情報処理学会論文誌 Vol.57 No.12 2627–2640 (Dec. 2016)

CHAPのみで多要素認証を実現するプロトコル実装方法の提案と評価

稲村 勝樹[1,a)]

受付日 2016年3月10日, 採録日 2016年9月6日

概要：近年，パスワードリスト攻撃など，ユーザの安易なパスワード設定を原因とする不正アクセスが多発している．そのため，パスワードとあわせて認証を行う方式の提案・実装が急務となっている．一方，CHAP はインターネット上のサービスにおいて認証方式のデファクトスタンダードの 1 つとなっている．したがって，CHAP から新しい認証方式に変更することは容易なことではない．そこで，本稿ではサーバやユーザ端末での処理方法を改良することで，CHAP の構造をそのまま活用しながらパスワードとそれ以外の認証を同時に行うことができる新しい認証方式を提案する．これにより，既存の認証プロトコルで多要素認証が実現可能となる．また，サービスの継続性を考慮したプロキシサーバ型改良方式も提案する．さらに提案方式のテストシステムを構築し，パフォーマンス測定による実装評価を行う．これにより，提案方式を使用することで低コストでパスワード認証と他の認証の併用が実現できることを示す．

キーワード：認証プロトコル，CHAP，2 要素認証，多要素認証

Proposal and Evaluation of Methods for Mounting Protocol of Multi-factor Authentication over CHAP Only

Masaki Inamura[1,a)]

Received: March 10, 2016, Accepted: September 6, 2016

Abstract: Recently, unlawful access often happens caused by users' mistakes of password setting, e.g. password-list-attacks. Therefore, proposing/implementation of other authentication methods, which are used together with password authentication, are urgent. Meanwhile, CHAP is used a lot of services over the Internet as de facto standard. Therefore, it is not easy to change a new authentication protocol into CHAP. I propose a new method which can send many types of authentication codes using intact CHAP with revising calculation on user terminal and servers. As a result, a multi-factor authentication over existing password-authenticaion protocol can be realized. In addition, I propose a revised method of my proposal using a proxy server because of continuity in services. Furthermore, I structure a test system using proposed method and evaluate performances of it. By using my proposal, other authentication method using password authentication together can be realized with a minimum cost burden.

Keywords: authentication protocol, CHAP, two-factor authentication, multi-factor authentication

1. まえがき

近年，インターネットは水道・電気・ガス・電話などと並ぶ重要な社会基盤技術と位置付けられようになった．特に東日本大震災においては，SNS（Social Networking Service）により被害状況や救援物資の到達状況など様々な情報がやりとりされることで，救済や復旧過程において大きな役割を担ったことが理解された．また，大手企業や大規模組織だけでなく，中小企業や小規模な組織，さらに個人事業者においてもインターネットを活用した情報・サービス提供や商取引が行われており，今後ますますインターネットの重要度が増していくと予想される．このようなイ

[1] 東京電機大学理工学部
School of Science and Engineering, Tokyo Denki University, Hatoyama-machi, Hiki-gun, Saitama 350–0394, Japan
[a)] minamura@rd.dendai.ac.jp

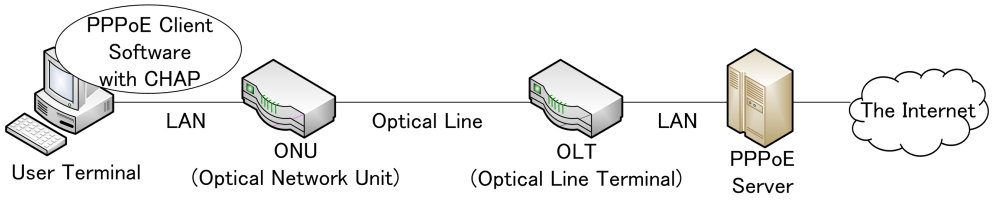

図 1 光回線による PPPoE における通信機器の構成
Fig. 1 Structure of telecomunication devices under PPPoE over optical line.

ンターネットを介して行われる様々な通信サービスではユーザ認証を行い，ユーザを特定したサービス提供を可能としている．

ユーザ認証において，最も低負荷かつ安全に行える認証プロトコルの 1 つにパスワードを用いたチャレンジ・レスポンス型認証プロトコルがある．その中でも CHAP（Challenge Handshake Authentication Protocol）[1] は，1 方向性ハッシュ関数を用いることで軽量な処理と安全性を実現し，ネットワークとの親和性が非常に高い．そのため，ユーザ端末とサービスプロバイダのサーバとの間を接続してデータ通信を行うための通信プロトコルである PPP（Point-to-Point Protocol）[2] のユーザ認証用など，幅広く標準的に使われている．一例として，光回線による PPPoE（PPP over Ethernet）における通信機器の構成を**図 1** に示す．ユーザ端末は ONU（Optical Network Unit）と LAN により接続され，ONU で光回線の信号に変換する．サービスプロバイダには OLT（Optical Line Terminal）が設置されており，PPPoE サーバと LAN により接続されている．ONU から送信された光回線の信号は OLT で LAN の信号に変換される．ユーザ端末には PPPoE クライアントソフトウェアを介して PPPoE サーバに接続し，そこを経由してインターネットに接続する．しかし，近年において，安易なパスワードの設定，あるいはパスワードリスト攻撃など，運用の不備やユーザのセキュリティ意識の低さを起因とする情報漏洩や不正利用の事件が頻発している [3]．

このような不正行為に対しては，パスワードを用いない認証方式を用意し，パスワード認証と併用してユーザ認証を行う多要素認証を導入することがセキュリティ対策の 1 つとしてあげられる [4]．特に一般的なのは，パスワードともう 1 つ別な認証用情報を利用する 2 要素認証と呼ばれる方式で，これまでにもいくつかの提案がされている [5], [6], [7], [8], [9], [10], [11], [12]．これらの提案方式において多要素認証を行うためには，CHAP とは異なる認証プロトコルの設計が必要となる．通信機器によっては，たとえば図 1 の中で ONU や OLT が認証方式として CHAP にしか対応していない場合があり，新たな認証プロトコルを実装するにあたり，これらの機器の改修や，場合によっては機器の交換が必要となる場合が考えられ，サービスの継続性やコストの増大といった課題が発生する．しかし，企業においては

- 近年の経済状況により IT 投資意向が減退している中で，情報セキュリティ投資についてもコスト削減が要求され，大規模なセキュリティ対策を行うことに対する経営層の理解を得られない．
- セキュリティ技術を導入・運用するのに必要とされる知識や能力を有する人材が不足している．
- 稼働している既存システムが存在し，新しい認証プロトコルのためのシステムの置き換え，あるいは機能追加を行う場合，既存システムで提供しているサービスが継続できるかの動作確認が必要となる．動かない場合はシステムの切り戻しも考慮する必要がある．
- 上記の作業を行うにあたって，長時間システムを停止することができない．

といった事情が存在する [13], [14], [15]．これらの事情は，すべての組織において，サイバー攻撃に対するセキュリティシステムを再構築しにくいといった課題があることを示している．

そこで本稿では，サーバやユーザ端末での演算処理を見直し，CHAP の通信パケットを拡張することなく 2 種類以上の認証情報を送ることが可能となる認証プロトコルについて検討を行う．特に，サービスの継続性を考慮し，現行のシステムへの影響を最小限にする多要素認証の実現手法を提案する．初めに，認証に用いる暗号学的アルゴリズムについて，1 方向性ハッシュ関数へ認証情報 2 種類を入力値とすることで CHAP のパケットフォーマットで記述されることを可能にし，ユーザとサーバとの間で一度の認証フェーズで既存のパスワード認証プロトコルの体系を崩すことなく 2 種類以上の認証情報によるユーザ認証を行えることを理論的に示す．これにより，既存の通信機器の改修や交換を行わず，コスト増大を抑えられることで多要素認証の導入・運用の促進が期待でき，不正侵入や情報漏洩などの危険性の削減につながるものと考えられる．さらに実際の運用ではサーバ側の対応（コスト面や，特に新システム導入時や問題発生した場合に切り戻すときのサービス継続性）を考える必要があることから，元のシステムを残したまま新しい方式を導入することが求められる．この点について，共通鍵暗号方式を用いたプロキシサーバ型の改良方式を提案することで，CHAP で運用されている通信機器やサーバを活用し，既存サービスの継続性も考慮しながら多要素認証の導入が可能となることを示す．さらにこの改

