(5)暗号・セキュリティ技術

巨大かつ多様な通信ネットワークにおいて，信頼性をサービスとして利用者に提供するための，新しい暗号アルゴリズムとプロトコルについて概説する．

クレイグ ジェントリー　ズルフィカー ラムザン
Craig Gentry　Zulfikar Ramzan

1. まえがき

次世代（4G）サービスを実現するには，そのセキュリティソリューションが非常にスケーラブル，かつ柔軟なものであることが要求される．なぜならば，利用者数が膨大になるだけでなく，利用するネットワーク環境もさまざまな形態が混合し，絶えず変化する状態になっていくためである．利用方法も多様化し，個別情報の安全なアクセス，他のサービスプロバイダとの安全なやり取りが要求される．4Gネットワークの開発においては，発展と柔軟性を抑制するのではなく促進するセキュリティソリューションを見出すことが重要である．

本稿では，USA研究所が現在取り組んでいるいくつかの暗号アルゴリズム研究の一端を紹介する．これらの暗号アルゴリズムは，さまざまな環境に存在する膨大な利用者に対してセキュリティサービスを提供できるように設計されたものである．まず，公開鍵暗号基盤（PKI：Public Key Infrastructure）管理を簡素化するために設計した3つの技術，すなわち階層的IDベース暗号方式，証明書不要の公開鍵暗号，および集合署名について説明する．次に，安全にコンテンツ配布を可能にするよう設計した2つの技術，すなわち新型のハッシュツリー（Hash Tree）を使用した効率的なマイクロクレジット方式，およびエンド・ツー・エンドの安全性を損ねることなく中間プロキシがストリームを動的に変換できるようにするストリーム認証方式について説明する．

2. PKIの簡素化

公開鍵暗号（PKC：Public Key Cryptography）では，信頼関係のない者同士が事前の情報交換なしで安全に通信でき，対称暗号方式よりも柔軟性があるが，通信を開始する前に，公開鍵が配布され認証されていることが必要である．例えば，受信者の公開鍵で暗号化したメッセージを送信する前に，送信者は受信者の公開鍵を取得し，信頼された機関からその公開鍵が本物であること，すなわち，ほかの誰かではなく受信者本人のものである，という保証を受けなければならない．

通常，公開鍵の配布と認証には，PKIが必要となる．PKIには認証局（CA：Certificate Authority）と呼ばれる第三者信頼機関が含まれ，これが公開鍵証明書を発行する．証明書は認証局のディジタル署名により複数の情報が安全に束ねられたもので，少なくともユーザ名Uとその公開鍵PK_Uを，多くの場合さらにシリアル番号SN_U（証明書の管理を簡素化する），証明書の発行日付I_U，有効期限E_Uを含む．認証局は署名Sig_{CA}（U, PK_U, SN_U, I_U, E_U）を発行することで，PK_UがユーザUの正式な公開鍵であり，現在の日付I_Uから将来の日付E_Uまで有効であることを証明する．

認証局は未来を予測できないため，状況によって本来の有効期限より前に証明書を失効させる必要も生じる．例えば，ユーザ固有の秘密鍵が誤って露呈したり，危殆化（Compromise）された場合，あるいはユーザが会社を辞めるなどの理由でその鍵の使用権がなくなる場合などが考えられる．このように証明書が失効可能な場合は，認証局がその証明書が現在有効かどうかを示す証明書ステータス情報を配布しないかぎり，その証明書を信頼すべきではない．証明書ステータス情報は，常に最新（例えば1日以内）のものでなければならない．またユーザの公開鍵の現行有効性を信頼しているすべての関係者に広く配布される必要がある．大量の最新の証明書情報を配布するタスクは，「証明書失効問題」と呼ばれている．

証明書失効問題に対する最も有名なアプローチは，証明書失効リスト（CRL：Certificate Revocation List）である．CRLは，本来の有効期限以前に失効した証明書のリストである．CAはその署名を添えて，このリストを定期的に発行する．CAはその証明書の多くを失効させることになりがちであり（文献[1]によると有効期限1年で発行された場合は10％に及ぶ），CAが多数のユーザを抱えている場合，CRLは相当な量になる．CAの最終更新以降に取り消された証明書だけをリストアップするΔ-CRLのように，このアプローチを改良したものもあるが，それでも送信コストと関連インフラコストはかなり高いものとなる．

