SSHに対する2つのRollback攻撃と対策について

鬼頭利之[†] 齋藤孝道[††]

ネットワーク通信における安全性を確保するセキュリティプロトコルの1つとしてSSH（Secure SHell）がある．SSHにはSSH1とSSH2があり，これらのプロトコルバージョンには互換性がない．また，利用できる暗号化アルゴリズムとユーザ認証方式が複数提供されている．そのため，接続の際にSSHサーバとクライアントは，両者がサポートするプロトコルバージョン，暗号化アルゴリズムおよびユーザ認証方式の中で，基本的には最も安全なものに合意することとなっている．しかしながら，このような場合，2つのタイプのRollback攻撃が実現することが分かった．本論文では，これらの攻撃を示すとともに，その原因と解決方法について示す．

On Two Types of Rollback Attacks against SSH and their Countermeasures

Toshiyuki Kito[†] and Takamichi Saito[††]

SSH (Secure SHell) is one of the security protocols to secure the network communication. There are two versions of SSH, e.g. SSH1 and SSH2, and they are not compatible. Also, SSH supports the various kinds of encryption algorithms and user-authentication methods. Moreover, SSH client and server must agree with protocol version, ciphersuites, and user-authentication method as most secure ways by establishing communication channel. Owing to it, we found SSH had defects. In this paper, we denote two types of Rollback attacks, and propose the countermeasures against such the attacks.

1. はじめに

SSH（Secure SHell）[1)~6)]は，盗聴や改竄，成りすましの脅威が存在する「安全でない」ネットワークを介して，遠隔の計算機上でコマンドの実行や遠隔の計算機とのファイル転送を，安全性☆を確保したうえで，実現するソフトウェアとして広く利用されている．このように通信における安全性を確保するプロトコルをセキュリティプロトコルと呼び，SSHのほかにSSL（Secure Socket Layer）[7)]，TLS（Transport Layer Security）[8)]およびIPSec[9)]等がある．

SSHには，SSH version1[2)]（以降，SSH1と呼ぶ）とSSH version2[3)~6)]（以降，SSH2と呼ぶ）の互換性のない2つのプロトコルバージョンがあり，SSH1とSSH2ともに，サーバとクライアントの間で，プロトコルバージョン交換，セッション鍵交換，ユーザ認証により，安全な通信路を確立する．この確立の過程をSSH Handshakeと呼ぶ．また，SSHは，SSH1とSSH2のいずれのプロトコルバージョンでも接続可能な運用ができ，実際にこうした運用を行うSSHサーバが全体の73.30%（2004年9月現在）という報告[10)]もある．さらに，3DES，RC4，Blowfishといった複数の暗号化アルゴリズムと，パスワードを用いたユーザ認証（以降，パスワードユーザ認証と呼ぶ）と公開鍵を用いたユーザ認証（以降，公開鍵ユーザ認証と呼ぶ）という2つのユーザ認証方式☆☆がある．これら複数の暗号化アルゴリズムやユーザ認証方式の利用を許す運用ができ，実際にこうした運用がなされていると考えられる．また，OpenSSH[11)]のユーザ認証方式に関しては，公開鍵ユーザ認証とパスワードユーザ認証の両方を利用でき，公開鍵ユーザ認証の利用を優先する設定がデフォルトとなっている．そのため，通常のSSH Handshakeにおいて，SSHサーバとクライアントは，プロトコルバージョン，暗号スイートおよびユーザ認証方式を合意しなければならない．この合意はSSHサーバとクライアントが事前に設定した優

† 株式会社東芝研究開発センター
Corporate Research & Development Center, TOSHIBA Corporation
†† 明治大学
Meiji University

☆ ここでの安全性とは真正性，機密性，完全性からなる．
☆☆ その他のユーザ認証方式もあるが，本論文では取り扱わないので省略する．

先度に基づき行われるが，OpenSSH のデフォルト設定のように，SSH2 を利用し，最も安全とされる暗号スイートと公開鍵ユーザ認証を利用するのが一般的である．しかしながら，他方が利用したい「プロトコルバージョン，より安全な暗号スイートとユーザ認証方式」をサポートしていない場合，各々の運用ポリシに従い「次善のプロトコルバージョン，暗号スイートとユーザ認証方式」で合意することができるように運用する．

このように複数の暗号スイートやユーザ認証方式の利用を許す運用を行う際，何かしらの攻撃が存在する可能性がある．たとえば，HTTPS（Hyper Text Transfer Protocol over SSL）で用いる SSL においても，SSL クライアントとサーバは，プロトコルバージョンや暗号スイートの合意を行っており，攻撃者が任意のプロトコルバージョンや暗号スイートを強制できるという攻撃がある[12]．そこで，Java 言語で実装した攻撃プログラムを用いて SSH（OpenSSH-4.1）において確認したところ，この攻撃が成立することが分かった．本論文では，これをセッションハイジャック Rollback 攻撃と呼び，さらに，これとは異なるタイプの Rollback 攻撃（以降，ミスリード Rollback 攻撃と呼び，詳細は後述する）が成立することも分かり，セッション Rollback 攻撃と同様に，Java 言語で実装した攻撃プログラムを作成することにより確認した．本論文では，セッションハイジャック Rollback 攻撃とミスリード Rollback 攻撃という 2 つのタイプの Rollback 攻撃を示し，これらの攻撃に対する考察をしたうえでプロトコルの変更による技術対策と運用上の対策を示す．

以降，2 章において，本論文で用いる表記，SSH で用いる鍵の説明を行い，3 章において，SSH プロトコルを概説する．その後，4 章において，2 つのタイプの Rollback 攻撃の説明を定め，5 章と 6 章において，実際の攻撃をそれぞれ示す．7 章において，考察を行い，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，用語について説明する．

