

Studi Keamanan pada Protokol SSH

Karunia Ramadhan 13508056
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18056@students.if.itb.ac.id

Abstraksi—Protokol SSH merupakan sebuah bagian keamanan yang sangat penting saat melakukan *remote connection*, yang dalam hal ini adalah pengaksesan sebuah mesin (komputer) dari mesin lain. SSH sendiri sangat terkenal dari segi kemanannya, dan karena itu cenderung langsung digunakan oleh berbagai pihak. Sayangnya, keamanan SSH sendiri mungkin bergantung pada berbagai hal yang tidak akan langsung terkonfigurasi *out-of-the-box* dan masih memiliki berbagai hal yang perlu dipertimbangkan sampai bisa digunakan untuk mengirimkan informasi antar mesin secara benar-benar aman.

Kata Kunci— Keamanan informasi, protokol SSH, *remote connection*

I. PENDAHULUAN

Keamanan merupakan hal yang sangat penting dalam keberlangsungan perjalanan informasi. Dari jaman dahulu, mekanisme pengiriman data yang menjamin integritas dan keamanannya selalu dicari dan dibuat demi ketersampaiannya informasi dari satu pihak ke pihak lain. *Hieroglyph* bangsa mesir, *scytale* Yunani kuno, bahkan mesin *Enigma* Jerman pada perang dunia kedua, semua alat tersebut memiliki fungsi untuk menjaga kerahasiaan sebuah pesan yang akan dikirimkan. Ilmu dan seni yang digunakan untuk menjaga keamanan pesan tersebut sendiri adalah kriptografi.

Sejak ditemukannya dan berkembangnya komputer (1946-sekarang), keilmuan kriptografi telah memasuki tahap baru. Pengoperasian yang tadinya berlangsung pada mode karakter seperti *caesar cipher* dan *vigenere cipher*, sekarang berlangsung menggunakan mode dan operasi bit. Berbagai teknik kriptografi modern mulai muncul memanfaatkan komputer dan operasi-operasi yang hanya bisa dilakukannya. Seiring juga dengan munculnya teknologi jaringan komputer dan internet, kriptografi menjadi salah satu fitur penting dalam mengamankan pengiriman data dan informasi pada jaringan-jaringan tersebut.

Meningkatnya kebutuhan untuk mengamankan informasi membuat algoritma kriptografi kunci publik menjadi berkembang dan digunakan di berbagai macam protokol jaringan. Munculnya protokol SSH yang

menggantikan protokol rlogin, telnet, dan rsh yang sebelumnya tidak menyediakan keamanan secara kuat seakan menegaskan hal tersebut.

Protokol SSH, pertama didesain tahun 1995 dan telah digunakan oleh 20000 orang pada akhir tahun itu, merupakan protokol jaringan komputer yang berprinsip menciptakan *secure channel* untuk pengiriman data. Pada tahun 2000, diestimasikan bahwa telah ada 2 juta pengguna yang menggunakan protokol tersebut. Bahkan sampai saat ini, implementasi *open* dari SSH (OpenSSH, sekaligus implementasi SSH yang paling populer) telah muncul secara *default* di berbagai sistem operasi dan dipakai secara langsung oleh penggunanya. Tentunya, sesuai dengan kepopuleran tentang keamanan protokol SSH, banyak juga penyerangan yang ditujukan pada protokol ini. Salah satu bentuk penyerangan yang paling terkenal, *brute-force attack*, bahkan hanya mulai menyerang menggunakan percobaan berbagai kombinasi default username dan password yang digunakan pada SSH. Hal ini secara umum akan mampu menembus SSH yang tidak terkonfigurasi, dan hal ini harusnya membuat para pengguna berpikir untuk lebih mengamankan jalur komunikasi SSH mereka.

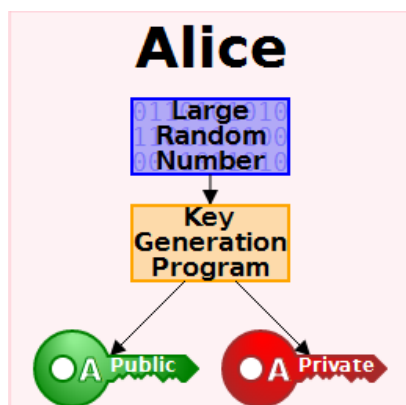
Pada makalah ini, penulis akan mengeksplorasi protokol SSH secara lebih lanjut, membahas bagaimana cara SSH sebenarnya mengamankan informasi yang dikirimkan dan mengujikan mengapa SSH berbeda dengan protokol-protokol sebelumnya. Penulis juga akan membahas berbagai metode penyerangan terhadap protokol SSH dan menganalisis cara pencegahannya secara lebih lanjut.