その他の提案としては，オンライン証明書状態プロトコル（OCSP：Online Certificate Status Protocol）と呼ばれているものがある．CAは証明書の状態照会に対し，オンラインで現在の証明書情報に署名したもので応答する．これによって照会ごとの送信コストは削減されるが，演算コストは大きく増加する．また，CAが集中管理されている場合は，DoS（Denial of Service）攻撃を受けやすくなりセキュリティが低下する．CAが分散され，各サーバが独自の秘密鍵を

使用している場合は，1つのサーバの危殆化がシステム全体の危殆化につながる[1]．

1984年に，Shamir[2]はPKIとは異なるアプローチについて述べている．これは，IDベース暗号方式（IBC：Identity-Based Cryptography）と呼ばれ，証明書を完全に排除するように設計されたものである．IBCでは，ユーザの公開鍵がそのアイデンティティID_U（IPアドレスや電子メールアドレス）から直接導き出される．ユーザの秘密鍵PrK_Uは鍵生成局（PKG：Private Key Generator）と呼ばれる第三者信頼機関によって生成される．PKGは公開鍵PK_{PKG}と「マスター」秘密鍵PrK_{PKG}を持ち，PrK_{PKG}とID_UからPrK_Uを計算する．IBCでは，送信者がPK_{PKG}とID_Uを知っていればユーザUに対するメッセージを暗号化できるため，ユーザの公開鍵を配布する必要がない．さらに，UはPKGから秘密鍵を受信している場合のみ復号化できるので，証明書も不要となる．ShamirはIDベース署名方式について提案しており，このIDベース暗号方式はごく最近になって開発された[3],[4]．USA研究所は，以下に述べる階層的IDベース暗号方式（HIBC：Hierarchical Identity-Based Cryptography）と証明書不要PKCによって，証明書の必要性を排除しただけでなく，基本的なIBCをさまざまな方法で改良した．

2.1 階層的IDベース暗号方式

IBCは公開鍵を配布する必要性を排除するが，ユーザU_1がユーザU_2と異なるPKGを使用している場合には，U_1はU_2に対するメッセージを暗号化する前にU_2のPKGから公開鍵を取得する必要があり，証明書要求が不要という利点が失われる．これに対して，単一のPKGで非常に大勢のユーザに対する公開鍵の生成を処理するのは後述するようにスケーラビリティの問題が大きい．HIBCはこのようなジレンマを解決するために，ルートPKGが秘密鍵の生成を下位PKGに委任することができ，かつ外部ユーザはルートPKGから公開鍵PK_{root}を取得するだけでよいという，階層的トポロジーを可能にする．HIBCは，Boneh-Franklin IBC方式と同様に，楕円曲線と「ペアリング」という数学的構成概念を利用している[5]．以下に，例をあげて具体的にHIBCを概説する．

HIBCでは，各ユーザが階層における位置を示すIDタプルを持っている．例えば，アリスの電子メールアドレスが“alice@cs.univ.edu”であれば，そのIDタプルは（edu, univ, cs, alice）のようなものになる．アリスの親階層はIDタプル（edu, univ, cs）を持ったクライアントであるため，彼女の大学CS学部のシステム管理者がこのクライアントを管理できる．

ルートPKGはマスター秘密鍵を使用して，階層で1段階下になるPKGの秘密鍵を生成する．例えば，上記のシステム管理者が大学からCS学部の秘密鍵を取得すると，アリスはCS学部から秘密鍵を取得することができる．これを達成するために，アリスはCS学部にIDタプルと身元証明を提出し，CS学部ではアリスの身元を確認した後に，独自の秘密鍵および彼女のIDタプル（および失効管理のために現在の日付などの広く入手できる追加情報）に応じた彼女の秘密鍵を計算する．CS学部は安全なチャネルを経由して，アリスに彼女の秘密鍵を送信する．アリスのCS学部は「ローカル」であるため，彼女のIDを提出したり安全なチャネルを確保するのは，非階層的なIBCに比べ容易である．CS学部より上位の階層は，アリスの秘密鍵の生成には直接関与しない．

