2.2.2.2 チャレンジレスポンス相互認証における鍵の運用管理について

楫 勇一

1. はじめに

近年は，インターネットを介して多くのサービスが提供されるようになってきている．誰でも無料で利用できるサービスも多いが，利用者から料金を徴収する場合，あるいは，登録利用者だけにサービスを提供する場合等においては，サービスを利用しているユーザが誰であるのかを見極めることが重要となる．悪意を持ったユーザは，他の正当なユーザになりすまし，不正にサービスを利用しようとするおそれもあるため，正当なユーザと不正ユーザとを峻別する，いわゆるユーザ認証の仕組みを構築することが必要となる．ユーザ認証の概念自体はインターネットの普及以前から存在しており，これまでに多くのユーザ認証手法が提案されている．

ごく初期の段階では，簡単な暗証番号やパスワードを利用する認証方式が用いられた[1]．これは，ユーザとサービス提供者（以下ではサーバと称する）との間で事前に数ケタの暗証番号，あるいは数文字のパスワードを合意しておき，サービス利用者がサーバに対してパスワードを提示できるかどうかにより，ユーザが本人であるかどうかを判定する手法である．サーバはユーザごとにパスワードを記録しておく必要があるが，ユーザは自分の選択したパスワードを一個記憶するだけでよく，ユーザにかかる負担が小さくて済むという利点を有する．その一方，パスワードを平文のまま通信路に送信することの危険性は大きく，安全な専用線を利用する場合や，あるいは，下位レイヤにおける暗号機能で通信内容が保護される場合等をのぞき，本方式の安全性はきわめて低い．また，サーバにおいてパスワード情報をそのまま（平文のまま）保管しておくと，管理上の手落ちや事故等の不測事態からユーザのパスワードが露呈してしまう可能性も高くなってしまうため，パスワード保護のための対策が必要となる[1]．たとえば，パスワード情報を保護エリアに保存する，暗号化して鍵を別途管理する，サーバではパスワードのハッシュ値のみを保管する等，いくつかの方式が提案されているが，いずれの方式にも一長一短があり，安全性，経済性，汎用性等を高い次元で実現する唯一の方式は，著者らの知る限り提案されていない．

パスワード提示に代わるユーザ認証の原理として，チャレンジレスポンス方

式がしばしば用いられる．チャレンジレスポンス方式の実現には様々なバリエーションがあるが，基本的に，ユーザが秘密情報（慣例的に「パスワード」あるいは「鍵」と呼ばれることも多い）を安全に管理していることが前提条件となる．サーバは，正しい秘密情報を知らないものには行えない解けない問題（チャレンジ）をユーザに出題し，その計算結果（レスポンス）が正しいかどうかを調べることにより，サービス要求を行ってきたユーザが本人であるか否かを判定する．チャレンジを決定する主体はサーバ側であるため，通常は，毎回異なるチャレンジを出題することになる．解くべき「問題」の設定が適切であれば，たとえ攻撃者が過去の盗聴記録を保持していたとしても新しい問題を解くことは難しいため，ユーザなりすましに対する防御手段としては有効である．

上述のとおり，ユーザ認証法については古くから盛んに研究が行われているが，ここ数年は，ユーザ認証法の評価に関して，あらたに考慮すべき項目が増えつつある．たとえば，フィッシング詐欺に見られるように，サーバになり済ました攻撃者が，ユーザから個人情報を盗み取るタイプの不正行為が横行しつつある．この場合，サーバがユーザを識別すると同時に，ユーザもサーバを識別できる，いわゆる相互認証の重要性が格段に高くなっている．その一方，ユーザがモバイル機器等を持ち歩き，あるいは，移動先に設置された不特定の計算機を利用して，各種のサービスを利用するような形態も一般的になりつつある．認証に関する情報を特定の計算機に紐づけるような方式では，このようなユビキタス環境における認証実現には対応することが難しいと考えられる．使い捨てパスワード生成器のような特殊ハードウェアの利用，あるいは，人間には記憶困難な長大な秘密情報の保持を要求するような方式も，上記と同じ意味で，ユーザの利便性を損なう面がある．

