e-ISSN: 2549-7952
p-ISSN: 2580-3336

Implementasian Raspberry Sebagai *Cloud Server* Untuk Media Penyimpanan Alternatif

Bagus Satrio[1], Robby Candra[2]
[1,2]Sistem Komputer, Fakultas Ilmu Komputer Dan Teknologi Informasi Universitas Gunadarma
E-mail: *[1]bagus.satrio@student.gunadarma.ac.id, [2]robby.c@staff.gunadarma.ac.id

Abstrak – *Data dan informasi yang dikirim dan diterima semakin banyak seiiring dengan semakin berkembangnya perangkat telekomunikasi. Data dan informasi tersebut harus dapat diakses dari mana saja dan kapan saja dan membutuhkan kapasitas media penyimpanan yang besar sehingga diperlukan media penyimpanan yang dapat memenuhi kebutuhan dalam mengakses data tersebut seperti pemanfaatan cloud computing. Tujuan mengimplementasikan Cloud server menggunakan Raspberry yaitu dapat memberikan alternatif untuk memanfaatkan media penyimpanan bagi pengguna pribadi untuk mengakses data dari mana saja dan dan kapan saja serta dapat mengurangi resiko dari kehilangan data–data penting yang disebabkan karena kerusakan media penyimpanan konvensional. Raspberry bisa menjadi sebuah infrastruktur yang dapat membantudalam masalah penyimpanan data dengan mengoptimalkannya sebagai cloudserver. File yang disimpan dalam cloud server tidak hanya dapat diakses antara administrator dengan user saja melainkan juga dapat diakses antara user dengan user. Aplikasi cloud server menggunakan raspberry dapat membantu dalam mengakses data dan informasi dari mana saja dan kapan saja dan dapat memberikan alternatif media penyimpanan yang dapat digunakan untuk menyimpan data serta mengurangi resiko kehilangan data karena kerusakan media penyimpanan konvensional.*

Kata Kunci *— Cloud server, Data, Raspberry*

1. PENDAHULUAN

Di Era modern ini khusunya pada bidang IT media penyimpanan menjadi sebuah bagian yang krusial, karena saat ini data dan informasi yang dikirim dan diterima semakin banyak seiiring dengan semakin berkembangnya perangkat telekomunikasi. Data dan informasi tersebut harus dapat diakses dari mana saja dan kapan saja dan membutuhkan kapasitas media penyimpanan yang besar, sehingga diperlukan media penyimpanan yang dapat memenuhi kebutuhan dalam mengakses data tersebut seperti pemanfaatan *cloud computing*. *Cloud computing* merupakan salah satu bentuk transformasi teknologi informasi dan komunikasi, *Cloud computing* sendiri memiliki beragam layanan, salah satunya *cloud storage* [2]. *Cloud Computing* adalah sebuah model untuk kenyamanan, akses jaringan *on-demand* untuk menyatukan pengaturan konfigurasi sumber daya komputasi (seperti, jaringan, server, media penyimpanan, aplikasi, dan layanan) yang dapat dengan cepat ditetapkan dan dirilis dengan usaha manajemen yang minimal atau interaksi dengan penyedia layanan [1].

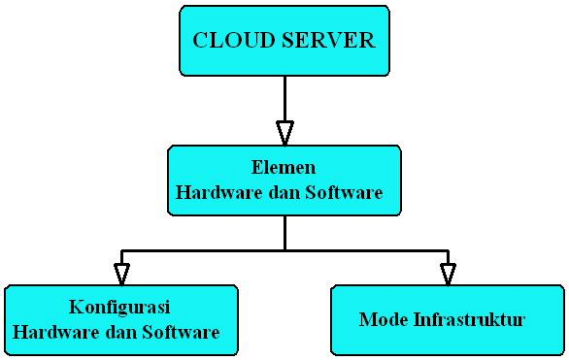
Teknologi *cloud computing* dihadirkan sebagai upaya untuk memungkinkan akses sumber daya dan aplikasi dari mana saja melalui jaringan Internet, sehingga keterbatasan pemanfaatan infrastruktur ICT yang sebelumnya ada dapat diatasi [1]. Selain itu menyimpan data pada media penyimpanan konvensional seperti flashdisk atau harddisk terakadang bisa saja media penyimpanan yang digunakan tersebut tersebut mengalami kerusakan. Sebuah data dan informasi yang tergabung dalam sistem informasi yang memiliki banyak kegunaan jika mempunyai suatu server yang baik, dalam artian server yang memiliki tingkat keamanan data yang tinggi seperti dapat melindungi data dari kerusakan media penyimpanan [3].

