

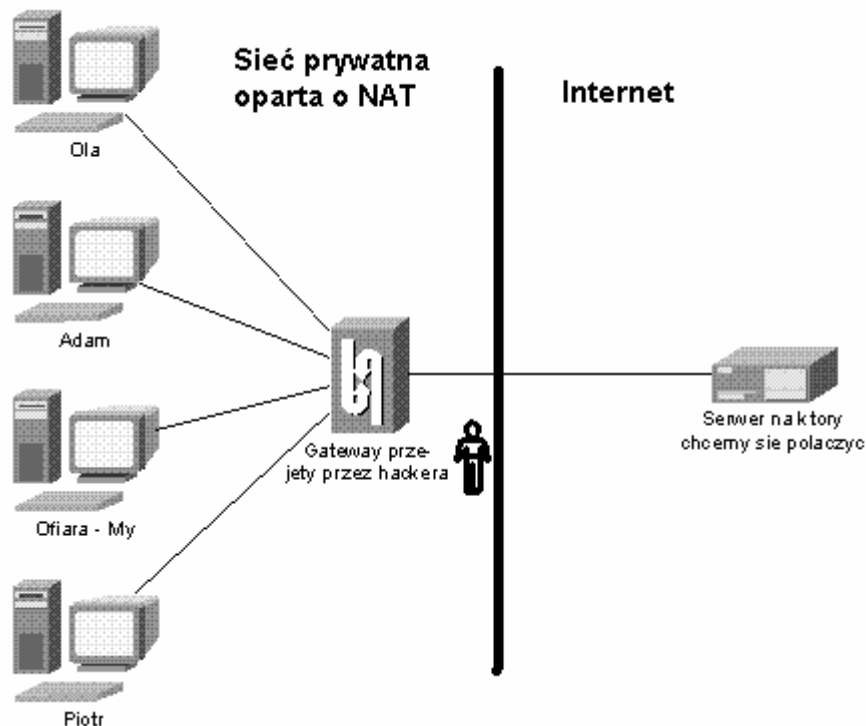
SSH: Identyfikacja Serwera

Autor: Damian Zelek, jimmy@hacking.pl

Data: 20 grudzień 2004

Wszyscy używamy/korzystamy z *SSH*. Dzięki temu protokołowi jesteśmy w stanie w bezpieczny sposób łączyć się z inną maszyną i wykonywać na niej w sposób zdalny komendy. Jesteśmy także w stanie przysyłać bezpiecznie pliki (*SCP*) a nawet tworzyć o niebo bezpieczniejszą alternatywę dla *FTP* (*RSSH + SCPOnly*), tworzyć szyfrowane tunele itp. Jednak wiele osób nie zdaje sobie sprawy z tego, iż chociaż *SSH* szyfruje przesyłane dane to dla atakującego wcale nie jest problemem podsłuchać Naszą sesję. Podsłuchiwanie to z powodu szyfrowania sesji przybiera trochę inny kształt: atakujący może symulować serwer, z którym chcemy się połączyć. Dzięki symulowaniu serwera będzie w stanie Nas "podsłuchiwać", ponieważ chociaż dane są szyfrowane to de facto będą one przechodziły (lub adekwatnie od intencji atakującego -- dochodziły) przez "niepowołaną" maszynę. Jest to atak typu *Man-In-The-Middle*. Tekst ten ma na celu przybliżenie Wam sposobu ochrony Naszej szyfrowanej sesji *ssh* dzięki kluczom identyfikacyjnym serwera.

Krótkie omówienie ataku Man-In-The-Middle



Jak widzimy na powyższym przykładzie Nasza maszyna (ofiara) chciała się połączyć zdalnie z serwerem. Jednak na drodze połączenia stanął komputer atakującego (ponieważ zarówno Nasz komputer jak i atakującego są w jednej i tej samej sieci wewnętrznej opartej o NAT a dodatkowo komputer przejęty przez hakera pełni rolę gateway'a to wcale nie było to trudne, a wręcz trywialne). Gateway, który jest pod kontrolą hakera, jest w stanie imitować serwer usługi, z którą chcemy się połączyć. Jeżeli do tego dojdzie, będziemy połączeni z maszyną włamywacza myśląc, iż jesteśmy połączeni z docelowym serwerem. Wszystkie dane

