SSHでパスワードなしにログインする方法のメモ

2008年04月07日
SSHで公開鍵認証ができるレンタルサーバならこれをやっとく rsync でもパスワード聞かれないし ssh-agent 常駐させなくてもいいし。 メモなので間違いは当然あるかも。他の方のページも参照してくださいませ。

2014年12月24日
「もう少しコンパクトにまとめられるべ？」と言われたのでそういう方向で書き直した。

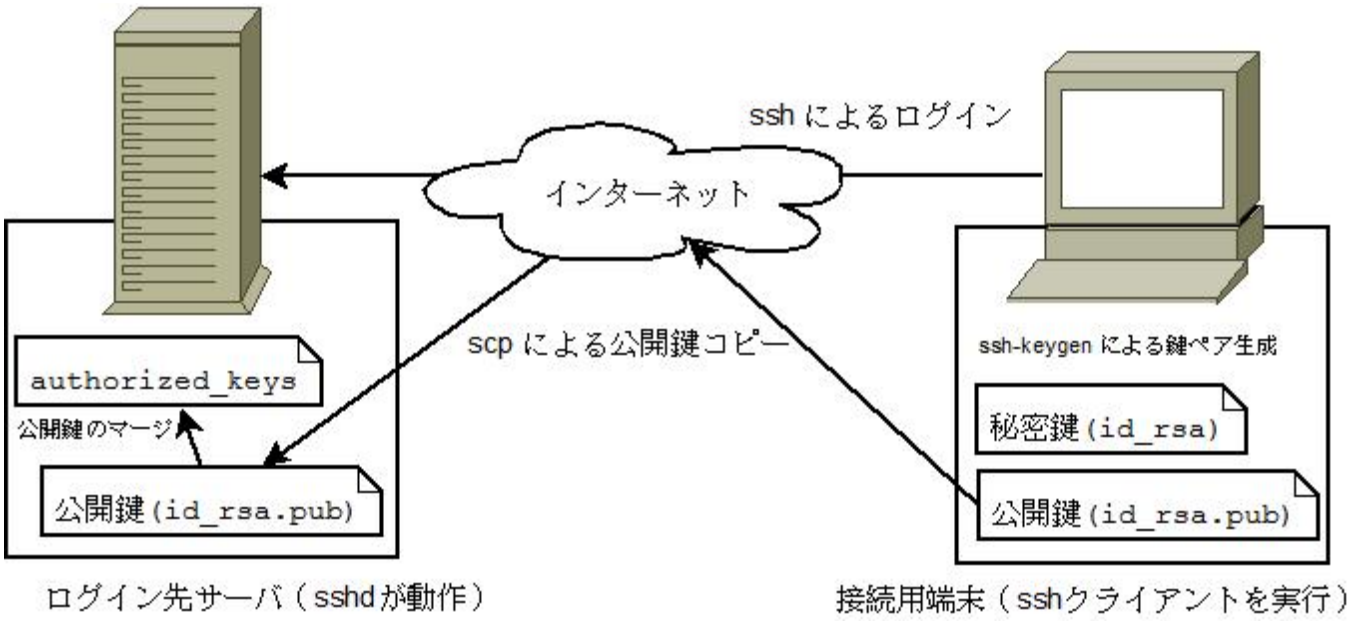
2016年01月09日
Bing Webマスターツールの指摘に従っていくつか修正。ついでに文面も修正。

前提

接続先のサーバにパスワード認証で SSH によるログインができること。 これができないと、ログイン先サーバに安全に公開鍵（後述）をコピーする手段が無い。レンタルサーバのコンソールを直接弄れるなら話は別だけど。

やりかた

ログイン先を amanda.hohehohe.jp 端末を amanda.hoge.jp とします。



ログイン先サーバへSSH ログイン可能なことを確認

SSHでログインできることを確認します。

```
$ ssh -l hogehohe amanda.hohehohe.jp
hogehohe@amanda.hohehohe.jp's password:□□□□□←パスワードを入力
```

```
Last login: Mon Mar 10 11:58:29 2008 from amanda.hoge.jp
[hogehohe@amanda.hohehohe.jp ~]$ exit
$
```

SSHでログインできるならログアウトして次。

公開鍵と秘密鍵のペアを作る

公開鍵と秘密鍵のペアを作ります。これは接続端末になる側のマシンで実施してください。 ちなみに筆者は FreeBSD 7.0 RELEASE のAMD64版を使ってます。