II. DASAR TEORI

A. Algoritma Kriptografi Kunci Publik

Algoritma kunci publik merupakan sebuah metode kriptografi dimana pesan yang dikirim hanya akan bisa dibaca oleh penerima yang diinginkan. Algoritma ini menggunakan prinsip kunci asimetri, dimana sebuah informasi umum (kunci publik) yang diperlukan untuk membuat pesan menjadi aman berbeda dari informasi yang dibutuhkan untuk membalikkan proses tersebut (kunci privat).

Dalam implementasinya, setiap pengguna memiliki sebuah pasangan kunci, yaitu kunci publik (enkripsi) dan kunci privat (dekripsi). Kunci publik akan disebarluaskan secara luas sedangkan kunci privat hanya akan dimiliki oleh sang pemilik. Pesan kemudian akan dienkripsi oleh kunci publik milik sang penerima, yang kemudian hanya akan bisa didekripsi oleh kunci privat miliknya juga. Kunci-kunci tersebut sebenarnya terikat secara matematik, tapi berbagai parameter digunakan dalam menentukannya sehingga penentuan kunci privat dari sebuah kunci publik akan menanggung biaya yang besar.



Gambar I – Pengenerasian Kunci Asimetrik

Masalah utama pada penggunaan algoritma kunci publik adalah kepercayaan bahwa suatu kunci publik benar milik orang yang mengaku pemilik (otentik) dan sebelumnya tidak diubah oleh pihak ketiga. Pendekatan untuk menangani masalah ini biasanya menggunakan *public key infrastructure* (PKI) yang akan mensertifikasi kebenaran kunci publik pemilik.

B. Protokol SSH

SSH merupakan sebuah protokol jaringan yang memungkinkan data untuk dipertukarkan menggunakan *secure channel* antara dua mesin yang berada dalam jaringan. Protokol ini menggunakan kriptografi berbasis kunci publik untuk mengotentifikasi komputer *remote* dan mengijinkan komputer *remote* tersebut untuk mengotentifikasi pengguna bila diperlukan.

Protokol SSH sendiri digunakan oleh berbagai aplikasi di berbagai *platform* seperti Unix, Microsoft Windows, Apple Mac OS X, dan Linux. Contoh kegunaan SSH adalah sebagai berikut :

- Untuk login kedalam *shell* di mesin *remote*.
- Untuk mengeksekusi perintah pada mesin *remote*.
- Melakukan transfer *file* secara aman (SFTP).
- Menggunakannya sebagai VPN yang terenkripsi secara lengkap.
- *Web browsing* melalui koneksi *proxy* yang

terenkripsi.

SSH memiliki tiga tahap untuk mencapai tujuannya, yaitu SSH-TRANS, SSH-AUTH, dan SSH-CONN :

1. Tahap Pertama - SSH Transport Protocol

- Tahap ini berfungsi untuk menyediakan koneksi yang dipercaya setelah sebelumnya baik *client* maupun *host* tidak mempunyai informasi satu sama lain.
- *Client* akan menghubungi *host* menggunakan *session* TCP dan masing-masing akan mengirimkan informasi protokol mereka dan menyetujui dengan protokol mana mereka akan berkomunikasi (SSH 1.0 vs SSH 2.0) beserta informasi-informasi lainnya.
- *Client* kemudian akan mengirimkan rekues kepada server untuk menginisialisasi pertukaran kunci secara Diffie-Hellman, spesifikasinya, dan *challenge message* awal. Server akan membalas pesan tersebut dengan kunci publik server dan pesan yang telah ditanda-tangani dengan kunci private server. Hal ini akan menyediakan validasi bahwa paket tersebut hanya bisa berasal dari server yang menyediakannya.
- *Client* akan mengecek list dari *host* yang dia ketahui (*.ssh/known_hosts*). Bila kunci publik server ada dalam list tersebut, maka data akan diasumsikan valid dan server tersebut dapat dipercaya. Bila tidak, suatu *warning* akan muncul untuk membuat *client* memeriksa keaslian tanda-tangan tersebut.
- Pada saat ini, baik server maupun klien mempunyai semua informasi yang digunakan untuk membuat kunci master untuk enkripsi *session*.