2.1 SSH で用いる鍵

本節では，SSH で用いるホスト鍵，サーバ鍵，セッション鍵，ユーザ鍵の 4 種類の鍵について説明する：

(1) ホスト鍵
SSH クライアントが SSH サーバを識別し，セッション鍵（後述）を共有する際に用いる公開鍵．

(2) サーバ鍵
暗号化されたデータをより解読されにくくするために，ホスト鍵で暗号化されたものをさらに暗号化するための公開鍵．この鍵は SSH1 のみが使用し，定期的（デフォルト設定では 1 時間）に変更される．

(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．この鍵の有効期限は，1 回のセッションのみ．

(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．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．

2.2 表記と用語

主体に関する表記

SSH は，SSH サーバ S と SSH クライアント C から構成される．また，攻撃者を $\mathcal{I}$ によって表す．$\mathcal{I}(S)$，$\mathcal{I}(C)$ 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 は任意の通報を示し，返信の際の確認通報は，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 を互いに通報し合うことを以下のように示す：

$$C \leftrightarrow S \ : \ Msg$$

たとえば，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：

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

さらに，以下の 2 つの通報の違いに注意されたい：

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

前者は C が通信相手を S であると想定しているが，実際には $\mathcal{I}$ に Msg を送信していることを示している．一方，後者は C がサーバのふ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．

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．Y が公開鍵の場合 Y に対する秘密鍵を Y^{-1} で示し，Y^{-1} によって電子署名された通報 X は，$\{X\}_{Y^{-1}}$ で示す．鍵 Z と鍵 Y を

用いて通報 X が鍵長が小さいほうの鍵から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，ユーザ鍵をそれぞれ示す．

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 のプロトコルバージョンであり，V_C は SSH クライアント C のプロトコルバージョンを示す．さらに，$v1$，$v2$，$v1+2$ と表記した場合は，SSH1 のみ，SSH2 のみ，SSH1 と SSH2 をサポートした運用を行っていることを示すプロトコルバージョンをそれぞれ表す．

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 は部分群の位数である．x，y は，C，S 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，K_{CS} は $g^{xy} \bmod p$ である．本論文では，これをセッション鍵とする．また，KS_S，KS_C における乱数の選択に関しては，議論の本質ではないので省略する．

ユーザ認証で用いる表記

$password$ と $uname$ は，ユーザのパスワードとログイン名を示す．$modulus$ は任意の公開鍵の一部もしくは全部を示す．$challenge$ は，S によって生成される 256 ビットの乱数を示す．特に，$password_C$（$uname_C$ や $modulus_C$）と書くとき，これらは C に関するものである．ただし，明らかなときには省略する．また，7.2.1.1 および 7.2.1.2 で用いる $password$ は，文字列をもとにして作成された暗号鍵である．作成方法は，PKCS#5 [13] がある．

その他の表記

RN は乱数を示す．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暗号スイートである．SA_C，SA_S と表記したとき，これらは C と S に関する暗号スイートである．S_{CS} は C と S の当該セッションを識別するセッション識別子を表す．$H(Msg_1)$ 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．

3. SSH プロトコル

3.1 SSH の概要

S と C の SSH Handshake は以下とおりである：

(1) プロトコルバージョン交換
S と C がプロトコルバージョンに関する情報を通報し合い，互いに同じプロトコルバージョンをサポートしているかどうかの確認を行う．

(2) セッション鍵交換
C は，S の認証にともないセッション鍵の交換を行う．また，バルクデータのための暗号化アルゴリズムの取り決めを行う．これは，S がサポートしているすべての暗号化アルゴリズムを SA に含めて提示し，C がそこから選択したものを SA' として返信することにより行われる．

(3) ユーザ認証
S が C を操作するユーザを認証する．当然のことながら，S と C は，当該セッションにおいて，同じユーザ認証方式を利用することに同意しなければならない．SSH1 では，セッション鍵交換における SA として S から C に提示され，その SA に基づき C が選択する．一方，SSH2 では，C が行いたいユーザ認証方式を要求し，これを S のポリシに基づき受け入れるかどうかを判断する．ただし，S が受け入れない場合，もしくは，認証が失敗した場合には，S はその返信とともに利用できるユーザ認証方式を提示することができる．なお，ユーザ認証を行わない運用もあるが，本論文ではこれを扱わない．

3.2 プロトコルバージョン交換

SSH1 と SSH2 ともに以下のやりとりを行う：

$$\begin{array}{ll}(\mathrm{A1}) & S \rightarrow C \ : \ V_S \\ (\mathrm{A2}) & C \rightarrow S \ : \ V_C \end{array} \qquad (1)$$

3.3 セッション鍵交換

S と C は，プロトコルバージョン交換において取り決められたプロトコルバージョンに基づきセッション鍵交換を行う．

3.3.1 SSH1 のセッション鍵交換

SSH1 におけるセッション鍵交換は以下のとおりである：

$$\begin{array}{ll}(\mathrm{B1}) & S \rightarrow C \ : \ P_S, H_S, SA, RN \\ (\mathrm{B2}) & C \rightarrow S \ : \ SA', RN, \{\{KS_{CS}\}_{P_S}\}_{H_S} \\ (\mathrm{B3}) & S \rightarrow C \ : \ \{ack_1\}_{KS_{CS}} \end{array} \qquad (2)$$

3.3.2 SSH2 のセッション鍵交換

SSH2 におけるセッション鍵交換は以下のとおりである：

$$\begin{array}{ll}(\mathrm{B1}) & S \leftrightarrow C \ : \ SA \\ (\mathrm{B2}) & C \rightarrow S \ : \ K_C \\ (\mathrm{B3}) & S \rightarrow C \ : \ K_S,\ P_S,\ \{H(V_C, V_S, \\ & SA_C, SA_S, P_S, K_C, K_S, KS_{CS})\}_{P_S^{-1}} \\ (\mathrm{B4}) & C \leftrightarrow S \ : \ ack \end{array} \tag{3}$$