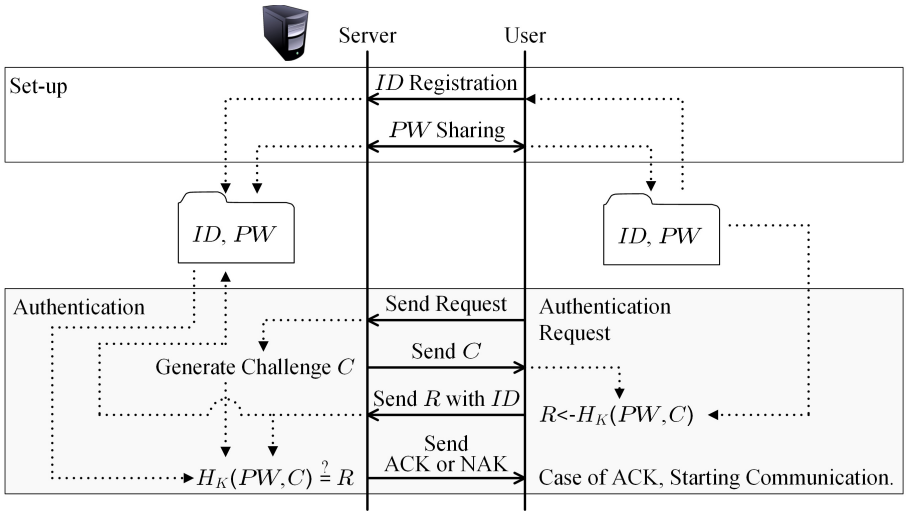

図 2 CHAP の手順
Fig. 2 Procedure of CHAP.

良方式について，テストシステムを構築して提案方式を実装し，パフォーマンス測定を行うことで，この方式が実現可能であることを示す．

本稿において，2 章で CHAP と既存の多要素認証方式の概要を示し，3 章でハッシュ関数への入力値に複数の認証情報を連接したものを用いることで，この CHAP の通信手順やパケットフォーマットをそのまま利用した多要素認証を可能とする提案方式について説明する．さらに 4 章で共通鍵暗号を用いることで追加の認証部分をプロキシサーバ化し，既存サービスの継続性にも考慮した改良方式について説明する．5 章でプロキシサーバ型改良方式を基に，2 要素認証を行うテストシステムの実装とその実験結果を示し，6 章でまとめとする．

2. 既存の認証方式

2.1 CHAP（Challenge Handshake Authentication Protocol）の概要

2.1.1 説明に用いる記号

この節において用いる記号は以下のとおりである．

ID：ユーザ識別子．

PW：認証に用いる秘密情報（パスワードなど）．

C：認証時に乱数生成器により生成されるチャレンジ．

R：レスポンス．

$H_K(\alpha,\beta)$：鍵付きハッシュ関数（鍵 α を用いてデータ β をハッシュ化）．

鍵付きハッシュ関数とはハッシュ化したいデータに鍵と呼ばれる数値データを連接して 1 方向性ハッシュ関数による計算を行う関数のことで，この鍵の共有者のみがハッシュ値の生成や検証を行うことが可能となる．代表的なものとしては HMAC（Hash-based Message Authentication Code）がある [16]．

2.1.2 前提条件

CHAP における前提条件は以下のとおりである．

- ユーザ認証を行うサーバは不正を行わない．
- パスワードはユーザとサーバで安全に共有し，第三者は知らないものとする．したがって，2.1.3 項における Set-up フェーズも安全な通信・手段（たとえば Set-up 用のセキュアな通信路を用意する，クライアントとサーバをダイレクトに接続し Set-up を行うなど）を用いて行われる．
- 第三者は通信において流れるパケット以外から認証に使われている情報を得ることはできない．

2.1.3 プロトコル手順

CHAP による認証では 2 つのフェーズがある．1 つは “Set-up” のフェーズであり，もう 1 つは “Authentication” のフェーズである．このプロトコル手順について**図 2** に示すとともに，それぞれの手順を以下に説明する．

Set-up:

(1) ユーザはユーザ登録として ID をサーバに送信する．

(2) ユーザは PW を生成してサーバに送信する．

(3) サーバは送られた ID と PW を関連付けて管理する．

Authentication:

(1) ユーザは認証要求として ID をサーバに送信する．

(2) サーバは C を生成し，この C をユーザに送信する．

(3) ユーザは PW とサーバから受信した C を用いて

$$R \leftarrow H_K(PW, C)$$

を計算し，この R をサーバに送信する．

(4) サーバは PW と (2) で生成した C を用いて

$$H_K(PW, C)$$

を計算し，この計算結果がユーザから受信した R と一致するか確認する．一致したら認証成功，不一致なら認証失敗とし，その認証結果をユーザに通知する．

2.1.4 パケットフォーマット

パケットフォーマットについて**図 3** に示すとともに，パケット内に示されている各フィールドの内容について以下に説明する．

パケットはヘッダ部とデータ部に分かれ，ヘッダ部は 4 バイトの固定長で構成される．5 バイト目からがデータ部となる．

ヘッダ部において，最初の 1 バイトは “Code” と呼ばれるフィールドで，パケットの種類を示している．ここで示される値は以下の 4 通りのみとなる．

0x01: Challenge.
0x02: Response.
0x03: Success.
0x04: Failure.

次の 1 バイトは “Identifier” と呼ばれるフィールドで，このフィールドの数値を基にユーザとの通信を行う．

ヘッダ部最後の 2 バイトは “Length” と呼ばれるフィールドで，ヘッダ部を含むパケット長を示している．

データ部は，“Code” で規定されている値を格納する．ただし，“Code” で示された値が “Success” か “Failure” であった場合は，このデータ部は通常 0 バイトとなる．

“Challenge” であった場合のパケットフォーマットを**図 4** に示す．データ部最初の 1 バイトは “Value Length” と呼ばれるフィールドで，この次にくる “Value” の長さを示している．2 バイト目からは “Value” と呼ばれるフィールドで，C を格納する．このとき，この C の大きさから “Value Length” の値を決定する．“Value” の後にくる最後のフィールドが “Name” であり，ここに ID を格納する．

上記の “Challenge” パケットに対応して返信される “Response” パケットのパケットフォーマットを**図 5** に示す．データ部最初の 1 バイトは “Value Length” と呼ばれるフィールドで，この次にくる “Value” の長さを示している．2 バイト目からは “Value” と呼ばれるフィールドで，R を格納する．このとき，この R の大きさから “Value Length” の値を決定する．“Value” の後にくる最後のフィールドが “Name” であり，ここに ID を格納する．

2.1.5 安全性

この手順はとてもシンプルであるが，1 方向性ハッシュ関数の性質から，第三者が認証時の通信を盗聴しても認証に必要な PW を入手することは計算理論的に難しいことが示されている [17]．また，C が乱数性を持ち，毎回異なる値が生成されるならば，リプレイ攻撃に対しても耐性がある．

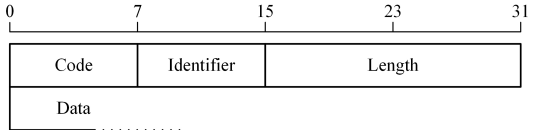

図 3 CHAP のパケットフォーマット
Fig. 3 Packet format of CHAP.

