# Implementasi *OpenVPN Server* Untuk Koneksi *Remote* Pada Perangkat *Android*

P. U. Rompas, A.S.M. Lumenta, A. M. Rumagit, B. A. Sugiarso

**ABSTRAK**

**Saat ini berbagai aplikasi yang mampu mengontrol perangkat *android* secara *remote* dari *web browser* di internet dapat ditemukan dengan mudah. Namun demikian, alamat IP (*Internet Protocol*) yang ada pada perangkat *android* biasanya berubah-ubah, tergantung pada penyedia layanan internet. Masalah ini membuat pengguna *android* sulit untuk melakukan koneksi pada perangkat *android*. Beberapa aplikasi menggunakan server sendiri untuk melakukan koneksi ke perangkat *android*, tetapi pengguna tidak dapat memastikan keamanan data, baik yang dikirim maupun yang diterima melalui server tersebut.**

**Tugas Akhir ini akan mengembangkan sebuah server VPN (*Virtual Private Network*) pada sistem operasi berbasis linux. Server VPN ini dapat memberikan alamat IP yang tetap *(Static IP Address)* pada perangkat *android*. Alamat IP yang diberikan server VPN ini memperbolehkan pengguna *android* untuk melakukan akses *remote* secara langsung pada perangkat *android*, tanpa dibatasi oleh alamat IP tidak tetap (*Dynamic IP Address)* yang diberikan oleh penyedia layanan internet. Server ini juga dapat melakukan enkripsi data yang ditransfer melalui internet, sehingga pengguna dapat melakukan transfer data melalui internet dengan aman.**

**Hasil yang didapatkan dari pengembangan ini adalah sebuah VPN server yang memberikan alamat IP yang konstan (*Static IP Address)* pada perangkat *android*. Dengan demikian, pengguna dapat melakukan akses *remote* secara langsung pada perangkat *android*. Server ini memiliki *proxy server* yang dapat melakukan enkripsi data melalui HTTPS (*Hyper Text Transfer Protocol Secure*), sehingga koneksi dari *web browser* ke server VPN dapat dijalankan secara aman.**

**Kata Kunci:**
**Akses *Remote, Android, Internet, Virtual Private Network*.**

## I. PENDAHULUAN

### *A. Latar Belakang*

Untuk mengatasi tingginya kebutuhan *user* akan *smartphone* maka dibutuhkan suatu mekanisme yang mudah untuk dapat mengakses *smartphone* secara mudah dan praktis, tanpa dibatasi oleh jaringan yang ada. Untuk mengakses *smartphone* dengan mudah terutama perangkat berbasis *android*, tersedia banyak software gratis yang memudahkan user untuk mengakses perangkat *android* mereka dengan mudah, melalui jaringan *WiFi*. Namun biasanya aplikasi seperti ini terbatas pada jaringan lokal saja. Solusinya adalah membuat sebuah *server* yang memungkinkan akses ke perangkat *android* tanpa dibatasi oleh jaringan yang ada.

### *B. Identifikasi Masalah*

1. Merancang dan membuat *server* VPN.
2. Membuat Proxy Server untuk meneruskan akses ke perangkat android dan menjamin keamanan data yang di transfer.
3. Konfigurasi *remote server* pada *android* .
4. Konfigurasi VPN *Client* pada perangkat *android* agar dapat terhubung ke *server*.

### *C. Pembatasan Masalah*

1. *Software* yang digunakan dalam pembuatan *server* VPN ini yaitu *OpenVPN*.
2. Menggunakan *Nginx* sebagai *proxy server* untuk meneruskan akses *remote* ke perangkat *android.*
3. *Remote server* yang akan digunakan adalah *i-Jetty Webserver* dan *i-Jetty Console*, yang merupakan *webserver open source* untuk *android*.

### *D. Tujuan Penulisan*

Tugas Akhir ini memiliki tujuan untuk dapat memudahkan *user* untuk mengakses data yang ada dalam perangkat *android* melalui koneksi apa saja tanpa dibatasi oleh jaringan, baik *WiFi, 3G* atau 2G. Sehingga file-file yang terdapat dalam perangkat *android* dapat diambil melalui *web browser* kapan saja dan di mana saja selama ada koneksi internet. Selain itu pembuatan server ini juga bertujuan untuk menjamin keamanan data yang di transfer melalui internet.

