RANCANG BANGUN KEAMANAN FTP SERVER MENGGUNAKAN ENKRIPSI AES 128 PADA TRANSPORT LAYER SECURITY MENGGUNAKAN SISTEM OPERASI LINUX UBUNTU

Disusun dan Diajukan Untuk Menempuhi Persyaratan Ujian Akhir Memperoleh
Gelar Sarjana Komputer pada Fakultas Sains Dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH

NAMA	**: WARDA MAYA IS PUTRI SINAGA**
NPM	**: 1514370540**
PROGRAM STUDI	**: SISTEM KOMPUTER**

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2020

ABSTRAK

WARDA MAYA IS PUTRI SINAGA

RANCANG BANGUN KEAMANAN FTP SERVER MENGGUNAKAN ENKRIPSI AES 128 PADA TRANSPORT LAYER SECURITY MENGGUNAKAN SISTEM OPERASI LINUX UBUNTU

File Transfer Protocol (FTP) masih menjadi media favorit yang digunakan untuk melakukan transfer file melalui jaringan internet atau lokal terutama file yang memiliki size besar. Hal ini disebabkan media komunikasi seperti email memiliki keterbatasan untuk mengirim ukuran file yang besar. FTP hanya menggunakan metode keamanan standar, yaitu menggunakan username dan password yang dikirim dalam bentuk tidak aman. Dalam bidang kriptografi terdapat dua konsep yang sangat penting atau utama yaitu enkripsi dan dekripsi. Enkripsi adalah proses data yang mau dikirim diubah menjadi bentuk yang tidak dikenali sebagai informasi awalnya dengan menggunakan metode tertentu. Dekripsi adalah mengubah kembali bentuk tersamar tersebut menjadi informasi awal. Sebuah pesan yang masih asli dan belum mengalami penyandian dikenal dengan istilah plaintext. Sebuah pesan atau data yang masih asli dan belum mengalami penyandian dikenal dengan istilah plaintext. AES merupakan salah satu algoritma simetri dan akan digunakan sebagai algoritma enkripsi dan dekripsi pada dokumen, sedangkan MD5 merupakan fungsi hash dan akan digunakan sebagai algoritma penyandian kunci, karena MD5 merupakan algoritma satu arah (Gumira, 2016). Metode algoritma Advanced Encryption Standard (AES). Hasil dari penelitian adalah Membangun dan mengkonfigurasi FTP server di Linux Ubuntu dengan menggunakan keamanan enkripsi AES-128 bit pada Transport Layer Security.

Kata kunci : File Transfer Protocol, Advanced Encryption Standard (AES), Enkripsi, Dekripsi,

DAFTAR ISI

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

BAB I

PENDAHULUAN

1.1 Latar Belakang

File Transfer Protocol (FTP) menjadi media yang digunakan untuk melakukan *transfer file* melalui jaringan *internet* atau local terutama *file* yang berukuran besar. Hal ini disebabkan media komunikasi seperti *email* memiliki keterbatasan untuk mengirim *file* yang besar. *FTP* hanya menggunakan metode autentikasi standar, yakni menggunakan *username* dan *password* yang dikirim dalam bentuk tidak aman. Pengguna yang sudah terdaftar dapat bisa *login* untuk mengakses, men-*download*, dan meng-*upload* berkas-berkas. Umumnya, para pengguna yang memiliki *Login* memiliki akses penuh terhadap beberapa direktori, sehingga mereka dapat membuat berkas, membuat direktori, dan bahkan menghapus berkas. Pengguna yang belum terdaftar dapat menggunakan metode *anonymous login*, yakni menggunakan *username anonymous* dan *password* yang diisi dengan alamat *e-mail* (Oklilas, 2014).

Dalam ilmu kriptografi terdapat dua konsep yang sangat penting atau utama yaitu enkripsi dan dekripsi. Enkripsi adalah proses data yang mau dikirim diubah menjadi bentuk yang tidak dikenali sebagai informasi awalnya dengan menggunakan metode tertentu. Dekripsi adalah mengubah kembali bentuk tersamar tersebut menjadi informasi awal. Sebuah pesan yang masih asli dan belum mengalami penyandian dikenal dengan istilah *plaintext*. Kemudian setelah disamarkan dengan suatu cara penyandian, maka *plaintext* ini disebut sebagai

chipertext. Proses penyamaran dari *plaintext* ke *ciphertext* disebut enkripsi (*encryption*), dan proses pengembalian dari *ciphertext* menjadi *plaintext* kembali disebut dekripsi (*decryption*) (Pabokory, 2018).

Advanced Encryption Standard (AES) merupakan salah satu algoritma simetri dan akan digunakan sebagai algoritma enkripsi dan dekripsi pada dokumen, sedangkan *MD5* merupakan fungsi *hash* dan akan digunakan sebagai algoritma penyandian kunci, karena *MD5* merupakan algoritma satu arah (Gumira, 2016). Metode algoritma *Advanced Encryption Standard (AES)*. Algoritma *Advanced Encryption Standard (AES)* dipilih penulis dalam menjaga keamanan pada sebuah data atau informasi tersebut, dikarenakan *AES* merupakan *cipher* yang berorientasi pada *bit*, sehingga memungkinkan untuk implementasi algoritma yang efisien ke dalam *software* dan *hardware*. *AES* memiliki ketahanan terhadap semua jenis serangan yang diketahui. *AES* terbukti kebal menghadapi serangan konvensional (linear dan diferensial *attack*) yang menggunakan statistik untuk memecahkan sandi, dan dalam setiap proses enkripsi dan dekripsi harus melakukan 10 perputaran dalam melakukan pengamanan maupun untuk membuka pengamanan tersebut (Pabokory, 2015).

Atas dasar pertimbangan itu, maka penulis tertarik membuat skripsi sistem pakar dengan judul : ”**Rancang Bangun Keamanan FTP Server Menggunakan Enkripsi AES-128 Pada Transport Layer Security Menggunakan Sistem Operasi Linux Ubuntu**”.

1.2. Rumusan Masalah

Berdasarkan identifikasi masalah diatas maka penulis menentukan suatu rumusan masalah yaitu:

1. Bagaimana merancang dan membangun *FTP server* dengan menggunakan enkripsi *AES-128* pada *Transport Layer Security (TLS)* yang aman dalam menghadapi serangan konvensional (linear dan diferensial *attack*) yang menggunakan statistic untuk memecahkan sandi, dan dalam setiap proses enkripsi dan dekripsi ?
2. Bagaimana membuktikan perbandingan fitur keamanan *Transport Layer Security (TLS)* menggunakan enkripsi *AES-128 bit* dalam mengamankan *Transfer Data* ?

1.3 Batasan Masalah

Agar penelitian yang dilakukan lebih terarah dan sistemastis, maka perlu dibuat batasan masalah yaitu :

1. Dalam penyusunan skripsi ini, masalah dibatasi pada membangun *FTP Server* pada sistem operasi *Linux Ubuntu.*
2. Membahas konfigurasi *FTP server* dengan keamanan enkripsi *AES-128* pada *TLS.*
3. *Software* yang digunakan pada sisi *server* adalah *TLS.*
4. *Software* yang digunakan pada sisi *client* adalah *CoreFTP Client.*
5. *Type file* yang digunakan adalah *file* berekstensi *.txt.*
6. Hanya dapat diimpelementasikan pada lingkungan jaringan *intranet* atau *Local Area Network (LAN)*

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah :

1. Membangun dan mengkonfigurasi *FTP server* di Linux *Ubuntu* dengan menggunakan keamanan enkripsi *AES-128* pada *TLS*.
2. Membuktikan perbandingan aplikasi *FTP server* dengan *TLS* dalam mengamankan *Transfer Data.*

1.5 Manfaat

Adapun maanfaat dari penelitian ini adalah :

1. Membantu pengguna komputer yang ingin melakukan pengiriman data yang aman baik itu sebagai *Server* atau sebagai *Client.*
2. Memberikan kenyamanan saat melakukan *download* dan *upload* berkas-berkas komputer antara *FTP client* dan *FTP server*.
3. Memberikan pilihan keamanan aplikasi *FTP server.*
4. Lebih hemat biaya karena menggunakan sistem operasi dan aplikasi yang *open source.*
5. Sebagai bahan referensi bagi peneliti lain yang ingin mengembangkan *FTP Server* berbasis *Linux* yang aman berbasis *TCP/IP*.