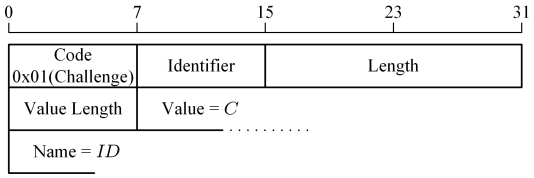

図 4 CHAP における “Challenge” パケットのフォーマット
Fig. 4 “Challenge” packet format of CHAP.

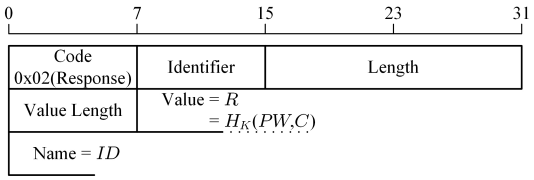

図 5 CHAP における “Response” パケットのフォーマット
Fig. 5 “Response” packet format of CHAP.

2.2 既存の多要素認証方式

これまでにも，通常のパスワード認証と合わせて利用する新たな多要素認証に関する提案が行われてきた [5], [6], [7], [8], [9], [10], [11], [12]．これらには方式の細かい違いはあるが，次の共通点がある．

- CHAP などのパスワード認証とは別の手順で 2 要素目以降の認証を行うようになっている．すなわち，パスワード認証とは独立な認証フェーズが存在する．
- パスワード認証と同時に 2 要素目以降の認証に用いるデータを同時に送信することができない．

このことにより，以下にあげるもののうち，少なくともいずれかの課題がある．

(1) CHAP とは異なるプロトコルを実装する必要がある．その場合，
  (a) CHAP しか通さないような通信機器を設置していた場合，新しい認証に対応した機器への改修または交換が必要となる．
  (b) CHAP とプロトコル体系が異なるために新たな安全性の定義と考察が必要だが，プロトコル部分の安全性に言及されていない．

  といった課題がある．

(2) CHAP などによるパスワード認証が完了後，改めて 2 要素目以降の認証を行う場合は，少なくともパスワード認証の処理は CHAP のプロトコル体系を変更する必要がない．しかし，
  (a) パスワード認証が成功しているため，少なくとも

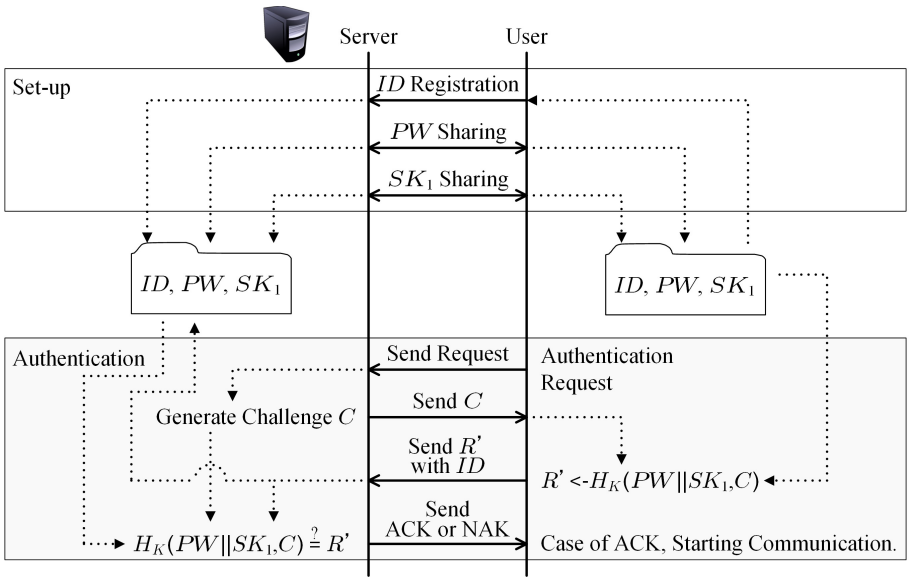

図 6 連接された認証情報を用いた提案方式の手順
Fig. 6 Procedure of a proposal using concatenation of verifying information.

たとえばPPPoEサーバまでのアクセスが可能となり，パスワードが漏洩していた場合はPPPoEサーバに侵入される危険性がある．

(b) パスワードおよび2要素目以降の認証全体の安全性に関して前項(1)(b)と同様の問題がある．

といった課題がある．

3. 連接した認証情報を用いた提案方式

3.1 プロトコル概要

3.1.1 記号

この章において用いる記号は以下のとおりである．

ID：ユーザ識別子．

PW：通常のCHAPで認証に用いる秘密情報（パスワードなど）．

SK_1：PW以外に認証に用いる秘密情報（PW以外に認証に用いる情報（たとえば端末固有情報や生体情報から得られる固定情報）から生成される値）．

C：認証時に乱数生成器により生成されるチャレンジ．

R'：提案方式において“Response”で実際に送信される値．

$H_K(\alpha,\beta)$：鍵付きハッシュ関数（鍵αを用いてデータβをハッシュ化）．

$\|$：前後の値の連接．

3.1.2 前提条件

提案方式における前提条件は以下のとおりである．

- ユーザ認証を行うサーバは不正を行わない．
- 認証に用いるパスワードはユーザとサーバで安全に共有し，第三者は知らないものとする．したがって，3.1.3項におけるSet-upフェーズも安全な通信・手段を用いて行われる．
- 第三者は通信において流れるパケット以外から認証に使われている情報を得ることはできない．

3.1.3 プロトコル手順

CHAPと同様に，提案方式による認証では“Set-up”と“Authentication”の2つのフェーズがある．このプロトコル手順について**図 6**に示すとともに，それぞれの手順を以下に説明する．

Set-up:

(1) ユーザはユーザ登録としてIDをサーバに送信する．

(2) ユーザはPWを生成してサーバに送信する．

(3) ユーザはさらにSK_1を生成してサーバに送信する．

(4) サーバは送られたID，PW，SK_1を関連付けて管理する．

Authentication:

(1) ユーザは認証要求としてIDをサーバに送信する．

(2) サーバはCを生成し，このCをユーザに送信する．

(3) ユーザはPW，SK_1とサーバから受信したCを用いて

$$R' \leftarrow H_K(PW\|SK_1, C)$$

を計算する．このR'をサーバに送信する．

(4) サーバはPW, SK_1と(2)で生成したCを用いて

$$H_K(PW\|SK_1, C)$$

を計算し，この計算結果がユーザから受信したR'と一致するか確認する．一致したら認証成功，不一致なら認証失敗とし，その認証結果をユーザに通知する．

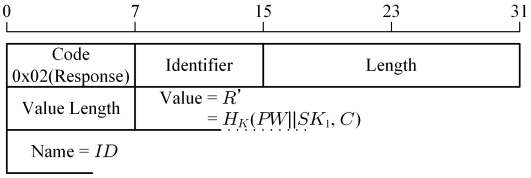

図 7 連接された認証情報を用いた提案方式における “Response” パケットのフォーマット

Fig. 7 “Response” packet format of a proposal using concatenation of verifying information.