Berdasarkan pada uraian tersebut di atas, maka pada penelitian ini akan dibahas tentang pengaplikasian *cloud computing* menggunakan Raspberry sebagai *cloud server*. Alasan penggunaan raspberry sebagai *cloud server* yaitu raspberry merupakan sistem atau sumber daya yang sangat kecil, dapat diakses secara *online.*Perkembangan dari sistem *cloud computing* dengan sistem menyewa sebuah *resource* berharga tinggi dapat dipangkas secara signifikan melalui penggunaan r*esource* kecil, dalam hal ini *Raspberry* merupakan evolusi dari beberapa *resource* di zaman dahulu yang membutuhkan tempat yang besar [4]. Tujuan penelitian ini mengimplementasikan *Cloud server* menggunakan *Raspberry,* dengan harapan dapat memberikan alternatif untuk memanfaatkan media penyimpanan bagi pengguna pribadi untuk mengakses data dari mana saja dan dan kapan saja serta dapatmengurangi resiko dari kehilangan data–data penting yang disebabkan karena kerusakan media penyimpanan konvensional.

2. METODE PENELITIAN

Perancangan *cloud server* menggunakan raspberry seperti yang ditunjukan oleh skema metodelogi penelitian pada gambar 1.

e-ISSN: 2549-7952
p-ISSN: 2580-3336



Gambar 1. Skema metodelogi perancangan *cloud server* menggunakan raspberry

Untuk membangun sebuah cloud server pada jaringa lokal ada beberapa perangkat yang dibutuhkan, diantaranya untuk hardware :

- Kabel UTP beserta RJ-45
- Raspberry pi 3 type B
- Adaptor 2 A
- Micro SD 32 GB (ukuran disesuaikan)

Sedangkan software yang digunakan adalah *ownCloud* yang merupakan sebuah perangkat lunak yang digunakan untuk berbagi berkas.

Sebuah jaringan membutuhkan sebuah infrastruktur yang baik, yang dimaksud sebagai infrastruktur disini adalah sebuah topologi. Topologi ini akan menghubungkan antara *client* dan *cloud server* sehingga dapat saling berhubungan melalui jaringan kabel ataupun nirkabel seperti yang ditunjukan pada gambar 2. Topologi ini akan menghubungkan raspberry pi sebagai server menuju *access point* sebagai penyebar jaringan. Hal ini memungkinkan admin dan para *client* yang terhubung pada jaringan *wireless* dapat mengakses *cloud server*.

Gambar 2. Mode infrastruktur *cloud*

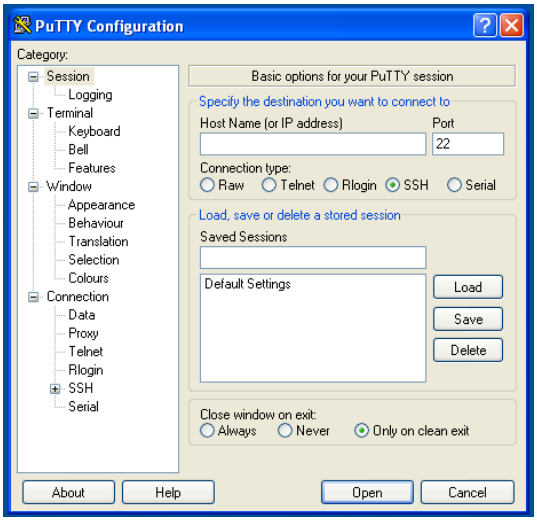
Pada konfigurasi hardware dan sotware, untuk memudahkan konfigurasi raspberry pi dapat dilakukan dengan me-*remote* melalui jaringan oleh komputer administrator. Sebelum melakukan *remote*, raspberry pi harus memiliki raspbian yang terinstall pada *memory card storage*. Setelah dipastikan bahwa raspberry sudah memiliki raspbian maka yang perlu dilakukan yaitu menset IP address yang akan digunakan oleh raspberry untuk disamakan satu *class* dengan komputer administrator. Letak file IP address pad raspberry terletak didalam memory card storage yaitu cmdline.txt.

