Firewall pada Linux

Oleh :

Ramos Luther

091 2200 546

Tugas Mata Kuliah IS Risk Management

Dosen :

Bambang Gunawan, SE., ST., MM.

Trigantoro Bramaningtyas, ST.

Program Magister Manajemen Sistem Informasi

Universitas Bina Nusantara

Jakarta

2009

Setting Firewall di Linux

Sebelum Setting mintalah IP publik ke ISP lengkap dengan netmask, broadcast dan dns-nya. Kemudian tentukan juga IP Lokal yang akan digunakan pada komputer client. Misal : (eth0) IP : 192.168.1.2 NETMASK : 255.255.255.0 GATEWAY : 192.168.1.1 BROADCAST : 192.168.1.255 NETWORK : 192.168.1.0DNS1 : 202.134.0.155DNS2 : 202.134.2.5

DNS3 : 203.130.193.74

(eth1)IP : 192.168.10.254/24NETMASK : 255.255.255.0BROADCAST : 192.168.10.255

NETWORK : 192.168.10.0

Catatan : loginlah ke mesin linux anda dengan username sebagai ROOT. Untuk melakukan perubahan tekan tombol (insert) dan untuk menyimpan perubahan tekan escape : wq (write quit).

Settinglah IP pada ethernet-0.# vi /etc/sysconfig/network-scripts/ifcfg-eth0ip static DEVICE=eth0 BOOTPROTO=static BROADCAST=192.168.1.255 IPADDR=192.168.1.2NETMASK=255.255.255.0NETWORK=192.168.1.0

ONBOOT=yes

dhcpDEVICE=eth0BOOTPROTO=dhcp

ONBOOT=yes

Settinglah IP MGW dan HostName, serta DNS Resolver # vi /etc/sysconfig/network NETWORKING=yes HOSTNAME=router

GATEWAY=192.168.1.1

vi /etc/resolv.confnameserver 202.134.0.155nameserver 202.134.2.5

nameserver 203.130.193.74

Settinglah IP pada ethernet-1# vi /etc/sysconfig/network-scripts/ifcfg-eth1DEVICE=eth1BOOTPROTO=staticBROADCAST=192.168.10.255IPADDR=192.168.10.254NETMASK=255.255.255.0NETWORK=192.168.10.0

ONBOOT=yes

Pastikan default gateway telah mengarah ke IP gateway ISP, # route –nDan untuk melihat IP masing-masing ethernet cobalah command berikut :

ifconfig|more

Setting IP Forwarding, agar paket dari jaringan client dapat berjalan ke jaringan di luarnya melalui gateway.# vi /etc/sysctl.conf

Ubah net.ipv4.ip_forward = 0 menjadi net.ipv4.ip_forward = 1

chkconfig –level 2345 network on

/etc/rc.d/init.d/network restart

Sekarang lakukan testing dengan melakukan ping ke:# ping 192.168.1.1# ping 202.134.0.155 atau 202.134.2.5# ping www.google.com

ping 192.168.10.0/24

Jika hasilnya Reply berarti settingnya sudah berhasil dan tinggal selangkah lagi.

Agar client atau jaringan lokal (LAN) yang terhubung dengan sistem linux anda (ke eth1) dapat mengakses internet, maka settinglah MGW dengan menggunakan source NAT IPTables dan Forwarding.# /etc/init.d/iptables stop

vi /etc/rc.d/rc.nat

Tambahkan scripts berikut # !/bin/sh # flushIptables –FIptables –F –t nat# Script iptables untuk Source NAT sesuai dengan ip di eth0 dan eth1 (IP Statik)/sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.10.0/24 -j SNAT –to-source 192.168.1.2# Script iptables jika ip external eth0 merupakan DHCP/sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.10.0/24 -j MASQUERADE# Script Forwarding/sbin/iptables -t nat -A PREROUTING -i eth1 -s 192.168.10.0/24 -p tcp –dport 80 -j REDIRECT –to-ports 3128/sbin/iptables -t nat -A PREROUTING -i eth1 -s 192.168.10.0/24 -p udp –dport 80 -j REDIRECT –to-ports 3128/sbin/iptables -t nat -A PREROUTING -i eth1 -s 192.168.10.0/24 -p tcp –dport 8080 -j REDIRECT –to-ports 3128

