社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2003－ＣＳＥＣ－22　(14)
2003／7／17

Rollback 攻撃に対する SSH の脆弱性

鬼頭　利之†　　齋藤　孝道††

† 株式会社東芝　〒 212-8582 神奈川県川崎市幸区小向東芝町 1
†† 東京工科大学　〒 192-0982 東京都八王子市片倉町 1404-1

E-mail: †Toshiyuki.Kito@jcom.home.ne.jp, ††saito@cc.teu.ac.jp

あらまし　「安全でない」ネットワークを通して,「安全な」通信を実現することができるツールとして，SSH (Secure SHell) が広く利用されている．SSH では過去のバージョンのシステムとの融和のため，様々な暗号化アルゴリズムとユーザ認証方式の利用が提供されている．また，互換性のない SSH1 と SSH2 の両者をサポートするように，両立して運用されることも多い．さて，このような SSH の利用においては，本論文において新たに示される Rollback 攻撃を用いて，SSH2 の利用者であっても，SSH1 の利用を強制させ，さらには，最も脆弱な暗号化アルゴリズム，かつ，パスワードを用いたユーザ認証方式を強制することができることが分かった．また，より重大な欠陥としては，攻撃者が，SSH サーバのミラーサーバであると偽って，接続させ，正規の SSH サーバを騙す MITM (Man in the Middle) 攻撃において，Rollback 攻撃を行うと，SSH1，SSH2 共に，公開鍵を利用してユーザ認証したい SSH クライアントであっても，パスワードを用いたユーザ認証方式を強制することができ，攻撃者は，パスワードを不正に奪取することができることを新たに発見した．本論文では，以上，2 種類，5 つの Rollback 攻撃を示す．

キーワード　Security Protocol, Authentication Protocol, Rollback Attack, SSH, Vulnerability

The Secure SHell's Vulnerability against Rollback Attacks

Toshiyuki KITO† and Takamichi SAITO††

† TOSHIBA Corporation,　1, Komukai Toshiba-cho, Saiwai-ku, Kawasaki, Kanagawa, 212-8582, Japan
†† Tokyo University of Technology,　Katakura-machi 1404-1, Hachioji, Tokyo, 192-0982, Japan

E-mail: †Toshiyuki.Kito@jcom.home.ne.jp, ††saito@cc.teu.ac.jp

Abstract　SSH (Secure SHell) is widely used as a software which can realize secure communication over insecure networks. SSH provides the usage of various encryption algorithms and the ways of user authentications for the harmony with the system of the past version. Also SSH1 and SSH2 are often used as one system for compatibility. In the case, we show that even if a user utilizes SSH2, she/he is forced to utilize the SSH1, the most weekest encryption algorithm and the password authentication. Moreover, as a more serious defect, even if SSH client tries to use public key for authentication in SSH1 and SSH2, the intruder can force to use password for authentication, when the intruder does Rollback attack in the case that he masquerades as the mirror server of the authorized one, she/he is connected by SSH client and does MITM (man in the middle) attack deceiving the authorized one. And it is newly found that the intruder can illegally deprive of user's password. In this paper, two kinds, five Rollback attacks are shown.

Key words　Security Protocol, Authentication Protocol, Rollback Attack, SSH, Vulnerability

1. はじめに

SSH (Secure SHell) [1] ～ [6] は，盗聴や改ざん，なりすましなどの脅威が存在する「安全でない」ネットワークを介して，遠隔端末の機能だけでなく，遠隔の計算機上でコマンドを実行したり，遠隔の計算機とのファイルの転送をすることを，「安全」に実現するソフトウェアとして広く利用されている．SSH は，SSH サーバプログラムと SSH クライアントプログラムから構成され，プロトコルバージョンの違いにより，SSH version.1 (以降，SSH1 と呼ぶ) と SSH version.2 (以降，SSH2 と呼ぶ) の互換性のない 2 つのバージョンがある．また，SSH1 は文献 [2] において，SSH2 は文献 [3] ～ [6] において，それの仕様がそれぞれ与えられている．SSH クライアントと SSH サーバは，**プロトコルバージョンの交換**，**セッション鍵の交換**，**ユーザ認証**といった 3 つの手続きをこの順序で行い，通信路を確立する．この際，SSH では過去のバージョンのシステムとの融和と，より強固な暗号化アルゴリズムの利用のため，3DES，RC4，IDEA など(注 1)様々な暗号化アルゴリズムの利用とその選択が提供されている．また，ユーザ認証方式としては，システムの管理者，利用者共に，より簡便なパスワードを用いたユーザ認証と，より安全とされている公開鍵を用いたユーザ認証がある(注 2)．さらに，OpenSSH [7] では，SSH1 と SSH2 の両者が共存し，運用できる．

さて，SSH では，上述の 3 つの手続きにおけるネゴシエーションを通して，最も強力な暗号化アルゴリズムを用い，より安全とされる公開鍵を用いたユーザ認証の利用を SSH2 において合意し，その利用により，SSH2 によって提供される最も安全な通信を利用することができる．しかしながら，SSH サーバ，もしくは，SSH クライアントのいずれかが，他方が利用したい「より安全な方式」をサポートしていない場合，やむを得ず，「次善の方式」の利用をすることになる．もちろん，運用上のポリシーで，そのような「妥協」を許さない運用形態もあり得るが，利便性の問題から，最も脆弱な暗号化アルゴリズムとパスワードを用いたユーザ認証を許す運用もないとは言えない．このような運用状況において，Rollback 攻撃を用いて，SSH2 の利用であっても，SSH1 の利用を強制させ，さらには，相互に利用できる最も脆弱な暗号化アルゴリズム，かつ，パスワードを用いたユーザ認証方式を強制することができることが分かった．この Rollback 攻撃は，攻撃者が正規の SSH サーバと SSH クライアントのやり取りに入り込みセッションをハイジャックすることで，より脆弱な方式を強制させるのだが，SSL (Secure Socket Layer) [8] では，少なくとも両者がそれを合意していることを通報認証を用いて，合意しているため，両者の意図を反映するという意味で，防ぐことができる．しかしながら，SSH は，本論文で新たに示されたこの Rollback 攻撃を防ぐことができないので，Rollback 攻撃に対して脆弱であると言える．繰り返すことになるが，本論文で示すこの Rollback 攻撃というのは，利便性を重視した脆弱な運用ポリシーに起因する問題ではなく，クライアントとサーバの両者が望まない「より脆弱な方式」を強制されてしまうというプロトコルの不備に起因することに読者はくれぐれも注意されたい．また，この Rollback 攻撃を，後述の Rollback 攻撃と区別するために，Active Rollback 攻撃と呼ぶ．