階層的IDベース暗号方式（階層的IDベース「署名」の形態も実現可能だが，以下は「暗号」を例にとって説明する）を使用すると，彼女以外の人（ここではボブとする）がPK_{root}とアリスのIDタプル（および失効管理のための追加情報）を知っているかぎり，彼女に対するメッセージを暗号化できる．ボブは，アリスやルート以外のPKGの公開鍵を取得する必要はない（実際，アリスやルート以外のPKGの公開鍵は存在しない）．また，ボブはアリスが最新の秘密鍵を持っていないかぎりメッセージを暗号化できないことを知っているので，アリスの最新証明書を取得する必要もない．ボブはPK_{root}とアリスのIDタプルをメッセージMと結合して，暗号文Cを生成する．アリスは彼女の秘密鍵を使って，Cを復号化する．暗号化と復号化の所要時間，および暗号文の長さはすべて，受信者（アリス）の階層の深さに比例するが，所要時間，暗号文の長さはともに一般にかなり小さい．

HIBCの興味深い特性は，秘密鍵の生成が階層的なものであるため，受信者の上位階層もすべてメッセージを復号化できることである．この特性は一部の設定では有用だが，そうでない場合が多い．2.2節では，この特性を排除しながら，証明書を必要としないIBCとHIBCのメリットを持つ方式である，証明書不要の公開鍵暗号方式を紹介する．

2.2 証明書不要の公開鍵暗号

証明書不要の公開鍵暗号方式[6]*1（後述の集約署名を使えば，証明書不要の署名としても実現可）は，受信者が自

*1 文献[6]では，「証明書ベースの暗号方式（certificate-based encryption）」という用語を使用しているが，その後，文献[7]で使用された「証明書不要（certificateless）」という用語の方が画期的なアイデアの本質をよくとらえている．

身の公開鍵／秘密鍵のペアを生成し，公開鍵と身元証明を第三者信頼機関（TTP：Trusted Third Party）に提供して「認証」を受ける点で，従来の（IDベースではない）公開鍵暗号方式に類似している．しかし，TTPは従来の証明書を生成するのではなく，IDベース暗号方式同様に受信者のIDベース復号化鍵を生成する．本質的に，送信者はそのメッセージを「二重に暗号化」する．すなわち，従来の公開鍵暗号方式で受信者個人の公開鍵を使用して一度暗号化した後，さらにIDベース暗号方式でTTPの公開鍵と受信者のID情報を使用して暗号化する．したがって，受信者が復号化するには，個人の秘密鍵と最新のIDベース復号化鍵が必要になる．

受信者のTTPが現在の秘密鍵を計算し受信者に提供（この行為が受信者を「認証」することに対応）していないかぎり受信者は復号化できない．この事実を送信者は知っていることから，送信者による証明書の状態確認が不要となる．またTTPは受信者個人の秘密鍵を知らないので，復号化することはできない．さらに，受信者個人の秘密鍵が保護されているかぎり，TTPは受信者のIDベース復号化鍵を平文で送信できる．

証明書不要の公開鍵暗号では，送信者は受信者の公開鍵を取得する必要があるが，公開鍵に証明書が不要であるため，取得は一度でよい．証明書状態照会を排除することで，TTPもそのインフラを削減できる．ネットワークのどこかにいる送信者から送られ，いずれかの受信者と関係がある証明書状態照会にTTPが応答しなければならない従来のPKIと異なり，証明書不要の公開鍵暗号を使用するTTPは，秘密鍵をそのユーザ（受信者）に送信するだけでよい．ビジネスモデルの観点から，TTPが自己のユーザへの対応に専念できるのは非常に魅力的である．また，秘密鍵は（文献[6]に説明されている「インクリメンタルCBE（Certificate Based Encryption）」で）要求を待たず複数ユーザに送付できるため，マルチキャストを利用すれば，CAの送信コストを劇的に削減できる．このようなプッシュモデルにより，TTPはDoSアタックを受けにくいものにできる．