2. Tahap Kedua – SSH Authentication Protocol

- Tahap ini digunakan untuk menyediakan suatu bentuk autentifikasi kepada server untuk membuktikan bahwa pengguna diijinkan untuk mengakses *resource* server.
- Pada tahap ini, *client* akan mencoba mengakses server dan server akan meminta autentifikasi (bisa berupa password, kunci publik, dan lainnya). Server kemudian akan menerima ataupun menolak informasi tersebut sesuai data yang ada pada server.

3. Tahap Ketiga – SSH Connection Protocol

- Pada tahap ini, SSH *client* telah membuat sebuah jalur komunikasi yang aman untuk berkomunikasi dengan server.
- *Client* kemudian akan berinteraksi pada server untuk mengakses sumber daya di server. Biasanya hal ini hanya terbatas pada akses

shell, tapi pada protokol SSH 2.0 ada berbagai fitur tambahan yang akan bisa diakses seperti *port forwarding* dan penerusan *X graphical client*.

- Tahap ini bisa mengurus berbagai sesi sekaligus. Fungsionalitas seperti transfer *file*, pengaksesan perintah secara *remote*, akses *shell*, dan lain lain masing-masing membutuhkan satu sesi.

III. PENGUJIAN PERBANDINGAN PROTOKOL SSH DENGAN TELNET

A. Konsep

Pengujian keamanan standar SSH akan dilakukan dengan cara membandingkannya dengan protokol Telnet, dimana telnet adalah protokol *remote* login terkenal yang pada akhirnya digantikan oleh SSH tahun 1990-an. Pengujian ini bertujuan untuk mengetahui perbedaan standar keamanan yang digunakan SSH dengan protokol yang sebelumnya dipakai.

Pengujian dilakukan dengan dua mesin (klien dan server) yang terhubung secara dalam jaringan secara *bridging* :

1. Mesin server menggunakan sistem operasi Ubuntu 10.04 yang berjalan dalam VirtualBox. Mesin ini menggunakan OpenSSH versi 5.3.

Konfigurasi IP mesin adalah : 192.168.1.109

2. Mesin klien menggunakan sistem operasi Windows 7 dengan client Telnet langsung dari *command prompt* dan client SSH menggunakan Putty r8981.

Konfigurasi IP mesin adalah : 192.168.1.105

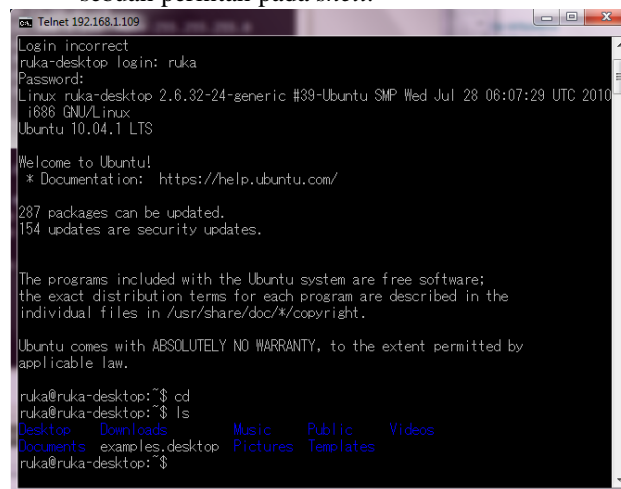
Dari kedua mesin ini, akan dilakukan komunikasi dengan kedua protokol yang kemudian paketnya akan dianalisis menggunakan Wireshark untuk melihat tingkat keamanannya.

B. Langkah Pengujian

1. Komunikasi menggunakan Telnet

- *Client* memperbolehkan *client* Telnet dijalankan -> Control Panel – Programs – Turn Windows Feature On/Off – Telnet Client
- Server mengunduh dan menjalankan Telnet Server -> `sudo apt-get install telnetd` dari terminal
- *Client* melakukan koneksi telnet ke server -> `telnet 192.168.1.109`
- *Client* memasukan autentifikasi dan melakukan