/sbin/iptables -t nat -A PREROUTING -i eth1 -s 192.168.10/24 -p udp –dport 8080 -j REDIRECT –to-ports 3128

chmod +x /etc/rc.d/rc.nat

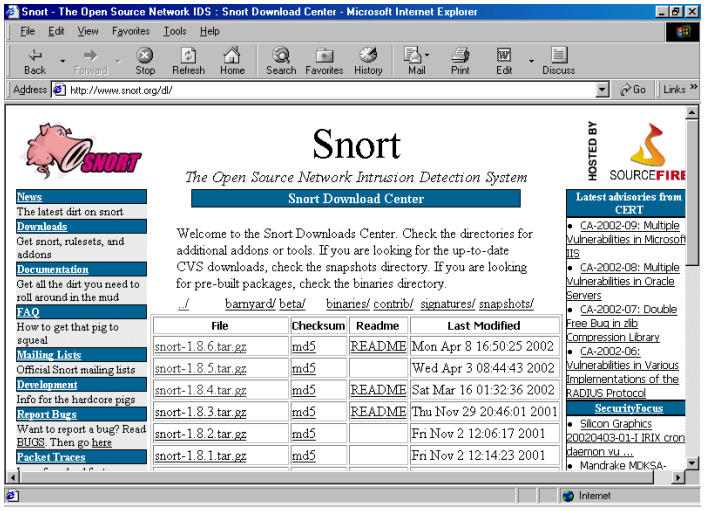
iptables –L –t nat

Simpanlah semua hasil konfigurasi di /etc/rc.local, sehingga Anda tidak perlu harus melakukan command-command sebelumnya setiap kali sistem di on-kan atau di-restart. Lakukan langkah berikut # vi /etc/rc.local:– Tambahkan script berikut –:# Local system initialization script# Put any local setup commands in here:#/etc/rc.d/rc.nat#

echo “”
Setting IDS di OS linux

Teknik Instalasi

Secara umum teknik instalasi snort di Linux sangat mudah. Bagi pemula mungkin ada baiknya menggunakan file RPM yang jauh lebih mudah instalasinya. Pada kesempatan ini saya menggunakan file tar.gz yang sedikit lebih sulit, walaupun sebetulnya tidak terlalu sulit juga. Beberapa langkah persiapan yang perlu dilakukan,



- Ambil source snort, saya biasanya mengambil source snort yang terakhir langsung dari www.snort.org. Biasanya file source tersebut berbentuk snort-*.tar.gz.

- Copykan file tar.gz tersebut ke directory /usr/local/src/

- Buka file snort-*.tar.gz menggunakan perintah # tar zxvf snort-*.tar.gz

- Biasanya source code snort akan terlihat di folder /usr/local/src/snort-*

- Pastikan library untuk capture packet (libpcap) terinstall, jika tidak yakin dapat menggunakan software manager melihat apakah libpcap terinstall. Jika belum terinstall install library libpcap tersebut, jika anda menggunakan Mandrake 8.0 hal ini cukup mudah dilakukan karena library tersebut terdapat pada CD Mandrake tsb.

Setelah semua persiapan selesai dilakukan, langkah yang perlu dilakukan untuk menginstalasi tidak banyak, yaitu:

- Masuk ke directory /usr/local/src/snort-*

  cd /usr/local/src/snort-*

- Konfigurasikan snort menggunakan

  ./configure

  untuk konfigurasi standar praktis tidak perlu di apa-apakan, biasanya pada saat konfigurasi ini kita akan diberitahukan jika ada module yang kurang yang perlu di install, seperti libpcap dll. Usahakan untuk mencari module tersebut di CD distribusi Linux yang kita miliki yang biasanya berbentuk RPM & mudah di install.

- Selanjutnya mengcompile source code menggunakan

  # make

  pastikan pada saat di install Linux di konfigurasi untuk melakukan development. Jika Linux tidak di install untuk melakukan development, compiler gcc biasanya tidak terinstall & kita tidak dapat menjalankan perintah make di atas.

- Setelah source di compile kita menginstall software snort menggunakan

  # make install

snort akan di install di directory yang sebenarnya. Default directory tempat instalasi snort adalah /usr/local/bin, /usr/local/man dll. Tentunya kita dapat meletakannya di directory lain selain /usr/local, dengan cara memberikan pilihan –prefix=PATH pada saat melakukan ./configure.

Untuk konfigurasi yang agak aneh-aneh misalnya ingin mengunakan ACID dll, maka kita perlu menambahkan beberapa switch / perintah setelah ./configure, beberapa switch yang mungkin akan digunakan seperti,

--with-snmp

menggunakan SNMP alerting code.

--with-mysql=DIR

mendukung mysql, perlu di on-kan jika kita menggunakan ACID dengan MySQL.

--with-odbc=DIR

mendukung database ODBC, perlu di on-kan jika kita menggunakan ACID dengan database yang tidak ada di daftar.

--with-postgresql=DIR

mendukung database Postgresql, perlu di on-kan jika kita menggunakan ACID dengan PostgreSQL.

--with-oracle=DIR

mendukung database Oracle, perlu di on-kan jika kita menggunakan ACID dengan Oracle.

--with-libpcap-includes=DIR

Jika script konfigurasi gagal memperoleh directory libpcap, maka kita dapat menset secara manual melalui switch ini.

--with-libpcap-libraries=DIR

Jika script konfigurasi gagal memperoleh directory libpcap, maka kita dapat menset secara manual melalui switch ini.