本稿では，上述したような要求事項の変化（追加）を考慮に入れ，様々なタイプのチャレンジレスポンス認証法の特性について議論を行う．ユーザにどのような秘密情報を持たせるか，サーバではどのような形でユーザの認証情報を記録するか，チャレンジをどのように構成するか等，チャレンジレスポンス認証には多くの自由度がる．異なる要素技術を利用すると，結果として得られる認証法の特性も変化するため，実際に使用される環境を十分に想定して各種の選択を行うことが望ましい．本稿では，暗号を利用するチャレンジレスポンス法のうち典型的ないくつかについて，それぞれの得失の評価を行う．

2. 評価すべき項目について

本節では，チャレンジレスポンス認証を考えるうえで考慮しないといけない点について整理する．通信路の盗聴等，基本的な脅威に対する安全性が確保されていることは，チャレンジレスポンス方式における必須事項であると考えられる．そのような必須事項を達成するための条件や効率等，やや運用面に重点を置いて議論を行うことが本稿の目的である．

2.1. ユーザが所持する情報に関する条件

チャレンジレスポンス法では，ユーザがある種の秘密情報を管理する必要がある．この秘密情報が不正者に漏えいすると，いわゆる「ユーザなりすまし」に容易に成功することになる．安全性を考えると，秘密情報はできるだけ大きく，かつランダムであることが望ましい．その一方，ユーザによる記憶が難しいくらい大きな，あるいはランダムな，秘密情報の使用を義務付けることは，ユーザにとって大きな負担となる．この場合，たとえば，ユーザが秘密情報をメモ等に残して持ち歩くという安全上の問題，あるいは，秘密情報を事前に格納したノートパソコンがないと利用できないといった利便上の問題が発生する可能性も高く，その場合，システム全体としてのセキュリティやユーザの利便性に大きな悪影響を及ぼすことになる．ユビキタス環境での利用を考えると，ユーザには，自ら選んだ短い情報だけを所持させることが望ましい．

１． ユーザが所有しなければならない秘密情報の量は小さいか

２． ユーザが所有しなければならない秘密情報は,ユーザ自身が自由に選べるか

認証方式によっては，秘密情報の他，公開情報や公開しても支障のない情報をユーザが保持しなければならない場合もある．公開情報の管理・保持コストは，秘密情報のそれに比べると小さいと考えられるが，それでも公開情報自体の量は小さいことが望ましい．

３． ユーザが所有しなければならない公開情報の量は大きいか

公開情報をネット経由で参照する等の手段も考えられるが，その場合，ユー

ザは URL のような参照先情報を保持する必要がある．すべてのユーザ，すべてのサービスに対して働くような唯一の参照先があれば，たとえば URL を短くしたり，あるいは，ブラウザ等の汎用プログラムにあらかじめ参照先を埋め込んでおく等の手段を取ることもできる．その意味で，公開情報がユーザやサービスに紐付けられているか否かという視点も，運用上には考慮すべきであると考えられる．

４． ユーザが所有しなければならない公開情報は,他ユーザと同一のものであるか

もちろん，ユーザが公開情報を保有しなくても良い場合は，上記の評価項目については考慮する必要が無い．

ユーザに少量の情報しか持たせない場合，いわゆる総当り攻撃や辞書攻撃の脅威を無視することができなくなる．原理的には，ユーザになりすました不正者が辞書攻撃により（あるいは総当り的に）秘密情報を推測し，サーバに対して認証要求を繰り返し行うことで，正しい秘密情報を割り出すことは常に可能である．しかし，そのような攻撃は容易に検知できるため，実質上はそれほど脅威には成りえない．むしろ通信路の盗聴に成功した攻撃者が，ユーザやサーバに知られることなく，通信内容に対して辞書攻撃的に解析行為を行うことのほうが，現実問題として深刻な脅威に成り得る．プロトコル等で開示する情報を適当に制御し，通信内容を用いた辞書攻撃を防ぐ方策が立てられているかどうかも，十分注意すべき点である．

５． 盗聴データを利用した辞書攻撃を行われても，安全性が確保できるか

2.2. サーバの信頼性に関する条件

多くの認証プロトコルでは，サービスを提供するサーバは信頼できる（能動的，受動的を問わず不正行為を行わない）ことを前提とすることが多い．この仮定自体は妥当であると考えられるが，事故等により，サーバから情報が流出する可能性が皆無ではない旨も承知しておく必要がある．現実に，クレジットカード会社や銀行等，情報管理に十分な対策を取っていると思われる企業においても情報流出事故が発生していることを考えると,サーバからの情報流出は,

それなりに想定しておくべき事態であるともいえる.