## II. LANDASAN TEORI

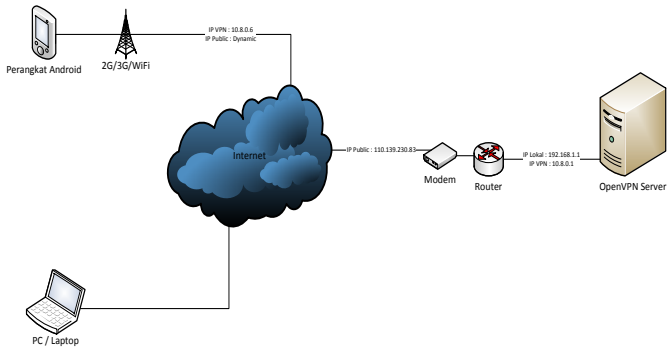
### *A. Virtual Private Network*

*Virtual Private Network* atau yang sering disebut dengan VPN adalah sebuah jaringan privat yang menggunakan infrastruktur telekomunikasi publik untuk saling bertukar informasi. Dengan adanya teknologi komunikasi ini, seseorang dapat terkoneksi ke jaringan publik serta menggunakannya untuk dapat bergabung dengan jaringan lokal. Dengan cara tersebut, maka akan didapatkan hak dan pengaturan yang sama seperti halnya berada di dalam kantor atau LAN itu sendiri, walaupun sebenarnya menggunakan jaringan milik publik. VPN dapat menghubungkan dua *end-system* atau dua komputer, atau pun antara dua atau lebih jaringan yang berbeda.

VPN merupakan perpaduan antara teknologi *tunneling* dan enkripsi Pada kedua ujung (*end system)* dari perangkat VPN ini biasanya telah menyepakati algoritma yang akan digunakan untuk melakukan proses dekripsi. Dalam dunia jaringan, *tunnel* diartikan sebagai suatu cara untuk meng-enkapsulasi atau membungkus paket IP di dalam paket IP yang lain.

### *B. OpenVPN*

*OpenVPN* adalah aplikasi *open source* untuk *Virtual Private Networking* (VPN), dimana aplikasi tersebut dapat membuat koneksi *point-to-point tunnel* yang telah terenkripsi. *OpenVPN* merupakan *full-featured* SSL VPN yang mengimplementasikan OSI *layer* 2 dan 3 *network extension* menggunakan standar SSL/TLS protokol, mendukung metode otentikasi klien berdasarkan sertifikat yang fleksibel, *smart card*, dan *username* / *password* serta memungkinkan pengguna atau kelompok tertentu melakukan akses kontrol terhadap kebijakan (*policies*) menggunakan aturan *firewall* yang diterapkan pada *interface* VPN *virtual*. *OpenVPN* bukan aplikasi *web proxy* dan tidak beroperasi melalui *web browser*.




Gambar 1. Topologi Jaringan

C. *Android*

*Android* adalah sistem operasi untuk telepon seluler yang berbasis *Linux*. *Android* menyediakan *platform* yang bersifat *open source* bagi para pengembang untuk menciptakan sebuah aplikasi.

*D. Router*

Menurut Handriyanto (2009), Router adalah perangkat yang akan melewatkan paket IP dari suatu jaringan ke jaringan yang lain, menggunakan metode *addressing* dan protokol tertentu untuk melewatkan paket data tersebut. Router memiliki kemampuan melewatkan paket IP dari satu jaringan ke jaringan lain yang mungkin memiliki banyak jalur diantara keduanya.

*E. Konsep Jaringan Komputer*

Menurut Riyanto (2011), Jaringan komputer adalah sekumpulan komputer yang terhubung satu dengan lainnya menggunakan protocol komunikasi melalui media komunikasi sehingga dapat menggunakan sumber daya bersama seperti harddisk, printer, dan sumber informasi lainnya

Tujuan dibangunnya jaringan computer adalah membawa suatu informasi secara tepat dan tanpa adanya kesalahan dari sisi pengirim menuju sisi penerima melalui media komunikasi.