Setelah selesai diinstal, snort dapat langsung digunakan untuk melakukan sniffing & logging, hanya untuk Network Intrusion Detection System (NIDS) kita perlu melakukan setup / konfigurasi snort. Proses konfigurasi akan sangat ditolong dengan membaca manual SnortUsersManual.pdf yang ada di file snort-*.tar.gz. atau menjalankan perintah ./snort –h

Mengoperasikan Snort

Secara umum snort dapat di operasikan dalam tiga (3) buah mode, yaitu

- Sniffer mode, untuk melihat paket yang lewat di jaringan.
- Packet logger mode, untuk mencatat semua paket yang lewat di jaringan untuk di analisa di kemudian hari.
- Intrusion Detection mode, pada mode ini snort akan berfungsi untuk mendeteksi serangan yang dilakukan melalui jaringan komputer. Untuk menggunakan mode IDS ini di perlukan setup dari berbagai rules / aturan yang akan membedakan sebuah paket normal dengan paket yang membawa serangan.

Sniffer Mode

Untuk menjalankan snort pada sniffer mode tidaklah sukar, beberapa contoh perintah-nya terdapat di bawah ini,

./snort –v

./snort –vd

./snort –vde

./snort –v –d –e

dengan menambahkan beberapa switch –v, -d, -e akan menghasilkan beberapa keluaran yang berbeda, yaitu

-v, untuk melihat header TCP/IP paket yang lewat.

-d, untuk melihat isi paket.

-e, untuk melihat header link layer paket seperti ethernet header.

Contoh hasil sniffing paket di jaringan menggunakan perintah /usr/local/bin/snort –v dapat dilihat berikut ini,

[root@gate andrew]# /usr/local/bin/snort -v

Log directory = /var/log/snort

Initializing Network Interface eth0

Initializing Snort

Checking PID path...

PATH_VARRUN is set to /var/run/ on this operating system

PID stat checked out ok, PID set to /var/run/

Writing PID file to "/var/run/"

Decoding Ethernet on interface eth0

--== Initialization Complete ==--

-*> Snort! <*-

Version 1.8.3 (Build 88)

By Martin Roesch (roesch@sourcefire.com, www.snort.org)

04/18-11:32:00.261488 192.168.120.232:2757 -> 192.168.120.255:8859

UDP TTL:64 TOS:0x0 ID:1735 IpLen:20 DgmLen:38

Len: 18

=+
=+=+

04/18-11:32:10.261514 192.168.120.232:2758 -> 192.168.120.255:8859

UDP TTL:64 TOS:0x0 ID:1736 IpLen:20 DgmLen:38

Len: 18

=+
=+=+

04/18-11:32:20.261518 192.168.120.232:2759 -> 192.168.120.255:8859

UDP TTL:64 TOS:0x0 ID:1737 IpLen:20 DgmLen:38

Len: 18

=+
=+=+

Tampak bahwa antar paket selalu di batasi tanda =+=+=+=+=+. Karena sniffer di aktifkan hanya menggunakan switch –v, maka hanya header network & transport protocol yang diperlihatkan. Dalam contoh di atas protokol yang digunakan adalah Internet Protocol (IP) & User Datagram Protocol (UDP). Kalimat pertama berisi header IP, beberapa informasi yang penting yang dapat dihat di atas dari kalimat pertama adalah,

04/18-11:32:20.261518 192.168.120.232:2759 -> 192.168.120.255:8859

04	= versi protokol IP yang digunakan
192.168.120.232	= IP address sumber paket
192.168.120.255	= IP address tujuan
2759	= port sumber
8859	= port tujuan

Kalimat ke dua berisi protokol UDP, yang berisi antara lain informasi

UDP TTL:64 TOS:0x0 ID:1737 IpLen:20 DgmLen:38

UDP	= memberitahukan bahwa dia protocol UDP.
TTL:64	= Time To Live 64, paket maksimal melalui 64 router.
ID	= nomor / identitas
IPLen:20	= panjang byte protokol IP 20 byte
DgmLen	= panjang byte seluruh paket 38 byte.

Tentunya hasil akan lain jika kita melihat paket TCP, seperti beberapa contoh berikut,

04/18-11:32:20.573898 192.168.120.114:1707 -> 202.159.32.71:110

TCP TTL:64 TOS:0x0 ID:411 IpLen:20 DgmLen:60 DF

******S* Seq: 0x4E70BB7C Ack: 0x0 Win: 0x16D0 TcpLen: 40

TCP Options (5) => MSS: 1460 SackOK TS: 6798055 0 NOP WS: 0

=+
=+=+

04/18-11:32:20.581556 202.159.32.71:110 -> 192.168.120.114:1707

TCP TTL:58 TOS:0x0 ID:24510 IpLen:20 DgmLen:60 DF

***A**S* Seq: 0x423A85B3 Ack: 0x4E70BB7D Win: 0x7D78 TcpLen: 40

TCP Options (5) => MSS: 1460 SackOK TS: 163052552 6798055 NOP WS: 0

=+
=+=+

04/18-11:32:20.581928 192.168.120.114:1707 -> 202.159.32.71:110

TCP TTL:64 TOS:0x0 ID:412 IpLen:20 DgmLen:52 DF

A* Seq: 0x4E70BB7D Ack: 0x423A85B4 Win: 0x16D0 TcpLen: 32

TCP Options (3) => NOP NOP TS: 6798056 163052552

=+
=+=+

Sama seperti sebelumnya maka line pertama di atas adalah milik Internet Protocol (IP), kita dapat melihat bahwa terjadi komunikasi antara IP 202.159.32.71 port 110 dengan IP 192.168.120.114 port 1707.