(注 1) :SSH1 では，脆弱性であるとされている DES もサポートされている．

(注 2) :その他に，rhosts 認証と rhosts 公開鍵認証がある．

「プロトコルバージョンの交換」の手続きの後，SSH クライアントは SSH サーバの認証とセッション鍵の交換を行う「セッション鍵交換」の手続きを行う．その後，セッション鍵で暗号化された通信路を通して，SSH サーバは SSH クライアントを操作するユーザの認証を行う．この一連の手続きのため，ユーザが正規のミラーサーバであると偽っている SSH サーバの振りをした攻撃者に接続し，ユーザが当該セッションを続けたい場合，SSH1 と SSH2 共に，「サーバを騙す MITM (Man In The Middle) 攻撃」を防ぐことができない [9]．ただし，この攻撃は，ユーザ認証において「パスワードを用いた認証」を利用しなければ成立しない．しかしながら，攻撃者は，ユーザに「公開鍵によるユーザ認証」を利用したいユーザでさえ「パスワードを用いたユーザ認証」を強制させることができることを今回発見した．また，この攻撃が OpenSSH-2.9 において実現可能であることを確認した．この Rollback 攻撃は，攻撃者が能動的にセッションをハイジャックし，通報を改ざんするわけではなく，「サーバを騙す MITM (Man In The Middle) 攻撃」自体がどちらかと言うと，待ち受ける攻撃であり，その上での Rollback 攻撃であるので，Passive Rollback 攻撃と呼び，前者の Active Rollback 攻撃と区別する．

本論文では，以上の 2 種類，5 つの Rollback 攻撃に関する SSH の脆弱性を具体的な通報のやり取りを示すことにより詳細に説明する．以降，本論文の構成は，2 節において本論文で用いる表記，SSH で用いる鍵の説明をし，3 節において本論文で扱う SSH のプロトコルの説明をする．その後，4 節において Active Rollback 攻撃を示し，5 節において Passive Rollback 攻撃を示す．最後に，本論文をまとめる．

2. 準　　備

本論文を通して用いる表記，用語，SSH で用いる鍵について示す．その他の表記，用語，安全性の基礎概念は，文献 [10], [11] に従う．

2.1 表記と用語

主体に関する表記と用語

SSH は，SSH サーバと SSH クライアントから構成される．SSH サーバは，SSH のサーバプログラムを起動している主体である．また，SSH クライアントは，SSH のクライアントプログラムを起動している主体である．そして，SSH サーバを S，SSH クライアントを C として表す．さらに，悪意のある第三者，つまり，攻撃者を $\mathcal{I}$ によって表す．最後に，$\mathcal{I}(S)$，$\mathcal{I}(C)$ は，S に成りすました攻撃者 $\mathcal{I}$，C に成りすました攻撃者 $\mathcal{I}$ をそれぞれ表す．

通報に関する表記

Msg は任意の通報を示す．ただし，返信の際の通報は，Ack とする．

C が Msg を S に通報する場合，以下のように示す：

$$C \rightarrow S \ : \ Msg$$

C と S が Msg を互いに通報する場合，以下のように示す：

$$C \leftrightarrow S \ : \ X$$

次に，他の主体に成りすます攻撃者についての表記を示す．例えば，主体 S に成りすます攻撃者 $\mathcal{I}$ が，主体 C に対して通報 Msg を通報する場合は，以下のように示す：

$$\mathcal{I}(S) \rightarrow C \ : \ X$$

また，本論文において以下の 2 つの通報は，意味合いがそれぞれ異なる：

$$C \rightarrow \mathcal{I}(S) \quad : \ Msg$$
$$C \rightarrow \mathcal{I} \quad : \ Msg$$

「$C \rightarrow \mathcal{I}(S)$: Msg」は，C が通信相手を S であると判断しているが，実際には $\mathcal{I}$ に Msg を送信していることを示している．「$C \rightarrow \mathcal{I}$: Msg」は，C がサーバの振りをする攻撃者を正規のサーバであると判断して，つまり，$\mathcal{I}$ を攻撃者とは思わずに，$\mathcal{I}$ に Msg を送信していることを示している．また，慣習で，同じ識別子を用いた通報で，異なる通報を表すこともあるが，議論をする上では問題ないため，本論文では，その慣習に従う．

暗号化と鍵に関する表記

$\{X\}_Y$ は，鍵 Y を用いて通報 X を暗号化した暗号文を示す．また，Y が公開鍵の場合，Y に対する秘密鍵を Y^{-1} で示す．そして，Y^{-1} によって電子署名された通報 X は，$\{X\}_{Y^{-1}}$ で示す．さらに，鍵 Z と鍵 Y を用いて通報 X が二重に暗号化された暗号文を $\{\{X\}_Y\}_Z$ で示す．次に KS_{CS} は，S と C のセッション鍵を示す．そして，P_S, H_S, P_C は，S のホスト鍵，S のサーバ鍵，ユーザの公開鍵をそれぞれ示す．それぞれの鍵については，後述する．

プロトコルバージョンに関する表記と用語

V_S は SSH サーバ S のバージョンであり，V_C は SSH クライアント C のバージョンを示す．さらに，$version1.0$，$version2.0$，$version1.5$ と表記した場合は，SSH1 のみをサポートしていることを示すバージョン情報，SSH2 のみをサポートしていることを示すバージョン情報，SSH1 と SSH2 をサポートしていることを示すバージョン情報をそれぞれ表す．

SSH2 での鍵交換で用いる表記

p は大きく安全とされる素数であり，g は GF(p) の部分群の生成元であり，q は部分群の位相である．次に，I_S は，S のセッション鍵の交換の通報を表し，I_C は，C のセッション鍵の交換の通報を表す．また，y, x は，S, C によって生成される乱数 $(1 < x, y < q)$ を示す．

K_S は $g^y \bmod p$ であり，K_C は $g^x \bmod p$ である．さらに，セッション鍵を生成するための情報である K_{CS} は，$g^{xy} \bmod p$ を示す．

SSH2 のユーザ認証で用いる表記

$password\_auth$ は，パスワードを用いたユーザ認証の手続きであることと，ユーザのパスワードを示す．$public\_auth$ は，公開鍵対によるユーザ認証の手続きであることと，デジタル署名を示す．

その他の表記

RN は，クッキーとしての乱数を示す．SA は，SSH1 では，暗号化アルゴリズム，ユーザ認証方式のリストを示し，SSH2 では，暗号化アルゴリズム，圧縮アルゴリズム等のリストを示す (以降，セキュリティアソシエーションと呼ぶ)．$username$ は，ユーザのログイン名を示す．ただし，$username_C$ と表記した場合は，それらは C に関するものである．$H(Msg_1)$ は，Msg_1 から生成されるハッシュ値を示す．同様に，$H(Msg_1, Msg_2)$ は，Msg_1 と Msg_2 を結合したハッシュ値を示す．