BAB II
LANDASAN TEORI

2.1. Jaringan Komputer

Jaringan komputer adalah himpunan koneksi antara 2 komputer atau lebih yang terhubung dengan media transmisi kabel atau tanpa kabel (*wireless*). Jaringan komputer melakukan pemakaian bersama atas sumber daya yang ada pada jaringan komputer tersebut. Pemakaian bersama dalam sumber daya termasuk di dalamnya adalah *file, printer, scanner* dan perangkat lainnya yang terhubung dalam jaringan. Pemakai *(user)* dapat mencetak pada satu atau lebih *printer* yang sama, mengakses suatu file yang terdapat pada komputer lain dan memakai *software* atau *hardware* yang dapat dipakai bersama-sama (Syafrizal, 2015).

Dalam sebuah jaringan komputer biasanya terdiri dari dua atau lebih komputer yang saling berhubungan satu sama lain dan saling berbagi sumber daya, misalnya *CD ROM, printer, scanner,* pertukaran *file* bahkan berkomunikasi secara elektronik. Komputer yang terhubung, dimungkinkan berhubungan dengan media kabel, saluran telepon, gelombang radio, satelit, sinar *infra merah*, atau tanpa kabel *(Syafrizal, 2015)*.

Jaringan Komputer memiliki beberapa manfaat dan keuntungan, antara lain :

1. Berbagi sumber: seluruh peralatan dan data yang dapat digunakan oleh setiap orang yang ada dijaringan tanpa dipengaruhi lokasi sumber dan pemakai. Misalnya: Staff perpustakaan mengirimkan daftar buku baru ke

perpustakaan dalam bentuk *printout* dengan langsung mencetaknya di*printer* perpustakaan.

2. Kehandalan tinggi: tersedianya sumber-sumber alternatif kapanpun diperlukan. Misalnya pada aplikasi perbankan atau militer, jika salah satu mesin tidak bekerja, kinerja organisasi tidak terganggu karena mesin lain mempunyai sumber yang sama.
3. Menghemat biaya, membangun jaringan dengan komputer-komputer kecil lebih murah dibandingkan dengan menggunakan *mainframe*. Data disimpan di sebuah komputer yang berfungsi sebagai *server* dan komputer lain yang menggunakan data tersebut berfungsi sebagai *client*.
4. Skalabilitas: meningkatkan kinerja dengan menambahkan komputer *server* atau *client* dengan mudah tanpa mengganggu kinerja komputer *server* atau komputer *client* yang sudah ada lebih dulu.
5. Media komunikasi: memungkinkan kerjasama antar orang-orang yang saling berjauhan melalui jaringan komputer baik untuk bertukar data maupun berkomunikasi.

2.2 Tipe - Tipe Jaringan Komputer

Secara umum jaringan komputer terbagi atas 3 jenis Berdasarkan area dan lokasi sebagai berikut :

2.2.1. *Local Area Network (LAN)*

LAN adalah singkatan dari *local area network*. Jenis jaringan LAN sangat sering dijumpai di kantor, warnet, sekolah, kampus yang membutuhkan hubungan atau koneksi antara dua komputer atau lebih dalam suatu ruangan. Jaringan LAN

juga merupakan jaringan yang sangat di pengaruhi oleh topologi jaringannya (Wongkar, 2015).

2.2.2. *Metropolitan Area Network (MAN)*

MAN singkatan dari *metropolitan area network*. Jaringan komputer *MAN* ini adalah suatu jaringan komputer dalam suatu wilayah dengan kecepatan *transfer* data sangat tinggi yang menghubungkan suatu lokasi seperti perkantoran, sekolah, kampus dan pemerintahan. Jaringan *MAN* ini adalah gabungan dari beberapa jaringan *LAN*. Jangkauan dari jaringan *MAN* ini bisa mencapai 20 hingga 100 kilometer (Wongkar, 2015).

2.2.3. *Wide Area Network (WAN)*

WAN singkatan dari wide area network. WAN adalah jenis jaringan komputer yang mencakup area yang sangat besar. contohnya adalah jaringan yang menghubugkan suatu wilayah atau suatu negara dengan negara lainnya (Wongkar, 2015).

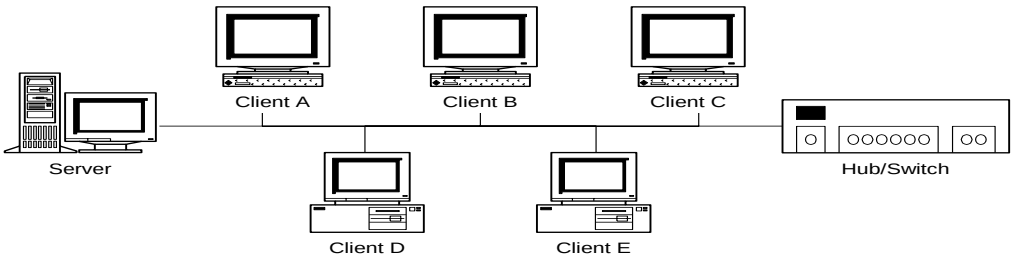
2.3 Topologi Jaringan

Topologi jaringan adalah gambaran perencanaan hubungan antar koneksi komputer dalam *Local Area Network* yang umumnya menggunakan media kabel (sebagai media transmisi), dengan konektor *RJ45*, *ethernet card*, dan perangkat pendukung lainnya (Syafrizal, 2015).

2.3.1. Topologi Bus

Pada *Topologi Bus* digunakan sebuah kabel tunggal atau kabel pusat di mana seluruh *workstation* dan *Server* dihubungkan. Pada topologi ini semua sentral dihubungkan secara langsung pada medium transmisi dengan konfigurasi

yang disebut *Bus*. Transmisi sinyal dari suatu sentral tidak dialirkan secara bersamaan dalam dua arah. *Topologi* jaringan *bus* tidak umum digunakan untuk interkoneksi antar sentral, tetapi biasanya digunakan pada sistem jaringan komputer. Topologi *bus* dapat digambarkan pada gambar 2.1.

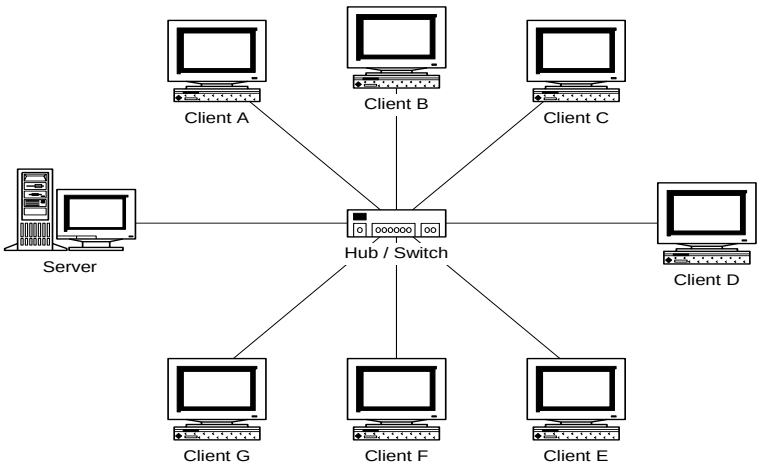


Gambar 2.1 Topologi Bus.