もっとも，サーバにおける情報管理の実態は個別に異なるため，どのような情報が流出するか，それがどのように流出するか等を画一的に議論することは困難である.たとえば,サーバ全体の機密情報である暗号鍵そのものの流出は,あまり現実的には発生しないと考えられる．サーバ固有の暗号鍵が流出するような事態が発生すると，サーバと完全に同一のクローンサーバを構築することも可能となるが，この場合，偽装サーバ（クローンサーバ）と真正サーバを外部から区別することは原理的に不可能であり，どのような手段を講じたとしても,安全性を確保することはできなくなってしまう.その一方,顧客リスト等,ユーザ管理情報の流出事故は多発しており，これについては十分に留意すべきである．通常，サーバの内部ではユーザ管理情報を暗号化する等の安全対策が取られることが多い．その場合であっても，管理作業の各段階で，復号されたデータが一時的に得られる場合もある．そのようなデータを目にしたオペレータが不正に（手順上許されていないのに）データを保存し，外部に持ち出すような行為から，多くの情報流出事故が発生している．すなわち，サーバ内部にある情報を暗号化しただけでは情報流出への対策としては不十分といわざるを得ない.

上の議論より,「サーバで厳重保管された鍵自体が流出することはないが,その鍵で暗号化されたデータは，復号された形で流出する可能性がある」ことを想定する必要がある.

サーバから流出した情報が不正者の手に渡った場合，その情報は，様々な攻撃に利用される恐れがある．最も直接的な脅威は，流出情報にユーザの秘密情報そのものが含まれる場合，ユーザの秘密情報が容易に推測できるような情報が含まれる場合，そのユーザになりすますことができるような情報が流出情報から計算できる場合であろう.したがって,認証方式を評価するにあたっては,以下の項目についても考慮すべきである.

６．　サーバからデータが流出した場合でも,ユーザの秘密や権利が守られるか

一人のユーザが複数のサービスで同一のパスワードを利用している可能性があることを想定すると，ある種の識別不可能性に類する条件を考えることにも意味があると考えられる.

7． 流出データに含まれる２ユーザの情報を比較したとき,それらユーザが同一のパスワードを使っていることを識別できるか

3. 典型的なチャレンジレスポンス認証法の実現について

はじめに述べたとおり，チャレンジレスポンス認証では個々のユーザが秘密情報を安全に管理することを前提とし，ユーザは，秘密情報なしでは行い得ない計算をして見せることにより,自分が間違いなく本人であることを主張する．この枠組みを実現するため，チャレンジレスポンス認証の実現においては，暗号を利用することが一般的である．すなわち，ユーザは暗号文を復号するための鍵を保有しておき，サーバはランダムに選ばれたデータの暗号文をユーザに対して送付する．この暗号文が正しく復号できればユーザは真正であるとみなし，正しく復号できない場合，そのユーザは本人であると認めない．どのような暗号を利用するか，鍵の公開・秘匿の制御をどのように行うかにより，特性の異なるいくつかのチャレンジレスポンス方式が得られる．本研究では，既に良く知られたタイプのチャレンジレスポンス方式に加え，公開鍵暗号をややイレギュラーな形で利用するようなチャレンジレスポンス方式についても検討し,次節にて各方式の特性について議論する．まず本節で典型的なチャレンジレスポンス法について検討を行い，次節にて，変則的な鍵管理を前提とするチャレンジレスポンス法について議論を行う．以下ではチャレンジレスポンス認証をCR 認証と略記する．

3.1. 共通鍵暗号ベースの CR 認証

もっともシンプルな CR 認証は,共通鍵暗号を利用するタイプのものである．すなわち，ユーザとサーバとの間で事前に共通鍵暗号の鍵を共有しておき，チャレンジ生成(暗号文生成)，レスポンス計算(暗号文復号)の両操作において，この鍵を利用する．この方式の CR 認証は早い時期から実用化が進んでおり，たとえば Kerberos[4] におけるユーザ認証などで本方式を選択することが可能となっている．