3.1.4 “Response” パケットフォーマット

提案方式におけるパケットフォーマットについて，“Challenge”，“Success”，“Failure” の各パケットについては通常の CHAP と内容は一緒となる．この節では “Response” パケットフォーマットについて**図 7** に示す．

このパケットにおいて，CHAP と内容が異なるのは “Value” のフィールドとなる．CHAP のときとは異なる R' の値が格納されている．ただし，本提案の R' が 1 方向性ハッシュ関数による演算結果の値であり，CHAP での R と区別がつかない．したがって，“Value” フィールドのサイズが提案プロトコルで出力される値を格納できるような，CHAP をプロトコルとして透過的に扱う環境では，通信機器において R が変わったとしてもその差違を判別することができないため，通信機器が影響を受けることはない．

3.2 考察

3.2.1 安全性

3.1 節で示した提案方式において，3.1.4 項で示したとおり，“Response” パケットの “Value” フィールドに秘密情報が使用されている．したがって，2.2 節で示した他の多要素認証方式とは異なり，提案方式は CHAP における安全性と同様の議論が行え，この “Value” フィールドの情報の安全性の議論に集約できる．その情報は 1 方向性ハッシュ関数の性質から，第三者が認証時の通信を盗聴しても認証に必要な PW および SK_1 を入手することは計算理論的に難しい．また，CHAP と同様に C が乱数性を持ち，毎回異なる値が生成されるならば，リプレイ攻撃に対しても耐性がある．

3.2.2 実用性

3.1 節で示した提案方式において，以下の利点がある．

- 認証で使用する秘密情報である PW および SK_1 のうち，どれか 1 つでも異なるものがあれば正しい R' は生成されない．したがって，パスワードだけに頼るユーザ認証よりも安全性が強固になる．
- 提案方式において，CHAP と内容が異なるのは “Response” パケットの “Value” フィールドのみであるが，3.1.4 項で示したとおり，CHAP の “Response” パケットと区別がつかない．したがって CHAP を実装している通信機器のパケットフォーマットをそのまま利用可能で，通信機器の改修や交換を必要としない．
- “Response” パケットの “Value Length” フィールドで表現できる “Value” フィールドのサイズは 255 バイト，すなわち 2040 ビットである．1 方向性ハッシュ関数の出力値は，たとえば一般的に使用される SHA256 の場合は 256 ビットであり，十分に仕様の範囲といえる．
- “Challenge”，“Success”，“Failure” の各パケット，および R の値は CHAP と同じである．したがって，R' の演算処理，および SK_1 の管理を行う追加認証サーバを設計し，既存サーバの前に設置することで提案方式が実現可能である．

このことは，特に 2.2 節の (1) (a) で示した他の多要素認証方式に対して，実装上の利点となる．

一方，この提案方式を利用する場合，以下の点に注意する必要がある．

- もし認証に失敗した場合，ユーザに対して PW および SK_1 のうちどの秘密情報に問題があったかを提示することができない．
- ユーザ端末の認証機能，およびサーバの機能は変更する必要があり，それぞれのソフトウェア改修のコストは必要となる．
- サーバは鍵付きハッシュ関数への入力値の連接処理があるため，既存サーバソフトウェアの更新が必須となり，その更新におけるサービスの継続性の課題が生じる．
- 前提により，攻撃者は通信パケット以外の情報を知ることはできない．しかし，もしサーバに侵入ができた場合は，同一サーバで認証情報を管理していたら，パスワードのみの漏洩リスクと変わらない．これに対し，各認証情報ごとに管理するサーバ（もしくはデータベース）を分散することで，すぐに 2 つの認証情報を入手できないようにする方法が考えられる．この場合，2 つの認証情報にアクセスする必要があることから，完全に認証が突破されるまでに多少の時間稼ぎを行える可能性がある．ただし，この場合においても認証サーバでは 2 つの認証情報を使用することから，認証サーバに侵入され通信パケットや処理内容を不正に入手することで，パスワードのみの漏洩リスクよりもそれほど大きな効果は得られず，またその場合はその設備を準備するコストが生じる．

4. 共通鍵暗号を用いたプロキシサーバ型改良方式

4.1 連接した認証情報との差違

3 章において，連接した認証情報を用いることで，CHAP のパケットフォーマットで記述することが可能な多要素認

証を構成できることを示した．一方で，3.2.2 項の注意で示したように，サーバソフトウェアの改修が必須となる．この場合，ログインのための認証情報の構成も変更する必要があり，また切り戻し時においてもその構成を含めて切り戻す必要がある．そのため，既存サーバソフトウェアの更新におけるサービスの継続性の課題が残されている．

この課題に対し，追加認証部分をプロキシサーバ化する方式で解決可能であると考えられる．この方式では，多要素認証へ移行する直前まで通常の CHAP 認証によるサービスを継続し，多要素認証開始時は CHAP 認証用の既存サーバと広域ネットワークの間にプロキシサーバを挟むような経路変更を行う．経路変更の手順は，たとえば以下のとおりに行われる．

(1) ユーザにサービスの変更時期を通告する．
(2) 変更時期になったら，プロキシサーバの機能を立ち上げる．
(3) 既存認証サーバは広域ネットワークとの接続ルータに向けていた接続先を，接続ルータは既存認証サーバに向けていた接続先を，それぞれプロキシサーバに設定する．
(4) 接続ルータ，プロキシサーバ，既存認証サーバの接続が完了したら，ユーザにサービス再開を通告する．
(5) ユーザは認証用のソフトウェアを新しい “Response” パケットを生成しサーバ側に送信するものに変更し，そのソフトウェアで認証を行う．

ここで上記の手順 (3) において，近年ではネットワークの設定はリブートを必要とせず変更できる機器もあり，その結果経路変更の時間は少なくて済む．もちろん安全を期するのであれば一度既存認証サーバや接続ルータを広域ネットワークから遮断して作業をするのがよいが，その場合においても手順 (3) で変更するのは既存認証サーバ，および接続ルータのネットワーク設定のみであり，この 2 つの機器についてシステムの大きな変更を必要としないことから，遮断時間を短くすることが期待できる．ユーザ側は，新しい “Response” パケットを生成するソフトウェアに変更することで，新しい認証方式に対応させる．またパスワード認証における認証情報を含めて既存サービスで運用されていたシステムが残されており，導入・切り戻し時には以下の手順で行う．

(1) ユーザにサービスの切り戻し時期を通告する．
(2) 既存認証サーバはプロキシサーバに向けていた接続先を広域ネットワークとの接続ルータに，広域ネットワークとの接続ルータはプロキシサーバに向けていた接続先を既存認証サーバに，それぞれ設定を戻す．
(3) 接続ルータ，既存認証サーバの接続が完了したら，ユーザにサービス再開を通告する．
(4) ユーザは認証用のソフトウェアを以前の “Response” パケットを生成しサーバ側に送信するものに変更し，そのソフトウェアで認証を行う．

この場合においても，手順 (2) においてネットワークの接続設定を変更するだけでよいため，サービスの停止時間を短くすることが期待できる．ユーザ側は，以前の “Response” パケットを生成するソフトウェアに変更することで，切り戻した際の認証方式に対応させる．

ただし，3 章の方式では PW と SK_1 を同時に用いて “Response” パケットの “Value” フィールドの値を計算しているため 2 つの認証情報の処理を分離することができず，そのままでは追加認証部分についてプロキシサーバを導入できない．そこで，この “Value” フィールドの値の計算方法を改良し，追加認証部分については共通鍵暗号を用いた認証方式を採用する．これは，既存認証方式で用いるレスポンス値を追加認証部分で用いる SK_1 を共有鍵として暗号化するものである．これにより，まずプロキシサーバにおいて追加認証部分における復号処理を行い，復号された情報は既存サーバに送信され，パスワード認証の処理を行う．

これにより，CHAP 認証から多要素認証への移行がサーバ側における既存の機器のネットワーク設定の変更とユーザ側の認証用ソフトウェアの更新だけで行えると同時に，追加認証機能を導入した際に不具合が生じたとしても同様にサーバ側のネットワーク設定の変更とユーザ側の認証用ソフトウェアを元に戻す作業だけで切り戻しが行え，サービスの継続性の問題を最小限にする．