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hogehohe/.ssh/id_rsa):□□□←デフォルトでいいならそのままEnterキー
Created directory '/home/hogehohe/.ssh'.
Enter passphrase (empty for no passphrase):□□□□□←何も入力せずEnterキー
Enter same passphrase again:□□□□□□□□□□□□□□□←何も入力せずEnterキー
Your identification has been saved in /home/hogehohe/.ssh/id_rsa.
Your public key has been saved in /home/hogehohe/.ssh/id_rsa.pub.
The key fingerprint is:
b3:5f:2a:7f:cb:42:dd:ae:bb:cf:0e:ca:d0:97:ce:83 hogehohe@amanda.hoge.jp
$
```

ポイントはパスフレーズを入力しないこと。 パスフレーズ入れちゃうと、パスワードを入力しなくてもよい代わりにパスフレーズを入力しなきゃならなくなります。

公開鍵をログイン先のサーバに scp でコピーする

公開鍵をネットワークに公開してはいけません。取れる手段の中で一番安全な方法を使ってコピーします□ scp でコピーするなら 素の ftp よりは安全って程度の話です。ネットワークに公開鍵を晒した事に変わりはありません。

本当ならFDとかCD-Rとかの物理的手段でログイン先サーバに公開鍵を持っていく必要があります。

```
$ scp id_rsa.pub hogehohe@amanda.hohehohe.jp:/home/hogehohe/id_rsa.pub
hogehohe@amanda.hohehohe.jp's password:□□□□□←パスワードを入力
id_rsa.pub                    100%  411     0.4KB/s   00:00
$
```

なぜ公開鍵を公開してはいけないのか？
それは、公開鍵認証で接続しようとしているサーバが目的のログイン先サーバである事を保証できなくなってしまうためです。 認証自体は公開鍵暗号により安全にできるかもしれませんが、目的のログイン先サーバと接続できていなければ意味がありません。……偽サーバに安全な認証で接続したい訳ではないですよね？

もう少し詳しい説明は最後の「 注意 」をお読みください。

公開鍵のマージ

再度 SSH でログイン先にログインし、authorized_keys ファイルに公開鍵をマージします。

```
 $ ssh -l hogehohe amanda.hohehohe.jp
 hogehohe@amanda.hohehohe.jp's password:□□□□□←パスワードを入力
 Last login: Mon Mar 10 12:08:30 2008 from amanda.hoge.jp
 [hogehohe@amanda.hohehohe.jp ~]$ ls -l
 total 4
 -rw-r--r--  1 mynameis users  411 Mar 10 11:38 id_rsa.pub□□□□□←scpでコピーしたこのファイルを
 [hogehohe@amanda.hohehohe.jp ~]$ cd .ssh
 [hogehohe@amanda.hohehohe.jp .ssh]$ ls -l
 total 4
 -rw-------  1 hogehohe users 412 Jan 26 23:18 authorized_keys
 [hogehohe@amanda.hohehohe.jp .ssh]$ cat ../id_rsa.pub >> authorized_keys□□□←このファイルの最後に追加する
 [hogehohe@amanda.hohehohe.jp .ssh]$ exit
 $
```

場所は ~/.ssh にあると思うけど環境により違うので調べてください。

パスワードなしのログインを確認する

マージが済んだら早速試してみます。