(łącznie z hasłem użytym podczas logowania) przechwyci hacker. Tylko od niego zależy czy chce uzyskać tylko i wyłącznie hasło "na szybko" czy też, bardziej prawdopodobnie, będzie chciał przejąć i korzystać z Naszego konta na serwerze. Jeżeli tak, to w tym celu będzie on odbierał wszystkie Nasze pakiety, analizował je i przysyłał dalej do serwera. Dzięki takiemu działaniu nieświadomy "user" nie będzie nawet podejrzewać, iż coś jest nie tak. Należy tu także wspomnieć, iż hacker wcale nie musi przejąć gatewaya. Może użyć technik, które pozwolą to pominąć...

Identyfikacja serwera

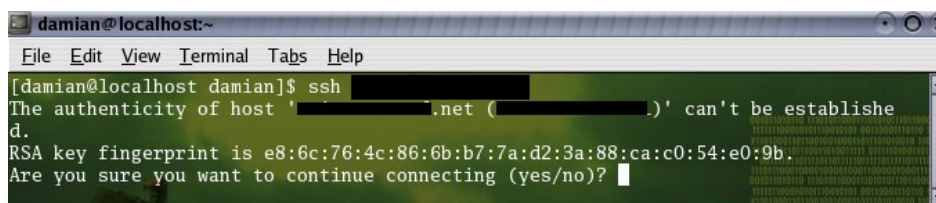
Standardowo SSH udostępnia Nam możliwość "obrony" przed powyższym atakiem (oraz wieloma innymi). Jest to możliwe dzięki temu, iż klient ssh podczas łączenia się z serwerem sprawdzi jego klucz (weryfikujący tożsamość serwera). Klucz ten w bazie Naszego klienta może znaleźć się na kilka sposobów:

- najbezpieczniejszym sposobem jest skontaktowanie się z administratorem serwera, z którym chcemy się połączyć i prośba o podanie klucza SSH (mądrym rozwiązaniem ze strony administratora jest stworzenie strony internetowej szyfrowanej za pomocą np. SSL, na której będzie dostępny klucz)
- ręczne wprowadzenie klucza dostępnego od osób trzecich (np. znajomy, który także posiada konto na danym serwerze i łączy się z nim za pomocą SSH)
- akceptacja klucza przy pierwszym połączeniu

To ostatnie rozwiązanie jest najczęściej stosowanym. Łącząc się pierwszy raz z danym serwerem, dostaniemy ostrzeżenie, iż nie posiadamy w swojej bazie klucza owego serwera (Rysunki 1, 2). Teraz od Nas zależy czy zaufamy serwerowi bądź nie. Często zatwierdza się tę decyzję i od tej pory, jeżeli ktoś stanie na drodze pomiędzy Nami a serwerem dostaniemy ostrzeżenie, iż klucz, który dostaliśmy od serwera (tak naprawdę od usługi na komputerze hackera) nie zgadza się z tym, który jest w Naszej bazie kluczy. Wykryliśmy próbę ataku!!! (tzn. niedokładnie, ale o tym później).



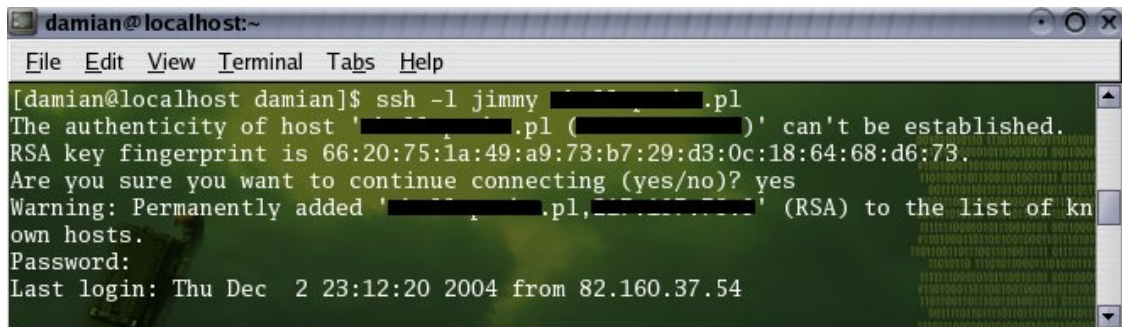
Rys. 1. Ostrzeżenie wywołane przez klienta ssh (Windows, PuTTY)



Rys. 2. Ostrzeżenie widoczne w konsoli (Unix/Linux)

Pierwsze podejście...