sebuah perintah pada *shell*.



```
Telnet 192.168.1.109
Login incorrect
nuka-desktop login: nuka
Password:
Linux nuka-desktop 2.6.32-24-generic #39-Ubuntu SMP Wed Jul 28 06:07:29 UTC 2010
i686 GNU/Linux
Ubuntu 10.04.1 LTS

Welcome to Ubuntu!
 * Documentation: https://help.ubuntu.com/

287 packages can be updated.
154 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

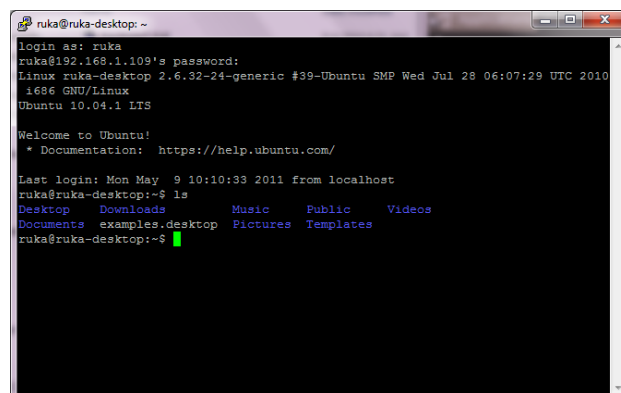
nuka@nuka-desktop:~$ cd
nuka@nuka-desktop:~$ ls
Desktop  Downloads  Music      Public     Videos
Documents examples_desktop Pictures  Templates
nuka@nuka-desktop:~$
```

Gambar II – Telnet pada Client

- Penguji mencari paket yang dikirimkan melalui Wireshark dan mengikuti paket tersebut.

2. Komunikasi menggunakan SSH

- Server mengunduh dan menjalankan Telnet Server -> `sudo apt-get install openssh-server`
- *Client* menjalankan putty dan melakukan login ke server.



```
nuka@nuka-desktop:~
login as: nuka
nuka@192.168.1.109's password:
Linux nuka-desktop 2.6.32-24-generic #39-Ubuntu SMP Wed Jul 28 06:07:29 UTC 2010
i686 GNU/Linux
Ubuntu 10.04.1 LTS

Welcome to Ubuntu!
 * Documentation: https://help.ubuntu.com/

Last login: Mon May 9 10:10:33 2011 from localhost
nuka@nuka-desktop:~$ ls
Desktop  Downloads  Music      Public     Videos
Documents examples_desktop Pictures  Templates
nuka@nuka-desktop:~$
```

Gambar III – SSH pada Client

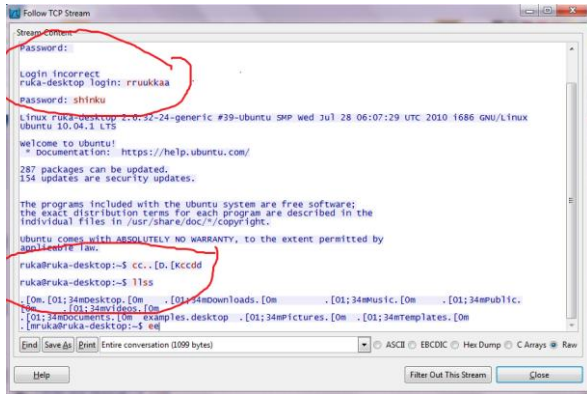
- Penguji mencari paket yang dikirimkan melalui Wireshark dan mengikuti paket tersebut.

C. Analisis Hasil Pengujian

Dalam pengujian ini, penulis mendapatkan hasil yang cukup terduga, dimana paket pada telnet tidak terenkripsi sama sekali dan paket pesan pada protokol SSH terenkripsi.

1. Komunikasi menggunakan Telnet

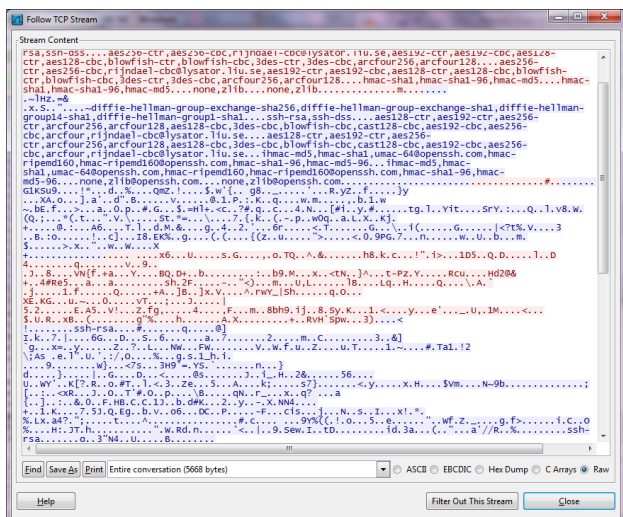
- *Wireshark* berhasil mendapatkan paket yang tidak terenkripsi sama sekali, dalam hal ini username, password, perintah yang dilakukan, bahkan respon dari server. Hal ini dikarenakan protokol telnet sama sekali tidak mengenkripsi datanya pada jaringan.