Sumber : Syafrizal, 2015.

2.3.2. Topologi Bintang (Star)

Pada *Topologi Sta*r, masing-masing *workstation* dihubungkan secara langsung ke *server* atau *hub*. Dalam *topologi* jaringan bintang, salah satu sentral dibuat sebagai sentral pusat. Bila dibandingkan dengan sistem *mesh*, sistem ini mempunyai tingkat kerumitan jaringan yang lebih sederhana sehingga sistem menjadi lebih ekonomis. Topologi bintang dapat digambarkan pada gambar 2.2.



Gambar 2.2 Topologi Star

Sumber : Syafrizal, 2015.

2.4. *IP Address*

IP Address merupakan pengenal yang digunakan untuk memberi alamat pada tiap-tiap komputer dalam jaringan. Format *IP Address* adalah bilangan 32 *bit* yang tiap 8 bitnya dipisahkan oleh tanda titik, dan secara teoritis dapat mengalamati hingga 4 miliar *host* komputer atau lebih tepatnya 4.294.967.296 host di seluruh dunia, jumlah *host* tersebut didapatkan dari 256 (didapatkan dari 8 bit) dipangkat 4 (karena terdapat 4 oktet) sehingga nilai maksimal dari alamat *IP* versi 4 tersebut adalah 255.255.255.255 dimana nilai dihitung dari nol sehingga nilai nilai *host* yang dapat ditampung adalah 256x256x256x256=4.294.967.296 *host* (Syafrizal, 2015).

2.4.1. Kelas - Kelas *IP Address*

Alamat *IP* versi 4 dibagi ke dalam beberapa kelas, dilihat dari oktet pertamanya, seperti terlihat pada tabel. Sebenarnya yang menjadi pembeda kelas *IP* versi 4 adalah pola biner yang terdapat dalam oktet pertama (utamanya adalah *bit-bit* awal/*high-order bit*), tapi untuk lebih mudah mengingatnya, akan lebih cepat diingat dengan menggunakan representasi desimal.

Tabel 2.1 Pembagian Kelas Pada *IP Address* Versi 4.

Kelas Alamat IP	Oktet pertama (desimal)	Oktet pertama (biner)	Digunakan oleh
Kelas A	1–126	0xxx xxxx	Alamat *unicast* untuk jaringan skala besar
Kelas B	128–191	10xx xxxx	Alamat *unicast* untuk jaringan skala menengah hingga skala besar
Kelas C	192–223	110x xxxx	Alamat *unicast* untuk jaringan skala kecil
Kelas D	224–239	1110 xxxx	Alamat *multicast* (bukan alamat *unicast*)
Kelas E	240–255	1111 xxxx	Direservasikan;umumnya digunakan sebagai alamat percobaan (eksperimen); (bukan alamat unicast)

1. *IP Address* Kelas A

IP Address kelas A diimplementasikan untuk jaringan skala besar. Nomor urut *bit* tertinggi di dalam alamat *IP* kelas A selalu diset dengan nilai 0 (nol). Tujuh bit berikutnya untuk melengkapi oktet pertama akan membuat sebuah *network identifier*. 24 *bit* sisanya (atau tiga oktet terakhir) merepresentasikan *host identifier*. Ini mengizinkan kelas A memiliki hingga 126 jaringan, dan 16,777,214 *host* tiap jaringannya.

2. *IP Address* Kelas B

IP Address kelas B diimplementasikan untuk jaringan skala menengah hingga skala besar. Dua *bit* pertama di dalam oktet pertama alamat *IP* kelas B selalu diset ke bilangan biner 10.14 *bit* berikutnya (untuk melengkapi dua *oktet* pertama), akan membuat sebuah *network identifier*. 16 *bit* sisanya (dua oktet terakhir) merepresentasikan *host identifier*. Kelas B dapat memiliki 16,384 *network*, dan 65,534 *host* untuk setiap *network*-nya.

3. *IP Address* Kelas C

IP Address kelas C diimplementasikan untuk jaringan berskala kecil. Tiga bit pertama di dalam oktet pertama alamat kelas C selalu diset ke nilai biner 110. 21 bit selanjutnya (untuk melengkapi tiga oktet pertama) akan membentuk sebuah *network identifier*. 8 bit sisanya (sebagai oktet terakhir) akan merepresentasikan *host identifier*. Ini memungkinkan pembuatan total 2,097,152 buah *network*, dan 254 *host* untuk setiap network-nya.

4. *IP Address* Kelas D

IP Address kelas D diimplementasikan hanya untuk alamat-alamat *IP multicast*, sehingga berbeda dengan tiga kelas di atas. Empat *bit* pertama di dalam *IP* kelas D selalu diset ke bilangan biner 1110. 28 *bit* sisanya digunakan sebagai alamat yang dapat digunakan untuk mengenali host. Untuk lebih jelas mengenal alamat ini, lihat pada bagian Alamat *Multicast* IPv4.

5. *IP Address* Kelas E

Alamat *IP* kelas E diimplementasikan sebagai alamat yang bersifat "eksperimental" atau percobaan dan dicadangkan untuk digunakan pada masa depan. Empat *bit* pertama selalu diset kepada bilangan biner 1111. 28 bit sisanya digunakan sebagai alamat yang dapat digunakan untuk mengenali *host.*

2.5. *Client-Server*

Client-Server merupakan model komunikasi antara dua atau lebih komputer guna melakukan pembagian tugas. Pembangunan aplikasi yang memanfaatkan konsep komputasi tersebar telah digantikan oleh teknologi *Web Service* namun pada area yang masih mengutamakan kecepatan adalah hal yang utama. Teknologi *Java RMI* tidak hanya dapat dibangun dalam satu komputer melainkan ke banyak komputer. Dari hasil implementasi teknologi *Java RMI* terdapat keuntungan pengaksesan data yang cepat karena adanya pembagian fungsi antara *RMI server* dan *RMI client*. *Remote Method Invocation (RMI)* dapat didefinisikan sebagai sebuah fasilitas standar *Java* yang berguna melakukan pemanggilan (*invocation*) suatu *methode* dari jarak jauh (*remote*) didalam jaringan (Anthony, Tanaamah, & Wijaya, 2017).

2.6. *FTP*

File Transfer Protocol (FTP) adalah *client* dan *server* protokol yang menyediakan fasilitas untuk *transfer* data dalam jaringan protokol yang digunakan untuk pertukaran *file* antara dua *host* dalam jaringan *TCP/IP*. Sebuah *FTP server* dapat di-set sebagai *FTP* publik sehingga setiap orang dapat mengakses data-data yang ada di server *FTP* dengan menggunakan *login anonymous* atau *FTP*. Selain itu, *FTP* juga dapat diatur agar *server* hanya dapat diakses oleh user tertentu saja dan tidak untuk *public* (Arman, 2017).

2.7. Kriptografi

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Dalam ilmu kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut sebagai *plaintext* (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah *plaintext* melibatkan penggunaan suatu bentuk kunci. Pesan plaintext yang telah dienkripsi (atau dikodekan) dikenal sebagai *ciphertext* (teks sandi). Di dalam kriptografi kita akan sering menemukan berbagai istilah atau terminology (Pabokory, 2015 : 22).

Beberapa istilah yang harus diketahui yaitu :

1. Pesan, Plainteks, dan Cipherteks

   Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (*plaintext*) atau teksjelas (*cleartext*).