多くの場合，共通鍵暗号の鍵は決められた鍵空間内で自由に選ぶことができるため，ユーザは自分にとって覚えやすい任意の鍵を選択する（あるいは，自分の覚えやすい情報からハッシュ関数等により作り出す)ことができる.また，一個の鍵が暗号化と復号の双方に利用できることから，ユーザからサーバに対

してチャレンジを発行することも容易となっている．一個の鍵だけで相互認証が実現できるというメリットは，運用上大きいといえる．

以上の理由により，本方式は，前節の評価項目 1 から 4 については優れた特性を有するといえる．一方，暗号化等の鍵は，ユーザが記憶する情報だけを利用して選択あるいは生成されることになるため，それほど多くの自由度は持ち得ないと考えられる．したがって評価項目 5 については問題があると考えるのが妥当と思われる．また，サーバがユーザと同一の鍵を保持する必要があるため，評価項目 6 および 7 について，非常に大きなリスクを有するといえる．

3.2. 公開鍵暗号ベースの CR 認証：通常の鍵運用

共通鍵暗号ではなく，RSA[5]に代表される公開鍵暗号を利用する形で CR 認証を実現することもできる[6]. 通常, ユーザが秘密の復号鍵を所持しておき，サーバには，公開可能な暗号鍵を事前配布しておくような運用が取られる．共通鍵暗号利用の場合と異なり，本方式では，基本的に片側方向の認証しか実現できない点に注意が必要である．双方向認証を実現するためには，たとえば SSH の実装等で見られるように，ユーザがサーバの公開鍵（暗号鍵）を保持し，立場を入れ替えた形で CR 認証を行う必要がある．一般に，公開鍵暗号の鍵は共通鍵暗号に比べて長くなり，また，鍵の選択においては数学的な制限条件が課せられることが多く，任意の系列を鍵にすることは難しい．さらに，公開可能な情報もかなりの情報量になるため，これら一連の鍵情報をユーザ自身が記憶することは困難である．したがって，公開鍵ベースの CR 認証を利用するためには，ユーザは，補助的な記憶装置や端末等を携行する必要がある．それら端末が手元にない状態では，CR 認証を介したサービス享受ができないことになる．

以上の理由により，本方式は，前節の評価項目 1 から 3 について，問題を有するといえる．評価項目 4 について，同一サーバを利用するユーザは全員同じ公開情報を有するが，サーバが異なると，公開情報も異なることとなる．メジャーなサービスについてはサーバの公開鍵等をブラウザ側に持たせ，一般ユーザの負担を軽減する等の実装も考えられるが，すべてのサービスについて網羅的に同種の対応を行うことは困難であると考えられる．本方式において鍵組をランダムに生成した場合，可能な鍵空間は非常に大きくなるため，評価項目 5 については問題にならないと考えられる．また，評価項目 6 について，

公開鍵暗号の特性により，サーバからのデータ流出時でも，ユーザの秘密鍵が露呈することはないといえる．一方，一つの復号鍵に対して複数の暗号化鍵が定義できるような暗号を利用しない限り，評価項目 7 の安全性は達成することができない.

4. 変則的なチャレンジレスポンス認証法の実現について

前節では, 公開鍵暗号を利用した CR 認証の典型的な実現法について検討した．公開鍵暗号を利用する方式の最大の問題点は，各ユーザが，自らの選択の余地のない大きな情報を携行する必要があるという点にある．

一方，公開鍵暗号の中には，鍵を比較的自由に選択できる方式もいくつか知られている．たとえば ElGamal 暗号[3]では，素数 p と GF(p) の生成元 g を選択した後，秘密情報（秘密鍵）x を任意に選んで，公開鍵 $g^x \bmod p$ を決定する手順を取る．この場合, たとえば「ユーザパスワードのハッシュ値」を x とする等の対応により，ユーザは特別な補助記憶を使わずに，自らの記憶のみから秘密鍵（復号鍵）を構成することができることになる．