2.2 SSH で用いる鍵について

(1) ホスト鍵

SSH クライアントが，SSH サーバを識別する際に用いる公開鍵．SSH のサーバプログラムをインストールする際に SSH サーバが作成する．この鍵の有効期限はないが，この鍵に対応する秘密鍵が漏洩した時は更新が必要である．

(2) サーバ鍵

データをより解読されにくくするための公開鍵．この鍵は，SSH1 だけに使用され，有効期限は標準で，1 時間である．

(3) セッション鍵

認証後のデータ通信を暗号化するための共通鍵．この鍵を意図せぬ第三者と共有した場合，通信路の機密性が損失する．この鍵の有効期限は，1 回のセッションのみである．

(4) ユーザ鍵

SSH サーバがユーザを識別するために用いる公開鍵．また，この鍵の有効期限はないが，この鍵に対する秘密鍵が漏洩した時は更新が必要である．ここで，ユーザとは SSH クライアント上で操作を行う主体を示す．

3. SSH, Secure SHell について

3.1 概　　要

SSH サーバと SSH クライアントは，**プロトコルバージョンの交換**，**セッション鍵の交換**，**ユーザ認証**という 3 つの手続きをこの順序で行い，通信路を確立する．ここでは，これらの手続きを概観する．詳細は，各仕様を記述した文献に詳しい [2], [4], [6].

まず，**プロトコルバージョンの交換**では，主に SSH サーバと SSH クライアントの両者のサポートしている SSH のプロトコルバージョンが一致していることを確認し合う．SSH1 と SSH2 は，異なるプロトコルを使用しているため，両者に互換性はない．そのため，SSH サーバと SSH クライアントの両者のプロトコルバージョンが一致しないと通信路を張ることができない．ただし，前述したとおり，OpenSSH のサーバでは，両方の接続を可能としている．

次に，**セッション鍵の交換**における手続きでは，SSH クライアントが，接続をしている通信相手が目的の SSH サーバであるか認証し，さらに，通信路のデータを暗号化するセッション鍵，当該のセッションを識別するためのセッション識別子の交換が行われる．さらに，暗号化アルゴリズムの取り決めを行い，SSH1 では，ユーザ認証方式の取り決めも行う．SSH1 における「セッション鍵の交換」の手続きが実行されると，まず，暗号化アルゴリズム，ユーザ認証方式の取り決めが行われる．SSH サーバがサポートする暗号化アルゴリズム，ユーザ認証方式を SSH クライアントに提示し，SSH クライアントが SSH サーバが提示したものから選択し，それを SSH サーバに通報することで行われる．次に，セッション鍵の交換が行われる．SSH クライアントがセッション鍵を生成し，それをサーバ鍵とホスト鍵で二重に暗号化し，SSH1 のサーバに送信される．一方，SSH2 では，まず，セッション鍵の交換が行われる．セッション鍵の交換は改良された Diffie-Hellman 鍵配送 (注 3) によって共有される．また，その他の鍵交換方法も，文献 [12] で示されている．

最後の**ユーザ認証**の手続きでは，SSH サーバが SSH クライアントを操作するユーザの認証を行う．また，SSH2 では，ユーザの認証と共にユーザ認証方式の取り決めが行われる．この取り決めは，SSH クライアントが実行したいユーザ認証方式を SSH サーバに要求し，SSH サーバがその要求を受け入れるか，拒否するかの返信をすることにより行われる．本論文では，ユーザ認証方式として，パスワードを用いたユーザ認証 (以後，パスワードユーザ認証と呼ぶ) と公開鍵によるユーザ認証 (以後，公開鍵ユーザ認証と呼ぶ) を扱う．

3.2 プロトコルバージョンの交換

「プロトコルバージョンの交換」の手続きについて説明する．この手続きは，SSH サーバ S と SSH クライアント C が同じバージョンをサポートしているかどうかをお互いに確認する．SSH のシステムが通報するプロトコ

(注 3)：*diffie-hellman-group1-sha1* と呼ばれ，文献 [4] で示されている．

ルバージョンには，SSH1 のみをサポート，SSH2 のみをサポート，SSH1 と SSH2 をサポートすることを示す 3 種類がある．例えば，SSH1 のみをサポートしている場合は"SSH-1.5-1.2.22"であり，SSH1 と SSH2 をサポートしている場合は"SSH-1.99-1.2.22"であり，SSH2 のみをサポートしている場合は"SSH-2.0-1.2.22"となる [4]．ここで，"SSH-"の後の数字は，プロトコルバージョンを表し，その後の "1.2.22" はソフトウェアバージョンを表す：

$$\begin{array}{ll}(\text{手続き 1}) & S \rightarrow C : V_S \\ (\text{手続き 2}) & C \rightarrow S : V_C \end{array} \quad (1)$$

最初に，C が S に接続をすると，S はまず自身のプロトコルバージョンを C に送信する (手続き 1)．C はこれを受信すると，C のプロトコルバージョンを S に送信する (手続き 2)．

3.3 セッション鍵の交換

「プロトコルバージョンの交換」の手続きにおいて取り決められたバージョンに基づいて，SSH サーバと SSH クライアントは「セッション鍵の交換」の手続きを行う．

3.3.1 SSH1 におけるセッション鍵の交換

$$\begin{array}{ll}(\text{手続き 1}) & S \rightarrow C : P_S, H_S, SA, RN \\ (\text{手続き 2}) & C \rightarrow S : \\ & SA', RN, \{\{KS_{CS}\}_{P_S}\}_{H_S} \\ (\text{手続き 3}) & S \rightarrow C : \{ack_1\}_{KS_{CS}} \\ (\text{手続き 4}) & C \rightarrow S : \{username\}_{KS_{CS}} \\ (\text{手続き 5}) & S \rightarrow C : \{ack_2\}_{KS_{CS}} \end{array} \quad (2)$$

「プロトコルバージョンの交換」の手続きの後，S は C に「ホスト鍵 P_S」，「サーバ鍵 H_S」，「セキュリティアソシエーション SA」，「乱数 RN」の 4 つを通報する (手続き 1)．S から C に送信される SA には，S が C との間で利用したい暗号化アルゴリズムとユーザ認証方式が含まれている．これを受信した SSH クライアント C は，暗号化アルゴリズムを決定するために，S がサポートしているアルゴリズムから 1 つ選択し，SA' を生成する．この後，C は生成した「セキュリティアソシエーション SA'」をサーバ鍵，ホスト鍵で 2 重に暗号化されたセッション鍵 KS_{CS}，「S が生成した乱数 RN」と共に S に送信する (手続き 2)．S はこれを受信すると，C が選択した暗号化アルゴリズムを S が送信したそれから選択していることを確認する．この確認をすると，S は C が選択したものがセッション鍵で暗号化するための暗号化アルゴリズムであると判断する．その上で，S はセッション鍵を受け取ったことを示す通報を送信する (手続き 3)．次に，C は *username* を送信する (手続き 4)．最後に，S は *username* で示されたユーザが存在するかどうかを示す ack_2 を送信する (手続き 5)．この後，C は S が送信した SA によって示されたユーザ認証方式から 1 つ選択し，そのユーザ認証方式におけるユーザ認証の手続きを行う．