dwc_otg.lpm_enable=0
console=ttyAMA0,115200
kgdboc=ttyAMA0,115200console=tty1
root=/dev/mmcblk0p2 rootfstype=ext4
elevator=deadline rootwait
ip=192.168.137.10

IP address 192.168.37.10 merupakan IP yang digunakan oleh raspberry, IP address nantinya akan digunakan untuk me-*remote* raspberry pi pada komputer administrator. IP address juga harus disesuaikan dengan IP pada router sehingga semua yang terhubung dengan router dapat mengakses *cloud* server nantinya.

3. HASIL DAN PEMBAHASAN

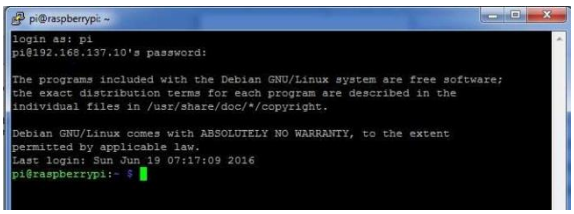
Untuk mengakses raspberry dengan cara me-*remote* dapat menggunakan berbagai macam tool. Tool yang banyak digunakan dikalangan administrator jaringan adalah putty. Putty merupakan tool remote admin yang powerfull, userfriendly, dan beberapa fitur yang dapat membantu seorang administrator. Protokol yang digunakan untuk me-*remote* raspberry adalah protokol SSH. SSH adalah protokol yang digunakan untuk melapisi pengiriman data seperti HTTP, FTP dan SMTP sehingga koneksi menjadi aman dan juga untuk bisa mengakses situ –situs yang menggunakan protokol SSL Untuk melakukan remote pada raspberry pi via protokol SSH diharuskan untuk mengisi IP address yang akan dituju yaitu IP address dari raspberry. Setelah itu isi pada bagian port SSH yang di gunakan, kemuian klik open untuk melanjutkan. Konfigurasi menggunakan putty ini seperti yang ditunjukan pada gambar 3.



Gambar 3. Konfigurasi Putty

e-ISSN: 2549-7952
p-ISSN: 2580-3336

Setelah komunikasi antar perangkat terjalin, putty akan menampilkan shell prompt seperti yang ditunjukan pada gambar 4 dari sistem operasi raspberry yang mengaharuskan administrator melakukan autentikasi kedalam sistem. Administrator harus memiliki hak akses penuh untuk melakukan konfigurasi pada sistem yang ada.

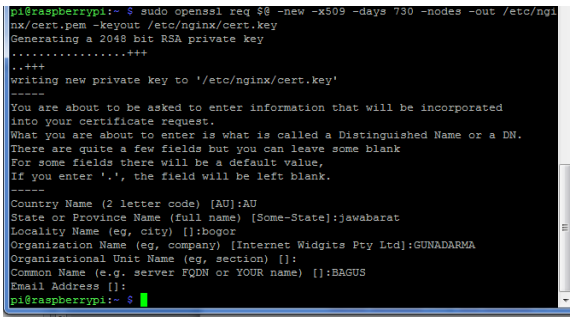


Gambar 4. Tampilan *Remote* SSH

Untuk membangun sebuah web server yang dapat mengakses jaringan internet dibutuhkan beberapa paket konfigurasi yang perlu diinstall pada raspberry. Paket–paket ini dibutuhkan untuk mengatur web server yang akan digunakakan oleh *cloud server*. Web server diperlukan untuk proses pengiriman data antar *cloud* dan server tempat penyimpanan berkas. Berikut konfigurasi untuk meng-*install* paket-paket tersebut :

```
sudo apt-get install nginx openssl ssl-cert php5-cli php5-
sqlite php5-gd php5-common php5-cgi sqlite3 php-pear php-apc
curl libapr1 libtool curl libcurl4-openssl-dev php-xml-parser
php5 php5-dev php5-curl php5-gd php5-fpm memcached php5-
memcache varnish
```