```
 $ ssh -l hogehohe amanda.hohehohe.jp
 Last login: Sun Mar 10 12:10:33 2008 from amanda.hoge.jp
 [hogehohe@amanda.hohehohe.jp ~]$ ls -l
 total 4
 -rw-r--r--  1 mynameis users  411 Mar 10 11:38 id_rsa.pub
 [hogehohe@amanda.hohehohe.jp ~]$ rm id_rsa.pub
 [hogehohe@amanda.hohehohe.jp ~]$
```

パスワードを聞かれずにログインできました。

注意

くどいですが□SSHで公開鍵を使った公開鍵認証を利用する場合、クライアントの公開鍵を公開してはいけません□

『公開鍵なら漏れても(公開されても)問題ないよ』と言ってしまう人等がいます。
http://q.hatena.ne.jp/1326632540 の質問者とその回答者みたいなのが怖い。

SSHでの公開鍵は「 ログイン先（接続先）サーバに公開」するものです。 世界□The Internet□へ公開するのではありません。

公開鍵方式は、

- 公開鍵で暗号化して、秘密鍵で復号する。
- 公開鍵は公開しても構わない、秘密鍵は絶対ばれてはいけない
- 公開鍵で暗号化したものは、秘密鍵でしか復号できない
- 公開鍵で暗号化したものが盗まれても、復号するためにはえらい労力と時間が要る（様になっているはず）

という性質があります。
※秘密鍵で暗号化したものが公開鍵で復号できる、のは公開鍵暗号方式ではありません□□□□のアルゴリズムではそれができるってだけです□□□□が公開鍵暗号方式の全てではありません。

参考
公開鍵暗号方式(Public key encryption system)
RSA暗号 - 出典:フリー百科事典『ウィキペディア□Wikipedia□□

SSHの公開鍵認証では、接続先サーバとクライアント間で以下のようなやり取りが発生します。

1. クライアントから接続したい旨の要求が来る
2. 接続先サーバは自分に登録済みであるクライアントの公開鍵で暗号化した質問をクライアントに送り返す
3. クライアントは自分の秘密鍵で復号して質問を読み、接続先サーバへ質問の回答を返す
4. 接続先サーバは回答を確認して、答えが正しければ共通鍵を生成しクライアントの公開鍵で暗号化した共通鍵をクライアントへ送る
5. クライアントは自分の秘密鍵で復号し共通鍵を得る
6. 双方で共通鍵が作成されたので、共通鍵を使った通信が開始される

接続先サーバが事前に登録されたクライアントの公開鍵を使って情報を暗号化し「接続先サーバ→クライアント」の通信方向で流れる情報に公開鍵暗号の性質が適用されます。秘密鍵を持つクライアントでしか情報を復号できない状態となります。

さて、ここで「公開鍵って言うくらいだし公開しちゃってもいいよね！」とクライアントの公開鍵を公開してしまうとどうなるでしょうか。

誰でもその公開鍵を入手し好き勝手にいろいろなサーバへ登録できる状態ですから、悪意ある人がその公開鍵を登録した偽サーバを作り、接続先サーバに成りすましするかもしれません。
偽サーバには、誰でも入手可能な正しいクライアントの公開鍵が登録されているのですから、**何の問題もなく偽サーバへ公開鍵認証でログインできちゃいます。**

公開鍵暗号を使って安全に情報を伝達し通信を始めることができたけどその通信相手が偽サーバだったという、暗号化を施しても意味がない結果となってしまう可能性があるわけです。

偽サーバへ誘導されたとしても、初めて接続するサーバの場合「フィンガープリントを確認してね。ログインしてもいい？」とSSHが聞いてきますので異常に気付くチャンスはあります。
ですが、

- フィンガープリントをきちんと確認しない人
- 目的サーバのフィンガープリントなんかしらん人
- フィンガープリントの警告を見ない/見せないクライアントアプリ

だったら、正しい接続先サーバに接続したかどうかわかりませんから、もしかすると気付かないかもしれないですね。

SSLの場合

接続先確認と暗号化を行っているだけでユーザ認証を行っているわけではありません。

公開鍵は接続先サーバから公開されたものを使い、その公開鍵の認証局デジタル署名を使って公開された公開鍵が目的のサーバの鍵であるか確認することができます。
第三者機関である認証局が接続先の保証をしてくれることで、その接続先サーバは正しい接続先だ、と判断することができます。

署名付きの公開鍵と秘密鍵のペアが盗まれていたり、認証局がだまされていればこの方法も崩壊することになりますが、まぁ気にしたらキリがありません。

追記：SSHの公開鍵認証における公開鍵を秘密にしなくても構わんじゃろ、というお人がいたので

> @_john_doe_ SSHの公開鍵を秘密にしておけと書いてあるけど理由が大間違いですねこれ
>
> — Shin Suzuki (@suzukis) 2015, 9月 8
>
> @_john_doe_ 接続先の検証はまさにその目的のために存在するホスト鍵の検証でやればいいのであって、それが役に立たないという前提なら「秘密の公開鍵」による認証だって無意味です。
>
> — Shin Suzuki (@suzukis) 2015, 9月 8
>
> SSHの公開鍵を秘密にしておくべき、という主張 on @Qiita http://t.co/dctWZX9Mja
>
> — Shin Suzuki (@suzukis) 2015, 9月 9

「SSHの公開鍵を秘密にしておくべき、という主張」で「論理的にも技術的にも成立しません」と言われてます。うへぇ。 ※偽サーバが認証スルーってのは反則だー：：

調べたのですが、この人の言う「ホスト鍵の検証」を具体的にどうやって設定・適用しろと言うてるのかがわからんのです。

- 正しくホスト鍵をクライアント側で登録できていれば偽ホストに導かれても「WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!」って言われてログインできないよね？という話？
  →自分の管理下のサーバならおかしい事がわかりますね。相手がレンタルサーバだった場合、鍵更新の連絡してくれるならいいですね。してくれなきゃクライアント側でknown_hostsいじらなきゃなんないからなぁ。意味無しに。自分の公開鍵で守るしかないじゃない。
  それに初回接続時が偽サーバでフィンガープリント確認せず処理継続しちゃったらもう気が付けないよね。
- 接続先ホストのsshdでホストベース認証(Hostbased認証)を有効にすればいいだろ？という話？
  →自分の管理下のサーバなら接続してくるクライアントを片っ端から登録できますね。相手がレンタルサーバでそういうサービスしてくれるならぜひ利用するべきです。でもホスト鍵の検証じゃないですよねこれ。話が違う。
- 公開鍵認証に設定があってそれを有効にすればホスト鍵の検証が行われ接続先が正しいかどうかわかるだろ？という話？
  →私その設定知らないです。わからんです。それを教えてほしい。クライアントにホスト鍵の検証（どう検証されるのかはわからないけど）を実施させる設定があるの？
- 上記以外？
  →『証明書による認証』という認証局の署名付き証明書を使う実装がOpenSSHにあるけどそれの事？
  サーバ側が対応できてもクライアントが対応できてないと使えないですね。

「ホスト鍵の検証」の具体的な設定等の話があれば推測もできるんだろうけど、やはりどういう話なのかが分からないです。
SSLの様に第三者となる認証局が入って処理する実装のSSHサービスSSHクライアントを使え」という話なら理解できます。でもそれは公開鍵認証の説明じゃないです。

FreeBSD, Unix, SSH, 技術資料