3.3.2 SSH2 におけるセッション鍵の交換

まず，SSH2 における「セッション鍵の交換」の手続きについて説明する：

$$\begin{array}{ll}(\text{手続き 1}) & S \leftrightarrow C : SA \\ (\text{手続き 2}) & C \rightarrow S : K_C \\ (\text{手続き 3}) & S \rightarrow C : K_S,\ P_S,\ H_{P_S^{-1}} \\ (\text{手続き 4}) & C \leftrightarrow S : ack \end{array} \quad (3)$$

SSH サーバ S と SSH クライアント C は，セキュリティアソシエーション SA を通報し合う (手続き 1)．この SA は，送信する主体がサポートする暗号化アルゴリズム，圧縮アルゴリズム等を示し，SSH1 と異なり，ユーザ認証方式は含まれない．次に，C は乱数 x を生成し，K_S $(= g^x \bmod p)$ を計算し，S にそれを送信する (手続き 2)．S はそれを受信した後，乱数 y を生成し，セッション鍵 KS_{CS} $(= g^{xy} \bmod p)$ を計算する．次に，S は K_S $(= g^y \bmod p)$ を計算し，$V_C, V_S, I_C, I_S, P_S, K_C, K_S, KS_{CS}$ を結合したハッシュ値 $H(= H(V_C, V_S, I_C, I_S, P_S, K_C, K_S, KS_{CS}))$ を計算する．この後，S は KS_{CS}，P_S，P_S^{-1} で署名をした H を送信する (手続き 3)．C は，S と同様に，KS_{CS} $(= g^{xy} \bmod p)$ と H を計算し，P_S を用いて H の電子署名を検証する．次に，S と C はお互いにセッション鍵を共有されていることを確認する (手続き 4)．

3.4 ユーザ認証

「セッション鍵の交換」の手続きを終了した時点で，SSH サーバと SSH クライアントは，セッション鍵 KS_{CS} を共有している．そのため，「ユーザ認証」の手続きにおいて，このセッション鍵を用いて SSH クライアントを操作するユーザの認証を行う．

ここでは，本論文で扱う SSH2 におけるパスワードユーザ認証と公開鍵ユーザ認証の手続きを示す．

3.4.1 SSH2 におけるパスワードユーザ認証

$$\begin{array}{ll}(\text{手続き 1}) & C \rightarrow S : \\ & \{username, password\_auth\}_{KS_{CS}} \\ (\text{手続き 2}) & S \rightarrow C : \{ack\}_{KS_{CS}} \end{array} \quad (4)$$

最初に，SSH クライアント C はユーザのログイン名 *username* とパスワードユーザ認証の要求とユーザのパスワードを示す *password_auth* を送信する (手続き 1)．この通報を受信した S は，パスワードユーザ認証の手続きを認め，ユーザの認証が成功したか失敗したかどうかを示す *ack* を送信する (手続き 2)．パスワードユーザ認証の手続きを認めない場合，または，認証に失敗した場合は，別のユーザ認証方式を C に要求することができる．S が別の認証を要求した場合は，ユーザはそのユーザ認証方式を行わなければ，ユーザ認証は成功しない．

3.4.2 SSH2 における公開鍵ユーザ認証

$$\begin{array}{ll}(\text{手続き 1}) & C \rightarrow S : \\ & \{username, public\_auth\}_{KS_{CS}} \\ (\text{手続き 2}) & S \rightarrow C : \{ack\}_{KS_{CS}} \end{array} \quad (5)$$

SSH クライアント C は，ユーザ名 *username*，公開鍵ユーザ認証の要求とユーザ鍵，その鍵の秘密鍵によるデジタル署名を示す *public_auth* を SSH サーバ S に送信する (手続き 1)．これを受信した S は，公開鍵ユーザ認証の手続きを認め，ユーザの認証が成功したか失敗したかどうかを示す *ack* を送信する (手続き 2)．公開鍵ユーザ認証の手続きを認めない場合，または，認証に失敗した場合は，パスワードユーザ認証と同様に，別のユーザ認証方式を C に示すことができる．S が別の認証を要求した場合は，ユーザはそのユーザ認証方式を行わなければならない．

4. Active Rollback 攻撃

Active Rollback 攻撃は，攻撃者 $\mathcal{I}$ が，SSH サーバ S と SSH クライアント C のセッションをハイジャックして C，

または，S の平文の通報を書き換えることにより，S と C が不当な利益を被る攻撃である．この攻撃は，S と C が SSH1 と SSH2 をサポートした利用を行っていることが前提であり，SSH1 と SSH2 の両者をサポートする実装システムでは考慮すべき攻撃である．そのため，OpenSSH 特有の問題であり，この攻撃は，OpenSSH [7] で確認した．

4.1 Version Rollback 攻撃

SSH1 と SSH2 をサポートしている C と S は，通常 SSH2 を利用するように取り決められている．しかし，「プロトコルバージョンの交換」の手続きにおいて攻撃者 $\mathcal{I}$ が，サーバのバージョン情報を書き換えることによって，SSH1 と SSH2 をサポートしている C と S が，SSH1 の利用を両者の意図に反して強制させられる．また，$\mathcal{I}$ が，クライアントのバージョン情報を書き換えることによって，C が SSH2 を利用しようとし，S は SSH1 を利用しようとする．結果として，S と C は通信路を張ることができない．これら 2 種類の攻撃を以下に示す．

サーバのバージョン情報の書き換え

最初に，C は S に接続をしようとした時，攻撃者 $\mathcal{I}$ が DNS (Domain Name System) サーバからの情報を書き換え，C から S への接続を奪い取る：

$$\begin{array}{ll}(\text{手続き 1}) & S \to \mathcal{I}(C) : version1.5 \\ (\text{手続き 1}') & \mathcal{I}(S) \to C : version1.0 \\ (\text{手続き 2}) & C \to \mathcal{I}(S) : version1.0 \\ (\text{手続き 2}') & \mathcal{I}(C) \to S : version1.0 \end{array} \quad (6)$$

サーバ S は，SSH1 と SSH2 のプロトコルをサポートしていることを示すバージョン情報 $version1.5$ を C に送信し，それを $\mathcal{I}$ が奪う (手続き 1)．次に，$\mathcal{I}(S)$ は，S が SSH1 のみをサポートしているようにバージョン情報を書き換えて C に送信する (手続き 1′)．これを受信した C は S が SSH1 のみをサポートしていると判断する．次に，C は $\mathcal{I}(S)$ に SSH1 をサポートしていることを送信する (手続き 2)．$\mathcal{I}(C)$ は，C から受け取ったバージョン情報を S に送信する (手続き 2′)．これを受信した S は，C が SSH1 のみをサポートしていると判断する．この後，S と C は SSH1 のセッション鍵の交換の手続きを実行する．また，SSH1 の「セッション鍵の交換」の手続き，「ユーザ認証」の手続きにおいて，お互いが通報したバージョンを SSL の様に通報認証などを用いて確認することはないため，攻撃者による SSH1 の強制は成立する．