あるいは ID ベース暗号[2]の場合, 任意の識別子を暗号鍵とすることができるため,「ユーザパスワードのハッシュ値」を公開鍵として利用できることになる．もっと一般的に,「ユーザパスワードのハッシュ値」を疑似乱数生成のシード（確率的チューリング機械のランダムテープの中身）として鍵生成アルゴリズムを実行し，認証時に，ユーザが on-the-fly で暗号化鍵や復号鍵を生成するような仕組みを考えることも可能である．

CR 認証にこのような仕組み（あるいは，ElGamal 暗号[3]や ID ベース暗号のような特別な性質を持った暗号）を組み合わせることにより，SSH 等で利用されている典型的な CR 認証の特性を改善できるのではないか, というのが本節で紹介する CR 認証法について検討する動機である．

また，上記とはやや独立した視点として，公開される鍵の管理方法について検討の余地があるのではないかと考えられる．公開鍵暗号を利用してなんらかのセキュリティ機構を構築する場合，その公開鍵（暗号化鍵）が露呈しても安全性が損なわれないよう，各種の設計がなされることが一般的である．実際の運用においては，公開鍵であっても必要以上に露出させない運用が「推奨」されることも多いが，これを,「公開鍵も完全に秘匿するような運用を行う」と割り切った場合, 従来の CR 認証では得られなかった特性が得られる可能性もある．本節では，この考えに基づく CR 認証方式についても検討する．

4.1. 鍵を on-the-fly で生成する CR 認証法

ここでは, 3.2 節で述べた公開鍵暗号の鍵組を二つ利用する CR 認証において, ユーザが必要とする鍵を on-the-fly で生成するような仕組みについて検討する．すなわち，ユーザは自ら選んだ短いパスワードのみを記憶し，認証時には，記憶したパスワードから自らの秘密鍵とサーバの公開鍵を生成する．ただし，3.2 節の仕組みを忠実になぞるのではなく，ユーザが公開情報を携行する負担を免れるための措置を盛り込む．具体的には，サーバが使用する鍵組をユーザ毎に異なるものとし，サーバ公開鍵が，ユーザのパスワード情報に従属して定まるようにする. 以下では, 適当なハッシュ関数 h および共通鍵暗号 E', 暗号化操作，復号操作がそれぞれ E( ), D( ) と与えられる公開鍵暗号系，与えられた乱数初期値 r を用いて二組の鍵組を生成するような確率的鍵生成アルゴリズム K(r) の存在を仮定し，具体的なプロトコルを示す.

ユーザ登録

ユーザは自らのパスワード p を決定し，$(pk_1, sk_1), (pk_2, sk_2) \leftarrow K(h(p))$ として二つの鍵組 $(pk_1, sk_1), (pk_2, sk_2)$ を生成する．安全なオフライン通信路等を用いて，生成した鍵のうち pk_1 と sk_2 をサーバに登録し，サーバは，二つの鍵をユーザに紐付けて厳重管理する.

認証段階

サービス要求を受けたサーバはそのユーザの鍵情報を取り出し，$m_1 = E_{pk1}(n_1)$ をユーザに対して送信する．ここで n_1 はランダムに選択されたデータ（いわゆる nonce）である. ユーザは自らのパスワードを用いて$(pk_1, sk_1), (pk_2, sk_2)$ を計算し，$n_1 \leftarrow D_{sk1}(m_1)$ として n_1 を得た上で，$m_2 = E_{pk2}(n_2, E'_{n1||n2}(ACK))$ をサーバに返送する．ここで n_2 はユーザが選択する nonce とし，ACK は一連の計算成功を意味する固定メッセージとする．サーバは sk_2 を用いて $(x_1, x_2) \leftarrow D_{sk2}(m_2)$ を計算し，x_2 と $E'_{n1||x1}(ACK)$ が一致するかどうかを検査する．両者が一致すればユーザが真正と認め，一致しなければ，当該要求を拒絶する.