## III. METODOLOGI PENELITIAN

*A. Desain Jaringan*

Server ini dirancang untuk menangani koneksi *remote* ke perangkat *android* menggunakan *OpenVPN Server* yang di instal pada *Linux Ubuntu 12.04.1 LTS 32bit.* Koneksi internet yang digunakan dalam pembuatan *server* ini adalah koneksi Telkom *Speedy*. *Server* ini berada pada jaringan lokal yang ada di belakang *router* berbasis *Mikrotik RouterOS* yang terhubung ke *internet* dan memiliki *fixed IP address*. *Router* ini dikonfigurasi untuk meneruskan koneksi TCP di *port* 3080 pada IP *external* ke *port* 3080 pada *Server* dimana *nginx* mendengarkan setiap TCP request yang masuk dan meneruskan ke IP *OpenVPN Client* di *port* 8080 pada perangkat *android* melalui *OpenVPN Server*, sehingga *i-Jetty* pada perangkat *android* dapat di akses secara *remote* menggunakan koneksi 3G maupun koneksi 2G. Pada Gambar 1 terlihat bagaimana perangkat *android* yang memiliki *Dynamic IP Address* dan IP VPN 10.8.0.6 terhubung ke *server* VPN dengan IP VPN 10.8.0.1, sehingga perangkat *android* dapat di akses melalui *OpenVPN Server* yang memiliki IP *Public* 110.139.230.83 dari ISP Telkom.

*B. Perangkat Percobaan*

Perangkat Lunak Yang Dibutuhkan

1. *Ubuntu Linux 12.04.1 LTS 32bit*
2. *OpenVPN Server 2.2.2*
3. *Nginx 1.3.3*
4. *MikroTik RouterOS™*
5. *SuperUser v3.1.3(46) with SU Binary v3.1.1(17)*
6. *Android Kernel 2.6.35.7-perf-CL882825 dragonn@arch #188*
7. *i-jetty 3.1*
8. *OpenVPN Installer for Android 0.2.4*
9. *OpenVPN Settings for Android 0.4.12*
10. *MikroTik WinBox Loader v2.2.18*
11. *PuTTY 0.61*
12. *DropBear SSH Server*
13. *Terminal Emulator*

Perangkat Keras Yang Dibutuhkan

1. *Motherboard Advance G41*
2. *Processor Intel Celeron 3.06 GHz,*
3. *Ram 1 GB DDR 2 PC 5400,*
4. *Power Supply Max Power 450Watt,*
5. *Hardisk Samsung 40 GB,*
6. *VGA Nvdia GForce 7200 GS,*
7. *Monitor Samsung SyncMaster 740N,*
8. *Casing SimX,*
9. *Keyboard Mouse SPC,*
10. *MikrotikOS Router*