Konfigurasi pertama yang dilakukan yaitu konfigurasi pada pada SSL certificate seperti yang ditunjukan pada gambar 5. SSL certificate dibutuhkan untuk menjaga informasi sensitif selama dalam proses pengiriman melalui Internet dengan cara dienkripsi, sehingga hanya penerima pesan yang dapat memahami dari hasil enkripsi tersebut. Hal ini sangat penting, karena informasi yang kita kirimkan di Internet membutuhkan proses perjalanan dari komputer ke komputer sampai mencapai server tujuan.



Gambar 5. SSL Certificate

Setelah melakukan konfigurasi pada SSL certificate selanjutnya dilakukan chmod pada file cert yang telah buat dengan mengetikkan :

sudo chmod 600 /etc/nginx/cert.pem
sudo chmod 600 /etc/nginx/cert.key

Setelah mengganti hak akses pada file tcert, beberapa file config pada web server dihapus dengan cara :

sudo sh -c "echo '' > /etc/nginx/sites-available/default"

Untuk mengconfig file config web server dapat dilakukan dengan cara mengetikkan perintah konfigurasi sebagai berikut :

sudo nano /etc/nginx/sites-available/default

Ganti isi file config tersebut dengan :

```
upstream php-handler {
server 127.0.0.1:9000;
#server unix:/var/run/php5-fpm.sock;
}
server {
listen 80;
server_name 192.168.137.10;
return 301 https://$server_name$request_uri;
# enforce https
}
server {
listen 443 ssl;
server_name 192.168.137.10;
ssl_certificate /etc/nginx/cert.pem;
ssl_certificate_key /etc/nginx/cert.key;
# Path to the root of your installation
root /var/www/ownCloud;
client_max_body_size 1000M; # set max
upload size
fastcgi_buffers 64 4K;
rewrite ^/caldav(.*)$ /remote.php/caldav$1
redirect;
rewrite ^/carddav(.*)$ /remote.php/carddav$1
redirect;
rewrite ^/webdav(.*)$ /remote.php/webdav$1
redirect;
index index.php;
error_page 403 /core/templates/403.php;
error_page 404 /core/templates/404.php;
location = /robots.txt {
allow all;
log_not_found off;
access_log off;
}
location ~
^/(?:\.htaccess|data|config|db_structure\.xml|R
EADME) {
deny all;
}
location / {
# The following 2 rules are only needed with
webfinger
rewrite ^/.well-known/host-meta
/public.php?service=host-meta last;
```

e-ISSN: 2549-7952
p-ISSN: 2580-3336

```
rewrite ^/.well-known/host-meta.json
/public.php?service=host-meta-json last;
rewrite ^/.well-known/carddav
/remote.php/carddav/ redirect;
rewrite ^/.well-known/caldav
/remote.php/caldav/ redirect;
rewrite ^(/core/doc/[^\/]+/)$ $1/index.html;
try_files $uri $uri/ index.php;
}
location ~ \.php(?:$|/) {
fastcgi_split_path_info ^(.+\.php)(/.+)$;
include fastcgi_params;
fastcgi_param SCRIPT_FILENAME
$document_root$fastcgi_script_name;
fastcgi_param PATH_INFO
$fastcgi_path_info;
fastcgi_param HTTPS on;
fastcgi_pass php-handler;
}
# Optional: set long EXPIRES header on static
assets
location ~*
\.(?:jpg|jpeg|gif|bmp|ico|png|css|js|swf)$ {
expires 30d;
# Optional: Don't log access to assets
access_log off;
}
}
```

Berikut adalah penjelasan tentang isi file yang telah dicopy pada file konfigurasi :

```
upstream php-handler {
server 127.0.0.1:9000;
#server unix:/var/run/php5-fpm.sock;
}
server {
listen 80;
server_name 192.168.137.10;
return 301 https://$server_name$request_uri;
# enforce https
}
```