本方式の特性

本方式では，ユーザが必要とする情報は全て p から計算されることになる

ため，評価項目 1 および 2 については，優れた特性を有することになる．3.2 節の方式における公開情報に相当するものも p から生成されるため，ユーザは p 以外の情報を別途携行する必要もなく，評価項目 3 （および 4）についても良好である．一方，本方式では，辞書攻撃等によって得られたユーザパスワードの推測値が正しいか否か，受動的攻撃者に識別される問題点がある．たとえば，ユーザパスワードの推測値として p' がある場合，p' に対応する鍵組 (pk'_1, sk'_1), (pk'_2, sk'_2) を計算することができる．これを用いて $n'_1 \leftarrow D_{sk'1}(m_1)$, $(x'_1, x'_2) \leftarrow D_{sk'2}(m_2)$ を計算し，x'_2 と $E'_{n'1||x'1}(ACK)$ が一致するかどうかを確かめることにより，p'=p かどうかを判定することができてしまう．したがって本方式は，評価項目 5 については脆弱であるということになる．サーバからのデータ流出について，たとえ pk_1, sk_2 が他者に知られたとしても，ここから直ちに p が露呈することはない．また，pk_1, sk_2 を用いても当該ユーザになりすますことはできないため，評価項目 6 については好ましいといえる．その一方，鍵情報が p から一意に定まってしまうため，評価項目 7 については問題が残る．

4.2. 公開鍵を秘密管理する方式

前節では鍵組を二つ利用する方式を検討したが，サーバ側において公開鍵（暗号化鍵）を完全に秘匿できる場合は，一つの鍵組だけで相互認証を実現することもできる．この前提条件では，あるデータを正しく暗号化できるのは正しいサーバ（およびユーザ本人）だけになるため，「暗号化できるか否か」を検査することで，サーバの真正性を判定することができるためである．

本節では，ユーザが秘密鍵（復号鍵）を，サーバが公開鍵（暗号化鍵）を保有し，それ以外の鍵情報を利用しない認証法を考える．前節と同じく，ユーザの記憶情報に従属する形で秘密鍵を計算できるようにしたい．ここでは，前節と同じく，確率的鍵生成アルゴリズムの乱数初期値を与える方式で手順を説明するが，たとえばこの部分に ElGamal 暗号[3]のような暗号方式を使うことも可能である．ここでは，K( ) は一組の鍵組のみを生成するアルゴリズムとする．

ユーザ登録

ユーザは自らのパスワード p を決定し，(pk, sk) ← K(h(p)) として鍵組 (pk, sk) を生成する．安全なオフライン通信路等を用いて pk をサーバに登録し，サーバは，pk をユーザに紐付けて厳重管理する．

認証段階

ユーザはサーバに対し，（自らの識別子と） nonce n_1 を送信する．サービス要求を受けたサーバは，そのユーザの鍵情報を取り出し，$m_1 = E_{pk}(n_1, n_2)$ をユーザに対して送信する．ここで n_2 はサーバの選択した nonce である．ユーザは自らのパスワードを用いて(pk, sk) を計算し，$(x_1, x_2) \leftarrow D_{sk}(m_1)$ として $x_1=n_1$ となっているか確認する．両者が一致した場合，ユーザは $m_2=E'_{x2}$(ACK) をサーバに返信し，サーバは，m_2 と E'_{n2}(ACK) が一致するかどうか検査する．両者が一致すれば，ユーザは正しく n_2 を復号できたと考えられるため，このユーザを真正であると認める．

本方式の特性

前節の方式と同じく，ユーザは p だけを記憶すれば良いため，評価項目 1 から 4 については優れた特性を有するといえる．また，前節と同様の理由により，正しいパスワードと間違ったパスワードが識別できるため，評価項目 5 については問題がある．評価項目 6 および 7 についても，前節の方式と同じ性質を有する．本稿の評価項目には上げていないが，本方式は，前節方式よりも少ない計算量的で同等の利便性を得ることができる．

4.3. ベース暗号の利用

サーバにおいてユーザの個別情報が秘密管理できる前提では，前節における鍵の役割を交換し，ユーザ自身は暗号化鍵を保有し，サーバ側に復号鍵を持たせることも可能である．前節同様，確率的鍵生成アルゴリズムを利用して CR 認証を実現する方式を考えることも可能であるが，そのようにして得られた方式は前節の方式とほぼ同様の特性を有するため，とくに興味を引く対象とはいえない．