3.4 ユーザ認証

セッション鍵交換の時点で，S と C は KS_{CS} を共有しているため，この鍵を用いてユーザ認証を行う．

3.4.1 SSH1 のパスワードユーザ認証

セッション鍵交換 (2) の後に行われるパスワードユーザ認証は以下のとおりである：

$$\begin{array}{ll}(\mathrm{C1}) & C \rightarrow S \ : \ \{uname\}_{KS_{CS}} \\ (\mathrm{C2}) & S \rightarrow C \ : \ \{ack_1\}_{KS_{CS}} \\ (\mathrm{C3}) & C \rightarrow S \ : \ \{password\}_{KS_{CS}} \\ (\mathrm{C4}) & S \rightarrow C \ : \ \{ack_2\}_{KS_{CS}} \end{array} \tag{4}$$

3.4.2 SSH1 の公開鍵ユーザ認証

セッション鍵交換 (2) の後に行われる公開鍵ユーザ認証は以下のとおりである：

$$\begin{array}{ll}(\mathrm{C1}) & C \rightarrow S \ : \ \{uname\}_{KS_{CS}} \\ (\mathrm{C2}) & S \rightarrow C \ : \ \{ack_1\}_{KS_{CS}} \\ (\mathrm{C3}) & C \rightarrow S \ : \ \{modulus\}_{KS_{CS}} \\ (\mathrm{C4}) & S \rightarrow C \ : \ \{\{challenge\}_{P_C}\}_{KS_{CS}} \\ (\mathrm{C5}) & C \rightarrow S \ : \ \{H(challenge, S_{CS})\}_{KS_{CS}} \\ (\mathrm{C6}) & S \rightarrow C \ : \ \{ack_2\}_{KS_{CS}} \end{array} \tag{5}$$

3.4.3 SSH2 のパスワードユーザ認証

セッション鍵交換 (3) の後に行われるパスワードユーザ認証は以下のとおりである：

$$\begin{array}{ll}(\mathrm{C1}) & C \rightarrow S \ : \ \{uname, password\}_{KS_{CS}} \\ (\mathrm{C2}) & S \rightarrow C \ : \ \{ack\}_{KS_{CS}} \end{array} \tag{6}$$

3.4.4 SSH2 の公開鍵ユーザ認証

セッション鍵交換 (3) の後に行われる公開鍵ユーザ認証は以下のとおりである：

$$\begin{array}{ll}(\mathrm{C1}) & C \rightarrow S \ : \ \{uname, P_C, \\ & \{uname, P_C, S_{CS}\}_{P_C^{-1}}\}_{KS_{CS}} \\ (\mathrm{C2}) & S \rightarrow C \ : \ \{ack\}_{KS_{CS}} \end{array} \tag{7}$$

4. 2 つのタイプの Rollback 攻撃

本章では，セッションハイジャック Rollback 攻撃とミスリード Rollback 攻撃の 2 つのタイプの Rollback 攻撃について概説する．

4.1 セッションハイジャック Rollback 攻撃

Rollback 攻撃が行われていない状況下では，S がサポートするプロトコルバージョン，暗号スイートおよびユーザ認証方式が S から C に提示され，C は，SSH2，最も安全とされる暗号スイートと公開鍵ユーザ認証を選択したことを示す返信を行う．

セッションハイジャック Rollback 攻撃は，C が S に接続した際に，何かしらの理由で，たとえば，DNS（Domain Name System）サーバが返す情報を書き換えることにより，SSH のセッションが $\mathcal{I}$ によってハイジャック（これをセッションハイジャックと呼ぶ）された後，SSH のセッションが $\mathcal{I}$ を経由しているにもかかわらず，C には S と接続しているように見えており，S には C から接続していると見えている．そのうえで，意図しないプロトコルバージョン，暗号スイートおよびユーザ認証方式を強制する攻撃である．つまり，S がサポートするプロトコルバージョン，暗号スイートおよびユーザ認証方式に関する通報を $\mathcal{I}$ が都合のよいものに改竄し，C に送信することにより，C はその通報を S が送信したと判断する攻撃である（**図 1**）．この攻撃は，dsniff（後述）というツールの一部の機能を用いてセッションをハイジャックし，実装した攻撃プログラムで成立することを確認した．

4.2 ミスリード Rollback 攻撃

ミスリード Rollback 攻撃は，何かしらの理由で，C が自らの意思で $\mathcal{I}$ に接続した際，$\mathcal{I}$ が S と C との間に介在し，意図しないプロトコルバージョン，暗号スイートおよびユーザ認証方式を強制する攻撃である．すなわち，ミスリード Rollback 攻撃が行われた際，C は $\mathcal{I}$ へ接続する意思があり，$\mathcal{I}$ へ接続している．そ

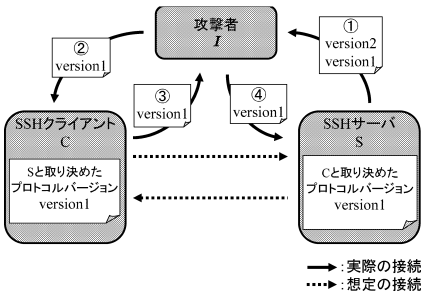

図 1 セッションハイジャック Rollback 攻撃の概念図
Fig. 1 Overview of Session-Hijack Rollback attack.