4.2 プロトコル概要

4.2.1 記号

この章において用いる記号は以下のとおりである．

ID：ユーザ識別子．
PW：通常の CHAP で認証に用いる秘密情報（パスワードなど）．
SK_1：共通鍵暗号に用いる共通鍵（PW 以外に認証に用いる情報（たとえば端末固有情報や生体情報から得られる固定情報）から生成される値）．
C：認証時に乱数生成器により生成されるチャレンジ．
R：通常の CHAP で用いられるレスポンス．
R'：改良方式において “Response” で実際に送信される値．
$H_K(\alpha,\beta)$：鍵付きハッシュ関数（鍵 α を用いてデータ β をハッシュ化）．
$ENC(\alpha,\beta)$：共通鍵暗号による暗号化（鍵 α を用いてデータ β を暗号化）．
$DEC(\alpha,\beta)$：共通鍵暗号による復号（鍵 α を用いてデータ β を復号，上記 $ENC(\alpha,\beta)$ に対応）．

4.2.2 前提条件

改良方式における前提条件は以下のとおりである．

- ユーザ認証を行うサーバは不正を行わない．

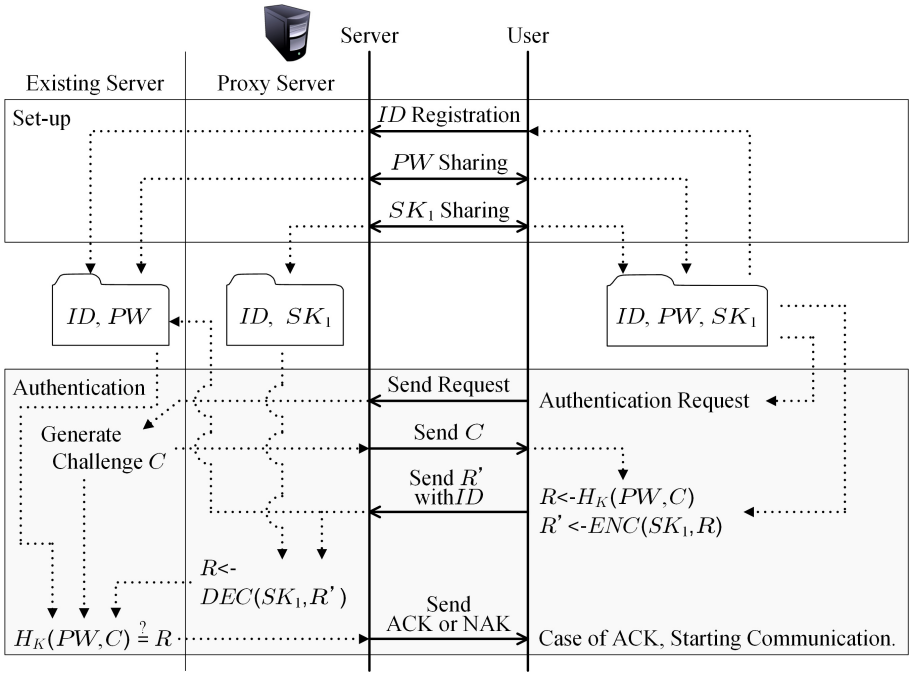

図 8 プロキシサーバ型改良方式の手順
Fig. 8 Procedure of a revised version using a proxy server.

- 認証に用いるパスワードと共通鍵はユーザとサーバで安全に共有し，第三者は知らないものとする．したがって，4.2.3 項における Set-up フェーズも安全な通信・手段を用いて行われる．
- 第三者は通信において流れるパケット以外から認証に使われている情報を得ることはできない．

4.2.3 プロトコル手順

CHAP と同様に，改良方式による認証では “Set-up” と “Authentication” の 2 つのフェーズがある．このプロトコル手順について**図 8** に示すとともに，それぞれの手順を以下に説明する．

Set-up:

(1) ユーザはユーザ登録として ID をサーバに送信する．
(2) ユーザは PW を生成してサーバに送信する．
(3) ユーザはさらに SK_1 を生成してサーバに送信する．
(4) サーバは送られた ID，PW，SK_1 のうち，ID と PW を既存サーバ側で，ID と SK_1 をプロキシサーバ側で，それぞれ関連付けて管理する．

Authentication:

(1) ユーザは認証要求として ID をサーバに送信する．
(2) 既存サーバは C を生成し，この C をユーザに送信する．
(3) ユーザは PW とサーバから受信した C を用いて

$$R \leftarrow H_K(PW, C)$$

を計算する．この R を，パスワード以外の秘密情報から生成した共通鍵 SK_1 を用いて

$$R' \leftarrow ENC(SK_1, R)$$

を生成する．この R' をサーバに送信する．
(4) プロキシサーバは SK_1 とユーザから受信した R' を用いて

$$R \leftarrow DEC(SK_1, R')$$

を計算し，この R を既存サーバに送信する．
(5) 既存サーバは PW と (2) で生成した C を用いて

$$H_K(PW, C)$$

を計算し，この計算結果がプロキシサーバから受信した R と一致するか確認する．一致したら認証成功，不一致なら認証失敗とし，その認証結果をユーザに通知する．

4.2.4 “Response” パケットフォーマット

改良方式におけるパケットフォーマットについて，3.1.4 項と同様に “Challenge”，“Success”，“Failure” の各パケットについては通常の CHAP と内容は一緒となり，“Response” パケットフォーマットについて**図 9** に示す．このパケットにおいて，CHAP と内容が異なるのは “Value” のフィールドとなるが，R' の値は暗号化の結果であり，3.1.4 項と同様にこちらも乱数との区別がつかない．したがって，“Value” フィールドのサイズが提案プロトコルで出力される値を格納できるような，CHAP をプロトコルとして透過的に扱う環境では，通信機器において R が R' に変わったとしても

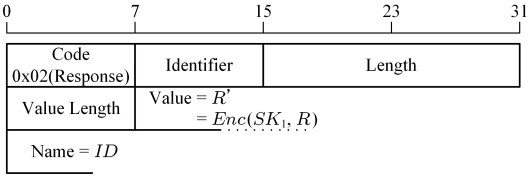

図 9 改良方式における “Response” パケットのフォーマット
Fig. 9 “Response” packet format of a revised version.

その差違を判別することができないため，通信機器が影響を受けることはない．

4.3 考察

4.3.1 安全性

4.2 節で示した改良方式において，4.2.4 項で示したとおり，“Response” パケットの “Value” フィールドに秘密情報が使用されている．したがって，2.2 節で示した他の多要素認証方式とは異なり，提案方式は CHAP における安全性と同様の議論が行え，この “Value” フィールドの情報の安全性の議論に集約できる．その情報は共通鍵暗号や 1 方向性ハッシュ関数の性質から，第三者が認証時の通信を盗聴しても認証に必要な PW および SK_1 を入手することは計算理論的に難しい．また，CHAP と同様に C が乱数性を持ち，毎回異なる値が生成されるならば，リプレイ攻撃に対しても耐性がある．

4.3.2 実用性

4.2 節で示した改良方式において，以下の利点がある．

- 認証で使用する秘密情報である PW および SK_1 のうち，どれか 1 つでも異なるものがあれば正しい R' は生成されない．したがって，パスワードだけに頼るユーザ認証よりも安全性が強固になる．
- 提案方式において，CHAP と内容が異なるのは “Response” パケットの “Value” フィールドのみであるが，4.2.4 項で示したとおり，CHAP の “Response” パケットと区別がつかない．したがって CHAP を実装している通信機器のパケットフォーマットをそのまま利用可能で，通信機器の改修や交換を必要としない．
- “Response” パケットの “Value Length” フィールドで表現できる “Value” フィールドのサイズは 255 バイト，すなわち 2040 ビットである．1 方向性ハッシュ関数の出力値は，たとえば一般的に使用される SHA256 の場合は 256 ビットである．また，共通鍵暗号の出力値については，たとえば一般的に使用される AES の場合はブロックサイズが 128 ビットであり，15 ブロック分まで格納することができる．したがって，十分に仕様の範囲といえる．
- “Challenge”，“Success”，“Failure” の各パケット，および R の値は CHAP と同じである．したがって，R' の暗号化・復号の処理，および SK_1 の管理を行う追加認証サーバを設計し，既存サーバの前に設置することで提案方式が実現可能であり，簡便かつ短時間で既存システムから更新可能である．

