# SSH - 公開鍵暗号によるセキュリティ

ytoku (Tokushige Yuuki)

ytoku@mma.club.uec.ac.jp

予定 :

- SSH とは何か
- SSH の使い方
- 公開鍵による認証

# 自己紹介

- 徳重佑樹（ytoku)
  - 電気通信学部　情報工学科３年
- 興味分野
  - プログラミング
  - セキュリティ
- 脆弱性を探すのが趣味
  - PoC for CVE-2011-0536
  - CVE-2010-0667, CVE-2010-0668, CVE-2010-0669

# SSH とは何か

- Secure SHell の略
- 他の計算機に安全にログインして操作するためのプロトコル・ソフトウェア
- 歴史的には rsh・telnet が使われてきた
  - これらはメッセージが平文で流れ、盗聴・改竄・なりすましなどに脆弱
- 暗号学的に安全なプロトコルが求められた!

プロトコル = 通信手順。機械が通信をするための約束事。

# telnet への攻撃例

- サーバになりすまして、ログインしてきた人のログインパスワードを詐取
  - 攻撃手段は DNS キャッシュポイズニングや ARP スプーフィングなどいろいろ
- パケットスニッフィングして盗聴とかも

# SSH の歴史 (1)

- 初期バージョンは 1995 年にヘルシンキ工科大学で開発された。
  - 初期のバージョンの ssh はフリーソフトウェアであったが、後に ssh は商品化されソースが公開されなくなった
- オープンソース実装 OpenSSH
  - OpenBSD プロジェクトがフリーソフトウェアであった頃のオリジナル ssh を元に開発した。現在の主流。
- Windows 向けクライアント実装としては putty など

フリーソフトウェア ＝
改造・再配布の自由 (free) が保障されているソフトウェア。

# SSH の歴史 (2)

- オリジナルの SSH のプロトコルは SSH-1 と呼ばれる。
  - しかし、SSH-1 にはいくつもの脆弱性が発見されており使用は推奨されない。(VU#315308, VU#13877 など )
- 改良された新規格が 1996-1997 年に作られており SSH-2 と呼ばれる。SSH-1 とは互換性がない。
  - RFC に策定されたのはこちらのバージョンが元になっている。
- 参考文献 Daniel J. Barrett and Richard E. Silverman. ”SSH, The Secure Shell: The Definitive Guide”. O’Reilly, 2001.

# ssh コマンドの使い方

- ssh [--option] [user@]hostname [command]
  - 必要に応じてオプション
  - ユーザ名 @ ホスト名
    - ユーザ名 @ を省略するとローカルのログインユーザ名でログインする
  - 実行するコマンド ( 省略するとシェルが起動 )
- 設定などは ~/.ssh/ 以下に置かれる
- ~/.ssh/config に接続先別の設定が可能
- Windows からの場合は cygwin を使うなり putty を使うなり

# 公開鍵による認証

- SSH のログイン方式はいくつか用意されているが公開鍵による認証が基本
  - パスワード方式も使えることが多い
- 公開鍵認証のメリット
  - パスワードが違うサーバに対しても同一の鍵 + パスフレーズでログインできる
  - ssh-agent を使うとパスフレーズを入力するのは一度だけにできる
  - 特定のコマンドのみ実行を許可するなど、細かい設定が可能

# そもそも、公開鍵暗号とは？

- 一般的な暗号は大別して共通鍵暗合方式と公開鍵暗号方式に分けられる。
- 共通鍵暗号は暗号化と復号で同じ鍵を使う
  - 例 : DES, AES など
  - 問題点： 通信相手の数だけ鍵を作らないと安全にならない。また、どうやってその鍵を安全に送信するのか？
- 公開鍵暗号は暗号化と復号の鍵を別にしておく
  - 例 : RSA, ElGamal 暗号など
  - 暗号化鍵 (= 公開鍵 ) から復号鍵 (= 秘密鍵 ) を計算することができないことが条件
  - 暗号化鍵は公開してしまう

復号 = 暗号化された文を平文に戻すこと

# 公開鍵認証

- 公開鍵暗号のアイデアを応用すると、逆に誰が作ったかを検証するデジタル署名にも使える。
- 代表格 : RSA, DSA
  - RSA 暗号は暗号化にも署名にも使える
  - DSA は署名専用
- アイデア： 通信の開始時に両者が共同で作ったその場限りのトークン ( セッション識別子 ) を作ってそれに署名する
- これを使ってサーバとクライアントが相互に認証してなりすましを防ぐ

# 挑戦問題

- Alice と Bob が通信しようとしている。Alice は通信相手の計算機 C が Bob のものであることを確認したい。
- Alice は Bob の公開鍵を知っており、Alice の計算機 A と計算機 C の間の通信は盗聴も改竄もされないことが保証されている。
- Alice は自分が生成した予測不可能なその場限りの値を計算機 C に送信して、デジタル署名して送り返してもらえば計算機 C が Bob の計算機か確認できると考えた。
- これは正しいだろうか？計算機 C を所有する攻撃者 Mallory がその計算機を Bob のものと偽る余地を考えよ。