Di bawah kalimat milik protocol IP, ada satu kalimat yang berisi informasi umum dari paket yang dikirim merupakan gabungan informasi milik protokol IP & TCP. Beberapa informasi seperti,

TCP TTL:64 TOS:0x0 ID:412 IpLen:20 DgmLen:52 DF

TCP TTL :64 = ada 64 router yang masih bisa di lewati.

TOS:0x0 = Type of Service dari protocol IP, 0x0 adalah servis normal.

IpLen:20 = Panjang protocol IP 20 byte.

DgmLen:52 = Panjang seluruh paket 52 byte.

Selanjutnya ada dua kalimat yang semuanya milik Transmission Control Protocol (TCP).

A* Seq: 0x4E70BB7D Ack: 0x423A85B4 Win: 0x16D0 TcpLen: 32

TCP Options (3) => NOP NOP TS: 6798056 163052552

Yang agak menarik adalah melihat state / kondisi sambungan, terlihat dari jenis paket yang dikirim, seperti

A* = packet acknowledge

******S* = paket sinkronisasi hubungan

***A**S* = paket acknowledge sinkronisasi hubungan

selebihnya adalah nomor urut paket data yang dikirim Sequence number (Seq), dan Acknowledge number (Ack) yang menunjukan sejauh ini nomor paket mana yang sudah diterima dengan baik.

Jenis paket lain yang kadang-kadang terlihat di layar adalah paket Address Resolution Protocol (ARP). Contoh adalah,

04/18-11:32:25.451498 ARP who-has 192.168.120.114 tell 192.168.120.1

04/18-11:32:25.451671 ARP reply 192.168.120.114 is-at 0:0:F0:64:96:AE

ARP digunakan untuk menanyakan address dari hardware, seperti ethernet card address, atau callsign amatir radio di AX.25. Dalam contoh di atas 192.168.120.1 menanyakan ke jaringan berapa ethernet address dari 192.168.120.114. Dalam kalimat selanjutnya 192.168.120.114 menjawab bahwa dia menggunakan ethernet card dengan address 00:00:f0:64:96:ae.

Setelah melihat semua paket yang lewat kita dapat menekan tombol Control-C (^C) untuk mematikan program snort. Akan tampak pada layar berbagai statistik yang sangat berguna untuk melihat kondisi jaringan

Snort analyzed 255 out of 255 packets, dropping 0(0.000%) packets

Breakdown by protocol: Action Stats:

TCP: 211 (82.745%) ALERTS: 0

UDP: 27 (10.588%) LOGGED: 0

ICMP: 0 (0.000%) PASSED: 0

ARP: 2 (0.784%)

IPv6: 0 (0.000%)

IPX: 0 (0.000%)

OTHER: 15 (5.882%)

DISCARD: 0 (0.000%)

==========

Fragmentation Stats:

Fragmented IP Packets: 0 (0.000%)

Fragment Trackers: 0

Rebuilt IP Packets: 0

Frag elements used: 0

Discarded(incomplete): 0

Discarded(timeout): 0

Frag2 memory faults: 0

==========

TCP Stream Reassembly Stats:

TCP Packets Used: 0 (0.000%)

Stream Trackers: 0

Stream flushes: 0

Segments used: 0

Stream4 Memory Faults: 0

==========

Snort received signal 2, exiting

[root@gate andrew]#

Packet Logger Mode

Beberapa perintah yang mungkin dapat digunakan untuk mencatat paket yang ada adalah

./snort –dev –l ./log

./snort –dev –l ./log –h 192.168.0.0/24

./snort –dev –l ./log –b

perintah yang paling penting untuk me-log paket yang lewat adalah

-l ./log

yang menentukan bahwa paket yang lewat akan di log / di catat ke file ./log. Beberapa perintah tambahan dapat digunakan seperti –h 192.168.0.0/24 yang menunjukan bahwa yang di catat hanya packet dari host mana saja, dan –b yang memberitahukan agar file yang di log dalam format binary, bukan ASCII.

