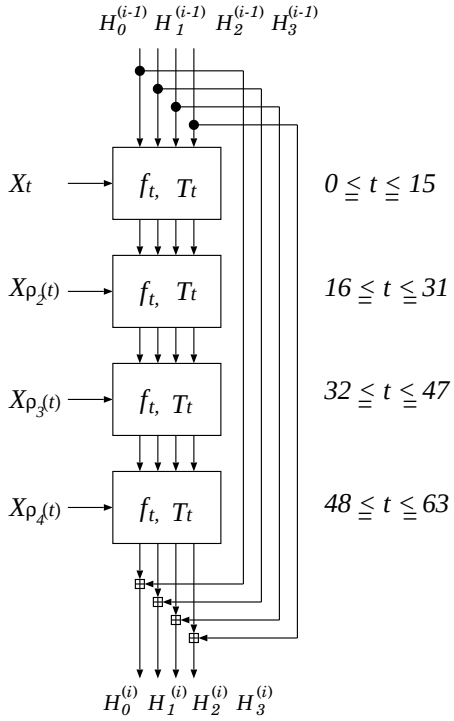


図 3.2: MD5 圧縮関数

1. メッセージブロックを X_j にコピーする。

$$X_j = M_j^{(i)} \quad (0 \leq j \leq 15)$$

2. 4 つの 32 ビット長のワード (A, B, C, D) を $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$\begin{aligned} A_0 &= H_0^{(i-1)} \\ B_0 &= H_1^{(i-1)} \\ C_0 &= H_2^{(i-1)} \\ D_0 &= H_3^{(i-1)} \end{aligned}$$

3. ラウンド 1 $(0 \leq t \leq 15)$

$$\begin{cases} A_{t+1} &= D_t \\ B_{t+1} &= B_t + \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_t + T_t) \\ C_{t+1} &= B_t \\ D_{t+1} &= C_t \end{cases}$$

4. ラウンド 2 $(16 \leq t \leq 31)$

$$\begin{cases} A_{t+1} &= D_t \\ B_{t+1} &= B_t + \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_{\rho_2(t)} + T_t) \\ C_{t+1} &= B_t \\ D_{t+1} &= C_t \end{cases}$$

5. ラウンド 3 $(32 \leq t \leq 47)$

$$\begin{cases} A_{t+1} &= D_t \\ B_{t+1} &= B_t + \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_{\rho_3(t)} + T_t) \\ C_{t+1} &= B_t \\ D_{t+1} &= C_t \end{cases}$$

6. ラウンド 4 $(48 \leq t \leq 63)$

$$\begin{cases} A_{t+1} &= D_t \\ B_{t+1} &= B_t + \mathrm{ROTL}^{s[t]}(A_t + f_t(B_t, C_t, D_t) + X_{\rho_4(t)} + T_t) \\ C_{t+1} &= B_t \\ D_{t+1} &= C_t \end{cases}$$

各ラウンドにおけるローテーションビット数 ($s[t]$) は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ラウンド 1	7	12	17	22	7	12	17	22	7	12	17	22	7	12	17	22
ラウンド 2	5	9	14	20	5	9	14	20	5	9	14	20	5	9	14	20
ラウンド 3	4	11	16	23	4	11	16	23	4	11	16	23	4	11	16	23
ラウンド 4	6	10	15	21	6	10	15	21	6	10	15	21	6	10	15	21

ラウンド 2、ラウンド 3、ラウンド 4 においてメッセージ適用順序を示す関数 $\rho_2(t)$, $\rho_3(t)$, $\rho_4(t)$ は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_2(t)$	1	6	11	0	5	10	15	4	9	14	3	8	13	2	7	12
$\rho_3(t)$	5	8	11	14	1	4	7	10	13	0	3	6	9	12	15	2
$\rho_4(t)$	0	7	14	5	12	3	10	1	8	15	6	13	4	11	2	9

ラウンド定数 T_t $(0 \leq t \leq 63)$ は正弦関数から導かれた定数である。

$$T_t = (4294967296 \times |\sin(t)|) \text{ の整数部分}$$

7. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_0^{(i-1)} + A_{64} \\ H_1^{(i)} &= H_1^{(i-1)} + B_{64} \\ H_2^{(i)} &= H_2^{(i-1)} + C_{64} \\ H_3^{(i)} &= H_3^{(i-1)} + D_{64} \end{aligned}$$

で計算する。

上記手続き 1.〜 7.(圧縮関数) を N 回繰り返した最終的な 128 ビットの値

$$H^{(N)} = H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)}$$

がメッセージ M のハッシュ値である。

3.3 RIPEMD

3.3.1 概要

RIPEMD は 1992 年にヨーロッパの RIPE (Race Integrity Primitive Evaluation) プロジェクトの成果の一つとして提案されたハッシュ関数である [RIPE92]。RIPEMD は、MD4、MD5 と同じく、ビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として、128 ビットのハッシュ値を出力する。

RIPEMD は den Boer と Bosselaer による MD4 の圧縮関数の最終 2 ラウンドに対する攻撃 [DB92] を考慮して改良されたアルゴリズムで、MD4 圧縮関数に変更を加えた関数を 2 並列に実行する構成となっている。演算は 32 ビット単位で行われ、全体は 3 ラウンド ×16 ステップを 2 並列に実行する構成となっている。

MD4 と異なる点は、以下の通りである。

- 各ステップでのローテーションビット数及びメッセージワードの適用順序が MD4 と異なる。
- 左ライン、右ラインの 2 並列処理は、ラウンド定数のみが異なる。
- 圧縮関数の最後で左右のラインの結果が加算される。

RIPEMD は、1995 年に Dobbertin により、3 ラウンド中の最初の 2 ラウンド及び最終 2 ラウンドに対する衝突攻撃が発表され、RIPEMD アルゴリズム自体が標準化されることはなかったが、設計指針は RIPEMD-160, RIPEMD-128 に引き継がれている。

3.3.2 技術仕様

RIPEMD で使用される関数

RIPEMD では、MD4 と同じ以下の論理関数 $f_0, f_1, \ldots, f_{47}$ が用いられる。

$$f_t(x,y,z) = \begin{cases} (x \wedge y) \vee (\neg x \wedge z) & (0 \leq t \leq 15) \\ (x \wedge y) \vee (x \wedge z) \vee (y \wedge z) & (16 \leq t \leq 31) \\ x \oplus y \oplus z & (32 \leq t \leq 47) \end{cases}$$

RIPEMD の前処理

1. 入力メッセージ M について、メッセージ長が 512 ビットの倍数になるように初期パディングされたメッセージ

$$M||1||0^k||l$$

を計算する。ただし、l は M のメッセージ長のバイナリ表現 (64 ビット[4])、k は $l + 1 + k \equiv 448 \pmod{512}$ を満たす正の最小値である。

2. 初期パディングされたメッセージは N 個の 512 ビット単位のブロック (block) $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 32 ビット長のワード

$$M^{(i)} = M_0^{(i)} || M_1^{(i)} || \cdots || M_{15}^{(i)}$$

からなる。

3. 初期値として

$$\begin{aligned} H_0^{(0)} &= \texttt{67452301} \\ H_1^{(0)} &= \texttt{efcdab89} \\ H_2^{(0)} &= \texttt{98badcfe} \\ H_3^{(0)} &= \texttt{10325476} \end{aligned}$$

を設定する。

RIPEMD のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \le i \le N$ の順に以下の手続き 1.〜6. (圧縮関数) (図 3.3 参照) を実行する。

1. メッセージブロックを X_j にコピーする。

$$X_j = M_j^{(i)} \quad (0 \le j \le 15)$$

2. 左右のラインそれぞれについて、4 つの 32 ビット長のワード $(A_{Lt}, B_{Lt}, C_{Lt}, D_{Lt})$, $(A_{Rt}, B_{Rt}, C_{Rt}, D_{Rt})$, $(0 \le t \le 48)$ がハッシュ値を計算するためのバッファとして用いられる。これらを $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$\begin{aligned} A_{L0} &= A_{R0} = H_0^{(i-1)} \\ B_{L0} &= B_{R0} = H_1^{(i-1)} \\ C_{L0} &= C_{R0} = H_2^{(i-1)} \\ D_{L0} &= D_{R0} = H_3^{(i-1)} \end{aligned}$$

[4] メッセージ長が 2^{64} より大きい場合、下位 64 ビットのみが使われ、上位ビットは無視される。

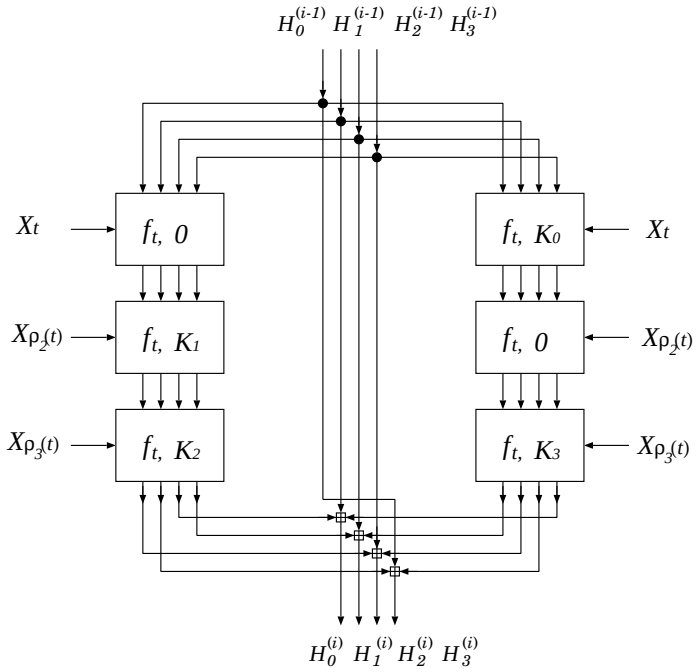


図 3.3: RIPEMD 圧縮関数

3. ラウンド 1 $(0 \leq t \leq 15)$

$$\text{左ライン} \begin{cases} A_{Lt+1} &= D_{Lt} \\ B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_t(B_{Lt}, C_{Lt}, D_{Lt}) + X_t) \\ C_{Lt+1} &= B_{Lt} \\ D_{Lt+1} &= C_{Lt} \end{cases}$$

$$\text{右ライン} \begin{cases} A_{Rt+1} &= D_{Rt} \\ B_{Rt+1} &= \mathrm{ROTL}^{s[t]}(A_{Rt} + f_t(B_{Rt}, C_{Rt}, D_{Rt}) + X_t + K_0) \\ C_{Rt+1} &= B_{Rt} \\ D_{Rt+1} &= C_{Rt} \end{cases}$$

4. ラウンド 2 $(16 \leq t \leq 31)$

$$\text{左ライン} \begin{cases} A_{Lt+1} &= D_{Lt} \\ B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_t(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho_2(t)} + K_1) \\ C_{Lt+1} &= B_{Lt} \\ D_{Lt+1} &= C_{Lt} \end{cases}$$

$$\text{右ライン} \begin{cases} A_{Rt+1} &= D_{Rt} \\ B_{Rt+1} &= \mathrm{ROTL}^{s[t]}(A_{Rt} + f_t(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho_2(t)}) \\ C_{Rt+1} &= B_{Rt} \\ D_{Rt+1} &= C_{Rt} \end{cases}$$

5. ラウンド 3 $(32 \leq t \leq 47)$

$$\text{左ライン} \begin{cases} A_{Lt+1} &= D_{Lt} \\ B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_t(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho_3(t)} + K_2) \\ C_{Lt+1} &= B_{Lt} \\ D_{Lt+1} &= C_{Lt} \end{cases}$$

$$\text{右ライン} \begin{cases} A_{Rt+1} &= D_{Rt} \\ B_{Rt+1} &= \mathrm{ROTL}^{s[t]}(A_{Rt} + f_t(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho_3(t)} + K_3) \\ C_{Rt+1} &= B_{Rt} \\ D_{Rt+1} &= C_{Rt} \end{cases}$$

各ラウンドにおけるローテーションビット数 $(s[t])$ は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ラウンド 1	11	14	15	12	5	8	7	9	11	13	14	15	6	7	9	8
ラウンド 2	7	6	8	13	11	9	7	15	7	12	15	9	7	11	13	12
ラウンド 3	11	13	14	7	14	9	13	15	6	8	13	6	12	5	7	5

ラウンド 2、ラウンド 3 においてメッセージ適用順序を示す関数 $\rho_2(t)$, $\rho_3(t)$ は以下で定義される。

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_2(t)$	7	4	13	1	10	6	15	3	12	0	9	5	14	2	11	8
$\rho_3(t)$	3	10	2	4	9	15	8	1	14	7	0	6	11	13	5	12

K_0, K_1, K_2, K_3 はラウンド定数と呼ばれる 32 ビットワードの定数で、下記の値の整数部分である。

	K_0	K_1	K_2	K_3
	$2^{30} \cdot \sqrt[3]{2}$	$2^{30} \cdot \sqrt{2}$	$2^{30} \cdot \sqrt{3}$	$2^{30} \cdot \sqrt[3]{3}$
16 進表現	`50a28be6`	`5A827999`	`6ED9EBA1`	`5C4DD124`

6. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_1^{(i-1)} + C_{L48} + D_{R48} \\ H_1^{(i)} &= H_2^{(i-1)} + D_{L48} + A_{R48} \\ H_2^{(i)} &= H_3^{(i-1)} + A_{L48} + B_{R48} \\ H_3^{(i)} &= H_0^{(i-1)} + B_{L48} + C_{R48} \end{aligned}$$

で計算する。

上記手続き 1.〜 6.(圧縮関数) を N 回繰り返した最終的な 128 ビットの値

$$H^{(N)} = H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)}$$

がメッセージ M のハッシュ値である。

3.4 RIPEMD-160

3.4.1 概要

RIPEMD-160 は、RIPEMD の強度を高めたアルゴリズムとして、RIPEMD-128 とともに 1996 年に Dobbertin, Bosselaers, Preneel により提案されたハッシュ関数である [DBP96]。

RIPEMD-160 は、ビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として 160 ビットのハッシュ値を出力する。

RIPEMD-160 は 2 つのほぼ同じ形をした関数を 2 並列に実行する。2 つの関数は右ラインおよび左ラインと呼ばれ、各々5 ラウンド 80 ステップで構成される。

RIPEMD-160 は国際規格 ISO/IEC 10118-3 [ISO/IEC10118-3] 及び電子政府推奨暗号リスト [CRYPTREC03] に採用されている。

3.4.2 技術仕様

RIPEMD-160 で使用される関数

$$
\begin{aligned}
f_1(x,y,z) &= x \oplus y \oplus z \\
f_2(x,y,z) &= (x \wedge y) \vee (\neg x \wedge z) \\
f_3(x,y,z) &= (x \wedge \neg y) \oplus z \\
f_4(x,y,z) &= (x \wedge y) \vee (y \wedge \neg z) \\
f_5(x,y,z) &= x \oplus (y \vee \neg z)
\end{aligned}
$$

これらのブール関数の適用順序は次の通りである。

ライン	ラウンド 1	ラウンド 2	ラウンド 3	ラウンド 4	ラウンド 5
左	f_1	f_2	f_3	f_4	f_5
右	f_5	f_4	f_3	f_2	f_1

RIPEMD の前処理

1. 入力メッセージ M について、メッセージ長が 512 ビットの倍数になるように初期パディングされたメッセージ

   $$M||1||0^k||l$$

   を計算する。ただし、l は M のメッセージ長のバイナリ表現 (64 ビット[5])、k は $l + 1 + k \equiv 448 \pmod{512}$ を満たす正の最小値である。

[5] メッセージ長を表現する場合、下位 32 ビット表現のあとに上位 32 ビット表現をする。なお、メッセージ長が 2^{64} より大きい場合、下位 64 ビットのみが使われ、上位ビットは無視される。

2. 初期パディングされたメッセージは N 個の 512 ビット単位のブロック (block) $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 32 ビット長のワード

$$M^{(i)} = M_0^{(i)}||M_1^{(i)}||\cdots||M_{15}^{(i)}$$

からなる。

3. 初期値として

$$\begin{aligned} H_0^{(0)} &= \texttt{67452301} \\ H_1^{(0)} &= \texttt{efcdab89} \\ H_2^{(0)} &= \texttt{98badcfe} \\ H_3^{(0)} &= \texttt{10325476} \\ H_4^{(0)} &= \texttt{c3d2e1f0} \end{aligned}$$

を設定する。

RIPEMD-160 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \le i \le N$ の順に以下の手続き 1.〜8. (圧縮関数) (図 3.4 参照) を実行する。

1. メッセージブロックを X_j にコピーする。

$$X_j = M_j^{(i)} \quad (0 \le j \le 15)$$

2. 左右のラインそれぞれについて、4 つの 32 ビット長のワード $(A_{Lt}, B_{Lt}, C_{Lt}, D_{Lt})$, $(A_{Rt}, B_{Rt}, C_{Rt}, D_{Rt})$, $(0 \le t \le 48)$ がハッシュ値を計算するためのバッファとして用いられる。これらを $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$\begin{aligned} A_{L0} &= A_{R0} = H_0^{(i-1)} \\ B_{L0} &= B_{R0} = H_1^{(i-1)} \\ C_{L0} &= C_{R0} = H_2^{(i-1)} \\ D_{L0} &= D_{R0} = H_3^{(i-1)} \\ E_{L0} &= E_{R0} = H_4^{(i-1)} \end{aligned}$$

3. ラウンド 1 $(1 \le t \le 16)$

$$\text{左ライン} \begin{cases} A_{Lt+1} &= E_{Lt} \\ B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_1(B_{Lt}, C_{Lt}, D_{Lt}) + X_t + K_1) + E_{Lt} \\ C_{Lt+1} &= B_{Lt} \\ D_{Lt+1} &= \mathrm{ROTL}^{10}(C_{Lt}) \\ E_{Lt+1} &= D_{Lt} \end{cases}$$

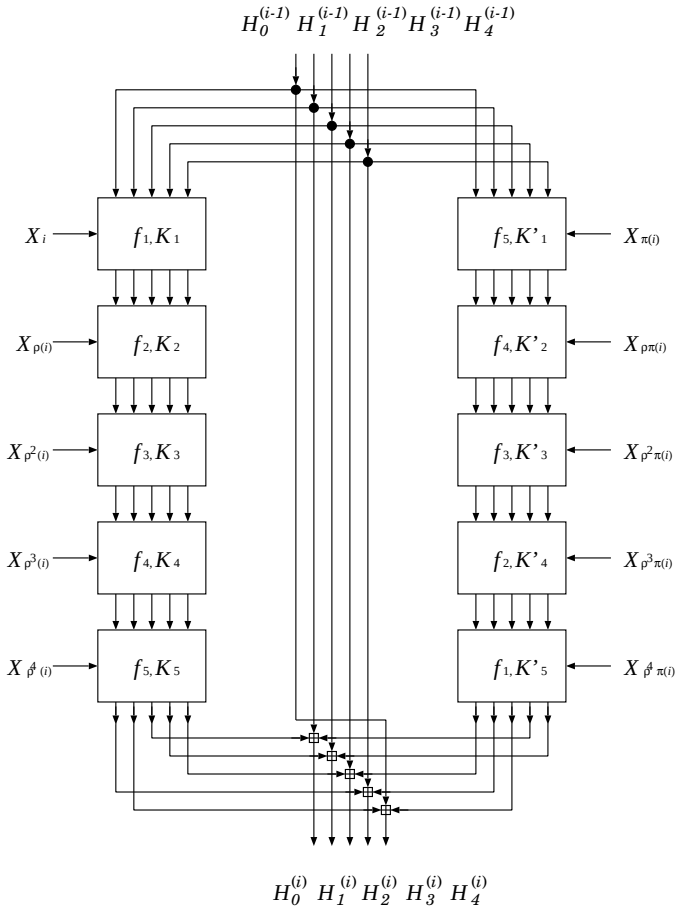


図 3.4: RIPEMD-160 圧縮関数

$$
\text{右ライン}\begin{cases}
A_{Rt+1} &= E_{Rt} \\
B_{Rt+1} &= \mathrm{ROTL}^{s'[t]}(A_{Rt} + f_5(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\pi(t)} + K'_1) + E_{Rt} \\
C_{Rt+1} &= B_{Rt} \\
D_{Rt+1} &= \mathrm{ROTL}^{10}(C_{Rt}) \\
E_{Rt+1} &= D_{Rt}
\end{cases}
$$

4. ラウンド 2 ($17 \leq t \leq 32$)

$$
\text{左ライン}\begin{cases}
A_{Lt+1} &= E_{Lt} \\
B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_2(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho(t)} + K_2) + E_{Lt} \\
C_{Lt+1} &= B_{Lt} \\
D_{Lt+1} &= \mathrm{ROTL}^{10}(C_{Lt}) \\
E_{Lt+1} &= D_{Lt}
\end{cases}
$$

$$
\text{右ライン}\begin{cases}
A_{Rt+1} &= E_{Rt} \\
B_{Rt+1} &= \mathrm{ROTL}^{s'[t]}(A_{Rt} + f_4(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho\pi(t)} + K'_2) + E_{Rt} \\
C_{Rt+1} &= B_{Rt} \\
D_{Rt+1} &= \mathrm{ROTL}^{10}(C_{Rt}) \\
E_{Rt+1} &= D_{Rt}
\end{cases}
$$

5. ラウンド 3 ($33 \leq t \leq 48$)

$$
\text{左ライン}\begin{cases}
A_{Lt+1} &= E_{Lt} \\
B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_3(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho^2(t)} + K_3) + E_{Lt} \\
C_{Lt+1} &= B_{Lt} \\
D_{Lt+1} &= \mathrm{ROTL}^{10}(C_{Lt}) \\
E_{Lt+1} &= D_{Lt}
\end{cases}
$$

$$
\text{右ライン}\begin{cases}
A_{Rt+1} &= E_{Rt} \\
B_{Rt+1} &= \mathrm{ROTL}^{s'[t]}(A_{Rt} + f_3(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho^2\pi(t)} + K'_3) + E_{Rt} \\
C_{Rt+1} &= B_{Rt} \\
D_{Rt+1} &= \mathrm{ROTL}^{10}(C_{Rt}) \\
E_{Rt+1} &= D_{Rt}
\end{cases}
$$

6. ラウンド 4 ($49 \leq t \leq 64$)

$$
\text{左ライン}\begin{cases}
A_{Lt+1} &= E_{Lt} \\
B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_4(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho^3(t)} + K_4) + E_{Lt} \\
C_{Lt+1} &= B_{Lt} \\
D_{Lt+1} &= \mathrm{ROTL}^{10}(C_{Lt}) \\
E_{Lt+1} &= D_{Lt}
\end{cases}
$$

$$
\text{右ライン}\begin{cases}
A_{Rt+1} &= E_{Rt} \\
B_{Rt+1} &= \mathrm{ROTL}^{s'[t]}(A_{Rt} + f_2(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho^3\pi(t)} + K'_4) + E_{Rt} \\
C_{Rt+1} &= B_{Rt} \\
D_{Rt+1} &= \mathrm{ROTL}^{10}(C_{Rt}) \\
E_{Rt+1} &= D_{Rt}
\end{cases}
$$

7. ラウンド 5 ($65 \leq t \leq 80$)

$$
\text{左ライン}\begin{cases}
A_{Lt+1} &= E_{Lt} \\
B_{Lt+1} &= \mathrm{ROTL}^{s[t]}(A_{Lt} + f_5(B_{Lt}, C_{Lt}, D_{Lt}) + X_{\rho^4(t)} + K_5) + E_{Lt} \\
C_{Lt+1} &= B_{Lt} \\
D_{Lt+1} &= \mathrm{ROTL}^{10}(C_{Lt}) \\
E_{Lt+1} &= D_{Lt}
\end{cases}
$$

$$
\text{右ライン}\begin{cases}
A_{Rt+1} &= E_{Rt} \\
B_{Rt+1} &= \mathrm{ROTL}^{s'[t]}(A_{Rt} + f_1(B_{Rt}, C_{Rt}, D_{Rt}) + X_{\rho^4\pi(t)} + K'_5) + E_{Rt} \\
C_{Rt+1} &= B_{Rt} \\
D_{Rt+1} &= \mathrm{ROTL}^{10}(C_{Rt}) \\
E_{Rt+1} &= D_{Rt}
\end{cases}
$$

但し、メッセージワードの適用順序に対し、以下の置換 ρ と π が定義されている。

置換 ρ の定義

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho(i)$	7	4	13	1	10	6	15	3	12	0	9	5	2	14	11	8

置換 π の定義

$$\pi(i) = 9i + 5 \pmod{16}$$

メッセージワードの適用順序は次のようになる。

ライン	ラウンド 1	ラウンド 2	ラウンド 3	ラウンド 4	ラウンド 5
左	恒等変換	ρ	ρ^2	ρ^3	ρ^4
右	π	$\rho\pi$	$\rho^2\pi$	$\rho^3\pi$	$\rho^4\pi$

ラウンド定数は下記の値の整数部分である。

左ライン	K_1	K_2	K_3	K_4	K_5
	0	$2^{30} \cdot \sqrt{2}$	$2^{30} \cdot \sqrt{3}$	$2^{30} \cdot \sqrt{5}$	$2^{30} \cdot \sqrt{7}$
16 進表現	00000000	5A827999	6ED9EBA1	8F1BBCDC	A953FD4E
右ライン	K'_1	K'_2	K'_3	K'_4	K'_5
	$2^{30} \cdot \sqrt[3]{2}$	$2^{30} \cdot \sqrt[3]{3}$	$2^{30} \cdot \sqrt[3]{5}$	$2^{30} \cdot \sqrt{7}$	0
16 進表現	50A28BE6	5C4DD124	6D703EF3	7A6D76E9	00000000

また、ステップ関数で用いられる左巡回シフト量 $s[t], s'[t]$ はあらかじめ定められている。

8. i 番目の中間ハッシュ値を

$$
\begin{aligned}
H_0^{(i)} &= H_1^{(i-1)} + C_{L80} + D_{R80} \\
H_1^{(i)} &= H_2^{(i-1)} + D_{L80} + E_{R80} \\
H_2^{(i)} &= H_3^{(i-1)} + E_{L80} + A_{R80} \\
H_3^{(i)} &= H_4^{(i-1)} + A_{L80} + B_{R80} \\
H_4^{(i)} &= H_0^{(i-1)} + B_{L80} + C_{R80}
\end{aligned}
$$

で計算する。

上記手続き 1.〜 8.(圧縮関数) を N 回繰り返した最終的な 160 ビットの値

$$H^{(N)} = H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}$$

がメッセージ M のハッシュ値である。

3.5 RIPEMD-128

3.5.1 概要

RIPEMD-128 は、RIPEMD の強度を高めたアルゴリズムとして 1996 年に Dobbertin, Bosselaers, Preneel により提案されたハッシュ関数である [DBP96]。

RIPEMD-128 は、ビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として 128 ビットのハッシュ値を出力する。

RIPEMD-128 は、RIPEMD-160 をベースに設計されており、(A, B, C, D) の 4 変数のみ使うこと、5 ラウンドのうち 4 ラウンドのみを使うこと、各ラウンドで用いるブール関数とラウンド定数のみが異なっている。

RIPEMD-128 は国際規格 ISO/IEC 10118-3 [ISO/IEC10118-3] に採用されている。

3.5.2 技術仕様

RIPEMD-128 は、RIPEMD-160 をベースに設計されており、(A, B, C, D) の 4 変数のみ使うこと、5 ラウンドのうち 4 ラウンドのみを使うこと、各ラウンドで用いるブール関数とラウンド定数のみが異なっている。

ブール関数の適用順序は次の通りである。

ライン	ラウンド 1	ラウンド 2	ラウンド 3	ラウンド 4
左	f_1	f_2	f_3	f_4
右	f_4	f_3	f_2	f_1

ラウンド定数は以下の値の整数部分である。

ライン	ラウンド 1	ラウンド 2	ラウンド 3	ラウンド 4
左	0	$2^{30} \cdot \sqrt[3]{2}$	$2^{30} \cdot \sqrt[3]{3}$	$2^{30} \cdot \sqrt[3]{5}$
右	$2^{30} \cdot \sqrt[3]{2}$	$2^{30} \cdot \sqrt[3]{3}$	$2^{30} \cdot \sqrt[3]{5}$	0

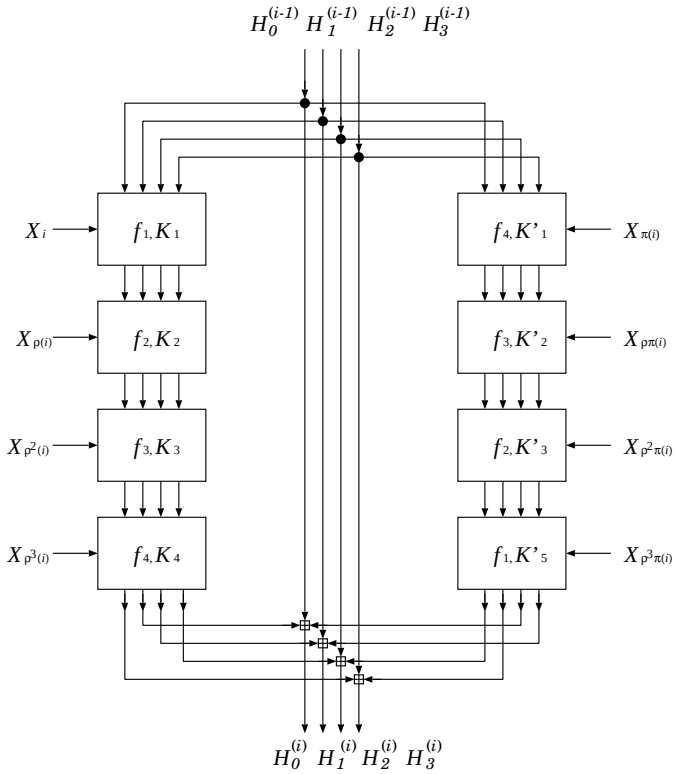


図 3.5: RIPEMD-128 圧縮関数

3.6 SHA-1

3.6.1 概要

SHS (Secure Hash Standard) は米国商務省 技術標準機関 NIST (National Institute of Standards and Technology) が制定したハッシュ関数の標準規格である。SHA (Secure Hash Algorithm) は MD4 及び MD5 をベースに米国国家安全局 NSA (National Security Agency) により設計され、1993 年に NIST により米国連邦政府情報処理規格 FIPS (Federal Information Processing Standard) 180 [FIPS180] として制定された。

SHA はビット長が 512 ビットの倍数になるようにパディングされた任意のメッセージを入力として、160 ビットのハッシュ値を出力する。

その後、SHA (SHA-0) のメッセージスケジュール関数のみに若干の仕様変更 (1 ビットローテーションの追加) がなされ、このアルゴリズム (SHA-1) が、SHA (SHA-0) を置き換える形で 1995 年に FIPS180-1 [FIPS180-1] として制定された。

3.6.2 技術仕様

SHA-1 で使用される関数

SHA-1 では、32 ビットワードを入力変数および出力変数とする、以下の論理関数 $f_0, f_1, \ldots, f_{79}$ が用いられる。

$$f_t(x,y,z) = \begin{cases} \mathrm{Ch}(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) & (0 \le t \le 19) \\ \mathrm{Parity}(x,y,z) = x \oplus y \oplus z & (20 \le t \le 39) \\ \mathrm{Maj}(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) & (40 \le t \le 59) \\ \mathrm{Parity}(x,y,z) = x \oplus y \oplus z & (60 \le t \le 79) \end{cases}$$

SHA-1 の前処理

1. 入力メッセージ M について、メッセージ長が 512 ビットの倍数になるように初期パディングされたメッセージ

   $$M||1||0^k||l$$

   を計算する。ただし、l は M のメッセージ長のバイナリ表現 (64 ビット)、k は $l + 1 + k \equiv 448 \pmod{512}$ を満たす正の最小値である。

2. 初期パディングされたメッセージは N 個の 512 ビット単位のブロック (block) $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 32 ビット長のワード

   $$M^{(i)} = M_0^{(i)}||M_1^{(i)}||\cdots||M_{15}^{(i)}$$

   からなる。

3. 初期値として

$$\begin{aligned} H_0^{(0)} &= \texttt{67452301} \\ H_1^{(0)} &= \texttt{efcdab89} \\ H_2^{(0)} &= \texttt{98badcfe} \\ H_3^{(0)} &= \texttt{10325476} \\ H_4^{(0)} &= \texttt{c3d2e1f0} \end{aligned}$$

を設定する。

SHA-1 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \leq i \leq N$ の順に以下の手続きを実行する。

1. 次式で定義する SHA-1 メッセージスケジュール関数を用いて拡張メッセージ W_t を計算する。

$$W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \mathrm{ROTL}^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) & 16 \leq t \leq 79 \end{cases}$$

2. 5 個のバッファ変数を $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$\begin{aligned} a_0 &= H_0^{(i-1)} \\ b_0 &= H_1^{(i-1)} \\ c_0 &= H_2^{(i-1)} \\ d_0 &= H_3^{(i-1)} \\ e_0 &= H_4^{(i-1)} \end{aligned}$$

3. $0 \leq t \leq 79$ に対して以下の計算を繰り返す。

$$\begin{cases} T &= \mathrm{ROTL}^5(a_t) + f_t(b_t, c_t, d_t) + e_t + K_t + W_t \\ e_{t+1} &= d_t \\ d_{t+1} &= c_t \\ c_{t+1} &= \mathrm{ROTL}^{30}(b_t) \\ b_{t+1} &= a_t \\ a_{t+1} &= T \end{cases}$$

ただし、$K_t (0 \leq t \leq 79)$ は 32 ビットワードの定数である。

4. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_0^{(i-1)} + a_{80} \\ H_1^{(i)} &= H_1^{(i-1)} + b_{80} \\ H_2^{(i)} &= H_2^{(i-1)} + c_{80} \\ H_3^{(i)} &= H_3^{(i-1)} + d_{80} \\ H_4^{(i)} &= H_4^{(i-1)} + e_{80} \end{aligned}$$

で計算する。

上記手続き 1.〜 4. を N 回繰り返した最終的な 160 ビットの値

$$H^{(N)} = H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}$$

がメッセージ M のハッシュ値である。

3.7 Whirlpool

3.7.1 概要

Whirlpool は Rijmen, Barreto により提案されたハッシュ長 512 ビットの暗号学的ハッシュ関数である。このハッシュ関数は NESSIE (New European Schemes for Signatures, Integrity, and Encryption) [NESSIE] で提案された [BR00]。ISO/IEC はこのハッシュ関数を標準暗号の一つとして標準化した [ISO/IEC10118-3]。

Whirlpool はその発表から標準化策定まで 2 度の改良を行なっている。これらについては別途詳細に触れるとして，本稿の安全性評価の対象は ISO/IEC で定義されている最終版 (2006.1.5 時点) とする。

3.7.2 技術仕様

内部ブロック暗号

それぞれ 512 ビットのパラメータ (FB) とメッセージ値 (M) を入力とし，512 ビットの出力 (O) を生成する。この処理の概要を図 3.6 に示す。

入力の 1024 ビットは左右それぞれ 512 ビットデータとして扱われ，各々8 バイト四方の行列構造として扱う。関数内部では，以下で説明する段関数対を 10 回繰り返す処理からなる。各々の段関数対では，同一の段関数 2 度処理することで，2 つの 512 ビット入力から 2 つの 512 ビット出力を生成する。左半分 (512 ビット) 入力は固定値 ($Const^i$) をパラメータとした段関数を処理するのに対して，右半分 (512 ビット) 入力では，左半分で生成した中間値をパラメータとして段関数を処理する。

各々の段関数は図 3.7 に示すように 4 種類の関数からなる; (1)S ボックス変換，(2) バイトシフト，(3) 線形変換，(4) 排他論理和。段関数ではこれら 4 つの関数をこの順で処理する。以下に具体的な処理を記載する。

S ボックス変換では，8 ビット入出力の単射な置換表を各バイトに適用 (合計 64 回) する。バイトシフト処理では，第 i 列 ($0 \leq i \leq 7$, 最左列が 0) を下方向に i 個ずらす。線形変換では，各行を線形な変換を行なう。排他論理和では決められた値をそれぞれのバイトに排他論理和する。決められた値とは段関数のパラメータに相当し，全体構成における左のパスでは仕様で定められる固有値が用いられ，右のパスでは左のパスの決められた中間値が用いられる。

各バージョンの違い

NESSIE へ提案された最初のバージョンと最終的に ISO/IEC で標準化されたものとの差は 2 つの要素関数の仕様である。これらの違いを示す (NESSIE の最終的な候補アルゴリズムには，前者の S ボックスの変更のみが反映されたものである)。

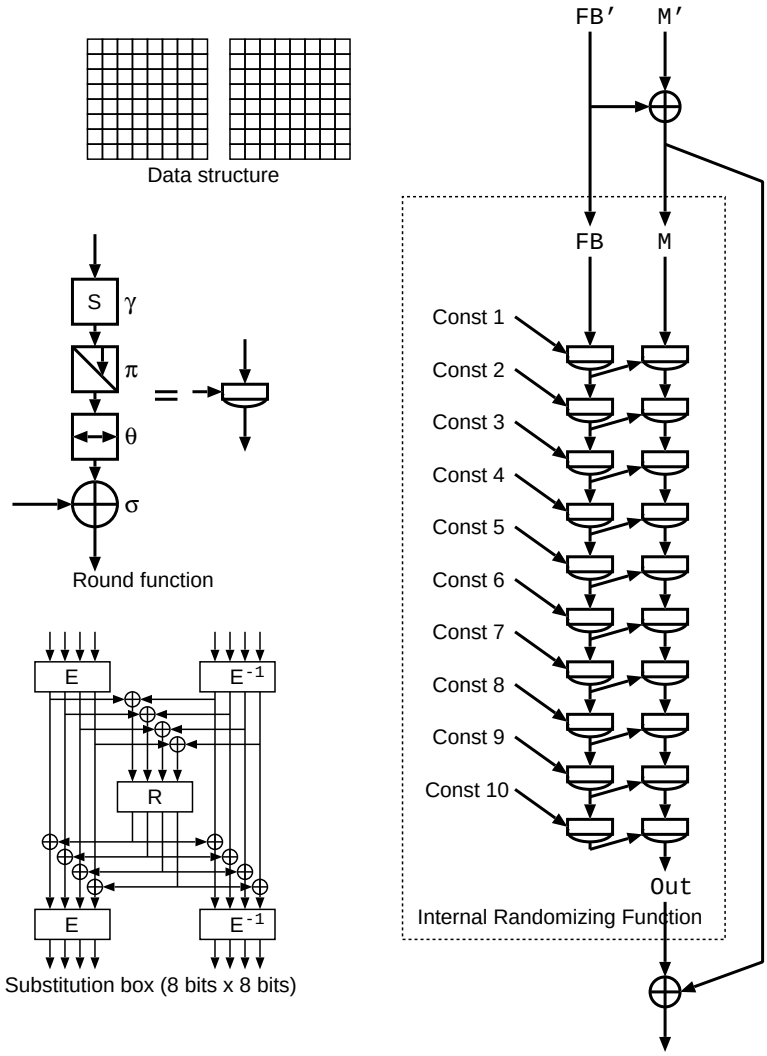


図 3.6: Whirlpool ハッシュ関数を構成する内部の処理

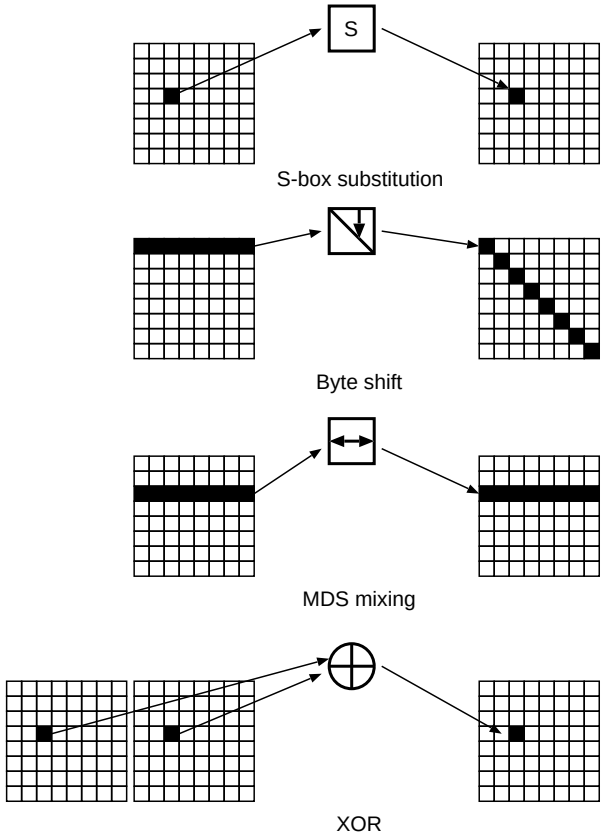


図 3.7: Whirlpool ハッシュ関数を構成する内部の処理

S-box　NESSIE に提案された最初のバージョンでは，SHA-1 による疑似乱数生成を用いた生成法による定義であった。しかし，この乱数表作成には安全性の欠陥があり線形確率が当初 $15 \cdot 2^{-6}$ として提案されていたところが，本当は $16 \cdot 2^{-6}$ であることが判明した。

```
old_sbox[256] = {
 0x68, 0xd0, 0xeb, 0x2b, 0x48, 0x9d, 0x6a, 0xe4,
 0xe3, 0xa3, 0x56, 0x81, 0x7d, 0xf1, 0x85, 0x9e,
 0x2c, 0x8e, 0x78, 0xca, 0x17, 0xa9, 0x61, 0xd5,
 0x5d, 0x0b, 0x8c, 0x3c, 0x77, 0x51, 0x22, 0x42,
 0x3f, 0x54, 0x41, 0x80, 0xcc, 0x86, 0xb3, 0x18,
 0x2e, 0x57, 0x06, 0x62, 0xf4, 0x36, 0xd1, 0x6b,
 0x1b, 0x65, 0x75, 0x10, 0xda, 0x49, 0x26, 0xf9,
 0xcb, 0x66, 0xe7, 0xba, 0xae, 0x50, 0x52, 0xab,
 0x05, 0xf0, 0x0d, 0x73, 0x3b, 0x04, 0x20, 0xfe,
 0xdd, 0xf5, 0xb4, 0x5f, 0x0a, 0xb5, 0xc0, 0xa0,
 0x71, 0xa5, 0x2d, 0x60, 0x72, 0x93, 0x39, 0x08,
 0x83, 0x21, 0x5c, 0x87, 0xb1, 0xe0, 0x00, 0xc3,
```

```
 0x12, 0x91, 0x8a, 0x02, 0x1c, 0xe6, 0x45, 0xc2,
 0xc4, 0xfd, 0xbf, 0x44, 0xa1, 0x4c, 0x33, 0xc5,
 0x84, 0x23, 0x7c, 0xb0, 0x25, 0x15, 0x35, 0x69,
 0xff, 0x94, 0x4d, 0x70, 0xa2, 0xaf, 0xcd, 0xd6,
 0x6c, 0xb7, 0xf8, 0x09, 0xf3, 0x67, 0xa4, 0xea,
 0xec, 0xb6, 0xd4, 0xd2, 0x14, 0x1e, 0xe1, 0x24,
 0x38, 0xc6, 0xdb, 0x4b, 0x7a, 0x3a, 0xde, 0x5e,
 0xdf, 0x95, 0xfc, 0xaa, 0xd7, 0xce, 0x07, 0x0f,
 0x3d, 0x58, 0x9a, 0x98, 0x9c, 0xf2, 0xa7, 0x11,
 0x7e, 0x8b, 0x43, 0x03, 0xe2, 0xdc, 0xe5, 0xb2,
 0x4e, 0xc7, 0x6d, 0xe9, 0x27, 0x40, 0xd8, 0x37,
 0x92, 0x8f, 0x01, 0x1d, 0x53, 0x3e, 0x59, 0xc1,
 0x4f, 0x32, 0x16, 0xfa, 0x74, 0xfb, 0x63, 0x9f,
 0x34, 0x1a, 0x2a, 0x5a, 0x8d, 0xc9, 0xcf, 0xf6,
 0x90, 0x28, 0x88, 0x9b, 0x31, 0x0e, 0xbd, 0x4a,
 0xe8, 0x96, 0xa6, 0x0c, 0xc8, 0x79, 0xbc, 0xbe,
 0xef, 0x6e, 0x46, 0x97, 0x5b, 0xed, 0x19, 0xd9,
 0xac, 0x99, 0xa8, 0x29, 0x64, 0x1f, 0xad, 0x55,
 0x13, 0xbb, 0xf7, 0x6f, 0xb9, 0x47, 0x2f, 0xee,
 0xb8, 0x7b, 0x89, 0x30, 0xd3, 0x7f, 0x76, 0x82};
```

線形確率に関する上記確率をよりよいものとするために，別の手法によるSボックスの構成が行なわれた。具体的には，2つの4ビット置換表 (E(とその逆変換の E^{-1}), R) を線形変換でつなぎ8ビットのSボックスとするものである。E は代数的な構成により生成した変換表であり，R は疑似ランダムに生成した変換表である。これらが構成する新しい変換表を以下に示す。

```
new_sbox[256] = {
unsigned char s[256] ={
 0x18, 0x23, 0xc6, 0xe8, 0x87, 0xb8, 0x01, 0x4f,
 0x36, 0xa6, 0xd2, 0xf5, 0x79, 0x6f, 0x91, 0x52,
 0x60, 0xbc, 0x9b, 0x8e, 0xa3, 0x0c, 0x7b, 0x35,
 0x1d, 0xe0, 0xd7, 0xc2, 0x2e, 0x4b, 0xfe, 0x57,
 0x15, 0x77, 0x37, 0xe5, 0x9f, 0xf0, 0x4a, 0xda,
 0x58, 0xc9, 0x29, 0x0a, 0xb1, 0xa0, 0x6b, 0x85,
 0xbd, 0x5d, 0x10, 0xf4, 0xcb, 0x3e, 0x05, 0x67,
 0xe4, 0x27, 0x41, 0x8b, 0xa7, 0x7d, 0x95, 0xd8,
 0xfb, 0xee, 0x7c, 0x66, 0xdd, 0x17, 0x47, 0x9e,
 0xca, 0x2d, 0xbf, 0x07, 0xad, 0x5a, 0x83, 0x33,
 0x63, 0x02, 0xaa, 0x71, 0xc8, 0x19, 0x49, 0xd9,
 0xf2, 0xe3, 0x5b, 0x88, 0x9a, 0x26, 0x32, 0xb0,
 0xe9, 0x0f, 0xd5, 0x80, 0xbe, 0xcd, 0x34, 0x48,
```

```
0xff, 0x7a, 0x90, 0x5f, 0x20, 0x68, 0x1a, 0xae,
0xb4, 0x54, 0x93, 0x22, 0x64, 0xf1, 0x73, 0x12,
0x40, 0x08, 0xc3, 0xec, 0xdb, 0xa1, 0x8d, 0x3d,
0x97, 0x00, 0xcf, 0x2b, 0x76, 0x82, 0xd6, 0x1b,
0xb5, 0xaf, 0x6a, 0x50, 0x45, 0xf3, 0x30, 0xef,
0x3f, 0x55, 0xa2, 0xea, 0x65, 0xba, 0x2f, 0xc0,
0xde, 0x1c, 0xfd, 0x4d, 0x92, 0x75, 0x06, 0x8a,
0xb2, 0xe6, 0x0e, 0x1f, 0x62, 0xd4, 0xa8, 0x96,
0xf9, 0xc5, 0x25, 0x59, 0x84, 0x72, 0x39, 0x4c,
0x5e, 0x78, 0x38, 0x8c, 0xd1, 0xa5, 0xe2, 0x61,
0xb3, 0x21, 0x9c, 0x1e, 0x43, 0xc7, 0xfc, 0x04,
0x51, 0x99, 0x6d, 0x0d, 0xfa, 0xdf, 0x7e, 0x24,
0x3b, 0xab, 0xce, 0x11, 0x8f, 0x4e, 0xb7, 0xeb,
0x3c, 0x81, 0x94, 0xf7, 0xb9, 0x13, 0x2c, 0xd3,
0xe7, 0x6e, 0xc4, 0x03, 0x56, 0x44, 0x7f, 0xa9,
0x2a, 0xbb, 0xc1, 0x53, 0xdc, 0x0b, 0x9d, 0x6c,
0x31, 0x74, 0xf6, 0x46, 0xac, 0x89, 0x14, 0xe1,
0x16, 0x3a, 0x69, 0x09, 0x70, 0xb6, 0xd0, 0xed,
0xcc, 0x42, 0x98, 0xa4, 0x28, 0x5c, 0xf8, 0x86};
```

線形変換　一つは線形変換を定義する行列 C である。オリジナルでは以下のように定義されていた。下線で示す部分行列に逆行列が存在せず，この行列は安全性評価上期待される条件を満たしていないことが示された [SS03]。

$$C_{\mathrm{old}} = \begin{pmatrix} 0x01 & 0x01 & 0x03 & 0x01 & 0x05 & 0x08 & 0x09 & 0x05 \\ 0x05 & 0x01 & 0x01 & 0x03 & 0x01 & 0x05 & 0x08 & 0x09 \\ 0x09 & \underline{0x05} & 0x01 & 0x01 & 0x03 & \underline{0x01} & 0x05 & 0x08 \\ 0x08 & 0x09 & 0x05 & 0x01 & 0x01 & 0x03 & 0x01 & 0x05 \\ 0x05 & 0x08 & 0x09 & 0x05 & 0x01 & 0x01 & 0x03 & 0x01 \\ 0x01 & \underline{0x05} & 0x08 & 0x09 & 0x05 & \underline{0x01} & 0x01 & 0x03 \\ 0x03 & 0x01 & 0x05 & 0x08 & 0x09 & 0x05 & 0x01 & 0x01 \\ 0x01 & 0x03 & 0x01 & 0x05 & 0x08 & 0x09 & 0x05 & 0x01 \end{pmatrix}.$$

そこで，改良により以下の行列に変更された。

$$C_{\mathrm{new}} = \begin{pmatrix} \texttt{0x01} & \texttt{0x01} & \texttt{0x04} & \texttt{0x01} & \texttt{0x08} & \texttt{0x05} & \texttt{0x02} & \texttt{0x09} \\ \texttt{0x09} & \texttt{0x01} & \texttt{0x01} & \texttt{0x04} & \texttt{0x01} & \texttt{0x08} & \texttt{0x05} & \texttt{0x02} \\ \texttt{0x02} & \texttt{0x09} & \texttt{0x01} & \texttt{0x01} & \texttt{0x04} & \texttt{0x01} & \texttt{0x08} & \texttt{0x05} \\ \texttt{0x05} & \texttt{0x02} & \texttt{0x09} & \texttt{0x01} & \texttt{0x01} & \texttt{0x04} & \texttt{0x01} & \texttt{0x08} \\ \texttt{0x08} & \texttt{0x05} & \texttt{0x02} & \texttt{0x09} & \texttt{0x01} & \texttt{0x01} & \texttt{0x04} & \texttt{0x01} \\ \texttt{0x01} & \texttt{0x08} & \texttt{0x05} & \texttt{0x02} & \texttt{0x09} & \texttt{0x01} & \texttt{0x01} & \texttt{0x04} \\ \texttt{0x04} & \texttt{0x01} & \texttt{0x08} & \texttt{0x05} & \texttt{0x02} & \texttt{0x09} & \texttt{0x01} & \texttt{0x01} \\ \texttt{0x01} & \texttt{0x04} & \texttt{0x01} & \texttt{0x08} & \texttt{0x05} & \texttt{0x02} & \texttt{0x09} & \texttt{0x01} \end{pmatrix}.$$

固有定数

その他実装に必要な定数を最後に示す。ブロック暗号における各段の定数 $Const^i, i = 1, \ldots, 10$ は S ボックスを使って次のように定義される。

$$\begin{aligned} Const^i_{0,c} &= s[8(i-1)+c], \quad \text{for} \quad c = 0, \ldots, 7, \\ Const^i_{r,c} &= 0, \quad \text{for} \quad r = 1, \ldots, 7, c = 0, \ldots, 7. \end{aligned}$$

なお，ハッシュ関数を構成するための，フィードバック値の初期値はバイト値 0 で埋められた 64 バイトデータである。

3.8 SHA-256

3.8.1 概要

SHA-256は、SHA-384, SHA-512とともに2000年に米国商務省技術標準機関 NISTにより提案され、2002年にFIPS180-2 [FIPS180-2] として制定された。

SHA-256は、ビット長が512ビットの倍数になるようにパディングされた任意のメッセージを入力として256ビットのハッシュ値を出力する。

SHA-256はISO/IEC 10118-3 [ISO/IEC10118-3] の国際規格にも採用されている。

3.8.2 技術仕様

SHA-256で使用される関数

SHA-256では、32ビットワードを入力変数および出力変数とする、以下の6種類の論理関数が用いられる。

$$\begin{cases} \mathrm{Ch}(x,y,z) & = (x \wedge y) \oplus (\neg x \wedge z) \\ \mathrm{Maj}(x,y,z) & = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \\ \Sigma_0^{256}(x) & = \mathrm{ROTR}^2(x) \oplus \mathrm{ROTR}^{13}(x) \oplus \mathrm{ROTR}^{22}(x) \\ \Sigma_1^{256}(x) & = \mathrm{ROTR}^6(x) \oplus \mathrm{ROTR}^{11}(x) \oplus \mathrm{ROTR}^{25}(x) \\ \sigma_0^{256}(x) & = \mathrm{ROTR}^7(x) \oplus \mathrm{ROTR}^{18}(x) \oplus \mathrm{SHR}^3(x) \\ \sigma_1^{256}(x) & = \mathrm{ROTR}^{17}(x) \oplus \mathrm{ROTR}^{19}(x) \oplus \mathrm{SHR}^{10}(x) \end{cases}$$

SHA-256の前処理

1. 入力メッセージ M について、メッセージ長が512ビットの倍数になるように初期パディングされたメッセージ

   $$M||1||0^k||l$$

   を計算する。ただし、l は M のメッセージ長のバイナリ表現(64ビット)、k は $l+1+k \equiv 448 \pmod{512}$ を満たす正の最小値である。