Zobaczmy jak wygląda typowe połączenie...



```
damian@localhost:~  
File Edit View Terminal Tabs Help  
[damian@localhost damian]$ ssh -l jimmy [redacted].pl  
The authenticity of host '[redacted].pl ([redacted])' can't be established.  
RSA key fingerprint is 66:20:75:1a:49:a9:73:b7:29:d3:0c:18:64:68:d6:73.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '[redacted].pl,[redacted]' (RSA) to the list of known hosts.  
Password:  
Last login: Thu Dec  2 23:12:20 2004 from 82.160.37.54
```

Rys. 3. Przebieg typowego połączenia

Łączymy się z serwerem po raz pierwszy, więc dostaliśmy ostrzeżenie o nieznanym kluczu. Założmy, iż ufamy temu serwerowi i połączymy się (wyberzemy opcje YES). Jak widać na powyższym rysunku Nasz system dodał klucz owego serwera na stałe do "bazy danych kluczy". Przy następnym połączeniu nie powinniśmy dostać żadnych ostrzeżeń (jeżeli dostaniemy to będzie znaczyło to iż najprawdopodobniej ktoś "imituje" serwer i w takim przypadku NIE wyrazimy zgody na dokończenie połączenia a tym samym nie staniemy się łupem dla hackera).

W momencie akceptacji połączenia Nasz klient SSH zapisał klucz serwera w Naszym systemie w pliku `$HOME/.ssh/known_hosts`. To właśnie ten plik jest Naszą bazą kluczy, która przechowuje wszystkie zatwierdzone bądź ręcznie wprowadzone klucze. Przykładowa wpis w tym pliku wygląda tak:

```
serwer.pl,217.123.245.54 ssh-rsa  
AAAAB3NzaC1yc2EAAAABIWAAAIEAt2cX8Lpo9eDGUxv5Q5iEnuLOeu7xiyMwuPZcE  
Uwav5GIwgVqWlJOnJwLOZVLjRpilI+crKR3DHH3BBQDgIPfZRPJRVtjt4hztGJZFvkdX  
jDehz8VaqsaxQS4kFZDBGQtdufe6ng7f4RNmy6ibPu4T2D3bbpI+2G0GStMyaj77kE=
```

Budowa pliku składa się kolejno z poszczególnych wpisów akceptacji. Z kolei zaś poszczególne wpisy są bardzo intuicyjne i łatwe do zrozumienia. Składają się z: host/IP serwera, określenie typu klucza oraz sam klucz publiczny. Istnieje także druga "baza kluczy", ale o innym prioritycie. Baza znajduje się w `/etc/ssh/ssh_known_hosts`. Przechowuje ona hosty mające potwierdzenie z poziomu administratora (czyli odgórnie). Do modyfikacji pliku należy użyć odpowiedniego parametru w pliku konfiguracyjnym klienta ssh (`/etc/ssh/ssh_config`).

Rodzaje kluczy

Wyróżniamy 3 rodzaje kluczy: SSHv1 RSA (rsa1), SSHv2 RSA (rsa) i SSHv2 DSA (dsa). Opcje odnośnie tych kluczy i ich używania znajdziemy w pliku konfiguracyjnym ssh `/etc/ssh/sshd_config` (m.in. możliwość ładowania tylko określonych kluczy `[RSAAuthentication yes]` - włączenie tych bardziej bezpiecznych). Po zainstalowaniu OpenSSH wszystkie 3 klucze powinny zostać stworzone w standardowej procedurze automatycznej. Jeżeli tak się nie stało możemy tworzyć je ręcznie:

```
ssh-keygen -t rsa /etc/ssh/ssh_host_rsa_key
```

Zamiast rsa możemy oczywiście podstawić rsa1 i dsa... Wzmianka o rodzajach kluczy jest niezbędna do zrozumienia poniższej sekcji.