Gambar IV – Hasil sniffing paket pada TELNET

2. Komunikasi menggunakan SSH

- *Wireshark* berhasil mendapatkan paket yang sudah terenkripsi. Pengiriman data mulai dari tahap kedua (Connection) terenkripsi secara penuh.
- Hal yang juga perlu diperhatikan disini adalah terlihatnya pertukaran informasi pada paket mengenai spesifikasi dan informasi algoritma enkripsi. Hal ini dikarenakan pada tahap pertama (Transport), SSH masih menggunakan protokol TCP yang tidak terenkripsi.



Gambar V – Hasil sniffing paket pada SSH

IV. PENYERANGAN PADA PROTOKOL SSH

A. Konsep

Pada bagian ini, penulis akan menganalisis berbagai penyerangan terhadap protokol SSH. Untuk penyerangan yang memerlukan uji coba, konfigurasi mesin yang digunakan akan sama dengan bagian sebelumnya. Sayangnya, pengujian belum sempat mendapatkan hasil eksperimen sebenarnya (masih berupa hipotesis hasil) karena *tools* yang digunakan mengalami masalah saat kompilasi.

B. Analisis Penyerangan

1. Root Login

- Kelemahan protokol SSH 'out-of-the-box' yang ada pada berbagai sistem operasi (dalam bentuk openSSH) adalah diperbolehkannya login root langsung. Hal ini menyebabkan apabila password root diketahui, penyerang bisa memegang penuh kendali server.
- Salah satu cara penyerangan bisa dilakukan dengan *tool* *thc-hydra* yang telah dikompilasi sebelumnya dengan *cygwin* -> *hydra -l root 192.168.1.109 ssh2*
- Hydra akan melakukan *brute-force* pada kombinasi password dan hasil pencarian akan didapat dengan waktu yang bergantung keanehan kunci.

2. Brute Force User Login

- SSH yang ada tanpa konfigurasi biasanya tidak punya keamanan batasan untuk mencegah penyerangan autentifikasi login secara *brute force*.
- Dengan cara yang cukup sama, *hydra* bisa melakukan kombinasi *brute force* terhadap pasangan username (masukan file) dan password. Cara ini biasanya akan memakan waktu cukup lama karena juga perlu mencari user yang ada pada sistem.

C. Saran Pencegahan Penyerangan

1. Pencegahan Root Login

- Dengan mencegah login langsung *root* pada protokol SSH, server bisa mencegah serangan yang langsung terpusat pada *root* dan mencegah terambil-alihnya sistem secara langsung.
- Hal ini bisa dicapai dengan mengubah konfigurasi *ssh* pada */etc/ssh/sshd_config*, bagian "PermitRootLogin" menjadi bernilai "No".

2. Penggunaan Username dan Password yang Lebih Aman

- Saran yang cukup umum dan memang baik untuk menghadapi penyerangan *brute-force* adalah menggunakan kata kunci yang lebih aman. Hal ini akan menyebabkan proses penyerangan menjadi jauh lebih lambat dan bisa terdeteksi.
- Atribut yang penting untuk dipertimbangkan adalah panjang kunci panjang, kata yang tidak standar (gabungan huruf, angka, dan simbol), dan kata yang mudah diingat (oleh pengguna) tapi sulit ditebak.

3. Penggunaan Port SSH yang Tidak Standar

- Protokol SSH biasanya menggunakan port 22 dan memang secara umum penyerangan akan ditargetkan ke port tersebut. Dengan mengubah port SSH, penyerangan tidak bisa langsung dilakukan tetapi harus mengetahui port mana yang digunakan terlebih dahulu.
- Pengubahan port dapat dicapai dengan mengubah konfigurasi SSH bagian "Port" menjadi nomor port yang akan digunakan.

4. Pembatasan Login Memanfaatkan iptables

- Untuk mencegah penyerangan *brute force* secara lebih lanjut, pengurus server bisa memanfaatkan iptables. Dengan memasukkan rule-rule yang ketat pada iptables, penyerangan akan bisa ditunda dan diblok.
- Salah satu aturan yang bisa digunakan adalah membatasi jumlah login dalam sejumlah waktu tertentu ->

```
sudo iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW -m recent --set --name SSH
```

```
sudo iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW -m recent --update --seconds 60 --hitcount 8 --rttl --name SSH -j DROP
```

Aturan diatas akan membatasi jumlah login menjadi hanya delapan buah dalam satu menit untuk protokol SSH.