2. 初期パディングされたメッセージは N 個の512ビット単位のブロック $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16個の32ビット長のワード

   $$M^{(i)} = M_0^{(i)}||M_1^{(i)}||\cdots||M_{15}^{(i)}$$

   からなる。

3. 初期値として

$$
\begin{aligned}
H_0^{(0)} &= \texttt{6a09e667} \\
H_1^{(0)} &= \texttt{bb67ae85} \\
H_2^{(0)} &= \texttt{3c6ef372} \\
H_3^{(0)} &= \texttt{a54ff53a} \\
H_4^{(0)} &= \texttt{510e527f} \\
H_5^{(0)} &= \texttt{9b05688c} \\
H_6^{(0)} &= \texttt{1f83d9ab} \\
H_7^{(0)} &= \texttt{5be0cd19}
\end{aligned}
$$

を設定する。

SHA-256 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \leq i \leq N$ の順に以下を実行する。

1. 次式で定義する SHA-256 メッセージスケジュール関数を用いて拡張メッセージ W_t を計算する。

$$
W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \sigma_1^{256}(W_{t-2}) + W_{t-7} + \sigma_0^{256}(W_{t-15}) + W_{t-16} & 16 \leq t \leq 63 \end{cases}
$$

2. 8 個のバッファ変数を $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$
\begin{aligned}
a_0 &= H_0^{(i-1)} \\
b_0 &= H_1^{(i-1)} \\
c_0 &= H_2^{(i-1)} \\
d_0 &= H_3^{(i-1)} \\
e_0 &= H_4^{(i-1)} \\
f_0 &= H_5^{(i-1)} \\
g_0 &= H_6^{(i-1)} \\
h_0 &= H_7^{(i-1)}
\end{aligned}
$$

3. $0 \leq t \leq 63$ に対して以下の計算を繰り返す。

$$\begin{cases} T_1 &= h_t + \Sigma_1^{256}(e_t) + \mathrm{Ch}(e_t, f_t, g_t) + K_t^{256} + W_t \\ T_2 &= \Sigma_0^{256}(a_t) + \mathrm{Maj}(a_t, b_t, c_t) \\ h_{t+1} &= g_t \\ g_{t+1} &= f_t \\ f_{t+1} &= e_t \\ e_{t+1} &= d_t + T_1 \\ d_{t+1} &= c_t \\ c_{t+1} &= b_t \\ b_{t+1} &= a_t \\ a_{t+1} &= T_1 + T_2 \end{cases}$$

ただし、K_t^{256} は 32 ビットワードの定数 (FIPS PUB 180-2 参照) である。

4. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_0^{(i-1)} + a_{64} \\ H_1^{(i)} &= H_1^{(i-1)} + b_{64} \\ H_2^{(i)} &= H_2^{(i-1)} + c_{64} \\ H_3^{(i)} &= H_3^{(i-1)} + d_{64} \\ H_4^{(i)} &= H_4^{(i-1)} + e_{64} \\ H_5^{(i)} &= H_5^{(i-1)} + f_{64} \\ H_6^{(i)} &= H_6^{(i-1)} + g_{64} \\ H_7^{(i)} &= H_7^{(i-1)} + h_{64} \end{aligned}$$

で計算する。

上記手続き 1.〜 4. を N 回繰り返した最終的な 256 ビットの値

$$H^{(N)} = H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)} || H_4^{(N)} || H_5^{(N)} || H_6^{(N)} || H_7^{(N)}$$

がメッセージ M のハッシュ値である。

3.9 SHA-224

3.9.1 概要

SHA-224 は、2004 年 2 月に FIPS 180-2, Change Notice 1 [FIPS180-2a] に追加されたハッシュ長 224 ビットのハッシュ関数である。それまで FIPS 180-2 には、RSA-2048 と同等のセキュリティレベルに対応する 112 ビットセキュリティのハッシュ関数がなかった。

SHA-224 は、メッセージ長が 2^{64} ビット未満のメッセージを入力として 224 ビットのハッシュ値を出力する。演算は SHA-256 と同じく 32 ビット単位で行われる。

3.9.2 技術仕様

SHA-224 は以下の 2 点を除き、SHA-256 と同じ仕様である。

1. 初期値 $H^{(0)}$ を以下の値に設定する。

$$\begin{aligned} H_0^{(0)} &= \texttt{c1059ed8} \\ H_1^{(0)} &= \texttt{367cd507} \\ H_2^{(0)} &= \texttt{3070dd17} \\ H_3^{(0)} &= \texttt{f70e5939} \\ H_4^{(0)} &= \texttt{ffc00b31} \\ H_5^{(0)} &= \texttt{68581511} \\ H_6^{(0)} &= \texttt{64f98fa7} \\ H_7^{(0)} &= \texttt{befa4fa4} \end{aligned}$$

2. 圧縮関数を N 回繰り返した最終的な 256 ビットの値 $H^{(N)}$ の左 224 ビットの値

$$H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}||H_5^{(N)}||H_6^{(N)}$$

を SHA-224 でのメッセージ M のハッシュ値とする。

3.10 SHA-512

3.10.1 概要

SHA-512 は、SHA-256, SHA-384 とともに、2000 年に米国商務省 技術標準機関 NIST により提案され、2002 年に FIPS180-2 [FIPS180-2] として制定された。

SHA-512 のアルゴリズムは、SHA-256 のワード長 32 ビットを 64 ビットに変更し、メッセージスケジュール関数の繰り返し回数を増やしたものである。

SHA-512 は、ビット長が 1024 ビットの倍数になるようにパディングされた任意のメッセージを入力として 512 ビットのハッシュ値を出力する。

SHA-512 は ISO/IEC 10118-3 [ISO/IEC10118-3] の国際規格にも採用されている。

3.10.2 技術仕様

SHA-512 で使用される関数

SHA-512 では、64 ビットワードを入力変数および出力変数とする、以下の 6 種類の論理関数が用いられる。

$$\begin{cases} \mathrm{Ch}(x,y,z) &= (x \wedge y) \oplus (\neg x \wedge z) \\ \mathrm{Maj}(x,y,z) &= (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \\ \Sigma_0^{512}(x) &= \mathrm{ROTR}^{28}(x) \oplus \mathrm{ROTR}^{34}(x) \oplus \mathrm{ROTR}^{39}(x) \\ \Sigma_1^{512}(x) &= \mathrm{ROTR}^{14}(x) \oplus \mathrm{ROTR}^{18}(x) \oplus \mathrm{ROTR}^{41}(x) \\ \sigma_0^{512}(x) &= \mathrm{ROTR}^{1}(x) \oplus \mathrm{ROTR}^{8}(x) \oplus \mathrm{SHR}^{7}(x) \\ \sigma_1^{512}(x) &= \mathrm{ROTR}^{19}(x) \oplus \mathrm{ROTR}^{61}(x) \oplus \mathrm{SHR}^{6}(x) \end{cases}$$

SHA-512 の前処理

1. 入力メッセージ M について、メッセージ長が 1024 ビットの倍数になるように初期パディングされたメッセージ

   $$M||1||0^k||l$$

   を計算する。ただし、l は M のメッセージ長のバイナリ表現 (128 ビット)、k は $l + 1 + k \equiv 896 \pmod{1024}$ を満たす正の最小値である。

2. 初期パディングされたメッセージは N 個の 1024 ビット単位のブロック $\{M^{(i)}\}_{i=1}^{N}$ に分割される。ただし、各々の $M^{(i)}$ は、16 個の 64 ビット長のワード

   $$M^{(i)} = M_0^{(i)}||M_1^{(i)}||\cdots||M_{15}^{(i)}$$

   からなる。

3. 初期値として

$$
\begin{aligned}
H_0^{(0)} &= \texttt{6a09e667f3bcc908} \\
H_1^{(0)} &= \texttt{bb67ae8584caa73b} \\
H_2^{(0)} &= \texttt{3c6ef372fe94f82b} \\
H_3^{(0)} &= \texttt{a54ff53a5f1d36f1} \\
H_4^{(0)} &= \texttt{510e527fade682d1} \\
H_5^{(0)} &= \texttt{9b05688c2b3e6c1f} \\
H_6^{(0)} &= \texttt{1f83d9abfb41bd6b} \\
H_7^{(0)} &= \texttt{5be0cd19137e2179}
\end{aligned}
$$

を設定する。

SHA-512 のハッシュ値計算

N 個のメッセージブロック $M^{(1)}, \ldots, M^{(N)}$ の $M^{(i)}$ に対して、$1 \leq i \leq N$ の順に以下を実行する。

1. 次式で定義する SHA-512 メッセージスケジュール関数を用いて拡張メッセージ W_t を計算する。

$$
W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \sigma_1^{512}(W_{t-2}) + W_{t-7} + \sigma_0^{512}(W_{t-15}) + W_{t-16} & 16 \leq t \leq 79 \end{cases}
$$

2. 8 個のバッファ変数を $(i-1)$ 番目のハッシュ値 $H^{(i-1)}$ で初期化する。

$$
\begin{aligned}
a_0 &= H_0^{(i-1)} \\
b_0 &= H_1^{(i-1)} \\
c_0 &= H_2^{(i-1)} \\
d_0 &= H_3^{(i-1)} \\
e_0 &= H_4^{(i-1)} \\
f_0 &= H_5^{(i-1)} \\
g_0 &= H_6^{(i-1)} \\
h_0 &= H_7^{(i-1)}
\end{aligned}
$$

3. $0 \leq t \leq 79$ に対して以下の計算を繰り返す。

$$\begin{cases} T_1 &= h_t + \Sigma_1^{512}(e_t) + \mathrm{Ch}(e_t, f_t, g_t) + K_t^{512} + W_t \\ T_2 &= \Sigma_0^{512}(a_t) + \mathrm{Maj}(a_t, b_t, c_t) \\ h_{t+1} &= g_t \\ g_{t+1} &= f_t \\ f_{t+1} &= e_t \\ e_{t+1} &= d_t + T_1 \\ d_{t+1} &= c_t \\ c_{t+1} &= b_t \\ b_{t+1} &= a_t \\ a_{t+1} &= T_1 + T_2 \end{cases}$$

ただし、K_t^{512} は 64 ビットワードの定数 (FIPS PUB 180-2 参照) である。

4. i 番目の中間ハッシュ値を

$$\begin{aligned} H_0^{(i)} &= H_0^{(i-1)} + a_{80} \\ H_1^{(i)} &= H_1^{(i-1)} + b_{80} \\ H_2^{(i)} &= H_2^{(i-1)} + c_{80} \\ H_3^{(i)} &= H_3^{(i-1)} + d_{80} \\ H_4^{(i)} &= H_4^{(i-1)} + e_{80} \\ H_5^{(i)} &= H_5^{(i-1)} + f_{80} \\ H_6^{(i)} &= H_6^{(i-1)} + g_{80} \\ H_7^{(i)} &= H_7^{(i-1)} + h_{80} \end{aligned}$$

で計算する。

上記手続き 1.〜 4. を N 回繰り返した最終的な 512 ビットの値

$$H^{(N)} = H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)} || H_4^{(N)} || H_5^{(N)} || H_6^{(N)} || H_7^{(N)}$$

がメッセージ M のハッシュ値である。

3.11 SHA-384

3.11.1 概要

SHA-384は、SHA-256, SHA-512とともに2000年に米国商務省技術標準機関 NISTにより提案され、2002年にFIPS180-2 [FIPS180-2] として制定された。

SHA-384は、ビット長が1024ビットの倍数になるようにパディングされた任意のメッセージを入力として384ビットのハッシュ値を出力する。SHA-384はSHA-512とほぼ同じ仕様で、初期値と出力方法のみが異なる。

SHA-384はISO/IEC 10118-3 [ISO/IEC10118-3] の国際規格にも採用されている。

3.11.2 技術仕様

SHA-384は以下の2点を除き、SHA-512と同じ仕様である。

1. 初期値 $H^{(0)}$ を以下の値に設定する。

$$
\begin{aligned}
H_0^{(0)} &= \texttt{cbbb9d5dc1059ed8} \\
H_1^{(0)} &= \texttt{629a292a367cd507} \\
H_2^{(0)} &= \texttt{9159015a3070dd17} \\
H_3^{(0)} &= \texttt{152fecd8f70e5939} \\
H_4^{(0)} &= \texttt{67332667ffc00b31} \\
H_5^{(0)} &= \texttt{8eb44a8768581511} \\
H_6^{(0)} &= \texttt{db0c2e0d64f98fa7} \\
H_7^{(0)} &= \texttt{47b5481dbefa4fa4}
\end{aligned}
$$

2. 圧縮関数を N 回繰り返した最終的な512ビットの値 $H^{(N)}$ の左384ビットの値

$$H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}||H_5^{(N)}$$

を SHA-384 でのメッセージ M のハッシュ値とする。

第4章　安全性解析

4.1　ハッシュ関数の安全性

ハッシュ関数の安全性は、2.4 章に示した汎用攻撃に対する耐性と個々のアルゴリズムの解析結果の両面から評価することができる。

4.1.1　汎用攻撃に対する耐性指標

2.4 章に示したように、ハッシュ関数の汎用攻撃 (衝突攻撃、原像探索攻撃、第二原像探索攻撃) に対する攻撃計算量の上限はハッシュ長 n にのみ依存する。汎用攻撃に対する耐性指標を表 4.1 に示すように定義する。

表 4.1: 汎用攻撃に対する耐性指標

ハッシュ長	衝突攻撃耐性	原像探索攻撃耐性	指標
≤ 128 bit	$\leq 2^{64}$	$\leq 2^{128}$	C
≤ 160 bit	$\leq 2^{80}$	$\leq 2^{160}$	B
≤ 224 bit	$\leq 2^{112}$	$\leq 2^{224}$	A
$256 \leq n \leq 512$ bit	$\leq 2^{n/2}$	$\leq 2^{n}$	AA

電子政府推奨暗号リスト [CRYPTREC03] では、ハッシュ長が 256 ビット以上のハッシュ関数 (指標 AA) を推奨している。

4.1.2　ハッシュ関数解析による脆弱性指標

ハッシュ関数のアルゴリズムを解析し、衝突攻撃、原像探索攻撃、第二原像探索攻撃に対する実際の攻撃計算量を評価した結果、表 4.1 に示す計算量の上限値より少ない計算量での攻撃が見つかった場合に、そのハッシュ関数アルゴリズムは「破れた」または「脆弱性が見つかった」という。

但し、その解析結果が、繰り返し型ハッシュ関数における「圧縮関数」レベルでの評価なのか、「ハッシュアルゴリズム全体」に対する評価なのかを区別することが重要である。

2.3.1 章で述べたように、繰り返し型ハッシュ関数 H では、圧縮関数 f を繰り返し適用することで長いメッセージのハッシュ値を計算する (図 4.1 参照)。

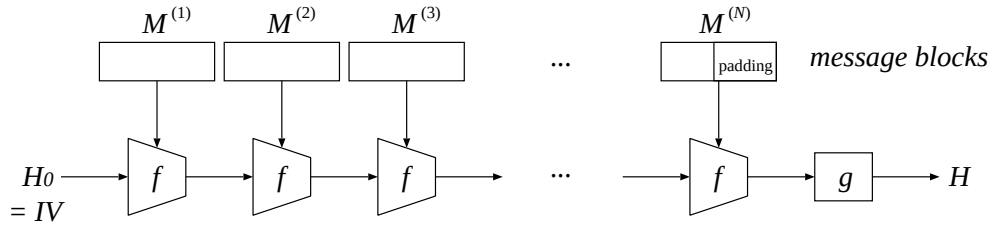


図 4.1: 繰り返し型ハッシュ関数 (再掲)

圧縮関数の衝突　ある初期ベクトル (IV) に対し、圧縮関数の出力が一致するような (1 ブロックの) 入力メッセージのペアを発見できた場合、すなわち

$$f(IV, X) = f(IV, X')$$

なる入力メッセージペア (X, X') を発見した場合、「圧縮関数の衝突が発見された」という。

圧縮関数の擬似衝突　初期ベクトル IV を任意に選べる条件下で、異なる初期ベクトル $IV'(\neq IV)$ に対し、

$$f(IV, X) = f(IV', X')$$

なる入力メッセージペア (X, X') を発見した場合、「圧縮関数の擬似衝突が発見された」という。

短縮／変形版圧縮関数の衝突　3 章で示したように、各ハッシュ関数の圧縮関数は複数のステップの繰り返しからなっており、繰返し回数 (段数) を削減したり、メッセージスケジュール関数を省略した圧縮関数の衝突解析を行っている場合もあるので注意が必要である。すなわち、圧縮関数 f を変形した関数 f' について

$$f'(IV, X) = f'(IV, X')$$

なる入力メッセージペア (X, X') を発見した場合、本報告書では「短縮／変形版圧縮関数に衝突が発見された」とする。

圧縮関数の衝突とハッシュ関数の衝突の関係　圧縮関数に衝突が見つかった場合、ハッシュ関数の衝突につながるかどうかはハッシュ関数の仕様に定められている初期値 IV_0 に対して以下のような関係が成り立つかどうかによる。

$$\begin{aligned} H(X) &= f(f(IV_0, X), P) \\ &= f(f(IV_0, X'), P) \\ &= H(X') \end{aligned}$$

ここで、P は X 及び X' に付加されるメッセージである。

衝突攻撃による安全性への影響度　このような衝突攻撃の攻撃結果が与える安全性への影響度を表 4.2 に定義する。

表 4.2: 衝突攻撃による安全性への影響度

評価結果	影響度
衝突攻撃につながる問題点は見つかっていない	0
短縮／変形版圧縮関数に衝突が発見された	1
圧縮関数に近似衝突が発見された	2
圧縮関数に擬似衝突が発見された	3
圧縮関数に衝突が発見された	4
ハッシュ関数に近似衝突が発見された	5
ハッシュ関数に擬似衝突が発見された	6
ハッシュ関数に衝突が発見された	7

これらの「汎用攻撃に対する耐性指標」と「ハッシュ関数解析結果が与える安全性への影響度」を組み合わせることで、ハッシュ関数の安全性をより適切に表現することができる。例えば、「ハッシュ長 160 ビットのハッシュ関数の圧縮関数に、(2^{80} より少ない計算量で) 衝突が発見された」場合、

このハッシュ関数の安全性は B-4

と表わすことができる。もし、衝突攻撃につながる問題点が何も見つかっていないハッシュ関数があったとしても、そのハッシュ長が 128 ビットであれば、このハッシュ関数の安全性は C-0 であり、汎用攻撃に対する耐性の観点から、長期利用には望ましくない、と判断できる。

4.2 既知の解析結果

本章では、MD4, MD5, RIPEMD, RIPEMD-128, RIPEMD-160, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, Whirlpool の各アルゴリズムについて、これまでに知られている解析結果をまとめる。

4.2.1 MD4

MD4 の解析は、1995 年以前には den Boer と Bosselaers [DB92] による解析 (圧縮関数 3 ラウンド中の最終 2 ラウンドの衝突攻撃)、及び Vaudenay [V95] による近似衝突 ("almost-collision"[1]) しか知られていなかったが、1996 年に Dobbertin により約 2^{20} 回の MD4 圧縮関数の計算量で衝突が見つけられることが発表された [D96a, D98]。この時点で既に「MD4 については利用すべきでない [D98]」とされていたが、2004 年、Wang ら [WFLY04, WLFCY04, WY04] により、極めて少ない計算量で衝突が見つけられることが発表された。MD4 の安全性は十分でなく、衝突発見困難性が求められる電子署名などのアプリケーションでの利用は避けるべきと思われる。

年	発表者 及び 解析結果	安全性
1992	den Boer, Bosselaers による短縮版圧縮関数の衝突攻撃 [DB92]	C-1
1995	Vaudenay による近似衝突攻撃 [V95]	C-5
1996	Dobbertin による衝突攻撃 [D96a]	C-7
2004	Wang らによる衝突攻撃 [WFLY04, WLFCY04, WY04]	C-7

4.2.2 MD5

MD5 の解析については、1993 年に den Boer と Bosselaers [DB94] により、MD5 圧縮関数に対して擬似衝突 (pseudo-collision) が発表され、1996 年に Eurocrypt'96 rump session にて、Dobbertin により、MD5 圧縮関数に対して衝突が Pentium PC で約 10 時間の計算量 (当時) で見つけることができると発表された [D96b]。この MD5 圧縮関数の衝突は、MD5 のアルゴリズム全体では擬似衝突 (pseudo-collision) になる。すなわち、仕様とは異なる別の初期値 IV' に対して、異なるメッセージからハッシュ値の一致する衝突を見つけることができるというものである。

Dobbertin は、この攻撃を踏まえて MD5 の安全性について次のように述べている。「この攻撃で MD5 の実アプリケーションが脅威にさらされるというわけではないが、その時期は近づいたと思われる。... 将来、衝突発見困難性が求められるアプリケーションにおいて MD5 を実装すべきでない時期がやってくるだろう [D96c]。」

その後、2004 年に Wang ら [WFLY04, WY04] により、IBM P690 で約 1 時間で衝突が見つけられることが発表された。この攻撃は、仕様通りの初期値 IV を用いて、ハッシュ値の一致する異なる 2 つの 1024 ビットメッセージを見つけるものである。

[1] 文献 [V95] では"almost-collision" の用語が使われていたが、本報告書の「近似衝突 (near-collision)」と同義.

この攻撃を利用して、実際に X.509 公開鍵証明書の偽造が可能であることも報告されている [LWW05]。衝突発見困難性が求められる電子署名での MD5 の利用は控えるべきと思われる。

年	発表者 及び 解析結果	安全性
1993	den Boer, Bosselaers による圧縮関数の擬似衝突攻撃 [DB94]	C-3
1996	Dobbertin による圧縮関数の衝突攻撃 [D96b] (= ハッシュ関数の擬似衝突攻撃)	C-4 (=C-6)
2004	Wang らによる衝突攻撃 [WFLY04, WY04]	C-7

4.2.3 RIPEMD

RIPEMD の解析については、1995 年に Dobbertin による、3 ラウンド中の最初の 2 ラウンド及び最終 2 ラウンドに対する衝突攻撃が知られていたが [D97]、2004 年に Wang ら [WFLY04, WLFCY04, WY04] により、RIPEMD の完全な仕様に対して衝突が見つけられることが発表された。

衝突発見困難性が求められる電子署名などでの RIPEMD の利用は控えるべきと思われる。

年	発表者 及び 解析結果	安全性
1995	Dobbertin による短縮版圧縮関数の衝突攻撃 [D97]	C-1
2004	Wang らによる衝突攻撃 [WFLY04, WLFCY04, WY04]	C-7

4.2.4 RIPEMD-128

RIPEMD-128 の安全性解析に関する文献は、我々の調査した限りでは見つかっておらず、現在のところ、問題点は発見されていない。

但し、ハッシュ長が 128 ビットであるため、衝突攻撃による攻撃計算量は高々2^{64}であり、長い将来に渡って利用されるアプリケーションには適さない。

年	発表者 及び 解析結果	安全性
2006	現在のところ、問題点は見つかっていない。	C-0

4.2.5 RIPEMD-160

RIPEMD-160 の安全性評価は [C06] で内部の右ライン, 左ラインの関数について差分パスの評価が行われた以外、我々の調査した限りでは見つかっておらず、現在のところ、問題点は発見されていない。

但し、ハッシュ長が 160 ビットであるため、衝突攻撃による攻撃計算量は高々2^{80}であり、将来に渡って安全であるとは保証できない。

年	発表者 及び 解析結果	安全性
2006	現在のところ、問題点は見つかっていない。	B-0

4.2.6 SHA (SHA-0)

SHA (SHA-0) に対する解析については、1995 年に Chabaud と Joux により、SHA-0 の圧縮関数の衝突を 2^{61} の計算量で見つけられることが示されたのが Birthday attack より少ない計算量での最初の攻撃結果である [CJ98]。その後、2004 年に Biham と Chen により、SHA-0 の近似衝突 (near-collision) [2] や、65 段に短縮した SHA-0 の衝突などが発見された [BC04a]。さらに、2004 年 8 月に Joux らにより、SHA-0 の衝突が発見された (攻撃計算量は約 2^{51})[JCLJ04]。Biham らはその後の解析をふまえ、「衝突が脅威となるアプリケーションで SHA-0 を利用すべきでない [BC04a, updated version]」と述べている。

また、2004 年に発表された Wang らのレポート [WFLY04, WY04] には、SHA-0 の衝突が約 2^{40} 回の SHA-0 の圧縮関数の計算量で見つけられるとの記述もある。衝突発見困難性が求められる電子署名などのアプリケーションでは、SHA(SHA-0) の利用は控えるべきと思われる。

年	発表者 及び 解析結果	安全性
1995	Chabaud, Joux による圧縮関数の衝突攻撃 [CJ98]	B-4
2004	Biham, Chen による近似衝突攻撃 [BC04a]	B-5
2004	Joux らによる衝突攻撃 [JCLJ04]	B-7
2004	Wang らによる衝突攻撃 [WFLY04, WY04]	B-7

4.2.7 SHA-1

SHA-1 に対する解析については、2004 年まで Birthday attack より少ない計算量での攻撃は知られていなかったが、2004 年に Crypto 2004 rump session にて、Biham と Chen により、36 段に短縮した SHA-1 の圧縮関数の衝突、45 段に短縮した SHA-1 の圧縮関数の近似衝突 (near-collision) などが Birthday attack より少ない計算量で見つけられることが発表された [BC04b]。

2005 年 2 月に Rijmen と Oswald[RO04] により、SHA-1 のメッセージ拡大をうまく扱えるように Chabaud と Joux の攻撃 [CJ98] を改良し、53 段に短縮した SHA-1 の圧縮関数の衝突を発見したことが示され、さらに Wang らにより SHA-1 の衝突が 2^{69} 回の SHA-1 の計算より少ない計算量で発見できることが報告された [WYY05]。

Wang の衝突攻撃の概要と調査の背景

攻撃の概要　Wang の衝突攻撃は、SHA-1 ハッシュ関数に期待されてきた衝突発見困難性に対する攻撃である。衝突に対する攻撃とは、定義される SHA-1 ハッシュ関数についてどのようなメッセージでもよいので出力が同じとなる異なるメッセージを生成することである。一般にはこの衝突対が 1 組あれば無数の衝突対を生成できる。

[2] 160 ビットのハッシュ値のうち、142 ビットまでが一致する.

Wang らの衝突攻撃の目的も上記衝突を発見することである。Wang らの手法についてより具体的に説明する。Wang らの攻撃では、攻撃の過程で無数に生成されるメッセージ M に対してそのうちのひとつが (さらに攻撃の過程で M に対して一意に生成される同じ長さの) 別の M' と衝突を起こすことが期待できることを使った攻撃である。一般にはこのような攻撃には非現実的な数の M の生成をする必要があるが、Wang らの手法ではその数をかなり現実的なレベルまで削減することに成功したものである。

その手法は暗号解読における差分解読法という手法をベースに使っている。メッセージ M とメッセージ M' の数値としての「差」を考える; これをメッセージ差分 $\Delta M(= M' - M)$ と呼ぶ。攻撃の発見を主張/検証するには、まずこのメッセージ差分が SHA-1 ハッシュ値の計算過程で矛盾なく消失するように、内部変数の差分を決定し、その具体的方法を示さねばならない。Wang らはこれを具体的に示した。

この差分パスを形成する上での Wang らの手法の特徴に、1 つのメッセージブロック (512 ビット) で構成されるメッセージに対してではなく、2 つのメッセージブロック (1024 ビット) で衝突を生成するための差分パスを構成した点がある。最初に SHA-1 に入力されるメッセージブロックを第 1 メッセージブロック、次に入力されるものを第 2 メッセージブロックと呼ぶ。

攻撃手順は以下の通り。

Step 0 差分パスの発見
攻撃の対象であるハッシュ関数の仕様を詳細に分析することで衝突を起こすような差分パスを発見すること。一度見つかればこれをもとに以降のステップを独立に実行することでさまざまな衝突を生成できる。この解析には高度な技術と解析の能力・コストが必要。このフェーズが Wang らによって行われた。

Step 1 第 1 メッセージブロック $M1$ に対するペア $M1'$ の生成

1-1 $M1$ を任意に生成する。

1-2 $M1$ に対して、[十分条件] なるビット単位の修正を施す

1-3 修正した $M1$ から $M1'$ を計算する。

1-4 $M1$, $M1'$ をそれぞれ SHA-1 の処理を行う。その結果が期待される差分パス (わずかな確率でしか成立することが期待できない) を満たしているかどうか判定する。していれば Step 2 へ。

1-5 上記 [1-4] が成功するまで 1-1〜1-4 を繰り返す。

Step 2 第 2 メッセージブロック $M2$ に対する衝突ペア $M2'$ の生成

2-0 $M1$, $M1'$ の処理結果をブロックの処理開始データにセット

2-1,...,2-5 Step 1-1〜1-5 とほぼ同じことを行う— $(M1, M1')$ をそれぞれ $(M2, M2')$ と読み替える。一部メッセージ修正の方法, メッセージ差分などが第 1 ブロックと異なる場合がある。

Spep 3 [Step 1-4], [Step 2-4] で成功した $M1$, $M1'$, $M2$, $M2'$ はSHA-1 の衝突データそのものである。メッセージ対 $(M1||M2, M2||M2')$ を出力する。

Step 0 については、Wang らが行っており特別な理由がない限り攻撃者は Step 0 を行う必要はない。さらに Step 2 以降に必要な計算量は Step 1 で必要とされる計算量を上回らないことが示されている。よって SHA-1 の衝突データの生成に必要な計算量は Step 1 で必要とされる計算量である。

調査の背景

SHA-1 の安全性の検討には、以下の技術的な検討と考察が必要である。

検討 1 Wang らの結果の追試 (正しいのか、真偽判定に不明点はないか)

検討 2 最終的な攻撃の実現可能性 (改良の余地、攻撃計算量、実装容易性)

安全性の評価は必ずしも検証が自明であるとは限らず、特に最終的な実験結果が伴わないような理論結果についてはその検証が必要である (でなければ、仮に主張が誤っていた場合、本来安全なハッシュ関数について攻撃可能であると評価しかねない)。Wang らの攻撃はこれら検証が自明ではなく、その追試には高度な専門技術が必要である。

次に (2) の実現可能性について補足説明する。上記 (1) として正しい、あるいは改良の余地などが見通せたとしても、この攻撃が実際に実現、実行、終了可能かどうかの判定も自明ではない。今後の改良の余地や、もし改良がある場合の改良の度合い、そしてこれらを包含した形での実現可能性 (ハードウェア、コーディング、計算時間など) を検討する必要がある。

把握している問題点と結論 ハッシュ関数・暗号利用モード調査 WG では、下記の外部評価依頼を行った。

評価 1 電気通信大学, 電気通信学部 情報通信工学科ハッシュ関数 (SHA-1) の安全性評価及び攻撃手法整理

評価 2 グラーツ工業大学 (オーストリア) Evaluation of SHA-1, SHA-224, SHA-256 SHA-384, and SHA-512

依頼調査として、評価内容を主に 2 つに大類別した [検討 1][検討 2]。これらそれぞれを [評価 1], [評価 2] での中心的評価項目として扱うものとした (ただし評価は「指定した [検討 1,2] に限定」はされない)。以下に [評価 1][評価 2] で得られた評価・検討結果についてまとめる。

Wang らの結果の考察と問題点 [評価 1] により攻撃の大筋については検証が行われた。しかし、評価を詳細に行っていく過程で一部の技術的指摘に不明点などがあった。この評価結果について、ハッシュ関数・暗号利用モード調査 WG において審議しした。審議結果は以下の通り。

Wang らの攻撃アルゴリズムに関する技術的不明点

a. 局所衝突の探索法とディスターバンスベクトルの最適性

b. 内部変数差分の決定法

c. 計算量見積もりの妥当性

技術的不明点 a は、これを明らかにすることにより、差分パスが他にも存在するのかどうか、あるいは今後どう改良の余地があるのかが解明される。

技術的問題点 b は (a にも書いた攻撃の余地があるかどうかにも関連するが) 具体的な攻撃の実現について重要な技術要素である。第 1 ブロックのメッセージ修正についてはその記述があるが、第 2 ブロックのそれについては大枠のみ記載されているだけである。技術的問題点 c は、仮に動いた場合の必要となる計算量の計算が技術的に複雑であることに起因し、これも現状開示された情報による検証が自明でない。

これらの問題点を主な検討事項に踏まえながら、[評価 2] ならびに [評価 1] について検討し、以下のように技術的問題点別に整理した。

局所衝突の探索法とディスターバンスベクトルの最適性 (技術的不明点 a) 算術差分の決定では、第 1 メッセージブロックに対する差分 $\Delta M1$ の導出が目的である。この結果として、メッセージ差分 $\Delta M1$ が求まるだけでなく、(衝突十分条件とも呼ばれる) 確率的に充足を期待する条件が求まる。この条件にかかる確率を積算することで全体での成立確率を算出でき、結果として必要な M の試行の数の見積もりに繋がる。この算術差分を導出するにあたって以下のような大枠で探索が行われたとされる。

探索手順の概要

1. 局所衝突とそのルールについて整理し、可能な局所衝突を場合分けする
2. 局所衝突を組み合わせる。なおこれには局所衝突の発生位置を指定するアンカーのパターンである、ディスターバンスベクトル (差分挿入位置ベクトル) で記述する。
3. 内部変数の差分 (とメッセージブロック差分 $\Delta M1$ も) が求まる
4. 内部変数差分を実現するための十分条件 (sufficient condition) が求まる

局所衝突の見直し 全体の差分の構成方法として、まず SHA-1 の内部関数が任意の 6 ステップで局所衝突 (ローカルコリジョン) という衝突のパターンがあり、これを組み合わせることで全体の衝突を構成する指針で、差分の構成をしている。これは既存の研究結果の流用である。まず、Wang らの新規の結果として、この局所衝突を再度見直したことがある。従来衝突には使えないであろうとされていたタイプや、より複雑なものについても複雑なルールや解析を適用することで、より豊富な組み合わせで差分を構成することができた。結果として効率を下げるなどにつながった。

局所衝突の組合せ探索とパスの最適性　ローカルコリジョンの発生位置を決めているのが手順 2 で決定されるディスターバンスベクトルである。これから攻撃計算量に相当する尺度の大部分 (一部はメッセージ修正にも影響する) が算出されるため、攻撃計算量の観点からこの最適性を問う必要がある。実際のところ具体的な探索についての情報開示は十分でなかった。

[評価 2] にはこの最適性を裏付ける評価結果が得られている。具体的には、十分条件の数とその充足確率、ならびに各ラウンドのブール関数の近似ルールを限定し簡単なモデル化したものについて、符号理論における最小距離問題を解いた。この手法では、符号語のハミング重みが攻撃の計算量のべき指数に近似される関係があるため、最小符号を求めることが（いくつかの条件下ではあるが）最良の差分を求めることにあたり、今回の差分に相当する符号語が一致した。

以上のことから、本質的な差分パスとそれから計算される充足確率の最適性は極めて高く、この観点からの劇的な攻撃の改良は困難と考えられる。

内部変数差分の決定法 (技術的不明点 b)　Step 0 で求まった差分パスを実現させるにあたり、第 1 ラウンドで特別に充足を強制させる。処理の最初であるため攻撃者が生成したメッセージに諸々のビット修正を加えることができる。これがメッセージ修正である。

例えば第 1 ブロックのレジスタ入力については、SHA-1 の仕様である初期値と矛盾なく差分を制御する必要がある。初期値の制御の具体的方法は Wang らの報告に明記されており、攻撃の実装上第 1 ブロックは問題にはならない。しかし、SHA-1 の安全性を検討する上で、その具体的方法の導出法が (攻撃改良の可能性の観点から) 重要である。この点で Wang の論文では詳細には示されていない。

さらにこのメッセージ修正は、第 2 メッセージブロックでも発生するが、第 2 メッセージブロックに対するメッセージ修正はその指針の記載があるのみで修正方法の具体的方法 (やその例) の記載がない。この不明点は単に今後の安全性評価の上での不明点のみならず、現状の攻撃アルゴリズム自身の実装可能性にも影響する場合がある。

第 1 ブロックのメッセージ修正については実際に改良の余地があった由が報告され、2006 年 1 月時点で改良したメッセージ修正について部分的に検証されている。また第 2 ブロックにおけるメッセージ修正の実現性については、評価 2 の解析での指摘も特に大きくないことから、攻撃実装可能性に影響するほどの不明点ではないと考える。

この探索攻撃のアルゴリズムは、MD5 に対するものと根本的には同一である。また MD5 については、1-1 及び 2-1 については明らかにされていない状況であったが、Wang の発表から 1 年程度で別の研究者グループにより明らかにされた。このような状況を踏まえると、これら不明な点は近い将来に明らかになると予想できる。

上記の予想から、内部変数に与える差分の決定法に関する問題点については「1 メッセージブロック目と 2 メッセージブロック目における 1 ラウンド目内部変数に対する差分の決定手法が明らかになっていないが、攻撃に必要な計算量の見積もりやアルゴリズムの検証において障害にならない」と結論した。

計算量見積もりの妥当性　Wang の衝突攻撃のアルゴリズムには、「内部変数差分の決定法 (技術的不明点 B)」があるものの、これらは計算量見積もりに影響を与えず、コリジョン探索に必要な計算量は 2^{69} 回の SHA-1 実行であることが [C06, WYY05a] で確認された。結論として「Wang の SHA-1 の衝突攻撃に必要な計算量見積もりは 2^{69} 回の SHA-1 実行で妥当である」と判断した。

Wang らが発表した SHA-1 の計算量が 2^{63} へ削減する攻撃手法 [WYY05b] は、[C06] より部分的に検証されている。この手法は、ローカルコリジョンにおける差分パスをずらすことにより、結果的に Message modification においてステップ 1-22 がステップ 1-27 に延長され計算量の削減を実現しているが、その他の部分での計算量増加について考察が十分とはいえない。しかしながら上記技術的問題点が明らかになると、改良法による差分確率と内部変数への差分の検証やディスターバンスベクトルの最適性の確認が可能になり探索実行条件が整うと予想される。

攻撃の実現可能性　Wang らが発表した衝突攻撃のアルゴリズムは、計算量見積もりが確認できる程度に明らかになっているが、2006 年 2 月の時点では第三者が実装可能な状況にはない。しかし、攻撃アルゴリズムの大筋については確認できており、不明な部分も近い将来明らかになると予想する。

また、衝突探索に必要な攻撃アルゴリズムは、

1. 高い並列処理度
2. 極めて小さい必要メモリ量

という二つの特徴から、計算量単体評価の実現性と、攻撃全体の実現性のギャップは極めて小さいと考え、「Wang らが発表した SHA-1 の計算量が 2^{63} の攻撃手法は、近い将来に第三者による実装が可能になり、極めて大きな脅威となると考えられる」と結論づけた。

衝突生成データの条件　以上の攻撃の概要から導かれる衝突生成のための条件についてまとめる．現状の攻撃により衝突が生成されるためには異なる二つのメッセージが少なくとも次の条件を揃える場合である．

1. メッセージ対は両方が同じ長さでかつパディング後のブロック (512 ビット) 数が 2 以上であること，かつ
2. メッセージブロックとして “異なるブロック” が連続 2 ブロック (差分ブロック列と呼ぶ)，あるいはその組合せであること，かつ
3. 差分ブロック列でのビットのビットの異なるビットと同値のビットのパターンがある特定のパターンであること，かつ
4. 差分ブロックにおける片方のメッセージ 100〜200 ビットの値が決められた値であること．

上記のような条件を満たさないメッセージ対は今回の攻撃の直接の対象にはならない．

今後の改良と新しい結果の方向性について　差分パスの最適性については，攻撃を検討した上でもっともらしいある条件下で最適性が証明されている．しかし，メッセージ修正やその方法，効率の点では，実際の計算機実装などにも依存する要素が大きく，細かい実行時間評価は今後の研究成果で明らかになってくると思われる．

今後は，第二ブロックでの具体的なメッセージ修正ルールも含め，これらの実験的側面での解析が進むと考える．

年	発表者 及び 解析結果	安全性
2004	Biham, Chen による 36 段短縮版圧縮関数の衝突攻撃 [BC04b]	B-1
2004	Rijmen, Oswald による 53 段短縮版圧縮関数の衝突攻撃 [RO04]	B-1
2005	Wang, Yin, Yu による衝突攻撃 [WYY05a]	B-7

4.2.8 Whirlpool

2003 年に白井、渋谷により、Whirlpool のアルゴリズムで採用されている diffusion matrix の分岐数 (branch number) が、提案者らの主張していた 9 ではなく、実際は 8 であったことが示され [SS03]、diffusion matrix が変更された[3]。しかしながらこの変更により、当初示されていた安全性解析には影響はない。

Whirlpool は 2000 年に提案された後、NESSIE プロジェクト [K02a, K02b]、及び CRYPTREC プロジェクト [C06] で評価されたが、ハッシュ関数の安全性に直接関わる結果は発表されていない。今後の解析に注意していく必要がある。

年	発表者 及び 解析結果	安全性
2006	現在のところ、問題点は見つかっていない。	AA-0

4.2.9 SHA-256/224

SHA-256 に対する解析は、Gilbert と Handshuh による評価 [GH02a] が最初の公開文書である。この中で、SHA-256 を含む全ての SHA-2 ファミリーの衝突を見つけるための、最も確率の高い”differential collision pattern” の確率が 2^{-66} であり、SHA-256 の攻撃計算量は 2^{132} となることから、SHA-256 は衝突攻撃に対して耐性をもつという結論が導かれている。

その後、Hawkes ら [HPR04] により、この “differential collision pattern” の確率は、算術加算をオペレータとする差分定義のもとでは、より大きな値となると主張されている。但し、Hawkes らの解析ではメッセージスケジュール関数の解析は不十分であり、単純な解析による楽観的な攻撃計算量が導かれているにすぎない [HPR04]。

以上の研究の他にも、SHA-256 の簡易版の解析 [KPPRR05, YB05] や SHA-256 のメッセージスケジュールの解析 [PRR05] などの研究が行われているが、SHA-224 及び SHA-256 の安全性を脅かす結果は得られていない。よって、SHA-224 及び SHA-256

[3]変更後の仕様が ISO/IEC 10118-3 に採用されている。

の衝突攻撃に対する安全性を評価するには、引続き今後の動向に注意していく必要がある。

年	発表者 及び 解析結果	安全性
2004	Hawkes らによる圧縮関数の解析 [HPR04]	A-0 (SHA-224) AA-0 (SHA-256)
2005	Pramstaller らによる圧縮関数の解析 [PRR05]	A-0 (SHA-224) AA-0 (SHA-256)

4.2.10 SHA-384/512

SHA-384, SHA-512 に対する解析は、Gilbert と Handshuh による評価 [GH02b] が最初の公開文書である。この中で、SHA-384/512 を含む全ての SHA-2 ファミリーの衝突を見つけるための最も確率の高い”differential collision pattern” の確率が 2^{-66} であり、SHA-384/512 の攻撃計算量は 2^{264} となることから、SHA-384/512 は衝突攻撃に対して耐性をもつという結論が導かれている。

その後、Hawkes ら [HPR04] により、この “differential collision pattern” の確率は、算術加算をオペレータとする差分定義のもとでは、より大きな値となると主張されている。但し、Hawkes らの解析ではメッセージスケジュール関数の解析は不十分であり、単純な解析による楽観的な攻撃計算量が導かれているにすぎない [HPR04]。

よって、SHA-384 及び SHA-512 の衝突攻撃に対する安全性を評価するには、引続き今後の動向に注意していく必要がある。

年	発表者 及び 解析結果	安全性
2004	Hawkes らによる圧縮関数の解析 [HPR04]	AA-0

4.2.11 まとめ

以上の章で述べた、個別のアルゴリズムに対する既知の解析結果をもとに、汎用攻撃に対する耐性と、解析結果が与える安全性への影響度の両面から見た、現時点でのハッシュ関数の安全性を表 4.3 及び図 4.2 に示す。

表 4.3: 既知の解析結果にもとづくハッシュ関数の安全性

ハッシュ関数	安全性
MD4	C-7
MD5	C-7
RIPEMD	C-7
RIPEMD-128	C-0
RIPEMD-160	B-0
SHA(SHA-0)	B-7
SHA-1	B-7
SHA-224	A-0
SHA-256	AA-0
SHA-384	AA-0
SHA-512	AA-0
Whirlpool	AA-0

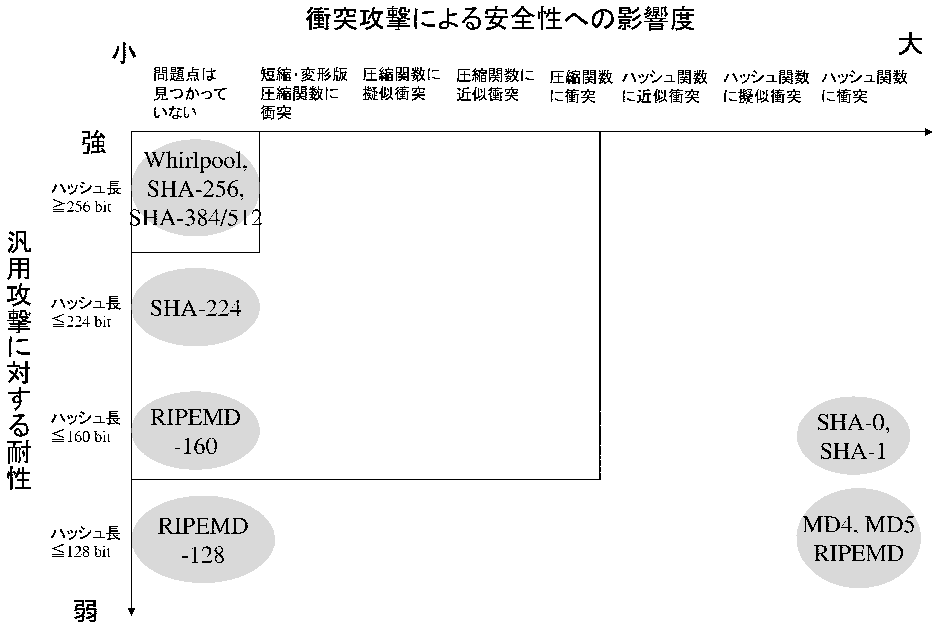


図 4.2: 既知の解析結果にもとづくハッシュ関数の安全性

第5章　ソフトウェア実装性能

表 5.1 及び図 5.1 に主な専用ハッシュ関数のソフトウェア実装性能を示す。実装性能は、入力メッセージのバイト当たりで必要なサイクル数で示している。すなわち、数値が小さいほど高速であることを意味する。

表 5.1: ソフトウェア実装性能 (cycles/byte)

アルゴリズム	PIII/Win98 (系列１)	PIII/Linux (系列２)	PIII/Win00 (系列３)	P4/Linux (系列４)
MD4	–	4.7	4.5	6.4
MD5	3.66	7.2	6.8	9.4
RIPEMD-128	6.64	–	–	–
RIPEMD-160	11.34	18	16	26
SHA (SHA-0)	–	15	12	23
SHA-1	8.30	15	13	25
SHA-256	20.59	39	39	40
SHA-384	–	83	74	122
SHA-512	40.18	83	74	122
Whirlpool	36.52	46	73	60

PIII/Win98: Pentium III (800MHz, 256MB RAM), Windows 98, Visual C++, MASM 6.15. 文献 [NM03] に示されている様々な実装方法のうち、最速のものを引用.

PIII/Linux: Pentium III (450MHz, 256MB RAM), Linux 2.4.17, gcc 3.1.1 など. 測定環境は何種類かあるが、サイクル数はほぼ同一 [NESSIE03].

PIII/Win00: Pentium III (850MHz, 256MB RAM), Windows 2000, gcc 2.95.3 など. 測定環境は何種類かあるが、サイクル数はほぼ同一 [NESSIE03].

P4/Linux: Pentium 4 (1.8GHz), Linux 2.4.0, gcc 2.95.2 など. 測定環境は何種類かあるが、サイクル数はほぼ同一 [NESSIE03].

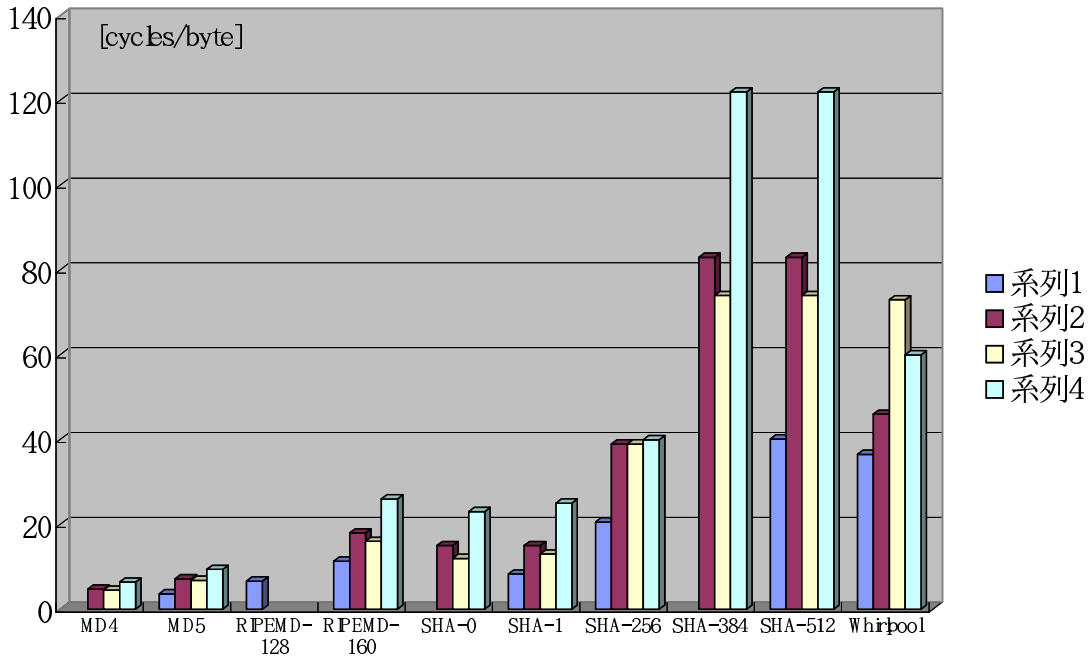


図 5.1: ソフトウェア実装性能 (cycles/byte)

第6章　まとめ

本報告書では、暗号学的ハッシュ関数、中でも特に広く利用されている専用ハッシュ関数について、その技術的特徴と採用・標準化動向について調査し、最近の解析結果を踏まえた安全性評価をまとめた。

依然として広く使われていると思われる MD4, MD5 の他、ISO/IEC 10118-3 や FIPS 180-2 として標準化されている RIPEMD-128, RIPEMD-160, SHA-1, SHA-224/256, SHA-384/512, Whirlpool についてその安全性を表 6.1, 表 6.2 にまとめた.

表 6.1: ハッシュ関数に関する安全性のまとめ (1/2)

名称	ハッシュ長 (bit)	安全性	標準
MD4	128	C-7	RFC 1320
		極めて容易に衝突を発見する方法が存在する。衝突発見困難性が求められる応用での利用は避けるべき。	
MD5	128	C-7	RFC 1321
		容易に衝突を発見する方法が存在する。衝突発見困難性が求められる応用での利用は控えていくべき。	
RIPEMD-128	128	C-0	ISO/IEC 10118-3
		現在のところ問題点は見つかっていないが、ハッシュ長が 128 ビットであるため、長い将来に渡る応用には不適。	
RIPEMD-160	160	B-0	電子政府推奨暗号 注) ISO/IEC 10118-3
		現在のところ問題点は見つかっていないが、ハッシュ長が 160 ビットであるため、将来に渡る安全性は保証できない。	

注) RIPEMD-160 については、電子政府推奨暗号リストにおいて「新たな電子政府用システムを構築する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。」という注釈がついている。

表 6.2: ハッシュ関数に関する安全性のまとめ (2/2)

名称	ハッシュ長 (bit)	安全性	標準
SHA-1	160	B-7	電子政府推奨暗号 注)
		2^{69} 回の SHA-1 の計算より少ない計算量で衝突が発見できることが報告された。技術詳細は未公開であり、調査が必要。	FIPS 180-2 ISO/IEC 10118-3
SHA-224/256	224/256	A-0/AA-0	電子政府推奨暗号
		現在のところ致命的な問題点は見つかっていないが、今後の動向に注意が必要。	FIPS 180-2 ISO/IEC 10118-3
SHA-384/512	384/512	AA-0	電子政府推奨暗号
		現在のところ致命的な問題点は見つかっていないが、今後の動向に注意が必要。	FIPS 180-2 ISO/IEC 10118-3
Whirlpool	512	AA-0	ISO/IEC10118-3
		現在のところ問題点は見つかっていないが、提案者以外による安全性解析は発表されていないため、今後の動向に注意が必要。	

注) SHA-1 については、電子政府推奨暗号リストにおいて「新たな電子政府用システムを構築する場合、より長いハッシュ値のものが使用できるのであれば、256 ビット以上のハッシュ関数を選択することが望ましい。ただし、公開鍵暗号での仕様上、利用すべきハッシュ関数が指定されている場合には、この限りではない。」という注釈がついている。

参考文献

[BC04a] E. Biham and R. Chen, “Near-collisions of SHA-0,” Advances in Cryptology — CRYPTO 2004, Lecture Notes in Computer Science Vol. 3152, Springer-Verlag, 2004, pp. 290–305. Updated version available at http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2004/CS/CS-2004-09.ps.gz

[BC04b] E. Biham and R. Chen, “New Results on SHA-0 and SHA-1,” Short Talk Presented at CRYPTO 2004 Rump Session, 2004.

[BCS04] J. Black, M. Cochran, and T. Shrimpton, “On the Impossibility of Highly Efficient Blockcipher-based Hash-functions,” Cryptology ePrint Archive, Report 2004/062, http://eprint.iacr.org/2004/062.

[BR00] P.S.L.M. Barreto and V. Rijmen, “The Whirlpool Hashing Function,” First Open NESSIE Workshop, Leuven, Belgium, 13–14, November 2000, was revised and revised version is available at https://www.cosic.esat.kuleuven.be/nessie/tweaks.html.

[C03a] 独立行政法人 情報処理推進機構, 通信・放送機構, ブロック暗号を使った秘匿、メッセージ認証、及び認証暗号を目的とした利用モードの技術調査報告, 2003. http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/documents/mode_wg040607_000.pdf, http://cryptrec.nict.go.jp/PDF/wat_rep040427/mode_wg040607.pdf

[C06] 独立行政法人 情報処理推進機構, 通信・放送機構, ハッシュ関数 Whirlpol, RIPEMD-160 の強度評価と SHA-1 の改良方法の技術調査報告 2006, (公開予定), http://www.cryptrec.jp/.