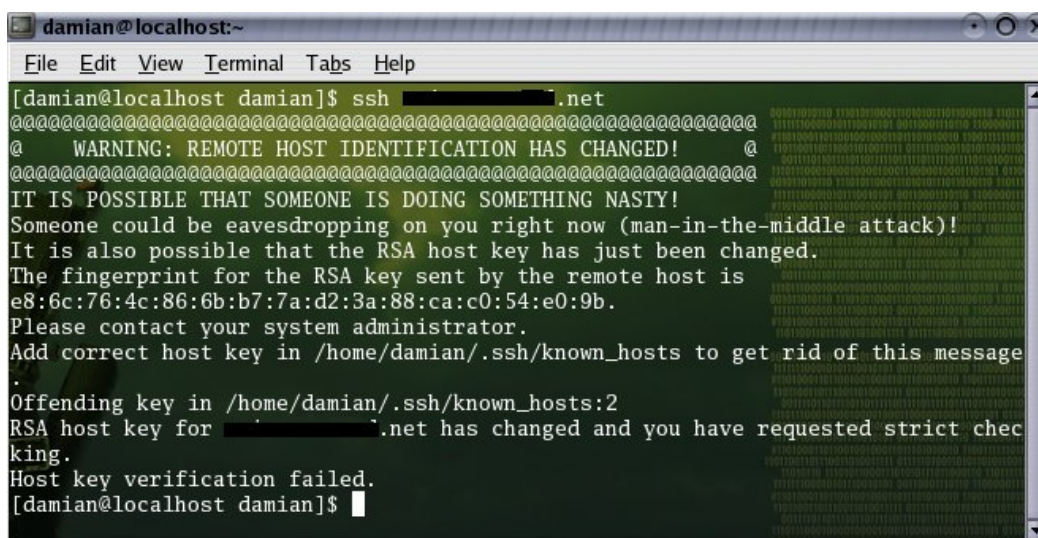
Pomocy!!! Wpadłem

Jak już wspominałem wcześniej nie jest zadaniem wcale trudnym podsłuchanie/przejęcie Naszej sesji SSH przez osobę z Naszej sieci. Do tego celu stworzono kilka dobrych programów "gotowców", więc każdy początkujący "hacker" lub nawet wścibski kolega z sieci będzie w stanie Nas przechwycić. Pamiętajmy o jednym - będzie w stanie Nas przechwycić tylko za sprawą Naszej własnej głupoty poprzez akceptację nieznanego klucza z Naszej strony. Jeżeli jesteś przezorny sekcja ta nie jest dla Ciebie. Jeżeli jednak miała miejsce wpadka (przypadkowe/nieumyślne zaakceptowanie nieznanego klucza MIMO ostrzeżenia) sprawa nie jest jeszcze całkowicie przegrana. Można sprawdzić czy atak naprawdę miał miejsce....

Zacznijmy od tego, iż zmiana klucza wcale nie musi oznaczać próby ataku. Do jego zmiany może dojść także z powodu:

- zmiana IP/host/DNS serwera
- nieudolny reinstal systemu
- update/upgrade ssh
- zmiana protokołu między SSHv1 i SSHv2

Teraz, jak wychodzić z opresji?. Dzień, jak co dzień, normalne logowanie w celu sprawdzenia poczty...



```
damian@localhost:~  
File Edit View Terminal Tabs Help  
[damian@localhost damian]$ ssh [redacted].net  
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  
@   WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!   @  
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!  
Someone could be eavesdropping on you right now (man-in-the-middle attack)!  
It is also possible that the RSA host key has just been changed.  
The fingerprint for the RSA key sent by the remote host is  
e8:6c:76:4c:86:6b:b7:7a:d2:3a:88:ca:c0:54:e0:9b.  
Please contact your system administrator.  
Add correct host key in /home/damian/.ssh/known_hosts to get rid of this message  
.  
Offending key in /home/damian/.ssh/known_hosts:2  
RSA host key for [redacted].net has changed and you have requested strict check-  
ing.  
Host key verification failed.  
[damian@localhost damian]$
```

Rys. 4. ...coś jednak nie gra!