Untuk membaca file log dapat dilakukan dengan menjalankan snort dengan di tambahkan perintah –r nama file log-nya, seperti,

./snort –dv –r packet.log

./snort –dvr packet.log icmp

Intrusion Detection Mode

Mode operasi snort yang paling rumit adalah sebagai pendeteksi penyusup (intrusion detection) di jaringan yang kita gunakan. Ciri khas mode operasi untuk pendeteksi penyusup adalah dengan menambahkan perintah ke snort untuk membaca file konfigurasi –c nama-file-konfigurasi.conf. Isi file konfigurasi ini cukup banyak, tapi sebagian besar telah di set secara baik dalam contoh snort.conf yang dibawa oleh source snort.

Beberapa contoh perintah untuk mengaktifkan snort untuk melakukan pendeteksian penyusup, seperti

./snort –dev –l ./log –h 192.168.0.0/24 –c snort.conf

./snort –d –h 192.168.0.0/24 –l ./log –c snort.conf

Untuk melakukan deteksi penyusup secara prinsip snort harus melakukan logging paket yang lewat dapat menggunakan perintah –l nama-file-logging, atau membiarkan snort menggunakan default file logging-nya di directory /var/log/snort. Kemudian menganalisa catatan / logging paket yang ada sesuai dengan isi perintah snort.conf.

Ada beberapa tambahan perintah yang akan membuat proses deteksi menjadi lebih effisien, mekanisme pemberitahuan alert di Linux dapat di set dengan perintah –A sebagai berikut,

-A fast, mode alert yang cepat berisi waktu, berita, IP & port tujuan.

-A full, mode alert dengan informasi lengkap.

-A unsock, mode alert ke unix socket.

-A none, mematikan mode alert.

Untuk mengirimkan alert ke syslog UNIX kita bisa menambahkan switch –s, seperti tampak pada beberapa contoh di bawah ini.

./snort –c snort.conf –l ./log –s –h 192.168.0.0/24

./snort –c snort.conf –s –h 192.168.0.0/24

Untuk mengirimkan alert binary ke workstation windows, dapat mengunakan perintah di bawah ini,

./snort –c snort.conf –b –M WORKSTATIONS

Agar snort beroperasi secara langsung setiap kali workstation / server di boot, kita dapat menambahkan ke file /etc/rc.d/rc.local perintah di bawah ini

/usr/local/bin/snort –d –h 192.168.0.0/24 –c /root/snort/snort.conf –A full –s –D

atau

/usr/local/bin/snort –d –c /root/snort/snort.conf –A full –s –D

dimana –D adalah switch yang men-set agar snort bekerja sebagai Daemon (bekerja dibelakang layar).

Setup snort.conf

Secara umum ada beberapa hal yang perlu di-set pada snort.conf, yaitu:

- Set konfigurasi dari jaringan kita.
- Konfigurasi pemrosesan sebelum di lakukan proses deteksi penyusup.
- Konfigurasi output.
- Konfigurasi rule untuk melakukan deteksi penyusup.

Secara umum kita terutama perlu menset konfigurasi jaringan saja, sedang setting lainnya dapat dibiarkan menggunakan default yang ada. Khususnya konfigurasi rule jika ingin gampang kita ambil saja contoh file *.rules yang ada di snort.tar.gz.

Konfigurasi jaringan yang perlu dilakukan sebetulnya tidak banyak, hanya mengisi

```
var HOME_NET 192.168.0.0/24
```

memberitahukan snort IP jaringan lokal-nya adalah 192.168.0.0/24 (satu kelas C). Sisanya dapat di diamkan saja menggunakan nilai default-nya.

Bagian konfigurasi output dapat kita mainkan sedikit untuk menset kemana alert & informasi adanya portscan di kirim, secara default akan dimasukan ke /var/log/snort. Sedikit modifikasi perlu dilakukan jika kita menginginkan untuk menggunakan ACID untuk menganalisa alert yang ada. Output perlu dimasukan ke database, seperti MySQL.

Rules biasanya terdapat pada file *.rules. Untuk mengedit sendiri rules agak lumayan, kita membutuhkan pengetahuan yang dalam tentang protokol, payload serangan dll. Untuk pemula sebaiknya menggunakan contoh *.rules yang di sediakan oleh snort yang dapat langsung dipakai melalui perintah include pada snort.conf.