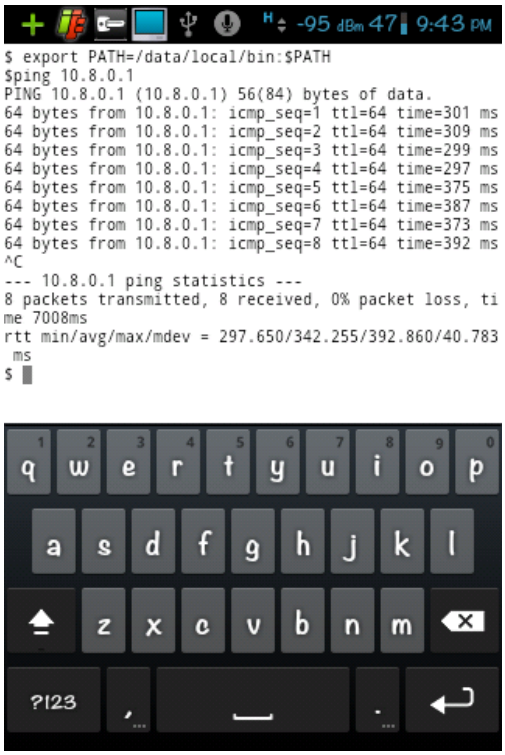
## IV. HASIL DAN PEMBAHASAN

*A. Pembuatan OpenVPN Server*

*OpenVPN Server* adalah program *opensource (freeware)* untuk membuat jalur dari *server* ke *multiclient* dengan saluran terenkripsi antara satu dengan yang lainnya. *OpenVPN* dapat menghasilkan hubungan langsung dengan beberapa komputer lewat NATs (*Network Address Translators*) dan *firewall*. *OpenVPN Server* ini dibuat berbasiskan *linux* dengan pertimbangan dari segi ekonomi dan dari segi teknis. Pertimbangan dari segi ekonomi yaitu kebanyakan distro *linux* bersifat gratis. Dan dari segi teknis yaitu *linux* lebih stabil jika dibandingkan dengan sistem operasi lainnya.

Langkah – langkah Pembuatan *OpenVPN Server* ini adalah sebagai berikut :

1. Instalasi *Linux Ubuntu 12.04.1 LTS 32 bit* dengan versi *kernel 2.6.38.6-smp*, instalasi *Ubuntu Linux* tidak akan diuraikan dalam jurnal ini.
2. Instalasi dan Konfigurasi *OpenVPN Server*, dalam hal ini penulis menggunakan *OpenVPN Server 2.2.2*. Serta pembuatan *certificate* untuk *OpenVPN Server* dan *OpenVPN Client*.
3. Instalasi dan Konfigurasi *nginx*, konfigurasi pada *nginx* untuk *port forwarding* dari internet ke *OpenVPN Server* serta pembuatan *certificate* untuk keamanan data.




Gambar 2. Pengujian Ping ke *OpenVPN Server* pada perangkat *Android*

Untuk menginstall *OpenVPN Server*, ketikkan perintah berikut pada *terminal:*

$ sudo apt-get install openvpn

Kemudian dilanjutkan dengan melakukan pembuatan *Certificate* dan *Key* untuk *OpenVPN Server* dengan langkah-langkah sebagai berikut :

$ sudo mkdir /etc/openvpn/easy-rsa/
$ sudo cp –r /usr/share/doc/openvpn/examples/easy-rsa/2.0/* /etc/openvpn/easy-rsa/
$ sudo pico /etc/openvpn/easy-rsa/vars

Lakukan konfigurasi file *vars* dengan mengisikan variabel-variabel yang ada sesuai dengan konfigurasi server yang akan dibuat. Setelah selesai melakukan konfigurasi, lanjutkan dengan membuat *Certificate Authority (CA).*

$ su
# source vars
# ./clean-all
# ./build-ca

Kemudian lanjutkan dengan membuat *Key*.

# ./build-key-server openvpnserver

Isikan sesuai konfigurasi server yang akan dibuat, setelah itu ketikkan perintah berikut untuk membuat *Diffie-Hellman Key*.

# ./build-dh

Setelah selesai membuat *certificate* dan *key* untuk *OpenVPN server*, langkah selanjutnya adalah membuat key untuk *client* dengan mengetikkan perintah :

# ./build-key client1

*B. Instalasi dan Konfigurasi Nginx*

*Nginx* digunakan sebagai *proxy server* dalam pembuatan server ini. *Nginx* bertugas untuk mengalihkan koneksi dari port 3080 pada server ke port 8080 pada IP VPN perangkat *android.* Untuk melakukan instalasi *nginx* pada *Ubuntu Linux* ketikkan perintah :

$ sudo apt-get install nginx

*C. Konfigurasi OpenVPN Client Pada Perangkat Android*

Untuk melakukan konfigurasi *OpenVPN Client* pada perangkat *android*, pastikan terlebih dahulu *OpenVPN* telah terinstal pada perangkat *android*. Langkah pertama yang harus dilakukan adalah memindahkan file "ca.crt", "client1.ovpn", "client1.key", "client1.crt" ke folder /sdcard/openvpn yang ada pada perangkat *android.* Setelah memindahkan file-file tersebut, edit file "client1.ovpn" menggunakan *text editor* yang ada pada perangkat *android* kemudian tambahkan line :

remote 110.139.230.xxx 1194

setelah melakukan konfigurasi *OpenVPN Client*, jalankan *OpenVPN Client* dengan membuka aplikasi *OpenVPN Settings* yang telah terinstal, kemudian centangkan opsi "OpenVPN" dan opsi "client1.ovpn".