[CJ98] F. Chabaud and A. Joux, “Differential Collisions in SHA-0,” Advances in Cryptology — CRYPTO’98, Lecture Notes in Computer Science Vol.1462, Springer-Verlag, 1998, pp. 56–71.

[CMBRR] C.De Canniere, F. Mendel, N. Pramstaller, C.Rechberger, and V. Rijmen, “SHA Evaluation Report for CRYPTREC,” Technial Report IAIK – 2006/01/21, 2006.

[CRYPTREC03] 総務省, 経済産業省, 電子政府推奨暗号リスト, 平成 15 年 2 月 20 日. http://www.soumu.go.jp/joho_tsusin/security/pdf/cryptrec_01.pdf

[D89] I. B. Damgård, “A Design Principle for Hash Functions,” Advances in Cryptology — CRYPTO’89, Lecture Notes in Computer Science Vol. 435, Springer-Verlag, 1990, pp. 416–427.

[D96a] H. Dobbertin, “Cryptanalysis of MD4,” Fast Software Encryption — FSE’96, Lecture Notes in Computer Science Vol. 1039, Springer-Verlag, 1996, pp. 53–69.

[D96b] H. Dobbertin, “Cryptanalysis of MD5 Compress,” Rump Session Talk at EUROCRYPT’96, 1996.

[D96c] H. Dobbertin, “The Status of MD5 after a Recent Attack,” CryptoBytes Vol. 2, No. 2, 1996, pp. 1–6.

[D97] H. Dobbertin, “RIPEMD with Two-round Compress Function is Not Collision-free,” Journal of Cryptology Vol. 10, No. 1, 1997, pp. 51–70.

[D98] H. Dobbertin, “Cryptanalysis of MD4,” Journal of Cryptology Vol. 11, No. 4, 1998, pp. 253–271.

[DB92] B. den Boer and A. Bosselaers, “An Attack on the Last Two Rounds of MD4,” Advances in Cryptology — CRYPTO’91, Lecture Notes in Computer Science Vol. 576, Springer-Verlag, 1992, pp. 194–203.

[DB94] B. den Boer and A. Bosselaers, “Collisions for the Compression Function of MD5,” Advances in Cryptology — EUROCRYPT’93, Lecture Notes in Computer Science Vol. 773, Springer-Verlag, 1994, pp. 293–304.

[DBP96] H. Dobbertin, A. Bosselaers, and B. Preneel, “RIPEMD-160, A Strengthened Version of RIPEMD,” Fast Software Encryption — FSE’96, Lecture Notes in Computer Science Vol. 1039, Springer-Verlag, 1996, pp. 71–82.

[FIPS180] National Institute of Standards and Technology, Federal Information Processing Standards Publication 180, Secure Hash Standard, May 11, 1993.

[FIPS180-1] National Institute of Standards and Technology, Federal Information Processing Standards Publication 180-1, Secure Hash Standard, (supersedes FIPS 180) April 17, 1995.

[FIPS180-2] National Institute of Standards and Technology, Federal Information Processing Standards Publication 180-2, Secure Hash Standard, (supersedes FIPS 180-1) August 1, 2002.

[FIPS180-2a] National Institute of Standards and Technology, Federal Information Processing Standards Publication 180-2 with Change Notice to Include SHA-224, Secure Hash Standard, February 25, 2004.

[GH02a] H. Gilbert and H. Handschuh, "Evaluation Report, Security Level of Cryptography — SHA-256," 2002. Available at `http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1045_IPA-SHA256.pdf`.

[GH02b] H. Gilbert and H. Handschuh, "Evaluation Report, Security Level of Cryptography — SHA-384 and SHA-512," 2002. Available at `http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1046_SHA_384_512.pdf`.

[GH03] H. Handschuh and H. Gilbert, "Security Analysis of SHA-256 and Sisters," Selected Areas in Cryptography — SAC 2003, Lecture Notes in Computer Science Vol. 3006, Springer-Verlag, pp. 175–193, 2004.

[H04] S. Hirose, "Provably Secure Double-block-length Hash Functions in a Black-box Model," ICISC 2004 Pre-proceedings, pp. 485–497, 2004.

[HH05] S. Hirose and M. Hattori, "A Note on Security of Double-block-length Hash Functions," The 2005 Symposium on Cryptography and Information Security (SCIS 2005) Proceedings, pp. 559–564, 2005.

[HPR04] P. Hawkes, M. Paddon, and G. G. Rose, "On Corrective Patterns for the SHA-2 Family," Cryptology ePrint Archive, Report 2004/207, `http://eprint.iacr.org/2004/207`.

[ISO/IEC10118-1] ISO/IEC 10118-1: 2000, Hash-functions – Part 1: General (2nd edition).

[ISO/IEC10118-2] ISO/IEC 10118-2: 2000, Hash-functions – Part 2: Hash-functions Using an n-bit Block Cipher Algorithm (2nd edition).

[ISO/IEC10118-3] ISO/IEC 10118-3: 2004, Hash-functions – Part 3: Dedicated Hash-functions (3rd edition).

[ISO/IEC10118-4] ISO/IEC 10118-4: 1998, Hash-functions – Part 4: Hash-functions Using Modular Arithmetic.

[JCLJ04] A. Joux, P. Carribault, C. Lemuet, and W. Jalby, "Collision in SHA-0," Posted to `sci.crypt` NNTP News Group, August 12, 2004.

[K02a] L. R. Knudsen, "Non-random properties of reduced-round Whirlpool," public report `NES/DOC/UIB/WP5/016`, NESSIE, June 2002.

[K02b] L. R. Knudsen, "Quadratic relations in Khazad and Whirlpool," public report `NES/DOC/UIB/WP5/017`, NESSIE, June 2002.

[KPPRR05] N. Pramstaller, C. Rechberger and V. Rijmen, "Analysis of Simplified Variants of SHA-256," In Proceedings of WEWoRC 2005, pp. 123–134, 2005.

[LWW05] A. Lenstra, X. Wang and B. de Weger, "Colliding X.509 Certificates," Cryptology ePrint Archive, Report 2005/067, Available at `http://eprint.iacr.org/2005/067`, March 1, 2005.

[M89] R. Merkle, "One Way Hash Functions and DES", Advances in Cryptology — CRYPTO'89, Lecture Notes in Computer Science Vol. 435, Springer-Verlag, pp. 428–446, 1989.

[MOV97] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

[NESSIE] NESSIE, New European Schemes for Signatures, Integrity, and Encryption, `https://www.cosic.esat.kuleuven.be/nessie/`.

[NESSIE03] NESSIE, "Performance of Optimized Implementations of the NESSIE Primitives, version 2.0," February 20, 2003. Available at `https://www.cosic.esat.kuleuven.ac.be/nessie/deliverables/D21-v2.pdf`.

[NM03] J. Nakajima and M. Matsui, "Performance Analysis and Parallel Implementation of Dedicated Hash Functions on Pentium III," IEICE Transactions on Fundamentals Vol. E86-A, No. 1, January, 2003, pp. 54–63.

[PRR05] N. Pramstaller, C. Rechberger and V. Rijmen, "Preliminary Analysis of the SHA-256 Message Expansion," NIST First Cryptographic Hash Workshop, 2005. http://www.csrc.nist.gov/pki/HashWorkshop/2005/Oct31_Presentations/Rechberger_PreliminaryAnalysisOfSHA256.pdf.

[R90] R. L. Rivest, "The MD4 Message Digest Algorithm," Advances in Cryptology — Crypto'90, Lecture Notes in Computer Science Vol. 537, Springer-Verlag, 1991, pp. 303–311.

[R92a] R. L. Rivest, "The MD4 Message-digest Algorithm," Request for comments (RFC) 1320, IETF, 1992.

[R92b] R. L. Rivest, "The MD5 Message-digest Algorithm," Request for comments (RFC) 1321, IETF, 1992.

[RIPE92] Research and Development in Advanced Communication Technologies in Europe, "RIPE Integrity Primitives: Final Report of RACE Integriy Primitives Evaluation (R1040)," RACE, June 1992.

[RIPE95] RIPE Consortium, RIPE Integrity Primitives — Final Report of RACE Integriy Primitives Evaluation (R1040), Lecture Notes in Computer Science Vol. 1007, Springer-Verlag, 1995.

[RO04] V. Rijmen and E. Oswald, "Update on SHA-1," Topics in Cryptology — CT-RSA 2005, Lecture Notes in Computer Science Vol. 3376, Springer-Verlag, 2005, pp. 58–71. Updated version is available at http://eprint.iacr.org/2005/010, January 14, 2005.

[SS03] T. Shirai and K. Shibutani, "On the Diffusion Matrix Employed in the Whirlpool Hashing Function," March 11, 2003. Available at https://www.cosic.esat.kuleuven.ac.be/nessie/reports/phase2/whirlpool-20030311.pdf.

[V95] S. Vaudenay, "On the Need for Multipermutations: Cryptanalysis of MD4 and SAFER," Fast Software Encryption — FSE'95, Lecture Notes in Computer Science Vol. 1008, Springer-Verlag, 1995, pp. 286–297.

[WFLY04] X. Wang, D. Feng, X. Lai, and H. Yu, "Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD," Cryptology ePrint Archive, Report 2004/199, http://eprint.iacr.org/2004/199, August 16 (revised August 17), 2005.

[WLFCY04] X. Wang, X. Lai, D. Feng, H. Chen, and H. Yu, "Cryptanalysis of the Hash Functions MD4 and RIPEMD," to be appeared in Advances in Cryptology, — EUROCRYPT 2005.

[WY04] X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions," to be appeared in Advances in Cryptology, — EUROCRYPT 2005.

[WYY05] X. Wang, Y. Yin, and H. Yu, "Collision Search Attacks on SHA1," February 13, 2005. Available at `http://www.infosec.sdu.edu.cn/sha-1/shanote.pdf`.

[WYY05a] X. Wang, Y.L. Yin, and H. Yu, "Finding Collisions in the Full SHA-1," *Advances in Cryptology — CRYPTO 2005, 25th Annual International Cryptology Conference*, Lecture Notes in Computer Science vol. 3621, pp. 17–36, Springer-Verlag, 2005.

[WYY05b] X. Wang, A.C. Yao, and F. Yao, "Cryptanalysis on SHA-1," Keynote Speech of *Cryptographic Hash Workshop*, `http://www.csrc.nist.gov/pki/HashWorkshop/2005/Oct31_Presentations/Wang_SHA1-New-Result.pdf`.

[YB05] H. Yoshida and A. Biryukov, "Analysis of a SHA-256 Variant," 12th Annual Workshop on Selected Areas in Cryptography (SAC 2005), 2005.

[ZPS92] Y. Zheng, J. Pieprzyk, and J. Seberry, "HAVAL — A One-way Hashing Algorithm with Variable Length of Output," Advances in Cryptology — Auscrypt'92, Lecture Notes in Computer Science Vol. 718, Springer-Verlag, 1993, pp. 83–104.

付録５

暗号利用モード・MACに関する技術調査報告

平成17年12月 (平成18年 5月改訂)

ハッシュ関数・暗号利用モード調査WG

ハッシュ関数・暗号利用モード調査 WG 委員構成

主査：古原 和邦　東京大学生産技術研究所

委員：廣瀬 勝一　福井大学工学部

委員：川村 信一　株式会社東芝

委員：古屋 聡一　株式会社日立製作所

委員：盛合 志帆　株式会社ソニー・コンピュータエンタテインメント

目 次

1　はじめに

ブロック暗号は，暗号学的プリミティブの一つであり暗号化関数と復号関数により構成される．暗号化関数は，固定長の入力と鍵を受け入れ，入力を入力と同じ長さの出力に攪拌する．また，暗号化の際に利用した鍵と復号関数を用いることで出力を元の入力に戻すことができる．一般に，ブロック暗号は，入出力ペアから鍵情報を推定すること，および，鍵を知らずに出力から入力（あるいは入力から出力）を推測することが困難となるように設計されている．

しかしながら，ブロック暗号は固定長の入力しか変換できないため，長い平文を処理する際には，その利用方法を工夫する必要がある．その利用方法がブロック暗号利用モードであり，ブロック暗号利用モードには，秘匿性のみを提供するモードやメッセージ認証機能を提供するモード，更にはその両方を提供するモードなどがある．

本報告書では，各種標準化作業や学術出版物などで知られている暗号利用モードを調査し，その安全性，処理効率，その他工業的，学術的性質などについてまとめる．ただし，改ざん検出機能に関しては，ブロック暗号以外にハッシュ関数を用いるものも調査の対象としている．

1.1　各用語の簡単な説明

ブロック暗号は固定長，n ビットの平文を暗号化する．多くのブロック暗号では $n = 64$ や，$n = 128$ である．ブロック暗号は，ブロック暗号利用モードのプリミティブとして用いられる．ブロック暗号利用モードの主な機能はメッセージの**秘匿**と**認証**である．本報告書では，秘匿のためのブロック暗号利用モードを**暗号化方式**, 認証のためのブロック暗号利用モードを**メッセージ認証コード** (Message Authentication Code, MAC) という．また，これらの機能を併せもつブロック暗号利用モードを**メッセージ認証つき暗号化方式**という．

- 暗号化方式はブロック暗号を用いて，n ビットより長いメッセージを暗号化する．送信者と受信者が秘密鍵 K を共有しており，送信者は暗号化アルゴリズム $\mathcal{E}$ を用いて，平文 M と鍵 K から暗号文 $C = \mathcal{E}_K(M)$ を計算し，C を受信者に送る．M の長さは n ビットよりも長くてよい．受信者は復号アルゴリズム $\mathcal{D}$ を用いて，暗号文 C と鍵 K から平文 $M = \mathcal{D}_K(C)$ を計算する．

  例として，ECB, CBC, OFB, CFB, CTR などがある．図 1 参照.

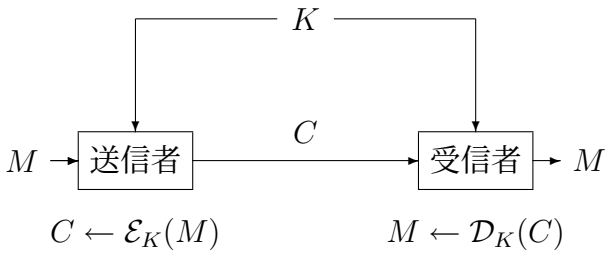


図 1: 暗号化方式のモデル

- メッセージ認証コードはブロック暗号を用いて，メッセージが偽造，改ざんされることを防ぐ技術である．送信者と受信者が秘密鍵 K を共有しており，送信者はタグ生成アルゴリズム $\mathcal{G}$ を用いて，平文 M と鍵 K からタグ $T = \mathcal{G}_K(M)$ を計算し，メッセージ，タグのペア (M, T) を受信者に送る．M の長さは，n ビットよりも長くてよい．T は固定長であり，32, 64, 96, 128 ビット程度の長さが一般的である．(M, T) を受け取った受信者は確認アルゴリズム $\mathcal{V}$ を用いて，受理信号，もしくは改ざん検出信号を出力する．$\mathcal{V}$ は，受け取ったメッセージに対し，$T^* = \mathcal{G}_K(M)$ を計算し，$T = T^*$ なら受理信号を，そうでなければ改ざん検出信号を出力する．

  例として，CBC MAC, EMAC, OMAC, PMAC などがある．図 2 参照.

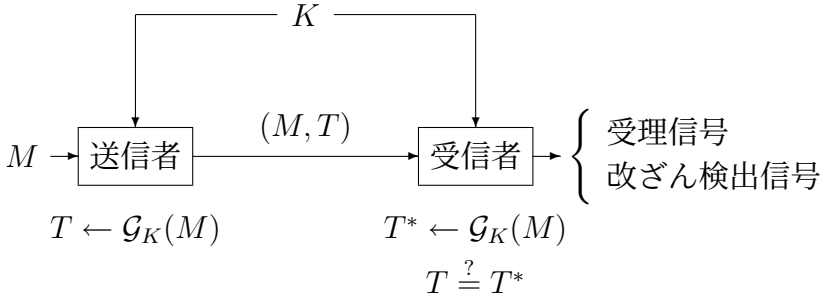


図 2: メッセージ認証コードのモデル

- メッセージ認証つき暗号化方式は，暗号化方式とメッセージ認証コードの機能を併せもつ．送信者と受信者が秘密鍵 K を共有しており，送信者は暗号化アルゴリズム $\mathcal{E}$ を用いて，平文 M と鍵 K から暗号文 $C = \mathcal{E}_K(M)$ を計算し，C を受信者に送る．受信者は復号ア

ルゴリズム $\mathcal{D}$ を用いて，暗号文 C と鍵 K から平文 $M = \mathcal{D}_K(C)$ を，もしくは改ざん検出信号を出力する．

例として，CCM, IAPM, OCB などがある．また，任意の暗号化方式とメッセージ認証コードを組み合わてメッセージ認証つき暗号化方式を構成する方法が知られている．図 3 参照．

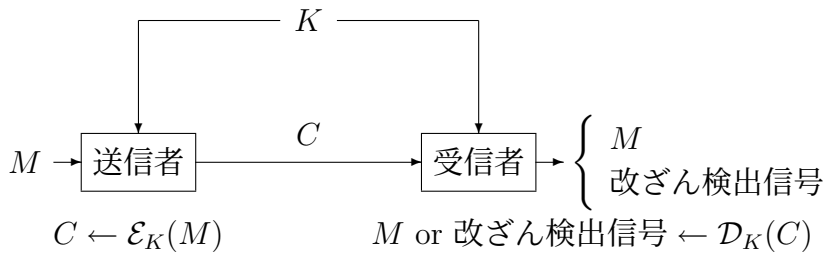


図 3: メッセージ認証つき暗号化方式のモデル

ブロック暗号利用モードの安全性　Bellare, Kilian, Rogaway により，CBC MAC の安全性が数学的に示された [BKR00]．ブロック暗号が安全な擬似ランダム置換族であれば，CBC MAC は偽造不可能性の意味で安全であることを示している．以降，多くのブロック暗号利用モードの安全性は，この種の証明可能安全性を根拠にしている．以下，暗号化方式，メッセージ認証コード，メッセージ認証つき暗号化方式，それぞれについて，安全性の定義を概説する．

- 暗号化方式に対しては，いくつかの安全性定義が存在する [BDJR97] が，ランダムビット列からの識別不能性が一般的である．暗号文 C か，もしくは C と同じ長さのランダムビット列 R が与えられ，有意な確率でこの 2 つを見分けることができないとき，暗号化方式は安全である，という．

- メッセージ認証コードに対しては，偽造不可能性が一般的である．(鍵 K を知らずに) $T = \mathcal{G}_K(M)$ となる (M, T) を出力できないとき，メッセージ認証コードは安全である，という．

- メッセージ認証つき暗号化方式では，暗号化方式の安全性定義とメッセージ認証コードの安全性定義の両方を考える．

  – 暗号文 C か，もしくは C と同じ長さのランダムビット列 R が与えられ，有意な確率でこの 2 つを見分けることができない．

– (鍵 K を知らずに) 改ざん検出信号 $\neq DEC_K(C)$ となる C を出力できない.

上記二つが成り立つとき，メッセージ認証つき暗号化方式は安全である，という.

ブロック暗号利用モードの効率　本報告書は，安全性を主眼においているが，効率についても述べる．主に以下の点について述べる.

- 鍵長：安全性が変わらないのであれば，短いほうがよい.
- ブロック暗号鍵スケジューリングの呼び出し回数：一般的にブロック暗号鍵スケジューリングは計算時間がかかり，安全性が変わらないのであれば，少ないほうがよい.
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：安全性が変わらないのであれば，少ないほうがよい.
- 事前計算するべきブロック暗号の呼び出し回数：これらは，メッセージ M によらず実行できる．安全性が変わらないのであれば，少ないほうがよい.
- 並列処理性：ブロック暗号の並列処理が可能であれば，ハードウェア上で高速に実装できる.

2　記号や用語の厳密な定義

本章では本報告書で扱う記法をまとめるとともに，ブロック暗号，およびその利用モードの一般的な定義と，安全性の定義を述べる.

いくつかの標準化などでは，特定の利用モードを 64 ビットブロック暗号への適用のみに限定した記載などをしている場合がある．しかし，本稿で扱う利用モードすべては，処理単位長に特化した利用モードであることはない．よって，その暗号学的な本質を議論することを目的として，汎用的なブロック暗号に対するモードとして議論を進める．具体的には，内部で用いるブロック暗号のブロック長を n ビットとする.

本稿では，排他的論理和演算を多用する．本稿ではそのサイズは文脈から明らかであり単に “$\oplus$” で示す.

あるシステムから固定長文字列を逐次的に生成する場合について，その文字列生成システム (または，生成する文字列) の性質として “nonce” (ナ

ンス，と読む) を説明する．これは，生成時点より前には生成されたことがないような値を出力するものである．その例としてカウンタや時刻情報などがあるが，これらは無限にこの性質をもつものではない．そういう意味では乱数発生系列も確率的ではあるが，多くの場合において，“nonce”の性質を持っているといえる．

主に安全性の議論で参照される技術用語に，ランダム関数，擬似ランダム関数，及び擬似ランダム置換がある．これらの正確な定義は専門書，及び技術論文に譲るとしてここでは簡単にその説明をする．

ランダム関数とは，与えた入力に対してその出力が決定されるものの，その出力は，どんな情報からも推測できないランダムな値であるような関数のモデルである．このような関数は現実に存在するかどうかは別にして，そのような関数の振舞いをブロック暗号の性質に見立てて，利用モードの安全性を議論することがある．

しかし，ランダム関数は入力に対する出力がどのような方法を用いても推測できない，という性質は，現在のブロック暗号にそれを求めるのは無理がある．ブロック暗号には鍵入力がありこの鍵が求まってしまえば，どの入力がどの出力を出すかがわかってしまう．そこで，ある程度の時間，計算量をかけた上で破れるかもしれないようなランダム関数のモデルを擬似ランダム関数モデルという．これを証明などで扱う場合，パラメータがつく．

また，ブロック暗号は入力，出力の間に単射という性質がある．これ自身もランダム関数にはない特殊な性質であるので，実際のブロック暗号は，単なる擬似ランダム関数ではなく，さらに弱い擬似ランダム置換というモデルまで落して考えることが多い．ここで扱うモデルを擬似ランダム置換という．

ディスクセクタ暗号の部分などで，universal hash(汎用ハッシュ) と呼ばれる関数を考えることがある．これは，ある種の関数であって，ざっくり説明すると任意長の入力とパラメータから固定長の出力を出す関数であって，任意に固定した二つの入力が衝突する場合というのが，パラメータすべてのうちごくわずかであることが，どのような二入力メッセージについても言えるような関数である．ただし，パラメータを知っている攻撃者のようなものがいれば，衝突を作ることは必ずしも難しくない．

その他の用語，記号については以下のとおりに定義する:

ρ	ランダム関数モデル
π	擬似ランダム関数
$a \ll k$	レジスタ値 a を左に k ビットシフトする演算
$\mathrm{msb}_k(a)$	レジスタ値 a の上位 k ビットの値
$Enc_K(\cdot)$	あるブロック暗号プリミティブの暗号化処理 (鍵 K)
$Dec_K(\cdot)$	あるブロック暗号プリミティブの復号処理 (鍵 K)

2.1 記法

A が集合である場合，$a \xleftarrow{R} A$ は A から a を一様ランダムに選ぶことをあらわす．A がアルゴリズムである場合，$a \leftarrow A$ は A の実行結果を a とする，ということをあらわす．A が確率的アルゴリズムでれば，$a \xleftarrow{R} A$ と表記する．整数 l に対し，$\{0,1\}^l$ はすべての l ビット列の集合をあらわす．また，$\{0,1\}^{\leq l}$ は，l ビット以下のすべてのビット列の集合をあらわす．長さ 0 のビット列 ε もこれに含める．同様に，$(\{0,1\}^l)^+$ は，長さが l の整数倍のすべてのビット列からなる集合をあらわす．すなわち，$(\{0,1\}^l)^+ = \bigcup_{l'=1,2,\ldots} \{0,1\}^{ll'}$ である．同様に，$\{0,1\}^*$ は，すべてのビット列の集合を表す．すなわち，$\{0,1\}^* = \bigcup_{l=0,1,2,\ldots} \{0,1\}^l$ である．a と b が同じ長さのビット列であれば，$a \oplus b$ はそれらのビットごとの排他的論理和をあらわす．a がビット列のとき，$|a|$ は a のビット長をあらわす．

2.2 ブロック暗号

ブロック暗号 (block cipher) E とは，$E : \mathcal{K}_E \times \mathcal{M}_E \to \mathcal{M}_E$ なる関数である．$\mathcal{K}_E$ は，鍵空間とよばれ，$\mathcal{K}_E = \{0,1\}^k$ のとき，k を鍵長という．$\mathcal{M}_E$ は，メッセージ空間，もしくは平文空間とよばれ，$\mathcal{M}_E = \{0,1\}^n$ のとき，n をブロック長という．ただし，すべての $K \in \mathcal{K}_E$ に対し，$E(K,\cdot)$ は $\mathcal{M}_E$ 上の置換でなくてはならない．$E(K,\cdot)$ は $\mathcal{M}_E$ 上の置換なので，その逆関数 $E^{-1}(K,\cdot)$ が存在する．すべての鍵 $K \in \mathcal{K}_E$ とすべての平文 $X \in \mathcal{M}_E$ に対し，$E^{-1}(K; E(K,X)) = X$ であり，すべての鍵 $K \in \mathcal{K}_E$ とすべての暗号文 $Y \in \mathcal{M}_E$ に対し，$E(K; E^{-1}(K,Y)) = Y$ である．関数 $E(K,\cdot)$ を暗号化関数，関数 $E^{-1}(K,\cdot)$ を復号関数という．それぞれ $E_K(\cdot)$, $E_K^{-1}(\cdot)$ と表記する．

2.3 暗号化方式

暗号化方式はメッセージ秘匿のためのブロック暗号利用モードである．暗号化方式 *ENC* は，三つのアルゴリズム $\mathit{ENC} = (\mathit{ENC}\text{-}\mathcal{K}, \mathit{ENC}\text{-}\mathcal{E}, \mathit{ENC}\text{-}\mathcal{D})$ から成る．$\mathit{ENC}\text{-}\mathcal{K}$ を鍵生成アルゴリズム，$\mathit{ENC}\text{-}\mathcal{E}$ を暗号化アルゴリズム，$\mathit{ENC}\text{-}\mathcal{D}$ を復号アルゴリズムという．また，メッセージ空間 $\mathcal{M}_{\mathit{ENC}}$ をもつ．

ここで，鍵生成アルゴリズム $\mathit{ENC}\text{-}\mathcal{K}$ は確率的アルゴリズムであり，入力はなく，ランダムな鍵 K を出力する．$K \xleftarrow{R} \mathit{ENC}\text{-}\mathcal{K}$ と表記する．暗号化アルゴリズム $\mathit{ENC}\text{-}\mathcal{E}$ は，確率的，決定的，状態をもつ，もしくは nonce を利用するアルゴリズムである．鍵 K とメッセージ $M \in \mathcal{M}_{\mathit{ENC}}$ を入力とし，暗号文 C を出力する．$C \xleftarrow{R} \mathit{ENC}\text{-}\mathcal{E}(K; M)$ や $C \leftarrow \mathit{ENC}\text{-}\mathcal{E}(K; M)$ と表記する．また，乱数や状態を明示的に入力に示すこともある．

暗号化アルゴリズムが確率的アルゴリズムである場合，入力 (K, M) が与えられるたびに乱数を選び，それを用いて暗号文 C を計算する．アルゴリズムが呼び出されるたびに乱数を選びなおす．同じ入力で 2 度アルゴリズムを呼び出したとしても，同じ出力になるとは限らない．

暗号化アルゴリズムが状態をもつアルゴリズムである場合，まずある定められた方法に従って状態を初期化する．入力 (K, M) が与えられると，(K, M) と現在の状態に応じて暗号文 C を計算し，状態を更新し，新しい状態を保持する．多くの場合，状態は単なるカウンタである．

暗号化アルゴリズムが nonce を用いるアルゴリズムである場合，メッセージごとに異なる値である nonce を用いる．メッセージを数えるカウンタは，メッセージごとに異なる値であるので，nonce として用いることができる．ただし，nonce はカウンタのように値が増える (あるいは減る) 必要はなく，単に異なるメッセージに対しては，異なる値であればよい．

復号アルゴリズム $\mathit{ENC}\text{-}\mathcal{D}$ は，決定的アルゴリズムであり，鍵 K と暗号文 C を入力とし，メッセージ M を出力する．$M \leftarrow \mathit{ENC}\text{-}\mathcal{D}(K; C)$ と表記する．

全ての鍵 K と全てのメッセージ M に対し，

$$\mathit{ENC}\text{-}\mathcal{D}(K; \mathit{ENC}\text{-}\mathcal{D}(K; M)) = M$$

でなければならない．

$\mathit{ENC}\text{-}\mathcal{E}(K; \cdot)$ と $\mathit{ENC}\text{-}\mathcal{D}(K; \cdot)$ を，$\mathit{ENC}\text{-}\mathcal{E}_K(\cdot)$ や $\mathit{ENC}\text{-}\mathcal{D}_K(\cdot)$ と表記する．

例として，一般的な ECB, CBC, OFB, CFB, CTR [SP800-38A], ディスク暗号化用の NR [NR99], CMC [HR03b], 3GPP の $f8$ [3GPPa, 3GPPb] などがある．

2.4 メッセージ認証コード

メッセージ認証コード MAC は，メッセージ認証のためのブロック暗号利用モードである．メッセージ認証コード MAC は，三つのアルゴリズム $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ から成る．$MAC\text{-}\mathcal{K}$ を鍵生成アルゴリズム，$MAC\text{-}\mathcal{G}$ をタグ生成アルゴリズム，$MAC\text{-}\mathcal{V}$ を確認アルゴリズムという．また，メッセージ空間 $\mathcal{M}_{MAC}$ とタグ空間 $\mathcal{T}_{MAC}$ をもつ．

鍵生成アルゴリズム $MAC\text{-}\mathcal{K}$ は確率的アルゴリズムであり，入力はなく，鍵 K を出力する．$K \xleftarrow{R} MAC\text{-}\mathcal{K}$ と表記する．

タグ生成アルゴリズム $MAC\text{-}\mathcal{G}$ は，確率的，決定的，もしくは状態をもつアルゴリズムである．鍵 K とメッセージ $M \in \mathcal{M}_{MAC}$ を入力とし，タグ $T \in \mathcal{T}_{MAC}$ を出力する．$T \xleftarrow{R} MAC\text{-}\mathcal{G}(K; M)$ や $T \leftarrow MAC\text{-}\mathcal{G}(K; M)$ と表記する．また，乱数や状態を明示的に入力に示すこともある．

$MAC\text{-}\mathcal{G}$ が確率的アルゴリズムである場合，入力 (K, M) が与えられるたびに乱数を選び，それを用いてタグ T を計算する．アルゴリズムが呼び出されるたびに乱数を選びなおす．同じ入力で 2 度アルゴリズムを呼び出したとしても，同じ出力になるとは限らない．

$MAC\text{-}\mathcal{G}$ が状態をもつアルゴリズムである場合，まずある定められた方法に従って状態を初期化する．入力 (K, M) が与えられると，(K, M) と現在の状態に応じてタグ T を計算し，状態を更新し，新しい状態を保持する．多くの場合，状態は単なるカウンタである．

確認アルゴリズム $MAC\text{-}\mathcal{V}$ は，決定的アルゴリズムであり，鍵 K, メッセージ $M \in \mathcal{M}_{MAC}$, タグ $T \in \mathcal{T}_{MAC}$ を入力とし，accept or reject を出力する．$MAC\text{-}\mathcal{V}(K; M; T) = \text{accept}$ や，$MAC\text{-}\mathcal{V}(K; M; T) = \text{reject}$ と表記する．

全ての鍵 K と全てのメッセージ M に対し，

$$MAC\text{-}\mathcal{V}(K; M; MAC\text{-}\mathcal{G}(K; M)) = \text{accept}$$

でなければならない．

$MAC\text{-}\mathcal{G}(K; \cdot)$ と $MAC\text{-}\mathcal{V}(K; \cdot; \cdot)$ を，$MAC\text{-}\mathcal{G}_K(\cdot)$ や $MAC\text{-}\mathcal{V}_K(\cdot, \cdot)$ と表記する．

例として，CBC MAC [BKR00], EMAC [BB+95, PR00], RMAC [JJ+02a, JJ+02b, JJ+02c], XCBC [BR00], TMAC [KI03], OMAC [IK03a], XOR MAC [BGR95], XECB MAC [GD01a], PMAC [BR02], $f9$ [3GPPa, 3GPPb] などがある．

2.5 メッセージ認証つき暗号化方式

メッセージ認証つき暗号化方式はメッセージ秘匿とメッセージ認証の機能を併せ持つブロック暗号利用モードである．メッセージ認証つき暗号化方式 AE は，三つのアルゴリズム $AE = (AE\text{-}\mathcal{K}, AE\text{-}\mathcal{E}, AE\text{-}\mathcal{D})$ から成る．$AE\text{-}\mathcal{K}$ を鍵生成アルゴリズム，$AE\text{-}\mathcal{E}$ を暗号化アルゴリズム，$AE\text{-}\mathcal{D}$ を復号アルゴリズムという．また，メッセージ空間 $\mathcal{M}_{AE}$ をもつ．

ここで，鍵生成アルゴリズム $AE\text{-}\mathcal{K}$ は確率的アルゴリズムであり，入力はなく，ランダムな鍵 K を出力する．$K \xleftarrow{R} AE\text{-}\mathcal{K}$ と表記する．暗号化アルゴリズム $AE\text{-}\mathcal{E}$ は，確率的，決定的，状態をもつ，もしくは nonce を利用するアルゴリズムである．鍵 K とメッセージ $M \in \mathcal{M}_{AE}$ を入力とし，暗号文 C を出力する．$C \xleftarrow{R} AE\text{-}\mathcal{E}(K; M)$ や $C \leftarrow AE\text{-}\mathcal{E}(K; M)$ と表記する．また，乱数や状態を明示的に入力に示すこともある．

復号アルゴリズム $AE\text{-}\mathcal{D}$ は，決定的アルゴリズムであり，鍵 K と暗号文 C を入力とし，メッセージ M, もしくは改ざん検出信号を出力する．$M \leftarrow AE\text{-}\mathcal{D}(K; C)$ や，改ざん検出信号 $\leftarrow AE\text{-}\mathcal{D}(K; C)$ と表記する．

全ての鍵 K と全てのメッセージ M に対し，

$$AE\text{-}\mathcal{D}(K; AE\text{-}\mathcal{D}(K; M)) = M$$

でなければならない．

$AE\text{-}\mathcal{E}(K; \cdot)$ と $AE\text{-}\mathcal{D}(K; \cdot)$ を，$AE\text{-}\mathcal{E}_K(\cdot)$ や $AE\text{-}\mathcal{D}_K(\cdot)$ と表記する．

例として，IACBC, IAPM [J01], OCB [RBBK01a], XCBC [GD01a], CCM [WHF02, J02], CWC [KVW03], EAX [BRW03] などがある．

3 各々の利用モードを定義する文書

利用モードは各種標準化や，学術文書において定義されることが多い．ここでは，利用モードを定義する文書についての紹介を行う．

3.1 商務省連邦情報処理規格 (FIPS), 特殊文書 (SP)

米国では政府などで用いる暗号技術の方式を FIPS(Federal Information Processing Standard, 商務省連邦情報処理規格) で定めている [WWW4]. FIPS は NIST (National Institute of Standards and Technology, 商務省技術標準局)[WWW5] で編集が行なわれ，管理されている．

3.1.1 ブロック暗号プリミティブ

本報告の主要な対象技術はブロック暗号利用モードである．ここではブロック暗号プリミティブの代表例としてFIPSで記載のブロック暗号のうちDES，AESについて，仕様の概要を紹介する．本報告で扱う利用モードの適用対象はこれらに限定される訳ではない.

Data Encryption Standard (DES) ブロック暗号に関するNISTの標準としては，DES(Data Encryption Standard, データ暗号化規格) がFIPS46(1977年1月15日) で定義されており，現在その改訂などによりFIPS46-3 (2003年11月時点) が公開されている [FIPS46-3]．FIPS46-3では，DESのブロック暗号としての強度を高める目的でTDEA (Triple Data Encryption Algorithm, 三連DES) が定義されており，三つ鍵版 (K_1, K_2, K_3) の定義をもとに，鍵利用オプションとして二個鍵版 $(K_1 = K_3)$ やDESコンパチブル版 $(K_1 = K_2 = K_3)$ が定義されている．これが通称トリプルDES(T-DES, 3DES) である．

DESは鍵長64ビットであるが，そのうちパリティビット8ビットは暗号学的強度に寄与しないため，実質56ビットである．ブロックサイズは64ビットである．TDEAはブロックサイズは変わらず，実質鍵長が，三つ鍵版168ビット，二つ鍵版112ビットである．

Advanced Encryption Standard (AES) ―SやTDEAの安全性への懸念を受けて，NISTは1997年からの標準化活動の結果として，2001年11月26日，AES(Advanced Encryption Standard, 次世代暗号標準) をFIPS197として定義した [FIPS197].

AESはブロック長128ビットで，鍵長は128ビット，192ビット，256ビットの三つの鍵長の処理 (AES-128, AES-192, AES-256) が定義されている．

3.1.2 ブロック暗号利用モード

歴史的に標準を紹介すると，NISTはDESをFIPS掲載してから間もなく，DESの利用方法を定める **DES利用モード**をFIPS81で定義した (1980年12月2日) [FIPS81]．また，仕様書の誤植の変更として1981年11月20日にChange Noticeが発行された．FIPS81では，ECB, CBC, k-CFB, k-OFB の4つのモードが定義されている．ただし，この文書に対するChange Notice 2(1996年5月31日) において，k-OFBに関しては $k < 64$ では使うべきでなくこれを以降サポートしないことが記載された.

Change Notice 3 は 64-bit OFB のテストベクトルのみ記載されている．同様に FIPS113 では，DES を使ったメッセージ認証符号の生成方法として CBC-MAC を定義している [FIPS113].

NIST は次に AES のための利用モードを定義するが，ここでは FIPS ではなく Special Publication としての発行が準備されている [WWW6]. 2003 年 11 月時点では，5 つの秘匿に関する利用モードが SP800-38A(2001 年 12 月版) として定義されている (2001 年 12 月版)[SP800-38A]．これには，FIPS81 で定義した 4 つのモードに加えて，CTR モードが追加されている．また OFB モードは安全性の観点からパラメータ k はブロックサイズのみとし，末端処理の定義を付け加えている．これら方式は，FIPS 認定の任意のブロック暗号アルゴリズムに適用できると記載されている．

またブロック暗号からメッセージ認証符号を生成するためのモードについては，2005 年 11 月現在，SP800-38B(2005 年 5 月版) として CMAC を定義している [SP800-38B]．そして，もうひとつ，認証暗号 (すなわち，復号時，暗号文の改竄を検出できる暗号処理) の標準は 2005 年 11 月現在，SP800-38C として CCM を定義している [SP800-38C].

SP800-38A　暗号化方式 (ECB, CBC, CFB, OFB, CTR)
SP800-38B　メッセージ認証コード (CMAC)
SP800-38C　メッセージ認証つき暗号化方式 (CCM)

また，DES を定義する FIPS46-3 でも ANSI X9.52 で定義される 7 つの利用モードの利用を認めている．7 つとは，すべて TDEA 用であって 4 つは ECB, CBC, CFB, OFB であり，残りは ANSI X9.52 版 CBC, CFB, OFB モードである (これらはインターリービング，すなわちパイプライン処理系にも適用できるような仕様変更がなされている).

3.2 ISO/IEC

ISO(International Organization for Standardization, 国際標準化機構), 及び IEC(International Electrotechnical Commission, 国際電気標準会議) は一部の国際規格を共同で策定している．特に暗号技術に関する分野では，ISO/IEC JTC 1/SC27 などで標準化会議が開かれ，暗号や情報セキュリティに関する ISO/IEC 標準文書が作成されている．

利用モードに関する標準化文書としては，IS 8372(64 ビットブロック暗号利用モード)[ISO8372]，ISO 10116[ISO10116] (n ビットブロック暗号利用モード, 2002 年 6 月 26 日) がある．IS 8372 の記述は，ISO 10116 に統合されることから，近い将来 IS が抹消されることになる．現在文書中

には，ECB, CBC, CFB, OFB の 4 つのモードが定義されているが，次の改訂作業で CTR モードが新たに加わる方向で議論が進んでいる．

また，ISO の金融取引に関する標準化 TC68 では，8731-1 で CBC-MAC を定義している．

ISO 8372　64 ビットブロック暗号利用モードである．4 つの暗号化方式 ECB, CBC, CFB, OFB を定めている．DES に関する FIPS 81 と ANSI X3.106 を一般化し，任意の 64 ビットブロック暗号を対象としたものになっている．

ISO 9797　メッセージ認証コードである．CBC MAC を定めている．同様の標準として，ISO 8731-1, ISO 9807, ANSI X9.9, ANSI X9.19 がある．

ISO 10116　ISO 8372 を n ビットブロック暗号について定めたものである．

ISO 8631-1　ISO の TC68 では金融サービスのためのセキュリティ標準を定めている．以下の標準を定めている．

ISO 8731-1　メッセージ認証コード, CBC MAC
ISO 10126　メッセージ暗号化

3.3　JIS

JIS(日本工業規格) は，JISC(Japanese Industrial Standard Committee, 日本工業標準調査会) が制定・改正を行なう日本の工業標準となる国家規格である．具体的には，JISC での審議のあと，主務大臣により制定され，JSA(Japanese Standards Association, 日本規格協会) から発行される．

JIS での利用モードに関する規格として，JIS X 5052, JIS X 5003 がある．前者は ISO 8372 ならびに ANSI X3.106 (American National Standard for Information Systems – Data Encryption Algorithm – Modes of Operation) と同一であり，JIS X 5053 は ISO/IEC 10116 と同一である．

3.4　金融に関するセキュリティ標準

ANSI X3　ANSI (American National Standards Institute) では，以下の 2 つの標準を定めている．

表 1: 秘匿のための利用モード一覧

略号	名前	日本語
2DEM	2D-Encryption Mode	二次元暗号
ABC	Accumulated Block Chaining	累積ブロック連鎖
CTR	Counter Mode Encryption	カウンタ
IGE	Infinite Garble Extention	無限改竄拡張

ANSI X3.92　FIPS 46 で定められている DES
ANSI X3.106　FIPS 81 で定められている DES の利用モード

ANSI X9　ANSI X9 シリーズでは，以下の標準を定めている．

ANSI X9.9　メッセージ認証コード, CBC MAC
ANSI X9.19　メッセージ認証コード, CBC MAC
ANSI X9.23　メッセージ暗号化
ANSI X9.52　Triple DES と利用モード

3.5　ANSI

ANSI(American National Standards Institute, アメリカ規格協会)[WWW7] では，主に ANSI X3.106, X3.92 で共通鍵暗号技術を標準化している．具体的には，ANSI X3.92 は DES を定義し，X3.106 でその利用モードを定義する．

3.6　AES 利用モード候補方式

NIST が AES の利用モードを策定する活動でも，いくつかの利用モードが提案された．2003 年 11 月時点での公開されている提案利用モードは表 1, 2, 3, 4 のとおり．

3.7　IEEE ディスクセクター暗号

IEEE(the Institute of Electrical and Electronics Engineers, Inc., 電気電子学会) の Security in Storage WG[WWW1] では，セクターレベルの記憶装置における機密情報を守る構想を定義し，暗号アルゴリズムや利用モードを定義している．2003 年 11 月まで，5 回の会合 (2002 年 6 月

表 2: AES に提案された MAC 生成のための利用モード

略号	名前	日本語
OMAC	OMAC: One-Key CBC	一個鍵 CBC
PMAC	Parallelizable M-.A-.Code	並列 MAC
RMAC	Randomized MAC	撹拌 MAC
TMAC	Two-Key CBC-MAC	二個鍵 CBC-MAC
XCBC	Extended Cipher Block Chaining MAC	拡張 CBC-MAC
XECB	eXtended Electronic Code Book MAC	拡張 ECB-MAC

表 3: AES に提案された認証暗号のための利用モード

略号	名前
CCM	Counter with CBC-MAC
CWC	Carter Wegman with Counter
CS	Cipher-State
EAX	A Conventional Authenticated-Encryption Mode
GCM	Galois/Counter Mode
IACBC	Integrity Aware Cipher Block Chaining
IAPM	Integrity Aware Parallelizable Mode
OCB	Offset Codebook
PCFB	Propagating Cipher Feedback
XCBC	eXtended Cipher Block Chaining Encryption

表 4: AES に提案されたその他の利用モード

略号	名前	日本語
KFB	Key Feedback Mode	鍵フィードバック
AES-hash	AES-hash	AES ハッシュ

20 日 New York/ 2002 年 10 月 10 日 Ontario, Canada/ 2002 年 12 月 10 日 Maryland/ 2003 年 4 月 10 日 San Diego, CA/ 2003 年 8 月 21〜22 日 Goleta, CA) とワークショップ (SISW2003, 2003 年 10 月 31 日 Washington D.C.) が開催された．

2005 年 11 月時点では、AES を使った LRW モード (Liskov-Rivest-Wagner) と、EME の二つの操作モードの提案がドラフトとして公開されている．

3.8 NESSIE

NESSIE (New European Schemes for Signatures, Integrity, and Ecnryption) [WWW8] はヨーロッパで 2000 年 1 月に開始された 3 年間のプロジェクトで，ブロック暗号，メッセージ認証コード，公開鍵暗号，ハッシュ関数といった暗号プリミティブの評価を行うことを目的としている．2003 年 2 月の最終報告書が公開され，メッセージ認証コードでは EMAC が portfolio に含まれた．

3.9 その他工業製品や業界標準などで利用されたもの

3GPP(3rd Generation Partnership Project) では，ブロック暗号 KASUMI [3GPPb] 及びその利用モード [3GPPa] が作成されている．暗号化方式として $f8$ が，メッセージ認証コードとして $f9$ が策定されている．それぞれ従来よく知られた利用モードとは異なるものを用いている．

RFC2040 ではブロック暗号 RC5(TM) の利用方法として，CBC をベースにした末端処理つき CBC モードが記載されている．これは CTS (Cipher Text Stealing, 暗号文窃盗) と呼ばれている [RFC2040].