以上のことから，S と C が SSH2 をサポートしていたとしても，この手法により，SSH1 を強制させられる．

クライアントのバージョン情報の書き換え

サーバのバージョン情報の書き換えの場合と同様に，C は S に接続をしようとする時，攻撃者 $\mathcal{I}$ が DNS (Domain Name System) サーバからの情報を書き換え，C から S への接続を奪い取る：

$$\begin{array}{ll}(\text{手続き 1}) & S \to \mathcal{I}(C) : version1.5 \\ (\text{手続き 1}') & \mathcal{I}(S) \to C : version1.5 \\ (\text{手続き 2}) & C \to \mathcal{I}(S) : version2.0 \\ (\text{手続き 2}') & \mathcal{I}(C) \to S : version1.0 \end{array} \quad (7)$$

サーバ S は，$version1.5$ を SSH クライアント C に送信し，$\mathcal{I}$ が奪い，それを C に送信する (手続き 1,1′)．次に，C は $version2.0$ を $\mathcal{I}(S)$ に送信する (手続き 2)．これを受信した $\mathcal{I}$ は，C のバージョン情報を書き換えて，それを S に送信する (手続き 2′)．この後，C は SSH2 のセッション鍵の交換の手続きを実行し，S は SSH1 のセッション鍵の交換の手続きを実行する．

以上のことから，この攻撃により，C と S がそれぞれ異なるプロトコルを実行しようとする．その結果，S と C は，通信路を確立できない．

4.2 Cipher and Authentication Rollback 攻撃

「プロトコルバージョンの交換」の手続きにおいて，SSH1 を強制されているサーバ S とクライアント C は SSH1 におけるセッション鍵交換の手続きを実行する：

$$\begin{array}{ll}(\text{手続き 1}) & S \to \mathcal{I}(C) : P_S, H_S, SA, RN \\ (\text{手続き 1}') & \mathcal{I}(S) \to C : P_S, H_S, SA', RN \\ (\text{手続き 2}) & C \to \mathcal{I}(S) : SA'', RN, \\ & \quad \{\{KS_{CS}\}_{P_S}\}_{H_S} \\ (\text{手続き 2}') & \mathcal{I}(C) \to S : SA'', RN, \\ & \quad \{\{KS_{CS}\}_{P_S}\}_{H_S} \\ (\text{手続き 3}) & S \to \mathcal{I}(C) : \{ack_1\}_{KS_{CS}} \\ (\text{手続き 3}') & \mathcal{I}(S) \to C : \{ack_1\}_{KS_{CS}} \\ (\text{手続き 4}) & C \to \mathcal{I}(S) : \\ & \quad \{username_C\}_{KS_{CS}} \\ (\text{手続き 4}') & \mathcal{I}(C) \to S : \\ & \quad \{username_C\}_{KS_{CS}} \\ (\text{手続き 5}) & S \to \mathcal{I}(C) : \{ack_2\}_{KS_{CS}} \\ (\text{手続き 5}') & \mathcal{I}(S) \to C : \{ack_2\}_{KS_{CS}} \end{array} \quad (8)$$

サーバ S は最初の通報を送信する (手続き 1)．C に成り済ました攻撃者 $\mathcal{I}(C)$ はこれを受け取ると，SA' として暗号化アルゴリズムとユーザ認証方式を書き換える．暗号化アルゴリズムは，S がサポートしたものの中で，より脆弱なもののみをサポートしているように書き換え，ユーザ認証方式はパスワードのみをサポートしているように書き換える．この SA' のデータ構造は，5.3 節で後述する．$\mathcal{I}(S)$ は，SA 以外は変更せずに C に送信する (手続き 1′)．これを受信した C は，S がサポートしている暗号化アルゴリズムとユーザ認証方式は，$\mathcal{I}$ が書き換えた脆弱な暗号化アルゴリズムとパスワードユーザ認証のみをサポートしていると判断する．その後，C は $\mathcal{I}$ が書き換えた暗号化アルゴリズムで SA'' を作成せざるを得ず，それを $\mathcal{I}(S)$ に送信する (手続き 2)．$\mathcal{I}(S)$ は，S に SA'' をそのまま送信する (手続き 2′)．これを受信した S は，選択された暗号化アルゴリズムが $\mathcal{I}$ が強制させたものであるにもかかわらず，C が選択したと判断する．その後，S はセッション鍵を受け取ったことを示す通報を送信する (手続き 3)．また，$\mathcal{I}(S)$ も同様にそれを C に送信する (手続き 3′)．次に，C は $username_C$ をセッション鍵 KS_{CS} で暗号化し，それを送信する (手続き 4)．$\mathcal{I}(C)$ はそのまま S に送信する (手続き 4′)．最後に，S は $username_C$ で示されたユーザが存在するかどうかを示す ack_2 を送信し (手続き 5)，これを受信した $\mathcal{I}(C)$ はそのまま S に送信する (手続き 5′)．この後，C は S がサポートしているユーザ認証方式は，パスワード認証のみと判断しているため，C と S はユーザ認証の手続きにおいて，パスワードユーザ認証を行う．

以上のことから，S と C がお互いに強固な暗号化アルゴリズムをサポートしていたとしても，攻撃者 $\mathcal{I}$ は，S と C の通信路を暗号化する暗号化アルゴリズムを $\mathcal{I}$ の任意の選

択によって強制させることができることが分かった．また，公開鍵ユーザ認証を行うことを前提としたサーバ，もしくは，ユーザであっても，攻撃者によって，パスワードユーザ認証を強制されることができる．これら2つは，サーバ，クライアントの意図に反するものである．また，これらの脆弱性は，SSH1のみに存在するものであり，SSH2では，通報認証が行われるため存在しない．

5. Passive Rollback 攻撃

5.1 概　　要

SSH に対する攻撃方法の一つとして，「サーバを騙すMITM 攻撃」がある [9]．これは，SSH1，SSH2 におけるユーザ認証の不備が原因であり，この脆弱性のため，攻撃者は，パスワードユーザ認証において，パスワードを奪取することができる．ただし，この攻撃は，公開鍵ユーザ認証に対しては適用できない．しかしながら，公開鍵ユーザ認証を利用したい SSH クライアント (ユーザ) にも，パスワードユーザ認証を強制させ，パスワードを奪取することができることが分かった．次節以降では，SSH1，SSH2 のクライアント，サーバ共に，より安全な公開鍵ユーザ認証をサポートしているにも関わらず，上述の「サーバを騙すMITM 攻撃」において，Rollback 攻撃を行うことによって，パスワードユーザ認証を強制させ，パスワードを奪取する攻撃のシナリオを説明する．また，この攻撃は Active Rollback 攻撃と異なり，OpenSSH のみに限らず一般的な SSH に対しても実現可能な攻撃である．

5.2 サーバを騙す MITM 攻撃

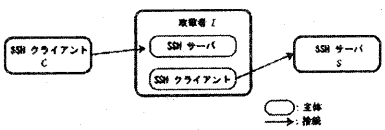

図 1　サーバを騙す MITM 攻撃の概観

「サーバを騙すMITM 攻撃」は，正規のサーバのミラーサーバであるなどとして偽っている攻撃者 $\mathcal{I}$ に，SSH クライアント C が接続する際の攻撃である [9], [13].