*D. Pengujian Sistem*

Dalam melakukan pengujian sistem, penulis menggunakan menggunakan aplikasi *PuTTY* pada sistem operasi Windows 7. Aplikasi ini digunakan untuk menguji koneksi secara keseluruhan dengan melakukan akses *remote* melalui koneksi *SSH* ke *OpenVPN Server* dan kemudian dilanjutkan dengan melakukan akses *remote* dari *OpenVPN Server* ke perangkat *Android* yang telah terinstall *DropBear SSH Server.*

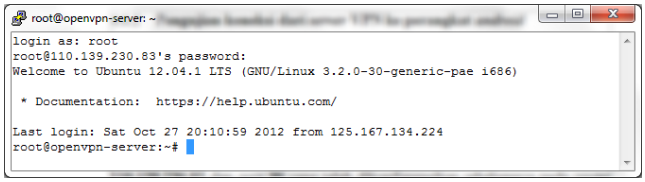
Untuk melakukan pengujian apakah perangkat *android* telah terhubung ke server *OpenVPN* yang telah dibuat, buka aplikasi *terminal emulator* kemudian lakukan ping ke server *OpenVPN* yang memiliki IP 10.8.0.1 dengan mengetikkan perintah :

ping 10.8.0.1

jika perangkat *android* telah terhubung ke server *OpenVPN*, maka akan didapat hasil seperti yang terlihat pada gambar 2 sebagai berikut :

64 bytes from 10.8.0.1 : icmp_seq=1 ttl=64 time=301ms
64 bytes from 10.8.0.1 : icmp_seq=2 ttl=64 time=309ms
64 bytes from 10.8.0.1 : icmp_seq=3 ttl=64 time=299ms
64 bytes from 10.8.0.1 : icmp_seq=4 ttl=64 time=297ms
64 bytes from 10.8.0.1 : icmp_seq=5 ttl=64 time=375ms
64 bytes from 10.8.0.1 : icmp_seq=6 ttl=64 time=387ms
64 bytes from 10.8.0.1 : icmp_seq=7 ttl=64 time=373ms
64 bytes from 10.8.0.1 : icmp_seq=8 ttl=64 time=392ms


Gambar 3. Konfigurasi *PuTTY*




Gambar 4. Koneksi *remote* ke *OpenVPN Server* menggunakan *PuTTY*

--- 10.8.0.1 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7008ms

rtt min/avg/max/mdev = 297.650/342.255/392.860/40.783 ms

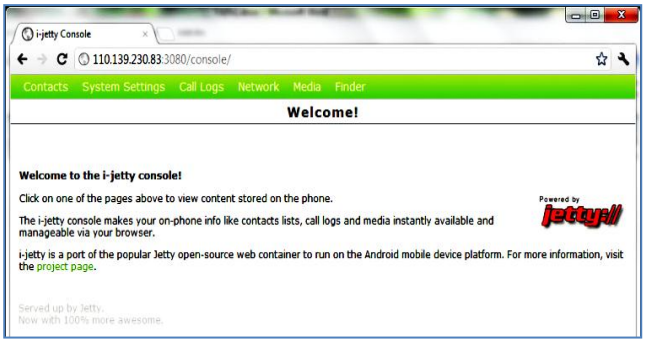
Setelah perangkat *android* telah benar-benar terhubung ke server, langkah berikutnya adalah melakukan pengujian koneksi ke server dengan cara membuka aplikasi *PuTTY* kemudian masukkan IP dan *port* server yang akan di akses, dalam hal ini, penulis menggunakan IP **110.139.XXX.XXX** dan *port* **90** yang telah dikonfigurasikan sebelumnya pada *router* sebagai akses masuk SSH ke *server Ubuntu.* Kemudian pilih connection type "SSH" seperti pada gambar 3.
Jika dilakukan dengan benar, akan muncul tampilan login ke server VPN. Masukkan *username* dan *password*, jika dilakukan dengan benar akan muncul tampilan seperti pada gambar 3. setelah itu lakukan pengujian ping ke perangkat *android* dengan perintah:

ping 10.8.0.6

Jika perangkat *android* telah terhubung ke server, maka akan di dapatkan hasil sebagai berikut :