A jednak nie!! Klucz ssh się zmienił. Musimy zapisać RSA fingerprint (ta długa liczba oddzielona dwukropkami). Następnie logujemy się (jak widać i tak musimy się zalogować, czyli nawet jeżeli do ataku doszło kilka dni wcześniej a my dopiero teraz zdaliśmy sobie sprawę z powagi sytuacji to "fałszywy" klucz został zapisany do Naszego *known_hosts* a za pomocą komendy *ssh-keygen* można "otrzymać" RSA fingerprint -- czyli de fakto także możemy sprawdzić czy to był atak czy też nie). Po zalogowaniu przechodzimy do katalogu */etc/ssh* i listingujemy wszystkie pliki **.pub*

```
ls *.pub  
ssh_host_dsa_key.pub ssh_host_key.pub ssh_host_rsa_key.pub
```

Odpowiednie użycie komendy *ssh-keygen* + odpowiedni plik **.pub* zwróci Nam fingerprint klucza. Zastanówmy się przez chwilę, który plik **.pub* wybrać. Jeżeli wiemy, iż korzystamy ze

starszej wersji protokołu SSHv1 będzie nam potrzebny plik *ssh_host_key.pub*. Dla RSA fingerprint użyjemy pliku z "rsa" w nazwie a analogicznie do DSA fingerprint użyjemy pliku z "dsa" w nazwie. Wybór nie jest obojętny: musisz wybrać zależnie DSA lub RSA odwołując się do ostrzeżenia, które dostaliśmy na początku:

```
...  
RSA key fingerprint is 98:2e:d7:e0:de:9f:ac:67:28:c2:42:2d:37:16:58:4d.  
...
```

Jak widać tutaj był RSA fingerprint więc komenda wyglądałaby tak: `ssh-keygen -lf /etc/ssh/ssh_host_rsa_key.pub`

Jeżeli RSA fingerprint teraz uzyskany zgadza się z tym wyświetlonym podczas logowania (lub analogicznie z tym, który został w Naszym *known_hosts*) to znaczy to, iż wszystko jest na swoim miejscu. W innym wypadku prawdopodobnie staliśmy się celem ataku - szybko skontaktuj się z administratorem serwera w celu zmiany hasła bądź ewentualnego wyjaśnienia przyczyny zmiany kluczy publicznych bez wcześniejszego powiadomienia Nas o tym. Należy także zaznaczyć, iż będąc zwykłym "userem" z kontem shell na jakimś serwerze masz zapewniony dostęp do plików **.pub* w */etc/ssh* i komendy *ssh-keygen*. Tak więc jest dana Tobie możliwość SAMODZIELNEGO sprawdzenia czy stałeś się ofiarą hakera czy też nie....

Hacking zmorą usera

Opisany wyżej sposób "detekcji powłamaniowej" działa jeżeli atak był przeprowadzany przez niedoświadczonych hackerów/script-kiddiez. Jest wiele bardziej zawitych technik, dzięki którym taka detekcja jest ograniczona lub niemożliwa. Podczas wielu takich prób wykonywanych przeze mnie wystarczy wspomnieć chociaż o analizie pakietów odbieranych przez hakera a wysyłanych potem do serwera ze szczególnym naciskiem na pochodne *logout*, *ssh-keygen* bazujące na połączeniu in-out (tak aby fingerprint usługi zwracany był z serwera a nie usługi na komputerze hakera), etc. Dlatego należy zdać sobie sprawę z tego, iż nigdy nie będziemy w 100% bezpieczni. Jednak dobrą przeszkodą dla włamywacza będzie stosowanie się do paru zasad:

- UNIKANIE logowania w wypadku, gdy klucz się nie zgadza
- ręczne wpisywanie kluczy do *known_hosts* i tylko z pierwszej ręki (np. admin, serwisant; jeżeli administrator zdaje sobie sprawę z tego, iż prawdopodobnie nie będzie miał czasu na każdorazowe podawanie *fingerprint klucza powinien postarać się o wcześniej wspomniane wystawienie klucza na szyfrowanej via SSL stronie www bądź rozważenie możliwości użycia */etc/motd* jako "informatora")
- ustawienie opcji *StrictHostKeyChecking=* w pliku konfiguracyjnym klienta (*/etc/ssh/ssh_config* lub w konfiguracji w *\$HOME/.ssh*) tak aby wskazywała na *yes* (w takim wypadku zły klucz lub jego brak uniemożliwi logowanie -- dzięki temu zyskamy ochronę przed Naszą własną głupotą/nieuwagą a nawet więcej); ochrona taka została zaprezentowana na Rysunku4 (możliwość zalogowania/połączenia została Nam "odebrana").