5. Pengubahan Autentifikasi SSH Menjadi Penggunaan Kunci Publik

- Dengan menggunakan algoritma kunci publik dan men-disable autentifikasi berbasis password, pengurus server bisa mencegah penyerangan yang berbasis autentifikasi login.
- Konfigurasi awal adalah dengan mengubah file

konfigurasi SSH (sama seperti sebelumnya) bagian "PasswordAuthentication" menjadi "No", dan memastikan nilai-nilai berikut :

```
RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys
```

- *Client* kemudian akan perlu membuat pasangan kunci (publik dan privat) dengan cara mengeksekusi perintah : `ssh-keygen -t dsa`
- Server perlu mendapat kunci publik dari sang klien dan menaruhnya di folder server bagian `.ssh/authorized_keys`.
- Selanjutnya, bila klien ingin login, dia cukup melakukan login SSH biasa dan otomatis server akan mengecek apakah klien tersebut mempunyai kunci publik yang dipercayai oleh server. Bila ya, dan bila setelah dicocokkan dengan kunci privat pada klien, pasangan kunci tersebut cocok, maka klien akan terautentifikasi dan bisa mengakses server.
- Dengan menggunakan kunci publik, masalah penyerangan yang menggunakan autentifikasi login akan terselesaikan. Satu masalah dalam menggunakan algoritma ini adalah verifikasi dari kunci publik klien (apakah kunci tersebut benar milik klien, atau ternyata sudah dimodifikasi oleh penyerang). Penyerangan dengan metode *man-in-the-middle* mampu mengubah hal ini pada prinsipnya meskipun praktiknya akan sulit.

V. KESIMPULAN

1. Protokol SSH merupakan suatu bentuk protokol jaringan yang berprinsip menciptakan sebuah saluran aman untuk bertukar data antar dua mesin yang berbeda. Protokol ini berbasis algoritma kriptografi kunci publik.
2. Keunggulan protokol SSH dibanding protokol-protokol yang mirip sebelumnya adalah keamanan yang cukup baik. Protokol ini mengenkripsi semua bentuk komunikasi pada server dan klien setelah sebelumnya dibalik layar menyetujui protokol keamanan yang akan digunakan. Hal ini bisa dilihat dari perbandingan protokol SSH dengan telnet yang hanya mengirim *plaintexts* secara langsung melewati jaringan.
3. Meskipun keamanan pada protokol SSH sudah cukup aman, masih diperlukan konfigurasi lebih lanjut dalam menangani penyerangan yang standar dilakukan. Perbaikan tersebut terdiri dari :

- Pencegahan superuser 'root' untuk langsung login melalui protokol SSH.
 - Penggunaan nama user dan kata kunci yang lebih aman.
 - Penggunaan port SSH yang tidak standar.
 - Pembatasan login yang dilakukan terhadap protokol SSH.
 - Penggantian autentifikasi SSH menggunakan algoritma kunci publik.
4. Dengan digunakannya autentifikasi algoritma kunci publik pada protokol SSH, masalah penyerangan berbasis autentifikasi login akan terselesaikan. Meskipun begitu, masalah algoritma kunci publik secara umum (seperti kepercayaan terhadap kunci publik klien) juga akan menjadi masalah autentifikasi protokol SSH yang digunakan.

DAFTAR PUSTAKA

- [1] <http://www.darknet.org.uk/2007/02/thc-hydra-the-fast-and-flexible-network-login-hacking-tool/>
- [2] <http://www.differencebetween.net/technology/internet/difference-between-telnet-and-ssh/>
- [3] http://en.wikipedia.org/wiki/Public-key_cryptography
- [4] http://en.wikipedia.org/wiki/Secure_Shell
- [5] http://en.wikipedia.org/wiki/Tatu_Y1%C3%B6nen
- [6] <http://www.itwire.com/business-it-news/open-source/13841-hack-and-crack-proof-ssh-on-linux>
- [7] <http://www.linux.com/learn/tutorials/305769-advanced-ssh-security-tips-and-tricks>
- [8] <http://www.notesbit.com/index.php/scripts-unix/what-is-ssh-how-does-it-work-your-complete-guide-with-simple-practical-solution-linux-unix-windows/>
- [9] <http://people.clarkson.edu/~owensjp/pubs/leet08.pdf>
- [10] <http://www.thc.org/thc-hydra/>

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Mei 2011

ttd



Karunia Ramadhan 13508056