64 bytes from 10.8.0.6 : icmp_seq=1 ttl=64 time=445ms
64 bytes from 10.8.0.6 : icmp_seq=2 ttl=64 time=402ms
64 bytes from 10.8.0.6 : icmp_seq=3 ttl=64 time=439ms
64 bytes from 10.8.0.6 : icmp_seq=4 ttl=64 time=400ms


Gambar 5. Koneksi *remote* ke perangkat *android* melalui *OpenVPN Server* menggunakan *PuTTY*




Gambar 6. Tampilan aplikasi *i-Jetty Console*

--- 10.8.0.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms

rtt min/avg/max/mdev = 400.505/422.159/445.761/20.695 ms

langkah selanjutnya adalah menguji apakah perangkat *android* sudah dapat di akses secara remote dengan mengetikkan perintah:

ssh root@10.8.0.6

setelah mengetikkan perintah tersebut tekan 'enter', maka akan muncul tampilan login dari *DropBear SSH Server* yang terinstal pada perangkat *android* seperti pada gambar 5. Berikutnya adalah menguji apakah *i-Jetty* telah berjalan pada perangkat *android* kita dapat mengaksesnya lewat *browser* dengan mengetikkan IP server sebagai berikut :

http://110.139.230.83:3080/console/

Jika server telah berjalan dengan baik, maka akan muncul tampilan *i-Jetty* seperti pada Gambar 6. Jika *i-Jetty* telah berjalan dengan baik, maka kita sudah dapat melakukan koneksi *remote* ke perangkat *android* hanya dengan menggunakan *browser* pada perangkat apa saja yang terhubung ke internet.

Untuk menguji efisiensi penggunaan *OpenVPN Server* untuk koneksi remote ke perangkat *android*, maka kita perlu melakukan uji-coba *download* dan *upload* data ke perangkat *android* dengan berbagai macam koneksi untuk melihat kecepatan transfer data pada perangkat *android* yang terkoneksi dengan jaringan 3.5G/3G (HSDPA/UMTS), 2.5/2G (EDGE/GPRS) dan WiFi.

TABEL I. PERBANDINGAN KECEPATAN *UPLOAD*

| **Koneksi Yang Digunakan pada perangkat *android*** | **Waktu Yang Dibutuhkan untuk *upload*** |
|---|---|
| **WiFi** (down 1 mbps / up 256 kbps) | 2 Menit 11 Detik |
| **3G/HSDPA (**down 384 kbps / up 128kbps) | 2 Menit 20 Detik |
| **2G / EDGE** (down 128 kbps/ up 96kbps) | 4 Menit 30 Detik |

TABEL II. PERBANDINGAN KECEPATAN *DOWNLOAD*

| **Koneksi Yang Digunakan pada perangkat *android*** | **Waktu Yang Dibutuhkan untuk *download*** |
|---|---|
| **WiFi** (down 1 mbps / up 256 kbps) | 2 Menit 15 Detik |
| **3G/HSDPA (**down 384 kbps / up 128kbps) | 4 Menit 30 Detik |
| **2G / EDGE** (down 128 kbps/ up 96kbps) | 5 Menit 57 Detik |

Pada tabel 1 dan tabel 2 dapat dilihat perbandingan kecepatan *upload* dan *download* pada perangkat *android*, dengan file sebesar 4.310Kb. Koneksi yang digunakan pada *server OpenVPN* adalah koneksi Telkom Speedy dengan kecepatan *upload* 256 *kbps* dan kecepatan *downstream* 3 *mbps*. Sedangkan koneksi yang digunakan pada perangkat *android* adalah koneksi Telkomsel Flash dan juga koneksi *WiFi* dengan kecepatan 1 *mbps* untuk *download* dan 256 *kbps* untuk *upload.*