この攻撃が行われるためには，次に示す仮定が必要となる：

最初に，攻撃者は，ユーザに「NFS (Network File System) によって共有されるファイル資源を供給させることができる新たなミラーサーバである」などとして告知する．この告知を信じないユーザもいるだろうが，サーバの振りをしている攻撃者を正規のミラーサーバと信じ，接続するユーザが存在するかも知れない．この状況において，この告知を信じた SSH クライアント (ユーザ) は，サーバの振りをしている攻撃者に接続をする．さらに，ここで重要なことは，攻撃者は，正規のサーバが保持している計算機資源を全く保持しておらず，もちろん，ユーザのパスワードも知らない．

この仮定の元に，SSH1 における「サーバを騙すMITM 攻撃」について概観する．SSH2 に対しても，同様にこの攻撃が行われ，5.4.1 節において示す．

SSH クライアントが正規のサーバの振りをしている攻撃者に接続した後，SSH クライアント C と攻撃者 $\mathcal{I}$ は，「プロトコルバージョンの交換」の手続きを行う．また，SSH クライアント C によって接続された攻撃者 $\mathcal{I}$ も，同様に，正規のサーバに対し SSH クライアントとして接続し，「プロトコルバージョンの交換」の手続きを行う (図 1)．この後，「セッション鍵の交換」の手続きが行われる．まず，SSH サーバは自身のホスト鍵を攻撃者に送信する．同様に，攻撃者も自身のホスト鍵を SSH クライアントに送信する．次に，SSH クライアントは攻撃者に通信路を暗号化するためのセッション鍵 $KS_{C\mathcal{I}}$ を生成し，攻撃者のホスト鍵で暗号化し，送信する．同様に，攻撃者も SSH サーバにセッション鍵 $KS_{\mathcal{I}S}$ を送信し，SSH サーバのホスト鍵で暗号化し，送信する．最後に，SSH サーバは攻撃者にセッション鍵を共有したことを示し，同様に，攻撃者も SSH クライアントにセッション鍵を共有したことを示す．この時点で，SSH クライアントと攻撃者はセッション鍵 $KS_{C\mathcal{I}}$ を共有し，攻撃者と SSH サーバはセッション鍵 $KS_{\mathcal{I}S}$ を共有している (図 2)．

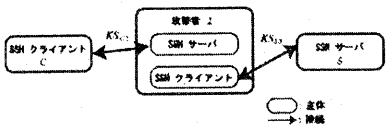

図 2　セッション鍵の共有状態

以上のことから，攻撃者は正規の SSH サーバの振りをすることにより，SSH クライアントと SSH サーバの間に入り込むことに成功する．さらに，SSH クライアント (ユーザ) がパスワードユーザ認証を実行した場合，ユーザのパスワードを奪取することができる．これにより，攻撃者は SSH クライアントの振りをしてサーバ上で自由にクライアントのリソースを利用できる．

5.3 SSH1 における攻撃のシナリオ

ここでは，SSH1 において，上述の「サーバを騙すMITM 攻撃」とその上での Rollback 攻撃を行うシナリオを説明する．攻撃者は，SSH クライアントと SSH サーバの間に入り込み，「プロトコルバージョンの交換」の手続きの後，「セッション鍵の交換」の手続きを行う：

$$
\begin{array}{ll}
\text{(手続き 1)} & S \to \mathcal{I}(C) : P_S, H_S, SA, RN \\
\text{(手続き 1')} & \mathcal{I} \to C : P_{\mathcal{I}}, H_{\mathcal{I}}, SA', RN \\
\text{(手続き 2)} & C \to \mathcal{I} : SA'', RN, \\
 & \qquad \{\{KS_{C\mathcal{I}}\}_{P_{\mathcal{I}}}\}_{H_{\mathcal{I}}} \\
\text{(手続き 2')} & \mathcal{I}(C) \to S : SA'', RN, \\
 & \qquad \{\{KS_{\mathcal{I}S}\}_{P_S}\}_{H_S} \\
\text{(手続き 3)} & S \to \mathcal{I}(C) : \{ack_1\}_{KS_{\mathcal{I}S}} \\
\text{(手続き 3')} & \mathcal{I} \to C : \{ack_1\}_{KS_{C\mathcal{I}}} \\
\text{(手続き 4)} & C \to \mathcal{I} : \{username_C\}_{KS_{C\mathcal{I}}} \\
\text{(手続き 4')} & \mathcal{I}(C) \to S : \{username_C\}_{KS_{\mathcal{I}S}} \\
\text{(手続き 5)} & S \to \mathcal{I}(C) : \{ack_2\}_{KS_{\mathcal{I}S}} \\
\text{(手続き 5')} & \mathcal{I} \to C : \{ack_2\}_{KS_{C\mathcal{I}}}
\end{array}
\tag{9}
$$

「セッション鍵の交換」の手続きが行われると，SSH サーバ S は攻撃者 $\mathcal{I}(C)$ に最初の通報を送信する (手続き 1)．S から送信される SA のデータ構造は，以下のようになる：

```
struct{
   32-bit int supported_ciphers_mask
   32-bit int supported_authentication_mask
}SA
```

supported_ciphers_mask は，S がサポートしている暗号化アルゴリズムすべてを示しており，

supported_authentication_mask は，S が「ユーザ認証」の手続きにおいてサポートしているユーザ認証方式すべてを示す．ここでは，supported_authentication_mask は，S が「パスワードユーザ認証」と「公開鍵ユーザ認証」をサポートしていることを示す情報が含まれていると仮定する．$\mathcal{I}$ はこれを受信すると，SA におけるユーザ認証方式を「パスワードユーザ認証」のみをサポートしているように SA' と書き換え，自身の公開鍵 $P_{\mathcal{I}}$ を含めた独自の通報を C に送信する (手続き 1′)．これを受信した C は，$\mathcal{I}$ がサポートしているユーザ認証方式は，パスワードユーザ認証のみであると判断する．次に，C は，SA'' を生成する．C が生成する SA'' のデータ構造は，以下のようになる：

```
struct{
   1 byte cipher_type
}SA'
```