2. Pengirim dan penerima komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan.

3. Enkripsi dan dekripsi Proses menyandikan plainteks menjadi *cipherteks* disebut enkripsi (*encryption*) atau *enciphering* (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan *cipherteks* menjadi *plainteks* semula disebut dekripsi (*decryption*) atau *deciphering* (standard nama menurutISO 7498-2).

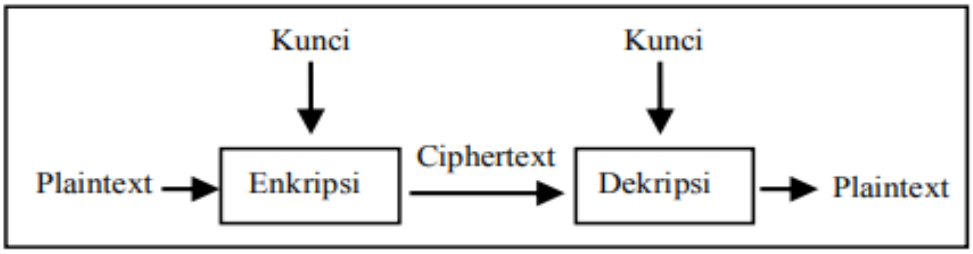
4. Cipher dan kunci Algoritma *kriptografi* disebut juga *cipher*, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa *cipher* memerlukan algoritma yangberbeda untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma *kriptografi* adalah relasi antara dua buah himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi *cipherteks*. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara dua himpunan tersebut.

Misalkan P menyatakan plainteks dan C menyatakan cipherteks,maka :

E(P) = C → fungsi enkripsi E memetakan P ke C

D(C) =P →fungsi dekripsi Dmemetakan C ke P

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesanasal, maka persamaan D(E(P)) = P harusbenar.Kriptografi mengatasimasalah keamanan data dengan menggunakankunci, yang dalam hal inialgoritma tidak dirahasiakan lagi, tetapi kunci harus tetap dijagakerahasiaannya. Kunci (*key*) adalah parameter yang digunakan untuk transformasi enkripsi dan dekripsi. Kunci biasanya berupa string atau deretan bilangan. Dengan menggunakan kunci K, maka fungsi enkripsi dan dekripsi dapat ditulis sebagai skema diperlihatkan pada Gambar 2.1.



Gambar 2.1.Skema enkripsi dan dekripsi dengan menggunakan kunci.
Sumber : (Pabokory, 2015 : 22)

2.8. Sejarah Kriptografi

Sejarah kriptografi sebagian besar merupakan sejarah kriptografi klasik, yaitu metode enkripsi yang menggunakan kertas dan pensil atau mungkin dengan bantuan alat mekanik sederhana. Secara umum algoritma kriptografi klasik dikelompokkan menjadi dua kategori, yaitu algoritma transposisi (*transposition*

cipher) dan algoritma substitusi (*substitution cipher*). *Cipher* transposisi mengubah susunan huruf-huruf di dalam pesan, sedangkan *cipher* substitusi mengganti setiap huruf atau kelompok huruf dengan sebuah huruf atau kelompok huruf lain **(Pabokory, 2015)**.

2.9. Tujuan Kriptografi

Dari paparan awal dapat dirangkumkan bahwa kriptografi bertujuan untuk member layanan keamanan. Yang dinamakan aspek-aspek keamanan:

1. Kerahasiaan (*confidentiality*)

   Adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak.
2. Integritas data (*data integrity*) adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.
3. Otentikasi (*authentication*) adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak - pihak yang berkomunikasi(*user autehentication*).
4. *Non-repudiation*adalah layanan untuk menjaga entitas yang berkomunikasi melakukan penyangkalan **(Pabokory, 2015: 22)**.

2.10. Jenis-jenis Algoritma Kriptografi

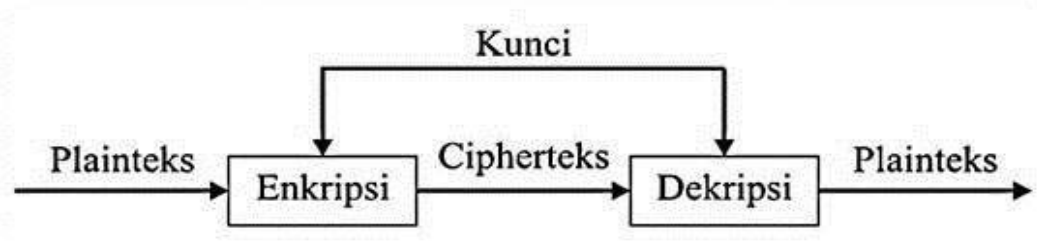
Algoritma kriptografi dibagi dua, yaitu algoritma simetri (menggunakan satu kunci), algoritma asimetri (menggunakan dua kunci berbeda untuk proses enkripsi dan dekripsi).

1. Algoritma Simetris

Dimana kunci yang digunakan pada proses enkripsi dan dekripsi adalah kunci yang sama. Dalam kriptografi kunci simetris dapat diasumsikan bahwa si penerima dan pengirim pesan telah terlebih dahulu berbagi kunci sebelum pesan dikirimkan.Keamanan dari sistem ini terletak pada kerahasiaan kuncinya.

Pada umumnya yang termasuk ke dalam kriptografi simetris ini beroperasi dalam mode blok (*block cipher*), yaitu setiap kali proses enkripsi atau dekripsi dilakukan terhadap satu blok data (yang berukuran tertentu), atau beroperasi dalam mode aliran (*stream cipher*), yaitu setiap kali enkripsi atau dekripsi dilakukan terhadap satu bit atau satu *byte* data.

Contoh algoritma simetris, yaitu : Trithemius, *Double Transposition Cipher*,DES (*Data Encryption Standard),* AES (*Advanced Encryption Standard)* (Kamil, 2016). Proses dari skema kriptografi simetris dapat dilihat pada gambar 2.2.



Gambar 2.2. Skema Algoritma Simetris
Sumber :(Kamil, 2016)

Kelebihan kriptografi simetris adalah **(Kamil, 2016)** :

a. Proses enkripsi atau dekripsi kriptografi simetris membutuhkan waktu yang singkat.

b. Ukuran kunci simetris *relative* lebih pendek.

c. Otentikasi pengiriman pesan langsung dari *ciphertext* yang diterima, karena kunci hanya diketahui oleh penerima dan pengirim saja.

Kelemahan kriptografi simetris adalah **(Kamil, 2016)** :

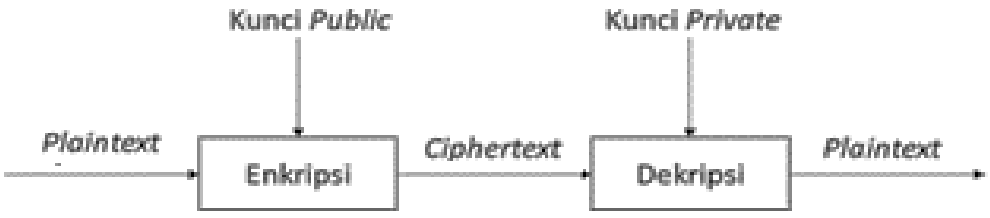
a. Kunci simetris harus dikirim melalui saluran komunikasi yang aman, dan kedua entitas yang berkomunikasi harus menjaga kerahasiaan kunci.

b. Kunci harus sering diubah, setiap kali melaksanakan komunikasi. Apabila kunci tersebut hilang atau lupa, maka pesan tersebut tidak dapat dibuka.