また，Kerberos Version 4 では，認証暗号の目的で PCBC が用いられていたが，安全性の観点で欠陥が見つかったため，Version 5 では使われなくなった．

3.10 その他学術論文などに提案されたもの

主に学会でもブロック暗号の利用については議論されている．この中には，自己同期式利用モード各種 [M91, JR99, AGPS02] , iaPCBC[GD99], NCBC, RPC などがある．

また，ブロック暗号の利用方法，という観点からもいくつかの提案があり，ブロック暗号を暗号学的一方向性ハッシュ関数に変換する利用モード

[BRS02, PGV94] や，ブロック暗号から，AONT(All-or-Nothing-Transform, 完全出鱈目変換) の手法を与える利用モード [R97]，さらには，秘密でない乱数鍵が刺さったブロック暗号を鍵つきブロック暗号に変換する手法 [EM97] (さらにこれに対する安全性の検討 [D93])，鍵長の短いブロック暗号の全数探索への強化方法 [KR96] (およびそれに関する検討 [M02]) などがある．

4 安全性の定義

利用モードの安全性の議論は 1990 年代から多く議論されるようになった．その大きな話題のひとつが，証明可能安全性に関する議論である．これは，内部で用いるブロック暗号を疑似ランダム置換 (PRP) としてモデル化しながら，利用モードが提供する機能を数学的に証明するものである．この利用モードにおける証明可能安全性についてより深く紹介する．

4.1 ブロック暗号の安全性

ブロック暗号の代表的な安全性の定義として，擬似ランダム置換としての安全性と強擬似ランダム置換としての安全性がある．

4.1.1 擬似ランダム置換族

ブロック暗号 $E : \mathcal{K}_E \times \mathcal{M}_E \to \mathcal{M}_E$ は，$\mathcal{M}_E$ 上の置換族 $\{E_K(\cdot) \in \mathrm{Perm}(\mathcal{M}_E) \mid K \in \mathcal{K}_E\}$ と捉えることができる．ここで，$\mathrm{Perm}(\mathcal{M}_E)$ は $\mathcal{M}_E$ 上のすべての置換の集合である．

直感的に,「あるブロック暗号が擬似ランダム置換族である」とは，適応的選択平文攻撃を行う任意の敵が，置換族 $\{E_K(\cdot) \in \mathrm{Perm}(\mathcal{M}_E) \mid K \in \mathcal{K}_E\}$ と $\mathcal{M}_E$ 上のすべての置換の集合 $\mathrm{Perm}(\mathcal{M}_E)$ を区別できないことをいう．

より厳密には，敵 A として，オラクルにアクセスできるアルゴリズムを考える．何回かの質問の後，A は 1 ビットを出力する．ブロック暗号 $E : \mathcal{K}_E \times \mathcal{M}_E \to \mathcal{M}_E$ の，敵 A に対する，擬似ランダム置換としての安全性は，アドバンテージ $\mathbf{Adv}^{\mathrm{prp}}_E(A)$ によって評価される．ここで，

$$\begin{aligned}\mathbf{Adv}^{\mathrm{prp}}_E(A) \stackrel{\mathrm{def}}{=} & \left|\Pr(K \stackrel{R}{\leftarrow} \mathcal{K}_E : A^{E_K(\cdot)} = 1)\right. \\ & \left. - \Pr(P \stackrel{R}{\leftarrow} \mathrm{Perm}(\mathcal{M}_E) : A^{P(\cdot)} = 1)\right|\end{aligned}$$

と定義され，$A^{E_K(\cdot)}$ は質問 X に対し，$Y = E_K(X)$ を返すオラクル $E_K(\cdot)$ を持つ敵 A を表し，$A^{P(\cdot)}$ は質問 X に対し，$Y = P(X)$ を返すオラクル $P(\cdot)$ を持つ敵 A を表す．特に断りがなければ，質問は適応的に行う．すなわち，ある質問に対する答えを得た後，次の質問を行う．

$\mathrm{Perm}(\mathcal{M}_E)$ から一様ランダムに選ばれた P を $\mathcal{M}_E$ 上のランダム置換，あるいは単に，ランダム置換という．

計算量理論的安全性　上記の定義はある一つの敵に対する評価である．一般的に，敵が利用できる資源をパラメータにし，そのパラメータを利用するすべての敵の最大のアドバンテージを考える．ブロック暗号の擬似ランダム置換族としての安全性を考える場合に扱う資源は，実行時間 t とオラクルへの質問回数 q である．ここで，実行時間に関しては，ある計算のモデルが固定されているとする．その単位時間によって，ランダムに K を選ぶ時間や，$E_K(X)$ の計算にかかる時間があらわせるものとする．また，実行時間 t には，A の記述に要する長さ (A を記述するプログラムの長さ) が含まれているものとし，また，A の実行に関するすべての時間が含まれる．これにはランダムに K を選ぶ時間や，(オラクルとの) 入出力にかかる時間，等も含まれる．以降のすべての実行時間 t は同様に定義される．

$$\mathbf{Adv}^{\mathrm{prp}}_E(t,q) \stackrel{\mathrm{def}}{=} \max_A \{\mathbf{Adv}^{\mathrm{prp}}_E(A)\}$$

と定義される．ただし，最大値は実行時間 t, オラクルへの質問回数 q のすべての敵 A についてとる．

この定義においては，正確には「安全な擬似ランダム置換族」という概念は存在しない．すべてのブロック暗号 E は，ある大きさの $\mathbf{Adv}^{\mathrm{prp}}_E(t,q)$ をもつ置換族である.「E が安全な擬似ランダム置換族である」や,「E が擬似ランダム置換族である」という表現や仮定は,「適当に大きい t と q に対し，$\mathbf{Adv}^{\mathrm{prp}}_E(t,q)$ が十分小さい」ということを意図している．厳密な安全性の定理を言う場合にはこれらの表現は用いない．

4.1.2　強擬似ランダム置換族

「あるブロック暗号が強擬似ランダム置換族である」とは，適応的選択平文暗号文攻撃を行う任意の敵が，置換族 $\{E_K(\cdot) \in \mathrm{Perm}(\mathcal{M}_E) \mid K \in \mathcal{K}_E\}$ と $\mathcal{M}_E$ 上のすべての置換の集合 $\mathrm{Perm}(\mathcal{M}_E)$ を区別できないことをいう．

より厳密には，敵 A として，2 つのオラクルにアクセスできるアルゴリズムを考える．何回かの質問の後，A は 1 ビットを出力する．ブロッ

ク暗号 $E : \mathcal{K}_E \times \mathcal{M}_E \to \mathcal{M}_E$ の，敵 A に対する，強擬似ランダム置換としての安全性は，アドバンテージ $\mathbf{Adv}_E^{\mathrm{sprp}}(A)$ によって評価される．ここで，

$$\begin{aligned}\mathbf{Adv}_E^{\mathrm{sprp}}(A) \stackrel{\mathrm{def}}{=} & \left|\Pr(K \xleftarrow{R} \mathcal{K}_E : A^{E_K(\cdot),E_K^{-1}(\cdot)} = 1)\right. \\ & \left. - \Pr(P \xleftarrow{R} \mathrm{Perm}(\mathcal{M}_E) : A^{P(\cdot),P^{-1}(\cdot)} = 1)\right|\end{aligned}$$

と定義され，$A^{E_K(\cdot),E_K^{-1}(\cdot)}$ は質問 X に対し，$Y = E_K(X)$ を返す暗号化オラクル $E_K(\cdot)$ と，質問 Y に対し，$X = E_K^{-1}(Y)$ を返す復号オラクル $E_K^{-1}(\cdot)$ を持つ敵 A を表し，$A^{P(\cdot),P^{-1}(\cdot)}$ は質問 X に対し，$Y = P(X)$ を返す暗号化オラクル $P(\cdot)$ と，質問 Y に対し，$X = P^{-1}(Y)$ を返す復号オラクル $P^{-1}(\cdot)$ を持つ敵 A を表す．特に断りがなければ，質問は適応的に行う．すなわち，ある質問に対する答えを得た後，次の質問を行う．

計算量理論的安全性　ブロック暗号の強擬似ランダム置換族としての安全性を考える場合に扱う資源は，実行時間 t, 暗号化オラクルへの質問回数 q_e, 復号オラクルへの質問回数 q_d である．

$$\mathbf{Adv}_E^{\mathrm{sprp}}(t, q_e, q_d) \stackrel{\mathrm{def}}{=} \max_A \{\mathbf{Adv}_E^{\mathrm{sprp}}(A)\}$$

と定義される．ただし，最大値は実行時間 t, 暗号化オラクルへの質問回数 q_e, 復号オラクルへの質問回数 q_d のすべての敵 A についてとる．

一般に,「E が安全な強擬似ランダム置換族である」や,「E が強擬似ランダム置換族である」という表現は,「適当に大きい t, q_e, q_d に対し，$\mathbf{Adv}_E^{\mathrm{sprp}}(t, q_e, q_d)$ が十分小さい」ということを意図している．

4.1.3　上記以外のブロック暗号の安全性

上記以外にもブロック暗号の安全性定義がいくつか存在する．鍵関連攻撃を考慮した安全性定義 [BK03] などがこれに含まれる．

また，理想的ブロック暗号モデルというブロック暗号のモデル化がある．ハッシュ関数のランダムオラクルに対応するものであり，RMAC [JJ+02a, JJ+02b] の安全性解析に用いられた．

4.2　秘匿の安全性

暗号学における利用モードに関する安全性とは，想定した攻撃者に対するメカニズムの性質を議論する．よって，攻撃者をきちんと定義する必要がある．

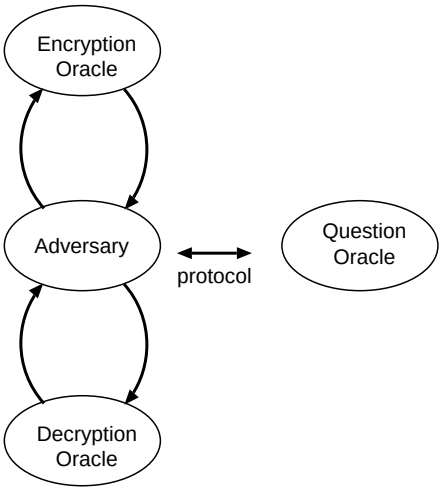


図 4: 証明可能安全性における攻撃者の例 (選択暗号文攻撃)

ここで考える攻撃者は限られた能力をもつものであって，指定されたこと以外の動作や，動作から得られる以外の情報の獲得は考えられていない．研究として，なるべく現実に近い，すなわち能力が高く，さまざまな能力をもつ攻撃者を検討する方向はあるが，完全ではない．

ここで考える攻撃者は，まず最低質問オラクルとのゲームを 1 回だけ行なう．また，攻撃者の能力としてそれとは別にさまざまなオラクルへの通信が可能である．

多くのモデルで，攻撃者が暗号化オラクルに対するアクセスを許している．これは攻撃者が，任意の (あとあと都合のよい) 平文を生成するとそれに対する暗号文を教えてもらえるものである．これを繰り返すことにより攻撃者が知識を獲得することが許される．

また，いくつかの暗号スキームに対する証明可能安全性では復号オラクル (暗号化と同様に，今度は暗号文に対して (必要であれば改竄検知をし，もし問題なければ) 平文を返答教えてくれるもの) を考える場合もある．

もし，攻撃者の能力として，選択平文攻撃を考えるならば，その安全性の検討では，攻撃者の暗号化オラクルのアクセスを検討する．また，選択暗号文攻撃では (通常，選択平文攻撃の能力を含んだ定義を考えることが多いので) 暗号化オラクルに加えて，復号オラクルを含めた評価を行なう．

以上の攻撃者の能力を特定した上で，スキームについての安全性を検討する．安全性は，秘匿，認証に分けて扱う．認証暗号は，これら秘匿，認証の両方の安全性を達成している．

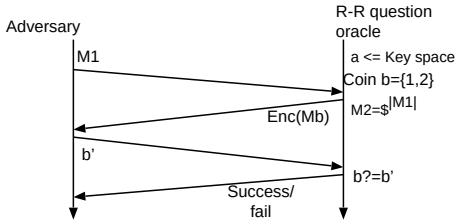


図 5: Real-or-Random notion を定義するゲームのプロトコル

まず，秘匿からはじめる．この分野における，秘匿の定義は厳密には複数存在する．しかし，その多くが計算量的に等価であることが知られているため，実質的にはひとつの安全性を達成すれば一般にいわれる秘匿の種類はある程度保証できる．

Real-or-Random (暗号文-乱数処理文識別) この秘匿に関する安全性を大雑把に理解するなら，攻撃者の目標は次の二つの暗号文を見分けることである，(1) 攻撃者自身が作成した平文に対応する暗号文，(2) その平文と同じ長さなだけで全然関係のない乱数を暗号化したもの．正式には，オラクルが行なう 2 種類のゲームで考える．オラクルはそれぞれのゲーム開始後には鍵を決定する．そして，攻撃者からメッセージ受信を待つ．ゲーム 1 では，メッセージを受信したら，さきほど決定した鍵で暗号化し，その結果を送信する．ゲーム 2 では，メッセージを受信しても，単にそれと同じ長さの乱数を発生し，それを送信する．

ある暗号化スキームが (ある条件下で，例えば選択平文攻撃などで) Real-or-Random で安全であるとは，(その条件が許される) どのような現実的な攻撃者も，ゲーム 1 とゲーム 2 を有意な確率で区別することが難しいことをいう．

定義 4.1 (Real-or-Random). 暗号化スキーム $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{K})$ が *Real-or-Random* の意味で $(t, q, \mu; \epsilon)$-安全であるとは，次で指定される任意の攻撃者の利得 $\mathsf{Adv}^{\mathrm{rr}}_A$ について下記が成り立つことである．攻撃者は，最大時間 t の間動作し，最大 q 回のオラクル質問 *(ここでは暗号化オラクルへの質問)* を行ない，これらの質問の長さが最大 μ ビットであるような攻撃者である．

$$\mathbf{Adv}^{\mathrm{rr}}_A = \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\cdot)} = 1] - \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\$^{(\cdot)})} = 1] \leq \epsilon.$$

Left-or-Right (左右平文暗号文識別) この秘匿に関する安全性でも二つのゲームを考える．質問オラクルへの攻撃者からの入力は，長さが同じ

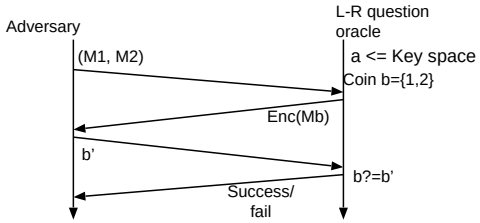


図 6: Left-or-Random notion を定義するゲームのプロトコル

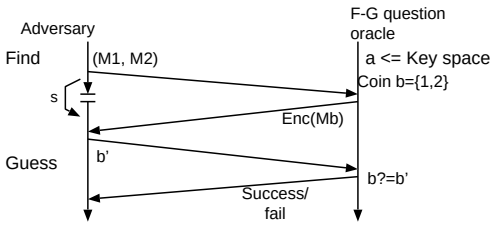


図 7: Find-then-Guess notion を定義するゲームのプロトコル

な平文のペアである (これらのペアが異なることが厳密には記載されていないが，同じであれば攻撃者が不当に利得を得る可能性があるので一般的には異なるもののみを考える)．オラクルは，ゲーム開始後には鍵を決定する．そして，攻撃者から二つの同じ長さのメッセージペア (M_1, M_2) の受信を待つ．メッセージを受信したら，ゲーム 1 では M_1 を，ゲーム 2 では M_2 をそれぞれ，先ほど生成した鍵で暗号化し，その結果を送信する．

ある暗号化スキームが (ある条件下で，例えば選択平文攻撃などで) Left-or-Right で安全であるとは，(その条件が許される) どのような現実的な攻撃者も，ゲーム 1 とゲーム 2 を有意な確率で区別することが難しいことをいう．

定義 4.2 (Left-or-Right)**.** 暗号化スキーム $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{K})$ が *Left-or-Right* の意味で $(t, q, \mu; \epsilon)$-安全であるとは，次で指定される任意の攻撃者の利得について下記が成り立つことである．攻撃者は，最大時間 t の間動作し，最大 q 回のオラクル質問 *(*ここでは暗号化オラクルへの質問*)* を行ない，これらの質問の長さが最大 μ ビットであるような攻撃者である *(*ただし，質問オラクルへのメッセージペア，(M_1, M_2) は同じ長さとする*)*.

$$\mathbf{Adv}^{\mathrm{lr}}_{A} = \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\mathsf{left}(\cdot,\cdot))} = 1] - \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\mathsf{right}(\cdot,\cdot))} = 1] \leq \epsilon.$$

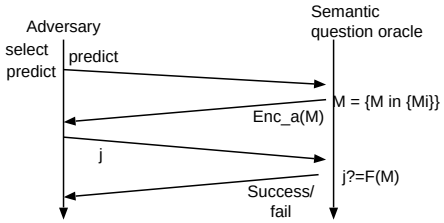


図 8: Semantic-security notion を定義するゲームのプロトコル

Find-then-Guess (発見-推測識別) Find-then-Guess は [GM84, MRS88] で扱っている多項式計算量的安全性の言い替えである．ここでは攻撃者は二つのステージを考える．第一の find ステージでは，攻撃者は最終的に同じ長さのメッセージペア (M_1, M_2) を生成するが，その目的は次のステージでこれらの暗号文を区別することである．また，この間に攻撃者は知識を蓄えることができ，最終的にあとで使う知識 s を生成してこのステージを終了する．

もうひとつの guess ステージでは，質問オラクルから暗号文 C を受信する．C はさきほどの (M_1, M_2) どちらかの暗号文である．攻撃者は知識 s を知っている．ここで，その暗号文 C がどちらの平文のものであるかを決めることができれば,「攻撃者の勝ち」とする．

ある暗号化スキームが (ある条件下で, 例えば選択平文攻撃などで) Find-then-Guess で安全であるとは，(その条件が許される) どのような現実的な攻撃者も，それらを有意な確率で区別することが難しいことをいう．

定義 4.3 (Find-then-Guess)**.** 暗号化スキーム $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{K})$ が *Find-then-Guess* の意味で $(t, q, \mu; \epsilon)$-安全であるとは，次で指定される任意の攻撃者の利得について下記が成り立つことである．攻撃者は，最大時間 t の間動作し，最大 q 回のオラクル質問 *(*ここでは暗号化オラクルへの質問*)* を行ない，これらの質問の長さが最大 μ ビットであるような攻撃者である．

$$\begin{aligned}\mathbf{Adv}^{\mathrm{fg}}_{A} \;=\; & 2 \cdot \Pr[a \leftarrow \mathcal{K} : (M_1, M_2, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{find}); b \leftarrow \{1,2\}; C \leftarrow \mathcal{E}_a(M_b) : \\ & A^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, C, s) = b] - 1 \leq \epsilon.\end{aligned}$$

Semantic (意味抽出) Goldwasser と Micali[GM84] では，semantic security を「暗号文が与えられてから平文に関してわかる情報というのは，暗号文がなくともわかるものだけだ」と説明している．ここでの semantic は公開鍵暗号における semantic security をそのまま適応する．f を平文を引数にとることができる関数とする．この関数は，攻撃者が (暗号文から)

知ろうとしている情報の種類を表していると考えることができる．平文空間は確率的な分布を取るものとして考える．任意の整数 m に対して,「平文空間における m 分布」とは，m ビット以下の文字列で代表される平文空間上の確率分布，の集合 $\mathcal{M} = \{\mathcal{M}_\gamma\}_{\gamma\in\{0,1\}^{\leq m}}$ とし，すべての $\mathcal{M}_\gamma$ が有効 (valid) とする．ここで，有効とはすべての確率分布 $\mathcal{M}_\gamma$ について，確率が非 0 の文字列すべてが同じ長さであり，その長さは最大で m である，ということを意味する．$p^*_{f,\mathcal{M}_\gamma} = \max_{C^*}\{\Pr[M \leftarrow \mathcal{M}_\gamma : f(M) = C^*]\}$ と定義する．これは平文の確率分布でもっともありうる $f(\cdot)$ 値である．

攻撃者は二つのステージを考える．第一の select ステージでは，攻撃者は都合の良い平文分布 $\mathcal{M}_\gamma$ を生成する．もうひとつの predict ステージでは，質問オラクルが，指定された平文分布に従って無作為にメッセージ M を生成し，暗号文 C を送信する．攻撃者はこれを受信し，$f(M)$ 値を予想しようとする．

ある暗号化スキームが (ある条件下で，例えば選択平文攻撃などで) Semantic で安全であるとは，(その条件が許される) 関数 f と分布 $\mathcal{M}$ に対して，どのような現実的な攻撃者も，$p^*_{f,\mathcal{M}_\gamma}$ を越える確率で $f(M)$ を予想することができない，ことをいう．

従来 (つまり公開鍵暗号で議論されていたところ) の定義では，この条件はすべての関数 f について成り立つ必要があった．共通鍵暗号においては，関数 f と確率分布 $\mathcal{M}$ がパラメータとする．このことで，ある特殊な平文の性質が，ある特別な分布において，ちゃんと情報が隠れいているか/いないかを議論できる．

定義 4.4 (Semantic)**.** 関数 f を，平文空間を入力としてなにかしらのバイナリ文字列を出力する関数とする．$\mathcal{M} = \{\mathcal{M}_\gamma\}_{\gamma\in\{0,1\}^{\leq n}}$ を平文空間における m 分布とする．

暗号化スキーム $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{K})$ が *Semantic* の意味で f と $\mathcal{M}$ に対して $(t, q, \mu; \epsilon)$-安全であるとは，次で指定される任意の攻撃者の利得について下記が成り立つことである．攻撃者は，最大時間 t の間動作し，最大 q 回のオラクル質問 *(*ここでは暗号化オラクルへの質問*)* を行ない，これらの質問の長さが最大 μ ビットであるような攻撃者である．

$$\mathbf{Adv}^{\mathrm{sm}}_A(f, \mathcal{M}) = \mathbf{E}[a \leftarrow \mathcal{K} : (\gamma, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{select}) : \alpha(a, \gamma, s)] \leq \epsilon.$$

ここで

$$\alpha(a, \gamma, s) = \Pr[M \leftarrow \mathcal{M}_\gamma; C \leftarrow \mathcal{E}_a(M) : A^{\mathcal{E}_a(\cdot)}(\mathsf{predict}, C, s) = f(M)] - p^*_{f,\mathcal{M}_\gamma}.$$

Ciphertext-Random (暗号文-乱数)　近年の認証暗号など新しい利用モードの提案では，Real-or-Random とは異なる，暗号文-乱数不可識別性

で秘匿の安全性を証明する利用モードがある．Real-or-Random に酷似するため詳細は省略する．

Real-or-Random では，Game 2 において，乱数の暗号文を返答していた．この定義では，乱数そのものを返信するプロトコルでゲームをする．

4.3 従来の秘匿定義の関係

上記，秘匿の定義のうち前 4 つの定義については，[BDJR97] で詳細に扱われており，同じ文献でこれら 4 つの定義間の関係が明らかにされている．

4 つの定義で最強とされるものは，Left-or-Right と Real-or-Random である．これらは安全性パラメータも損なわない多項式還元が実現されており，他の 2 つの定義へも効率的に還元できる．すなわち秘匿の定義としては最強の定義である．

これらに対して Find-then-Guess と Semantic については，安全性パラメータが多少欠損するものの，これらが言えれば上記 2 方式の安全性も保証することができる．

以上により，上記 4 つの定義のどれかを達成していれば共通鍵暗号における秘匿は十分なレベルが達成できていると言える．

4.4 メッセージ認証コードの安全性

メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の安全性には，弱偽造不可能性と強偽造不可能性がある．

どちらの場合も，敵 A として，タグ生成オラクルと確認オラクルにアクセスできるアルゴリズムを考える．$A^{MAC\text{-}\mathcal{G}_K(\cdot),MAC\text{-}\mathcal{V}_K(\cdot,\cdot)}$ は，メッセージ M に対し，タグ $T = MAC\text{-}\mathcal{G}_K(M)$ を返すタグ生成オラクル $MAC\text{-}\mathcal{G}_K(\cdot)$ と，メッセージ，タグのペア (M, T) に対し，accept or reject $= MAC\text{-}\mathcal{V}_K(M, T)$ を返す確認オラクル $MAC\text{-}\mathcal{V}_K(\cdot,\cdot)$ をもつ敵をあらわす．質問は適応的に行う．すなわち，ある質問に対する答えを得た後，次の質問を行う．

4.4.1 弱偽造不可能性

弱偽造不可能性の意味でメッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ を破ろうとする敵 A が，タグ生成オラクルに q 個のメッセージ $M_1, \ldots, M_q$ を質問し，その答え $T_1, \ldots, T_q$ を得たとする．また，確認オ

ラクルに q' 個のメッセージ，タグのペア $(M'_1, T'_1), \ldots, (M'_{q'}, T'_{q'})$ を質問したとする．

ある i に対し，$MAC\text{-}\mathcal{V}_K(M'_i, T'_i) = \text{accept}$ であり，$M'_i \notin \{M_1, \ldots, M_j\}$ であれば，A は弱偽造不可能性の意味で偽造に成功した，という．ここで，$\{M_1, \ldots, M_j\}$ は，(M'_i, T'_i) を確認オラクルに質問する以前に，タグ生成オラクルに送った質問である．

直感的には，見たことのないメッセージに対するタグを出力できたらなば，偽造に成功したことになる．

メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，敵 A に対する，弱偽造不可能性の意味での安全性は，アドバンテージ $\mathbf{Adv}^{\text{w-uf}}_{E}(A)$ によって評価される．ここで，

$$\mathbf{Adv}^{\text{w-uf}}_{MAC}(A) \stackrel{\text{def}}{=} \Pr(K \xleftarrow{R} MAC\text{-}\mathcal{K} : A^{MAC\text{-}\mathcal{G}_K(\cdot), MAC\text{-}\mathcal{V}_K(\cdot,\cdot)} \text{が} \\ \text{弱偽造不可能性の意味で偽造に成功})$$

と定義される．

計算量理論的安全性　メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，弱偽造不可能性の意味での安全性を考える場合に扱う資源は，実行時間 t, タグ生成オラクルへの質問回数 q, それら質問の長さ σ (ビット単位，もしくはブロック単位)，確認オラクルへの質問回数 q', それら質問の長さ σ' (ビット単位，もしくはブロック単位) である．実行時間 t はブロック暗号と同様に定義される．

$$\mathbf{Adv}^{\text{w-uf}}_{MAC}(t, q, \sigma, q', \sigma') \stackrel{\text{def}}{=} \max_{A} \left\{ \mathbf{Adv}^{\text{w-uf}}_{MAC}(A) \right\}$$

と定義される．ただし，最大値は実行時間 t, タグ生成オラクルへの質問回数 q, それら質問の長さ σ, 確認オラクルへの質問回数 q', それら質問の長さ σ' のすべての敵 A についてとる．

4.4.2　強偽造不可能性

強偽造不可能性の意味でメッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ を破ろうとする敵 A が，タグ生成オラクルに q 個のメッセージ $M_1, \ldots, M_q$ を質問し，その答え $T_1, \ldots, T_q$ を得たとする．また，確認オラクルに q' 個のメッセージ，タグのペア $(M'_1, T'_1), \ldots, (M'_{q'}, T'_{q'})$ を質問したとする．

ある i に対し，$MAC\text{-}\mathcal{V}_K(M'_i, T'_i) = \text{accept}$ であり，$(M'_i, T'_i) \notin \{(M_1, T_1), \ldots, (M_j, T_j)\}$ であれば，A は強偽造不可能性の意味で偽造に成功した，

という．$\{(M_1, T_1), \ldots, (M_j, T_j)\}$ は，(M'_i, T'_i) を確認オラクルに質問する以前に，タグ生成オラクルに送った質問とその答えである．

直感的には，見たことのないメッセージ，タグのペアを出力できたらなば，偽造に成功したことになる．タグが異なっていれば，メッセージ自体は見たことがあってもよい．

メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，敵 A に対する，強偽造不可能性の意味での安全性は，アドバンテージ $\mathbf{Adv}^{\text{s-uf}}_{E}(A)$ によって評価される．ここで，

$$\mathbf{Adv}^{\text{s-uf}}_{MAC}(A) \stackrel{\text{def}}{=} \Pr(K \stackrel{R}{\leftarrow} MAC\text{-}\mathcal{K} : A^{MAC\text{-}\mathcal{G}_K(\cdot), MAC\text{-}\mathcal{V}_K(\cdot,\cdot)} \text{ が} \\ \text{強偽造不可能性の意味で偽造に成功})$$

と定義される．

計算量理論的安全性　メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，強偽造不可能性の意味での安全性を考える場合に扱う資源は，弱偽造不可能性の場合と同様である．

$$\mathbf{Adv}^{\text{s-uf}}_{MAC}(t, q, \sigma, q', \sigma') \stackrel{\text{def}}{=} \max_{A} \left\{ \mathbf{Adv}^{\text{s-uf}}_{MAC}(A) \right\}$$

と定義される．ただし，最大値は実行時間 t, タグ生成オラクルへの質問回数 q, それら質問の長さ σ, 確認オラクルへの質問回数 q', それら質問の長さ σ' のすべての敵 A についてとる．

4.4.3　$MAC\text{-}\mathcal{G}$ が決定的アルゴリズムである場合の安全性

$MAC\text{-}\mathcal{G}$ が決定的アルゴリズムの場合，弱偽造不可能性の意味での安全性と強偽造不可能性の意味での安全性は同一の定義となる．また，この場合，タグ生成オラクルが確認オラクルのかわりになり得る．すなわち，タグ生成オラクルに M_i を質問し，T_i を得たなら，確認オラクルは質問 (M_i, T_i) に対しては accpet を返し，質問 (M_i, T'_i) (ただし $T'_i \neq T_i$) に対しては reject を返す．したがって，q' と σ' のパラメータを用いないで，q と σ にこれらを含めるのが一般的である．$MAC\text{-}\mathcal{G}$ が決定的アルゴリズムの場合，弱偽造不可能性と強偽造不可能性とを区別せず，単に偽造不可能性という．

敵 A として，タグ生成オラクルにアクセスできるアルゴリズムを考える．$A^{MAC\text{-}\mathcal{G}_K(\cdot)}$ は，メッセージ M に対し，タグ $T = MAC\text{-}\mathcal{G}_K(M)$ を返すタグ生成オラクル $MAC\text{-}\mathcal{G}_K(\cdot)$ をもつ敵をあらわす．質問は適応的に行う．すなわち，ある質問に対する答えを得た後，次の質問を行う．

偽造不可能性の意味でメッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ を破ろうとする敵 A がタグ生成オラクルにメッセージ $M_1, \ldots, M_j$ を質問し，その答え $T_1, \ldots, T_j$ を得たとする．タグ生成オラクルへの質問の途中，A は偽造文 (M_{j+1}, T_{j+1}) を出力する．

$MAC\text{-}\mathcal{V}_K(M_{j+1}, T_{j+1}) = \text{accept}$ であり，$M_{j+1} \notin \{M_1, , \ldots, M_j\}$ であれば，A は偽造不可能性の意味で偽造に成功した，という．$\{M_1, \ldots, M_j\}$ は，(M_{j+1}, T_{j+1}) を出力する以前に，タグ生成オラクルに送った質問である．ある偽造文が reject された場合でも，A はさらにタグ生成オラクルに質問を続け，あたらな偽造文を出力してよい．ただし，(M_{j+1}, T_{j+1}) はタグ生成オラクルに対する質問として数える．

直感的には，見たことのないメッセージに対するタグを出力できたらなば，偽造に成功したことになる．

メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，敵 A に対する，偽造不可能性の意味での安全性は，アドバンテージ $\mathbf{Adv}^{\mathrm{mac}}_{MAC}(A)$ によって評価される．ここで，

$$\mathbf{Adv}^{\mathrm{mac}}_{MAC}(A) \stackrel{\mathrm{def}}{=} \Pr(K \stackrel{R}{\leftarrow} MAC\text{-}\mathcal{K} : A^{MAC\text{-}\mathcal{G}_K(\cdot)} \text{ が 偽造不可能性の意味で偽造に成功})$$

と定義される．

計算量理論的安全性　メッセージ認証コード $MAC = (MAC\text{-}\mathcal{K}, MAC\text{-}\mathcal{G}, MAC\text{-}\mathcal{V})$ の，偽造不可能性の意味での安全性を考える場合に扱う資源は，実行時間 t, タグ生成オラクルへの質問回数 q (M' を含む), それら質問の長さ σ (ビット単位，もしくはブロック単位，M' の長さも含む) である．実行時間 t はブロック暗号と同様に定義される．

$$\mathbf{Adv}^{\mathrm{mac}}_{MAC}(t, q, \sigma) \stackrel{\mathrm{def}}{=} \max_A \{\mathbf{Adv}^{\mathrm{mac}}_{MAC}(A)\}$$

と定義される．ただし，最大値は実行時間 t, タグ生成オラクルへの質問回数 q, それら質問の長さ σ のすべての敵 A についてとる．

4.4.4　上記以外の安全性

上記以外にもいくつかの安全性定義が存在する．それらについては，そのつど説明をする．

4.5 攻撃者の能力

安全性の証明を考える上で攻撃者の能力を正確に決める必要がある．これについては，暗号化 (秘匿の利用モード，ならびに認証暗号の利用モード) と認証 (MAC 生成のモード) で独立に考える．

暗号における証明可能安全性では，攻撃者の能力として

A 攻撃者自身で都合良く選んだ平文に対して，それに対応する暗号文を知ることができる,

B 攻撃者自身で都合良く選んだ暗号文に対して，それに対応する平文を知ることができる．

の二つの能力を考える．そして暗号が扱われる現実世界や，これまで提案されてきた利用モードの性質から，現状 (B) のみが許されるような攻撃者は考えない．よって，暗号の安全性の前提となる攻撃者の種類は (A) のみを対象とした場合 (選択平文攻撃) か，もしくは (A)(B) 両方が可能な攻撃者を対象とした場合 (選択暗号文攻撃) のふたとおりどちらかである．

4.6 証明可能安全性の仮定

証明可能安全性と現実での暗号利用には大きな差がある．その差に関する研究結果もいくつか知られてきているが，それがすべてではない．

まず，初期値に関する議論がある．これらすべての証明可能安全性において，初期値を正しく生成する必要がある．しかし，それに必要とされる乱数性や信頼性 (カウンタのリセットを防止するメカニズムなど) を現実的に暗号に利用するのは多くの場合困難である．

次に攻撃者の能力である．多くの秘匿に関する証明可能安全性は，自分で生成した暗号文に対応する平文の情報を知ることができないことになっている．しかし，暗号文の改竄や，あるいは通信ノイズがあるような通信路の暗号処理では，攻撃者にいかなる復号結果を渡してはならない．現に，安易なチェックサムを用意してしまった暗号方式から，チェックサムの合否を用いて平文を読みとる攻撃手法が発見された事例がある [V02]．これについてはあとで述べる．

最後に，攻撃者の暗号文を取得できる能力は，メッセージ単位に限定されている，という点である．場合によっては，メッセージという単位よりもより細かい単位 (例えばブロック単位など) で攻撃を組み立てる攻撃者が存在するかもしれない．また，このような，攻撃が可能である場合，証明可能だった安全性が崩れる例が知られている [JMV02].

4.7　利用モードに対する攻撃

ここまで議論したような証明可能安全性は，現実世界が完全にモデル化されたとおりに動く場合に限って現実的に信頼できる．しかしながら，実際にはそうでない場合がある．安全性に関する議論の最後に，これまでに知られている攻撃関連の話題をいくつか紹介する．

[V02] では，CBC モードに対する攻撃を示している．不適切な実装として，改竄検知を目的としたパディングとそれによる改竄検知がいくつかの標準化で実装された．この改竄検知機能を利用することで，本来秘匿されるはずの情報を読みとることができる．CBC モードは秘匿にのみ使われるべきであり，不用意に，利用モードの範囲外のことを行なうと，もともとの安全性も崩れる典型的な例である．

[JMV02] では，CBC, IACBC, (そして公開鍵とのハイブリッド暗号 GEM) に対する秘匿に関する攻撃の可能性を示している．ここでは現実的には考えにくいほど強力な攻撃者を想定するが，攻撃は攻撃である．従来安全性評価は，攻撃者の判定したいメッセージ対はメッセージストリームを最後まで消化した上で暗号文の最初のブロックが生成されていた．ところが，オンライン処理を用いるときなどは，かならずしも暗号文出力のために，メッセージを受信終了をまたずに出力することは多い．このような攻撃者の場合，暗号文で見分けがつくようなメッセージ対を生成することができる，という攻撃である．

また DES に対する辞書攻撃，ならびに鍵の全数探索に対する強度向上を目的とした，DES の三重利用モードに対する解析がある．Biham は [B96] で，多くの多重利用モードが有効な強化策となっていないことを示している．こののち，Wagner はさらに初期値の制御を使うことにより，Biham が安全であろうとしたいくつかのモードについても別の懸念があることを指摘した [W98].

5　秘匿に関する利用モード

この章では，これまで知られている秘匿に関する利用モードのうち，主に工業的に用いられているものや，機能面などで重要視する必要のあるものを詳細に説明する．

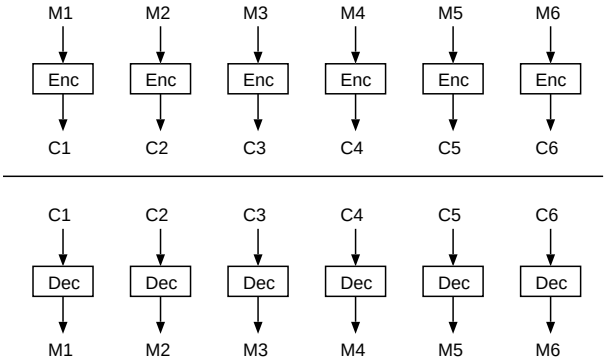


図 9: ECB モードの暗号化と復号のブロック図

5.1 ECB

仕様の概要 ECB(Electronic CodeBook, 電子辞書) モードは，平文長が n の倍数であるような平文に対して暗号化を行なう利用モードである [FIPS81]．手法は，平文を n ビット毎のブロックに分割し (それぞれを M_i とする)，それぞれ独立にブロック暗号の暗号化関数の入力とする．その結果得られた出力が暗号文ブロック (C_i) となり，暗号文はそれらを接続したものである．

$$C_i = Enc_K(M_i).$$

この利用モードには初期値がない．平文と鍵のみから暗号文が生成される．復号はその逆関数である．

$$M_i = Dec_K(C_i).$$

安全性 ECB モードには以下のような欠点があるため、その特性が必要でない限り利用すべきではない．具体的には，平文がオールゼロなど，ある文字列を繰り返すものを想定すると，暗号文もあるパターンを繰り返すことになる．一般化して，同じ平文パターンは同じ暗号文パターンとして再現されるため，暗号文からそのような情報が漏洩する．

この欠点を補う方法としては，平文ブロックが衝突しない (同じ値にならない) ように圧縮を掛けたり，平文としてエントロピの高いデータを用いることなどが挙げられる．しかしながら，これらの対策も万全ではないため，できる限り他の利用モードを使うべきである．

効率 平文長 $t \times n$ ビットに対して，ブロック暗号を t 回呼び出すのみであり，処理効率はよい．

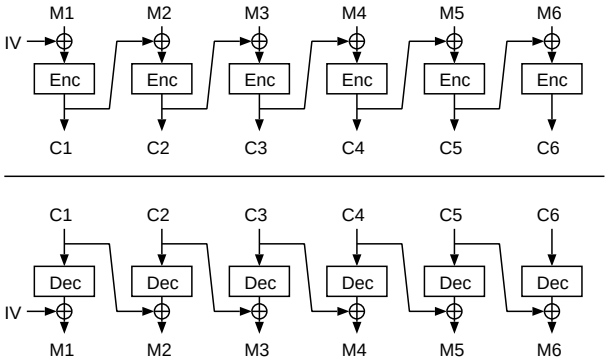


図 10: CBC モードの暗号化と復号のブロック図

エラー伝播　暗号文を伝送するなどした時に発生した 1 ビットのエラーは該当ブロック n ビットに影響を及ぼす可能性がある．

同期ずれについては (同期のずれる幅がブロック単位である特殊な場合を除いて)，別途再同期のメカニズムが必要である．

並列処理性など　暗号化, 復号ともに並列処理性と，処理順序不問性 (Out-of-order) 性がある．すなわち，ブロック単位でデータが入れ替わったとしても，その順序入れ替えをすることなく，到着した順序に復号処理に渡すことができる．もちろん，復号結果は，適切な順序に並べかえねば正しい平文には戻らない．

復号　復号時にも，暗号化処理と同様に並列処理性や Out-of-order の利点がある．復号処理には，ブロック暗号の復号関数を使う．

5.2 CBC

仕様の概要　CBC(Cipher Block Chaining, 暗号文ブロック連鎖) モードは，平文長が n の倍数であるような平文に対して暗号化を行なう利用モードである [FIPS81]．手法は，平文を n ビット毎のブロックに分割し (それぞれを M_i とする)，中間値 $H_i = M_i \oplus C_{i-1}, C_0 = IV$ を生成したあと，それをブロック暗号の暗号化関数の入力とする．その結果得られた出力が暗号文ブロック (C_i) となり，暗号文はそれらを接続したものである．

$$C_i = Enc_K(M_i \oplus C_{i-1}).$$

復号はその逆関数である．

$$M_i = Dec_K(C_i) \oplus C_{i-1}.$$

安全性　CBC モードの安全性は [BDJR97] で議論されている．ここでは，以下の条件がすべて満たされる場合において秘匿の意味で証明可能安全性を持つ．

1. 攻撃者は適応的選択平文攻撃のみである．
2. 初期値生成が以下に限定されるものである．
   (a) 攻撃者が事前に知ることができない乱数
   (b) 信頼できる nonce を一度ブロック暗号 (鍵は暗号化鍵でよい) で撹拌したもの
3. 内部で用いるブロック暗号が，擬似ランダム置換モデル以上の安全性をもつ．
4. 攻撃者の選択平文に対する暗号文の獲得は，メッセージ単位である．

より，具体的には，Left-or-Right 不可識別性 (秘匿に関する定義のひとつ) の観点からいくつかの安全性が [BDJR97] で示されており:

1. 内部のブロック暗号をランダム関数モデルに置き換えた場合の，CBC モードの安全性が与えられている．Left-or-Right におけるアドバンテージの定義は参考文献を参照頂くとして，そのアドバンテージが以下の式で評価できる．

   $$\mathbf{Adv}^{\mathrm{lr}}_{CBC-\rho} \leq (\mu^2/n^2 - \mu/n) \cdot 2^{-n}.$$

   ここで，攻撃者の能力として最大 q 回の選択平文質問を行ない，その平文長が合計 μ ビットとする．

2. 内部のブロック暗号を擬似ランダム関数モデルに置き換えた場合の，CBC モードの安全性が与えられている．具体的には，擬似ランダム関数のパラメータを $(t', q'; \epsilon')$ とすると，任意の q に対して，これを使った CBC モードについての安全性が $(t, q, \mu; \epsilon)$-安全であることをいうための定数 c が存在する．ここで

   $$(t, \mu, \epsilon) = (t' - c\mu, q'n, 2\epsilon' + (\mu^2/n^2 - \mu/n) \cdot 2^{-n}).$$

効率　平文長 $t \times n$ ビットに対して，ブロック暗号を t 回呼び出すのみであり，処理効率はよい．

エラー伝播　暗号文を伝送するなどした時に発生した 1 ビットのエラーは該当ブロック n ビットに影響を及ぼす可能性があり，次のブロックの該当部分 1 ビットが確実に反転する．

同期ずれについては，ECB と同様，特殊な場合を除いてそれ自身で回復しないため，別途再同期のためのメカニズムが必要である．

並列処理性など　暗号化には，まったくの並列処理性がない．一方，復号では，ブロック暗号処理に関する並列処理は可能である．しかしながら，平文データを復元するためには前ブロックの暗号文ブロックが必要であることを注意しなければならない．

また，ECB モードほど小さな単位では実現できないが，ある程度ブロックがまとまれば, Out-of-order 的な復号処理も可能な場合がある．すなわち，t ブロック単位でデータが入れ替わったとしても，その順序いれかえをすることなく，到着した順序に復号処理に渡すことができ，その場合，最初のブロックを除いた $t-1$ ブロックは正常に復号可能である．

ただし，例外的に並列処理が可能な運用もある．ANSI X3.106 や ISO 10116 では，CBC モードをインターリーブすることにより，ある程度の並列度を持たせることができる暗号方式を記載している．具体的には，独立な CBC モードを並列度数だけ飛ばしながらメッセージストリームを処理する仕様である．この場合，初期値も並列度数だけ用意せねばならず，それぞれ独立かつランダムに選択する必要がある．

復号　復号時に関する特別な注意事項はない．復号処理には，ブロック暗号の復号関数を使う．

CTS　CTS(CipherText Stealing, 暗号文窃盗) モードは，RFC2040[RFC2040] で提案された，CBC モード向けの端数処理モードである．RFC ではバイト単位の端数処理のみが定義されているが，単純に一般化することで n ビット以上の任意のビット数のメッセージに対して処理可能となる．

このモードはほとんどの処理が CBC モードであるので，安全性以外の主な特徴は CBC モードに準じる．

安全性については特別に議論された技術文書は見当たらないが，次のように考えることで秘匿に関する安全性は保持できていると考える．

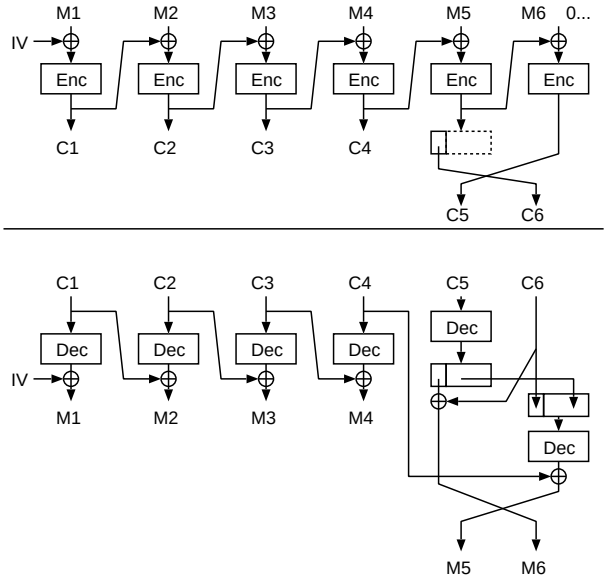


図 11: CTS モードの暗号化と復号のブロック図

CBC$^+$ という新しい利用モードを考える．従来の CBC モードを暗号化する場合に，n ビットの 0 パディングを行なってから CBC モードを処理する．CBC$^+$ と同様に安全であると考えられる．

ここで $m \times n$ ビット長の CBC$^+$ モードでの暗号化結果と $(m-1) \times n + t(1 \leq t < n)$ ビット長の CTS モードでの暗号化の強度は，後者，すなわち CTS のほうが強力である．なぜならば，CTS における任意の攻撃者の振舞いは，すべて前者に対する攻撃者として再現できるからである．よって CTS は CBC と同程度に強力であると考えられる．

5.3 k-CFB

仕様の概要 CFB(Cipher FeedBack, 暗号文フィードバック) モードは，パラメータ k を持つブロック暗号利用モードである [FIPS81]．平文長が k の倍数であるような平文に対して暗号化を行なう利用モードであることから，バイト単位のデータなど，データ単位長がブロック長の倍数でないような場合に用いられていた．便宜的に内部レジスタ R を考えながら処理を説明する．

k ビットの倍数長のメッセージ M は，k ビット毎のブロックに分割しする (それぞれを M_i とする)．初期値 IV は R の初期値 R_0 である．各ブ

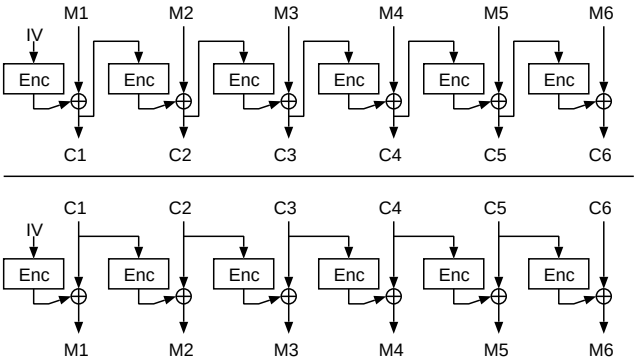


図 12: n-CFB モードの暗号化と復号のブロック図

ロックでのブロック処理は，まず中間値 $H_i = Enc_K(R_i)$ を生成することから始まり，このうち上位 k ビットの値 $\hat{H}_i$ を，平文ブロック M_i と排他的論理和をとることで暗号文ブロック $C_i = M_i \oplus \hat{H}_i$ を得る．最後に R を更新する．R を上位へ k ビットシフトし，シフトの際，0 が埋められた下位 k ビットに C_i を埋め込む．よって，$k = n$ の場合は，$R_i = C_i$ となる．

$$\begin{aligned} C_i &= M_i \oplus \mathrm{msb}_k(Enc_K(R_{i-1})), \\ R_i &= ((R_{i-1}) \ll k) \oplus C_i. \end{aligned}$$

復号はその逆関数である．

$$\begin{aligned} M_i &= C_i \oplus \mathrm{msb}_k(Enc_K(R_{i-1})), \\ R_i &= ((R_{i-1}) \ll k) \oplus C_i. \end{aligned}$$

安全性　CFB モードの安全性については，[AGPS02] で評価している．ここではレジスタ値の衝突確率をもとに Left-or-Right における攻撃者のアドバンテージの上限を求めている．$k < n$ の場合には，異なる時間のあいだでレジスタの値同士が独立でない．これも考慮した上での評価である．

ランダム関数を使った場合には，攻撃者のアドバンテージは

$$\epsilon^{\mathrm{lr}}_{\mathrm{CFB}-\rho} \leq q(q-1)2^{-l-1},$$

となる．ここで t は攻撃者の計算時間，q は攻撃者の質問回数，l はランダム関数の入力長，L はランダム関数の出力長である．

さらに，擬似ランダム関数を使った k-CFB モードについての安全性の評価結果も調べられており，l ビット入力-L ビット出力の $(t', q'; \epsilon')$-安全

な擬似ランダム置換を使った場合，CFB モードは $(t, q, \mu; \epsilon)$-安全である．ここで

$$(t, q, \mu, \epsilon) = (t' - q \times t_{\mathrm{CFB}} - t_{const}, q', q'L, 2\epsilon' + q(q-1)2^{-l-1}),$$

であり，t_{CFB} はランダム関数の呼び出しを除いたに CFB モード 1 ブロック処理に必要な処理時間である．

ただし，特に k が小さい場合には，初期値に注意する必要がある．例えば，0 ばかり続く平文（もしくは 1 ばかり続く平文）を初期値 $IV = 0^n$ や $IV = 1^n$ の $1 - CFB$ で暗号化した場合，約半分の鍵に対しては内部レジスタの更新がまったくおこなわれないため安全性に問題が生じることとなる [W02b].

効率　CFB モードは，パラメータの値に応じて処理効率が変化し，場合よっては，他のモードよりも極端に非効率的となる．

具体的には mk ビットのメッセージを暗号化するためには m 回のブロック暗号の呼び出しを必要とする．$k = n$ の場合，ECB や CBC と同じ程度の効率であるが，それ以外の場合，約 n/k 倍の処理量となる．

エラー伝播　1 ビットの暗号文におけるエラーにより，まず該当の平文ビットの反転が起こる．さらに該当エラーがレジスタに残る限り，平文回復ができないので，その間はエラーがおき続ける可能性がある．これは最悪，$\lceil n/k \rceil$ ブロック分，エラーが起こる可能性がある．

並列処理性など　CBC モードと同様，暗号化には並列処理性がない．復号では，該当ブロックのブロック暗号処理結果は次のブロック暗号処理に直接影響しない．よって構成上はパイプライニングなど並列処理性はある．しかし，該当ブロックを処理するためには，該当ブロック以前の暗号文ブロックが必要であるので，各々のブロック暗号エンジンでこれらをバッファリングするメカニズムが必要である．これらバッファは左右にずれているだけであるので，(並列度に応じた長い) バッファを共有することでも実現可能である．

また CBC モードと同様に，例外的に並列処理が可能な運用もある．ANSI X3.106 や ISO 10116 では，CFB をインターリーブすることにより，ある程度の並列度を持たせることができる暗号方式を記載している．具体的には，独立な CFB モードを並列度数だけ飛ばしながらメッセージストリームを処理する仕様である．この場合，初期値も並列度数だけ用

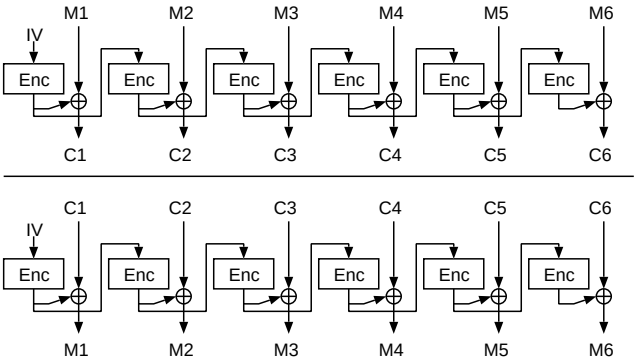


図 13: OFB モードの暗号化と復号のブロック図

意せねばならず，それぞれが安全であるためには，ランダムかあるいは攻撃者に選択されないような nonce にする必要がある．

復号　CFB モードでは，ブロック暗号の復号関数を利用しない．よって，CFB 暗号化，CFB 復号の両方の機能を実装する場合には，その実装コストは，CBC や ECB に比較して軽いことが期待できる．

自己同期性　CFB の大きな特徴として，自己同期性がある．これはブロック単位でのデータの欠損や挿入については，ある程度のエラーブロックをひきずりながらも，その後には，復号処理が回復するものである．

この機能は CBC にも一応あてはめることはできる (ただし同期パケットの境界がブロック暗号のブロック長) が，比較的大きいためこの機能が現実的に便利であることはあまりない．

CFB の場合，ブロック長を任意に設定することができるため，例えばバイト単位や，極端な例ではビット単位の同期ずれやデータ欠損挿入などにも回復する強みがある．ただし，注意を要するのは，ビット単位やバイト単位など短いデータ境界での自己同期を期待すればするほど，その処理負荷が大きくなる．

これを解決したのが，OCFB モードである．詳細は OCFB モード参照.

5.4　OFB

仕様の概要　OFB(Output FeedBack, 出力フィードバック) モードは，初期値のみに依存し逐次的に擬似乱数を生成しながら暗号化を行なう方法

であり，任意のビット長の平文を処理できる [FIPS81]．まず，平文を n ビット毎のブロックに分割 (それぞれを M_i とする) し，最後の端数の部分は端数ブロックとして扱う．初期値 IV を内部レジスタの初期値 H_0 とする．H_{i-1} をブロック暗号入力とし，暗号化処理の結果を H_i とする (すなわち次のブロックの内部レジスタの値にもなる)．これより暗号文ブロック $C_i = M_i \oplus H_i$ を生成する．

$$\begin{aligned} H_i &= Enc_K(H_{i-1}), \\ C_i &= M_i \oplus H_i. \end{aligned}$$

この利用モードには初期値がない．平文と鍵のみから暗号文が生成される．復号はその逆関数である．

$$\begin{aligned} H_i &= Enc_K(H_{i-1}), \\ M_i &= C_i \oplus H_i. \end{aligned}$$

安全性　OFB モードに関するきちんとした安全性の証明は知られていない．しかし，ブロック暗号出力全体をそのまま入力に戻すことで内部のブロック暗号が理想的である場合，周期が約 2^{n-1} になることが知られている．この周期の中では乱数性の高い鍵ストリームとして利用できるため，高い安全性が期待できる．

効率　OFB は ECB や CBC と同等の処理効率で暗号化，復号処理を行なうことができる．

エラー伝播　暗号文における 1 ビットのエラーは，対応する平文ビットの反転を起こす．しかし，それ以降のエラー伝播などの影響はない．

ただし，同期ずれについては (ECB と同様，ブロック長単位の同期ずれでない限り) 耐性がなく，同期ずれが起こるような場合には，別途再同期のメカニズムが必要である．

並列処理性など　暗号化，復号処理ともに，並列処理性はまったくない．

しかし，インターリービングによる並列処理が可能な運用方法が知られ，ANSI X3.106 や ISO 10116 などで記載されている．具体的には，独立な OFB モードを並列度数だけ飛ばしながらメッセージストリームを処理するものである．この場合，初期値も並列度数だけ用意せねばならず，それぞれが安全であるためには，ランダムかあるいは攻撃者に選択されないような nonce にする必要がある．

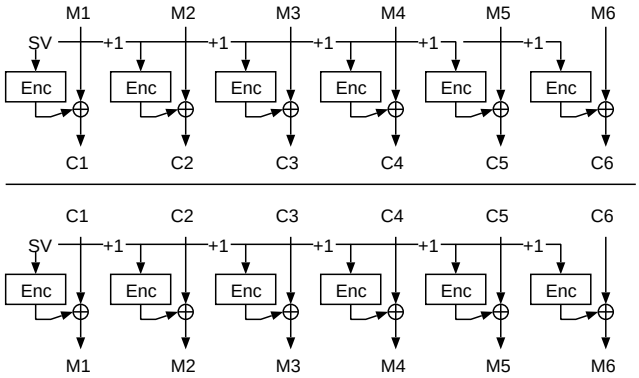


図 14: CTR モードの暗号化と復号のブロック図

復号　OFB モードでは，ブロック暗号の復号関数を利用しない．よって，OFB 暗号化，OFB 復号の両方の機能を実装する場合には，その実装コストは，CBC や ECB に比較して軽いことが期待できる．

k-OFB　FIPS-81 など古い利用モードの標準化では，OFB モードを k ブロック単位で行なうことも指摘されていた．これは，k-CFB と同様，n ビット単位でないデータを扱う場合への適用を考えたものであった．しかし，$k < n$ の場合，安全性の観点から大きな問題があることを理由に，OFB は $k = n$ として用いるべきとなった．よってこの古い仕様は今後使われるべきでない．

安全性の懸念には二通りある．第一には，脆弱な初期値の存在である．初期値に n ビットの 0 を与えて 1-OFB を実行すると約半数の鍵において 0 ばかり続く鍵系列が出力され，初期値に n ビット値 1 を与えて 1-OFB を実行すると約半数の鍵において 1 ばかり続く鍵系列が出力される [W02b]．また，第二の安全性の懸念として，$k < n$ の場合には，内部レジスタの更新関数が単射性を保証できなくなり，これが理由で周期の平均が $2^{n/2}$ ブロック程度となることが挙げられる．

5.5　CTR

仕様の概要　CTR(カウンタ) モードは，初期値のみに依存し逐次的に擬似乱数を生成しながら暗号化を行なう方法であり，任意のビット長の平文を処理できる [DH79, LRW00]．まず，平文を n ビット毎のブロックに分割 (それぞれを M_i とする) し，最後の端数の部分は端数ブロックとし

て扱う．開始値 SV を内部レジスタの初期値 R_1 とする．R_i をブロック暗号入力とし，暗号化処理の結果を H_i とする．これより暗号文ブロック $C_i = M_i \oplus H_i$ を生成する．次のブロックでは，内部レジスタ R を整数カウンタとして 1 数えあげる．

$$\begin{aligned} C_i &= M_i \oplus Enc_K(R_i), \\ R_{i+1} &= R_i + 1. \end{aligned}$$

復号はその逆関数である．

$$\begin{aligned} M_i &= C_i \oplus Enc_K(R_i), \\ R_{i+1} &= R_i + 1. \end{aligned}$$

ここで開始値とは，特殊な運用が必要な初期値である．CTR が安全な処理モードであるために，同一の鍵が用いられている間は常に異なるブロック暗号入力を与える必要がある．

CTR モードでは，内部状態の更新がカウンタであるため，システム要件から，カウンタの更新回数の限度などを知ることができる場合がある．このような情報を使いながら，うまく開始値を定義して，(同じ鍵のもとで) 複数の平文を安全に暗号化できるようにする．

具体的には，ひとつのメッセージ長が 32 ブロック未満で定義されるシステムでは，下位 5 ビットをカウンタ動作部分としてリザーブしておき，残り上位 $n-5$ ビットをメッセージ ID として固有な数字を埋め込む．こうすることにより，最大 2^{n-5} 個のメッセージを安全に処理できる[1]．

安全性　CTR モードの安全性については [BDJR97] で議論されている．該当の文献では (モードの名称は CTR でなく XOR であるが)，開始値が乱数の場合と，カウンタの場合との二種類について検討している．

前者，開始値が乱数の場合，ランダム関数を使ったスキームの安全性について，攻撃者のアドバンテージは

$$\mathbf{Adv}^{\mathrm{lr}}_{\mathrm{E}} \leq \mu(q-1)/(L \cdot 2^l),$$

となる．ここで t は攻撃者の計算時間，q は攻撃者の質問回数，l はランダム関数の入力長，L はランダム関数の出力長である．

さらに，擬似ランダム関数を使った，開始値が乱数の CTR モードについての安全性の評価結果も調べられており，l ビット入力-L ビット出力の

[1] 厳密には 2^{n-5} 個も暗号化してしまうと，別の情報が漏洩するため安全とはいえない．

$(t', q'; \epsilon')$-安全な擬似ランダム関数を使った場合，乱数開始値の CTR モードは $(t, q, \mu; \epsilon)$-安全である．ここで

$$(t, \mu, \epsilon) = (t' - c \cdot \frac{\mu}{L}(l + L), q'L, 2\epsilon' + \mu(q - 1)/(L \cdot 2^l)),$$

である．

また，カウンタを初期値にした CTR モードをランダム関数モデルといっしょに用いた場合，$\mathbf{Adv}^{\mathrm{lr}}_{\mathrm{E}} = 0$ となる．ここで攻撃者のパラメータとして，計算時間が最大 t，質問回数が最大 q，質問長が最大 $\mu < L2^l$ の場合を考える．

そして，擬似ランダム関数を使った，開始値がカウンタの CTR モードについては，l ビット入力-L ビット出力の $(t', q'; \epsilon')$-安全な擬似ランダム関数を使った場合，$(t, q, \mu; \epsilon)$-安全である．ここで

$$(t, \mu, \epsilon) = (t' - c \cdot \frac{\mu}{L}(l + L), \min(q'L, L2^l), 2\epsilon'),$$

である．

効率　CTR は ECB や CBC モードとほぼ同程度に効率的である．

エラー伝播　暗号文における 1 ビットのエラーは，対応する平文ビットの反転を起こす．しかし，それ以降のエラー伝播などの影響はない．

ただし，同期ずれについては (ECB と同様，ブロック長単位の同期ずれでない限り) 耐性がなく，同期ずれが起こるような場合には，別途再同期のメカニズムが必要である．

並列処理性など　暗号化復号ともに並列処理性が実現可能である．しかし，このためには，処理しているブロックが平文 (もしくは暗号文) の何ブロック目であるかという情報を処理系が知っている必要がある．従って，パイプライニングなどのようなメカニズムで，メッセージ (もしくは暗号文) を最初のブロックから処理する場合には問題とはならない．

同様に，何ブロック目のデータであるかがわかれば，処理順序不問性 (Out-of-order) 性も達成できる．すなわち，ブロック単位でデータが入れ替わったとしても，その順序いれかえをすることなく，到着した順序に復号処理に渡すことができる．もちろん，復号結果は，到着順序に応じて並べかえねば正しい平文には戻らない．

復号　CTR モードでは，ブロック暗号の復号関数を利用しない．よって，CTR 暗号化，CTR 復号の両方の機能を実装する場合には，その実装コストは，CBC や ECB に比較して軽いことが期待できる．

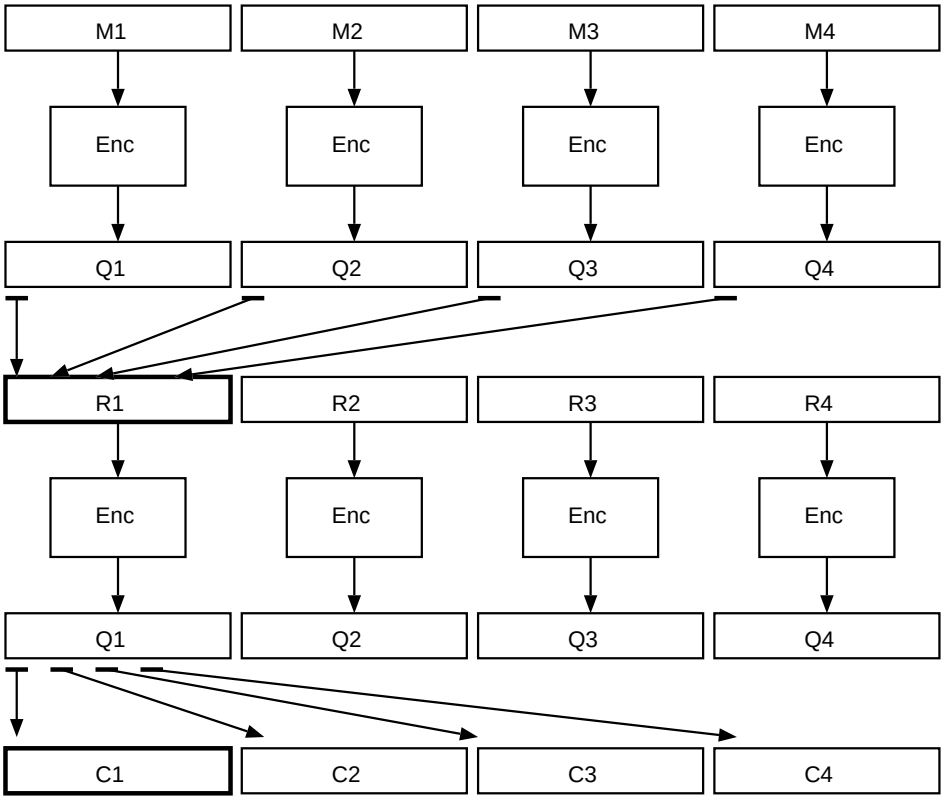


図 15: 2DEM モードの暗号化と復号のブロック図

5.6 2DEM

仕様の概要　この 2DEM (2D Encryption Mode, 二次元暗号化モード) の目的は，ECB の安全性の懸念，CBC の並列処理性の低さを克服することを目的に，主にバイトデータを二次元配列で解釈し，暗号化処理を行なうことを記述したものである [BA01].