インクリメンタルCBEを使用すると，TTPは頻繁に（例えば1時間ごとに）顧客の秘密鍵を更新する場合でも，膨大な顧客数を非常に効率よく処理することができる．TTPは定期的に$R_{period}(\log(N/R_{period}))$件に対応する秘密鍵を計算すればよい．ここで，$R_{period}$はその期間中に公開鍵が無効になったユーザの数，$N$は顧客の合計数である．$N$を2億5,000万に設定し，毎年の取消率を10％と仮定すると，1時間あたり無効になるユーザ数はR_{hour}＝(2.5億人/年)/(10×365日×24時間）となり，その計算結果は約2,850件になる．このR_{hour}値を使用すると，TTPは毎秒約13件の証明書を計算すればよいことが分かる．1GHz Pentium IIIプロセッサでも毎秒約280件の（楕円曲線ベース）秘密鍵を計算できることから，TTPの計算量はまったく妥当なものになる．計算方法と送信条件の全詳細とその他の分析については文献[6]を参照していただきたい．要約すると，証明書不要のPKCにより，ドコモなどのTTPはインフラはもちろんのこと，膨大なユーザの公開鍵を扱うのに必要な計算量と送信のオーバヘッドを劇的に削減できる．

2.3 集約署名方式（Aggregated Signature）

文献[8]で初めて紹介された集約署名方式は，複数の署名者による複数の（異なってもかまわない）メッセージに対する複数のディジタル署名を，単一署名と同一ビット長に圧縮する．例えば，深さtのPKIでは，ユーザはt個の証明書で構成される一連の証明書をもつことになる．ここで各証明書は，子の公開鍵に対する親の署名になる．これに対して集約署名方式を用いることで，PKIに伴う帯域幅のオーバヘッドを削減でき，帯域幅に制約のある環境では特に有益である．

文献[8]に述べられているような，あるタイプの集約署名は，復号化鍵として使用することもできる．したがって，（証明書不要の公開鍵暗号を使用する）送信者は，受信者が自己のIDベース復号鍵だけでなく，署名のある別のドキュメントを保有することを，復号化の条件にすることができる．いくつかの指定したエンティティによる「許可」を受信者が復号化できる条件にするこのような考え方は，PKIの簡素化を超える用途があるかもしれない．

3. 特定用途に合わせた暗号方式

3.1 FSPの課金を検証できるマイクロクレジット方式

モバイル通信の爆発的な普及に伴い，ユーザはしばしばFSP（Foreign Service Provider）による付加価値サービスにアクセスする機会がある．これらのプロバイダがユーザと直接やり取りし，その後で提供したサービスに関する詳細をユーザのHSP（Home Service Provider）に提供，HSPがユーザに料金を請求するという形態が考えられる．ここでの懸念事項は，FSPがHSPに提示する利用料以上に請求する可能性である．このような問題に対しUSA研究所は，ユーザがサービスの利用に際し小さなトークンを発行し，それに基づいて後で課金するという方法（マイクロクレジット方式と呼ぶ）で取り組んでいる．この方式では，HSPの検証時間は発行トークン数の対数であり，FSPの検証時間が一定であるという点で効率がよい．さらに，ユーザと

FSP間のトークン発行に要する通信コストも一定である．

この方式の背後にある基本的な考え方は以下のとおりである．各ユーザは，いくつかの「マイクロクレジットトークン」を生成する．USA研究所の方式では，トークンが比較的簡易に生成できる．例えば，2.1GHz Pentium IVでは，約5,000のマイクロクレジットを4.2msで生成できる[*2]．さらに，トークンのサイズも小さい（平均40バイト）．ユーザは署名用秘密鍵を使って一連のトークンに署名するため，これらのトークンを当該ユーザに一意的に結び付けることができる．また後述するとおり，一定単位のトークン群に対して1つの署名でよい．これで，FSPとやり取りする場合に，ユーザはトークンを特定の間隔で（例えば，パケット単位あるいは5秒ごと）送付する．FSPは各トークンを検証するために，暗号ハッシュ関数を各々実行する．tインターバル後にサービスが終了すると，FSPはHSPに対し，t番目のトークンと一緒に，トークンを有効化する$O(\log t)$の付加値とともに送信する．HSPはこれを$O(\log t)$の時間で確認し，有効であれば，tインターバル分のFSPサービスの料金をユーザに請求することになる．