2. Algoritma Asimetris

Berbeda dengan kriptografi kunci simetris, kriptografi kunci public memiliki dua buah kunci yang berbeda pada proses enkripsi dan dekripsinya. Dimana kunci yang digunakan untuk proses enkripsi atau sering disebut *public key* dan dekripsi atau sering disebut *private key* menggunakan kunci yang berbeda. Entitas pengirim akan mengenkripsi dengan menggunakan kunci *public*, sedangkan entitas penerima mendekripsi menggunakan kunci *private* **(Kamil, 2016)**.

Contoh algoritma asimetris, yaitu RSA (*Riverst Shamir Adleman)*, Knapsack, Rabin, ElGamal **(Munir, 2014)**.Skema dari kriptografi dapat dilihat pada gambar 2.3.



Gambar 2.3. Skema Algoritma Asimetris
Sumber : (Kamil, 2016)

Kelebihan kriptografi asimetris adalah **(Kamil,2016)** :

a. Hanya kunci *private* yang perlu dijaga kerahasiaannya oleh setiap entitas yang berkomunikasi. Tidak ada kebutuhan mengirim kunci *private* sebagaimana kunci simetri.

b. Pasangan kunci *private* dan kunci *public* tidak perlu diubah dalam jangka waktu yang sangat lama.

c. Dapat digunakan dalam pengaman pengiriman kunci simetris.

Kelemahan kriptografi asimetris adalah **(Kamil, 2016)** :

a. Proses enkripsi dan dekripsi umumnya lebih lambat dari algortima simetri, karena menggunakan bilangan yang besar dan operasi bilangan yang besar.

b. Ukuran *ciphertext* lebih besar dari *plaintext*.

c. Ukuran kunci relatif lebih besar daripada ukuran kunci simetris.

2.11. *Enkripsi*

Enkripsi adalah proses penyandian *plainteks* menjadi *cipherteks*, atau pengubahan data menjadi bentuk acak. Proses *enkripsi algoritma AES* terdiri dari 4 jenis *transformasi bytes*, yaitu *ShiftRows*, *SubBytes*, *Mixcolumns*, dan *AddRoundKey*. Pada awal proses *enkripsi*, input yang telah diinput ke dalam *state* akan mengalami *transformasi byte AddRoundKey*. Setelah itu, *state* akan mengalami proses perubahan *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang sebanyak Nr. Proses ini dalam *algoritma AES* disebut sebagai fungsi berulang. *Round* yang terakhir agak berbeda dengan *round-*

round sebelumnya dimana pada round terakhir, state tidak mengalami transformasi *MixColumns*. (Angga, dkk, 2018).

Enkripsi digunakan untuk menyandikan data-data atau informasi sehingga tidak dapat dibaca oleh orang lain. Dengan enkripsi, data kita disandikan (*Encrypted*) dengan menggunakan sebuah kunci (*key*). Untuk membuka (mendecrypt) data tersebut, digunakan kunci yang sama ketika mengenkrip. Enkripsi adalah proses menyembunyikan informasi dengan membuat informasi tersebut tidak dapat dibaca tanpa bantuan pengetahuan khusus. Dikarenakan enkripsi telah digunakan untuk mengamankan komunikasi di kebanyakan negara, hanya organisasi-organisasi tertentu dan individu yang memiliki kepentingan yang sangat mendesak akan kerahasiaan yang menggunakan enkripsi.

Keamanan dari enkripsi tergantung beberapa faktor salah satunya yaitu menjaga kerahasiaan kuncinya bukan algoritmanya. Proses enkripsi dapat diterangkan sebagai berikut:

Keterangan :

1. Input file dan key
2. Baca isi file
3. Lakukan perhitungan untuk melakukan enkripsi
4. Outputnya adalah cipherteks
5. Pilih Folder Penyimpanan
6. Selesai

Langkah-langkah pada proses enkripsi adalah sebagai berikut:

a. *Plaintext* diubah ke dalam bentuk bilangan. Untuk mengubah plaintext yang berupa huruf menjadi bilangan dapat digunakan kode *ASCII* dalam sistem bilangan desimal.
b. *Plaintext* m dinyatakan menjadi blok-blok m1,m2,m3,...., sedemikian sehingga setiap blok merepresentasikan nilai di dalam selang [0, n-1] , sehingga transformasinya menjadi satu ke satu.
c. Setiap blok mi dienkripsi menjadi blok ci dengan rumus mi = ci e mod n

2.12. *Deskripsi*

Dekripsi digunakan untuk mengembalikan data atau informasi yang sudah dienkripsi kebentuk awal sehingga dapat dibaca kembali, satu kaidah upaya pengolahan data menjadi sesuatu yang dapat diutarakan secara jelas dan tepat dengan tujuan agar dapat dipahami oleh orang yang tidak langsung mengalaminya sendiri dalam keilmuan, deskripsi diperlukan agar peneliti tidak melupakan pengalamannya dan agar pengalaman tersebut dapat dibandingkan dengan pengalaman peneliti lain, sehingga mudah untuk dilakukan pemeriksaan dan kontrol terhadap deskripsi tersebut. Pada umumnya deskripsi menegaskan sesuatu, seperti apa sesuatu itu kelihatannya, bunyinya, rasanya.

Deskripsi yang lengkap diciptakan dan dipakai dalam disiplin ilmu sebagai istilah teknik. Saat data yang dikumpulkan, deskripsi, analisis dan kesimpulannya lebih disajikan dalam angka maka hal ini dinamakan penelitian kuantitatif. Sebaliknya, apabila data deskripsi, dan analisis kesimpulannya disajikan dalam uraian kata maka dinamakan penelitian kualitatif. Proses deskripsi dapat diterangkan sebagai berikut:

Keterangan :

1. Pilih folder penyimpanan
2. Input file cipher & key
3. Baca isi file
4. Lakukan perhitungan untuk dekripsi
5. Outputnya adalah plaintext

Dekripsi adalah proses memperoleh kembali *plaintext* menjadi *ciphertext*, atau proses pengubahan kembali data yang berbentuk rahasia menjadi semula. *Transformasi byte* yang digunakan pada invers *cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*.

Langkah-langkah pada proses *dekripsi* adalah sebagai berikut :

a. Setiap blok *ciphertext* ci *didekripsi* kembali menjadi blok mi dengan rumus mi = ci d mod n
b. Kemudian blok-blok m1,m2,m3,...., diubah kembali ke bentuk huruf dengan melihat kode *ASCII* hasil *dekripsi*. (Yuza, dkk, 2018).

2.13. *Enkripsi Advanced Encryption Standard (AES)*

Algoritma kriptografi Rijndael yang dirancang oleh Vincent Rijmen dan John Daemen dipilih sebagai pemenang lomba algoritma kriptografi yang diadakan oleh *National Institutes of Standards and Technology* milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah kemudian dikenal dengan nama *Advanced Encryption Standard (AES).* Setelah mengalami beberapa proses standarisasi, Rijndael kemudian mengadopsi menjadi standard

algoritma kriptografi secara resmi pada 22 Mei 2002. Pada tahun 2006, AES merupakan salah satu algoritma terkenal yang digunakan dalam kriptografi kunci simetrik. Pada metode AES, jumlah blok *input*, blok *output*, dan *state* berjumlah 128 bit. Dengan besar data 128 bit, berarti Nb = 4 yang menunjukkan panjang data 22 tiap baris adalah 4 *byte*. Dengan blok *input* atau blok data berjumlah 128 *bit*, *key* yang digunakan pada algoritma *AES* tidak harus mempunyai besar yang sama dengan blok *input*. *Cipher* key pada algoritma *AES* bisa menggunakan kunci berjumlah 128 *bit*. Perbedaan panjang kunci akan mempengaruhi jumlah rotasi yang akan di implementasikan pada algoritma AES ini (K & Erlanshari, 2016).