具体的には，メッセージをまず ECB で処理したものを，バイト単位でインターリーブする．そうしてできたブロック列を再度 ECB で処理し，その結果を再度インターリーブして暗号文ブロック列とするものである．

5.7 ABC

仕様の概要　ABC (Accumulated Block Chaining, 累積ブロック連鎖) は，エラー伝播が最後まで続くような暗号利用モードとして AES 利用モードに提案された [K00]．しかし，提案は秘匿の目的のみであり，上記性質が

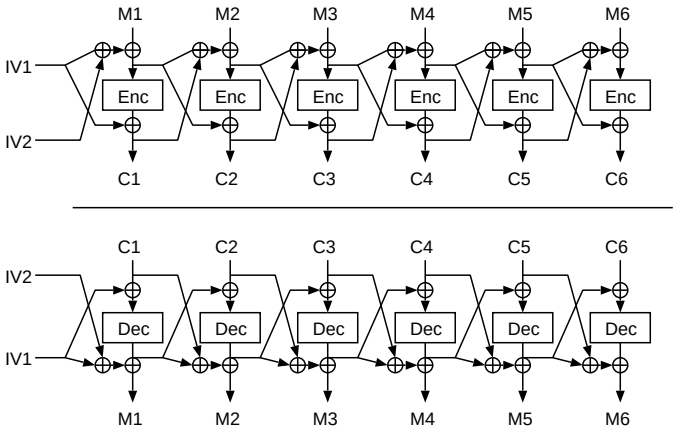


図 16: ABC モードの暗号化と復号のブロック図

暗号学的な意味のある安全性には特に関連していない．処理の流れを図 16 に示しておくが詳細な説明は省略する．

5.8 IGE

IGE (Infinite Garble Extension, 無限改竄拡張) は，もともと CBC と同じくらい古くに提案された利用モードである [C78]．AES の利用モードで，このモードに対する解析結果が発表され, メッセージ認証に対して安全でないことが示されている [GD00].

暗号化と復号の処理フローが同じである (つまり両方がそれぞれ上下対称) であるのは，なんらかの実装の利点があるかもしれないが，それ以上に，復号処理でブロック暗号の復号関数が必要となり，そうでない CFB, OFB, CTR などがより効率的である可能性が高い．ここでは詳細な記述は行なわない．

5.9 自己同期型利用モード

CFB モードは FIPS81 に掲載され，長い間使われ続けてきた．しかし，CFB モードが特徴とする自己同期性には一つの懸念があった．なるべく小さいデータ単位，例えば 1 ビットや 1 バイト単位などでの自己同期を行なうためには，それだけの処理負荷の増大を伴うことである．例えば，AES で 1-CFB を実行してしまうと，CBC モードの 128 倍もの処理が必

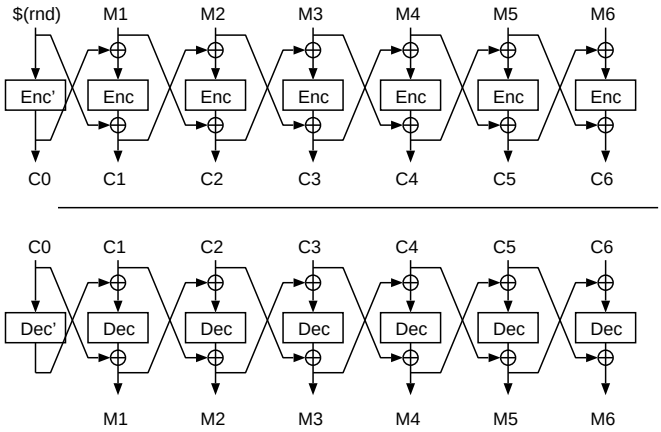


図 17: IGE モードの暗号化と復号のブロック図

要となる．しかしビット単位の自己同期が可能な唯一の標準利用モードであった．

しかし，効率の良い自己同期に関する一連の研究成果があり，標準化されるに至っていないものの，技術的に重要なものであるのでここに紹介する．

Maurer は自己同期に関する新しいアプローチとしてその設計手法と解析結果を発表した [M91]．この発表から遅れて，Jung, Ruland は [JR99] にて類似の手法を提案している．さらに，Alkassar, Geraldy, Pfitzmann, Sadeghi も同様な手法を提案している [AGPS02]．提案手法の主に共通する部分では，目的として任意の自己同期機能を実現しながらもその処理速度，厳密にはブロック暗号の呼び出し回数はなるべく ECB に近付けるものである．

具体的には，k-CFB を改良する方向で理解するとわかりやすい．ブロック暗号出力をこれまで捨てていたところをバッファとして動かせることにより効率化を行なっている．バッファが空になれば再度ブロック暗号処理を行ない新しい乱数列を充填する．

さらに同期回復のためのアイデアとして，暗号文パターンを監視し，特定のパターンが出現したところで，先述のバッファの残量を無視して，ブロック暗号を処理させ，バッファをフラッシュする．

これら二点のアイデアを使うことにより，同期が暗号文パターンで行なわれるため自己同期が実現でき，かつ，パターンサイズを適切に選ぶことでバッファから捨てる乱数長を減らすことができる．

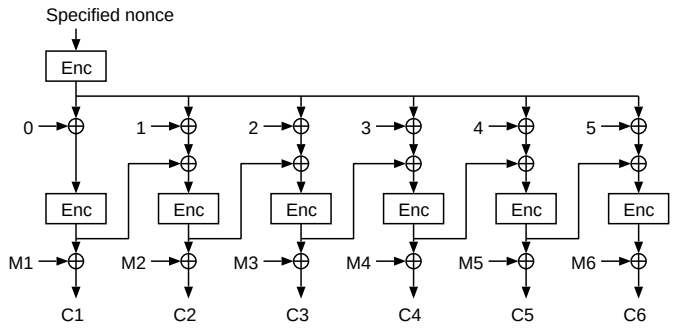


図 18: $f8$ モードの暗号化処理のブロック図

安全性に関する考察は [M91] でも行なわれているが，現在議論される証明可能安全性の観点からは，[AGPS02] に記載されている安全性の証明が参考になる．安全性に関する欠陥は今のところ見つかっていない．

また，処理速度の観点からの解析はさまざまな論文でなされており [H01a, H01b, H03a, JKRW01, AGPS02]，結果として，場合により CFB よりも非常に効率的な処理となっている．

5.10 $f8$ (3GPP)

3GPP では，ブロック暗号 KASUMI の利用モードとして二つの利用モード $f8$ と $f9$ を定義している [3GPPa]．それぞれ，秘匿，メッセージ認証に関する利用モードである．KASUMI の設計も含め，この標準化は，3GPP での利用を目的としており，モバイル端末と基地局間の無線区間の暗号化に特化している．従って，汎用目的にはあるべき性質などが棄却され，必要な目的に特化した方式であることを注意しておく．

$f8$ は, 仕様で定義された “nonce” 入力と鍵から鍵ストリームを生成する方法である．暗号化はこの鍵ストリームと入力ストリームとの排他的論理和をとることで行われる. 鍵ストリームの生成は，カウンタモードに CBC モードを組み合わせたようなものである．具体的には，鍵ストリームブロックを生成するために，ブロック暗号入力に “nonce” 値，カウンタ値，そして前ブロックの鍵ストリーム値全部の排他的論理和をとったものである．

これについては安全性の問題点はないように見える．また，処理効率も ECB 程度であり，復号には CFB などと同様，KASUMI の暗号化関数だけで処理が可能である．ただし，並列処理性能がない仕様となってい

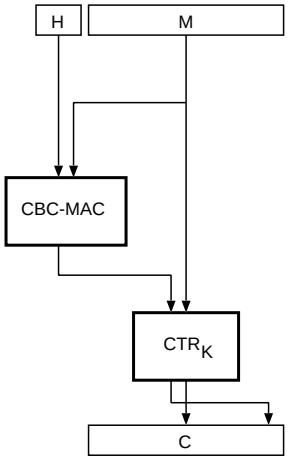


図 19: CCM モードの暗号化と復号のブロック図

る (が，これは必要ない実装だけで用いられるという理由で問題にはならない).

6 認証暗号に関する利用モード

この章では，これまで提案されてきた認証暗号に関する利用モードのうち，特に安全性の欠陥の見つかっていないものについて簡単に紹介する.

ここで,「本章にて扱わなかった認証暗号の利用モードはすべて安全性に欠陥がある」という意味ではないことに注意する.

6.1 CCM

仕様の概要 CCM (Counter with CBC-MAC)[WHF02] は CTR モードによる暗号化と CBC-MAC を組み合わせたメッセージ認証つき暗号化方式である. 具体的には, メッセージ，及び認証データに対して，CBC-MAC による MAC を生成し，MAC 処理で生成されたタグとメッセージを，CTR モードで暗号化する．CTR，CBC-MAC の両方に同じ秘密鍵を用いているので鍵のセットアップは 1 回である.

暗号処理や MAC 生成には，いくつかの詳細なパディングが記載されており，その一部に長さ情報が含まれる.

本方式は 2004 年 5 月に発行された NIST SP800-38C[SP800-38C] に規定されている.

安全性　CTR は秘匿の観点から安全性が証明されてはいるが，攻撃者の能力が，CTR で暗号化してくれる暗号化オラクルとの通信に限定されていることが重要である．CCM では，同じ暗号化オラクルでも暗号化処理が異なるため，攻撃者に得られる情報が異なる．よって従来の CTR モードの安全性証明は CCM の安全性とは独立と考えるべきである．

CCM についての安全性評価は Jonsson により [J02] で与えられている．ここでは，メッセージ認証，秘匿の二つについての安全性の議論が与えられている．

メッセージ認証に関する安全性評価では，一般的な改竄攻撃の定義を用いており，アドバンテージは攻撃者の改竄成功確率としている．攻撃者の能力として，1. 暗号化オラクルへの暗号化要求 (質問長上限 μ_E)，2. 改竄試行 (復号オラクルへ暗号文を質問し，その暗号文が有効か無効かの判定, 質問長上限 μ_F, 質問回数上限 q_F) の二つが許可されている．

ブロック暗号のブロック長を n，タグ長を t とすると，ブロック暗号を擬似ランダム関数で置き換えた場合の攻撃者のアドバンテージは

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CCM}} = \epsilon' + q_F \cdot 2^{-t} + (\mu_E + \mu_F)^2 \cdot 2^{-n-1},$$

となる．ここで，ϵ' は，擬似ランダム関数に関する安全性のパラメータであるが，質問回数などのその他のパラメータとの相関が示されておらず，不完全な記述である．

また，秘匿に関しては別の定義を用いている．この定義は Real-or-Random に類似するが，乱数の暗号文，ではなく，乱数そのものをオラクルは攻撃者に返す．あとは，アドバンテージの定義なども含めて，Real-or-Random と同じである．攻撃者は暗号化オラクルのみが利用可能であり，この場合擬似ランダム関数を使ったスキームに対する攻撃者のアドバンテージは

$$\mathbf{Adv}^{rr'}_{\mathrm{CCM}} = \epsilon' + (\mu_E)^2 \cdot 2^{-n-1},$$

となる．

効率　CCM は ECB や CBC モードの 2 倍のブロック暗号呼び出しを行なうため，処理量も ECB のそれに比べて約 2 倍である．

ただし，実質的に CTR モードと CBC-MAC の組合せであり，データサイズが (処理系が扱えるメモリサイズに対して) 大きい場合には注意が必要である．例えば，ストリーミングデータなどへの処理には，CCM として，内部で呼び出す CTR の処理と CBC-MAC の処理，両方を交互に行なうような実装を行なわないと，処理が不可能となる．この場合，中間データの保持のためにいくらかの必要レジスタサイズの増加が考えられる．

その他の懸念とされる事項が技術文書として公開されている．後の議論の章を参照頂きたい．

並列処理性など　まず，CCM 処理中の CTR 処理と，CBC-MAC 処理は並列処理が可能である．従って適切な実装により 2 並列度までは簡単に達成できる．しかし，CBC-MAC には並列処理機能がないため，それ以上の並列処理は CTR のみに適用可能となる．

これは復号処理についても同じことがいえる．

復号　CCM モードでは，ブロック暗号の復号関数を利用しない．よって，CCM 暗号化，CCM 復号の両方の機能を実装する場合には，その実装コストは，CBC や ECB に比較して軽いことが期待できる．

議論　CCM は IEEE 802.11 の標準ドラフトなど，いくつかの業界標準方式として採用されている実績がある [WHR02] このモードの利用に関する注意を記した文書が Rogaway, Wagner らにより公開されている [RW03]. 主に効率に関するコメントと安全性に関するコメントであるが，安全性は上記 [J02] の結果を否定するものではなく，CCM の NIST への提案文書 [WHF02] における安全性の主張に根拠がなく，かつ誤りと思われる宣言がいくつかある，という指摘に留まっている．

[RW03] で指摘する効率に関する注意点は次の 3 点である．

1. オンラインアルゴリズムでない．
2. ワード境界がずれる可能性がある．
3. 固定ヘッダ情報に対しての事前計算ができない．

その他，仕様が複雑であることや，タグ長 (改竄検知に関する安全性レベル) の柔軟性から考えうる安全性への懸念などが示されている．

これらを指摘した [RW03] では，CCM の代替として，EAX の利用を提案している．

6.2　CWC

仕様の概要　CWC (Carter Wegman with Counter) モードは，CTR モードの暗号化と，Universal hash (汎用ハッシュ) による MAC 生成とを利用したメッセージ認証つき暗号化方式である [KVW03].

具体的には，メッセージに対して CTR モードで一度暗号文を生成し，その暗号文に対して (暗号化されない付加データの入力を許して)MAC をつけるというものである．

MAC 生成は，universal hash という性質をもつ特殊な (パラメータつきの) ハッシュ関数を使って暗号文のハッシュ値を生成し，さらにこれを使い捨て的な乱数 (ただし，真の乱数ではなく，仕様で定義された計算方法で求められる，攻撃者には計算できない値) でマスクして暗号文に添付するものである．

本方式は NIST の策定している AES 利用モードへ提案された利用モードである [KVW03].

安全性　提案の文書では，128 ビットブロック暗号に限定した安全性評価を行なっている．

メッセージ認証については，内部で用いるブロック暗号を擬似ランダム関数に置き換えた時 (それに対する攻撃者のアドバンテージを ϵ' と定義した時)，MAC 偽造を目的とした攻撃者のアドバンテージは以下のようになる．

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CWC}} \leq \epsilon' + (\mu_M + \mu_A)/2^{133} + 2^{-125} + 2^{-t}.$$

ここで，μ_M, μ_A はそれぞれ，メッセージ，付加情報の長さの上限であり，t はタグ長に相当するアルゴリズムの安全性のパラメータの一つである．

また，内部で用いるブロック暗号を擬似ランダム置換とした場合には，質問回数が最大 $q-1$，オラクルへの質問長が最大 μ である時，改竄を行なう攻撃者のアドバンテージは以下のようになる．

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CWC}} \leq \epsilon' + (\mu/128 + 3q + 1)^2/2^{129} + (\mu_M + \mu_A)/2^{133} + 2^{-125} + 2^{-t}.$$

秘匿については，暗号文が乱数との識別できる/できないという定義で議論している．具体的な評価では，内部で用いるブロック暗号を擬似ランダム関数に置き換えた時 (それに対する攻撃者のアドバンテージを ϵ' と定義した時)，暗号文を乱数と区別する攻撃者のアドバンテージは以下のようになる．

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{CWC}} \leq \epsilon'.$$

また，内部で用いるブロック暗号を擬似ランダム置換とした場合には，質問回数が最大 $q-1$，オラクルへの質問長が最大 μ である時，改竄を行なう攻撃者のアドバンテージは以下のようになる．

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{CWC}} \leq \epsilon' + (\mu/128 + 3q + 1)^2/2^{129}.$$

効率　この利用モードは効率の評価がやや困難である．処理はCTRモードの処理とuniversal hashの計算の部分が大部であるが，後者がブロック暗号による処理でないもののそれ相応の処理となるため，処理するプラットフォームや開発に用いる記述言語などによりuniversal hashの計算の効率が大きく変化すると考えられる．

少なくともこれまでの利用モードには珍しい(秘密情報に依存した)算術乗算演算があるため，実装には注意が必要な場合がある．

並列処理性など　universal hashの処理はCTRモードの結果を用いるため，安直に実装してしまうとこれらの並列処理性がないような実装に陥る可能性がある．しかしながら，CTRモードの処理の最後のデータがMACの最初の処理に用いられるものではないので，仕様書から技術を十分読みとれば，CTRとuniversal hashとの両方の処理を交互に処理するような実装が可能である．

CTR自身は並列処理可能である一方，universal hashの並列処理には，冪乗計算を並列に行なうための工夫が必要である．そのための概要は示してあるが，一般のエンジニアがこれらの文面から並列処理を実現するには別の技術解説文書が必要である．

復号　CWCモードでは，ブロック暗号の復号関数を利用しない．よって，CWC暗号化，CWC復号の両方の機能を実装する場合には(もちろん，冪乗演算のコストが新たに必要だが)，CBCやECBに比較して軽いことが期待できる．

6.3　EAX

仕様の概要　EAX (A Conventional Authenticated-Encryption Mode)は，CTRモードとOMAC[IK03a]を組み合わせた利用モードである[BRW03]．機能としては，入力としてメッセージ，nonce，ヘッダ情報があり，暗号化することにより，メッセージの情報が秘匿されることが保証され，かつメッセージとヘッダ情報の認証が復号時に行なわれる．

具体的な処理としては，以下のような処理となる．メッセージは，nonceから生成された撹拌nonce N を開始値としてCTRモードにより暗号化する．この結果を暗号文とする．そしてN，暗号文のMAC，ヘッダ情報のMACとの排他的論理和をとり，その結果をタグとするものである．

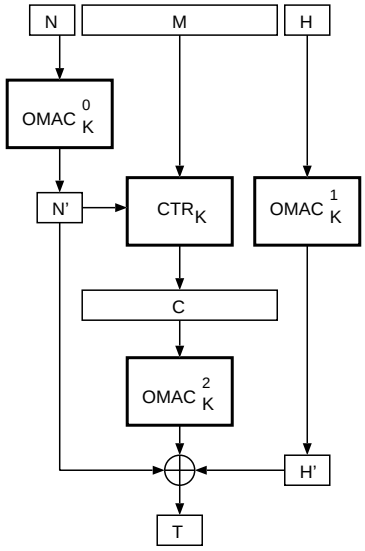


図 20: EAX モードの暗号化の処理を示すブロック図

安全性　証明可能安全性をもつとされているが，その証明についてはまだ公開されていない．近い将来公開される予定である．

効率　処理効率はヘッダ部分の処理と，秘匿するメッセージ部分の処理との重さが異なる．秘匿するメッセージ部分に対しては ECB の二倍必要であるが，ヘッダについては ECB と同等の処理速度である．

並列処理性など　暗号化や MAC 生成など，処理の本質となる部分が 3 つあるため，並列処理できる/できないという表現では説明次第ではあいまいになる．ここを整理しながら説明する．

ヘッダ部分への処理は他とはほぼ独立であり，ここは切り離して並列度に数えることができる．

メッセージについては，CTR と OMAC が直列に並んでいるため，それ自身では並列処理はできないように見える．しかしながら，CTR が処理した結果である暗号文が生成されれば OMAC 処理は開始できるので多少の遅れをもって並列処理可能である．

また，メッセージ長が長い場合には，CTR と OMAC を同時に動かす必要があるため，そのための実装には注意と工夫が必要である．

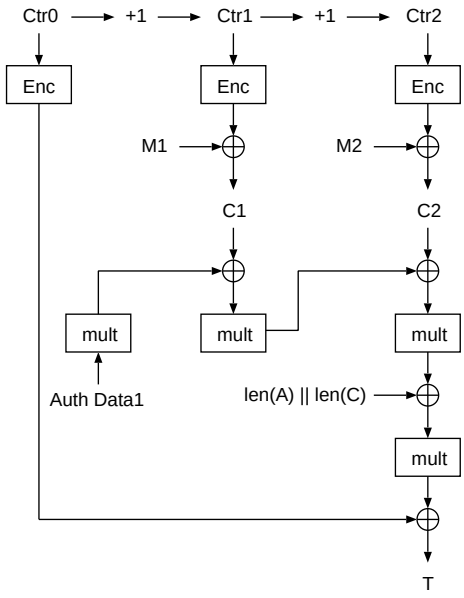


図 21: GCM モードの暗号化処理を示すブロック図

復号　EAX モードでは，ブロック暗号の復号関数を利用しない．よって，EAX 暗号化，EAX 復号の両方の機能を実装する場合には，その実装コストは，CBC や ECB に比較して軽いことが期待できる．

6.4　GCM

仕様の概要　GCM (Galois/Counter Mode) モードは、CTR モードの暗号化と universal hash 関数による MAC 生成を組み合わせたメッセージ認証つき暗号化方式である [MV05].

具体的には, メッセージに対して CTR モードで暗号文を生成し, その暗号文と, 暗号化はされない認証対象のメッセージに関して, GF(2^{128}) 上で定義された universal hash 関数により認証タグを計算し, 暗号文に付加する.

安全性　文献 [MV04] にて安全性評価が行なわれている．

メッセージ認証については，内部で用いるブロック暗号を擬似ランダム関数に置き換えた時 (それに対する攻撃者のアドバンテージを ϵ' と定義した時)，ブロック長を w, 質問回数が最大 q，オラクルへの質問長が最大

l_P, それぞれの質問について $\mathrm{len}(C)+\mathrm{len}(A) \le l$, $\mathrm{len}(IV) \le l_{IV}$ である時, MAC 偽造を目的とした攻撃者のアドバンテージは以下のようになる.

$$\begin{aligned}\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{GCM}} \quad \le \quad & \epsilon' + (l_P/w + 2q)^2 2^{-w-1} \\ & + q((l_P/w + 2q + 1)\lceil l_{IV}/w + 1\rceil 2^{1-w} + \lceil l/w + 1\rceil 2^{-t})\end{aligned}$$

秘匿については, 暗号文が乱数との識別できる/できないという定義で議論している. 具体的な評価では, 内部で用いるブロック暗号を擬似ランダム関数に置き換えた時, 攻撃者がブロック暗号と擬似ランダム関数を識別できるアドバンテージを ϵ' と定義した時, 暗号文を乱数と区別する攻撃者のアドバンテージは以下のようになる.

$$\begin{aligned}\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{GCM}} \quad \le \quad & \epsilon' + (l_P/w + 2q)^2 2^{-w-1} \\ & + q((l_P/w + 2q)\lceil l_{IV}/w + 1\rceil 2^{1-w} + \lceil l/w + 1\rceil 2^{-t})\end{aligned}$$

効率 文献 [MV04] における GCM のソフトウェア実装では, GF(2^{128}) 上の乗算を 256B, 4KB, 64KB のテーブル参照を用いて実装しており, テーブルサイズが大きいほど高速である. また、ハードウェア実装においても GF(2^{128}) 上の乗算は CWC で利用されている整数乗算に比べて極めて小型かつ高速に実装することができ, GCM はソフトウェア実装, ハードウェア実装ともに CWC に対し優位性をもっている.

並列処理性など universal hash 関数の積和演算は基本的に逐次処理であるが, GF(2^{128}) 上の乗算は AES の暗号化処理に対して非常に軽いため, CTR モードの高速性を犠牲にすることはほとんどない.

復号 GCM モードでは, ブロック暗号の復号関数を利用しない. よって, GCM 暗号化, GCM 復号の両方の機能を実装する場合には (もちろん, GF(2^{128}) 上乗算のコストが新たに必要だが AES の暗号化処理に対して非常に軽いため), CBC や ECB に比較して軽いことが期待できる.

6.5 IACBC/XCBC

仕様の概要 IACBC[J01, J00], XCBC[GD01a, GD01b] ともに, CBC モードにおいてブロック暗号の出力をマスクすることで本質的なメッセージ認証の安全性を与えた利用モードである.

IACBC について説明する. 処理するメッセージ長を m ブロックとすると, 第一の鍵と初期値 (乱数) から, $\lceil \log_2 m \rceil$ ブロックのマスクの種 W_i

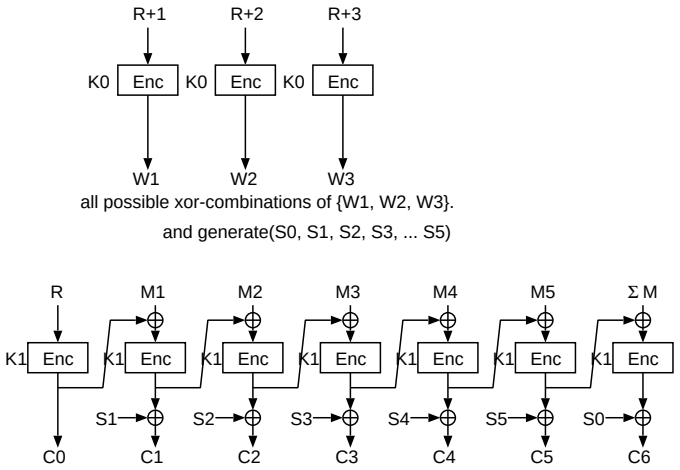


図 22: IACBC モードの暗号化処理を示すブロック図

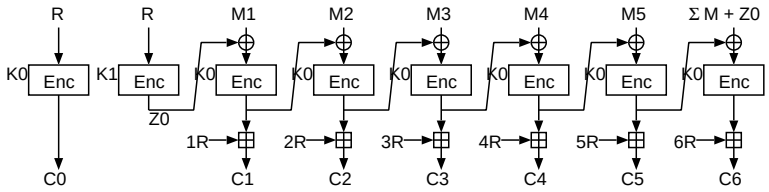


図 23: XCBC モードの暗号化処理を示すブロック図

を生成する．この W_i から約 m ブロック分の pairwise independent(ペア単位では独立) なブロック列 S_i を生成する ($\log_2 m$ 個の要素を含む集合から，可能な要素の組み合わせの数は $2^{\log_2 m} = m$ 通り)．

これら W_i ならびに S_i はメッセージ到着と同時に必要な時逐次的に生成が可能であるため，これらの処理はオンライン処理性を崩すことはない．

これら S_i 列をそれぞれブロック暗号の出力にマスクしながら第二の鍵で CBC モードのようなブロック連鎖を伴いながら暗号文を出力することが，スキームの主要部分となる．最後のブロックはメッセージのチェックサムとその暗号化のための末端処理がある．

一方，XCBC は，初期値である秘密乱数 R と二つの鍵から C_0,Z_0 を生成する．そして IACBC モードでいうところの S_i 列は，整数倍の R となり，ブロック毎に整数乗算 (おそらく 128 ビット幅) を行なう．暗号文の生成は，出力と S_i 列との算術加算の結果である．

仕様の定義を厳密に記すと，XCBC は暗号化を行なうものであり，秘

匿のみを保持する．このモードを使って，平文に特殊な秘密パディングを施したもの (仕様書では，その一つを XCBC-XOR と呼んでいる) が認証暗号の機能を達成することができる．

安全性　IACBC, XCBC ともに，近年の共通鍵暗号の安全性に関する議論を踏まえた安全性の証明を示している．

XCBC/XCBC-XOR についても秘匿とメッセージ認証両方の観点からの攻撃者のアドバンテージの上限を与えている．XCBC が $(q', t'; \epsilon')$- 安全な擬似ランダム関数を用いているとすると，初期値が乱数である XCBC は Left-or-Right に関する秘匿の意味で $(q, t, \mu; \epsilon)$-安全である．ただし

$$(t, \mu, \epsilon) = (t' - c\mu, q \cdot n, 2\epsilon' + (\mu^2/n^2 - \mu/n)2^{-n}).$$

XCBC-XOR に対するメッセージ認証に関する安全性として攻撃者の改竄成功確率の上限を与えている．(q', t', ϵ') を秘匿の場合の定義と同じとして，

$$\begin{aligned}\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{XCBC}} \quad \leq \quad & \epsilon + \frac{\mu_v(\mu_v - n)}{n^2 2^{n+1}} + \frac{q_e(q_e - 1)}{2^{n+1}} + \frac{(q_e + 1)\mu_v}{n2^n} \\ & + \frac{\mu_v}{n2^{n+1}}(\log_2 \frac{\mu_v}{l} + 3) + \frac{q_v \mu_e}{n2^n}(\log_2 \frac{\mu_e}{n} + 3).\end{aligned}$$

効率　メッセージ認証つき暗号化方式のスキームでは効率的な両方式である．IACBC は，メッセージのブロック数 m に対して $m + \log m$ 程度のブロック暗号呼び出しに加えて排他的論理和を基本とした補助演算が含まれる．メッセージ長が大きくなると，CBC や ECB に対する負荷処理は割合としてさほど大きくなくなる．

XCBC-XOR ではブロック暗号の呼び出し回数は ECB, CBC とほぼ同じである $m + 3$ 回程度の処理を行なうが，マスクの生成，ならびにマスク処理のために，算術加算 (また実装によっては算術乗算) の処理が含まれる．これらは 128 ビットのレジスタで処理される演算である．

並列処理性など　並列処理性については CBC モードと同様である．暗号化においては，主要な演算部分の並列処理性は達成できない．ただし，XCBC では，CBC で適用されたようにインターリーブする手法が記述されている．

復号　復号では，ブロック暗号プリミティブの暗号化演算，復号演算両方を利用するため，復号処理では両方を同時に実装する必要がある．

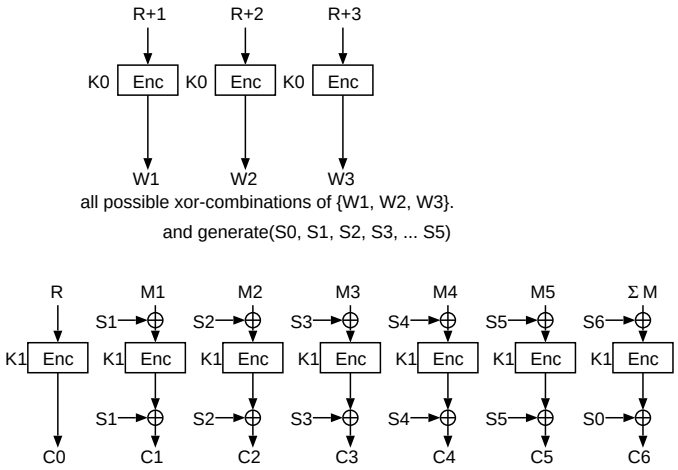


図 24: IAPM モードの暗号化処理を示すブロック図

6.6 IAPM/OCB

仕様の概要 IAPM[J00, J01]，OCB[RBBK01a, RBBK01b] ともにブロック連鎖のない暗号処理である．主要な撹拌部分は，ECB モードにおいて，ブロック暗号の入出力部で，ブロック位置に応じた秘密マスクを行なうことである．

IAPM では，これら秘密マスクを $\log m$ ブロックの W_i 列の (擬似) 乱数ブロックから m ブロックの pairwise independent ブロックを生成している．一方，OCB では，秘密鍵と nonce から生成する (擬似) 乱数 2 ブロック，L, R から S_i を生成している．具体的には，ブロック位置 i 番目には $\gamma_i L \oplus R$ を生成するような，線形式によるブロック列の生成である．

OCB では，これら線形列の生成が隣のブロックに対して排他的論理和を一度行なうだけでよいように gray code (自然数の並べかえであって，隣り合う整数どうしのバイナリ表現によるハミング距離が常に 1 であるような順列) の技術を使って生成している．

IAPM に比べて OCB は，後で提案されたこともあり様々な観点から改良と呼ぶことができる特徴がいくつもある．OCB では秘密鍵を 1 個利用する (IAPM は 2 個)．OCB では初期値として nonce であればよい (IAPM は乱数)．OCB では，ブロック暗号の呼び出しは m 回程度である (IAPM は約 $m + \log m$ 回)．OCB では端数処理の定義があり，パディングが本質的理由となる暗号文の増加が最小限に押えられている．

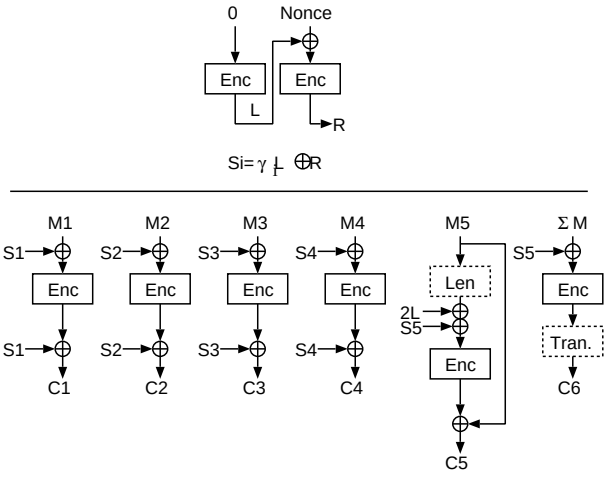


図 25: OCB モードの暗号化処理を示すブロック図

安全性　ランダム関数を内部で用いる OCB の，メッセージ認証に関するアドバンテージは以下のように示されている．

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{OCB}} \leq 1.5(\mu_e + 3q + 5\mu_v + 11)^2/2^n + 2^{-\tau}.$$

ここで，μ_e は攻撃者の q 回の暗号化オラクルへの質問で累積するブロック数，μ_v は復号への試行の数，τ はタグ長である．

一方，秘匿に関する安全性としては，暗号文-乱数不可識別性による秘匿の定義により評価を行なっており，この場合, 各パラメータ，変数はメッセージ認証と同じとして，ランダム関数を用いた OCB の秘匿に関する攻撃者のアドバンテージは以下のようになる．

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathrm{OCB}} \leq 1.5(\mu_e + 2q + 3)^2/2^n.$$

並列処理性など　これらの利用モードはメッセージ認証可能な暗号方式であってかつ，暗号化，復号ともに並列処理性があることが大きな特徴である．ただし並列処理を行なう場合にも処理するブロックのブロック位置はプロセッサが知っておく必要がある．

復号　復号処理においても，ブロック暗号の暗号化関数が必要であるので，復号デバイスには両方を実装する必要がある．

6.7 k-PCFB

仕様の概要 k-CFB に変更を加えた利用モードである [H01c]．従来の k-CFB モードは $k < n$ の時，内部レジスタの更新に前の情報の内部レジスタ値を使っていた．このモードでは，ブロック暗号処理の出力の一部と暗号文を使って内部レジスタを更新する．

この利用モードとして，特殊な平文 (前後に平文長がパディングされたもの) を使うことによりメッセージ認証も達成できると提案されている．

安全性 秘匿に関しては CFB の拡張の一種であり問題ないと考える．

メッセージ認証については特に安全性に関する技術的根拠が記載されていない．また，提案者自身の評価も公開されていないため，あまり研究者の興味を集めたモードでない．

実際に，スキームへの改竄が可能であることは簡単に示すことができる．$k = n$ の時は CFB と等価であるため，ブロック単位のデータ欠損にはある程度の遅れを伴うもののすぐに同期が回復する．メッセージ中に長さ情報として読みとれる部分を二箇所，(その値で指定される) 適切な幅で挿入しておけば，データ欠損時にもその改竄が検出できない．結果として部分だけを切りとる攻撃が既知平文攻撃により可能である．

7 ディスクセクタ向け暗号利用モード

IEEE の Security in Storage Working Group では，ハードディスクなどをセクター単位で暗号化することを直接的な応用先として，利用モードとその運用の観点から技術調査，標準化を行なっている [WWW1]．

この標準化における技術要件は，平文が暗号文の長さから変化することがない暗号化であって，かつ暗号文に対するいかなる改竄によっても平文が撹拌されていることを保証するものである．

現在標準化が策定中であり，提案された利用モードに関する情報は多くない．本報告では，提案方式を簡単に説明する．

7.1 EMD

EMD モードは Tweak 入力 (補助的な入力であって秘密情報ではないブロック暗号に対するパラメータのようなもの) をとりながら 2 パス処理により大きなブロックを撹拌する利用モードである [R02b].

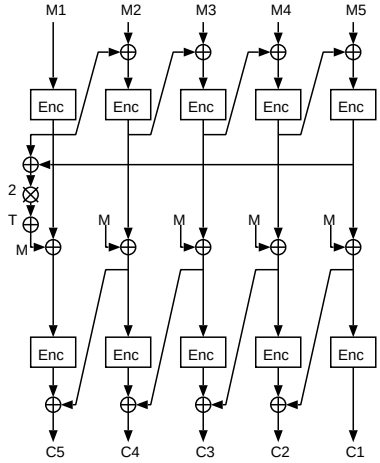


図 26: EMD モードの暗号化処理を示すブロック図

この利用モードは，証明にミスがあり，かつ効率的に PRP と識別可能であることが示された [J03].

7.2 EME

EME は，EMD の並列化可能なスキームに改良したものである．オリジナルのスキームは [R02b] で提案されたがこれについては攻撃方法が [J03] によって示された．これを改良した EME モードが [HR03a] で提案された．暗号化処理を図 27 に示す.

7.3 CMC

CMC は，EMD,EME モード [R02b, HR03a] に対して安全性の観点から改良した利用モードである [HR03b]．Tweak データ T と鍵 K をパラメータとして多ブロックを暗号化する方法である．構造は 3 層からなり，最初の層で CBC モードの暗号化のような処理を行なう．次の層では，その結果の一部を加工したデータを各ブロックにマスクする．最後の層では平文ブロックの逆の順序で CBC モードの復号のような処理を行なう.

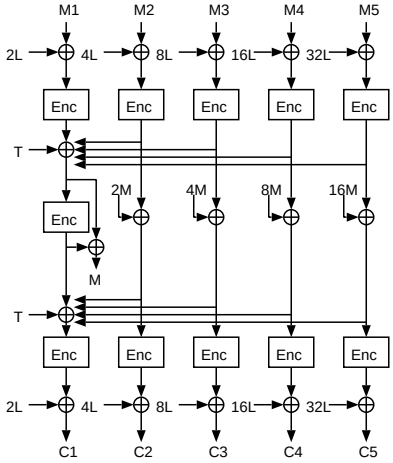


図 27: EME モードの暗号化の処理を示すブロック図

7.4 LRW

LRW は [LR02] で提案された Tweakable ブロック暗号である．これはディスクセクタ暗号の特にメッセージが 16 バイトと短いものに向けての操作モードとして提案された．

入力として鍵 K と Tweak I がある．鍵はさらにブロック暗号に使うための鍵 K_1(ブロック暗号の鍵長) とマスク鍵を生成するための K_2(ブロック長) からなる．暗号化処理では，有限体乗算により $K_2 \otimes I$ からマスク値 T を生成し，ブロック暗号処理前と処理後にそれぞれ T をマスクする．この処理を図 29 に示す．

7.5 NR

NR モードはブロック暗号の ECB モードに処理を加えた暗号処理方式である [NR03]．入力出力にそれぞれ拡張 Feistel 構造で構成される線形変換を導入する．入力側，出力側で二種類の線形変換を定義するが，各々の変換の内部では，3 つの universal hash を用いている．

8 認証に関する利用モード

本章では，認証に関する利用モードについて述べる．EMAC, RMAC, XCBC, TMAC, OMAC は CBC MAC の変形であり，XOR MAC, XECB

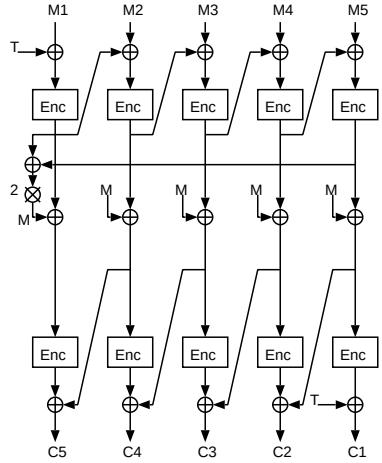


図 28: CMC モードの暗号化の処理を示すブロック図

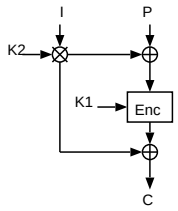


図 29: LRW モードの暗号化の処理を示すブロック図

MAC, PMAC は並列計算可能な方式である．また，$f9$ は 3GPP [3GPPa, 3GPPb] により標準化されている方式である．

以上の方式がブロック暗号を用いて構成されているのに対して，NMAC, HMAC はハッシュ関数を用いて構成される方式である．

8.1 CBC MAC

CBC MAC には，パディングの方法や，最終ブロックの処理など，いくつもの仕様がある．次に述べる仕様は，最も単純なものである．

方式 CBC MAC はブロック暗号 E, タグ長 τ, (メッセージ長を規定する) 定数 m をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならな

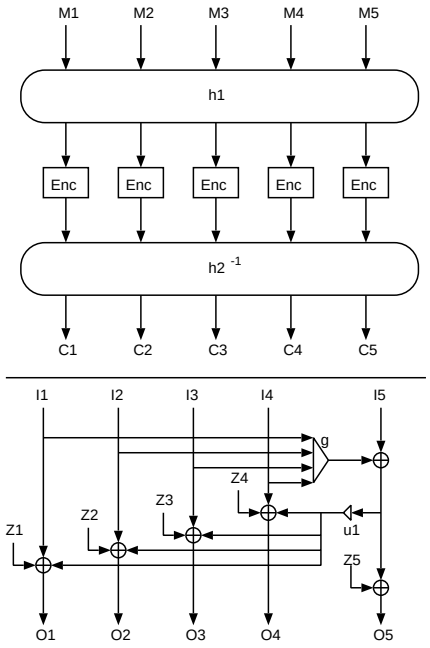


図 30: NR モードの暗号化処理 (上) と内部の universal hash の構成 (下) を示すブロック図

い．これらのパラメータを用いた CBC MAC を CBC$[E, \tau, m]$ と表記する．CBC$[E, \tau, m] =$ (CBC-$\mathcal{K}$, CBC-$\mathcal{G}$, CBC-$\mathcal{V}$) の鍵生成アルゴリズム CBC-$\mathcal{K}$, タグ生成アルゴリズム CBC-$\mathcal{G}$, 確認アルゴリズム CBC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム CBC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K \xleftarrow{R} \mathcal{K}_E$$

  を出力する．

- タグ生成アルゴリズム CBC-$\mathcal{G} : \mathcal{K}_E \times \{0,1\}^{mn} \to \{0,1\}^{\tau}$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^{mn}$, タグ空間は $\{0,1\}^{\tau}$ である．すなわち，鍵 $K \in \mathcal{K}_E$ とメッセージ $M \in \{0,1\}^{mn}$ を入力とし，タグ

  $$T = \text{CBC-}\mathcal{G}_K(M) \in \{0,1\}^{\tau}$$

  を出力する．図 31, 図 32 にあるように動作する．図 32 において，trunc は n ビットの入力のうち，左 τ ビットを出力する．

Algorithm CBC-$\mathcal{G}_K(M)$
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow E_K(X[i])$
$T \leftarrow$ the left most τ bits of $Y[m]$
return T

図 31: CBC MAC のタグ生成アルゴリズム CBC-$\mathcal{G}_K(\cdot)$.

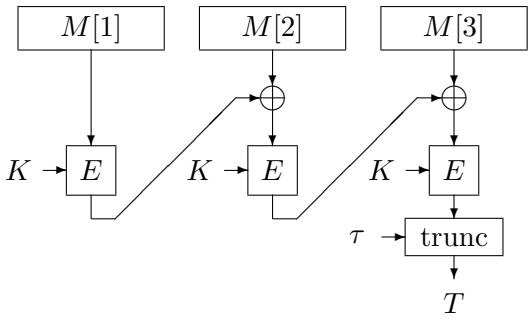


図 32: $M = M[1]M[2]M[3]$ の場合の CBC-$\mathcal{G}_K(M)$ の動作.

- 確認アルゴリズム CBC-$\mathcal{V}$: $\mathcal{K}_E \times \{0,1\}^{mn} \times \{0,1\}^{\tau} \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^{mn}$, タグ $T \in \{0,1\}^{\tau}$ を入力とし，

$$\text{accept or reject} = \text{CBC-}\mathcal{V}_K(M,T)$$

を出力する．図 33 にあるように動作する．

Algorithm CBC-$\mathcal{V}_K(M,T)$
$T' \leftarrow$ CBC-$\mathcal{G}_K(M)$
if $T = T'$ **then return** accept
else return reject

図 33: CBC MAC の確認アルゴリズム CBC-$\mathcal{V}_K(\cdot,\cdot)$.

安全性 Bellare, Kilian, Rogaway [BKR00] と Bellare, Pietrzak, Rogaway [BPR05] とにより安全性が解析されている．ブロック暗号 E が安全な擬似ランダム置換族であれば，$\text{CBC}[E,\tau,m] = (\mathcal{K}_{\text{CBC}}, \mathcal{G}_{\text{CBC}}, \mathcal{V}_{\text{CBC}})$ は，偽造不可能性の意味で安全な MAC であることが示されている．(タグ生成アルゴリズムは決定的なので，この場合，弱偽造不可能性と強偽造不可能性は同一の定義になる．) 上記の解析結果より，以下の定理が成立することがわかる．

定理 8.1. n, m, t, q を非負整数とし，$m^2 \le 2^{n-2}$ とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\mathbf{Adv}^{mac}_{CBC[E,\tau,m]}(t,q,mq) \le \mathbf{Adv}^{prp}_{E}(t',q') + \frac{m^2q^2}{2^{n+1}} + \frac{12mq^2}{2^n} + \frac{8m^4q^2}{2^{2n}} + \frac{1}{2^{\tau}}$$

である．ただし，$q' = mq$, $t' = t + O(nmq)$ であり，質問の長さはブロック単位である．

直感的に，定理 8.1 は，以下のことを示している： 実行時間 t, 高々 q 回の質問の後,

$$\mathbf{Adv}^{\text{mac}}_{\text{CBC}[E,\tau,m]}(A) = \epsilon$$

で偽造に成功する敵が存在すると仮定する．このとき，実行時間 $t' = t + O(nmq)$, 高々 $q' = mq$ 回の質問の後,

$$\mathbf{Adv}^{\text{prp}}_{E}(B) \ge \epsilon - \frac{m^2q^2}{2^{n+1}} - \frac{12mq^2}{2^n} - \frac{8m^4q^2}{2^{2n}} - \frac{1}{2^{\tau}}$$

なる敵 B が存在する．

しかし，上記の定理はメッセージ空間が，ある定数 m に対し，$\{0,1\}^{mn}$ となっていなければならない．そうでない場合，特に可変長のメッセージ空間 (たとえば $(\{0,1\}^n)^+$) に対しては，CBC MAC は安全な MAC ではなくなる．たとえば，図 34 の敵 A は，メッセージ空間を $(\{0,1\}^n)^+$ とした CBC MAC を偽造不可能性の意味で破る敵である．また，その成功確率は 1 である．

Algorithm $A^{\text{CBC-}\mathcal{G}_K(\cdot)}$
$M \leftarrow 0^n$
$T \leftarrow \text{CBC-}\mathcal{G}_K(M)$
$M' \leftarrow (M, T)$
return (M', T)

図 34: $A^{\text{CBC-}\mathcal{G}_K(\cdot)}$.

効率　CBC MAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K \in \mathcal{K}_E$ 一つのみである．
- ブロック暗号鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$(|M|/n)$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況　広範囲にわたって標準化されている．FIPS 113, ISO 9797, ISO 8731-1, ISO 9807, ANSI X9.9, ANSI X9.19 に含まれている．

なお，メッセージ長の問題を解決するために，パディング，最終出力の前に暗号化を施すなど，いくつかの変形がある．正確な仕様については，各標準の文書を参照されたい．

8.2 EMAC

方式 EMAC はブロック暗号 E とタグ長 τ をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならない．これらのパラメータを用いた EMAC を EMAC$[E,\tau]$ と表記する．EMAC$[E,\tau]$ = (EMAC-$\mathcal{K}$, EMAC-$\mathcal{G}$, EMAC-$\mathcal{V}$) の鍵生成アルゴリズム EMAC-$\mathcal{K}$, タグ生成アルゴリズム EMAC-$\mathcal{G}$, 確認アルゴリズム EMAC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム EMAC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K_1 \xleftarrow{R} \mathcal{K}_E$$

  と

  $$K_2 \xleftarrow{R} \mathcal{K}_E$$

  を出力する．

- タグ生成アルゴリズム EMAC-$\mathcal{G}$: $(\mathcal{K}_E)^2 \times (\{0,1\}^n)^+ \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $(\mathcal{K}_E)^2$, メッセージ空間は $(\{0,1\}^n)^+$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K_1, K_2 \in \mathcal{K}_E$ とメッセージ $M \in (\{0,1\}^n)^+$ を入力とし，タグ

  $$T = \text{EMAC-}\mathcal{G}_{K_1,K_2}(M) \in \{0,1\}^\tau$$

  を出力する．図 35, 図 36 にあるように動作する．

- 確認アルゴリズム EMAC-$\mathcal{V}$: $(\mathcal{K}_E)^2 \times (\{0,1\}^n)^+ \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K_1, K_2 \in \mathcal{K}_E$, メッセージ $M \in (\{0,1\}^n)^+$, タグ $T \in \{0,1\}^\tau$ を入力とし，

  $$\text{accept or reject} = \text{EMAC-}\mathcal{V}_{K_1,K_2}(M,T)$$

  を出力する．図 37 にあるように動作する．

安全性 Petrank, Rackoff により，安全性が解析されている [PR00]. ブロック暗号 E が安全な擬似ランダム置換族であれば，EMAC$[E,\tau]$ は，偽造不可能性の意味で安全な MAC であることが示されている．以下の定理が示されている．

Algorithm EMAC-$\mathcal{G}_{K_1,K_2}(M)$
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow E_{K_1}(X[i])$
$T \leftarrow$ the left most τ bits of $E_{K_2}(Y[m])$
return T

図 35: EMAC のタグ生成アルゴリズム EMAC-$\mathcal{G}_{K_1,K_2}(\cdot)$.

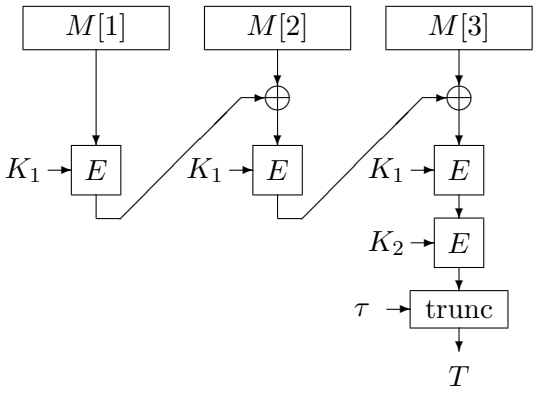


図 36: $M = M[1]M[2]M[3]$ の場合の EMAC-$\mathcal{G}_{K_1,K_2}(M)$ の動作.