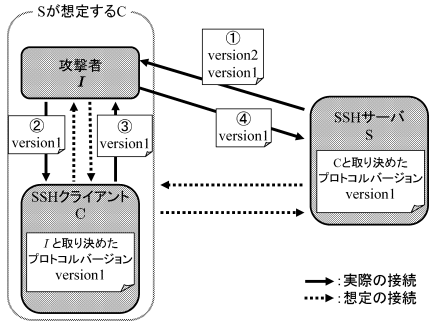

図 2 ミスリード Rollback 攻撃の概念図
Fig. 2 Overview of Misleading Rollback attack.

の後，C から接続された $\mathcal{I}$ は S へと接続を行うが，S は C から接続されたと判断してしまう．そのうえで，S が提示したプロトコルバージョン，暗号スイートおよびユーザ認証方式に関する通報を $\mathcal{I}$ が都合のよいものに改竄したうえで C に送信し，S がそれに対する返信を受け取った際，返信を行ったのが C であると S に判断させる攻撃である（**図 2**）．本攻撃はユーザが接続しなければ成立しないため，$\mathcal{I}$ は，ユーザが接続する環境を整え，ユーザからの接続を待ち受けながら，$\mathcal{I}$ の意図に合わせて改竄を行う攻撃プログラムを実行しておく．

5. セッションハイジャック Rollback 攻撃

5.1 攻撃成立のための運用前提

以下を運用の前提とする：

- C と S が，SSH1 と SSH2 の両方を利用でき，SSH2 の利用を優先する．
- C と S が，2 種類以上の暗号化アルゴリズムを利用でき，より安全な暗号化アルゴリズムの利用を優先する．
- C と S が，パスワードユーザ認証と公開鍵ユーザ認証が利用でき，公開鍵ユーザ認証の利用を優先する．

5.2 攻撃者の目的と想定される損害

$\mathcal{I}$ は，最終的に以下の 3 つの状況に至ることを目的とする：

- SSH1 の利用
- パスワードユーザ認証の利用
- 任意の暗号化アルゴリズムの利用

たとえば，3 番目の項目を含めて☆，これらの強制による直接的な損害はないが，管理者と利用者の運用ポリシを変更されるという問題が起こる．また，最終的に，以下の損害を被る可能性が高まり，この種の強制は問題視されている[12),14)]．

- C と S との間の通信が解読・改竄される．
- その結果，パスワードが奪取され，成りすまされる．

5.3 攻撃過程

攻撃の過程は以下のとおりである：

$$
\begin{array}{lll}
\text{(A1)} & S \to \mathcal{I}(C) & : v1+2 \\
\text{(A1}') & \mathcal{I}(S) \to C & : v1 \\
\text{(A2)} & C \to \mathcal{I}(S) & : v1 \\
\text{(A2}') & \mathcal{I}(C) \to S & : v1 \\
\text{(B1)} & S \to \mathcal{I}(C) & : P_S, H_S, SA, RN \\
\text{(B1}') & \mathcal{I}(S) \to C & : P_S, H_S, SA', RN \\
\text{(B2)} & C \to \mathcal{I}(S) & : SA'', RN, \{\{KS_{CS}\}_{P_S}\}_{H_S} \\
\text{(B2}') & \mathcal{I}(C) \to S & : SA'', RN, \{\{KS_{CS}\}_{P_S}\}_{H_S} \\
\text{(B3)} & S \to \mathcal{I}(C) & : \{ack_1\}_{KS_{CS}} \\
\text{(B3}') & \mathcal{I}(S) \to C & : \{ack_1\}_{KS_{CS}}
\end{array}
\tag{8}
$$

プロトコルバージョン交換を行うために，S は，$v1+2$ を $\mathcal{I}$ に送信する (A1)．$\mathcal{I}(S)$ は，S が SSH1 のみをサポートしているように $v1$ に書き換えて C に送信する (A1′)．これを受信した C は SSH1 を選択し，$v1$ を送信する (A2)．$\mathcal{I}(C)$ は，C から受け取った $v1$ を S に送信する (A2′)．これを受信した S は，C が SSH1 を選択したと判断する．

次に，SSH1 のセッション鍵交換を行うために，S は，2 種類以上の暗号化アルゴリズムが利用でき，パスワードユーザ認証と公開鍵ユーザ認証が利用できることを示した SA を含む通報を送信する (B1)．$\mathcal{I}(C)$ は，都合のよい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する SA' に書き換え，C に送信する (B1′)．これを受信した C は，SA' にある暗号化アルゴリズムとパスワードユーザ認証を選択し，これらに合意したことを示す SA'' を送信する (B2)．$\mathcal{I}(C)$ は S にそのまま送信する (B2′)．これを受信した S は，$\mathcal{I}$ が強制した暗号化アルゴリズムとパスワードユーザ認証を C が選択したと判断する．この後，ユーザ認証において，セッション鍵交換において合意した SSH1 のパスワードユーザ認証を行う．

以上のとおり，$\mathcal{I}$ は 5.2 節で示した目的を達成する．

6. ミスリード Rollback 攻撃

ミスリード Rollback 攻撃は，SSH1 または SSH2

☆ たとえば，下位互換のために組み込まれている DES は鍵長が短いために解読されやすく，DES Challenge III（1999 年 1 月）において，22 時間 15 分で解読されている．

のどちらのプロトコルバージョンでも実現する．そのため，本章では，ミスリード Rollback 攻撃が成立する運用前提と本攻撃に対する攻撃者の目的を述べた後，SSH1 と SSH2 へのミスリード Rollback 攻撃の過程をそれぞれ示す．

6.1 攻撃成立のための運用前提

以下を運用の前提とする：

- ユーザが正規のサーバのふりをする攻撃者に接続する環境を整える．
- C と S が，公開鍵ユーザ認証とパスワードユーザ認証を利用でき，公開鍵ユーザ認証の利用を優先する．