cipher_type は，supported_cipher_mask が示している暗号化アルゴリズムから 1 つ選択したものである．その上で，生成した SA'' を $\mathcal{I}$ に送信する (手続き 2)．それを受け取った $\mathcal{I}$ は，そのまま SA'' を S に送信する (手続き 2′)．その後，S は $KS_{\mathcal{I}S}$ を共有したことを示す通報を送信する (手続き 3)．$\mathcal{I}$ も，同様に，$KS_{C\mathcal{I}}$ を共有したことを示す通報を C に送信する (手続き 3′)．次に，C は，$username_C$ を $KS_{C\mathcal{I}}$ で暗号化して，送信する (手続き 4)．$\mathcal{I}(C)$ は，C に成りすますため，$username_C$ を $KS_{\mathcal{I}S}$ で暗号化して送信する (手続き 4′)．これを受信した S は $username_C$ が存在するかどうかを示す ack_2 を送信する (手続き 5)．$\mathcal{I}$ は，それをセッション鍵 $KS_{\mathcal{I}S}$ を用いて復号し，$KS_{C\mathcal{I}}$ で暗号化した ack_2 を送信する (手続き 5′)．

以上のことから，攻撃者は，SSH1 のユーザ認証の手続きにおいて，公開鍵ユーザ認証を利用したいユーザにさえ，パスワードユーザ認証を強制させることができ，その結果，パスワードを奪取して，サーバを騙すことができる．

5.4 SSH2 における攻撃のシナリオ

ここでは，SSH2 において，上述の「サーバを騙す MITM 攻撃」とその上での Rollback 攻撃を行うシナリオを説明する．SSH2 においても，SSH1 と同様に，攻撃者 $\mathcal{I}$ は SSH クライアント C に対してパスワードユーザ認証を強制することができることを示す．ここでも，5.2 節で示した仮定を前提とする．

5.4.1 セッション鍵の交換

SSH1 における「セッション鍵の交換」の手続きと同様に，攻撃者は，SSH クライアントと SSH サーバの間に入り込み，それぞれとの間で，「プロトコルバージョンの交換」の手続きの後，「セッション鍵の交換」の手続きを行う：

$$
\begin{array}{ll}
\text{(手続き 1)} & S \leftrightarrow \mathcal{I}(C) \quad : SA \\
\text{(手続き 1′)} & \mathcal{I} \leftrightarrow C \quad : SA' \\
\text{(手続き 2)} & C \rightarrow \mathcal{I} \quad : K_C \\
\text{(手続き 2′)} & \mathcal{I}(C) \rightarrow S \quad : K_{\mathcal{I}} \\
\text{(手続き 3)} & S \rightarrow \mathcal{I}(C) \quad : K_S, P_S, \\
 & \quad \{H(V_{\mathcal{I}}, V_S, I_{\mathcal{I}}, I_S, P_S, K_{\mathcal{I}}, K_S, KS_{\mathcal{I}S})\}_{P_S^{-1}} \\
\text{(手続き 3′)} & \mathcal{I} \rightarrow C \quad : K_{\mathcal{I}}, P_{\mathcal{I}} \\
 & \quad \{H(V_C, V_{\mathcal{I}}, I_C, I_{\mathcal{I}}, P_{\mathcal{I}}, K_C, K_{\mathcal{I}}, KS_{C\mathcal{I}})\}_{P_{\mathcal{I}}^{-1}} \\
\text{(手続き 4)} & C \rightarrow \mathcal{I} \quad : ack \\
\text{(手続き 4′)} & \mathcal{I}(C) \rightarrow S \quad : ack
\end{array}
\tag{10}
$$

C に成り済ました攻撃者 $\mathcal{I}(C)$ と SSH サーバ S，SSH クライアント C と $\mathcal{I}$ は，それぞれ，セキュリティアソシエーション SA，SA' をそれぞれ交換する (手続き 1，1′)．次に，C は K_C を計算し，$\mathcal{I}$ にそれを送信する (手続き 2)，同様に，$\mathcal{I}(C)$ も $K_{\mathcal{I}}$ を計算して，S にこれを送信する (手続き 2′)．これを受信した S は，セッション鍵 $KS_{\mathcal{I}S}$ を計算する．さらに，S は，K_S，P_S と共に，S の秘密鍵 P_S^{-1} で電子署名したハッシュ値を送信する (手続き 3)．同様に，$\mathcal{I}$ も，セッション鍵などを計算し，$K_{\mathcal{I}}$，$P_{\mathcal{I}}$ と共に，$P_{\mathcal{I}}^{-1}$ で電子署名したハッシュ値を送信する (手続き 3′)．C はセッション鍵 $KS_{C\mathcal{I}}$ を計算し，$P_{\mathcal{I}}$ を用いて電子署名を検証する．その後，C と $\mathcal{I}$，C に成りすました $\mathcal{I}(C)$ と S は，セッション鍵が共有されていることを確認する (手続き 4，4′)．

以上のことから，攻撃者 $\mathcal{I}$ は，SSH クライアント C と SSH サーバ S とのやり取りの間に入り込み，セッション鍵 $KS_{C\mathcal{I}}$ を，C と $\mathcal{I}$ との間で共有し，同様に，セッション鍵 $KS_{\mathcal{I}S}$ を，$\mathcal{I}$ と S との間で共有する．

5.4.2 ユーザ認証の手続き

SSH2 の「セッション鍵の交換」手続きの後，「ユーザ認証」の手続きを行う．この手続きにおいて，ユーザ認証方式の取り決めと C を操作するユーザの認証を行う：

$$
\begin{array}{ll}
\text{(手続き 1)} & C \rightarrow \mathcal{I} : \\
 & \quad \{username_C, public\_auth\}_{KS_{C\mathcal{I}}} \\
\text{(手続き 1′)} & \mathcal{I} \rightarrow C : \\
 & \quad \{ack_1\}_{KS_{C\mathcal{I}}} \\
\text{(手続き 2)} & C \rightarrow \mathcal{I} : \\
 & \quad \{username_C, password\_auth\}_{KS_{C\mathcal{I}}} \\
\text{(手続き 2′)} & \mathcal{I}(C) \rightarrow S : \\
 & \quad \{username_C, password\_auth\}_{KS_{\mathcal{I}S}} \\
\text{(手続き 3)} & S \rightarrow \mathcal{I}(C) : \quad \{ack_2\}_{KS_{\mathcal{I}S}} \\
\text{(手続き 3′)} & \mathcal{I} \rightarrow C : \quad \{ack_2\}_{KS_{C\mathcal{I}}}
\end{array}
\tag{11}
$$