Algorithm EMAC-$\mathcal{V}_{K_1,K_2}(M,T)$
$T' \leftarrow$ EMAC-$\mathcal{G}_{K_1,K_2}(M)$
if $T = T'$ **then return** accept
    else return reject

図 37: EMAC の確認アルゴリズム EMAC-$\mathcal{V}_{K_1,K_2}(\cdot,\cdot)$.

定理 8.2. $n, \tau \geq 1$ を整数，$t, q, \sigma \geq 1$ を $\sigma^2 \leq 2^{(n+1)/2}$ なる整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EMAC}[E,\tau]}(t, q, \sigma) \leq 2\mathbf{Adv}^{\mathrm{prp}}_{E}(t', q') + \frac{3\sigma^2}{2^n} + \frac{1}{2^{\tau-1}}$$

である．ただし，$q' = \sigma$, $t' = t + O(n\sigma)$ であり，質問の長さはブロック単位である．

定理 8.2 は，以下のことを示している：実行時間 t, 高々 q 回の質問をし，それらの質問が合計で高々 σ ブロックであり，

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EMAC}[E,\tau]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O(n\sigma)$, 高々 $q' = \sigma$ 回の質問の後，

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \geq \frac{\epsilon}{2} - \frac{3\sigma^2}{2^{n+1}} - \frac{1}{2^{\tau}}$$

なる敵 B が存在する．

効率 EMAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K_1, K_2 \in \mathcal{K}_E$ の二つである．
- ブロック暗号鍵スケジューリングの呼び出し回数：2 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$(|M|/n) + 1$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況 ISO/IEC 9797-1 に含まれている [ISOIEC9797-1]．また，NESSIE の portfolio にも含まれている [WWW8]．実装が容易であること，証明可能安全であること，効率と鍵スケジューリングが妥当であることなどが挙げられている [WWW8].

8.3 RMAC

RMAC にはいくつかのバージョンが存在する．Jaulmes, Joux, Vallete によって提案されたオリジナルの RMAC [JJ+02a, JJ+02b] と NIST が SP800-38B のドラフト版 [SP800-38B] で提案した RMAC である．それぞれ RMAC-JJV と RMAC-NIST と表記することにする．

Jaulmes, Joux, Vallete によって NIST に提案された文書 [JJ+02c] には二つの方式が提案されている．メッセージのパディングの仕方が異なり，それぞれ，RMAC-JJV1, RMAC-JJV2 と表記する．

方式 (RMAC-JJV1) はじめに，RMAC-JJV1 について述べる．

$$\text{RMAC-JJV1} = (\text{RMAC-JJV1-}\mathcal{K}, \text{RMAC-JJV1-}\mathcal{G}, \text{RMAC-JJV1-}\mathcal{V})$$

の鍵生成アルゴリズム RMAC-JJV1-$\mathcal{K}$, タグ生成アルゴリズム RMAC-JJV1-$\mathcal{G}$, 確認アルゴリズム RMAC-JJV1-$\mathcal{V}$ はそれぞれ以下のように動作する．

$\mathrm{Perm}(n)$ を n ビット上のすべての置換の集合とし，r を整数とする．ランダム置換 f_1 とは，$\mathrm{Perm}(n)$ から一様ランダムに選んだ f_1 である．$f_1 \xleftarrow{R} \mathrm{Perm}(n)$ と表記する．置換族 F_2 を以下のように定義する．

$$F_2 = \left\{ f_2^{(R)} \mid R \in \{0,1\}^r, f_2^{(R)} \in \mathrm{Perm}(n) \right\}$$

すなわち，各 $R \in \{0,1\}^r$ に対し，インデックス R を持つ置換 $f_2^{(R)} \in \mathrm{Perm}(n)$ からなる集合である．

- 鍵生成アルゴリズム RMAC-JJV1-$\mathcal{K}$ は確率的アルゴリズムであり，$f_1 \xleftarrow{R} \mathrm{Perm}(n)$ と，各 $R \in \{0,1\}^r$ に対し，

  $$f_2^{(R)} \xleftarrow{R} \mathrm{Perm}(n)$$

  を出力する．

  すなわち，鍵空間は $\mathrm{Perm}(n) \times F_2$ であり，鍵は，

  $$f_1, f_2^{(0,\ldots,0)}, f_2^{(0,\ldots,1,0)}, \ldots, f_2^{(1,\ldots,1)}$$

  となる．

- タグ生成アルゴリズム RMAC-JJV1-$\mathcal{G}$: $(\mathrm{Perm}(n) \times F_2) \times \{0,1\}^* \to (\{0,1\}^n \times \{0,1\}^r)$ は確率的アルゴリズムであり，鍵空間は $\mathrm{Perm}(n) \times F_2$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^n \times \{0,1\}^r$ である．

鍵 $f_1 \in \mathrm{Perm}(n)$, $f_2^{(R)}$ $(R \in \{0,1\}^r)$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{RMAC-JJV1-}\mathcal{G}_{f_1, f_2^{(R)}}(M) \in \{0,1\}^n \times \{0,1\}^r$$

を出力する．図 38, 図 39 にあるように動作する．

Algorithm RMAC-JJV1-$\mathcal{G}_{f_1, f_2^{(R)}}(M)$
$R \xleftarrow{R} \{0,1\}^r$
Pad M
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow f_1(X[i])$
$T' \leftarrow f_2^{(R)}(Y[m])$
$T \leftarrow (T', R)$
return T

図 38: RMAC-JJV1 のタグ生成アルゴリズム RMAC-JJV1-$\mathcal{G}_{f_1, f_2^{(\cdot)}}(\cdot)$.

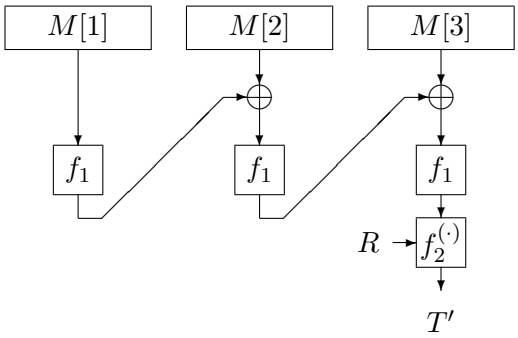


図 39: $M = M[1]M[2]M[3]$ の場合の RMAC-JJV1-$\mathcal{G}_{f_1, f_2^{(R)}}(M)$ の動作.

まず，2 行目で r ビットの乱数 R を生成し，この R をインデックスにもつ $f_2^{(R)}$ を最終ブロックの暗号化の際に用いる．3 行目では，M に 1 を連結してから，全体が n ビットの倍数になるよう 0 を連

結する．すなわち，

$$M \leftarrow M\|1\|0^{n-1-|M| \bmod n}$$

とする．M がすでに n の整数倍である場合は，10^{n-1} を連結する．

- 確認アルゴリズム RMAC-JJV1-$\mathcal{V}$: $(\mathrm{Perm}(n) \times F_2) \times \{0,1\}^* \times (\{0,1\}^n \times \{0,1\}^r) \rightarrow$ accept or reject は決定的アルゴリズムであり，鍵 $f_1 \in \mathrm{Perm}(n)$, $f_2^{(\cdot)} \in F_2$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\{0,1\}^n \times \{0,1\}^r)$ を入力とし，

$$\text{accept or reject} = \text{RMAC-JJV1-}\mathcal{V}_{f_1, f_2^{(R)}}(M, T)$$

を出力する．図 40 にあるように動作する．

Algorithm RMAC-JJV1-$\mathcal{V}_{f_1, f_2^{(R)}}(M, (T', R))$
Pad M
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow f_1(X[i])$
$T'' \leftarrow f_2^{(R)}(Y[m])$
if $T' = T''$ **then return** accept
    else return reject

図 40: RMAC-JJV1 の確認アルゴリズム RMAC-JJV1-$\mathcal{V}_{f_1, f_2^{(\cdot)}}(\cdot, \cdot)$.

方式 (RMAC-JJV2)　RMAC-JJV2 は，パディングの仕方が RMAC-JJV1 とは異なる．メッセージ長が n の倍数の場合は，10^{n-1} を連結する必要がなく，ブロック暗号の呼び出し回数を 1 回削減できる．

メッセージ長が n の倍数である場合は，$f_2^{(R)}$ を，そうでない場合は，それとは異なる $f'^{(R)}_2$ を用いる．

AES を用いた方式 (RMAC-JJV1)　ブロック暗号として AES を用いた場合の RMAC-JJV1 の実装方法が提案されている．まず，$r = 128$ として，128 ビット乱数 R を生成する．K_1 を 128 ビット鍵，K_2 を 128

ビット，もしくは 256 ビット鍵とする．f_1 として，AES_{K_1} を，$f_2^{(R)}$ として，$\mathrm{AES}_{K_2 \oplus R}$ (K_2 が 128 ビットの場合)，もしくは $\mathrm{AES}_{K_2 \oplus (R \| 0^{128})}$ (K_2 が 256 ビットの場合) とする．

AES を用いた方式 (RMAC-JJV2)　ブロック暗号として AES を用いた場合の RMAC-JJV2 の実装方法が提案されている．まず，K_1 を 128 ビット鍵，K_2 を 192 ビット，もしくは 256 ビット鍵とする．RMAC-JJV2 では K_2 の長さが R の長さよりも長くなくてはならないので，128 ビットの K_2 を用いることはできない．

$r = 128$ として，128 ビット乱数 R' を生成する．f_1 として，AES_{K_1} をもちいる．メッセージ長が n の倍数の場合，$f_2^{(R)}$ として，$\mathrm{AES}_{K_2 \oplus R}$, $R = (R' \| 1 \| 0^{63})$ (K_2 が 192 ビットの場合) もしくは $\mathrm{AES}_{K_2 \oplus R}$, $R = (R' \| 1 \| 0^{127})$ (K_2 が 256 ビットの場合) とする．

メッセージ長が n の倍数ではない場合，$f'^{(R)}_2$ として，$\mathrm{AES}_{K_2 \oplus R}$, $R = (R' \| 0 \| 0^{63})$ (K_2 が 192 ビットの場合) もしくは $\mathrm{AES}_{K_2 \oplus R}$, $R = (R' \| 0 \| 0^{127})$ (K_2 が 256 ビットの場合) とする．

方式 (RMAC-NIST)　NIST は SP800-38B のドラフト版で RMAC を提案した．R の扱いがオリジナルとは異なる．オリジナルでは，R が乱数であったのに対し，NIST の提案ではカウンタになることが許されていた．また，RMAC-JJV1 に対応する提案のみであり，RMAC-JJV2 に対応する提案はなかった．

RMAC-NIST はパラメータとして，ブロック暗号 $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, タグ長 τ, R の長さ r をとる．これらのパラメータを用いた場合，RMAC-NIST$[E, \tau, r]$ と表記する．

RMAC-NIST$[E, \tau, r] = (\text{RMAC-NIST-}\mathcal{K}, \text{RMAC-NIST-}\mathcal{G}, \text{RMAC-NIST-}\mathcal{V})$ の鍵生成アルゴリズム RMAC-NIST-$\mathcal{K}$, タグ生成アルゴリズム RMAC-NIST-$\mathcal{G}$, 確認アルゴリズム RMAC-NIST-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム RMAC-NIST-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K_1, K_2 \xleftarrow{R} \{0,1\}^k$$

  を出力する．

- タグ生成アルゴリズム RMAC-NIST-$\mathcal{G} : (\{0,1\}^k)^2 \times \{0,1\}^r \times \{0,1\}^* \to \{0,1\}^r \times \{0,1\}^\tau)$ は決定的アルゴリズムであり，鍵空間は $(\{0,1\}^k)^2$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^r \times \{0,1\}^\tau$ である．さらに NIST の仕様では R はタグ生成アルゴリズムへの入力として

扱われる．すなわち，鍵 $K_1, K_2 \in \{0,1\}^k$, $R \in \{0,1\}^r$, メッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{RMAC-NIST-}\mathcal{G}_{K_1,K_2}(R, M) \in \{0,1\}^r \times \{0,1\}^\tau$$

を出力する．図 41, 図 42 にあるように動作する．

Algorithm RMAC-NIST-$\mathcal{G}_{K_1,K_2}(R, M)$
Pad M
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow E_{K_1}(X[i])$
if $r = 0$ **then** $K_3 \leftarrow K_2$
    else $K_3 \leftarrow K_2 \oplus (R \| 0^{k-r})$
$T' \leftarrow$ the left most τ bits of $E_{K_2}(Y[m])$
$T \leftarrow (R, T')$
return T

図 41: RMAC-NIST のタグ生成アルゴリズム RMAC-NIST-$\mathcal{G}_{K_1,K_2}(\cdot,\cdot)$.

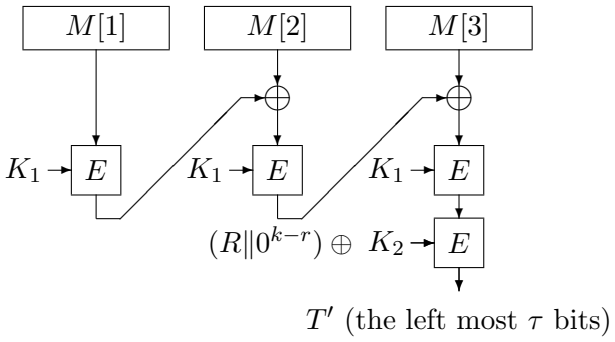


図 42: $M = M[1]M[2]M[3]$ の場合の RMAC-NIST-$\mathcal{G}_{K_1,K_2}(R, M)$ の動作.

3 行目では，M に 1 を連結してから，全体が n ビットの倍数になるよう 0 を連結する．すなわち，

$$M \leftarrow M \| 1 \| 0^{n-1-|M| \bmod n}$$

とする．M がすでに n の整数倍である場合は，10^{n-1} を連結する．

- 確認アルゴリズム RMAC-NIST-$\mathcal{V}$: $(\{0,1\}^k)^2 \times \{0,1\}^* \times (\{0,1\}^r \times \{0,1\}^\tau) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K_1, K_2 \in \{0,1\}^k$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\{0,1\}^r \times \{0,1\}^\tau)$ を入力とし，

$$\text{accept or reject} = \text{RMAC-NIST-}\mathcal{V}_{K_1,K_2}(M,T)$$

を出力する．図 43 にあるように動作する．

Algorithm RMAC-NIST-$\mathcal{V}_{K_1,K_2}(M,(R,T'))$
Pad M
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** m **do**
　　$X[i] \leftarrow M[i] \oplus Y[i-1]$
　　$Y[i] \leftarrow E_{K_1}(X[i])$
if $r = 0$ **then** $K_3 \leftarrow K_2$
　　else $K_3 \leftarrow K_2 \oplus (R \| 0^{k-r})$
$T'' \leftarrow$ the left most τ bits of $E_{K_2}(Y[m])$
if $T' = T''$ **then return** accept
　　else return reject

図 43: RMAC-NIST の確認アルゴリズム RMAC-NIST-$\mathcal{V}_{K_1,K_2}(\cdot,\cdot)$.

パラメータについて　RMAC-NIST はパラメータとして，ブロック暗号 $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, タグ長 τ, R の長さ r をとる．E としては，AES-128, AES-192, AES-256, Triple DES-112, Triple DES-168 のいずれかを，τ と r については，図 44 を提案している．

Parameter Set II 〜 V は一般的な使用に適している，と述べられている．

安全性　ランダム置換 f_1 とランダム置換族 $f_2^{(R)}$ $(R \in \{0,1\}^R)$ を用いた RMAC-JJV1 と RMAC-JJV2 について安全性が解析されている [JJ+02a, JJ+02b]．安全性の定義は，一般的な強偽造不能性に近いが，タグが異なっていなければならない点が異なる．

敵は，タグ生成オラクルと確認オラクルをもつ．タグ生成オラクルに q 個のメッセージ $M_1, \ldots, M_q$ を質問し，その答え $T_1, \ldots, T_q$ を得たとする．

Parameter Set	$n = 128$		$n = 64$	
	r	τ	r	τ
I	0	32	0	32
II	0	64	64	64
III	16	80	n/a	
IV	64	96	n/a	
V	128	128	n/a	

図 44: RMAC-NIST のパラメータ.

また, 確認オラクルに q' 個のメッセージ, タグのペア $(M'_1, T'_1), \ldots, (M'_{q'}, T'_{q'})$ を質問したとする.

ある i に対し, $MAC\text{-}\mathcal{V}_K(M'_i, T'_i) = \text{accept}$ であり, $T'_i \notin \{T_1, \ldots, T_j\}$ であれば, A は偽造に成功した, という. $\{T_1, \ldots, T_j\}$ は, (M'_i, T'_i) を確認オラクルに質問する以前に, タグ生成オラクルから返ってきた答えである.

直感的には, メッセージは見たことがあってもよいが, タグは見たことがあってはならない. たとえば, タグ生成オラクルに M_1 を送り, T_1 を得たとする. 次に, タグ生成オラクルに M_2 を送り, T_2 を得たとする. 強偽造不能性の定義では (M_1, T_2) を偽造文として許すが, 上記安全性の定義では, これは偽造文ではない.

ここで, アドバンテージ $\mathbf{Adv}^{\text{rmac-uf}}(A)$ を以下のように定義する.

$$\mathbf{Adv}^{\text{rmac-uf}}(A) \stackrel{\text{def}}{=} \Pr(f_1 \stackrel{R}{\leftarrow} \text{Perm}(n), f_2^{(R)} \stackrel{R}{\leftarrow} F_2 :$$

$$A^{\text{RMAC-JJV-}\mathcal{G}_{f_1, f_2^{(R)}}(\cdot), \text{RMAC-JJV-}\mathcal{V}_{f_1, f_2^{(R)}}(\cdot,\cdot)} \text{ が上記の意味で偽造に成功})$$

以下の定理が示されている [JJ+02a, JJ+02b].

定理 8.3. $n \geq 2$ を整数, $r = n$ とする. A を高々 σ ブロックの質問をする敵とする. このとき,

$$\mathbf{Adv}^{\text{rmac-uf}}(A) \leq \frac{4n\sigma + 4\sigma + 2}{2^n}$$

である.

上記の安全性のバウンドはほかに比べ, 非常に小さいことがわかる. $\sigma \approx 2^{n/2}$ とすると, XCBC や OMAC のバウンドは 1 を超えるのに対し, 上記のアドバンテージは小さい値のままである.

ただし，上記の定理は帰着を示していない．ランダム置換やランダム置換族を鍵として持つのは，非現実的である．

AES を用いた実装に対しても安全性解析がなされている [JJ+02a, JJ+02b] が，該当箇所には議論の不備が指摘されている [R02a]．XCBC や OMAC のように，ブロック暗号の擬似ランダム性に安全性を帰着させる結果は知られていない．[R02b] では帰着することは不可能である，と述べられている．

効率　RMAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K_1, K_2 \in \mathcal{K}_E$ の二つである．
- ブロック暗号鍵スケジューリングの呼び出し回数：鍵生成アルゴリズム実行時に 1 回必要であり，さらにタグ生成アルゴリズム，もしくは確認アルゴリズムを呼び出すごとに 1 回必要である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$(|M|/n) + 1$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況　NIST に提案され，2002 年 10 月，SP800-38B のドラフト版が提案された．

これに対し，いくつかのコメントが寄せられた．Knudsen は，Triple DES を用いたときの安全性の問題点を指摘した [K02]．Rogaway [R02b], Wagner [W02a], Black [B02] は，いずれも，ブロック暗号の擬似ランダム性に RMAC の安全性が帰着できない点を指摘した．また，そもそも birthday bound をこえる安全性への疑問も出された．MAC は多くの場合，暗号化方式と組み合わせて使用される．多くの暗号化方式，たとえば CTR や CBC は birthday bound をこえる安全性を有していない．これらの暗号化方式は birthday bound に到達すると，平文に関する情報を漏洩する．これらのコメントはいずれも，RMAC の決定を見直すべきだと主張している．

NIST は 2003 年 6 月，RMAC の決定を見直し，OMAC を提案すると発表し [WWW9]，2005 年 5 月，OMAC を正式に推奨方式として採用した [SP800-38B].

NESSIE でも考慮されたが，最終的には portfolio には含まれなかった [WWW8].

8.4 XCBC

方式 XCBC はブロック暗号 E とタグ長 τ をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならない．これらのパラメータを用いた XCBC を XCBC$[E,\tau]$ と表記する．XCBC$[E,\tau]$ = (XCBC-$\mathcal{K}$, XCBC-$\mathcal{G}$, XCBC-$\mathcal{V}$) の鍵生成アルゴリズム XCBC-$\mathcal{K}$, タグ生成アルゴリズム XCBC-$\mathcal{G}$, 確認アルゴリズム XCBC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XCBC-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K_1 \xleftarrow{R} \mathcal{K}_E$$

$$K_2 \xleftarrow{R} \{0,1\}^n$$

$$K_3 \xleftarrow{R} \{0,1\}^n$$

  を出力する．

- タグ生成アルゴリズム XCBC-$\mathcal{G}$: $(\mathcal{K}_E \times (\{0,1\}^n)^2) \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E \times (\{0,1\}^n)^2$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K_1 \in \mathcal{K}_E$, $K_2, K_3 \in \{0,1\}^n$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{XCBC-}\mathcal{G}_{K_1,K_2,K_3}(M) \in \{0,1\}^\tau$$

  を出力する．図 45, 図 46 にあるように動作する．XCBC は M の長さが n の倍数でなくてもよい．図 45 の 3 行目において，

$$M = M[1]M[2]\cdots M[m-1]M[m]$$

  は，$|M[1]| = |M[2]| = \cdots = |M[m-1]|$ かつ $1 \leq |M[m]| \leq n$ となるように分割される．$M = \varepsilon$ のときは例外である．この場合，$|M[m]| = 0$ となる．

  また，図 45 の 7 行目の関数 $\mathbf{pad}_n : \{0,1\}^{\leq n} \to \{0,1\}^n$ は以下のように定義される．a を長さが高々 n ビットのビット列とする ($a = \varepsilon$ でもよい)．ことのき，

$$\mathbf{pad}_n(a) = \begin{cases} a10^{n-|a|-1} & \text{if } |a| < n, \\ a & \text{if } |a| = n. \end{cases} \tag{1}$$

Algorithm XCBC-$\mathcal{G}_{K_1,K_2,K_3}(M)$
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** $m-1$ **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow E_{K_1}(X[i])$
$X[m] \leftarrow \mathbf{pad}_n(M[m]) \oplus Y[m-1]$
if $|M[m]| = n$ **then** $X[m] \leftarrow X[m] \oplus K_2$
    else $X[m] \leftarrow X[m] \oplus K_3$
$T \leftarrow$ the left most τ bits of $E_{K_1}(Y[m])$
return T

図 45: XCBC のタグ生成アルゴリズム XCBC-$\mathcal{G}_{K_1,K_2,K_3}(\cdot)$.

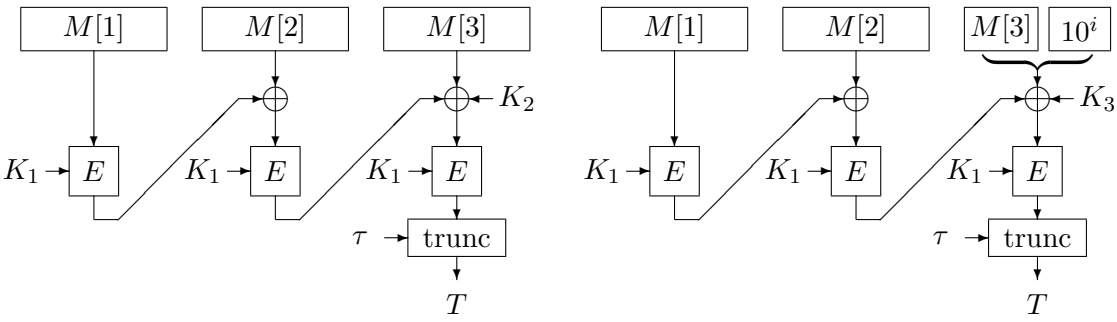


図 46: $M = M[1]M[2]M[3]$ の場合の XCBC-$\mathcal{G}_{K_1,K_2,K_3}(M)$ の動作.

- 確認アルゴリズム XCBC-$\mathcal{V}$: $(\mathcal{K}_E \times (\{0,1\}^n)^2) \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K_1 \in \mathcal{K}_E, K_2, K_3 \in \{0,1\}^n$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{XCBC-}\mathcal{V}_{K_1,K_2,K_3}(M,T)$$

を出力する．図 47 にあるように動作する．

Algorithm XCBC-$\mathcal{V}_{K_1,K_2,K_3}(M,T)$
$T' \leftarrow$ XCBC-$\mathcal{G}_{K_1,K_2,K_3}(M)$
if $T = T'$ **then return** accept
else return reject

図 47: XCBC の確認アルゴリズム XCBC-$\mathcal{V}_{K_1,K_2,K_3}(\cdot,\cdot)$.

安全性 Black, Rogaway により，安全性が解析されている [BR00]. ブロック暗号 E が安全な擬似ランダム置換族であれば，XCBC$[E,\tau]$ は，偽造不可能性の意味で安全な MAC であることが示されている．以下の定理が示されている [IK03b].

定理 8.4. $n, \tau \geq 1$ を整数，$t, q, \sigma \geq 1$ を $\sigma^2 \leq 2^{(n+1)/2}$ なる整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\mathbf{Adv}^{\text{mac}}_{\text{XCBC}[E,\tau]}(t,q,\sigma) \leq \mathbf{Adv}^{\text{prp}}_E(t',q') + \frac{3\sigma^2}{2^n} + \frac{1}{2^\tau}$$

である．ただし，$q' = \sigma$, $t' = t + O(n\sigma)$ であり，質問の長さはブロック単位である．

定理 8.4 は，以下のことを示している： 実行時間 t, 高々 q 回の質問をし，それらの質問が合計で高々 σ ブロックであり，

$$\mathbf{Adv}^{\text{mac}}_{\text{XCBC}[E,\tau]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O(n\sigma)$, 高々 $q' = \sigma$ 回の質問をし，

$$\mathbf{Adv}^{\text{prp}}_E(B) \geq \epsilon - \frac{3\sigma^2}{2^n} - \frac{1}{2^\tau}$$

なる敵 B が存在する．

効率　XCBC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K_1 \in \mathcal{K}_E$ と n ビットの鍵 $K_2, K_3 \in \{0,1\}^n$ の計 3 つが必要である．
- ブロック暗号鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$\max\{1, \lceil |M|/n \rceil\}$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況　NIST に提案されている [WWW9]．また，[FH03] や [H03b] で議論されている．

8.5　TMAC

方式　TMAC はブロック暗号 E とタグ長 τ をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならない．これらのパラメータを用いた TMAC を TMAC$[E, \tau]$ と表記する．TMAC$[E, \tau]$ = (TMAC-$\mathcal{K}$, TMAC-$\mathcal{G}$, TMAC-$\mathcal{V}$) の鍵生成アルゴリズム TMAC-$\mathcal{K}$, タグ生成アルゴリズム TMAC-$\mathcal{G}$, 確認アルゴリズム TMAC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム TMAC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K_1 \xleftarrow{R} \mathcal{K}_E$$

  と

  $$K_2 \xleftarrow{R} \{0,1\}^n$$

  を出力する．

- タグ生成アルゴリズム TMAC-$\mathcal{G} : (\mathcal{K}_E \times \{0,1\}^n) \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E \times \{0,1\}^n$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K_1 \in \mathcal{K}_E$, $K_2 \in \{0,1\}^n$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

  $$T = \text{TMAC-}\mathcal{G}_{K_1,K_2}(M) \in \{0,1\}^\tau$$

を出力する．図 48, 図 49 にあるように動作する．図 48, 図 49 において，$K_2 \cdot \mathtt{u}$ は，GF(2^n) 上の乗算である．一般的に $a \in \{0,1\}^n$ に対し，

$$a \cdot \mathtt{u} = \begin{cases} a \ll 1 & \text{if } \mathtt{msb}(a) = 0, \\ (a \ll 1) \oplus \mathtt{Cst}_n & \text{otherwise} \end{cases} \tag{2}$$

となる．ここで，(2) において，$a \ll 1$ は a の左 1 ビットシフトを表し，$a = a_{n-1}a_{n-2}\cdots a_1a_0$ と a をビット表現した場合，$a \ll 1 = a_{n-2}a_{n-3}\cdots a_00$ となる．すなわち，最上位ビットはなくなり，最下位ビットに 0 が補充される．また，$\mathtt{msb}(a)$ は a の最上位ビットを表し，$\mathtt{Cst}_n$ は n ビットの定数である．たとえば，$\mathtt{Cst}_{128} = 0^{120}10000111$ であり，$\mathtt{Cst}_{64} = 0^{59}11011$ である．TMAC も XCBC と同様，M

Algorithm TMAC-$\mathcal{G}_{K_1,K_2}(M)$
$Y[0] \leftarrow 0^n$
Partition M into $M[1]\cdots M[m]$
for $i \leftarrow 1$ **to** $m-1$ **do**
　　$X[i] \leftarrow M[i] \oplus Y[i-1]$
　　$Y[i] \leftarrow E_{K_1}(X[i])$
$X[m] \leftarrow \mathbf{pad}_n(M[m]) \oplus Y[m-1]$
if $|M[m]| = n$ **then** $X[m] \leftarrow X[m] \oplus K_2 \cdot \mathtt{u}$
　　else $X[m] \leftarrow X[m] \oplus K_2$
$T \leftarrow$ the left most τ bits of $E_{K_1}(Y[m])$
return T

図 48: TMAC のタグ生成アルゴリズム TMAC-$\mathcal{G}_{K_1,K_2}(\cdot)$.

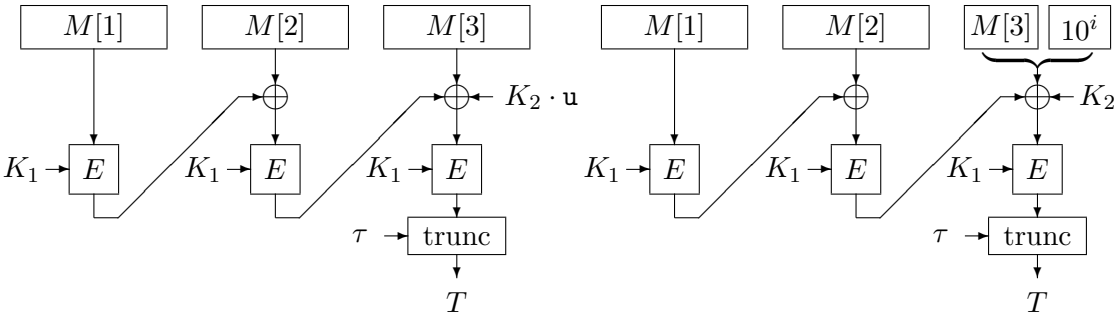


図 49: $M = M[1]M[2]M[3]$ の場合の TMAC-$\mathcal{G}_{K_1,K_2}(M)$ の動作.

の長さが n の倍数でなくてもよい．図 48 の 3 行目において，

$$M = M[1]M[2]\cdots M[m-1]M[m]$$

は，$|M[1]| = |M[2]| = \cdots = |M[m-1]|$ かつ $1 \le |M[m]| \le n$ となるように分割される．$M = \varepsilon$ のときは例外でり，この場合，$|M[m]| = 0$ となる．

また，図 48 の 7 行目の関数 $\mathbf{pad}_n : \{0,1\}^{\le n} \to \{0,1\}^n$ は (1) のように定義される．

- 確認アルゴリズム TMAC-$\mathcal{V} : (\mathcal{K}_E \times \{0,1\}^n) \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K_1 \in \mathcal{K}_E, K_2 \in \{0,1\}^n$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{TMAC-}\mathcal{V}_{K_1,K_2}(M,T)$$

を出力する．図 50 にあるように動作する．

Algorithm TMAC-$\mathcal{V}_{K_1,K_2}(M,T)$
$T' \leftarrow$ TMAC-$\mathcal{G}_{K_1,K_2}(M)$
if $T = T'$ **then return** accept
else return reject

図 50: TMAC の確認アルゴリズム TMAC-$\mathcal{V}_{K_1,K_2}(\cdot,\cdot)$.

安全性 Kurosawa, Iwata により，安全性が解析されている [KI03]．ブロック暗号 E が安全な擬似ランダム置換族であれば，TMAC$[E,\tau]$ は，偽造不可能性の意味で安全な MAC であることが示されている．以下の定理が示されている [IK03b].

定理 8.5. $n, \tau \ge 1$ を整数，$t, q, \sigma \ge 1$ を $\sigma^2 \le 2^{(n+1)/2}$ なる整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\mathbf{Adv}^{\text{mac}}_{\text{TMAC}[E,\tau]}(t,q,\sigma) \le \mathbf{Adv}^{\text{prp}}_E(t',q') + \frac{3\sigma^2}{2^n} + \frac{1}{2^\tau}$$

である．ただし，$q' = \sigma$, $t' = t + O(n\sigma)$ であり，質問の長さはブロック単位である．

定理 8.5 は，以下のことを示している： 実行時間 t, 高々 q 回の質問をし，それらの質問が合計で高々 σ ブロックであり，

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{TMAC}[E,\tau]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O(n\sigma)$, 高々 $q' = \sigma$ 回の質問をし，

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \geq \epsilon - \frac{3\sigma^2}{2^n} - \frac{1}{2^\tau}$$

なる敵 B が存在する．

効率　TMAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K_1 \in \mathcal{K}_E$ と n ビットの鍵 $K_2 \in \{0,1\}^n$ の計 2 つが必要である．
- ブロック暗号鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$\max\{1, \lceil |M|/n \rceil\}$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況　NIST に提案されている [WWW9].

8.6　OMAC/CMAC

OMAC は 2 つの方式 OMAC1 と OMAC2 の総称である．OMAC1 は NIST の SP800-38B に採用され、CMAC と呼ばれている．

方式 (OMAC1)　OMAC1 はブロック暗号 E とタグ長 τ をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならない．これらのパラメータを用いた OMAC1 を $\mathrm{OMAC1}[E,\tau]$ と表記する．

$$\mathrm{OMAC1}[E,\tau] = (\text{OMAC1-}\mathcal{K}, \text{OMAC1-}\mathcal{G}, \text{OMAC1-}\mathcal{V})$$

の鍵生成アルゴリズム OMAC1-$\mathcal{K}$, タグ生成アルゴリズム OMAC1-$\mathcal{G}$, 確認アルゴリズム OMAC1-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム OMAC1-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K \xleftarrow{R} \mathcal{K}_E$$

を出力する．

- タグ生成アルゴリズム OMAC1-$\mathcal{G}$: $\mathcal{K}_E \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K \in \mathcal{K}_E$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{OMAC1-}\mathcal{G}_K(M) \in \{0,1\}^\tau$$

を出力する．図 51, 図 52 にあるように動作する．図 51, 図 52 において，$L \cdot \mathtt{u}$ は (2) によって得られ，$L \cdot \mathtt{u}^2$ は $(L \cdot \mathtt{u}) \cdot \mathtt{u}$ として，(2) によって得られる．OMAC1 も XCBC, TMAC と同様，M の長さが n の倍数でなくてもよい．図 51 の 3 行目において，

$$M = M[1]M[2] \cdots M[m-1]M[m]$$

は，$|M[1]| = |M[2]| = \cdots = |M[m-1]|$ かつ $1 \le |M[m]| \le n$ となるように分割される．$M = \varepsilon$ のときは例外でり，この場合，$|M[m]| = 0$ となる．

また，図 51 の 7 行目の関数 $\mathbf{pad}_n : \{0,1\}^{\le n} \to \{0,1\}^n$ は (1) のように定義される．

- 確認アルゴリズム OMAC1-$\mathcal{V}$: $\mathcal{K}_E \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{OMAC1-}\mathcal{V}_K(M,T)$$

を出力する．図 53 にあるように動作する．

方式 (OMAC2) OMAC2 は OMAC1 とほぼ同様である．OMAC1 中の $L \cdot \mathtt{u}^2$ を $L \cdot \mathtt{u}^{-1}$ としたものが OMAC2 である．一般に $a \in \{0,1\}^n$ に対し，

$$a \cdot \mathtt{u}^{-1} = \begin{cases} a \gg 1 & \text{if } \mathtt{lsb}(a) = 0, \\ (a \gg 1) \oplus \mathtt{Cst}'_n & \text{otherwise.} \end{cases} \tag{3}$$

となる．ここで，上記 (3) において，$a \gg 1$ は a の右 1 ビットシフトを表す．$a = a_{n-1}a_{n-2}\cdots a_1a_0$ と a をビット表現した場合，$a \gg 1 =$

Algorithm OMAC1-$\mathcal{G}_K(M)$
$L \leftarrow E_K(0^n)$
$Y[0] \leftarrow 0^n$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** $m-1$ **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow E_K(X[i])$
$X[m] \leftarrow \mathbf{pad}_n(M[m]) \oplus Y[m-1]$
if $|M[m]| = n$ **then** $X[m] \leftarrow X[m] \oplus L \cdot \mathtt{u}$
else $X[m] \leftarrow X[m] \oplus L \cdot \mathtt{u}^2$
$T \leftarrow$ the left most τ bits of $E_K(Y[m])$
return T

図 51: OMAC1 のタグ生成アルゴリズム OMAC1-$\mathcal{G}_K(\cdot)$.

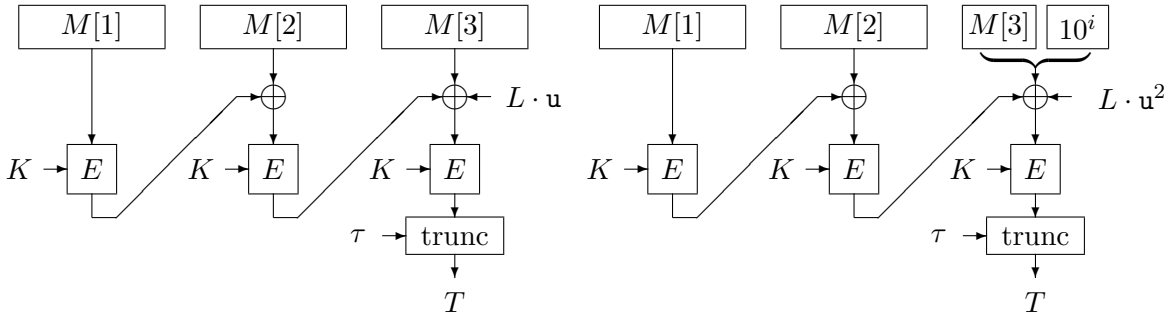


図 52: $M = M[1]M[2]M[3]$ の場合の OMAC1-$\mathcal{G}_K(M)$ の動作.

Algorithm OMAC1-$\mathcal{V}_K(M, T)$
$T' \leftarrow$ OMAC1-$\mathcal{G}_K(M)$
if $T = T'$ **then return** accept
else return reject

図 53: OMAC1 の確認アルゴリズム OMAC1-$\mathcal{V}_K(\cdot, \cdot)$.

$0a_{n-1}a_{n-2}\cdots a_2a_1$ となる．すなわち，最下位ビットはなくなり，最上位ビットに 0 が補充される．また，$\mathtt{lsb}(a)$ は a の最下位ビットを表し，$\mathtt{Cst}'_n$ は n ビットの定数である．たとえば，$\mathtt{Cst}'_{128} = 10^{120}1000011$ である．

安全性　Iwata, Kurosawa により，安全性が解析されている [IK03a]．ブロック暗号 E が安全な擬似ランダム置換族であれば，OMAC1$[E,\tau]$ と OMAC2$[E,\tau]$ は，偽造不可能性の意味で安全な MAC であることが示されている．以下の定理が示されている [IK03b]．

定理 8.6. $n, \tau \geq 1$ を整数，$t, q, \sigma \geq 1$ を $\sigma^2 \leq 2^{(n+1)/2}$ なる整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\begin{cases} \mathbf{Adv}^{\mathrm{mac}}_{\mathrm{OMAC1}[E,\tau]}(t,q,\sigma) \leq \mathbf{Adv}^{\mathrm{prp}}_{E}(t',q') + \dfrac{4\sigma^2}{2^n} + \dfrac{1}{2^\tau} \\ \mathbf{Adv}^{\mathrm{mac}}_{\mathrm{OMAC2}[E,\tau]}(t,q,\sigma) \leq \mathbf{Adv}^{\mathrm{prp}}_{E}(t',q') + \dfrac{4\sigma^2}{2^n} + \dfrac{1}{2^\tau} \end{cases}$$

である．ただし，$q' = \sigma$, $t' = t + O(n\sigma)$ であり，質問の長さはブロック単位である．

定理 8.6 は，以下のことを示している：実行時間 t, 高々 q 回の質問をし，それらの質問が合計で高々 σ ブロックであり，

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{OMAC1}[E,\tau]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O(n\sigma)$, 高々 $q' = \sigma$ 回の質問をし，

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \geq \epsilon - \frac{4\sigma^2}{2^n} - \frac{1}{2^\tau}$$

なる敵 B が存在する．OMAC2 についても同様である．

効率　OMAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K \in \mathcal{K}_E$ の一つのみである．
- ブロック暗号鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$\max\{1, \lceil |M|/n \rceil\}$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：$L = E_K(0^n)$ を計算するのに 1 回必要である．
- 並列処理性：並列処理はできない．

標準化状況　NIST は 2005 年 5 月，OMAC1 を推奨方式として採用した [SP800-38B]．なお，SP 800-38B では，OMAC1 は CMAC (Cipher-based MAC) と呼ばれている．

8.7 XOR MAC

2 つの方式が提案されており，一つは乱数を用いる XMACR 方式，もう一つはカウンタを用いる XMACC 方式である．どちらも関数族 $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ を用いる．F としてブロック暗号を用いてもよいが，入力長 n と出力長 n' は異なってもよい．XMACR, XMACC どちらも，パラメータとして，F と整数 b をとる．ただし，$b \leq n-1$ でなくてはならない．メッセージ M は $|M| \leq b \cdot 2^{n-b-1}$ であり，長さが b の整数倍になるようにパディングされているとする．たとえば，

$$M \leftarrow M \| 10^{(b-|M|-1) \bmod b} \tag{4}$$

とする．関数 $\mathbf{tag} : \{0,1\}^k \times (\{0,1\}^b)^+ \times \{0,1\}^{n-1} \to \{0,1\}^{n'}$ を以下のように定義する．

$$\mathbf{tag}(K, M, r) \stackrel{\text{def}}{=} F_K(0\|r) \oplus F_K(1\|\langle 1\rangle_{n-b-1}\|M[1]) \oplus \cdots \oplus F_K(1\|\langle m\rangle_{n-b-1}\|M[m])$$

ただし，$M = M[1] \cdots M[m]$ はパディングされていて，各 $M[1], \ldots, M[m]$ は b ビット，r は $(n-1)$ ビット，$\langle i\rangle_{n-b-1}$ は整数 i の $(n-b-1)$ ビット表現である．以下，$\mathbf{tag}(K, M, r)$ を $\mathbf{tag}_K(M, r)$ と表記する．図 54 参照．

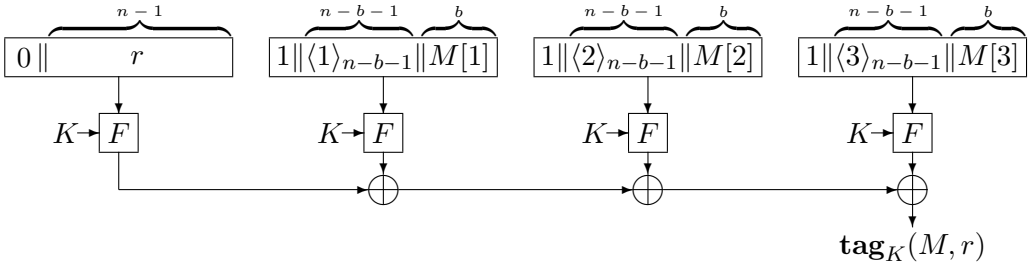


図 54: $M = M[1]M[2]M[3]$ の場合の $\mathbf{tag}_K(M, r)$ の動作．

方式 (XMACR) XMACR は，関数族 F とブロック長 n をパラメータとする．関数族 $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ を用いた場合は，$b \leq n-1$ でなくてはならない．これらのパラメータを用いた XMACR を XMACR$[F,b]$ と表記する．XMACR$[F,b]$ = (XMACR-$\mathcal{K}$, XMACR-$\mathcal{G}$, XMACR-$\mathcal{V}$) の鍵生成アルゴリズム XMACR-$\mathcal{K}$, タグ生成アルゴリズム XMACR-$\mathcal{G}$, 確認アルゴリズム XMACR-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XMACR-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K \xleftarrow{R} \{0,1\}^k$$

を出力する．

- タグ生成アルゴリズム XMACR-$\mathcal{G} : \{0,1\}^k \times \{0,1\}^* \to (\{0,1\}^{n-1} \times \{0,1\}^{n'})$ は確率的アルゴリズムであり，鍵空間は $\{0,1\}^k$, メッセージ空間は $\{0,1\}^*$, タグ空間は $(\{0,1\}^{n-1} \times \{0,1\}^{n'})$ である．すなわち，鍵 $K \in \{0,1\}^k$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{XMACR-}\mathcal{G}_K(M) \in (\{0,1\}^{n-1} \times \{0,1\}^{n'})$$

を出力する．図 55 にあるように動作する．

Algorithm XMACR-$\mathcal{G}_K(M)$
Pad M
$r \xleftarrow{R} \{0,1\}^{n-1}$
$T' \leftarrow \mathbf{tag}_K(M, r)$
$T \leftarrow (r, T')$
return T

図 55: XMACR のタグ生成アルゴリズム XMACR-$\mathcal{G}_K(\cdot)$.

ただし，2 行目は $M \in \{0,1\}^*$ に対しパディングを施し，$M \in (\{0,1\}^n)^+$ となるようにする．たとえば，(4) のようにする．3 行目では $n-1$ ビットの乱数 r を選ぶ.

- 確認アルゴリズム XMACR-$\mathcal{V}$: $\{0,1\}^k \times \{0,1\}^* \times (\{0,1\}^{n-1} \times \{0,1\}^{n'}) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in$

$\{0,1\}^k$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\{0,1\}^{n-1} \times \{0,1\}^{n'})$ を入力とし，

$$\text{accept or reject} = \text{XMACR-}\mathcal{V}_K(M,T)$$

を出力する．図 56 にあるように動作する．

Algorithm XMACR-$\mathcal{V}_K(M,(r,T'))$
Pad M
$T'' \leftarrow \mathbf{tag}_K(M,r)$
if $T' = T''$ **then return** accept
else return reject

図 56: XMACR の確認アルゴリズム XMACR-$\mathcal{V}_K(\cdot,\cdot)$.

ただし，2 行目は $M \in \{0,1\}^*$ に対し，(4) のようにする．

方式 (XMACC)　XMACC は，XMACR と同様，関数族 F とブロック長 n をパラメータとする．関数族 $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ を用いた場合は，$b \leq n-1$ でなくてはならない．これらのパラメータを用いた XMACC を XMACC$[F,b]$ と表記する．

$$\text{XMACC}[F,b] = (\text{XMACC-}\mathcal{K}, \text{XMACC-}\mathcal{G}, \text{XMACC-}\mathcal{V})$$

の鍵生成アルゴリズム XMACC-$\mathcal{K}$, タグ生成アルゴリズム XMACC-$\mathcal{G}$, 確認アルゴリズム XMACC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XMACC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K \xleftarrow{R} \{0,1\}^k$$

  を出力する．

- タグ生成アルゴリズム XMACC-$\mathcal{G}$: $\{0,1\}^k \times \{0,1\}^* \times \mathsf{Count} \to (\mathsf{Count} \times \{0,1\}^{n'})$ は決定的アルゴリズムであり，鍵空間は $\{0,1\}^k$, メッセージ空間は $\{0,1\}^*$, タグ空間は $(\mathsf{Count} \times \{0,1\}^{n'})$ である．さらに入力として，カウンタ $C \in \mathsf{Count}$ をとる．Count はカウンタの空間であり，$\mathsf{Count} = \{0,1,\ldots,2^{n-1}-1\}$ である．カウンタは送信者により保持されており，0 に初期化され，タグ生成アルゴリズムによって更新される．受信者はこれを保持しない．すなわち，鍵

$K \in \{0,1\}^k$, メッセージ $M \in \{0,1\}^*$, カウンタ $C \in \mathsf{Count}$ を入力とし，タグ

$$T = \text{XMACC-}\mathcal{G}_K(M) \in (\mathsf{Count} \times \{0,1\}^{n'})$$

を出力する．図 57 にあるように動作する．

Algorithm XMACC-$\mathcal{G}_K(M, C)$
Pad M
$C \leftarrow C + 1$
$T' \leftarrow \mathbf{tag}_K(M, \langle C \rangle_{n-1})$
$T \leftarrow (C, T')$
return T

図 57: XMACC のタグ生成アルゴリズム XMACC-$\mathcal{G}_K(\cdot)$.

ただし，$\langle C \rangle_{n-1}$ はカウンタ C の $(n-1)$ ビット表現であり，2 行目は $M \in \{0,1\}^*$ に対し，(4) のようにしてから，$M = M[1] \cdots M[m]$ とする．

- 確認アルゴリズム XMACC-$\mathcal{V} : \{0,1\}^k \times \{0,1\}^* \times (\mathsf{Count} \times \{0,1\}^{n'}) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \{0,1\}^k$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\mathsf{Count} \times \{0,1\}^{n'})$ を入力とし，

$$\text{accept or reject} = \text{XMACC-}\mathcal{V}_K(M, T)$$

を出力する．図 58 にあるように動作する．

Algorithm XMACC-$\mathcal{V}_K(M, (C, T'))$
$T'' \leftarrow \mathbf{tag}_K(M, C)$
if $T' = T''$ **then return** accept
else return reject

図 58: XMACC の確認アルゴリズム XMACC-$\mathcal{V}_K(\cdot, \cdot)$.

安全性　Bellare, Guérin, Rogaway により，安全性が解析されている [BGR95]. 関数族 F が安全な擬似ランダム関数族であれば，XMACR$[F, b]$ と XMACC$[F, b]$ は，強偽造不可能性の意味で安全な MAC であることが示されている．

擬似ランダム関数族　関数族 $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ に対し,「F が擬似ランダム関数族である」とは，適応的選択平文攻撃を行う任意の敵が，関数族 $\{F_K(\cdot) \in \mathrm{Rand}(n,n') \mid K \in \{0,1\}^k\}$ と $\{0,1\}^n$ から $\{0,1\}^{n'}$ へのすべての関数の集合 $\mathrm{Rand}(n,n')$ を区別できないことをいう．

より厳密には，敵 A として，オラクルにアクセスできるアルゴリズムを考える．何回かの質問の後，A は 1 ビットを出力する．関数族 $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ の，敵 A に対する，擬似ランダム関数としての安全性は，アドバンテージ $\mathbf{Adv}^{\mathrm{prf}}_F(A)$ によって評価される．ここで，

$$\begin{aligned}\mathbf{Adv}^{\mathrm{prf}}_F(A) \stackrel{\mathrm{def}}{=} & \left|\Pr(K \xleftarrow{R} \{0,1\}^k : A^{F_K(\cdot)} = 1)\right. \\ & \left. - \Pr(R \xleftarrow{R} \mathrm{Rand}(n,n') : A^{R(\cdot)} = 1)\right|\end{aligned}$$

と定義され，$A^{F_K(\cdot)}$ は質問 X に対し，$Y = F_K(X)$ を返すオラクル $F_K(\cdot)$ を持つ敵 A を表し，$A^{R(\cdot)}$ は質問 X に対し，$Y = R(X)$ を返すオラクル $R(\cdot)$ を持つ敵 A を表す．

計算量理論的安全性　関数族 F の擬似ランダム関数族としての安全性を考える場合に扱う資源は，実行時間 t とオラクルへの質問回数 q である．

$$\mathbf{Adv}^{\mathrm{prf}}_F(t,q) \stackrel{\mathrm{def}}{=} \max_A \left\{\mathbf{Adv}^{\mathrm{prf}}_E(A)\right\}$$

と定義される．ただし，最大値は実行時間 t, オラクルへの質問回数 q のすべての敵 A についてとる．

XMACR の安全性　$\mathbf{Adv}^{\text{s-uf}}_{\mathrm{XMACR}[F,b]}(t,q_g,q_v,m)$ を

$$\mathbf{Adv}^{\text{s-uf}}_{\mathrm{XMACR}[F,b]}(t,q_g,q_v,m) \stackrel{\mathrm{def}}{=} \max_A \left\{\mathbf{Adv}^{\text{s-uf}}_{\mathrm{XMACR}[F,b]}(A)\right\}$$

と定義する．ただし，最大値は実行時間 t, タグ生成オラクルへ高々 q_g 回，確認オラクルへ高々 q_v 回，それぞれの質問の長さが高々 m ブロックであるすべての敵 A についてとる．

XMACR については，以下の定理が示されている [BGR95].

定理 8.7. $n, n' \geq 1$ を整数，b を $b \leq n-1$ なる整数，$t, q_g, q_v, m \geq 1$ を整数とする．$F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ を関数族とする．このとき，

$$\mathbf{Adv}^{\text{s-uf}}_{\mathrm{XMACR}[F,b]}(t,q_g,q_v,m) \leq \mathbf{Adv}^{\mathrm{prf}}_F(t',q') + \frac{2q_s^2}{2^n} + \frac{2q_v^2}{2^{n'}}$$

である．ただし，$q' = (q_g + q_v) \cdot (m+1)$, $t' = t + O((n+n')q')$ である．

定理 8.7 は，以下のことを示している： 実行時間 t, タグ生成オラクルに高々 q_g 回，確認オラクルに高々 q_v 回の質問をし，それぞれの質問が高々 m ブロック (1 ブロックは b ビット) であり，

$$\mathbf{Adv}^{\text{s-uf}}_{\text{XMACR}[F,b]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O((n+n')q')$, 高々 $q' = (q_g + q_v) \cdot (m+1)$ 回の質問をし，

$$\mathbf{Adv}^{\text{prp}}_{E}(B) \geq \epsilon - \frac{2q_g^2}{2^n} - \frac{2q_v^2}{2^{n'}}$$

なる敵 B が存在する．

XMACC の安全性 $\mathbf{Adv}^{\text{s-uf}}_{\text{XMACC}[F,b]}(t, q_g, q_v, m)$ を

$$\mathbf{Adv}^{\text{s-uf}}_{\text{XMACC}[F,b]}(t, q_g, q_v, m) \stackrel{\text{def}}{=} \max_{A} \left\{ \mathbf{Adv}^{\text{s-uf}}_{\text{XMACC}[F,b]}(A) \right\}$$

と定義する．最大値のとりかたは，XMACR と同様である．

XMACC については，以下の定理が示されている [BGR95].