その一方，ユーザの選択した情報に従属する形で暗号鍵を計算する手法として，ID ベース暗号[2]を利用するような選択も考えられる．ID ベース暗号における鍵生成はユーザ本人だけで完結せず，いわゆる鍵管理サーバの協力が必要になる．このような枠組みの違いは，得られる CR 認証の特性にも影響を与える．本節では，サーバにおいてユーザの個別情報が秘密管理できる前提のもと，CR 認証の実現に ID ベース暗号を利用する方式について検討する．ここでは，オフライン的に鍵を発行する鍵管理サーバの存在を仮定し，暗号化鍵

（ID）u に対して鍵管理サーバが発行する復号鍵を sk_u と書くことにする．

ユーザ登録

ユーザ u は自らのパスワード p を決定し，u||p に対する復号鍵 $sk_{u||p}$ を鍵管理サーバに計算してもらう．$sk_{u||p}$ は安全なオフライン通信路を介してサービス提供サーバ（以下では，こちらを単にサーバと書く）に届けられ，サーバは $sk_{u||p}$ をユーザに紐付けて厳重管理する．

認証段階

ユーザはサーバに対し，$m_1 \leftarrow E_{u||p}(n_1,n_2)$ を送信する．サーバは $(x_1,x_2) \leftarrow D_{sk\_\{u||p\}}(m_1)$ を求め，$m_2=E'_{x1}(x_2, n_3)$ をユーザに返送する．ユーザは $(y_1,y_2) \leftarrow D'_{n1}(m_2)$ を計算し，$y_2=n_2$ ならばサーバが真正であると認め，$m_3=E'_{y2}(ACK)$ をサーバに送信する．サーバは $z \leftarrow D'_{n3}(m_3)$ が ACK と等しくなるかどうか検査し，等しければユーザを真正と認める．

本方式の特性

ユーザは，自分の識別子 u とパスワード p だけを記憶すれば良いため，評価項目 1 から 4 については優れた特性を有する．評価項目 5 について，本方式の特徴は，p に関する知識だけでは復号鍵 sk_p の計算ができない点にある．たとえ攻撃者が p の予測値 p' を持っていたとしても p' に対応する復号鍵は（鍵管理サーバに依頼して発行してもらわない限り）計算できず，受動的攻撃だけでは p と p' の識別を行うことができない．したがって本方式は，評価項目 5 について好ましい特性を有するといえる．評価項目 6 についても，サーバから流出した情報 $sk_{u||p}$ から p を特定することは困難なため，この場合でも安全性は満たされる．また，二人のユーザ u_1, u_2 が偶然同じパスワード p を使用していたとしても，ユーザの識別子は異なるため，この二人に紐付けられるサーバ上の情報 $sk_{u1||p}$, $sk_{u2||p}$ は異なるものとなる．したがって本方式は，評価項目 7 についても好ましい性質を有する．

5. まとめ

本研究では，様々な CR 認証方式の特徴について議論した．技術や環境の変化にともない，CR 認証の設計において，従来はあまり重要視されていなかった評価項目についても今後は十分に配慮する必要がある．また，本稿 4 節で

議論したように,「公開鍵暗号の鍵を公開しない運用」を行った場合の可能性等についても，従来はあまり顧みられていなかったように思われる．さらに，4.3節で紹介した方式のように，ユーザ，サーバの他に信頼できる第三者の存在を仮定し，その第三者に信頼のポイントを依拠するような枠組みについても，今後ますます検討の余地があると考えられる．

参考文献

[1] S.T.Bellovin and M.Meritt, Encrypted key exchange: Password based protocols secure against dictionary attacks and password file compromise,1st ACM Conference on Computer and Communications Security, pp.244-250, 1993.

[2] D.Boneh, Identity-Based Encryption from the Weil Pairing, CRYPTO 01, pp.213-229, 2001.

[3] E.ElGmal, A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, CRYPTO 84, pp.10-18, 1985.

[4] J.T.Kohl, B.C.Neuman and T.Y.Ts'o, The Evolution of the Kerberos Authentication Service, EurOpen Conference Proceedings, pp.295-313, 1991.

[5] R.L.Rivest, A.Shamir and L.M. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, **21**, 2, pp.120-126, 1978

[6] T.Ylonen and C.Lonvick, The Secure Shell (SSH) Authentication Protocol, RFC 4252, 2006.