QuasiModoツリー（従来のMerkleの改良版）を使ってmマイクロクレジットを生成するには，ユーザはまず，$m+1$の20バイト値をランダムに生成する．（ほぼ）平衡のとれたバイナリツリーのリーフに，これらの値を割り当てる．便宜上，横型のツリーの頂点には0から始まる番号をふる．すなわち，ルートは$v[0]$で，その左の子が$v[1]$，右の子が$v[2]$と表現する．それぞれの分岐点には，

$$b(v[i]) = H(b(v[2i+1]), b(v[2i+2]))$$

が割り当てられる．このHは暗号ハッシュ関数であるため，同じ値にマッピングされる2つの入力は特定できない．最後に，ユーザはこのツリーのルート値$b(v[0])$に署名する．ここでこの署名の確認鍵はHSP発行のユーザ証明書に含まれると仮定している．ユーザがサービスを利用したい場合は，まずルート，署名，および証明書をFSPに送信し，FSPはこれらの値を検証する．時間iごとに，ユーザは$b[2i-1]$と$b[2i]$を送信する．FSPでは，

$$b(v[i-1]) = H(b(v[2i-1]), b(v[2i]))$$

であることを確認し，正しければトークンを受け入れる．ここで，サービスが時間t後に終了すると仮定する．FSPは$b(v[2i-1])$と一緒に，$v[2i-1]$からツリーのルートまでのパス上にある全頂点と対になる頂点の値$b[j]$を，ユーザの証明書原本とツリーへの署名$b(v[0])$とともにHSPに送付する．これらの値によって，HSPは$b(v[0])$を計算し，ユーザの署名を検証することができる．署名が合致したら，HSPは使用時間t分のサービスの料金をユーザに請求する．詳細については，文献[9]を参照していただきたい．

FSPが過剰請求したい場合は，基礎をなすディジタル署名方式を偽造するか，ハッシュ関数の衝突を見つけるしかない．偽造できないと思われる署名方式があり，衝突を見つけられないと思われる暗号ハッシュ関数があることが，この方式の安全の高さを示している．

3.2 スイスチーズ認証方式

USA研究所ではメディアストリームを効率的に認証するために，コンテンツ配信ネットワークでプロキシによって変更できる2つの方式，LISSA（LInear multiplex Scheme for Simulcast Authentication）とTRESSA（TREe Scheme for Simulcast Authentication）を開発した．これらの方式により，受信者がコンテンツプロバイダの署名，すなわち受信データの信頼性と完全性の検証を継続しながら，プロキシがストリームを途中で動的に変更することが可能となる．このような認証方式により，コンテンツプロバイダはデータストリーム全体を一度にエンコードして署名することができる．署名に関するプロセスは各フレームごとに署名したり，デバイスやネットワーク構成，チャネル品質の組合せごとに異なるバージョンをエンコードして署名するような非常に複雑な処理を要するプロセスとは対照的である．また，プロキシもネットワークサービスとして提供できる．

これらの方式は，プロキシでストリームなどのデータセットからその一部を削除するようなさまざまなケースに対し，最終受信者の署名検証機能を損なうことなく適用できる．例えば，複数ファイル選択では，プロキシが同じストリームの複数バージョン（低，中，高品質など）を受信し，チャネル状態に応じて送信するファイルを選択する場合が考えられる．あるいはプロキシで，ストリームを定期的に切り替えることもできる．また，一般的な技術でスケーラブルな圧縮を実行する場合，ストリームは基本レイヤといくつかの拡張レイヤに分割される．基本レイヤだけでもストリームを表示でき，拡張レイヤによって品質が改善される．ネットワークが混雑すると，プロキシは拡張レイヤを取り除いて，ほかのサービス品質（QoS：Quality of Service）保証に充当できる．USA研究所の方式は，動的な広告の配置にも使用できる．発信元はストリームに複数の広告を盛り込み，仲介者がユーザの好みなどに応じて広告を選択提供することができる．ここでは，ほかのシナリオはその特例とみなすことができることから，代表的なシナリオとして，複数ファイル選択に絞って説明する．この方