このことは，特に 2.2 節の (1) (a) で示した他の多要素認証方式に対して，実装上の利点となる．

一方，この提案方式を利用する場合，以下の点に注意する必要がある．

- この方式は，パスワード認証を行うオリジナルの R を復元する必要があることから，共通鍵暗号を利用している．このことから，一般的に共通鍵暗号よりも処理が速い 1 方向性ハッシュ関数のみで構成される 3 章の提案方式より処理が遅くなることが予想される．
- もし認証に失敗した場合，ユーザに対して PW および SK_1 のうちどの秘密情報に問題があったかを提示することができない．
- ユーザ端末の認証機能については共通鍵暗号による暗号化の機能が追加されている．そのため，ユーザ端末において認証用のソフトウェア改修のコストは必要となる．
- 前提により，攻撃者は通信パケット以外の情報を知ることはできない．しかし，もしサーバに侵入ができた場合は，同一サーバで認証情報を管理していたら，パスワードのみの漏洩リスクと変わらない．これに対し，各認証情報ごとに管理するサーバ（もしくはデータベース）を分散することで，プロキシサーバ，既存認証サーバともに使用する認証情報のみしかアクセスできない設計とすることができる．この場合，仮にどちらかのサーバに侵入されたとしても入手できる認証情報はそのサーバで使用している情報のみとなり，もう一方の情報へはアクセスできないため，サービスの不正使用に必要な認証情報すべてを入手することが困難であり，このリスクを回避することが可能となる．ただし，その場合はその設備を準備するコストが生じる．

5. テストシステム実装

5.1 テスト概要

3 章ではパスワード認証のプロトコル体系のみで 2 要素認証が行える方式を理論的に構築できることを示し，4 章では実際のサービス提供に合わせ，既存のサーバ側の機器を流用できる現実的な改良方式を示した．この章では，現実的な運用方式における実現性について検証するため，4 章で提案した改良方式について，テストシステムを構築し，実装実験によりパフォーマンス測定を行う．

本実験では，大学における学生用ポータルサイトを設定し，Web を用いて授業などの情報にアクセスすることを想定する．そのポータルサイトへのアクセス時に，改良方式による認証を行う．テストシステムは CHAP 認証と改良方式による認証とを選択できるようにし，同一スペックの

マシンにおいてパフォーマンスを比較する．サーバ，クライアントのマシンはすべて仮想マシンにより実装し，それぞれの認証方式について複数のユーザからの認証処理を行うスレッドを仮想的に発生させる．本実験で使用したサーバマシン・ソフトウェアの組合せにおける同時スレッドの起動限界が 4,000 スレッドであり，そこまでにおける 1 秒間に処理できる認証の件数を 20 回計測し，その平均により評価した．

5.2 システム構成

構築したマシンの構成を以下に示す．なお，認証サーバ（兼 Web サーバ），プロキシサーバ，クライアントのマシンについてはハードウェア構成，仮想マシンのソフトウェア，仮想マシン上の OS については共通のものを使用している．

ハードウェア：

- **サーバマシン**：Intel Server System R 1000GZ/GL Product Family
- **CPU**：Intel Xeon E5-2620 v2（2 コアのみ稼働，または 8 コア稼働）
- **RAM**：DDR3-1333 8 GB

ソフトウェア：

- **OS 兼仮想マシン**：VMware ESXi5.5.0
- **仮想マシン上の OS**：Ubuntu 14.04 LTS desktop
- **Web サーバ**：Apache 2.4.7
- **Web ブラウザ**：Google Chrome 39.0.2171
- **セキュリティ**：OpenSSL 1.0.1j（1 方向性ハッシュ関数：SHA256，共通鍵暗号：AES-128）
- **ユーザ管理データベース**：MySQL 5.5.40

5.3 実験結果

CPU を 2 コアのみ稼働させた状態の実験結果を**図 10** に示す．

このとき，CHAP 認証は 1 秒間に約 550 人のスループットが得られるのに対し，改良方式による認証は 1 秒間に約 300 人のスループットとなっており，約 1.8 倍の差があることが示されている．また，CHAP 認証は 3,000 スレッドまでスループットが上昇するのに対し，改良方式による認証は 1,000 スレッドで頭打ちになっていることが示されている．

次に，CPU を 8 コア稼働させた状態の実験結果を**図 11** に示す．

このとき，どちらの認証も 1 秒間に 500 人以上のスループットが得られ，その差も 10%以下であることが示されている．さらに 2 コア稼働時と比較して，CHAP 認証は稼働コア数による差違は見受けられないが，改良方式による認証は稼働コア数による改善が見受けられる．また，どちらの認証も 3,000 スレッドまでスループットが上昇することが示されている．

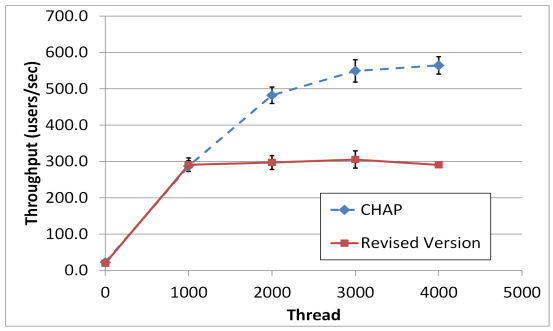

図 10 CPU 2 コア稼働時におけるスループット結果
Fig. 10 Results of throughput turning on 2 cores in the CPU.

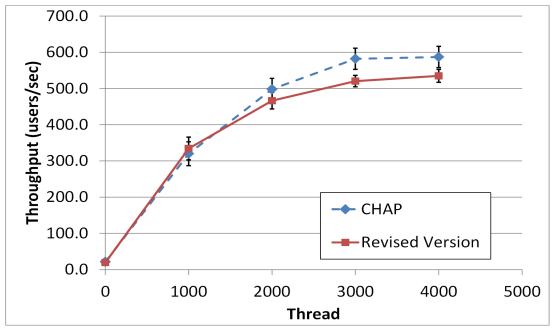

図 11 CPU 8 コア稼働時におけるスループット結果
Fig. 11 Results of throughput turning on 8 cores in the CPU.

この実験結果から，処理そのものの負荷は CHAP よりも改良方式の方が大きいが，改良方式はマルチコアによる分散処理を行うことでスループットの大幅な上昇が見込めると考えられる．一方，CHAP 認証は元々のプロトコルが軽量に作られているため，少ないコア数でも高速な処理が可能であるが，分散処理によるスループットの上昇はほとんど見込めないと考えられる．

したがって，分散処理が可能なシステムやマルチコア CPU を採用したサーバを用いることで，提案した改良方式は十分実用的であると考えられる．特に 2 要素認証で利用する場合は，8 コア以上の CPU を用いれば CHAP 認証と同程度のスループットが得られ，既存システムの新しい認証として十分機能すると考えられる．

RAM の搭載量の差違による実験も行ったが，2015 年現在において一般的に手に入る最小メモリサイズでもある 1 GB 以上の RAM を搭載していれば，計測されたスループットはほとんど同じであった．これは，認証で用いるデータサイズが非常に小さい（1 人あたりのデータサイズを 1 KB と仮定し，図 10 および図 11 で示された同時スレッドの起動限界まで処理を行ったと考えても，必要なメモリサイズは計算上たかだか数 MB〜十数 MB である）ことから，RAM の搭載量による差違がなかったと考えられる．