Pada bagian ini digunakan agar server dapat memasuki port 80 yang digunakan http melalui ip yang di miliki oleh server. Pada port 301 di gunakan untuk akses penggunaan transfer data TCP dan UDP.

```
server {
listen 443 ssl;
server_name 192.168.137.10;
ssl_certificate /etc/nginx/cert.pem;
ssl_certificate_key /etc/nginx/cert.key;
```

Bagian ini menggunakan port 443 yang akan di gunakan TCP/UDP untuk mendefinisikan penggunaan pemilihan transmisi data yang akan di guanakan, juga mendefinisikan penempatan SSL_certificate guna mengamankan proses transmisi.

```
# Path to the root of your installation
root /var/www/ownCloud;
client_max_body_size 1000M; # set max
upload size
fastcgi_buffers 64 4K;
rewrite ^/caldav(.*)$ /remote.php/caldav$1
redirect;
rewrite ^/carddav(.*)$ /remote.php/carddav$1
redirect;
rewrite ^/webdav(.*)$ /remote.php/webdav$1
redirect;
index index.php;
error_page 403 /core/templates/403.php;
error_page 404 /core/templates/404.php;
location = /robots.txt {
allow all;
log_not_found off;
access_log off;
}
location ~
^/(?:\.htaccess|data|config|db_structure\.xml|R
EADME) {
deny all;
}
location / {
# The following 2 rules are only needed with
webfinger
rewrite ^/.well-known/host-meta
/public.php?service=host-meta last;
rewrite ^/.well-known/host-meta.json
/public.php?service=host-meta-json last;
rewrite ^/.well-known/carddav
/remote.php/carddav/ redirect;
rewrite ^/.well-known/caldav
/remote.php/caldav/ redirect;
rewrite ^(/core/doc/[^\/]+/)$ $1/index.html;
try_files $uri $uri/ index.php;
}
location ~ \.php(?:$|/) {
fastcgi_split_path_info ^(.+\.php)(/.+)$;
include fastcgi_params;
fastcgi_param SCRIPT_FILENAME
$document_root$fastcgi_script_name;
fastcgi_param PATH_INFO
$fastcgi_path_info;
fastcgi_param HTTPS on;
fastcgi_pass php-handler;
}
```

Bagian ini menjelaskan penempatan file penyimpanan *cloud* semua file dimana o*wncloud* akan di install nantinya.

```
# Optional: set long EXPIRES header on static
assets
location ~*
\.(?:jpg|jpeg|gif|bmp|ico|png|css|js|swf)$ {
expires 30d;
# Optional: Don't log access to assets
access_log off;
```

e-ISSN: 2549-7952
p-ISSN: 2580-3336

}
}

Pengaturan yang bersifat opsional untuk menghapus beberapa jenis file yang disimpan di *cloud* setelah melewati batas waktu, yaitu 30 hari. Untuk konfigurasi file PHP adalah sebagai berikut :

sudo nano /etc/php5/fpm/php.ini

Ganti pada bagian upload_max_filesize menjadi 2000M

upload_max_filezise = 2000

Ganti pada bagian post_max _size menjadi 2000M

post_max_size = 2000M

Konfigurasi ini untuk mengatur batas maksimum *upload* pada *cloud*. Simpan dan keluar dari file konfigurasi. Selanjutnya lakukan perubahan pada file config berikut:

sudo nano /etc/php5/fpm/pool.d/www.conf

Lakukan perubahan pada bagian listen dengan menambahkan seperti berikut :

listen = 127.0.0.1:9000

Simpan dan keluar, lanjutkan dengan mengubah dhys-swapfile dengan mengetikkan :

sudo nano /etc/dphys-swapfile

Ubah pada bagian conf_swapsize menjadi :

CONF_SWAPSIZE = 512

Simpan dan keluar kemudian lakukan reboot pada raspberry.