6.2 攻撃者の目的と想定される損害

$\mathcal{I}$ は以下の 5 つを行うことが目的である：

- パスワードユーザ認証の利用
- パスワードの奪取
- （C に検知されないように）S の計算機資源（S が保持する C のデータを指す）の利用
- 任意のプロトコルバージョンの利用
- 任意の暗号化アルゴリズムの利用

ただし，最後の 2 つは，直接的な損害はなく，SSH1 と SSH2 を共存した運用や複数の暗号化アルゴリズムの利用を許す際に可能となる．

これらの目的を達成した場合，以下の損害をもたらす可能性がある：

- S に対して $\mathcal{I}$ は C に成りすます．
- $\mathcal{I}$ に S の計算機資源を提供してしまう．

6.3 攻 撃 過 程

6.3.1 SSH1 への攻撃

ここでは，SSH1 へのミスリード Rollback 攻撃を説明する．

$\mathcal{I}$ は，C に正規のサーバであると信じ込ませ，$P_{\mathcal{I}}$ を受け入れさせる．その後，C は $\mathcal{I}$ を正規のサーバと判断しているため，$\mathcal{I}$ に接続しているという認識を持ったうえで，$\mathcal{I}$ に自ら接続する．この点に関する議論は後述する．接続後，C と $\mathcal{I}$，$\mathcal{I}(C)$ と S のそれぞれの間で，プロトコルバージョンを交換し，セッション鍵交換へ移行する：

セッション鍵交換

$$
\begin{array}{lll}
\text{(B1)} & S \to \mathcal{I}(C) & : P_S, H_S, SA, RN \\
\text{(B1}') & \mathcal{I} \to C & : P_{\mathcal{I}}, H_{\mathcal{I}}, SA', RN \\
\text{(B2)} & C \to \mathcal{I} & : SA'', RN, \{\{KS_{C\mathcal{I}}\}_{P_{\mathcal{I}}}\}_{H_{\mathcal{I}}} \\
\text{(B2}') & \mathcal{I}(C) \to S & : SA'', RN, \{\{KS_{\mathcal{I}S}\}_{P_S}\}_{H_S} \\
\text{(B3)} & S \to \mathcal{I}(C) & : \{ack_1\}_{KS_{\mathcal{I}S}} \\
\text{(B3}') & \mathcal{I} \to C & : \{ack_1\}_{KS_{C\mathcal{I}}}
\end{array}
\tag{9}
$$

S は $\mathcal{I}(C)$ にセッション鍵交換における最初の通報を送信する (B1)．$\mathcal{I}$ 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利用に制限した SA' に書き換えた通報を C に送信する (B1′)．これを受信した C は，$\mathcal{I}$ が強制したパスワード認証に合意したことを示す SA'' を生成し，これを含んだ通報を送信する (B2)．これを受け取った $\mathcal{I}$ は，SA'' を S に送信する (B2′)．この後，S は $KS_{\mathcal{I}S}$ を共有したことを示す通報を送信する (B3)．$\mathcal{I}$ 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 (B3′)．

したがって，$\mathcal{I}$ は 6.2 節で示した「パスワードユーザ認証の利用」を達成する．

また，$KS_{C\mathcal{I}}$ を C と $\mathcal{I}$ との間で共有し，同時に，$KS_{\mathcal{I}S}$ を $\mathcal{I}$ と S との間で共有するため，「C と $\mathcal{I}$ の間」と「(C のふりをした) $\mathcal{I}$ と S の間」はそれぞれ異なるセッションであることに注意されたい．

ユーザ認証

$$
\begin{array}{lll}
\text{(C1)} & C \to \mathcal{I} & : \{uname_C\}_{KS_{C\mathcal{I}}} \\
\text{(C1}') & \mathcal{I}(C) \to S & : \{uname_C\}_{KS_{\mathcal{I}S}} \\
\text{(C2)} & S \to \mathcal{I}(C) & : \{ack_1\}_{KS_{\mathcal{I}S}} \\
\text{(C2}') & \mathcal{I} \to C & : \{ack_1\}_{KS_{C\mathcal{I}}} \\
\text{(C3)} & C \to \mathcal{I} & : \{password_C\}_{KS_{C\mathcal{I}}} \\
\text{(C3}') & \mathcal{I}(C) \to S & : \{password_C\}_{KS_{\mathcal{I}S}} \\
\text{(C4)} & S \to \mathcal{I}(C) & : \{ack_2\}_{KS_{\mathcal{I}S}} \\
\text{(C4}') & \mathcal{I} \to C & : \{ack_2\}_{KS_{C\mathcal{I}}}
\end{array}
\tag{10}
$$

C が $\mathcal{I}$ に $uname_C$ を送信する (C1)．$\mathcal{I}$ は C に成りすますために，$uname_C$ を送信する (C1′)．これを受信した S は，自身のデータベースに $uname_C$ が存在するかどうか確認し，次の通報を促すために ack_1 を送信する (C2)．$\mathcal{I}$ は S から送信された $ack1$ を C に送信する (C2′)．この後，C は $password_C$ を $\mathcal{I}$ に送信する (C3)．これを受信した $\mathcal{I}$ は $KS_{C\mathcal{I}}$ で復号することにより $password_C$ を得る．ここで，$\mathcal{I}$ は通常の SSH サーバに成りすますために，$password_C$ を S に送信する (C3′)．これを受信した S は接続者が $\mathcal{I}$ にもかかわらず，C であると判断する．最後に，S は ack_2 を $\mathcal{I}$ に送信し (C4)，$\mathcal{I}$ は同様にこれを C に送信する (C4′)．