この実験では共通鍵暗号に AES-128 を用いて認証を行っている．共通鍵暗号は CHAP で使用されている 1 方向性

ハッシュ関数よりも処理が重いため，他の共通鍵暗号でもCHAPよりはスループットが低くなると推定できる．しかし，共通鍵暗号には様々な種類があり，そのアルゴリズムの特徴（処理負荷，並列処理による高速化の可能性）により暗号化処理のスループットは異なってくる．したがって，AES-128よりも低負荷，あるいは並列処理の効率が良い共通鍵暗号を用いることで，さらなるスループットの向上が期待できる．

6. まとめ

本稿では，必要とされる改修・変更コストを極力少なくし，かつ機能として十分な安全性を確保しうる多要素認証方式を検討・提案した．認証において，既存のCHAPのプロトコル手順およびパケットフォーマットはそのまま活用し，1方向性ハッシュ関数を用いることで“Response”パケットの“Value”フィールドの値に複数の秘密情報からなる認証用のレスポンス値を生成する．これにより，サーバとユーザ端末のソフトウェア改修のみで，通信機器に変更を加えることなく，導入コストを抑えつつ安全性の向上が期待できる．さらに共通鍵暗号を用いることで，追加の認証部分をプロキシサーバで行う改良方式も提案した．これにより，追加認証部分のみを設計し，既存システムへの追加（および不具合発生時の切り離し）を容易にすることで，システム更新にともなうサービス停止時間を短くでき，サービスの継続性も考慮できる．

また，実際にテストシステムを構築し，実用面におけるパフォーマンス測定を行った．その結果，現在の一般的なサーバの能力があれば，提案方式が十分実用可能であることを示した．

今後は4.3.2項で示した注意点に対する改善案，ならびに5章のシステム実装におけるさらなるパフォーマンスの改善を検討する．さらに共通鍵暗号を用いることなく，3章で示した連接した認証情報を用いる方式によるサービスの継続性を検討する．

謝辞 本稿で行った実験にあたり，ハードウェア貸与，テストシステム構築・実装，パフォーマンス測定に関して多大なるご協力をいただいた東京電機大学理工学部情報システムデザイン学系システム評価研究室の藤本衡准教授，阿部智様，石野純様，北山珠実様に感謝いたします．

参考文献

[1] Simpson, W.A.: PPP Challenge Handshake Authentication Protocol (CHAP), Request for Comments, RFC 1994 (1996).
[2] Simpson, W.A.: The Point-to-Point Protocol (PPP), Request for Comments, RFC 1661 (1994).
[3] 独立行政法人情報処理推進機構：情報セキュリティ白書2014 (2014).
[4] Nagel, B.: Password Seeks Partner for Long-Term, Secure Relationship, *CIO: Business Technology Leadership*, Vol.11, No.2, pp.28–31, IDG (2010).
[5] Schneier, B.: Two-Factor Authentication: Too Little, Too Late, *Comm. ACM*, Vol.48, No.4, p.136 (2005).
[6] Hagalisletto, A.M. and Riiber, A.: Using the Mobile Phone in Two-Factor Authentication, *Proc. International Workshop on Security for Spontaneous Interaction* (*IWSSI 2007*) (2007).
[7] Aloul, F.A., Zahidi, S. and El-Hajj, W.: Two Factor Authentication Using Mobile Phones, *Proc. IEEE/ACS International Conference on Computer Systems and Applications* (*AICCSA 2009*), pp.641–644, IEEE press (2009).
[8] Rathgeb, C. and Uhl, A.: Two-Factor Authentication or How to Potentially Counterfeit Experimental Results in Biometric Systems, *Proc. International Conference on Image Analysis and Recognition* (*ICIAR 2010*), LNCS 6112, pp.296–305, Springer (2010).
[9] Fan, C., Ho, P. and Hsu, R.: Provably Secure Nested One-Time Secret Mechanisms for Fast Mutual Authentication and Key Exchange in Mobile Communications, *IEEE/ACM Trans. Networking*, Vol.18, No.3, pp.996–1009 (2009).
[10] Eldefrawy, M.H., Alghathbar, K. and Khan, M.K.: OTP-Based Two-Factor Authentication Using Mobile Phones, *Proc. International Conference on Information Technology: New Generations* (*ITNG 2011*), pp.327–331, IEEE press (2011).
[11] Acharya, S., Polawar, A. and Pawar, P.Y.: Two Factor Authentication Using Smartphone Generated One Time Password, *IOSR J. Computer Engineering*, Vol.11, No.2, pp.85–90 (2013).
[12] Hwang, T. and Gope, P.: Provably Secure Mutual Authentication and Key Exchange Scheme for Expeditious Mobile Communication Through Synchronously One-Time Secrets, *Wireless Personal Communications*, Vol.77, No.1, pp.197–224 (2014).
[13] 独立行政法人情報処理推進機構：2015年度中小企業における情報セキュリティ対策に関する実態調査，IPA調査・研究報告書，available from ⟨http://www.ipa.go.jp/files/000051252.pdf⟩ (2016).
[14] 杉浦 昌，谷川哲司：レガシー・マイグレーションにおけるセキュリティ上の問題点，NEC技法，Vol.58, No.2, pp.38–42 (2005).
[15] 中村修二：レガシー・マイグレーションの現実解，*UNISYS TECHNOLOGY REVIEW*, Vol.89, pp.69–83 (2006).
[16] Krawczyk, H., Bellare, M. and Canetti, R.: HMAC: Keyed-Hashing for Message Authentication, *Request for Comments*, RFC 2104 (1997).
[17] Rogaway, P. and Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance, *Proc. International Workshop on Fast Software Encryption* (*FSE 2004*), LNCS 3017, pp.371–388, Springer (2004).

付 録

A.1 認証情報の多要素化対応

A.1.1 連接した認証情報を用いた提案方式の多要素化

A.1.1.1 記号

この節において用いる記号は以下のとおりである．

ID：ユーザ識別子．

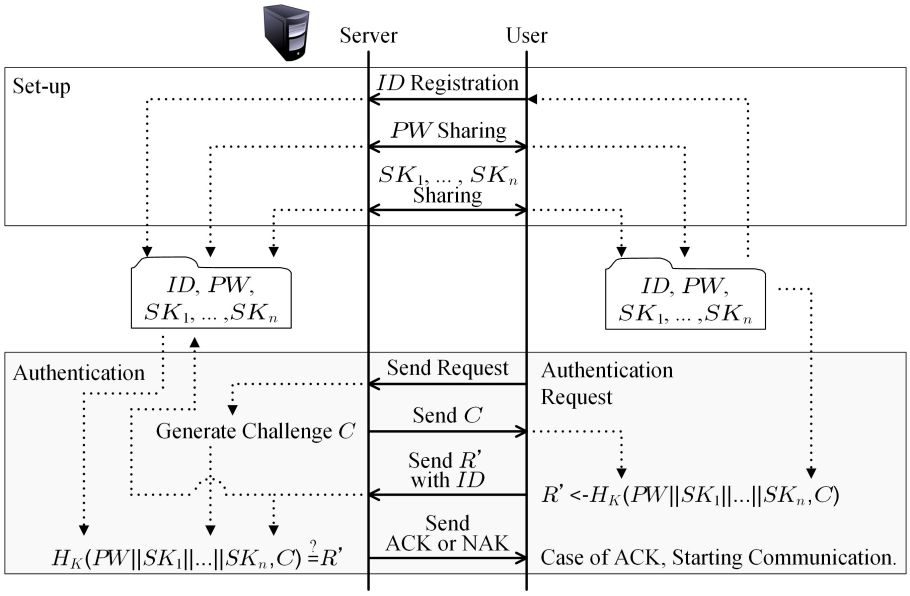

図 **A·1** 連接された認証情報を用いた提案多要素認証方式の手順

Fig. A·1 Procedure of a proposed multi-factor authentication using concatenation of verifying information.