Algoritma *Advanced Encryption Standard (AES)* dipilih penulis dalam menjaga keamanan pada sebuah data atau informasi tersebut, dikarenakan AES merupakan *cipher* yang berorientasi pada bit, sehingga memungkinkan untuk implementasi algoritma yang efisien ke dalam *software* dan *hardware*. *AES* memiliki ketahanan terhadap semua jenis serangan yang diketahui. Disamping itu kesederhanaan rancangan, kekompakan kode yang sederhana dan kecepatan pada berbagai *platform* dimiliki oleh algoritma *AES*. *AES* terbukti kebal menghadapi serangan konvensional (*linear* dan diferensial *attack*) yang menggunakan statistik untuk memecahkan sandi, dan dalam setiap proses enkripsi dan dekripsi harus melakukan 10 perputaran atau 10 iterasi (10 *Round*) dalam melakukan pengamanan maupun untuk membuka pengamanan tersebut (Pabokory, 2015).

2.13.1 Proses Enkripsi AES

Pada awal proses enkripsi, input yang telah diinput ke dalam *state* akan mengalami transformasi *byte AddRoundKey*. Setelah itu, *state* akan mengalami

proses transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang-ulang sebanyak Nr. Proses ini dalam algoritma AES disebut sebagai fungsi berulang. Perulangan yang terakhir agak berbeda dengan perulangan sebelumnya, dimana pada *perulangan* terakhir *state* tidak mengalami transformasi *MixColumns* (K & Erlanshari, 2016).

Proses Enkripsi metode AES menggunakan substitusi dan permutasi, dan sejumlah putaran (*cipher* berulang), dimana setiap putaran menggunakan kunci yang berbeda (kunci setiap putaran disebut kunci perulangan). Garis besar Algoritma AES yang beroperasi pada blok 128 bit dengan kunci 128 bit adalah sebagai berikut (di luar proses pembangkitan round key):

1. *AddRoundKey*: proses *XOR* antara *state* awal (plainteks) dengan *cipherkey*. Tahap ini juga disebut *initial round*. Putaran sebanyak Nr - 1 kali. Proses yang dilakukan pada setiap putaran.
2. *SubBytes*: proses Substitusi *byte* dengan menggunakan tabel substitusi (S-Box). Untuk setiap *byte* pada *array state*, misalkan S[r,c] = xy yang dalam hal ini xy adalah digit heksadesimal dari nilai S[r,c], maka nilai substitusinya, yang dinyatakan dengan S'[r,c], adalah elemen di dalam S-Box yang merupakan perpotongan baris x dengan kolom y.
3. *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping* pada 3 baris terakhir dari *array state*, dimana pada proses ini bit paling kiri akan dipindahkan menjadi *bit* paling kanan (rotasi bit). Jumlah pergeseran bergantung pada nilai baris (r). Baris r=1 digeser sejauh 1

byte, baris r=2 digeser sejauh 2 *byte*,dan baris r=3 digeser sejauh 3 *byte*. Baris r=0 tidak digeser.

4. *MixColumns*: mengacak data di masing-masing kolom *array state* Dalam proses *MixColumn* terdapat beberapa perkalian, yaitu *Matrix Multiplication* dan *Galois Field Multiplication.*
5. *AddRoundKey*: proses XOR antara state sekarang dengan *round key*.
6. *Final Round* (proses untuk putaran terakhir): *SubBytes, ShiftRows*, dan *AddRoundKey*.

2.14. *Transport Layer Security*

Dalam kegiatan transaksi *internet*, banyak dijumpai seperti *login*, belanja *online, transfer bank,* dan lain sebagainya yang memerlukan masukan data bersifat rahasia. *Transfer* data dilakukan melalui jaringan kabel atau *wifi*. Maka diperlukan mekanisme keamanan yang kuat untuk diimplementasikan pada *transport layer dari TCP/IP protokol* yang dikenal sebagai *Transport Layer Security (TLS)* atau *Secure Socket Layer (SSL*). Pengguna *internet* akan merasa aman setiap ada *HTTPS* bukan *HTTP* di kolom alamat *web browser*. Tetapi ditemukan juga serangan pada *TLS* (Fadhli, Munshi, & Wicaksono, 2016).

Secure Socket Layer (SSL), yang kini dikenal sebagai *Transport Layer Security (TLS)*, pertama kali dikembangkan oleh *Netscape*. *SSL* Versi 1.0 tidak pernah dipublikasikan, sedangkan *SSL* Versi 2.0 dirilis resmi pada tahun 1995. *TLS* dikenalkan pada tahun 1999 dan diperbaharui melalui *RFC* 5246 pada Agustus 2008 dan *RFC* 6176 pada Maret 2011 (Fadhli, Munshi, & Wicaksono, 2016).

TLS adalah kumpulan dari 3 *kriptografi*, yaitu:

1. *Authentication*
2. *Confidentiality*
3. *Integrity*

Protokol ini terdiri dari berbagai macam *cipher* untuk komunikasi yang aman. *Authentication* diperoleh dengan menggunakan *asimetric cipher* seperti *RSA, Diffie-Helman*, dan lain-lain. *Confidentiality* diperoleh dengan melakukan *enkripsi simetric* dari *plaintext* melalui transfer jaringan. Secara umum *simetric cipher* yang kuat diimplementasikan di *TLS* seperti AES, DES-3, RC4, dan sebagainya. *Intergity* diperoleh dengan menghitung *Message Authentication Code*. (Fadhli, Munshi, & Wicaksono, 2016).

Protokol *TLS* mendukung aplikasi untuk dapat berkomunikasi melalui jaringan dan menghindarkannya dari *eyesdropping*, perusakan, dan pemalsuan pesan. *TLS* menyediakan autentikasi antara dua sisi (*client* dan *server*) dan komunikasi privat melalui *internet* menggunakan kriptografi. Biasanya, autentikasi hanya dikenakan pada *server* (misal penjaminan identitas), sedangkan *client* tidak diautentikasi. Hal ini berarti pengguna (baik individu maupun aplikasi seperti *web browser*) dapat menjamin dengan siapa mereka berkomunikasi (penjaminan satu sisi). Level keamanan berikutnya adalah kedua sisi/pihak yang melakukan komunikasi dapat menjamin dengan siapa mereka berkomunikasi. Kondisi ini disebut dengan mutual authentication. Mutual authentication memerlukan infrastruktur kunci publik (PKI) untuk client, jika TLS-PSK atau

TLSSRP (yang dapat menjamin mutual authentication tanpa PKI) tidak digunakan. TLS meliputi tiga fase dasar yaitu:

1. Peer negotiation untuk pendukung algoritma
2. Pertukaran kunci dan autentikasi
3. Enkripsi sandi simetris dan autentikasi pesan

Selama fase pertama, *client* dan *server* bernegosiasi mengenai *cipher suites*, yaitu menentukan sandi yang akan digunakan, kunci yang dipertukarkan, algoritma autentikasi, dan kode autentikasi pesan (*MACs*). Pertukaran kunci dan algoritma autentikasi biasanya menggunakan algoritma kunci publik atau menggunakan *TLS-PSK (TLS-pre-shared key*). Kode autentikasi pesan dibangkitkan dari fungsi hash menggunakan konstruksi HMAC.

Beberapa kelebihan TLS dibandingkan dengan *SSL* adalah:

a Menggunakan algoritma hash autentikasi pesan yang lebih kuat (*HMAC*) dibandingkan dengan algoritma *MAC* yang digunakan sebelumnya pada *SSL*.

b. Pembangkitan kunci yang sudah dimodifikasi dengan menggunakan *MD5 (Message Digest 5)* dan *SHA*-1 (*Secure Hash Algorithm* 1) dengan *HMAC*.

c. Menggunakan baik *MD5* dan *SHA-1* dalam RSA *signature*.

d. Keterangan error yang lebih lengkap.