Beberapa contoh rules dari serangan / eksploit dapat dilihat berikut ini,

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"EXPLOIT ssh CRC32 overflow /bin/sh"; flags:A+; content:"/bin/sh"; reference:bugtraq,2347; reference:cve,CVE-2001-0144; classtype:shellcode-detect; sid:1324; rev:1;)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"EXPLOIT ssh CRC32 overflow NOOP"; flags:A+; content:"|90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90|"; reference:bugtraq,2347; reference:cve,CVE-2001-0144; classtype:shellcode-detect; sid:1326; rev:1;)
```

Cukup rumit bagi pemula, detail berbagai parameter rules terdapat di SnortUsersManual.pdf yang juga di sediakan bersama source snort.tar.gz.

A. Setting open vpn

1. download dan instalasi

a. linux ubuntu

Dengan level user root, gunakan perintah

apt-get install openvpn

setelah proses selesai konfigurasi openvpn akan berada di /usr/share/doc/openvpn/examples.

agar file konfigurasi asli tidak rusak atau berubah saat kita lakukan konfigurasi, pindahkan folder examples ke /etc/openvpn, yang otomatis terbuat saat downloading dan instalasi pada apt-get install.

b. win32

download file instalasi openvpn windows di http://openvpn.net/download.html.

akan berupa openvpn-2.0.9-install.exe.

jalankan installer dan ikuti saja petunjuk yang ada, atau mudahnya klik saja next hingga selesai.

pada windows akan terbuat shortcut pada start menu yang berisi software openvpn.

2. konfigurasi

Pada tahap ini, akan dilakukan konfigurasi pada linux terlebih dahulu, karena openvpn tergolong cross platforn software, sehingga bisa dioperasikan pada linux dan windows, maka tidak akan sulit, hanya akan ada penyesuaian pada masing masing platform.

a. linux ubuntu

setelah mengkopi folder examples tadi, masuk kefolder tersebut /etc/openvpn/examples. akan ada3 buah folder lagi yaitu easy-rsa, sample-config-file, sample-keys.

file file konfigurasi yang dibutuhkan berada pada folder easy-rsa/2.0/. folder ini akan berisi :

build-ca build-key-pass build-req-pass list-crl pkitool vars
build-dh build-key-pkcs12 clean-all Makefile README.gz whichopensslcnf
build-inter build-key-server inherit-inter openssl-0.9.6.cnf.gz revoke-full
build-key build-req keys openssl.cnf sign-req

file konfigurasi dasar adalah vars, dimana berisi informasi penting openvpn yang akan di buat, dalam hal ini openvpn akan di konfigurasi menggunakan easy-rsa :

```
# easy-rsa parameter settings

# NOTE: If you installed from an RPM,
# don't edit this file in place in
# /usr/share/openvpn/easy-rsa --
# instead, you should copy the whole
# easy-rsa directory to another location
# (such as /etc/openvpn) so that your
# edits will not be wiped out by a future
# OpenVPN package upgrade.
```

```
# This variable should point to
# the top level of the easy-rsa
# tree.
export D=`pwd`

# This variable should point to
# the openssl.cnf file included
# with easy-rsa.
export KEY_CONFIG=$D/openssl.cnf

# Edit this variable to point to
# your soon-to-be-created key
# directory.
#
# WARNING: clean-all will do
# a rm -rf on this directory
# so make sure you define
# it correctly!
export KEY_DIR=$D/keys

# Issue rm -rf warning
echo NOTE: when you run ./clean-all, I will be doing a rm -rf on $KEY_DIR

# Increase this to 2048 if you
# are paranoid.  This will slow
# down TLS negotiation performance
# as well as the one-time DH parms
# generation process.
export KEY_SIZE=1024
```

These are the default values for fields
which will be placed in the certificate.
Don't leave any of these fields blank.
export KEY_COUNTRY=KG
export KEY_PROVINCE=NA
export KEY_CITY=BISHKEK
export KEY_ORG="OpenVPN-TEST"
export KEY_EMAIL="me@myhost.mydomain"

Konfigurasi yang dibutuhkan hanya pada baris export KEY_SIZE=1024 dan lima baris terakhir.
Jika ingin openvpn lebih secure dan lebih lambat ubah export KEY_SIZE=1024 menjadi 2048, (if you are paranoid). ubah lima baris terakhir sesuai kebutuhan.
ubah dengan perintah
vi vars atau vim vars atau menggunakan editor lain
jika sudah jalankan file vars tersebut

. ./vars

./clean-all

perintah clean-all akan menghapus semua konfigurasi lama yang pernah dibuat pada folder keys

./build-ca

perintah build-ca untuk membuat sertifikat openvpn yang baru pada folder keys, akan muncul sekuen :

Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'ca.key'

You are about to be asked to enter information that will be incorporated
into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [KG]:
State or Province Name (full name) [NA]:
Locality Name (eg, city) [BISHKEK]:
Organization Name (eg, company) [OpenVPN-TEST]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:OpenVPN-CA
Email Address [me@myhost.mydomain]:

semua informasi pada [ ] adalah informasi yang ada pada vars yang sudah di edit tadi, tekan enter saja untuk pengisian default (sama dengan isi [ ]), kecuali seksi common name harus diisi manual sesuai dengan nama yang akan kita berikan pada sertifikat CA openvpn, kalau tidak proses akan gagal.
Akan ada file sertifikat openvpn (ca) pada folder keys.
Setelah itu akan dibuat keyt untuk server openvpn, dengan menggunakan perintah

./build-key-server server

Perintah ini akan membuat key openvpn-server bernama server, dengan menggunakan nama "server" akan berfungsi untuk autentikasi lebih lanjut.

Generating a 1024 bit RSA private key
.....++++++
...++++++
writing new private key to 'VPN-Server.key'

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [DE]:
State or Province Name (full name) [BY]:
Locality Name (eg, city) [Regensburg]:
Organization Name (eg, company) [Feilner-IT]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:server
Email Address [security@feilner-it.net]:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /usr/share/doc/openvpn/examples/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName :PRINTABLE:'DE'
stateOrProvinceName :PRINTABLE:'BY'
localityName :PRINTABLE:'Regensburg'
organizationName :PRINTABLE:'Feilner-IT'
commonName :PRINTABLE:'server'
emailAddress :IA5STRING:'security@feilner-it.net'
Certificate is to be certified until Nov 17 23:40:04 2015 GMT (3650 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated

Pada seksi 'A challenge password' dan 'An optional company name' bisa dikosongkan, dan yang penting adalah 2 pernyataan terakhir harus berisi argumen positif atau 'y'.

Setelah tahap ini selesai maka akan ada file sertifikat server pada folder keys