Konfigurasi selanjutnya yaitu melakukan instalasi pada *ownCloud*. *OwnCloud* merupakan sebuah perangkat lunak yang memungkinkan seorang administrator untuk membangun sebuah sarana server yang digunakan sebagai tempat melakukan pertukaran berkas, langkah-langkah instalasinya adalah sebagai berikut :

sudo mkdir -p /var/www/ownCloud
sudo wget https://download.ownCloud.org/community/ownCloud-8.1.1.tar.bz2
sudo tar xvf ownCloud-8.1.1.tar.bz2
sudo mv ownCloud/ /var/www/
sudo chown -R www-data:www-data /var/www
rm -rf ownCloud ownCloud-8.1.1.tar.bz2

Bagian ini berisikan keterangan untuk pembuatan dari folder yang akan digunakan untuk installasi dari *ownCloud* dan juga *path* atau *source* yang akan digunakan untuk proses mendownload *ownCloud*. Hal ini diperlukan untuk mengubah file htaccess

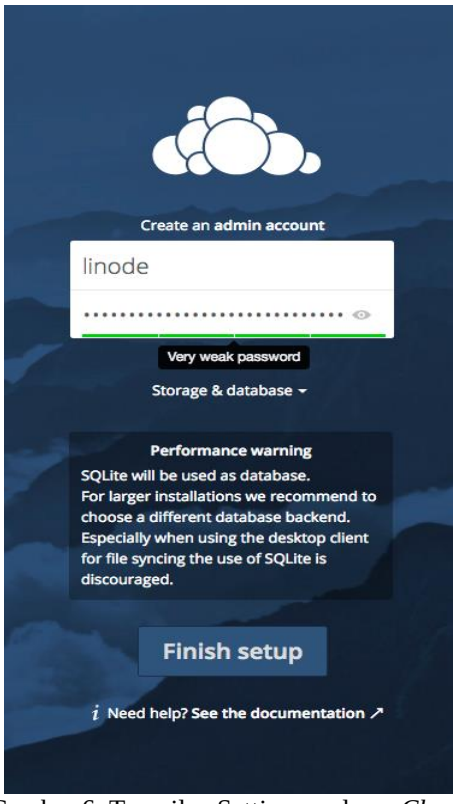
cd /var/www/ownCloud
sudo nano .htaccess
ganti nilai menjadi 2000M
php_value_upload_max_filesize 2000M
php_value_post_max_size 2000M
php_value_memory_limit 2000M

Setelah itu simpan dan keluar. Setelah itu lakukan konfigurasi pada file user.ini.

sudo nano .user.ini
ganti nilai pada file menjadi 2000M
upload_max_filesize=2000M
post_max_size=2000M
memory_limit=2000M

Pergantian angka menjadi 2000M ini guna membatasi batas upload yang ada pada *cloud*. Lakukan reboot pada raspberry dan kita sudah bisa melakukan pengaksesan pada *OwnCloud* menggunakan IP address dari raspberry.

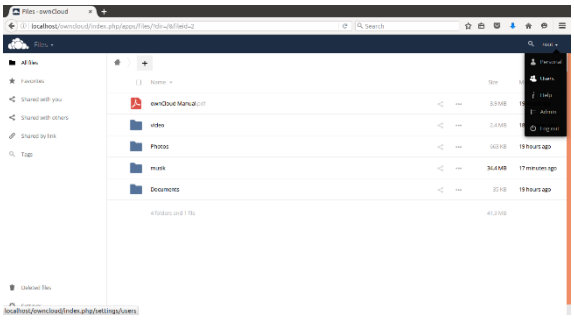
Untuk melakukan konfigurasi pada *ownCloud* dapat dilakukan dengan mengakses IP address dari raspberry pada *browser* yang ada pada komputer administrator. Hal yang pertama dilakukan adalah dengan membuat akun yang akan digunakan sebagai admin dari *cloud server* yang ditunjukan pada gambar 6.