2.15. Sistem Operasi

Sistem operasi adalah penghubung antara pengguna komputer dengan perangkat keras komputer. Sebelum ada sistem operasi, pengguna komputer hanya menggunakan sinyal analog dan sinyal digital. Dengan berkembangnya ilmu pengetahuan di bidang teknologi informasi, pada saat ini terdapat banyak sistem operasi dengan keunggulan dan kekurangan masing-masing. Ada beberapa pengertian yang dapat diberikan untuk sistem operasi, antara lain:

1. *Software* yang mengendalikan *hardware*, hanya berupa program biasa. Seperti beberapa file pada *DOS* (*Disk OperatingSystem*).
2. *Software* yang menjadikan *hardware* lebih mudah untuk digunakan.
3. Kumpulan program yang mengatur kerja hardware. Seperti: permintaan *user*.
4. *Resource manager/Resource allocator*. Seperti : mengatur memori, *printer*, Dll.
5. Sebagai program pengenal. Program yang digunakan untuk mengontrol program yang lainnya.
6. Sebagai *Kernel*, yaitu program yang terus menerus berjalan selama komputer dihidupkan.
7. Sebagai *Guardian*, yaitu mengatur atau menjaga komputer dari berbagai kejahatan komputer.

2.16. *Linux Ubuntu*

Ubuntu merupakan salah satu distribusi *Linux* yang berbasiskan *Linux Debian*. *Linux Ubuntu* resmi disponsori oleh perusahaan Canonical Limited yang

merupakan perusahaan milik seorang kosmonot bernama Mark Shuttleworth. Nama Ubuntu diambil dari nama konsep ideologi di Afrika Selatan, "*Ubuntu*" berasal dari bahasa kuno Afrika, yang berarti "rasa perikemanusian terhadap sesama manusia". Tujuan dari pembuatan Linux Ubuntu adalah membawa semangat yang terkandung di dalam Filosofi Ubuntu ke dalam dunia perangkat lunak. Ubuntu adalah sistem operasi berbasis kernel Linux, tersedia secara gratis dan bebas dan mempunyai dukungan baik yang berasal dari komunitas maupun tenaga ahli di bidang IT (Ngatmono, Riasti, & Sasongko, 2015).

2.17. Flowchart

Flowchart merupakan diagram simbol yang menunjukkan arus data dan tahapan operasi dalam sebuah sistem yang digunakan baik oleh editor maupun oleh personal sistem. Ada berbagai jenis *flowchart* secara teori, namun *flowchart* yang akan digunakan dalam memecahkan permasalahan distribusi dokumen sistem informasi keuangan penerimaan dan pengeluaran kas pada penulisan ini, adalah gabungan antara *flowchart* analitik, *flowchart* dokumen dan diagram distribusi formulir. Mengingat pemisahan dan pembagian tugas merupakan elemen pengendalian internal, membutuhkan teknik untuk membagi tugas pengolahan data antar personel dan atau departemen/bagian (Ratumurun, 2015).

Adapun jenis-jenis *flowchart* yang digunakan dalam penelitian ini adalah sebagai berikut.

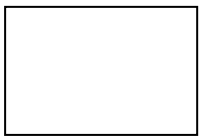
1. *Flowchart* Analitik, adalah bagan alir yang ditandai dengan penggunaan simbol yang dihubungkan dengan garis. *Flowchart* analitik mengidentifikasi

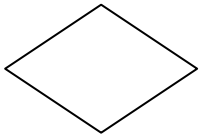
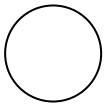
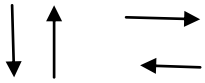
semua proses signifikan pada sebuah aplikasi, dengan penekanan pada pemrosesan tugas (Ratumurun, 2015).

2. *Flowchart* Dokumen, adalah bagan alir yang hanya terdiri dari simbol-simbol dokumen yang digunakan dalam flowchart tersebut. Tetapi, simbol lain pada dasarnya boleh saja digunakan untuk memperjelas suatu flowchart. Tujuan dari *flowchart* semacam ini adalah untuk mengetahui setiap dokumen yang digunakan dalam setiap sistem aplikasi dan mengidentifikasi titik awal dokumen, distribusi dokumen serta titik akhir setiap dokumen. Diagram distribusi formulir, adalah diagram alir yang menggambarkan distribusi setiap salinan formulir dalam sebuah organisasi.

Dalam diagram ini, penekanannya terletak pada siapa yang akan mendapatkan formulir tertentu, bukan pada bagaimana setiap formulir akan diproses (Ratumurun, 2015).

Tabel 2.2. Simbol diagram alur

	**Proses/prosesing** satu atau berapa himpunan penugasan yang akan dilaksanakan secara berurutan
	Input, data yang akan dibaca dan dimasukkan kedalam memori computer dari suatu alat input atau data yang harus melewati memori pula untuk dikeluarkan dari alat autput.
	**Terminal**, fumgsi sebagai awal (berisi ‘ Start’) dan juga sebagai akhir (berisi ‘ End’)dari suatu

	proses alur.
	**Decision,** atau kotak keputusan fumgsi untuk memutuskan arah atrau percabangan yang diambil sesuai kondisi yang dipenuhi yaitu benar atau salah.
	**Output/print,** berfungsi untuk mencetak (dan/ atau menyimpan) hasil output/ keluaran
	**Conector** /penghubung, sebagai penghubung bila diagram alur terputus di sebabkan misalnya oleh pergantian halaman.
	**Flowline,** menunjukan bagian arah intuksi dijalankan

Sumber : Ratumurun, 2015.

2.18. UML

Unified Modeling Language (UML) adalah alat perancangan sistem standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Ade Hendini, 2016).

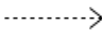
2.18.1 *Use Case Diagram*

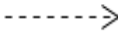
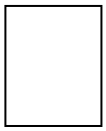
Use case Diagram menggambarkan fungsi sistem dari sudut pandang pengguna eksternal dan dalam sebuah cara yang mudah dipahami. *Use case* merupakan penyusunan kembali lingkup fungsional sistem yang disederhanakan lagi.

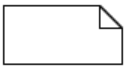
Use case diagram adalah suatu diagram yang berisi *use case diagram, actor*, serta *relationship* diantaranya. *Use Case Diagram* dapat digunakan untuk kebutuhan apa saja yang diperlukan dalam suatu sistem, sehingga sistem dapat digambarkan dengan jelas bagaimana proses dari sistem tersebut, bagaimana cara aktor menggunakan sistem, serta apa saja yang dapat dilakukan pada suatu sistem. (Indrajani, 2015 : 30).

Menurut Indrajani (2015 : 31) adapun simbol dari *use case* adalah sebagai berikut :

Tabel 2.3. Simbol U*se Case Diagram*

No	**Gambar**	**Nama**	**Keterangan**
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.

3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor

9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Sumber : Indrajani (2015 : 31).

2.18.2. *Activity Diagram*

Activity diagram menurut Indrajani (2015 : 37) adalah salah satu cara untuk memodelkan *event-event* yang terjadi dalam suatu *use case*. Diagram ini juga dapat digantikan dengan sejumlah teks.

Tabel 2.4. Simbol *Activity Diagram*

No	Gambar	Nama	Keterangan
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.

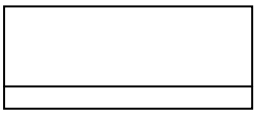
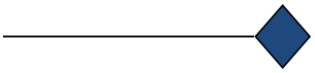
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

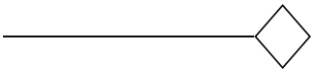
Sumber : Indrajani (2015 : 38).