ここで，ユーザは公開鍵ユーザ認証を要求したとしても，攻撃者によりパスワードユーザ認証を強制されることを示す．C は，公開鍵ユーザ認証を要求するため，$KS_{C\mathcal{I}}$ で暗号化された $username_C$，$public\_auth$ を正規のサーバに成りすました $\mathcal{I}$ に送信する (手続き 1)．C が送信する $public\_auth$ のデータ構造は，以下のようになる：

```
struct{
  string method_name
  string public_key_algorithm_name
  string public_key
  string public_key_blob
}public_auth
```

この時，手続き 1 の通報において，method_name は，C が利用したいユーザ認証方式を示し，公開鍵ユーザ認証では，"publickey"となる．public_key_algorithm_name は，公開鍵ユーザ認証において利用する公開鍵アルゴリズムを示す．public_key は，ユーザ鍵であり，public_key_blob は，ユーザ鍵の秘密鍵によるデジタル署名である．次に，$\mathcal{I}$ は，C に対して「パスワードユーザ認証」を強制するために，公開鍵ユーザ認証を行うことができないこととパスワードユーザ認証を要求する ack_1 をセッション鍵 $KS_{C\mathcal{I}}$ で暗号化し送信する (手続き 1′)．この ack_1 のデータ構造は，以下のようになる：

```
struct{
  byte SSH_MSG_USERAUTH_FAILURE
  string authentications
}ack_1
```

SSH_MSG_USERAUTH_FAILURE は，認証が失敗したことを示す通報である．また，authentications は，この後，実行できるユーザ認証方式を示し，ここでは，パスワードユーザ認証となる．これを受信した C は，パスワードユーザ認証を行なわなければならないので，当該セッションを継続したいユーザであれば，$username_C$ と *password_auth* を，正規のサーバに成り済ました攻撃者 $\mathcal{I}$ に送信する (手続き 2)．$\mathcal{I}$ が送信する *password_auth* のデータ構造は，以下のようになる：

```
struct{
  string method_name
  string plaintext_password
}password_auth
```

method_name は，C が利用するユーザ認証方式を示し，パスワードユーザ認証では，"password"となる．plaintext_password は，ユーザのパスワードであり，平文がセッション鍵で暗号化されて送信される．平文のパスワードがセッション鍵で暗号化されて送信されることは，文献[6]で仕様が策定されている．つまり，ここで C はパスワードユーザ認証を行うとともに，平文のパスワードをセッション鍵で暗号化して攻撃者に送信している．これを受信した $\mathcal{I}$ は，セッション鍵 $KS_{C\mathcal{I}}$ で復号することにより，ユーザのパスワードを奪取することができる．次に，C に成りすました $\mathcal{I}(C)$ はパスワードユーザ認証を行うことを $KS_{\mathcal{I}S}$ で暗号化して，S に要求する (手続き 2′)．これを受け取った S は，パスワードユーザ認証の要求を受け入れ，認証が成功したか失敗したかを示す ack_2 を送信する (手続き 3)．同様に，$\mathcal{I}$ もその通報を C に送信する (手続き 3′)．

以上のことから，SSH2においても，SSH1と同様に，攻撃者は公開鍵ユーザ認証を利用したいユーザにさえ，パスワードユーザ認証を強制させることができることを示した．このため，攻撃者はユーザのパスワードを奪取することができ，サーバを騙すことができる．

6. まとめ

本論文では，Active Rollback 攻撃と Passive Rollback 攻撃，全部で2種類5つの攻撃に対する SSH の脆弱性を示した．これらの Rollback 攻撃は，利便性を重視するあまり，脆弱となる運用ポリシーに起因する問題ではなく，SSH クライアントと SSH サーバが望まない「より脆弱な方式」を強制されてしまうというプロトコルの設計における問題である．主な原因は，ネゴシエーションにおける各主体を他方が通報認証していないことである．また，後者の Passive Rollback 攻撃における問題はさらに，文献[10],[11]で示されている安全に共有したセッション鍵でないにも関わらず，パスワードを平文で送信していることにある．

今回はSSH (OpenSSH) を題材にしたが，この Rollback 攻撃と言うのは，一般的な認証プロトコルを実装したシステムに組み込むときに考慮すべき攻撃である．

文　献

[1] D.J.Barrett, R.E.Silverman: SSH, The Secure Shell, O'REILLY, (February 2001). ISBN:0-596-00011-1.

[2] T.Ylonen: The SSH (Secure SHell) Remote Login Protocol, Internet Draft, Network Working Group, (November 1995).

[3] T.Ylonen, T.Kivinen, M.Saarinen, T.Rinne, and S.Lehtinen: SSH Protocol Architecture draft-ietf-secsh-architecture-11.txt, Internet Draft, Network Working Group, (November 2001).

[4] T.Ylonen, T.Kivinen, M.Saarinen, T.Rinne, and S.Lehtinen: SSH Transport Layer Protocol draft-ietf-secsh-transport-11.txt, Internet Draft, Network Working Group, (November 2001).

[5] T.Ylonen, T.Kivinen, M.Saarinen, T.Rinne, and S.Lehtinen: SSH Connection Protocol draft-ietf-secsh-connect-14.txt, Internet Draft, Network Working Group, (November 2001).

[6] T.Ylonen, T.Kivinen, M.Saarinen, T.Rinne, and S.Lehtinen: SSH Authetication Protocol dragt-ietf-secsh-userauth-13.txt, Internet Draft, Network Working Group, (November 2001).

[7] http://www.openssh.com/index.html

[8] A.Freier, P.Kocher, and P.Kaltorn: The SSL Protocol Version 3.0, http://home.netscape.com/eng/ssl3/draft302.txt.

[9] T.Saito, T.Kito, K.Umesawa and F.Mizoguchi: Architectural Defects of the Secure Shell, Proc. of 13th Internatinal Workshop on Data and Expert Systems Applications, pp.22-28, (2002). ISBN 0-7695-1668-8, IEEE Computer Society.

[10] 齋藤孝道：認証プロトコルの基礎，コンピュータソフトウェア (日本ソフトウェア科学会論文誌)，(チュートリアル)，Vol.19，No.4，pp.52-63，(2002).

[11] 齋藤孝道：認証プロトコルの機能と構成，コンピュータソフトウェア (日本ソフトウェア科学会論文誌)，(チュートリアル)，Vol.19，No5，pp.60-73，(2002).

[12] M.Friedl, N.Probos and W.A.Simpsom : Diffie-Hellman Group Exchange for the SSH Transport Layer Protocol dragt-ietf-secsh-dh-group-exchange-00.txt, INTERNET-DRAFT, Network Working Group, (April 2001).

[13] G.Lowe : Breaking and Fixing the Needham-Schroeder Public-Key Protocol using FDR, In T.Margaria and B.Steffen, editors, *Tools and Algorithms for the Construction and analysis of Systems*.Eds.), LNCS, Vol.1055, pp.147-166, (1996).