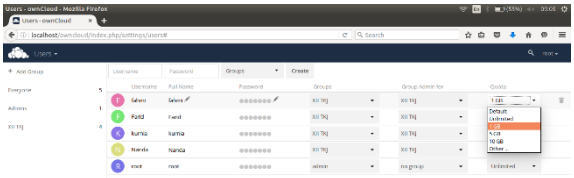
Gambar 6. Tampilan Setting awal *ownCloud*

e-ISSN: 2549-7952
p-ISSN: 2580-3336

Pada tahap ini akan didaftarkan akun pertama yang akan didaftarkan sebagai administrator dari *ownCloud.* Sebagai contoh dibuat akun bernama Administrator dengan password *admin1234.* Pada mode administrator seorang admin dapat membuat sebuah *user* yang akan digunakan oleh *client* untuk dapat mengakses *cloud server* yang ditunjukan pada gambar 7. Admin dapat menentukan jumlah kuota yang akan diberikan pada setiap *user* dan dapat membuat group untuk *user* untuk mempermudah dalam pengaksesan file yang ditunjukan pada gambar 8.



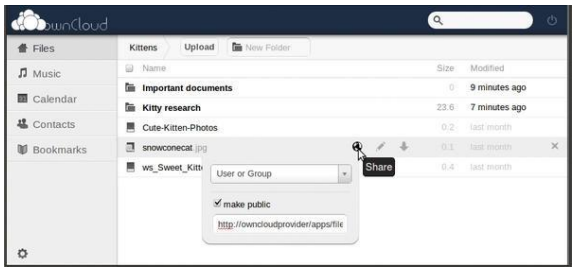
Gambar 7. Pembuatan *user*



Gambar 8. Kuota untuk masing-masing *user*

Percobaan pertama yang dilakukan yaitu meng-*upload* file yang dilakukan oleh administrator, file tersebut kemudian dapat di*download* oleh *client* yang sudah terhubung. Untuk setiap file yang di*upload* bisa dibagikan ke perorangan ataupun membagikan kedalam sebuah grup.File yang di*upload* tidak hanya antara administrator dan *user* saja tetapi juga bisa dilakukan antar *user*. Yang membedakan pada antara user dengan administrator adanya perbedaanmenu, ada beberapa menu yang hanya dapat diakses oleh administrator saja.

Percobaan kedua yang dilakukan yaitu membagikan file yang di*upload* oleh*user*yang satu kepada *user*yang lain. Ada beberapa file yang akan dicoba untuk melakukan proses*upload* dan *download* ini, diantaranya file dokumen , gambar dan sebuah file video.Untuk membagikan klik pada bagian share yang berada disamping file kemudianketikan kepada siapa akan membagikan file tersebut. Percobaan *upload* dan *sharing* file seperti yang ditunjukan pada gambar 9.



Gambar 9. Ujicoba *Upload*dan *Sharing*file

4. SIMPULAN

Berdasarkan hasil ujicoba yang dilakukan dapat disimpulkan bahwa sebuah *cloud server* berbasis raspberry dapat diaplikasikan untuk mengakses file. Proses *upload* dan *download* file dapat dilakukan baik antara administrator dengan *user* maupun antara *user* dengan *user*. Aplikasi *cloud server* menggunakan raspberry dapat membantu dalam mengakses data dan informasi dari mana saja dan kapan saja dan dapat memberikan alternatif media penyimpanan yang dapat digunakan untuk menyimpan data serta mengurangi resiko kehilangan data karena kerusakan media penyimpanan konvensional.

DAFTAR PUSTAKA

[1] Ahmad Ashari, Herri Setiawan, 2011, *Cloud Computing : Solusi ICT?, Jurnal Sistem Informasi (JSI) ISSN Print : 2085-1588 ISSN Online : 2355-4614, VOL. 3, NO. 2*

[2] Angga Prasetyo, 2015, *Perancangan Dan Analisa Cloud Storage Infrastructure As Service Dengan Kendali Raspberry*, Jurnal Ilmiah NERO Vol. 2, No.1

[3] Maimunah, Yohanes Ari Kuncoro Yakti, Neni Puspitasari, 2012, *Konsep Dan Penerapan Cloud Computing Untuk Meningkatkan Mutu Pembelajaran*, CSRID Journal, Vol.4 No.3, Hal. 220 – 230

[4] Sitti Aisa, Thabrani R, 2016, *Implementasi Private Cloud Menggunakan Raspberry PI Untuk Pengaksesan Data Pribadi*, Jurnal Penelitian Pos dan Informatika, Vol 6 No.2, Hal 137 – 152