以上のとおり，$\mathcal{I}$ は 6.2 節で示した「パスワードの奪取」と「S の計算機資源の利用」を達成する．

6.3.2 SSH2 への攻撃

ここでは，SSH2 へのミスリード Rollback 攻撃について示す．

SSH1 のミスリード Rollback 攻撃と同様に，C は

$\mathcal{I}$ を正規のサーバと判断しているため，$\mathcal{I}$ に接続しているという認識を持ったうえで，$P_{\mathcal{I}}$ を用いて $\mathcal{I}$ に自ら接続する．この点に関する議論は後述する．接続後，C と $\mathcal{I}$，$\mathcal{I}(C)$ と S のそれぞれの間で，プロトコルバージョンを交換し，セッション鍵交換へ移行する：

セッション鍵交換

$$
\begin{array}{lll}
\text{(B1)} & S \leftrightarrow \mathcal{I}(C) & : SA \\
\text{(B1}') & \mathcal{I} \leftrightarrow C & : SA' \\
\text{(B2)} & C \rightarrow \mathcal{I} & : K_C \\
\text{(B2}') & \mathcal{I}(C) \rightarrow S & : K_{\mathcal{I}} \\
\text{(B3)} & S \rightarrow \mathcal{I}(C) & : K_S, P_S, \\
 & & \{H(V_{\mathcal{I}}, V_S, SA_{\mathcal{I}}, SA_S, P_S, \\
 & & K_{\mathcal{I}}, K_S, KS_{\mathcal{IS}})\}_{P_S^{-1}} \\
\text{(B3}') & \mathcal{I} \rightarrow C & : K_{\mathcal{I}}, P_{\mathcal{I}}, \\
 & & \{H(V_C, V_{\mathcal{I}}, SA_C, SA_{\mathcal{I}}, \\
 & & P_{\mathcal{I}}, K_C, K_{\mathcal{I}}, KS_{C\mathcal{I}})\}_{P_{\mathcal{I}}^{-1}} \\
\text{(B4)} & C \rightarrow \mathcal{I} & : ack \\
\text{(B4}') & \mathcal{I}(C) \rightarrow S & : ack
\end{array}
\tag{11}
$$

セッション鍵交換では，「C と $\mathcal{I}$ の間」と「(C のふりをした) $\mathcal{I}$ と S の間」はそれぞれ異なるセッションを張り，$KS_{C\mathcal{I}}$ を C と $\mathcal{I}$ との間で共有し，$KS_{\mathcal{IS}}$ を $\mathcal{I}$ と S との間で共有する．

ユーザ認証

$$
\begin{array}{lll}
\text{(C1)} & C \rightarrow \mathcal{I} & : \{uname_C, P_C, \\
 & & \{uname, P_C, S_{C\mathcal{I}}\}_{P_C^{-1}}\}_{KS_{C\mathcal{I}}} \\
\text{(C1}') & \mathcal{I} \rightarrow C & : \{ack_1\}_{KS_{C\mathcal{I}}} \\
\text{(C2)} & C \rightarrow \mathcal{I} & : \\
 & & \{uname_C, password_C\}_{KS_{C\mathcal{I}}} \\
\text{(C2}') & \mathcal{I}(C) \rightarrow S & : \\
 & & \{uname_C, password_C\}_{KS_{\mathcal{IS}}} \\
\text{(C3)} & S \rightarrow \mathcal{I}(C) & : \{ack_2\}_{KS_{\mathcal{IS}}} \\
\text{(C3}') & \mathcal{I} \rightarrow C & : \{ack_2\}_{KS_{C\mathcal{I}}}
\end{array}
\tag{12}
$$

C は，公開鍵ユーザ認証を要求するため，その旨の通報を正規のサーバに成りすました $\mathcal{I}$ に送信する (C1)．これを受け取った $\mathcal{I}$ は，パスワードユーザ認証を要求する通報 ack_1 を C に送信する (C1′)．これを受け，C は，$uname_C$ と $password_C$ を正規のサーバに成りすました攻撃者 $\mathcal{I}$ に送信する (C2)．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．次に，C に成りすました $\mathcal{I}(C)$ は $password_C$ を S に送信する (C2′)．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 (C3)．同様に，$\mathcal{I}$ もその通報を C に送信する (C3′)．

以上のとおり，$\mathcal{I}$ は 6.2 節で示した目的を達成する．

7. 議　　論

7.1 Rollback 攻撃の実現性

本論文で示した 2 つのタイプの Rollback 攻撃を実現するうえで 2 つの論点がある．これらを以下で議論する．

7.1.1 ユーザの判断根拠

2 つのタイプの Rollback 攻撃において，パスワードユーザ認証が強制された場合に，SSH クライアントを操作するユーザは，公開鍵ユーザ認証ではないので検知できる．しかしながら，パスワードユーザ認証への変更が，何かしらの理由により，公開鍵ユーザ認証が利用できなくなったと判断し，パスワードユーザ認証を利用するユーザがいないとはいえない．逆に，パスワードユーザ認証と公開鍵ユーザ認証の両方を利用できる運用を行っているので，ユーザが「パスワードユーザ認証を利用する」と判断することは問題ではない．

プロトコルバージョンや暗号化アルゴリズムが意図しないものに強制された場合には，そのことすらユーザは知る術がない．また，これに対する対策として，取り決められたものを表示することが考えられるが，すべてのユーザが表示内容に注意を払うことはない．