## V. PENUTUP

### *A. Kesimpulan*

Dari pengujian yang dilakukan terhadap Tugas Akhir ini, maka kesimpulan yang didapat adalah sebagai berikut :

1. Sistem yang telah dibuat dapat menghubungkan perangkat *android* dengan *OpenVPN Server* sehingga perangkat *android* dapat diakses dari mana saja melalui koneksi 3G/2G/WiFi
2. Sistem yang dibuat dapat melakukan transfer file / data dari perangkat *android* ke pengguna.
3. Kecepatan koneksi bergantung pada besarnya kecepatan *upstream* koneksi internet pada *OpenVPN Server*.
4. Secara normal sistem ini berjalan selama koneksi internet pada perangkat *android* tidak terputus dan tetap terhubung ke server.
5. Sistem ini dapat berjalan dengan baik untuk menagani download file-file media yang berukuran dibawah 5 Mb,
6. Sistem ini dapat dimanfaatkan untuk keperluan koneksi *remote* ke perangkat *android*.

### *B. Saran*

Dari sistem yang telah dibuat dapat diberikan saran-saran untuk mengembangkan tugas akhir ini sebagai berikut :

1. Untuk upload dan *download* data yang lebih besar, hendaknya menggunakan koneksi internet yang lebih baik untuk *Server*, maupun perangkat *android.*
2. Koneksi dari browser ke server *OpenVPN* tidak di-enkripsi, sebaiknya membuat *certificate* kemudian meng-instruksikan *Nginx* untuk menjawab dengan koneksi SSL menggunakan *certificate* yang telah dibuat.
3. Sistem ini masih memiliki kelemahan jika digunakan di komputer rumah dengan IP Dinamis, maka user akan mengalami kesulitan dalam mengidentifikasi IP *Server*.

## DAFTAR PUSTAKA


[1] A. Riyanto, Pengenalan Jaringan, LIPI, 2011.

[2] C. Nedelcu, *Nginx HTTP Server*, Packt Publishing LTD, Birmingham, 2010.

[3] D. A. W. Wahyudi, Implementasi *Virtual Private Network Server* Menggunakan *Slackware 13* Untuk Keamanan Komunikasi Data (Studi Kasus : PT. Time Excelindo ISP), Sekolah Tinggi Manajemen Informatika dan Komputer, Yogyakarta, 2011.

[4] D.F. Handriyanto, Kajian Penggunaan *Mikrotik Router OS*™ Sebagai *Router* Pada Jaringan Komputer, Jurnal Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya, 2009.

[5] F. Schaeuffelhut, *Android OpenVPN Installer Project* hosting on Google Code. 2012. Tersedia di : http://code.google.com/p/android-openvpn-installer/

[6] F. Schaeuffelhut. *Android OpenVPN Settings Project* hosting on Google Code. 2012.Tersedia di : http://code.google.com/p/android-openvpn-settings/

[7] J. Bartel, *i-Jetty* : *Webserver for The Android Mobile Platform Project* hosting on Google Code. 2012.Tersedia di : http://code.google.com/p/i-jetty/

[8] W. Stallings, *Data and Computer Comunication 5th Edition. Prentice Hall*, USA, 1997.

[9] Z. Arifin, Mengenal *Wireless* LAN, Penerbit Andi, Yogyakarta, 2005.

[10] M. Feilner, *OpenVPN: Building and Integrating Virtual Private Networks*, Packt Publishing LTD,. Birmingham, 2006.

[11] Tentang modifikasi *kernel android* tersedia di : http://code.google.com

[12] Tentang *nginx* tersedia di : http://nginx.org

[13] Tentang *OpenVPN* tersedia di : http://www.openvpn.net

[14] Tentang *Virtual Private Networking (VPN)* Tersedia di : http://library.binus.ac.id/eColls/eThesis/Bab2/2011-1-00316-if%202.pdf