＊2　これらの数字はhttp://www.eskimo.com/~weidai/benchmarks.htmlから引用したものである．

式では，受信者の受け取るストリームがプロキシの操作により「穴」が開いた状態になることから，これをスイスチーズになぞらえて「スイスチーズ認証」と呼んでいる．

これらの方式は，次のように機能する．まず，どちらの方式もメディアストリームをフレーム（切替可能なデータ単位）に分割する．各フレームは，第1レイヤの暗号ハッシュ関数に個別に入力される．次に，その結果できたハッシュ値が第2レイヤの暗号ハッシュ関数によって再度ハッシュされ，ディジタル署名される．LISSA方式では，第2レイヤハッシュに標準反復チェーン構成が含まれている．TRESSA方式では，第2レイヤハッシュはMerkleツリーを使って実行される．これらの方式を図1に示す．

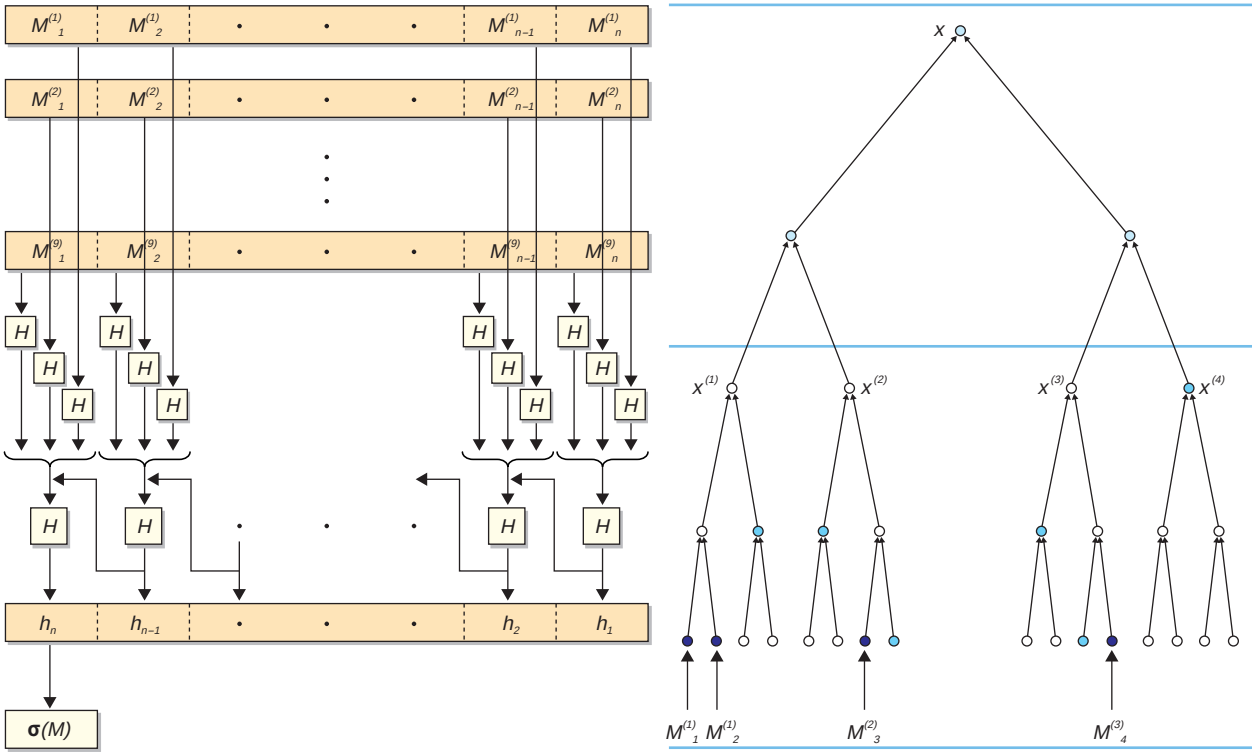
図1(a)では，LISSAで3つのストリームを処理している．各フレームは第1レイヤハッシュHでハッシュされ，その結果できたハッシュがさらにチェーン連結される．このチェーンが最終的なハッシュ値h_nに至るとディジタル署名され，σ(M）として署名になる．