以上のことから，SSH プロトコルは，一般的なユーザが 2 つのタイプの Rollback 攻撃が行われていることを検知する判断根拠に乏しい．

7.1.2 ミスリード攻撃に対する SSH パスワードユーザ認証の問題点

ミスリード Rollback 攻撃に対する SSH パスワードユーザ認証の問題点は 2 つある．

1 つは，SSH パスワードユーザ認証はプロトコルとして欠陥があることである．このことは，公開鍵ユーザ認証がこの攻撃を防ぐことができることからも明らかである．

もう 1 つは，ユーザを攻撃者へ誘導する方法が将来的に発見されないことが否めないことである．ミスリード Rollback 攻撃を実現するためには，C を正規のサーバのふりをしている攻撃者へ接続させる必要がある．これを奇妙であると思うかもしれないが，たとえば，「(攻撃者のサーバが) これまで接続してきたサーバと同じ計算資源を提供する新たなサーバである」として告知された場合，それを正規のサーバと信じ，接続するユーザが存在する可能性がないとはいえ

ない．

7.2 Rollback 攻撃への対策

7.2.1 項，7.2.2 項の対策により，5.2 節，6.2 節に示した想定される損害を回避できる．

7.2.1 プロトコルの変更による技術対策

SSH プロトコルの変更による Rollback 攻撃への対策を示す．

7.2.1.1 セッションハイジャック攻撃への対策

Rollback 攻撃の対策として，C と S が提案する暗号スイートが意図したとおりであることを保証する通報認証を用いた対策は，SSL，IPsec，SSH2 で用いられている．SSH2 では，3.3.2 項のプロトコル (3) の (B3) において，S が送受信した SA は（通報認証を実現するための）電子署名が付加されているため，これを受信した C は，自身が送受信した SA と異なっているかどうかを確認できる．そのため，SSH2 では，セッションハイジャック Rollback 攻撃を防ぐことができる．よって，SSH1 でも通報認証を用いた対策を行えばよい．たとえば，7.2.1.2 の SSH1 への対策におけるプロトコル (14) のようにすればよい．これにより，$\mathcal{I}$ がプロトコルバージョン，暗号化アルゴリズムとユーザ認証方式のいずれか 1 つ以上の強制を行った場合，$password$ を保持していない $\mathcal{I}$ は (P3) の通報：

$$\{\{N_C, C, S, KS_{CS}, V_{\mathcal{I}}, V_S, SA_{\mathcal{I}}, SA_S\}_{password}\}_{KS_{CS}}$$

を作成できないため，この攻撃を回避できる．

7.2.1.2 ミスリード攻撃への対策

ミスリード Rollback 攻撃は「C と $\mathcal{I}$ の間」と「(C のふりをした) $\mathcal{I}$ と S の間」は，それぞれ異なるセッションであるため通報認証を用いた対策では防ぐことができない．よって，パスワードユーザ認証において，攻撃を防止するために，パスワードとセッションユニークな情報から生成されるハッシュ値を認証のための生成情報とするチャレンジ・レスポンスを行いながら，意図したとおりにユーザ認証方式が取り決められたことを確認すればよい．以下にプロトコルの変更例を示す：

SSH2 への対策

SSH2 では，セッション鍵，これを生成するための情報および SA によってセッションを識別することができるため，パスワード認証を以下のようにすればよい．また，N_C は C のノンスとする：

$$\begin{aligned}\text{(P1)}\ C \to S\ :\ &\{\{uname, N_C, C, S, KS_{CS}, V_C, V_S, SA_C, SA_S\}_{password}\}_{KS_{CS}}\\ \text{(P2)}\ S \to C\ :\ &\{h(N_C, S, C, KS_{CS}, V_C, V_S, SA_S, SA_C), ack\}_{KS_{CS}}\end{aligned} \tag{13}$$

これにより，$\mathcal{I}$ がパスワードユーザ認証を強制した場合，$password$ を保持していない $\mathcal{I}$ は S に送信する (P1) の通報：

$$\{\{uname, N_C, C, S, KS_{\mathcal{I}S}, V_{\mathcal{I}}, V_S, SA_{\mathcal{I}}, SA_S\}_{password}\}_{KS_{\mathcal{I}S}}$$

を作成できないため，この攻撃を回避できる．

SSH1 への対策

SSH1 では，SA，S と C によってセッションを識別することができるため，パスワード認証を以下のようにすればよい．また，SSH2 への対策と同様に，N_C は C のノンスとする：

$$\begin{aligned}\text{(P3)}\ C \to S\ :\ &\{\{N_C, C, S, KS_{CS}, V_C, V_S, SA_C, SA_S\}_{password}\}_{KS_{CS}}\\ \text{(P4)}\ S \to C\ :\ &\{h(N_C, S, C, KS_{CS}, V_C, V_S, SA_S, SA_C), ack_2\}_{KS_{CS}}\end{aligned} \tag{14}$$

これにより，SSH2 の対策と同様に，$password$ を保持していない $\mathcal{I}$ は (P3) の通報：

$$\{\{N_C, C, S, KS_{\mathcal{I}S}, V_{\mathcal{I}}, V_S, SA_{\mathcal{I}}, SA_S\}_{password}\}_{KS_{CS}}$$

を作成できないため，この攻撃を回避できる．

しかしながら，これらの対策を行うためには，S が平文のパスワードを保持しなければならず，また，プロトコル自体の変更もあるため容易ではない．

7.2.2 運用上の対策

2 つのタイプの Rollback 攻撃への比較的容易な対策は，利用できるプロトコルバージョン，暗号スイートとユーザ認証方式を制限した運用を行うことである．

ここで，どのような運用を行っているとどの Rollback 攻撃が成立するかを**表 1** にまとめる：

セッションハイジャック Rollback 攻撃を防ぐには，SSH2 のみを利用するように限定すればよい．また，

表 1 プロトコルバージョンと Rollback 攻撃の関係
Table 1 Correlativity between Protocol Version and Rollback attack.