```
ls -l keys
```

```
-rw-r--r-- 1 root root 3653 2005-11-20 00:40 01.pem
-rw-r--r-- 1 root root 1233 2005-11-20 00:39 ca.crt
-rw------- 1 root root 887 2005-11-20 00:39 ca.key
-rw-r--r-- 1 root root 104 2005-11-20 00:40 index.txt
-rw-r--r-- 1 root root 21 2005-11-20 00:40 index.txt.attr
-rw-r--r-- 1 root root 0 2005-11-20 00:31 index.txt.old
-rw-r--r-- 1 root root 3 2005-11-20 00:40 serial
-rw-r--r-- 1 root root 3 2005-11-20 00:31 serial.old
-rw-r--r-- 1 root root 3653 2005-11-20 00:40 server.crt
-rw-r--r-- 1 root root 688 2005-11-20 00:40 server.csr
-rw------- 1 root root 887 2005-11-20 00:40 server.key
```

Setelah membuat sertifikat server, kemudian membuat sertifikat untuk client, dengan menggunakan perintah

```
./build-key client
```

Proses pembuatan akan sama dengan saat membuat key server, bagian yang terpenting adalah pada Common Name, harus diisi manual, dan jika akan membuat client key lebih dari 1 Common Name masing-masing client harus berbeda.

Setelah selesai kemudian membuat Diffie Hellman parameters dengan menggunakan perintah

```
./build-dh
```

```
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
.....+.....
```

.....+.....+.....+.....

.....

Durasi proses tergantung pada isi variabel export KEY_SIZE pada vars, semakin besar isinya (2048) semakin lama prosesnya.

Jika sudah selesai maka konfigurasi dasar sertifikat dan key pada openvpn manggunakan metode easy-rsa selesai.semua file yang dibutuhkan akan ditampung pada folder keys.

Berikut adalah contoh tabel distribusi file sertifikat dan keys

Filename	Needed By	Purpose	Secret
ca.crt	server + all clients	Root CA certificate	NO
ca.key	key signing machine only	Root CA key	YES
dh{n}.pem	server only	Diffie Hellman parameters	NO
server.crt	server only	Server Certificate	NO
server.key	server only	Server Key	YES
client1.crt	client1 only	Client1 Certificate	NO
client1.key	client1 only	Client1 Key	YES
client2.crt	client2 only	Client2 Certificate	NO
client2.key	client2 only	Client2 Key	YES
client3.crt	client3 only	Client3 Certificate	NO
client3.key	client3 only	Client3 Key	YES

Dengan ini administrator dapat menentukan hak akses pada setiap file tersebut. kemudian agar lebih cantik copy folder keys ke /etc/openvpn, gunakan perintah

openvpn --genkey --secret ta.key

untuk membuat authentication key server, yang digunakan sebagai tambahan kunci koneksi antara server dan client openvpn. gunakan perintah saat berada di /etc/openvpn (menjadi satu dengan server.conf).

Semua file diatas hanya sekedar dasar dan kebutuhan untuk openvpn saja, harus dibuat konfigurasi sendiri untuk server dan client.

MEMBUAT KONFIGURASI SERVER DAN CLIENT

Untuk lebih mudahnya copy file server.conf dari folder examples/sample-config-files ke folder /etc/openvpn, jika masih dalam bentuk .gz dekompres dulu dengan

gzip -d server.conf.gz

buka server.conf dengan editor

```
##################################################
# Sample OpenVPN 2.0 config file for             #
# multi-client server.                           #
#                                                #
# This file is for the server side               #
# of a many-clients <-> one-server               #
# OpenVPN configuration.                             #
#                                                #
# OpenVPN also supports                              #
# single-machine <-> single-machine              #
# configurations (See the Examples page          #
# on the web site for more info).                #
#                                                #
# This config should work on Windows             #
# or Linux/BSD systems.  Remember on             #
# Windows to quote pathnames and use             #
```