定理 8.8. $n, n' \geq 1$ を整数，b を $b \leq n-1$ なる整数，$t, q_g, q_v, m \geq 1$ を $q_g \leq 2^{n-1}$ なる整数とする．$F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^{n'}$ を関数族とする．このとき，

$$\mathbf{Adv}^{\text{s-uf}}_{\text{XMACC}[F,b]}(t, q_g, q_v, m) \leq \mathbf{Adv}^{\text{prf}}_{F}(t', q') + \frac{2q_v^2}{2^{n'}}$$

である．ただし，$q' = (q_g + q_v) \cdot (m+1)$, $t' = t + O((n+n')q')$ である．

定理 8.8 は，以下のことを示している： 実行時間 t, タグ生成オラクルに高々 q_g 回，確認オラクルに高々 q_v 回の質問をし，それぞれの質問が高々 m ブロック (1 ブロックは b ビット) であり，

$$\mathbf{Adv}^{\text{s-uf}}_{\text{XMACC}[F,b]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．このとき，実行時間 $t' = t + O((n+n')q')$, 高々 $q' = (q_g + q_v) \cdot (m+1)$ 回の質問をし，

$$\mathbf{Adv}^{\text{prp}}_{E}(B) \geq \epsilon - \frac{2q_v^2}{2^{n'}}$$

なる敵 B が存在する．

XMACC は，送信者がカウンタを保持しなければならない代わりに，安全性のバウンドが q_g に依存しない，という利点を持っている．

効率　XMACR, XMACC の効率は，以下のようにまとめられる．

- 鍵長：関数族 F (ブロック暗号) の鍵 $K \in \{0,1\}^k$ の一つのみである．
- F の鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかる F の呼び出し回数：パディングの定義により異なるが，(4) のようにした場合は，$\lceil(|M|+1)/b\rceil+1$ 回必要である．F として，ブロック暗号を用いた場合，CBC MAC では $|M|/n$ なので，CBC MAC と比べ，n/b 倍程度必要である．たとえば，$b = n/2$ とした場合，およそ 2 倍の呼び出し回数が必要である．
- 事前計算するべき F の呼び出し回数：XMACC における送信者は $F_K(0\|C)$ を計算できる．
- 並列処理性：並列処理可能である．CBC MAC とその変形は，ブロック暗号の並列処理ができない．

標準化状況　標準化された実績はない．

8.8　XECB MAC

3 つの方式：乱数を用いる XECB$-MAC 方式，送信者が状態を用いる XECBC-MAC 方式，状態と乱数を用いる XECBS-MAC 方式が提案されている．

どの方式もブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いる．

方式 (XECB$-MAC)　XECB$-MAC は乱数を用いる方式である．XECB$-MAC は，ブロック暗号 E をパラメータとする．これを XECB$-MAC[$E$] と表記する．

$$\text{XECB\$-MAC}[E] = (\text{XECB\$-}\mathcal{K}, \text{XECB\$-}\mathcal{G}, \text{XECB\$-}\mathcal{V})$$

の鍵生成アルゴリズム XECB$-$\mathcal{K}$, タグ生成アルゴリズム XECB$-$\mathcal{G}$, 確認アルゴリズム XECB$-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XECB$-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K \xleftarrow{R} \mathcal{K}_E$$

  を出力する．

- タグ生成アルゴリズム XECB$-$\mathcal{G}$: $\mathcal{K}_E \times \{0,1\}^* \to (\{0,1\}^n \times \{0,1\}^n)$ は確率的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^*$, タグ空間は $(\{0,1\}^n \times \{0,1\}^n)$ である．すなわち，鍵 $K \in \mathcal{K}_E$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{XECB\$-}\mathcal{G}_K(M) \in (\{0,1\}^n \times \{0,1\}^n)$$

を出力する．図 59 にあるように動作する．

Algorithm XECB$-$\mathcal{G}_K(M)$
$r_0 \xleftarrow{R} \{0,1\}^n$
$y_0 \leftarrow E_K(r_0)$
$z_0 \leftarrow E_K(r_0 + 1)$
Partition M into $M[1] \cdots M[m]$
if $|M[m]| = n$ **then** $Z \leftarrow \overline{z_0}$
else $Z \leftarrow z_0$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** $m+1$ **do**
$X[i] \leftarrow M[i] \oplus i \times y_0$
$Y[i] \leftarrow E_K(X[i])$
$T' \leftarrow Y[1] \oplus \cdots \oplus Y[m+1]$
$T \leftarrow (r_0, T')$
return T

図 59: XECB$-MAC のタグ生成アルゴリズム XECB$-$\mathcal{G}_K(\cdot)$.

ただし，+ や × の演算は $\bmod 2^n$ 上で行われる．2 行目の r_0 は n ビットの乱数である．6 行目の $\overline{z_0}$ は z_0 のビットごとの反転である．8 行目の $\mathbf{pad}_n(\cdot)$ は (1) で定義されている．

- 確認アルゴリズム XECB$-$\mathcal{V}$: $\mathcal{K}_E \times \{0,1\}^* \times (\{0,1\}^n \times \{0,1\}^n) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\{0,1\}^n \times \{0,1\}^n)$ を入力とし，

$$\text{accept or reject} = \text{XECB\$-}\mathcal{V}_K(M, T)$$

を出力する．図 60 にあるように動作する．

r_0 を用いてタグ生成を行い，一致していれば accpet を返す．

Algorithm XECB$-$\mathcal{V}_K(M,(r_0,T'))$
$y_0 \leftarrow E_K(r_0)$
$z_0 \leftarrow E_K(r_0+1)$
Partition M into $M[1]\cdots M[m]$
if $|M[m]|=n$ **then** $Z \leftarrow \overline{z_0}$
else $Z \leftarrow z_0$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** $m+1$ **do**
$X[i] \leftarrow M[i] \oplus i \times y_0$
$Y[i] \leftarrow E_K(X[i])$
$T'' \leftarrow Y[1] \oplus \cdots \oplus Y[m+1]$
$T''' \leftarrow (r_0, T')$
if $T' = T'''$ **then return** accept
else return reject

図 60: XECB$-MAC の確認アルゴリズム XECB$-$\mathcal{V}_K(\cdot,\cdot)$.

方式 (XECBC-MAC) XECBC-MAC は，送信者が状態を用いる方式である．XECB$-MAC と同様，ブロック暗号 E をパラメータとする．これを，XECBC-MAC$[E]$ と表記する．

$$\text{XECBC-MAC}[E] = (\text{XECBC-}\mathcal{K}, \text{XECBC-}\mathcal{G}, \text{XECBC-}\mathcal{V})$$

の鍵生成アルゴリズム XECBC-$\mathcal{K}$, タグ生成アルゴリズム XECBC-$\mathcal{G}$, 確認アルゴリズム XECBC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XECBC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K \xleftarrow{R} \mathcal{K}_E$$

  を出力する．

- タグ生成アルゴリズム XECBC-$\mathcal{G} : \mathcal{K}_E \times \{0,1\}^* \times \mathsf{Count} \to (\mathsf{Count} \times \{0,1\}^n)$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^*$, タグ空間は $(\mathsf{Count} \times \{0,1\}^n)$ である．さらに入力として，カウンタ $C \in \mathsf{Count}$ をとる．Count はカウンタの空間であり，$\mathsf{Count} = \{1,2,\ldots\}$ である．カウンタは送信者により保持され

ており，1 に初期化され，タグ生成アルゴリズムによって更新される．受信者はこれを保持しない．すなわち，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, カウンタ $C \in \mathsf{Count}$ を入力とし，タグ

$$T = \text{XECBC-}\mathcal{G}_K(M) \in (\mathsf{Count} \times \{0,1\}^n)$$

を出力する．図 61 にあるように動作する．

Algorithm XECBC-$\mathcal{G}_K(C, M)$
$y_0 \leftarrow E_K(C)$
$z_0 \leftarrow E_K(y_0)$
Partition M into $M[1] \cdots M[m]$
if $|M[m]| = n$ **then** $Z \leftarrow \overline{z_0}$
else $Z \leftarrow z_0$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** $m+1$ **do**
$X[i] \leftarrow M[i] \oplus i \times y_0$
$Y[i] \leftarrow E_K(X[i])$
$T' \leftarrow Y[1] \oplus \cdots \oplus Y[m+1]$
$C' \leftarrow C + 1$
$T \leftarrow (C, T')$
return T

図 61: XECBC-MAC のタグ生成アルゴリズム XECBC-$\mathcal{G}_K(\cdot)$.

ただし，C' は更新されたカウンタの値であり，$+$ と $\times$ の演算は，$\bmod 2^n$ 上で行われる．

- 確認アルゴリズム XECBC-$\mathcal{V} : \mathcal{K}_E \times \{0,1\}^* \times (\mathsf{Count} \times \{0,1\}^n) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\mathsf{Count} \times \{0,1\}^n)$ を入力とし，

$$\text{accept or reject} = \text{XMACC-}\mathcal{V}_K(M, T)$$

を出力する．図 62 にあるように動作する．

Algorithm XMACC-$\mathcal{V}_K(M,(C,T'))$
$y_0 \leftarrow E_K(C)$
$z_0 \leftarrow E_K(y_0)$
Partition M into $M[1]\cdots M[m]$
if $|M[m]| = n$ **then** $Z \leftarrow \overline{z_0}$
else $Z \leftarrow z_0$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** $m+1$ **do**
$X[i] \leftarrow M[i] \oplus i \times y_0$
$Y[i] \leftarrow E_K(X[i])$
$T'' \leftarrow Y[1] \oplus \cdots \oplus Y[m+1]$
if $T' = T''$ **then return** accept
else return reject

図 62: XECBC の確認アルゴリズム XECBC-$\mathcal{V}_K(\cdot,\cdot)$.

方式 (XECBS-MAC) XECBS-MAC は状態と乱数を用いる方式である．これも，ブロック暗号 E をパラメータとする．XECBS-MAC[E] と表記する．XECBS-MAC[E] = (XECBS-$\mathcal{K}$, XECBS-$\mathcal{G}$, XECBS-$\mathcal{V}$) の鍵生成アルゴリズム XECBS-$\mathcal{K}$, タグ生成アルゴリズム XECBS-$\mathcal{G}$, 確認アルゴリズム XECBS-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム XECBS-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K \xleftarrow{R} \mathcal{K}_E$$

を出力する．

- タグ生成アルゴリズム XECBS-$\mathcal{G} : \mathcal{K}_E \times \{0,1\}^* \times \mathsf{Count} \to (\mathsf{Count} \times \{0,1\}^n)$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^*$, タグ空間は $(\mathsf{Count} \times \{0,1\}^n)$ である．さらに入力として，カウンタ $C \in \mathsf{Count}$ をとる．Count はカウンタの空間であり，$\mathsf{Count} = \{1,2,\ldots\}$ である．カウンタは送信者により保持されており，1 に初期化され，タグ生成アルゴリズムによって更新される．受信者はこれを保持しない．すなわち，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, カウンタ $C \in \mathsf{Count}$ を入力とし，タグ

$$T = \text{XECBS-}\mathcal{G}_K(M) \in (\mathsf{Count} \times \{0,1\}^n)$$

を出力する．さらにアルゴリズム内部では，状態 R と R^* を用いる．これらは，n ビットのビット列であり，鍵ごとに更新される．メッセージごとに更新されるわけではない．この状態は送信者だけでなく，受信者も保持している．秘密鍵 K から導出してもよいと記されている [GD01a]．図 63 にあるように動作する．

Algorithm XECBS-$\mathcal{G}_K(C, M)$
Partition M into $M[1] \cdots M[m]$
if $|M[m]| = n$ **then** $Z \leftarrow \overline{R}$
else $Z \leftarrow R$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** m **do**
$X[i] \leftarrow M[i] \oplus C \times Z \oplus i \times R^*$
$Y[i] \leftarrow E_K(X[i])$
$T' \leftarrow Y[1] \oplus \cdots \oplus Y[m]$
$C' \leftarrow C + 1$
$T \leftarrow (C, T')$
return T

図 63: XECBS-MAC のタグ生成アルゴリズム XECBS-$\mathcal{G}_K(\cdot)$.

ただし，C' は更新されたカウンタの値であり，$+$ と $\times$ の演算は，$\bmod 2^n$ 上で行われる．

- 確認アルゴリズム XECBS-$\mathcal{V} : \mathcal{K}_E \times \{0,1\}^* \times (\mathsf{Count} \times \{0,1\}^n) \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, タグ $T \in (\mathsf{Count} \times \{0,1\}^n)$ を入力とし，

$$\text{accept or reject} = \text{XMACC-}\mathcal{V}_K(M, T)$$

を出力する．さらにアルゴリズム内部では，状態 R と R^* を用いる．これらは，n ビットのビット列であり，鍵ごとに更新される．メッセージごとに更新されるわけではない．この状態は，受信者も保持している．図 64 にあるように動作する．

安全性 Gligor, Donescu により，安全性が解析されている [GD01a]. ブロック暗号 E が安全な擬似ランダム置換族であれば，XECB$-MAC[$E$],

Algorithm XMACC-$\mathcal{V}_K(M,(C,T'))$
Partition M into $M[1]\cdots M[m]$
if $|M[m]|=n$ **then** $Z \leftarrow \overline{R}$
else $Z \leftarrow R$
$M[m] \leftarrow \mathbf{pad}_n(M[m])$
$M[m+1] \leftarrow Z$
for $i \leftarrow 1$ **to** m **do**
$X[i] \leftarrow M[i] \oplus C \times Z \oplus i \times R^*$
$Y[i] \leftarrow E_K(X[i])$
$T'' \leftarrow Y[1] \oplus \cdots \oplus Y[m]$
if $T'=T''$ **then return** accept
else return reject

図 64: XECBS の確認アルゴリズム XECBS-$\mathcal{V}_K(\cdot,\cdot)$.

XECBC-MAC[E], XECBS-MAC[E] は，いずれも，強偽造不可能性の意味で安全な MAC であることが示されている．

XECB$-MAC の安全性 $\mathbf{Adv}^{\text{s-uf}}_{\text{XECB\$-MAC}[E]}(t,q_g,\mu_g,q_v,\mu_v)$ を

$$\mathbf{Adv}^{\text{s-uf}}_{\text{XECB\$-MAC}[E]}(t,q_g,\mu_g,q_v,\mu_v) \stackrel{\text{def}}{=} \max_A \left\{\mathbf{Adv}^{\text{s-uf}}_{\text{XECB\$-MAC}[E]}(A)\right\}$$

と定義する．ただし，最大値は実行時間 t, タグ生成オラクルへ高々 q_g 回の質問を合計で高々 μ_g ブロック，確認オラクルへ高々 q_v 回の質問を合計で高々 μ_v ブロックであるすべての敵 A についてとる．

XECB$-MAC については，以下の定理が示されている [GD01a].

定理 8.9. $n \geq 1$ を整数，$t,q_g,\mu_g,q_v,\mu_g \geq 1$ を整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^{n'}$ をブロック暗号とする．このとき，

$$\begin{aligned}&\mathbf{Adv}^{\text{s-uf}}_{\text{XECB\$-MAC}[E]}(t,q_g,\mu_g,q_v,\mu_g)\\ &\quad\leq \mathbf{Adv}^{\text{prf}}_E(t',q') + \frac{\mu_v((\log\mu_v)+3)}{2^n} + \frac{q_g\mu_v^2}{2^n}\\ &\quad+\left(q_g+2q_v+\tfrac{\mu_g}{2}\right)\tfrac{\mu_g((\log\mu_v)+3)}{2^{n+1}}\end{aligned}$$

である．ただし，$\mu_s+\mu_g \leq q'$, $t \leq t'$ である．

バウンドは最後の項が最も大きく，ほかの MAC と比べると，通常の birthday paradox バウンドよりも log スケールで悪いことがわかる．

XECBS-MAC の安全性　$\mathbf{Adv}^{\text{s-uf}}_{\text{XECBS-MAC}[E]}(t, q_g, \mu_g, q_v, \mu_v)$ も XECB$-MAC[$E$] と同様に定義する．

XECBS-MAC については，以下の定理が示されている [GD01a].

定理 8.10. $n \geq 1$ を整数，$t, q_g, \mu_g, q_v, \mu_g \geq 1$ を整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^{n'}$ をブロック暗号とする．このとき，

$$\begin{aligned}
&\mathbf{Adv}^{\text{s-uf}}_{\text{XECBS-MAC}[E]}(t, q_g, \mu_g, q_v, \mu_g) \\
&\quad \leq \mathbf{Adv}^{\text{prf}}_{E}(t', q') + \frac{q_v}{2^n} + \frac{\mu_v((\log \mu_v) + 3)}{2^{n+1}} \\
&\qquad + (q_v + \mu_g)\frac{q_s((\log q_s) + 3)}{2^{n+1}} + (q_v + \mu_g)\frac{\mu_g((\log \mu_v) + 3)}{2^{n+1}}
\end{aligned}$$

である．ただし，$\mu_s + \mu_g \leq q'$, $t \leq t'$ である．

XECBC-MAC の安全性　$\mathbf{Adv}^{\text{s-uf}}_{\text{XECBC-MAC}[E]}(t, q_g, \mu_g, q_v, \mu_v)$ も上記二つと同様に定義する．

XECBC-MAC については，以下の定理が示されている [GD01a].

定理 8.11. $n \geq 1$ を整数，$t, q_g, \mu_g, q_v, \mu_g \geq 1$ を整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^{n'}$ をブロック暗号とする．このとき，

$$\begin{aligned}
&\mathbf{Adv}^{\text{s-uf}}_{\text{XECBC-MAC}[E]}(t, q_g, \mu_g, q_v, \mu_g) \\
&\quad \leq \mathbf{Adv}^{\text{prf}}_{E}(t', q') + \frac{\mu_v^2}{2^{n+1}} + \frac{q_v}{2^n} + \frac{\mu_v((\log \mu_v) + 3)}{2^{n+1}} \\
&\qquad + (q_v + \mu_g)\frac{q_g((\log q_g) + 3)}{2^{n+1}} + (q_v + \mu_g)\frac{\mu_g((\log \mu_v) + 3)}{2^{n+1}} + \frac{\mu_g^2}{2^{n+1}}
\end{aligned}$$

である．ただし，$\mu_s + \mu_g \leq q'$, $t \leq t'$ である．

効率　XECB MAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号 E の鍵 $K \in \mathcal{K}_E$ の一つのみである．
- E の鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかる F の呼び出し回数：XECB$-MAC と XECBC-MAC では，$\lceil |M|/n \rceil + 3$ 回，XECBS-MAC では，$\lceil |M|/n \rceil$ 回必要である．しかし，XECBS-MAC では，R と R^* の生成にブロック暗号を呼び出す必要がある場合がある．

- 事前計算するべき F の呼び出し回数：XECBS-MAC では，R と R^* の生成にブロック暗号を呼び出す必要がある場合がある．
- 並列処理性：並列処理可能である．

標準化状況　NIST に提案されている [WWW9].

8.9 PMAC

方式　PMAC はブロック暗号 E とタグ長 τ をパラメータとする．ブロック長 n のブロック暗号 $E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなくてはならない．これらのパラメータを用いた PMAC を PMAC$[E, \tau]$ と表記する．PMAC$[E, \tau]$ = (PMAC-$\mathcal{K}$, PMAC-$\mathcal{G}$, PMAC-$\mathcal{V}$) の鍵生成アルゴリズム PMAC-$\mathcal{K}$, タグ生成アルゴリズム PMAC-$\mathcal{G}$, 確認アルゴリズム PMAC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム PMAC-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K \xleftarrow{R} \mathcal{K}_E$$

を出力する．

- タグ生成アルゴリズム PMAC-$\mathcal{G} : \mathcal{K}_E \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\mathcal{K}_E$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K \in \mathcal{K}_E$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{PMAC-}\mathcal{G}_K(M) \in \{0,1\}^\tau$$

を出力する．図 65, 図 66 にあるように動作する． PMAC は M の長さが n の倍数でなくてもよい．図 65 の 3 行目において，

$$M = M[1]M[2]\cdots M[m-1]M[m]$$

は，$|M[1]| = |M[2]| = \cdots = |M[m-1]|$ かつ $1 \leq |M[m]| \leq n$ となるように分割される．$M = \varepsilon$ のときは例外である．この場合，$|M[m]| = 0$ となる．

5 行目の $\gamma_i \cdot L$ は，以下のように計算される．

$$\begin{cases} \gamma_1 \cdot L = L \\ \gamma_i \cdot L = (\gamma_{i-1} \cdot L) \oplus (L \cdot \mathbf{u}^{\mathbf{ntz}(i)}) \quad (i \geq 2) \end{cases}$$

Algorithm PMAC-$\mathcal{G}_K(M)$
$L \leftarrow E_K(0^n)$
Partition M into $M[1] \cdots M[m]$
for $i \leftarrow 1$ **to** $m-1$ **do**
    $X[i] \leftarrow M[i] \oplus \gamma_i \cdot L$
    $Y[i] \leftarrow E_K(X[i])$
$\Sigma \leftarrow Y[1] \oplus Y[2] \oplus \cdots \oplus Y[m-1] \oplus \mathbf{pad}_n(M[m])$
if $|M[m]| = n$ **then** $X[m] \leftarrow \Sigma \oplus L \cdot \mathtt{u}^{-1}$
    else $X[m] \leftarrow \Sigma$
$T \leftarrow$ the left most τ bits of $E_K(X[m])$
return T

図 65: PMAC のタグ生成アルゴリズム PMAC-$\mathcal{G}_K(\cdot)$.

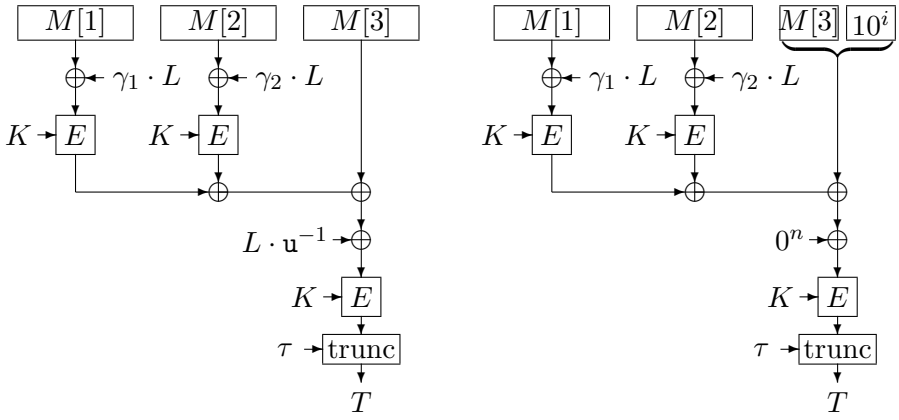


図 66: $M = M[1]M[2]M[3]$ の場合の PMAC-$\mathcal{G}_K(M)$ の動作.

ここで，$\mathbf{ntz}(i)$ は，i をビット表現したときの，最下位ビットから連続して並ぶ 0 の個数である．もしくは，$\mathbf{ntz}(i)$ は，2^z が i を割り切る最大の z である．たとえば，$\mathbf{ntz}(7) = 0, \mathbf{ntz}(8) = 3$ である．

$\gamma_i \cdot L$ は事前計算でいくつか計算しておいてもよいし，必要に応じて計算してもよい．

また，図 65 の 7 行目の関数 $\mathbf{pad}_n : \{0,1\}^{\leq n} \to \{0,1\}^n$ は (1) と同様である．8 行目の $L \cdot \mathtt{u}^{-1}$ は，(3) のように計算される．

- 確認アルゴリズム PMAC-$\mathcal{V} : \mathcal{K}_E \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \mathcal{K}_E$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{PMAC-}\mathcal{V}_K(M, T)$$

を出力する．図 67 にあるように動作する．

Algorithm PMAC-$\mathcal{V}_K(M, T)$
$T' \leftarrow$ PMAC-$\mathcal{G}_K(M)$
if $T = T'$ **then return** accept
else return reject

図 67: PMAC の確認アルゴリズム PMAC-$\mathcal{V}_K(\cdot, \cdot)$.

安全性　Black, Rogaway により，安全性が解析されている [BR02]．ブロック暗号 E が安全な擬似ランダム置換族であれば，PMAC$[E, \tau]$ は，偽造不可能性の意味で安全な MAC であることが示されている．具体的には，以下の定理が示されている [BR02].

定理 8.12. $n, \tau \geq 1$ を整数，$t, q, \sigma \geq 1$ を整数とする．$E : \mathcal{K}_E \times \{0,1\}^n \to \{0,1\}^n$ をブロック暗号とする．このとき，

$$\mathbf{Adv}^{\text{mac}}_{\text{PMAC}[E,\tau]}(t, q, \sigma) \leq \mathbf{Adv}^{\text{prp}}_E(t', q') + \frac{1.5\sigma'^2}{2^n} + \frac{1}{2^\tau}$$

である．ただし，$\sigma' = \sigma + q + 1$, $q' = \sigma + 1$, $t' = t + O(n\sigma)$ であり，質問の長さはブロック単位である．

定理 8.12 は，以下のことを示している： 実行時間 t, 高々 q 回の質問をし，それらの質問が合計で高々 σ ブロックであり，

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{PMAC}[E,\tau]}(A) = \epsilon$$

なる敵 A が存在すると仮定する．$\sigma' = \sigma + q + 1$ とする．このとき，実行時間 $t' = t + O(n\sigma)$, 高々 $q' = \sigma$ 回の質問をし，

$$\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \geq \epsilon - \frac{1.5\sigma'^2}{2^n} - \frac{1}{2^\tau}$$

なる敵 B が存在する．

効率　PMAC の効率は，以下のようにまとめられる．

- 鍵長：ブロック暗号の鍵 $K \in \mathcal{K}_E$ の計 1 つが必要である．
- ブロック暗号鍵スケジューリングの呼び出し回数：1 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$\max\{1, \lceil |M|/n \rceil\}$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：1 回である．
- 並列処理性：並列処理可能である．

標準化状況　NIST に提案されている [WWW9].

8.10　$f9$ (3GPP)

方式　$f9$ はブロック暗号 KASUMI : $\{0,1\}^{128} \times \{0,1\}^{64} \to \{0,1\}^{64}$ を用いる．鍵生成アルゴリズム $f9$-$\mathcal{K}$, タグ生成アルゴリズム $f9$-$\mathcal{G}$, 確認アルゴリズム $f9$-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム $f9$-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K \xleftarrow{R} \{0,1\}^{128}$$

  を出力する．

- タグ生成アルゴリズム $f9\text{-}\mathcal{G} : \mathcal{K}_E \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\{0,1\}^{64}$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^{32}$ である．さらに，32 ビットのカウンタ COUNT, 32 ビットの乱数 FRESH, 1 ビットの direction identifier DIRECTION を入力にもつ．これらは，送受信者の間で共有されている．図 68, 図 69 にあるように動作する．

Algorithm $f9\text{-}\mathcal{G}_K(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M)$
$M \leftarrow \mathtt{pad}_{64}(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M)$
Break M into 64-bit blocks $M[1] \| \cdots \| M[m]$
$Y[0] \leftarrow 0^{64}$
For $i = 1$ to m do:
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow \mathrm{KASUMI}_K(X[i])$
$T \leftarrow \mathrm{KASUMI}_{K \oplus \mathtt{KM}}(Y[1] \oplus \cdots \oplus Y[m])$
$T \leftarrow$ the leftmost 32 bits of T
Return T

図 68: $f9$ のタグ生成アルゴリズム $f9\text{-}\mathcal{G}_K(\cdot)$.

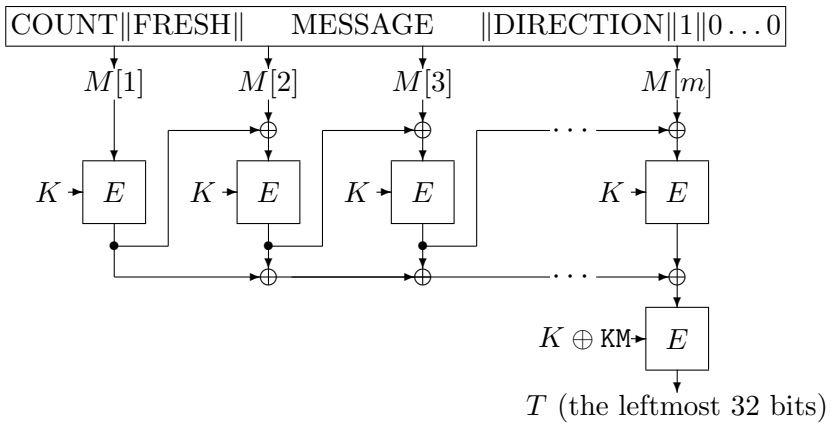


図 69: $f9\text{-}\mathcal{G}_K(M)$ の動作．E は KASUMI である．

2 行目の $\mathtt{pad}_{64}(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M)$ は以下のように動作する：まず，COUNT, FRESH, M, DIRECTION を連結し，次に 1 ビットの “1” を連結し，最後に，全体の長さが 64 ビットの

整数倍になるように，“0” を連結する．すなわち，

$$\mathtt{pad}_{64}(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M) \\ = \mathrm{COUNT}\|\mathrm{FRESH}\|M\|\mathrm{DIRECTION}\|1\|0^{63-(|M|+1 \bmod 64)} .$$

とする．KM は，128 ビットの定数であり，KM = 0xAA...AA である．

- 確認アルゴリズム $f9\text{-}\mathcal{V} : \{0,1\}^{64} \times \{0,1\}^{*} \times \{0,1\}^{32} \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \{0,1\}^{64}$, メッセージ $M \in \{0,1\}^{*}$, タグ $T \in \{0,1\}^{32}$ を入力とし，

$$\text{accept or reject} = f9\text{-}\mathcal{V}_K(M, T)$$

を出力する．さらに，送受信者の間で共有されている 32 ビットのカウンタ COUNT, 32 ビットの乱数 FRESH, 1 ビットの direction identifier DIRECTION を入力にもつ．

図 70 にあるように動作する．

Algorithm $f9\text{-}\mathcal{V}_K(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M, T)$
$T' \leftarrow f9\text{-}\mathcal{G}_K(\mathrm{COUNT}, \mathrm{FRESH}, \mathrm{DIRECTION}, M)$
if $T = T'$ **then return** accept
else return reject

図 70: $f9$ の確認アルゴリズム $f9\text{-}\mathcal{V}_K(\cdot, \cdot)$.

安全性　Hong, Kang, Preneel, Ryu により，KASUMI が安全な擬似ランダム置換であれば，$f9$ が偽造不能性の意味で安全な MAC であることが主張された [HKPR03] が，証明の不備が指摘されている [IK03c].

効率　$f9$ の効率は，以下のようにまとめられる．

- 鍵長：KASUMI の鍵 $K \in \{0,1\}^{64}$ 一つのみである．
- ブロック暗号鍵スケジューリングの呼び出し回数：K と $K \oplus$ KM の 2 回である．
- メッセージ M に対するタグを生成するのにかかるブロック暗号の呼び出し回数：$\lceil |M|/n \rceil + 2$ 回の呼び出しである．
- 事前計算するべきブロック暗号の呼び出し回数：必要ない．
- 並列処理性：並列処理はできない．

標準化状況　3GPP により標準化されている [3GPPa, 3GPPb].

8.11　NMAC, HMAC

8.11.1　NMAC

方式　NMAC (Nested MAC) は反復型ハッシュ関数 F とタグ長 τ をパラメータとする．出力長 n の反復型ハッシュ関数 $F : \{0,1\}^* \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなければならない．これらのパラメータを用いた NMAC を NMAC$[F,\tau]$ と表記する．NMAC$[F,\tau]$ = (NMAC-$\mathcal{K}$, NMAC-$\mathcal{G}$, NMAC-$\mathcal{V}$) の鍵生成アルゴリズム NMAC-$\mathcal{K}$, タグ生成アルゴリズム NMAC-$\mathcal{G}$, 確認アルゴリズム NMAC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム NMAC-$\mathcal{K}$ は確率的アルゴリズムであり，

  $$K_1 \xleftarrow{R} \{0,1\}^n$$

  と

  $$K_2 \xleftarrow{R} \{0,1\}^n$$

  を出力する．

- タグ生成アルゴリズム NMAC-$\mathcal{G}$: $(\{0,1\}^n \times \{0,1\}^n) \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\{0,1\}^n \times \{0,1\}^n$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K_1 \in \{0,1\}^n$, $K_2 \in \{0,1\}^n$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

  $$T = \text{NMAC-}\mathcal{G}_{K_1,K_2}(M) \in \{0,1\}^\tau$$

  を出力する．図 71 にあるように動作する．なお，図 71 で，F_{K_1}, F_{K_2} はそれぞれ反復型ハッシュ関数 F の初期値を K_1, K_2 として計算を行うことを表す．

Algorithm NMAC-$\mathcal{G}_{K_1,K_2}(M)$
$T \leftarrow$ the left most τ bits of $F_{K_1}(F_{K_2}(M))$
Return T

図 71: NMAC のタグ生成アルゴリズム NMAC-$\mathcal{G}_{K_1,K_2}(\cdot)$.

- 確認アルゴリズム NMAC-$\mathcal{V}$: $(\{0,1\}^n \times \{0,1\}^n) \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K_1 \in \{0,1\}^n$, $K_2 \in \{0,1\}^n$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{NMAC-}\mathcal{V}_{K_1,K_2}(M,T)$$

を出力する．図 72 にあるように動作する．

Algorithm NMAC-$\mathcal{V}_{K_1,K_2}(M,T)$
$T' \leftarrow$ NMAC-$\mathcal{G}_{K_1,K_2}(M)$
if $T = T'$ **then return** accept
else return reject

図 72: NMAC の確認アルゴリズム NMAC-$\mathcal{V}_{K_1,K_2}(\cdot,\cdot)$.

安全性 Bellare, Canetti, Krawczyk により安全性が解析されている [BCK96].

まず準備として弱衝突計算困難性 (weak collision resistance) の定義を述べる．関数 $G : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ について，$\mathcal{K}$ を鍵の集合，$\mathcal{D}$, $\mathcal{R}$ をそれぞれ G の定義域，値域とする．

敵 A として，G を計算するオラクルにアクセスできるアルゴリズムを考える．$A^{G(K,\cdot)}$ は $x \in \mathcal{D}$ に対して $G(K,x)$ を返すオラクル $G(K,\cdot)$ をもつ敵を表す．質問は適応的に行う．すなわち，ある質問に対する答を得た後，次の質問を行う．

$A^{G(K,\cdot)}$ が $G(K,x) = G(K,x')$ かつ $x \neq x'$ を満たす x, x' を出力したとき，A は弱衝突計算困難性の意味で関数 G の衝突計算に成功したという．$G : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ の敵 A に対する弱衝突計算困難性の意味での安全性はアドバンテージ $\mathbf{Adv}^{\text{wcr}}_G(A)$ によって評価される．ここで，

$$\mathbf{Adv}^{\text{wcr}}_G(A) = \Pr[K \xleftarrow{\text{R}} \mathcal{K} : A^{G(K,\cdot)} \text{が弱衝突計算困難性の意味で衝突計算に成功}]$$

と定義される．

$G : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ の弱衝突計算困難性の意味での安全性を考える場合に扱う資源は，実行時間 t，オラクルへの質問回数 q，それらの質問の長さ σ である．

$$\mathbf{Adv}^{\text{wcr}}_G(t,q,\sigma) \stackrel{\text{def}}{=} \max_A \{\mathbf{Adv}^{\text{wcr}}_G(A)\}$$

と定義される．ただし，実行時間 t，オラクルへの質問回数 q，それらの質問の長さ σ のすべての敵 A について最大値をとる．

[BCK96] における NMAC の安全性についての議論より，以下の定理が成立することが分かる．

定理 8.13.

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{NMAC}[F,\tau]}(t,q,\sigma) \leq \mathbf{Adv}^{\mathrm{mac}}_{f^{(\tau)}}(t+O(q\,\sigma),q,b) + \mathbf{Adv}^{\mathrm{wcr}}_{F}(t,q,\sigma)$$

ここで $f^{(\tau)}$ は圧縮関数 f の出力の左から τ ビットのみを出力する関数である．また $O(q\,\sigma)$ については，圧縮関数 f の *1* 回の計算時間を定数と仮定している．

効率　NMAC の効率は以下のようにまとめられる．

- 鍵長：ハッシュ関数の初期値として使用される鍵 $K_1, K_2 \in \{0,1\}^n$ の二つである．
- メッセージ M に対するタグを生成するのにかかるハッシュ関数の圧縮関数の呼び出し回数：ハッシュ関数 $F : \{0,1\}^* \to \{0,1\}^n$ の圧縮関数を $f : \{0,1\}^n \times \{0,1\}^b \to \{0,1\}^n$ とする．ここで，b は圧縮関数に入力されるメッセージのブロック長である．F が MD5, RIPEMD-160, SHA-1, SHA-256, SHA-384, SHA-512 などのハッシュ関数であることを想定すれば，F_{K_2} の圧縮関数の呼び出し回数は $\lceil |M|/b \rceil$ 回または，$\lceil |M|/b \rceil + 1$ 回である．また，F_{K_1} の圧縮関数の呼び出し回数は 1 回である．
- 並列処理性：並列処理はできない．

標準化状況　NMAC は応用を指向した HMAC の理論的基礎として設計されたものであり，それ自体については標準化の実績はない．

8.11.2　HMAC

方式　HMAC (Hash-based MAC) は反復型ハッシュ関数 F とタグ長 τ をパラメータとする．出力長 n の反復型ハッシュ関数 $F : \{0,1\}^* \to \{0,1\}^n$ を用いた場合は，$\tau \leq n$ でなければならない．これらのパラメータを用いた HMAC を HMAC$[F,\tau]$ と表記する．HMAC$[F,\tau]$ = (HMAC-$\mathcal{K}$, HMAC-$\mathcal{G}$, HMAC-$\mathcal{V}$) の鍵生成アルゴリズム HMAC-$\mathcal{K}$，タグ生成アルゴリズム HMAC-$\mathcal{G}$，確認アルゴリズム HMAC-$\mathcal{V}$ はそれぞれ以下のように動作する．

- 鍵生成アルゴリズム HMAC-$\mathcal{K}$ は確率的アルゴリズムであり，

$$K \xleftarrow{R} \{0,1\}^n$$

を出力する．

- タグ生成アルゴリズム HMAC-$\mathcal{G} : \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^\tau$ は決定的アルゴリズムであり，鍵空間は $\{0,1\}^n$, メッセージ空間は $\{0,1\}^*$, タグ空間は $\{0,1\}^\tau$ である．すなわち，鍵 $K \in \{0,1\}^n$ とメッセージ $M \in \{0,1\}^*$ を入力とし，タグ

$$T = \text{HMAC-}\mathcal{G}_K(M) \in \{0,1\}^\tau$$

を出力する．図 73 にあるように動作する．

Algorithm HMAC-$\mathcal{G}_K(M)$
$T \leftarrow$ the left most τ bits of $F(\overline{K} \oplus \texttt{opad}, (F(\overline{K} \oplus \texttt{ipad}, M))$
Return T

図 73: HMAC のタグ生成アルゴリズム HMAC-$\mathcal{G}_K(\cdot)$.

ハッシュ関数 $F : \{0,1\}^* \to \{0,1\}^n$ の圧縮関数を $f : \{0,1\}^n \times \{0,1\}^b \to \{0,1\}^n$ とする．ここで，b は圧縮関数に入力されるメッセージのブロック長である．図 73 で，$\overline{K}$ は，長さが b ビットになるまで K に 0 を付加して得られる系列である．opad は，00110110 を長さが b ビットになるまで繰り返して得られる系列である．ipad は，01011100 を長さが b ビットになるまで繰り返して得られる系列である．

- 確認アルゴリズム HMAC-$\mathcal{V} : \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^\tau \to$ accept or reject は決定的アルゴリズムであり，鍵 $K \in \{0,1\}^n$, メッセージ $M \in \{0,1\}^*$, タグ $T \in \{0,1\}^\tau$ を入力とし，

$$\text{accept or reject} = \text{HMAC-}\mathcal{V}_K(M,T)$$

を出力する．図 74 にあるように動作する．

Algorithm HMAC-$\mathcal{V}_K(M, T)$
$T' \leftarrow$ HMAC-$\mathcal{G}_K(M)$
if $T = T'$ **then return** accept
else return reject

図 74: HMAC の確認アルゴリズム HMAC-$\mathcal{V}_K(\cdot, \cdot)$.

安全性 Bellare, Canetti, Krawczyk により，安全性が解析されている [BCK96].

HMAC の安全性は NMAC の安全性に基づく．$K_1 = f(IV, \overline{K} \oplus \texttt{opad})$, $K_2 = f(IV, \overline{K} \oplus \texttt{ipad})$ とすれば,

$$F(\overline{K} \oplus \texttt{opad}, (F(\overline{K} \oplus \texttt{ipad}, M)) = F_{K_1}(F_{K_2}(M))$$

となる[2]．ここで，IV はハッシュ関数 F で定義されている初期値である．NMAC では鍵 K_1, K_2 が無作為に選択されるので，f が K を鍵とする疑似ランダム関数であるという仮定を設ければ，NMAC の安全性の解析は HMAC に適用できることになる．

opad, ipad はハミング距離が大きくなるように選択されている．

効率 HMAC の効率は以下のようにまとめられる．

- 鍵長：ハッシュ関数の入力の一部として使用される鍵 $K \in \{0,1\}^n$ の一つである．この鍵の長さはハッシュ関数 F の出力長と等しい．
- メッセージ M に対するタグを生成するのにかかるハッシュ関数の圧縮関数の呼び出し回数：一力ブロック $\overline{K} \oplus \texttt{opad}$ と $\overline{K} \oplus \texttt{ipad}$ の処理のため，NMAC での呼び出し回数より 2 回多い．ただし，K を変更しない限り $K_1 = f(IV, \overline{K} \oplus \texttt{opad})$, $K_2 = f(IV, \overline{K} \oplus \texttt{ipad})$ を計算して保持しておけば，NMAC と同じ回数となる．
- 並列処理性：並列処理はできない．

[2][BCK96] ではこのように主張されているが，NMAC と HMAC とでは，F への入力長が異なることになるので，パディングを考慮すると，通常はこの等式は成立しない．ただし，この等式が成立するように NMAC のパディングを変更することは可能であり，このような変更は NMAC の安全性の証明に影響を与えない．

標準化状況　米国情報処理標準であり，FIPS198 に制定されている [FIPS198]．NESSIE の portfolio に含まれている [WWW8]．IPSec では，任意の認証アルゴリズムの実装が許されているが，最低限 MD5 と SHA-1 による HMAC の実装がなされなければならない [RFC2402]．

9　まとめ

本報告では，暗号利用モードと MAC に関して，その動作原理や安全性の考え方などについて解説すると共に，各種標準化作業や学術出版物などで知られている方式を調査し，それらの工業的，学術的性質，安全性，処理効率などについてまとめた．また，それらを用途別に，秘匿に関する利用モード，認証暗号に関する利用モード，ディスクセクタ向け暗号利用モード認証に関する利用モードに分類した．

提供される機能は，利用モードにより異なるため，利用モードを選択する際にはその性質について十分理解しておく必要がある．また，その際の参考として本報告書が役立つことを期待する．

表 5: 秘匿に関する暗号利用モードのまとめ

	秘匿性に関する注意点	1 ビット エラーの 伝播範囲	処理 速度	並列性		復号 関数 の実装	機能に 関する コメント
				暗号化	復号		
ECB	・暗号文を見るだけで平文ブロックが同じ値であるか否かを判定できる。よって、1) 平文がブロックサイズより小さい、2) 平文ブロックが全て異なる、などの特別な理由がない限り用いるべきでない。特に、長い文章を暗号化する際の利用は避けた方がよい。	1 ブロック	1	有り	有り	必要	
CBC	・$2^{n/2}$ ブロック程度以上の平文を暗号化すると、中間一致攻撃により、暗号文を見るだけで平文に関する 1 ブロック分の情報が得られる可能性がある。	1 ブロック +1 ビット	1	無し	有り	必要	
k-CFB	・$2^{n/2}$ ブロック程度以上の平文を暗号化すると、中間一致攻撃により、暗号文を見るだけで平文に関する 1 ブロック分（k ビット）の情報が得られる可能性がある。 ・k が小さい場合、初期値と平文の組み合わせによっては、鍵ストリームの周期が極端に小さくなる恐れがある。	$(\lfloor \frac{n}{k} \rfloor \sim \lceil \frac{n}{k} \rceil)$ 平文ブロック + 1 ビット（1 平文ブロック =k ビット）	$\frac{k}{n}$	無し	有り	不要	k ビット単位の自動同期回復機能がある。
OFB (k-OFB)	・$k < n$ の場合 $2^{n/2}$ ブロック程度で周期を形成。（k が小さい場合、IV によっては、暗号文から容易に鍵の候補を絞り込むことが可能。） ・初期値の運用を誤ると重大な欠陥につながる恐れがあり、注意が必要。	1 ビット	1	無し	無し	不要	
CTR	・初期値の運用を誤ると重大な欠陥につながる恐れがあり、注意が必要。	1 ビット	1	有り	有り	不要	

参考文献

[3GPPa] 3GPP TS 35.201 v 3.1.1. Specification of the 3GPP confidentiality and integrity algorithms, Document 1: $f8$ and $f9$ specification. Available at `http://www.3gpp.org/tb/other/algorithms.htm`.

[3GPPb] 3GPP TS 35.202 v 3.1.1. Specification of the 3GPP confidentiality and integrity algorithms, Document 2: KASUMI specification. Available at `http://www.3gpp.org/tb/other/algorithms.htm`.

[AGS97] V. Afanassiev, C. Gehrmann, and B. Smeets, "Fast Message Authentication Using Efficient Polynomial Evaluation," *Fast Software Encryption, 4th International Workshop, FSE'97, Haifa, Israel, January 20–22, 1997, Proceedings*, ed. E. Biham, pp. 190–204, Lecture Notes in Computer Science vol. 1267, Springer-Verlag, 1997.

[AGPS02] A. Alkassar, A. Geraldy, B. Pfitzmann, and A. R. Sadeghi, "Optimized Self-synchronizing Mode of Operation," *Fast Software Encryption, 8th International Workshop, FSE2001, Yokohama, Japan, April 2–4, 2001, Revised Papers*, ed. M. Matsui, pp. 78–91, Lecture Notes in Computer Science vol. 2355, Springer-Verlag, 2002.

[ANSIX3.106] ANSI X 3.106, American National Standard for Information Systems – Data Encryption Algorithm – Modes of Operation," American National Standard Institute, 1983.

[ANSIX3.92] ANSI X 3.92, American National Standard for Information Systems – Data Encryption Algorithm," American National Standard Institute, 1981.

[AB99] J.H. An and M. Bellare, "Constructing VIL-MACs from FIL-MACs: Message Authentication under Weakened Assumptions," *Advances in Cryptology — CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15–19, 1999. Proceedings*, ed. M. Wiener, pp. 252–269, Lecture Notes in Computer Science vol. 1666, Springer-Verlag, 1999.

[AB01] J.H. An and M. Bellare, "Does Encryption with Redundancy Provide Authenticity?" *Advances in Cryptology, — EUROCRYPT*

2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6–10, 2001, Proceedings, ed. B. Pfitzmann, pp. 512–528, Lecture Notes in Computer Science vol. 2045, Springer-Verlag, 2001.

[BA01] A. A. Belal and M.A.Abdel-Gawad, "2D-Encryption Mode," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2001, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[BGR95] M. Bellare, R. Guérin, and P. Rogaway, "XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions," *Advances in Cryptology — CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1995. Proceedings*, ed. D. Coppersmith, pp. 15–28, Lecture Notes in Computer Science vol. 963, Springer-Verlag, 1995.

[BCK96] M. Bellare, R. Canetti, and H. Krawczyk, "Keying Hash Functions for Message Authentication," *Advances in Cryptology — CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1996. Proceedings*, ed. N. Koblitz, pp. 1–15, Lecture Notes in Computer Science vol. 1109, Springer-Verlag, 1996.

[BDJR97] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. Proceedings of *The 38th Annual Symposium on Foundations of Computer Science, FOCS '97,* pp. 394–405, IEEE, 1997.

[BN00] M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," *Advances in Cryptology — ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3–7, 2000 Proceedings*, ed. T. Okamoto, pp. 531–545, Lecture Notes in Computer Science vol. 1976, Springer-Verlag, 2000.

[BKR00] M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *JCSS,* Vol. 61,

No. 3, pp. 362–399, 2000. Earlier version in *Advances in Cryptology — CRYPTO '94, LNCS 839,* pp. 341–358, Springer-Verlag, 1994.

[BPR05] M. Bellare, K. Pietrzak, and P. Rogaway. Improved Security Analyses for CBC MACs. *Advances in Cryptology — CRYPTO 2005, LNCS 3621,* pp. 527–545, Springer-Verlag, 2005.

[BBKN01] M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre, "Online Ciphers and the Hash-CBC Construction," *Advances in Cryptology — CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, Aug. 2001, Proceedings*, ed. J. Kilian, pp.292–309, Lecture Notes in Computer Science vol. 2139, Springer-Verlag, 2001.

[BK03] M. Bellare, and T. Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. *Advances in Cryptology — EUROCRYPT 2003, LNCS 2656,* pp. 491–506, Springer-Verlag, 2003.

[BRW03] M. Bellare, P. Rogaway, and D. Wagner, "A Conventional Authenticated-Encryption Mode," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2003, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[BB+95] A. Berendschot, B. den Boer, J. Boly, A. Bosselaers, J. Brandt, D. Chaum, I. Damgaard, M. Dichtl, W. Fumy, M. van der Ham, C. Jansen, P. Landrock, B. Preneel, G. Roelofsen, P. de Rooij, J. Vandewalle, *Final Report of Race Integrity Primitives, Lecture Notes in Computer Science*, vol. 1007, Springer-Verlag, 1995.

[B96] E. Biham, "Cryptanalysis of Triple-Modes of Operation," Technion technical report CS885, 1996, available at `http://www.cs.technion.ac.il/~biham/publications.html`.

[BHKKR99] J. Black, S. Halevi, H. Krawczyk, T. Krovets, and P. Rogaway, "UMAC: Fast and Secure Message Authentication," *Advances in Cryptology — CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15–19,*

1999. Proceedings, ed. M. Wiener, pp. 216–233, Lecture Notes in Computer Science vol. 1666, Springer-Verlag, 1999.

[BR00] J. Black and P. Rogaway, "CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions," *Advances in Cryptology — CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 2000. Proceedings*, ed. M. Bellare, pp. 197–215, Lecture Notes in Computer Science vol. 1880, Springer-Verlag, 2000.

[BR01] J. Black and P. Rogaway. Comments to NIST concerning AES modes of operations: A suggestion for handling arbitrary-length messages with the CBC MAC. *Second Modes of Operation Workshop.* Available at `http://www.cs.ucdavis.edu/~rogaway/`.

[BR02] J. Black and P. Rogaway, "A Block-Cipher Modes of Operation for Parallelizable Message Authentication," *Advances in Cryptology — EUROCRYPT 2002, International Conference on the Theory and Application of Cryptographic Techniques, Amsterdam, the Netherlands, April 28–May 2, 2002. Proceedings*, ed. L. Knudsen, pp. 384–397, Lecture Notes in Computer Science vol. 2332, Springer-Verlag, 2002.

[BRS02] J. Black, P. Rogaway, and T. Shrimpton, "Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV," *Advances in Cryptology — CRYPT 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 2002, Proceedings*, ed. M. Yung, pp. 320–335, Lecture Notes in Computer Science vol. 2442, Springer-Verlag, 2002.

[B02] J. Black. Comments on the RMAC algorithm. Comments to NIST. Available at `http://csrc.nist.gov/CryptoToolkit/modes/comments/`.

[C78] C. M. Campbell, "Design and Specification of Cryptographic Capabilities," *Computer Security and the Data Encryption Standard*, (ed.) D. K. Brandstad, National Bureau of Standards Special Publications 500-27, U. S. Department of Commerce, February 1978, pp. 54–66.

[CW79] L. Carter and M. Wegman, "Universal Hash Functions," *Journal of Computer and System Sciences*, vol. 18, 1979.

[D93] J. Daemen, "Limitations of the Even-Mansour Construction," *Advances in Cryptology — ASIACRYPT '91, International Conference on the Theory and Application of Cryptology*, eds. H. Imai, R.L. Rivest, and T. Matsumoto, pp. 495–499, Lecture Notes in Computer Science vol. 739, Springer-Verlag, 1993.

[DR99] J. Daemen and V. Rijmen, *AES Proposal: Rijndael*, AES Algorithm Submission, September 3, 1999, available at `http://www.nist.gov/CryptoToolkit`.

[DH79] W. Diffie and M. E. Hellman, "Privacy and Authentication: An Introduction to Cryptography," Proceedings of the IEEE 67/3, 1979, pp. 397–427.

[SP800-38A] M. Dworkin, National Institute of Standards and Technology, Special Publication 800-38A, 2001 Edition, Recommendation for Block Cipher Modes of Operation: Methods and Techniques, December 2001.

[SP800-38B] M. Dworkin, National Institute of Standards and Technology, Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, May 2005.

[SP800-38C] M. Dworkin, National Institute of Standards and Technology, Special Publication 800-38C, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality, May, 2004.

[EM97] S. Even and Y. Mansour, "A Construction of a Cipher from a Single Pseudorandom Permutation," *J. of Cryptology*, 10(3) 151–161, Summer 1997.

[FIPS46-3] National Institute of Standards and Technology, Federal Information Processing Standards Publication 46-3, Data Encryption Standard (DES).

[FIPS81] National Institute of Standards and Technology, Federal Information Processing Standards Publication 81, DES Modes of Operation (DES), 1980.