図1(b)では，TRESSAで4つのストリームを処理している．この場合もやはり，各フレームがハッシュされ，リーフの頂点に値が割り当てられる．内部頂点は，その子ハッシュと結び付いた値をとる．最後に，ルートxが署名される．LISSAでは，フレームを欠落させる場合，そのフレームの第1レイヤハッシュを代わりに送信する．TRESSAでも同じ値を送信するのだが，フレームをクラスタ化してサブツリー全体のリーフを形成している場合は，そのサブツリーのルートの値を送信する．実装プロトタイプによれば，LISSA方式は各フレームに署名する基本的なアプローチよりだいたい3～18倍高速である．その他の詳細と包括的なセキュリティ分析については，文献[10]を参照していただきたい．

4. あとがき

本稿ではまず，PKIの簡素化に関する3つの研究結果について説明した．次に，コンテンツやサービスの安全な配布に関する2つの成果について述べた．すなわち，新型のハッシュツリーを使用して，HSPがパートナーであるFSPから正確な請求情報を安全に受信できるようにする非常に効率的なマイクロクレジット方式とエンド・ツー・エンドのセキュリティを壊すことなく中間プロキシがストリームを動的に変換するストリーム認証方式について説明した．

本稿は，暗号および情報セキュリティの分野におけるUSA研究所の研究概要を説明したものである．4Gビジョンは，多くのチャンスと課題を提示するものと考えており，ドコモのビジョンを実現するだけでなく，より大きなビジ



（a）LISSA方式　　（b）TRESSA方式

図1　LISSA方式とTRESSA方式

ネスチャンスを可能にするセキュリティ技術とビルディングブロックを開発することが目標である．

文　献

[1] S. Micali: “ Scalable Certificate Validation and Simplified PKI Management, ”Proc. of 1st Annual PKI Research Workshop, 2002.

[2] A. Shamir: “ Identity-Based Cryptosystems and Signature Schemes, ” Proc. of Crypto 1984.

[3] D. Boneh and M. Franklin: “ Identity-Based Encryption from the Weil pairing, ”Proc. of Crypto 2001.

[4] C. Cocks: “ An Identity-Based Encryption Scheme Based on Quadratic Residues, ”Proc of Cryptography and Coding, 2002.

[5] C. Gentry and A. Silverberg: “ Hierarchical ID-Based Cryptography, ” Proc. of Asiacrypt 2002.

[6] C. Gentry: “ Certificate-Based Encryption and the Certificate Revocation Problem, ”Proc. of Eurocrypt 2003.

[7] S. S. Al-Riyami and K. G. Paterson: “ Certificateless Public Key Cryptography, ”Proc. of Asiacrypt 2003.

[8] D. Boneh, C. Gentry, B. Lynn and H. Shacham: “ Aggregate and Verifiably Encrypted Signatures from Bilinear Maps, ” Proc. of Eurocrypt 2003.

[9] C. Gentry and Z. Ramzan: “ Microcredits for Verifiable Foreign Service Provider Metering, ”Technical Report.

[10] C. Gentry, A. Hevia, R. Jain. T. Kawahara and Z. Ramzan: “ Swiss-Cheese Authentication for Streaming Media, ”Technical Report.

用 語 一 覧

CA：Certificate Authority（認証局）
CBE：Certificate Based Encryption
CRL：Certificate Revocation List（証明書失効リスト）
DoS：Denial of Service
FSP：Foreign Service Provider
HIBC：Hierarchical Identity－Based Cryptography（階層的IDベース暗号方式）
HSP：Home Service Provider
IBC：Identity－Based Cryptography（IDベース暗号方式）
LISSA：LInear multiplex Scheme for Simulcast Authentication
OCSP：Online Certificate Status Protocol（オンライン証明書状態プロトコル）
PKC：Public Key Cryptography（公開鍵暗号）
PKG：Private Key Generator（鍵生成局）
PKI：Public Key Infrastructure（公開鍵暗号基盤）
QoS：Quality of Service（サービス品質）
TRESSA：TREe Scheme for Simulcast Authentication
TTP：Trusted Third Party（第三者信頼機関）