```
# double backslashes, e.g.:                                    #
# "C:\\Program Files\\OpenVPN\\config\\foo.key"                #
#                                                              #
# Comments are preceded with '#' or ';'                        #
################################################

# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one.  You will need to
# open up this port on your firewall.
port 1194

# TCP or UDP server?
;proto tcp
proto udp

# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
```

```
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one.  On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key).  Each client
# and the server must have their own cert and
# key file.  The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys.  Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
```

```
cert server.crt
key server.key  # This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh1024.pem 1024
# Substitute 2048 for 1024 if you are using
# 2048 bit keys.
dh dh1024.pem

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0

# Maintain a record of client <-> virtual IP address
# associations in this file.  If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist ipp.txt

# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface.  Then you must manually set the
# IP/netmask on the bridge interface, here we
```

```
# assume 10.8.0.4/255.255.255.0.  Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients.  Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100

# Push routes to the client to allow it
# to reach other private subnets behind
# the server.  Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.
;push "route 192.168.10.0 255.255.255.0"
;push "route 192.168.20.0 255.255.255.0"

# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).

# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
# Then create a file ccd/Thelonious with this line:
```

```
#   iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN.  This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.

# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
#   ifconfig-push 10.9.0.1 10.9.0.2

# Suppose that you want to enable different
# firewall access policies for different groups
# of clients.  There are two methods:
# (1) Run multiple OpenVPN daemons, one for each
#     group, and firewall the TUN/TAP interface
#     for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
#     modify the firewall in response to access
#     from different clients.  See man
#     page for more info on learn-address script.
;learn-address ./script

# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
```

```
# (The OpenVPN server machine may need to NAT
# the TUN/TAP interface to the internet in
# order for this to work properly).
# CAVEAT: May break client's network config if
# client's local DHCP server packets get routed
# through the tunnel.  Solution: make sure
# client's local DHCP server is reachable via
# a more specific route than the default route
# of 0.0.0.0/0.0.0.0.
;push "redirect-gateway"

# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses.  CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
;push "dhcp-option DNS 10.8.0.1"
;push "dhcp-option WINS 10.8.0.1"

# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
;client-to-client

# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names.  This is recommended
# only for testing purposes.  For production use,
# each client should have its own certificate/key
```

```
# pair.
#
# IF YOU HAVE NOT GENERATED INDIVIDUAL
# CERTIFICATE/KEY PAIRS FOR EACH CLIENT,
# EACH HAVING ITS OWN UNIQUE "COMMON NAME",
# UNCOMMENT THIS LINE OUT.
;duplicate-cn

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0 # This file is secret

# Select a cryptographic cipher.
```

```
# This config item must be copied to
# the client config file as well.
;cipher BF-CBC        # Blowfish (default)
;cipher AES-128-CBC   # AES
;cipher DES-EDE3-CBC  # Triple-DES

# Enable compression on the VPN link.
# If you enable it here, you must also
# enable it in the client config file.
comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
;user nobody
;group nobody

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
```

```
# current connections, truncated
# and rewritten every minute.
status openvpn-status.log
# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it.  Use one
# or the other (but not both).
;log         openvpn.log
;log-append  openvpn.log

# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 3

# Silence repeating messages.  At most 20
# sequential messages of the same message
# category will be output to the log.
;mute 20
```

edit isi file ini sesuai kebutuhan.

sama halnya dengan file konfigurasi untuk client, supaya terorganisir terlebih dahulu buat folder 'configs' pada /etc/openvpn, untuk menyimpan semua file key

untuk client. setelah itu copy client.conf dari sample-config-file ke folder 'configs' yang di buat tadi. buka dengan editor

```
##############################################
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.     #
#                                            #
# This configuration can be used by multiple #
# clients, however each client should have   #
# its own cert and key files.                #
#                                            #
# On Windows, you might want to rename this  #
# file so it has a .ovpn extension           #
##############################################

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one.  On XP SP2,
# you may need to disable the firewall
```

```
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
;proto tcp
proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote my-server-1 1194
;remote my-server-2 1194

# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific local port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
```

```
;user nobody
;group nobody

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here.  See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets.  Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
ca ca.crt
cert client.crt
key client.key
```

```
# Verify server certificate by checking
# that the certicate has the nsCertType
# field set to "server".  This is an
# important precaution to protect against
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the nsCertType
# field set to "server".  The build-key-server
# script in the easy-rsa folder will do this.
;ns-cert-type server

# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
;cipher x

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
comp-lzo

# Set log file verbosity.
verb 3

# Silence repeating messages
```

;mute 20

edit sesuai kebutuhan.

Copy semua file dan key yang dibutuhkan client dari folder keys ke folder configs. kemudian distribusikan configs tersebut ke client yang akan menggunakan openvpn.

b. win32

Karena server side adalah linux dan win32 berperan sebagai client, jadi hanya copy saja folder 'configs' tadi ke windows dengan menggunakan winscp atau dengan cara lain.