PW：通常の CHAP で認証に用いる秘密情報（パスワードなど）．

SK_i：PW 以外に認証に用いる秘密情報（PW 以外に認証に用いる情報（たとえば端末固有情報や生体情報から得られる固定情報）から生成される値）$[1 \leq i \leq n,$ n は PW を除いて認証に用いる情報の個数$]$．

C：認証時に乱数生成器により生成されるチャレンジ．

R'：提案方式において “Response” で実際に送信される値．

$H_K(\alpha, \beta)$：鍵付きハッシュ関数（鍵 α を用いてデータ β をハッシュ化）．

$\|$：前後の値の連接．

A.1.1.2 プロトコル手順

CHAP と同様に，提案方式による認証では “Set-up” と “Authentication” の 2 つのフェーズがある．このプロトコル手順について**図 A·1** に示すとともに，それぞれの手順を以下に説明する．

Set-up:

(1) ユーザはユーザ登録として ID をサーバに送信する．

(2) ユーザは PW を生成してサーバに送信する．

(3) ユーザはさらに $SK_1, \ldots, SK_n$ を生成してサーバに送信する．

(4) サーバは送られた $ID, PW, SK_1, \ldots, SK_n$ を関連付けて管理する．

Authentication:

(1) ユーザは認証要求として ID をサーバに送信する．

(2) サーバは C を生成し，この C をユーザに送信する．

(3) ユーザは PW とサーバから受信した C を用いて

$$R' \leftarrow H_K(PW \| SK_1 \| \ldots \| SK_n, C)$$

を計算する．この R' をサーバに送信する．

(4) サーバは $PW, SK_1, \ldots, SK_n$ と (2) で生成した C を用いて

$$H_K(PW \| SK_1 \| \ldots \| SK_n, C)$$

を計算し，この計算結果がユーザから受信した R' と一致するか確認する．一致したら認証成功，不一致なら認証失敗とし，その認証結果をユーザに通知する．

A.1.2 プロキシサーバ型改良方式の多要素化

A.1.2.1 記号

この節において用いる記号は以下のとおりである．

ID：ユーザ識別子．

PW：通常の CHAP で認証に用いる秘密情報（パスワードなど）．

SK_i：共通鍵暗号に用いる共通鍵（PW 以外に認証に用いる情報（たとえば端末固有情報や生体情報から得られる固定情報）から生成される値）$[1 \leq i \leq n,$ n は PW を除いて認証に用いる情報の個数$]$．

C：認証時に乱数生成器により生成されるチャレンジ．

R：通常の CHAP で用いられるレスポンス．

R'：改良方式において “Response” で実際に送信される値．

$H_K(\alpha, \beta)$：鍵付きハッシュ関数（鍵 α を用いてデータ β をハッシュ化）．

$ENC(\alpha, \beta)$：共通鍵暗号による暗号化（鍵 α を用いて

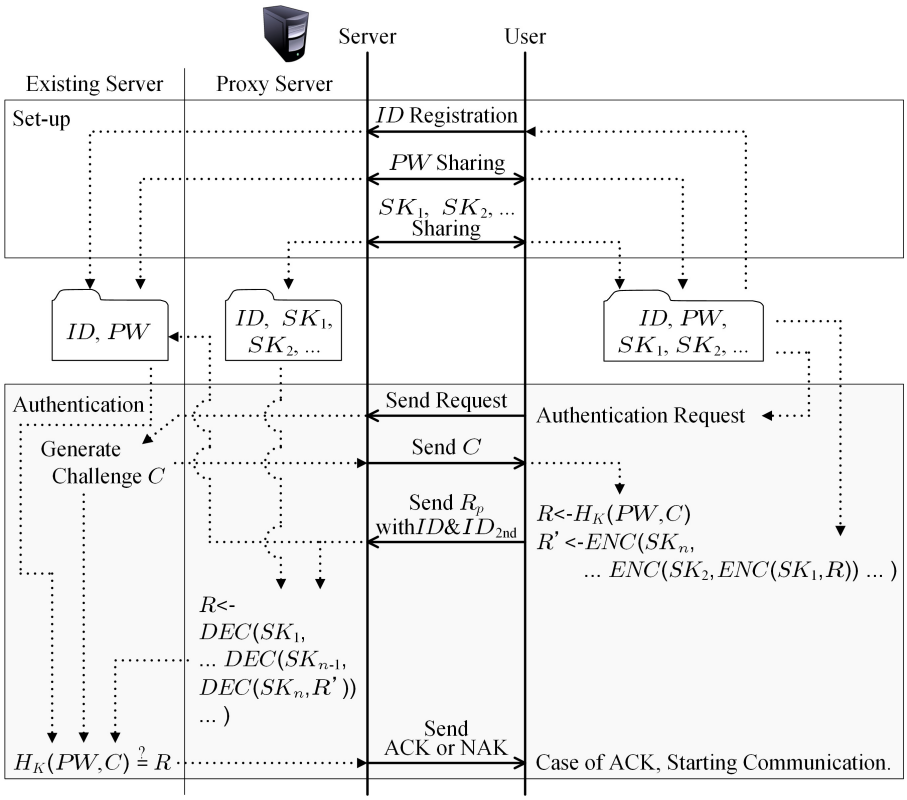

図 A·2 プロキシサーバ型多要素認証方式の手順

Fig. A·2 Procedure of a revised multi-factor authentication using a proxy server.

データ β を暗号化).

$DEC(\alpha,\beta)$：共通鍵暗号による復号（鍵 α を用いてデータ β を復号，上記 $ENC(\alpha,\beta)$ に対応).

A.1.2.2 プロトコル手順

CHAP と同様に，改良方式による認証では “Set-up” と “Authentication” の 2 つのフェーズがある．このプロトコル手順について図 **A·2** に示すとともに，それぞれの手順を以下に説明する．

Set-up:

(1) ユーザはユーザ登録として ID をサーバに送信する．

(2) ユーザは PW を生成してサーバに送信する．

(3) ユーザはさらに $SK_1,\ldots,SK_n$ を生成してサーバに送信する．

(4) サーバは送られた $ID, PW, SK_1,\ldots,SK_n$ のうち，ID と PW を既存サーバ側で，ID と $SK_1,\ldots,SK_n$ をプロキシサーバ側で，それぞれ関連付けて管理する．

Authentication:

(1) ユーザは認証要求として ID をサーバに送信する．

(2) 既存サーバは C を生成し，この C をユーザに送信する．

(3) ユーザは PW とサーバから受信した C を用いて

$$R \leftarrow H_K(PW,C)$$

を計算する．この R を，n 個のパスワード以外の秘密情報から生成した共通鍵 $SK_1,\ldots,SK_n$ を用いて

$$R' \leftarrow ENC(SK_n,\ldots ENC(SK_2, ENC(SK_1,R))\ldots)$$

を生成する．この R' をサーバに送信する．

(4) プロキシサーバは $SK_1,\ldots,SK_n$ とユーザから受信した R' を用いて

$$R \leftarrow DEC(SK_1,\ldots DEC(SK_{n-1}, DEC(SK_n,R'))\ldots)$$

を計算し，この R を既存サーバに送信する．

(5) 既存サーバは PW と (2) で生成した C を用いて

$$H_K(PW,C)$$

を計算し，この計算結果がプロキシサーバから受信した R と一致するか確認する．一致したら認証成功，不一致なら認証失敗とし，その認証結果をユーザに通知する．

稲村 勝樹 （正会員）

1972 年生．1998 年東京工業大学工学部有機材料工学科卒業．2000 年北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．同年第二電電（株）（現 KDDI（株））入社，（株）京セラ DDI 未来通信研究所（退職時（株）KDDI 研究所，現（株）KDDI 総合研究所）配属．2014 年東京理科大学大学院工学研究科博士後期課程修了．同年東京電機大学理工学部情報システムデザイン学系助教，現在に至る．暗号・電子署名アルゴリズム，情報セキュリティの研究に従事．電子情報通信学会，映像情報メディア学会，INSTICC 各会員．博士（工学）．