	セッションハイジャック攻撃	ミスリード攻撃
SSH1	△	○
SSH2	×	○
SSH1，2	○	○

○：成立，×：不成立
△：暗号化アルゴリズム，ユーザ認証方式のみ強制可能
SSH1，2：SSH1 と SSH2 を共存した運用

ミスリード Rollback 攻撃を防ぐには，SSH2 の利用に限定し，不要な暗号化アルゴリズムの利用をやめ，公開鍵ユーザ認証のみに限定すればよい．

7.3 既存の MITM 攻撃との違い

SSH1 に対して，SSH のセッションをハイジャックして，MITM（Man-In-The-Middle）攻撃を実現させ，パスワードを奪取する dsniff[15] というツールがある．このツールを用いて，C は S に接続したいにもかかわらず，DNS サーバが返す情報を書き換え，C と S との SSH のセッションをハイジャックし，攻撃者自身のホスト鍵 $P_{\mathcal{I}}$ を C へ送り込むことができる．しかし，この MITM 攻撃を実現するためには，送り込まれた $P_{\mathcal{I}}$ をユーザが受け取った際，ユーザの端末上に警告が表示されるにもかかわらず，$P_{\mathcal{I}}$ を受け入れなければならない．つまり，C が受け取った $P_{\mathcal{I}}$ を S のホスト鍵として受け入れるという「公開鍵のすり替え」を利用して攻撃を実現する．そのため，この攻撃を防ぐためには，「接続したい相手」と「ホスト鍵」の対応を保証する仕組みを導入すればよい．たとえば，PKI（Public Key Infrastructure）を用いて防ぐことができる．この攻撃は，すでに知られており☆，本論文で扱っている攻撃とは異なることに注意されたい．

一方，セッションハイジャック Rollback 攻撃は，C が S に接続しようとして，P_S を受け取っており，ミスリード Rollback 攻撃は，C が $\mathcal{I}$ に接続しようとして，$P_{\mathcal{I}}$ を受け取る．このように，公開鍵のすり替えを前提とはしておらず，既存の MITM 攻撃とは異なることが分かる．この違いを含め，攻撃モデルの比較が文献 16) に書かれている．この文献でいうレスポンダを騙す攻撃がミスリード Rollback 攻撃となっている．

8. ま と め

本論文では，セッションハイジャック Rollback 攻撃とミスリード Rollback 攻撃という 2 つのタイプの Rollback 攻撃を示し，これらの攻撃の実現性について述べた．さらに，これらの Rollback 攻撃を防ぐための対策として，プロトコルの変更による技術対策と運用上の対策を示した．

謝辞 編集委員ならびに査読委員の有益な御指摘に感謝申し上げます．

☆ 実際，SSH の実装においても警告が表示される．すなわち，SSH では，接続したい相手とホスト鍵の正しい対応を保持した運用を前提としている．

参 考 文 献

1) Barrett, D.J. and Silverman, R.E.: *SSH, The Secure Shell*, O'REILLY ISBN:0-596-00011-1, (Feb. 2001).
2) Ylonen, T.: The SSH (Secure SHell) Remote Login Protocol, Internet Draft, Network Working Group (Nov. 1995).
3) Ylonen, T., Kivinen, T., Saarinen, M., Rinne, T. and Lehtinen, S.: SSH Protocol Architecture draft-ietf-secsh-architecture-22.txt, Internet Draft, Network Working Group (Mar. 2005).
4) Ylonen, T. and Lonvick, C.: SSH Transport Layer Protocol draft-ietf-secsh-transport-24.txt, Internet Draft, Network Working Group (Mar. 2005).
5) Ylonen, T. and Lonvick, C.: SSH Connection Protocol draft-ietf-secsh-connect-25.txt, Internet Draft, Network Working Group (Mar. 2005).
6) Ylonen, T. and Lonvick, C.: SSH Authetication Protocol dragt-ietf-secsh-userauth-27.txt, Internet Draft, Network Working Group (Mar. 2005).
7) Freier, A.O., Karlton, P. and Kocher, P.C.: The SSL Protocol Version 3.0, draft-freier-ssl-version3-02.txt, INTERNET-DRAFT, Transport Layer Security Working Group (Nov. 1996).
8) Dierks, T. and Allen, C.: The TLS Protocol, RFC 2246, Network Working Group (Jan. 1999).
9) Kent, S. and Atkinson, R.: Security Architecture for the Internet Protocol, RFC 2401, Network Working Group (Nov. 1998).
10) http://www.openssh.org/usage/ja/index.html
11) http://www.openssh.com/index.html
12) Wagner, D. and Schneier, B.: Analysis of the SSL 3.0 Protocol, *Proc. 2nd USENIX Workshop on Electronic Commerce*, USENIX Press, pp.29–40 (1996).
13) Password Based Encryption Standard, PKCS #5, RSA Laboratories (Nov. 1993).
14) Eric Rescorla，齋藤孝道，鬼頭利之，古森 貞：マスタリング TCP/IP SSL/TLS 編，オーム社，ISBN: 4274065421 (2003/11).
15) http://www.monkey.org/~dugsong/dsniff
16) 齋藤孝道：攻撃シナリオを用いた認証プロトコルの安全性検証器の実装，SCIS 予稿集，pp.1495–1500 (2005).

(平成 17 年 7 月 8 日受付)
(平成 18 年 2 月 1 日採録)

鬼頭 利之（正会員）

2003 年東京理科大学大学院理工学研究科情報科学専攻修士課程修了．同年（株）東芝入社．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所属．ネットワークセキュリティ，システムセキュリティの研究開発に従事．

齋藤 孝道（正会員）

1998 年東京理科大学大学院理工学研究科情報科学専攻修士課程修了．現在，明治大学理工学部情報科学科助教授．博士（工学）．