# ユーザ認証用の鍵の作り方

- ssh-keygen コマンドで生成できる
- $ ssh-keygen -t rsa -b 2048
  - -t　オプションは鍵の種類
  - -b　オプションは鍵長
- なお、コメントは完全にユーザが識別するためだけの情報で、計算機はこれを無視する
- デフォルトで鍵が作られる場所 (RSA の場合 )
  - 秘密鍵　~/.ssh/id_rsa
  - 公開鍵　~/.ssh/id_rsa.pub

# 鍵の種類：DSA vs RSA

- OpenSSH は RSA と DSA に対応している。どっちを選ぶといいのか
- DSA は規格上、鍵長が 1024bit 固定
  - よって、強度をあげられない
- RSA は特許があったため DSA が使われてきた経緯がある
  - 既に切れている ( 米 4,405,829 号 1983 年 -2000 年 )

RSA を使った方がいい

# RSA の鍵長

- 近い将来 1024bit の RSA 暗号が安全ではなくなる
  - 2009 年 12 月についに 768bit が解読された!
    - RSA Factoring Challenge: RSA-768
    - NTT を含む共同研究による
- 2010 年以降は 2048bit の鍵を使うべき
  - 米国の暗号標準の勧告による。

# 鍵のパスフレーズ

- 秘密鍵が盗まれたら大変
  - 例えば物理的な盗難とか
- それに備えて秘密鍵自体を暗号化しておこう
- 秘密鍵は「パスフレーズ」を元に暗号化される
- このパスフレーズはサーバに伝わることはない

パスフレーズ ＝ パスワードと大体同義。本来、単語を複数並べてパスワードよりも強度を高めたものをパスフレーズと呼ぶ。

# 公開鍵の登録

- 作った鍵ペアのうち公開鍵をサーバに登録する
  - 登録先： サーバの ~/.ssh/authorized_keys
- 公開鍵ファイルの中身はテキストファイル 1 行になっているので、それを authorized_keys ファイルに追加すればよい
- コピペするとか scp コマンドでアップロードするとか
- 公開鍵は見られて構わないが改竄されないような方法で登録しなければならない
- これでユーザの鍵を使って、サーバがクライアントを本物か確認できるようになった

# そのサーバは果たして本物か

- せっかくサーバがクライアントを確認しても、クライアントが偽者のサーバに接続してしまったら台無し
- サーバも公開鍵と秘密鍵の鍵ペアを持っている
- 始めて接続したサーバの場合、最初にサーバの公開鍵のフィンガープリントが表示される
- 安全な手段でサーバの鍵のフィンガープリントを入手しておき比較する必要がある

フィンガープリント = ( 鍵の ) 指紋。鍵から生成した比較的短い値で、同一のフィンガープリントになるような鍵を他に作るのは極めて難しい。

# Finger prints

- sun
  - RSA b9:e2:7e:61:27:cc:57:56:65:ee:c8:fc:da:59:21:53
  - DSA 6c:71:41:72:4c:b7:6d:89:25:85:39:3a:75:4e:12:0b
- nest
  - RSA 18:a3:3d:bd:5b:05:d3:e8:0b:bb:7d:4e:92:cf:8c:d1
  - DSA b3:88:12:0d:c5:d3:ef:08:4f:f9:ef:a6:b5:ca:6d:46

# 余談：フィンガープリント蒐集

- 既に接続したことのあるホストのフィンガープリントを得るには？
  - 答え ＄ ssh-keygen -l -f .ssh/known_hosts

# たのしい仲間 (1)

- scp
  - rcp の置き換え機能。少数のファイルをリモートから / リモートへコピーするのに適している。
  - scp filepath [user@]host:filepath
  - scp [user@]host:filepath filepath
- sftp
  - FTP を置き換えるべくして作られた SSH の機能
  - FTP に SSL を加えた ftps と名前が紛らわしい

# たのしい仲間 (2)

- ポートフォワーディング
  - TCP/IP 通信の”トンネル”を造る
  - ssh -L8888:proxy-east.uec.ac.jp:8080 sun
    - 自分の 8888 番に繋ぐと接続先 ( 大学内のサーバ ) から学内のプロキシサーバ (8080 番 ) に繋がる
- X11 フォワーディング
  - ポートフォワーディングの応用
  - ssh -X

# まとめ

- SSH で安全にリモートログイン
- SSH-1 と SSH-2 があるが SSH-1 は安全ではない
- 2048bit の RSA 鍵で公開鍵認証しよう
- 公開鍵認証を用いると秘密ではない情報を使って相手が本物かを確かめられる
- フィンガープリントは飾りじゃないのよ

# 挑戦問題答え

- 正しくない。計算機 C を持つ攻撃者 Mallory は、 Alice から値を受け取ったら Alice のふりをして真の Bob の計算機 B に値を渡し、Bob に署名させて Alice に返せばよい。( 中間者攻撃 )
- SSH-2 ではこれを防ぐために鍵交換で暗号鍵を作るときに「セッション識別子」( 暗号鍵のハッシュ値 ) を作ってそれに署名する。
  - 鍵交換によって作られるので片方が生成される識別子が特定のものになるように操作することは出来ない。
  - セッション識別子を操作することが出来ないので Alice のふりをして Bob にそのセッション識別子に署名させることが出来ない。