[FIPS113] National Institute of Standards and Technology, Federal Information Processing Standards Publication 113, Computer data authentication, 1994.

[FIPS197] National Institute of Standards and Technology, Federal Information Processing Standards Publication 197, Advanced Encryption Standard (AES).

[FIPS198] National Institute of Standards and Technology, Federal Information Processing Standards Publication 198, The Keyed-Hash Message Authentication Code (HMAC), 2002.

[FH03] S. Frankel, and H. Herbert. The AES-XCBC-MAC-96 algorithm and its use with IPsec. Available at `http://www.ieft.org/`.

[FMP03] P.-A. Fouque, G. Martinet, and G. Poupard, "Practical Symmetic On-line Encryption," *FSE2003, Tenth Annual Workshop on Fast Software Encryption, February 24–26, 2003, AF-Borgen, Lund, Sweden. Pre-proceedings*, pp. 379–392, Department of Information Technology of Lund Institute of Technology, Lund University, 2003.

[GD99] V. Gligor and P. Donescu, "Integrity-aware PCBC Encryption Schemes," *Security Protocols, 7th International Workshop Cambridge, UK, April 19–21, 1999 Proceedings*, eds. B. Christianson, B. Crispo, J.A. Malcolm, and M. Roe, Lecture Notes in Computer Science vol. 1796, Springer-Verlag, 2000.

[GD00] V. D. Gligor and P. Donescu, "On Message Integrity in Symmetric Encryption," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2000, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[GD01a] V.D. Gligor and P. Donescu, "Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes,"

Fast Software Encryption, 8th International Workshop, FSE2001, Yokohama, Japan, April 2–4, 2001. Revised Papers, ed. M. Matsui, pp. 92–108, Lecture Notes in Computer Science vol. 2355, Springer-Verlag, 2002.

[GD01b] V. D. Gligor and P. Donescu, "Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2001, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali, "How to Construct Random Functions," *Journal of the ACM*, Vol. 33, No. 4, 792–807, Octorber 1986.

[GM84] S. Goldwasser and S. Micali, "Probabilistic Encryption," J. Computer & System Sciences, 28: pp.270–299, 1984.

[HK97] S. Halevi and H. Krawczyk, "MMH: Software Message Authentication in the Gbit/second Rates," *Fast Software Encryption, 4th International Workshop, FSE'97, Haifa, Israel, January 20–22, 1997, Proceedings*, ed. E. Biham, pp. 172–189, Lecture Notes in Computer Science vol. 1267, Springer-Verlag, 1997.

[HR03a] S. Halevi and P. Rogaway, "A Parallelizable Enciphering Mode," Working draft for SISWG, Security in Storage Working Group, March 2003, document available at `http://www.siswg.org/docs/`.

[HR03b] S. Halevi, and P. Rogaway. A tweakable enciphering mode. *Advances in Cryptology — CRYPTO 2003, LNCS 2729,* pp. 482–499, Springer-Verlag, 2003.

[HP99] H. Handschuh and B. Preneel, "On the Security of Double and 2-key Triple Modes of Operation," *Fast Software Encryption, 6th International Workshop, FSE'99, Rome, Italy, March 1999, Proceedings*, ed. L. Knudsen, pp. 215–230, Lecture Notes in Computer Science vol. 1636, Springer-Verlag, 1999.

[H01c] H. Hellström, "Propagating Cipher Feedback," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2001, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[H01a] H. M. Heys, "Delay Characteristics of Statistical Cipher Feedback Mode," *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing — PACRIM 2001, Victoria, British Columbia*, 2001, available at `http://www.engr.mun.ca/~howard/Research/Papers/index.html`.

[H01b] H. M. Heys, "An Analysis of the Statistical Self-Synchronization of Stream Ciphers," *Proceedings of INFOCOM 2001, Anchorage, Alaska*, pp. 897–904, 2001, available at `http://www.engr.mun.ca/~howard/Research/Papers/index.html`.

[H03a] H. M. Heys, "Analysis of the Statistical Cipher Feedback Mode of Block Ciphers," *IEEE Transactions on Computers*, vol.=52, no. 1, pp. 77–92, January 2003, available at `http://www.engr.mun.ca/~howard/Research/Papers/index.html`.

[H03b] P. Hoffman. The AES-XCBC-PRF-128 algorithm for IKE. Available at `http://www.ieft.org/`.

[HKPR03] D. Hong, J-S. Kang, B. Preneel, and H. Ryu. A concrete security anallysis for 3GPP-MAC. *Fast Software Encryption, FSE 2003, LNCS 2887,* pp. 154–169, Springer-Verlag, 2003.

[ISO8372] ISO 8372: 1987, Information Processing – Modes of operation for a 64-bit block cipher algorithm (ANSI X3.92-1981 を参照している).

[ISOIEC9797-1] ISO/IEC 9797-1. Information technology — security techniques — data integrity mechanism using a cryptographic check function employing a block cipher algorithm. International Organization for Standards, Geneva, Switzerland, 1999. Second edition.

[ISO10116] ISO/IEC 10116:1997, Information technology – Security techniques – Modes of operation for an n-bit block cipher algorithm, 2002-6-26.

[IK03a] T. Iwata and K. Kurosawa. OMAC: One-Key CBC MAC. *Fast Software Encryption, FSE 2003, LNCS 2887,* pp. 129–153, Springer-Verlag, 2003.

[IK03b] T. Iwata and K. Kurosawa. Stronger security bounds for OMAC, TMAC and XCBC. *Progress in Cryptology — INDOCRYPT 2003, LNCS 2904,* pp. 402–415, 2003.

[IK03c] T. Iwata and K. Kurosawa. On the correctness of security proofs for the 3GPP confidentiality and integrity algorithms. *Ninth IMA International Conference on Cryptography and Coding, LNCS 2898,* pp. 306–318, Springer-Verlag, 2003.

[JJ+02a] É. Jaulmes, A. Joux, and F. Valette. On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction. *Fast Software Encryption, FSE 2002, LNCS 2365,* pp. 237–251, Springer-Verlag, 2002.

[JJ+02b] É. Jaulmes, A. Joux, and F. Valette. On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction. Full version. Available at Cryptology ePrint Archive, Report 2001/074, `http://eprint.iacr.org/`.

[JJ+02c] É. Jaulmes, A. Joux, and F. Valette. RMAC, A randomized MAC beyond the birthday paradox limit. *Second Modes of Operation Workshop,* 2001. Available at `http://csrc.nist.gov/CryptoToolkit/modes`.

[JMV02] A. Joux, G. Martinet, and F. Valette, "Blockwise Adaptive Attackers: Revisiting the (In)security of Some Provably Secure Encryption Modes: CBC, GEM, IACBC," *Advances in Cryptology — CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, Aug. 18–22, 2002. Proceedings*, ed. M. Yung, pp. 17–30, Lecture Notes in Computer Science vol. 2442, Springer-Verlag, 2002.

[J03] A. Joux, "Cryptanalysis of the EMD Mode of Operation," *Advances in Cryptology — EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4–8, 2003, Proceedings*, ed. E.Biham, pp. 1–16, Lecture Notes in Computer Science vol. 2656, Springer-Verlag, 2003.

[JKRW01] O. Jung, S. Kuhn, C. Ruland, and K. Wollenweber, "Enhanced modes of operation for the encryption in high-speed networks and their impact on QoS," *Information Security and Privacy: 6th Australasian Conference, ACISP 2001, Sydney, Australia, July 11–13, 2001, Proceedings*, eds. V. Varadharajan and Y. Mu, pp. 344–359, Lecture Notes in Computer Science vol. 2119, Springer-Verlag, 2001.

[JR99] O. Jung and C. Ruland, "Encryption with statistical self-synchronization in synchronous broadbad networks," *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES '99, Worcester, MA, USA, August 12–13, 1999, Proceedings*, eds. C.K.Koç and C.Paar , pp. 340–352, Lecture Notes in Computer Science vol. 1717, Springer-Verlag, 1999.

[J02] J. Jonsson, "On the Security of CTR + CBC-MAC," *Selected Areas in Cryptography, 9th Annual Workshop, SAC 2002, St. John's, Newfoundland, Canada, Aug. 2002, Revised Papers*, ed. K. Nyberg and H. Heys, pp. 76–93, Lecture Notes in Computer Science vol. 2595, Springer-Verlag, 2002.

[J00] C. S. Jutla, "Encryption Modes with Almost Free Message Integrity," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2000, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[J01] C.S. Jutla,"Encryption Modes with Almost Free Message Integrity," *Advances in Cryptology — EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6–10, 2001. Proceedings*, ed. B. Pfitzmann, pp. 529–544, Lecture Notes in Computer Science vol. 2045, Springer-Verlag, 2001.

[KY00a] J. Katz and M. Yung, "Complete Characterization of Security Notions for Probabilistic Private-key Encryption," Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing, ACM, 2000.

[KY00b] J. Katz and M. Yung, "Unforgeable Encryption and Chosen Cipher Secure Modes of Operation," *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 2000, Proceedings*, ed. B. Schneier, pp. 284–299, Lecture Notes in Computer Science vol. 1978, Springer-Verlag, 2001.

[KR96] J. Kilian and P. Rogaway, "How to Protect DES against Exhaustive Search (an Analysis of DESX)," *Advances in Cryptology — CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1996. Proceedings*, ed. N. Koblitz, pp. 252–267, Lecture Notes in Computer Science vol. 1109, Springer-Verlag, 1996.

[K00] L. R. Knudsen, "Block Chaining Modes of Operation," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2000, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[K02] L.R. Knudsen. Analysis of RMAC. Comments to NIST. Available at `http://csrc.nist.gov/CryptoToolkit/modes/comments/`.

[KVW03] T. Kohno, J. Viega, and D. Whiting, "The CWC Authenticated Encryption (Associated Data) Mode," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2003, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[KI03] K. Kurosawa and T. Iwata, "TMAC, Two-Key CBC MAC," *Topics in Cryptology — CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, April 13–17, 2003. Proceedings*, ed. M. Joye, pp. 33–49, Lecture Notes in Computer Science vol. 2612, Springer-Verlag, 2003.

[LN94] R. Lidl and H. Niederreiter. Introduction to finite fields and their applications, revised edition. Cambridge University Press, 1994.

[LRW00] H. Lipmaa, P. Rogaway, and D. Wagner, "Comments to NIST Concerning AES Modes of Operations: CTR-mode Encryption," available at `http://csrc.nist.gov/`.

[LR02] M. Liskov, R.L. Rivest, and D. Wagner, "Tweakable Block Ciphers," *Advances in Cryptology — CRYPTO 2002, 22nd Annual International Cryptology Conference Santa Barbara, California, USA, August 18–22, 2002. Proceedings*, ed. M. Yung, pp. 31–46, Lecture Notes in Computer Science vol. 2442, Springer-Verlag, 2002.

[LR88] M. Luby and C. Rackoff, "How to Construct Pseudorandom Permutations from Pseudorandom Functions," *SIAM J. Comput.*, vol. 17, no. 2, April 1988.

[L96] S.Lucks, "Faster Luby-Rackoff Ciphers," *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 1996, Proceedings*, ed. D. Gollmann, pp. 189–203, Lecture Notes in Computer Science vol. 1039, Springer-Verlag, 1996.

[M91] U. M. Maurer, "New Approaches to the Design of Self-Synchronizing Stream Ciphers," *Advances in Cryptology — EUROCRYPT '91, Brighton, UK*, ed. D.W.Davies, pp. 458–471, Lecture Notes in Computer Science vol. 547, Springer-Verlag, 1991.

[MRS88] S. Micali, C. Rackoff, and R. Sloan, "The notion of security for probabilistic cryptosystems," *SIAM J. of Computing*, April 1988.

[M02] C.J. Mitchell, "The Security of Two-key DESX," COSIC Seminar, Katholieke Universiteit Leuven, 15th March 2002, Leuven, Belgium.

[MI02a] S. Moriai and H. Imai, "2-Key XCBC: The CBC-MAC for Arbitrary Length Messages by the Two-key Construction," a talk at the *Recent Results* session of *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4–6, 2002.*

[MI02b] S. Moriai and H. Imai, "2-Key XCBC: The CBC MAC for Arbitrary-Length Messages by the Two-Key Construction," *Proceedings of SCIS2002, The 2002 Symposium on Cryptography and Information Security*, The Institute of Electronics, Information and Communication Engineers, 2002 (in Japanese).

[MV04] D. McGrew, J. Viega, "The Security and Performance of the Galois/Counter Mode (GCM) of Operation," *Progress in Cryptology — INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings.* pp. 343-355, Lecture Notes in Computer Science vol. 3348, Springer-Verlag, 2004. Full version is available at `http://eprint.iacr.org/2004/193/`.

[MV05] D. McGrew, J. Viega, "The Galois/Counter Mode of Operation (GCM)," May 2005, available at `http://csrc.nist.gov/CryotoToolKit/modes/proposedmodes/`.

[NR99] M. Naor and O. Reingold. On the construction of pseudorandom permutations: Luby-Rackoff revised. *J. Cryptology,* vol. 12, no. 1, pp. 29–66, Springer-Verlag, 1999.

[NR03] M. Naor and O. Reingold, "A Pseudo-Random Encryption Mode," Working draft for SISWG, Security in Storage Working Group, document available at `http://www.siswg.org/docs/`.

[PR00] E. Petrank and C. Rackoff. CBC MAC for real-time data sources. *J.Cryptology,* Vol. 13, No. 3, pp. 315–338, Springer-Verlag, 2000.

[PGV94] B. Preneel, R. Govaerts, and J. Vandewalle, "Hash functions based on block ciphers: A synthetic approach," *Advances in Cryptology — CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1993. Proceedings*, ed. D.R. Stinson, pp. 368–378, Lecture Notes in Computer Science vol. 773, Springer-Verlag, 1994.

[PvO96] B. Preneel and P. C. van Oorschot. On the security of two MAC algorithms. *Advances in Cryptology — EUROCRYPT '96, LNCS 1070,* pp. 19–32, Springer-Verlag, 1996.

[RFC2040] R. Baldwin and R. Rivest, "The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms," RFC 2040 (1996), available at `http://www.ietf.org/rfc/rfc2040.txt`.

[RFC2402] S. Kent, "IP Authentication Header," RFC 2402 (1998), available at `http://www.ietf.org/rfc/rfc2402.txt`.

[R97] R.L. Rivest, "All-Or-Nothing Encryption and the Package Transform," *Fast Software Encryption, 4th International Workshop, FSE'97, Haifa, Israel, January 20–22, 1997, Proceedings*, ed. E. Biham, pp. 210–218, Lecture Notes in Computer Science vol. 1267, Springer-Verlag, 1997.

[R95] P. Rogaway. Bucket hashing and its application to fast message authentication. *Advances in Cryptology — CRYPTO '95, LNCS 963,* pp. 29–42, Springer-Verlag, 1995.

[RBBK01a] P. Rogaway, M. Bellare, J. Black, and T. Krovetz, "OCB: A Block-cipher Mode of Operation for Efficient Authenticated Encryption," *Eighth ACM conference on computer and communications security* CCS-8, ACM Press, 2001.

[RBBK01b] P. Rogaway, M. Bellare, J. Black, and T. Krovetz, "OCB: A Block-cipher Mode of Operation for Efficient Authenticated Encryption," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2001, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[R02a] P. Rogaway. Comments on NIST's RMAC proposal. Comments to NIST. Available at `http://www.cs.ucdavis.edu/~rogaway/xcbc/index.html`. Also available at `http://csrc.nist.gov/CryptoToolkit/modes/comments/`.

[R02b] P. Rogaway, "The EMD Mode of Operation (A Tweaked, Wide-Blocksize, Strong PRP)," Cryptology ePrint Archive 2002/148, `http://eprint.iacr.org/2002/148/`.

[RW03] P. Rogaway and D. Wagner, "A Critique of CCM," available at `http://www.cs.berkeley.edu/~daw/papers/ccm.html`.

[V02] S. Vaudenay, "Security Flaws Induced by CBC Padding — Applications to SSL, IPSEC, WTLS...," *Advances in Cryptology — EUROCRYPT 2002, International Conference on the Theory and Application of Cryptographic Techniques, Amsterdam, the Netherlands, April 28–May 2, 2002, proceedings*, ed. L. Knudsen, pp. 534–

546, Lecture Notes in Computer Science vol. 2332, Springer-Verlag, 2002.

[WC81] M. Wegman and L. Carter, "New Hash Functions And Their Use in Authentication And Set Equality," *Journal of Computer and System Sciences*, vol. 22, 1981.

[W98] D. Wagner, "Cryptanalysis of Some Recently-proposed Multiple Modes of Operation," *Fast Software Encryption, 5th International Workshop, FSE'98, Paris, France, March 1998. Proceedingds*, ed. S. Vaudenay, pp. 254–269, Lecture Notes in Computer Science vol. 1372, Springer-Verlag, 1998.

[W02a] D. Wagner, Comments on RMAC. Comments to NIST. Available at `http://csrc.nist.gov/CryptoToolkit/modes/comments/`.

[W02b] D. Wagner, "OFB and CFB modes: A Cautionary Note Regarding IV Selection," Rump-session Talk at CRYPTO 2002, 22nd Annual International Cryptology Conference Santa Barbara, California, USA, Aug. 18–22, 2002.

[WHF02] D. Whiting, R. Housley, and F. Ferguson, "Counter with CBC-MAC (CCM) — AES Mode of Operation," Submitted document for Modes of Operation for Symmetric Key Block Ciphers, Computer Security Resource Center of Information Technology Laboratory, National Institute of Standards and Technology, 2002, available at `http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/`.

[WHR02] D. Whiting, R. Housley, and N. Ferguson, "AES Encryption & Authentication Using CTR Mode & CBC-MAC," *IEEE P802.11 doc 02/001r2*, May 2002.

[WWW1] SISWG, Security in Storage Working Group, An IEEE Information Assurance Activity, URL at `http://www.siswg.org/`

[WWW2] CRYPTREC, Cryptography Research and Evaluation Committees, `http://www.ipa.go.jp/security/enc/CRYPTREC/`

[WWW3] CRYPTREC, Cryptography Research and Evaluation Committees, `http://cryptrec.nict.go.jp/`

[WWW4] http://www.itl.nist.gov/fipspubs/

[WWW5] http://www.nist.gov/

[WWW6] http://csrc.nist.gov/publications/nistpubs/

[WWW7] http://www.ansi.org/

[WWW8] http://www.cosic.esat.kuleuven.ac.be/nessie/

[WWW9] http://www.nist.gov/

付録 6

擬似乱数検定のための CRYPTREC ミニマムセット仕様書

平成 18 年 2 月

CRYPTREC 事務局

1 背景

電子政府推奨暗号リストにおける擬似乱数系では、SHA-1 を使った擬似乱数生成器が例示されている。しかし、用途によっては、例示された擬似乱数生成器以外でも、暗号学的に安全性が確認できれば、用途によっては利用できる場合もある。そこで電子政府で使用される擬似乱数生成器が、少なくとも高い乱数性を持つことを検証するためのツールが必要と考えられている。
また、乱数の検定法には様々な観点からの検定法が存在しており，それらを複数集めて検定ツールとしてまとめられたものもいくつか存在する。代表的なものとしては、NIST FIPS PUB 140-2、NIST Special Publication (SP) 800-22、DIEHARD、”The Art of Computer programming 準数値算法 ” D.Knuth 著に記載されたものなどが知られている。しかし、これらの検定ツールを比較検討すると、

1) 検定ツールごと採用されている検定法が異なり、検定法の選択基準が明確になっていない

2) 同じ検定手法でも検定ツール毎に閾値等の設定値が異なる場合がある

など問題点が存在する。特に、NIST FIPS PUB 140-2 と NIST SP 800-22 には、いくつかの検定法に不具合があることを指摘した学術論文があり、さらに、2002 年度版暗号技術評価報告書においても同様の指摘がなされている。
他方、暗号モジュール委員会で検討中の暗号モジュール評価においても乱数検定が必要になるという背景があり、2003 年度に擬似乱数生成系調査ワーキンググループ設置し、CRYPTREC としての乱数検定ミニマムセットの策定を目標に、擬似乱数生成系の調査および検定法の調査を開始した。
2003 年度の調査の結果、CRYPTREC の乱数検定ミニマムセットにおいては理論的な裏付けが無いものについては、積極的には採用しない方向に決まった。また、理論的裏付けがあったとしても、計算機実験が行なえる程度の実際的な追試を行い、閾値等の設定値が適切かどうかの確認が必要であるとの判断となった。ただし乱数検定法は全部で 250 種類ほど知られており、全部を確認することは非現実的であるので、2004 年度以降は調査の範囲を絞ることも課題となった。その結果、2004 年度以降の活動方針を以下のように定めた。

1) FIPS 140-2、SP 800-22、DIEHARD で採用されている各検定法の調査範囲の絞り込みと理論的根拠の確認及び計算機実験による検証

2) 1) の結果を踏まえての CRYPTREC 擬似乱数検定ミニマムセットへの導入の判断を行う

3) CRYPTREC 擬似乱数検定ミニマムセットの暗号モジュール評価ツールへの組み込みの検討を行う

2004 年度としては、2003 年度に定めた活動方針に添って以下の活動を行なった。

1) FIPS 140-2、SP 800-22、DIEHARD で採用されている各検定法の調査範囲の絞り込みと理論的根拠の確認及び計算機実験による検証

2) 1) の結果を踏まえての CRYPTREC 擬似乱数検定ミニマムセットへの導入の判断を行う

特に，1) の検定方法に関しては、離散フーリエ変換検定するための数学理論を構築するための予備調査と 2003 年度未解決であった分散値の理論的確認と、擬似乱数生成系に対する各種検定方式の理論的根拠の確認及び計算機実験による検証を実施した。

2005 年度には、2003 年度および 2004 年度の調査結果に基づき、CRYPTREC としての乱数検定のためのミニマムセットとして、NIST で公表している SP 800-22 の 16 種類の検定法の中から 14 種類を採択し、「乱数検定ミニマムセット仕様」として本仕様書の作成を行なった。

2　乱数検定のためのミニマムセット案

乱数検定のためのミニマムセットとして下記のものを定める

1. 頻度検定 (Frequency Test)
2. ブロック単位の頻度検定 (Frequency Test within a Block)
3. 連検定 (Runs Test)
4. ブロック単位の最長連検定 (Test for the Longest Run of Ones in Block)
5. 2 値行列ランク検定 (Binary Matrix Rank Test)
6. 重なりのないテンプレート適合検定 (Non-overlapping Template Matching Test)
7. 重なりのあるテンプレート適合検定 (Overlapping Template Matching Test)
8. Maurer のユニバーサル統計検定 (Maurer's "Universal Statistical " Test)
9. 線形複雑度検定 (Linear Complexity Test)
10. 系列頻度検定 (Serial Test)
11. 累積和検定 (Cumulative Sums (Cusum) Test)
12. ランダム回遊検定 (Random Excursions Test)
13. 変形ランダム回遊検定 (Random Excursions Variant Test)
14. 近似エントロピー検定 (Approximate Entropy Test)

3　検定に対する可否判断基準

ミニマムセットで採用されているすべての検定は、0 と 1 からなる乱数列を対象としている。また、ミニマムセットの検定では、各検定ごとに p-value が得られる。p-value とは、真の乱数生成器が検定を行っている系列よりも乱数らしからぬ系列を生成する確率である。例として、頻度検定の場合を考える。このとき、p-value は以下のように求める。試行としては 100 万系列を 1000 本考える。

1. $X_1, X_2, \cdots, X_n$ を $\{1, -1\}$ の中の値をとる n 個の確率変数とし、$S_n = X_1 + X_2 + \cdots + X_n$ とする。

2. 系列が真の乱数生成器からの出力ならば、

$$\begin{aligned} \mu &= 0 \\ \sigma^2 &= n \end{aligned}$$

   となるので、中心極限定理より、

$$\lim_{n\to\infty} P\left(\frac{S_n}{\sqrt{n}} \leq z\right) = \Phi(z)$$

   となる。なお、

$$\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx$$

   は標準正規分布の累積分布関数である。

3. 統計量 $s = |S_n|/\sqrt{n}$ を考える。このとき、

$$P(s \leq z) = 2\Phi(z) - 1$$

   が得られ、

$$p\text{-value} = 2[1 - \Phi(s)]$$

   である。なお、ミニマムセットでは、$\Phi(z)$ の代わりに誤差関数

$$erfc(z) = \int_{z}^{\infty} \frac{2}{\sqrt{\pi}} e^{-x^2} dx$$

   を用いれば

$$p\text{-value} = erfc(s/\sqrt{2})$$

   となる。

ミニマムセットでは、p-value< 0.01 のときに良い乱数生成器ではないと判断する。

なお、統計量がカイ 2 乗統計量の場合、p-value は、

$$\int_{z}^{\infty} \frac{1}{2\Gamma(\frac{N}{2})} \left(\frac{t}{2}\right)^{\frac{N}{2}-1} e^{-\frac{t}{2}} dt$$

により求める。ただし、N は χ^2 分布の自由度であり、

$$\Gamma(\alpha) = \int_{0}^{\infty} x^{\alpha-1} e^{-x} dx$$

である。

ミニマムセットでは、1 本の標本系列に対する仮説検定で乱数生成アルゴリズムを評価するのは無意味であるという考え方から、複数の標本系列 (NIST では 1000 程度を推奨している) に対し検定を行い、

1. p-value の一様性
2. p-value が 0.01 より大きくなる割合

から乱数列の評価を行う。1. では、得られた p-value が区間 $[0, 1)$ で一様に分布しているかどうかを調べるために、$[0, 1)$ を 10 の区間に分割し、分割した区間ごとの頻度が一様になっているかどうかをカイ 2 乗検定にて検定する。カイ 2 乗検定により得られた p-value が 0.0001 以上ならば、乱数列は良い乱数生成器であると判断する。また、2. では、標本の数を m としたとき、0.01 以上となる p-value の数の割合が

$$0.99 \pm 3\sqrt{\frac{0.99 \times 0.01}{m}}$$

の範囲に入っている場合は、乱数列は良い乱数生成器であると判断する。

4　検査の概要と可否判断

4.1　頻度検定

乱数列のビットの出現頻度を求めその度数分布が一様になっているかどうかを標準正規分布の誤差関数にて検定する。

入力	$\{\varepsilon_i\}$	:	0 か 1 の値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本 100 万系列を 1000 本
出力	$erfc(s_{obs})$	:	標準正規分布の誤差関数からの p-value
処理内容	ステップ 1	:	系列を $\pm$ に変換し、$S_n = X_1 + X_2 + \cdots + X_n$ $(X_i = 2\varepsilon_i - 1)$ を計算する。
	ステップ 2	:	統計値 $s_{obs} = \lvert\frac{S_n}{\sqrt{n}}\rvert$ を計算する。
	ステップ 3	:	p-value $= erfc(s_{obs}/\sqrt{2})$　を計算する。

4.2 ブロック単位の頻度検定

乱数列の文字の出現頻度を求めその度数分布が一様になっているかどうかをカイ 2 乗検定にて検定する。

入力	M	:	各ブロックのビット長
	$\{\varepsilon_i\}$	:	0 か 1 の値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
出力	$\chi_0^2(obs)$	:	カイ 2 乗統計量からの p-value
処理内容	ステップ 1	:	入力系列を $N = \lfloor \frac{n}{M} \rfloor$ の重なりの無いブロックに分割する。使わないビットは無視する。
	ステップ 2	:	$\pi_i = \frac{\sum_{j=1}^{M} \varepsilon_{(i-1)M+j}}{M}$ を $1 \leq i \leq N$ に対して計算する。
	ステップ 3	:	カイ 2 乗統計量 χ^2 を $\chi^2(obs) = 4M \sum_{i=1} N(\pi_i - 1/2)^2$ により計算する。
	ステップ 4	:	p-value $= \mathrm{igamc}(N/2, \chi^2(obs)/2)$ を計算する。ここで igamc は不完全なガンマ関数である。

4.3 連検定

区間 $[0,1)$ 上を分布する乱数列を上昇連、下降連で分割し、部分列の長さでクラスを定義する。部分列を長さに応じてクラスに割り当て、その度数をカイ 2 乗検定にて検定し、連の偏りを調べる。なお、乱数列が 0 と 1 からなる乱数の場合は、区間 $[0,1)$ 上に分布する乱数列への変換が必要である。

入力	n	:	ビット列の長さ
	$\{\varepsilon_i\}$	:	0 と 1 の c 中から値をとる乱数列
出力	p-value	:	カイ 2 乗統計量からの p-value（
処理内容	ステップ 1	:	事前に 1 が出力される確率 $\pi = \frac{\sum_i \varepsilon_i}{n}$ を計算する
	ステップ 2	:	頻度テストをパスするかどうか決定する。もしも $\tau \leq \|\pi - 1/2\|$ が示されるならば連検定を行う必要は無い。検定を行う必要が無い場合は p-value は 0.0000 に設定される。ここで $\tau = \frac{2}{\sqrt{n}}$ は事前に定義されていなければならない。
	ステップ 3	:	検定値 $V_n(obs) = \sum_{k=1}^{n-1} r(k) + 1$ を計算する。ここで、$\varepsilon_k = \varepsilon_{k+1}$ ならば $r(k) = 0$ で、さもなければ $r(k) = 1$ である。
	ステップ 4	:	p-value $= erfc(\frac{\|V_n(obs) - 2n\pi(1-[pi)\|}{2\sqrt{2n\pi(1-\pi)}})$ を計算する

4.4 ブロック単位の最長連検定

0 と 1 からなる乱数列を M ビット単位で分割し、最長連の長さに応じて各部分列を 7 個のクラスに割り当てその度数をカイ 2 乗検定にて検定し最長連の長さの偏りを調べる。なお、M は乱数列の長さによって決まる。

入力　$\{\varepsilon_i\}$　：　0 と 1 の中から値をとる乱数列
　　n　：　乱数列の長さ（乱数の個数）100 万系列を 1000 本
　　M　：　各ブロックのビット数

出力　p-value　：　カイ 2 乗統計量からの p-value

処理内容　ステップ 1　：　乱数列を長さ M のブロックに分割する。ただし、ブロックサイズは、$128 \leq n < 6272$ のときは $M = 8$、$6272 \leq n < 750,000$ のときは $M = 128$、$750,000 \leq n$ のときは $M = 10000$ とする。

ステップ 2　：　$M = 8$ のときは、各ブロックを 1 の最長連の長さに応じて、f_0 から f_3 の $K = 4$ 個のクラスに分割し、各クラスの度数を求める。クラスに対応する長さおよびクラスの確率は

クラス	f_0	f_1	f_2	f_3
度数	〜1	2	3	4〜
確率	0.2148	0.3672	0.2305	0.1875

となる。また、$M = 128$ のときは、以下のような $K = 6$ 個のクラスに割り当てる。

クラス	f_0	f_1	f_2	f_3	f_4	f_5
度数	〜4	5	6	7	8	9〜
確率	0.1174	0.2430	0.2493	0.1752	0.1027	0.1124

となる。また、$M = 10000$ のときは、以下のような $K = 7$ 個のクラスに割り当てる。

クラス	f_0	f_1	f_2	f_3	f_4	f_5	f_6
度数	〜10	11	12	13	14	15	16〜
確率	0.0882	0.2092	0.2483	0.1933	0.1208	0.0675	0.0727

ステップ 3　：　カイ 2 乗統計量 $\chi^2(obs) = \sum_{i=0}^{K} \frac{(f_i - N\pi_i)^2}{N\pi_i}$ を求める。

ステップ 4　：　$\chi^2(obs)$ は、自由度 $K-1$ の χ^2 分布に従うので、この分布から p-value を計算する。

最長連検定は、長さ 20000 ビットの 1 ブロックに対する検定である。

4.5　2 値行列ランク検定：(32×32) の 2 値行列ランク検定

0 と 1 からなる乱数列の 1024 ビットの部分列から (32×32) の 2 値行列を構成し、各部分列を行列のランクに応じてを 3 個のクラスに割り当て、その度数をカイ 2 乗検定にて検定しランクの偏りを調べる。

入力　$\{X_i\}$　：　0 と 1 の中から値をとる乱数列

出力　p-value　：　カイ 2 乗統計量からの p-value

処理内容

ステップ 1　：　行列のランクを、29 以下、30, 31, 32 の 4 個のクラスに分ける。

ステップ 2　：　X_i を列の先頭から順に 32 ビットずつ取り出し、32 ビットの整数からなる新しい列 $\{Y_i\}$ を作る。

ステップ 3　：　連続する 32 個の Y_i を順に並べて $GF(2)$ 上の 32×32 行列を作る。32 個の組は $(Y_1, Y_2, \cdots, Y_{32}), (Y_{33}, Y_{33}, \cdots, Y_{64})$ のように重ね合わせずに作っていく。

ステップ 4　：　ステップ 3 で得られた行列のランクを計算し、ランクに応じてクラスの度数に加えていく。各クラスの度数を f_0, f_1, f_2, f_3 とする。

ステップ 5　：　ステップ 2〜4 を 40,000 回くり返す。

ステップ 6　：　カイ 2 乗統計量 χ_0^2 を計算する。各クラスの確率の値は、次の表の値を用いる。χ_0^2 は自由度 3 の χ^2 分布に従うので、この分布から p-value を計算する。

〜28	29	30	31
0.0052854502	0.1283502644	0.5775761902	0.2887880952

4.6 重なりの無いテンプレート適合検定

0 と 1 からなる乱数列を 8 つのブロックに分割し、各ブロックごとに m ビットの窓を先頭からスライドさせ、窓と m 文字のテンプレートが適合する回数を調べる。8 ブロックそれぞれの適合回数をカイ 2 乗検定にて検定し適合回数の偏りを調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
	m	:	テンプレートの長さ
	B	:	m ビットのテンプレート
出力	p-value	:	カイ 2 乗統計量からの p-value
処理内容	ステップ 1	:	乱数列を長さ $M = 131,072 = 2^{17}$ の $N = 8$ 個のブロックに分割する。
	ステップ 2	:	ブロック j のテンプレートの適合回数を W_j とする。m ビットの窓をブロックの先頭にセットし、テンプレートが適合しないときは、窓を 1 ビットずらし、適合するときは窓を m ビットずらす。
	ステップ 3	:	中心極限定理より、各ブロックの適合回数はは平均 $\mu = (M-m+1)/2^m$、分散 $\sigma^2 = M\left(\frac{1}{2^m} - \frac{2m-1}{2^{2m}}\right)$ の正規分布に従うことを用いて、カイ 2 乗統計量 $\chi^2(obs) = \sum_{j=1}^{N} \frac{(W_j - \mu)^2}{\sigma^2}$ を求める。
	ステップ 4	:	$\chi^2(obs)$ は、自由度 N の χ^2 分布に従うので、この分布から p-value を計算する。

ミニマムセットでは、テンプレートの長さを 2 から 10 まで選択できるが、9 または 10 が推奨されている。なお、NIST のツールでは、乱数列の長さが 1,000,000 の場合のみ検定することができる。

4.7　重なりのあるテンプレート適合検定

0 と 1 からなる乱数列を 968 個のブロックに分割し、各ブロックを m 文字のテンプレートが適合する回数により 6 個のクラス割り当て、その度数をカイ 2 乗検定にて検定し適合回数の偏りを調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
	m	:	テンプレートの長さ
	B	:	m ビットのテンプレート
出力	p-value	:	カイ 2 乗統計量からの p-value
処理内容	ステップ 1	:	乱数列を長さ $M = 1032$ の $N = 968$ 個のブロックに分割する。
	ステップ 2	:	ブロック j のテンプレートの適合回数を W_j とする。m ビットの窓をブロックの先頭から 1 ビットずつずらして行き、テンプレートと窓の適合回数を数える。
	ステップ 3	:	適合回数 0 のクラスを f_0、1 のクラスを f_1、2 のクラスを f_2、3 のクラスを f_3、4 のクラスを f_4、5 以上のクラスを f_5 とし、ブロックの適合回数 W_j から各クラスの度数を求める。
	ステップ 4	:	カイ 2 乗統計量 $\chi^2(obs) = \sum_{i=0}^{5} \frac{(f_i - N\pi_i)}{N\pi_i}$ を求める。クラス f_i に対応する確率 π_i は、 $\pi_0 = e^{-\eta}$ $\pi_1 = \frac{\eta}{2} e^{-\eta}$ $\pi_2 = \frac{\eta e^{-\eta}}{8}[\eta + 2]$ $\pi_3 = \frac{\eta e^{-\eta}}{8}\left[\frac{\eta^2}{6} + \eta + 1\right]$ $\pi_4 = \frac{\eta e^{-\eta}}{16}\left[\frac{\eta^3}{24} + \frac{\eta^2}{2} + \frac{3\eta}{2} + 1\right]$ $\pi_5 = 1 - (\pi_0 + \pi_1 + \pi_2 + \pi_3 + \pi_4)$ となる。ただし、$\eta = \lambda/2,\ \lambda = (M - m + 1)/2^m$ である。
	ステップ 5	:	$\chi^2(obs)$ は、自由度 5 の χ^2 分布に従うので、この分布から p-value を計算する。

4.8 Maurer のユニバーサル統計検定

0 と 1 からなる乱数列における長さ L ビットのパターンの間隔を調べることにより乱数列の一様性・圧縮可能性を調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
	L	:	ブロックの長さ
	Q	:	初期系列のブロック数
出力	p-value	:	正規分布統計量からの p-value
処理内容	ステップ 1	:	乱数列を長さ L のブロックに分割し、先頭から Q ブロック分を初期セグメントとし、残りの K ブロック分をテストセグメントとする。ただし、$K = \lfloor (n - QL)/L \rfloor$ とする。
	ステップ 2	:	T_j $(0 \leq j \leq 2^L)$ の初期値を $T_j = 0$ とし、「i 番目の L ビットブロックを 2 進数とみなしたときの値が j のときに、$T_j = i$ とする」という処理を初期セグメントの先頭ブロックから初期セグメントの最終ブロックまで順に行う $(1 \leq i \leq Q)$。
	ステップ 3	:	sum の初期値を 0 とし、「i 番目の L ビットブロックを 2 進数とみなしたときの値が j のときに、$sum = sum + \log_2(i - T_j)$ とし、さらに $T_j = i$ とする」という処理をテストセグメントの先頭ブロックからテストセグメントの最終ブロックまで順に行う $(Q+1 \leq i \leq Q+K)$。また、$f_n = sum/K$ を求める。
	ステップ 4	:	f_n は下記の表にある平均、分散の正規分布に従うので、この分布から p-value を計算する 。

L	平均	分散
6	5.2177052	2.954
7	6.1962507	3.125
8	7.1836656	3.238
9	8.1764248	3.311
10	9.1723243	3.356
11	10.170032	3.384
12	11.168765	3.401
13	12.168070	3.410
14	13.167693	3.416
15	14.167488	3.419
16	15.167379	3.421

4.9 線形複雑度検定

0 と 1 からなる乱数列を長さ M のブロックに分割し、ブロックごとの線形複雑度を求めることにより乱数列の周期性を調べる。

入力　$\{X_i\}$　：　0 と 1 の中から値をとる乱数列
　　n　：　乱数列の長さ（乱数の個数）100 万系列を 1000 本
　　M　：　ブロックの長さ

出力　p-value　：　カイ 2 乗統計量からの p-value

処理内容　ステップ 1　：　乱数列を長さ M のブロックに分割し、$N = \lfloor n/m \rfloor$ とする。
　　ステップ 2　：　Berlekamp-Massey アルゴリズムを用いて、ブロック i の線形複雑度 L_i を求める $(i = 1, \cdots, N)$。
　　ステップ 3　：　$\mu = \dfrac{M}{2} + \dfrac{(9 + (-1)^{M+1})}{36} + \dfrac{(M/3 + 2/9)}{2^M}$ を求め、$T_i = (-1)^M \times (L_i - \mu) + 2/9$ を求める $(i = 1, \cdots, N)$。
　　ステップ 4　：　T_i の値により 7 つのクラス $\nu_0, \nu_1, \cdots, \nu_6$ を定め、各クラスの度数を求める。

クラス	T_i の範囲
ν_0	$T_i \leq -2.5$
ν_1	$-2.5 < T_i \leq -1.5$
ν_2	$-1.5 < T_i \leq -0.5$
ν_3	$-0.5 < T_i \leq 0.5$
ν_4	$0.5 < T_i \leq 1.5$
ν_5	$1.5 < T_i \leq 2.5$
ν_6	$2.5 < T_i$

　　ステップ 5　：　カイ 2 乗統計量 χ_0^2 を計算する。クラス i の確率の値 π_i は、次の表の値を用いる。χ_0^2 は自由度 6 の χ^2 分布に従うので、この分布から p-value を計算する。

クラス	確率
π_0	0.01047
π_1	0.03125
π_2	0.125
π_3	0.5
π_4	0.25
π_5	0.0625
π_6	0.02078

4.10 系列頻度検定

0 と 1 からなる乱数列における長さ m ビットのパターン長さ $m-1$ ビットのパターン、長さ $m-2$ ビットのパターンが一様に出現しているかを調べることにより乱数列の一様性・圧縮可能性を調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
	m	:	ブロックの長さ
出力	p-value	:	カイ 2 乗統計量からの p-value(2 個)
処理内容	ステップ 1	:	重なりのある m ビットブロックの列を $(X_1, X_2, \cdots, X_m), (X_2, X_3, \cdots, X_{m+1}) \cdots$ のように定める。同様に、重なりのある $m-1$ ビットブロックの列、重なりのある $m-2$ ビットブロックの列を定める。
	ステップ 2	:	m ビットパターン $i_1 i_2 \cdots i_m$ の出現頻度 $\nu_{i_1 i_2 \cdots i_m}$、$m-1$ ビットパターン $i_1 i_2 \cdots i_{m-1}$ の出現頻度 $\nu_{i_1 i_2 \cdots i_{m-1}}$、$m-2$ ビットパターン $i_1 i_2 \cdots i_{m-2}$ の出現頻度 $\nu_{i_1 i_2 \cdots i_{m-2}}$ を求める。
	ステップ 3	:	$\Psi_m^2 = \frac{2^m}{n} \sum_{i_1 i_2 \cdots i_m} \left(\nu_{i_1 i_2 \cdots i_m} - \frac{n}{2^m}\right)^2 = \frac{2^m}{n} \sum_{i_1 i_2 \cdots i_m} \nu_{i_1 i_2 \cdots i_m}^2 - n$、$\Psi_{m-1}^2 = \frac{2^{m-1}}{n} \sum_{i_1 i_2 \cdots i_{m-1}} \left(\nu_{i_1 i_2 \cdots i_{m-1}} - \frac{n}{2^{m-1}}\right)^2 = \frac{2^{m-1}}{n} \sum_{i_1 i_2 \cdots i_{m-1}} \nu_{i_1 i_2 \cdots i_{m-1}}^2 - n$、$\Psi_{m-2}^2 = \frac{2^{m-2}}{n} \sum_{i_1 i_2 \cdots i_{m-2}} \left(\nu_{i_1 i_2 \cdots i_{m-2}} - \frac{n}{2^{m-2}}\right)^2 = \frac{2^{m-2}}{n} \sum_{i_1 i_2 \cdots i_{m-2}} \nu_{i_1 i_2 \cdots i_{m-2}}^2 - n$ を求める。
	ステップ 4	:	$\nabla \Psi_m^2 = \Psi_m^2 - \Psi_{m-1}^2$ および $\nabla^2 \Psi_m^2 = \Psi_m^2 - 2\Psi_{m-1}^2 + \Psi_{m-2}^2$ を求める。
	ステップ 5	:	$\nabla \Psi_m^2$ は自由度 2^{m-1} の χ^2 分布に従うので、この分布から p-value を計算する。同様に、$\nabla^2 \Psi_m^2$ は自由度 2^{m-2} の χ^2 分布に従うので、この分布から p-value を計算する。

4.11 累積和検定

0 と 1 からなる乱数列 $X_1, X_2, \cdots X_n$ に対し、$S_i = \sum_{j=1}^{i}(2X_j - 1)$ および $S'_i = \sum_{j=n-i+1}^{n}(2X_j - 1)$ $(1 \leq i \leq n)$ の絶対値の最大値を求め、その偏りを調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
出力	p-value	:	正規分布統計量からの p-value(2 個)
処理内容	ステップ 1	:	$S_1 = 2X_1 - 1$ とし、$S_k = S_{k-1} + 2X_k - 1$ を求める $(2 \leq k \leq n)$。さらに、$z = \max_{1 \leq k \leq n} \lvert S_k \rvert$ を求める。[Mode 0]
	ステップ 2	:	$p - \text{value} = 1 - \sum_{k=(-n/z+1)/4}^{(n/z-1)/4} \left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}} \right) - \Phi\left(\frac{(4k-1)z}{\sqrt{n}} \right) \right] + \sum_{k=(-n/z-3)/4}^{(n/z-1)/4} \left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}} \right) - \Phi\left(\frac{(4k+1)z}{\sqrt{n}} \right) \right]$ を求める。ただし、Φ は標準正規分布の累積分布関数である。
	ステップ 3	:	$S_1 = 2X_n - 1$、$S_k = S_{k-1} + 2X_{n-k} - 1$ とし、ステップ 1、2 を行う。[Mode 1]

4.12 ランダム回遊検定

0 と 1 からなる乱数列 $X_1, X_2, \cdots X_n$ に対し、$S_i = \sum_{j=1}^{i}(2X_j - 1)$ $(1 \leq i \leq n)$ を求め、$S_i = 0$ から次に 0 になるまでを 1 つのサイクルとみなし、-4〜-1、および、1〜4 の 8 種類の状態ごとにサイクルの出現度数の偏りを調べる。

入力　$\{X_i\}$　：　0 と 1 の中から値をとる乱数列
　　n　：　乱数列の長さ（乱数の個数）100 万系列を 1000 本

出力　p-value　：　カイ 2 乗統計量からの p-value(8 個)

処理内容　ステップ 1　：　$S_1 = 2X_1 - 1$ とし、$S_k = S_{k-1} + 2X_k - 1$ を求める $(2 \leq k \leq n)$。さらに、$0, S_1, S_2, \cdots, S_n, 0$ という新しい数列 S' を定める。

ステップ 2　：　0 から始まり次に現れる 0 までを 1 サイクルとし、 数列 S' からサイクルを求める。なお、数列 S' のサイクル数 J が $J < 500$ ならば、検定を中止する。

ステップ 3　：　8 種類の状態値を-4〜-1、および、1〜4 とし、各サイクルごとに状態値の出現度数を求める。

ステップ 4　：　状態値 x の出現度数が k となるサイクルの数を $\nu_k(x)$ とし、8 種類のすべての状態に対し $\nu_k(x)$ を求める $(1 \leq k \leq 5)$。ただし、$\nu_5(x)$ は状態値 x の出現度数が 5 回以上のサイクル数とする。

ステップ 5　：　8 種類のすべての状態値に対し、カイ 2 乗統計量 $\chi^2(obs) = \sum_{k=0}^{5} \dfrac{(\nu_k(x) - J\pi_k(x))^2}{J\pi_k(x)}$ を求める。ただし、

	$\pi_0(x)$	$\pi_1(x)$	$\pi_2(x)$	$\pi_3(x)$	$\pi_4(x)$	$\pi_5(x)$
$\|x\| = 1$	0.5000	0.2500	0.1250	0.0625	0.0312	0.0312
$\|x\| = 2$	0.7500	0.0625	0.0469	0.0352	0.0264	0.0791
$\|x\| = 3$	0.8333	0.0278	0.0231	0.0193	0.0161	0.0804
$\|x\| = 4$	0.0875	0.0156	0.0137	0.0120	0.0105	0.-733

である。

ステップ 6　：　$\chi^2(obs)$ は自由度 5 の χ^2 分布に従うので、この分布から 8 種類のすべての状態値に対する p-value を計算する。

4.13 変形ランダム回遊検定

0 と 1 からなる乱数列 $X_1, X_2, \cdots X_n$ に対し、$S_i = \sum_{j=1}^{i}(2X_j - 1)$ $(1 \leq i \leq n)$ を求め、-9〜-1、および、1〜9 の 18 種類の状態の出現度数の偏りを調べる。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
出力	p-value	:	正規分布統計量からの p-value(18 個)
処理内容	ステップ 1	:	$S_1 = 2X_1 - 1$ とし、$S_k = S_{k-1} + 2X_k - 1$ を求める $(2 \leq k \leq n)$。さらに、$0, S_1, S_2, \cdots, S_n, 0$ という新しい数列 S' を定める。
	ステップ 2	:	0 から始まり次に現れる 0 までを 1 サイクルとし、 数列 S' からサイクルを求める。なお、数列 S' のサイクル数を J とする。
	ステップ 3	:	18 種類の状態値を-9〜-1、および、1〜9 とし、状態値 x の出現度数 $\xi(x)$ を求める。
	ステップ 4	:	18 種類のすべての状態値に対する p-value を $p - \text{value} = erfc\left(\dfrac{\lvert\xi(x) - J\rvert}{\sqrt{2J(4\lvert x\rvert - 2)}}\right)$ により求める。ただし、$erfc$ は標準正規分布の誤差関数である。

4.14 近似エントロピー検定

0 と 1 からなる乱数列における長さ m ビットのパターン長さ $m+1$ ビットのパターンが一様に出現しているかを調べることにより乱数列の一様性・圧縮可能性を調べる。ただし、$m < \log(n) - 7$ とする。

入力	$\{X_i\}$	:	0 と 1 の中から値をとる乱数列
	n	:	乱数列の長さ（乱数の個数）100 万系列を 1000 本
	m	:	ブロックの長さ
出力	p-value	:	カイ 2 乗統計量からの p-value
処理内容	ステップ 1	:	重なりのある m ビットブロックの列を $(X_1, X_2, \cdots, X_m), (X_2, X_3, \cdots, X_{m+1}) \cdots$ のように定める。同様に、重なりのある $m+1$ ビットブロックの列を定める。
	ステップ 2	:	すべての m ビットパターンに対して、出現頻度 $\#i$ および $C_i^m = \#i/n$ を求める。ただし、i は m ビットブロックを 2 進数とみなしたときの値とする。さらに、$\phi^{(m)} = \sum_{i=0}^{2^m-1} \pi_i \log \pi_i$ を求める。ただし、$\pi_i = C_j^m,\ j = \log_2 i$ とする。
	ステップ 3	:	すべての $m+1$ ビットパターンに対して、出現頻度 $\#i$ および $C_i^{m+1} = \#i/n$ を求める。ただし、i は $m+1$ ビットブロックの値を 2 進数とみなしたときの値とする。さらに、$\phi^{(m+1)} = \sum_{i=0}^{2^{m+1}-1} \pi_i \log \pi_i$ を求める。ただし、$\pi_i = C_j^{m+1},\ j = \log_2 i$ とする。
	ステップ 4	:	カイ 2 乗統計量 $\chi_0^2 = 2n[\log 2 - (\phi^{(m)} - \phi^{(m+1)})]$ を求める。
	ステップ 5	:	χ_0^2 は自由度 2^m の χ^2 分布に従うので、この分布から p-value を計算する。

5　可否判定条件

以下に p-value の判定条件を定める。条件を満たさない場合はランダムでないと判定する。

表 1: 可否判定条件

検定法	判定条件
頻度検定	p-value ≥ 0.01 ならばランダムと判定
ブロック単位の頻度検定	p-value ≥ 0.01 ならばランダムと判定
連検定	p-value ≥ 0.01 ならばランダムと判定
ブロック単位の最長連検定	p-value ≥ 0.01 ならばランダムと判定
2 値行列ランク検定	p-value ≥ 0.01 ならばランダムと判定
重なりのないテンプレート適合検定	p-value ≥ 0.01 ならばランダムと判定
重なりのあるテンプレート適合検定	p-value ≥ 0.01 ならばランダムと判定
Maurer のユニバーサル統計検定	p-value ≥ 0.01 ならばランダムと判定
線形複雑度検定	p-value ≥ 0.01 ならばランダムと判定
系列頻度検定	p-value ≥ 0.01 ならばランダムと判定
累積和検定	p-value ≥ 0.01 ならばランダムと判定
ランダム回遊検定	p-value ≥ 0.01 ならばランダムと判定
変形ランダム回遊検定	p-value ≥ 0.01 ならばランダムと判定
近似エントロピー検定	パラメータの範囲を狭め、 $\log(n) - 7$ 以下ならばランダムと判定

正誤表

報告書等	項目等	頁	改訂内容
CRYPTREC Report 2005 暗号技術監視委員会報告書(c05_wat_final.pdf, 作成日 06/4/19)	3.2.2 技術仕様 MD5 のハッシュ値計算 3., 4., 5., 6.における式 B_{t+1}	93	ラウンド1,2,3,4を定義する式のうち、B_{t+1}において下記の修正を4つの式すべてに適用する。 (誤) B_{t+1} = ROTL… (正) B_{t+1} = B_t + ROTL…
同上	3.4.2 技術仕様 RIPEMD の前処理 1.の文中の脚注 5	99	(旧) メッセージ長が2^{64}より… (新) メッセージ長を表現する場合、下位32ビット表現のあとに上位32ビット表現をする。なお、メッセージ長が2^{64}より…
同上	表 5 の 5 行 1 列目の欄	267	(旧) OFB (新) OFB (k-OFB)
同上	表 5 の 5 行 2 列目の欄	267	(旧) $2^{n/2}$ブロック程度で… (新) $k<n$ の場合 $2^{n/2}$ ブロック程度で…

発行日　2006 **年** 4 **月** 19 **日**　**第** 1 **版**
2006 **年** 5 **月** 17 **日**　**第** 2 **版**

発行者

- **〒** 184-8795
  東京都小金井市貫井北町四丁目 2 **番** 1 **号**
  独立行政法人 情報通信研究機構
  (**情報通信セキュリティ研究センター セキュリティ基盤グループ**)
  NATIONAL INSTITUTE OF
  INFORMATION AND COMMUNICATIONS TECHNOLOGY
  4-2-1 NUKUI-KITAMACHI, KOGANEI
  TOKYO, 184-8795 JAPAN

- **〒** 113-6591
  東京都文京区本駒込二丁目 28 **番** 8 **号**
  文京グリーンコートセンターオフィス 16 **階**
  独立行政法人 情報処理推進機構
  (**セキュリティセンター 暗号グループ**)
  INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
  BUNKYO GREEN COURT CENTER OFFICE
  2-28-8 HONKOMAGOME, BUNKYO-KU
  TOKYO, 113-6591 JAPAN