2.18.3 Class Diagram

Menurut Indrajani (2015 : 35), *Class diagram* digunakan untuk menggambarkan perbedaan yang mendasar antara *clas*, hubungan antara *class*, dan di mana *sub-sistem class* tersebut. Simbol-simbol yang digunakan *class diagram* adalah sebagai berikut :

Tabel 2.5. Simbol yang digunakan dalam *Class Diagram*.

Simbol	Nama	Fungsi
	*Class*	Menggambarkan *Class* baru pada diagram.
	Association	Menggambarkan relasi antar asosiasi
	*Composition*	Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka class tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut.

	Depedency	Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain.
	*Aggregation*	*Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut sebagai relasi.

Sumber : Indrajani (2015 : 35).

BAB III

METODE PENELITIAN

3.1. Tahapan Penelitian

Adapun tahapan penelitian yang akan dilakukan dalam penulisan Skripsi ini adalah dengan metode *enkripsi AES-128* sebagai berikut :

1. Analisis Sistem

   Pada tahap ini dilakukan analisis kebutuhan dari *software* yang akan dirancang dan dibuat, meliputi analisis fungsi/proses yang dibutuhkan, analisis *output*, analisis *input*, dan analisis kebutuhan.

2. Perancangan

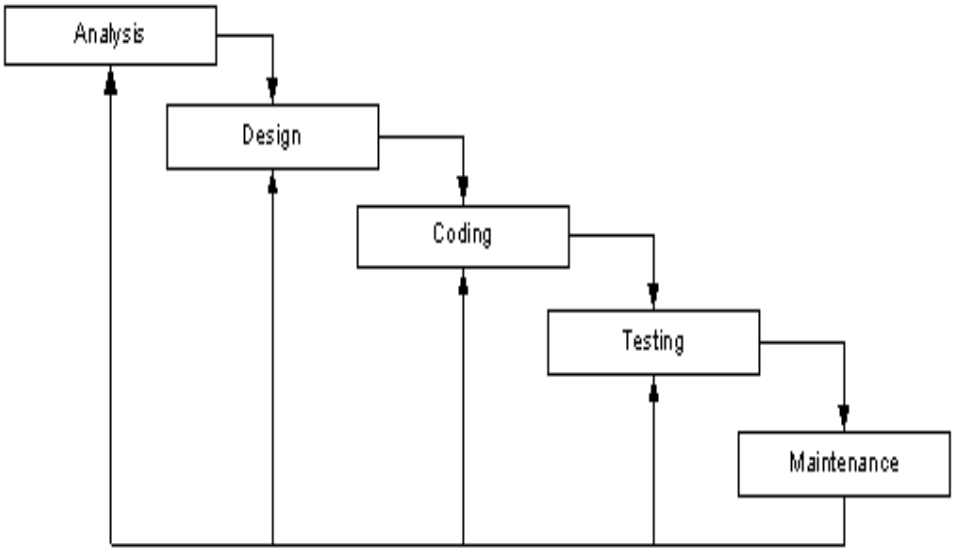
   Pada tahap ini dilakukan perancangan *software* yang bertujuan untuk memberikan gambaran apa yang seharusnya dikerjakan oleh *software* dan bagaimana tampilannya, meliputi rancangan *output*, rancangan *input*, rancangan struktur data yang digunakan, rancangan struktur *software* dan rancangan algoritma *software*.

3. Pengujian / *Testing*

   Dalam tahap ini dilakukan pengabungan modul-modul yang telah dibuat dan dilakukan pengujian atau *testing*. Pengujian ini dilakukan untuk mengetahui apakah *software* yang dibuat telah sesuai dengan desainnya dan apakah masih terdapat kesalahan atau tidak.

4. Perawatan / *Maintenance*

   Tahap ini merupakan tahapan akhir dalam model *waterfall*. *Software* yang sudah jadi dijalankan serta dilakukan pemeliharaan (*Maintenance*). Pemeliharaan ini termasuk memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya.



Gambar 3.1. Metode Perancangan *Waterfall*

3.2 Analisa Sistem Kebutuhan Sistem

Sistem yang sedang berjalan adalah *server File Transfer Protocol (FTP)* pada lingkungan sistem operasi *Linux Ubuntu* yang merupakan aplikasi yang ditempatkan pada sisi *Client* dan *Server*, dimana sistem hanya melayani permintaan pengguna terhadap *Server File Transfer Protocol (FTP)* diasumsikan sudah terpasang aplikasi *File Transfer Protocol (FTP) server* yang akan melayani permintaan pengguna.

Sharing File melakukan koneksi antar dua *host* atau komputer. Dengan hubungan yang terkoneksi maka *Sharing File* dapat melakukan pengiriman data ke *Server (upload)* atau menyalin data dari *server (download)* sebuah *file* atau lebih *file* antar komputer yang terhubung lokal maupun melalui jaringan lokal *(intranet*). *Sharing File* digunakan sebagai suatu sarana pendukung untuk pertukaran dan penyebarluasan sebuah data atau lebih dalam suatu jaringan *intranet*. Secara teknik, aplikasi *Sharing File* dibagi menjadi dua bagian yaitu :

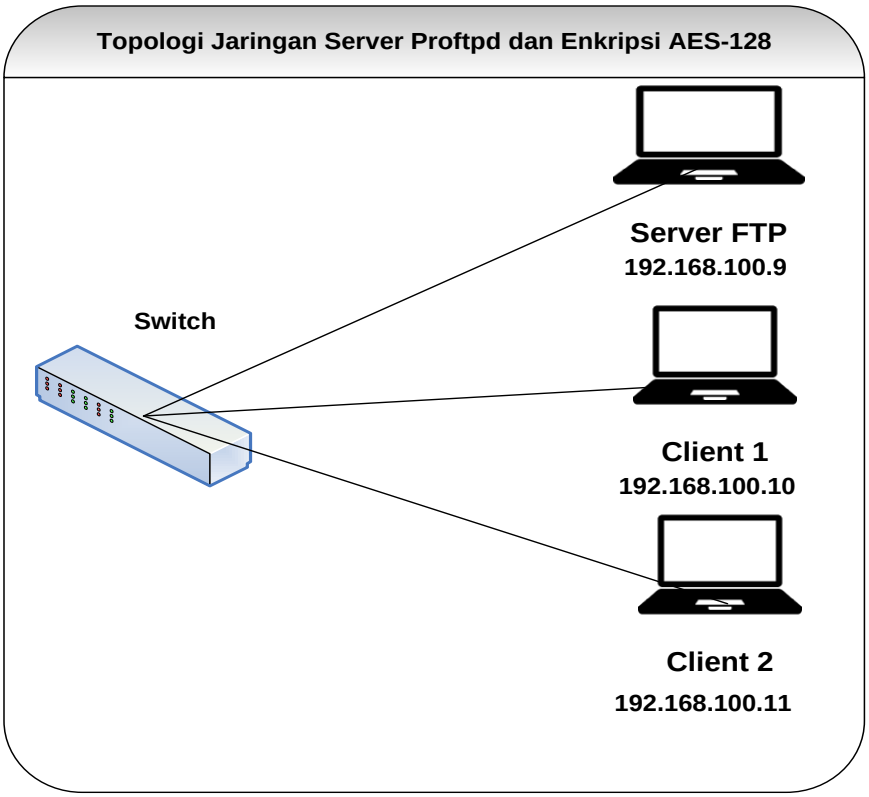
1. Aplikasi *Sharing File Client*, Aplikasi yang dijalankan pada sisi *Client*, aplikasi ini dimanfaatkan untuk transaksi *Sharing File* yang bersifat dua arah (*active Sharing*).
2. Aplikasi *Sharing File Server*, Aplikasi yang dijalankan pada sisi *Server*, aplikasi ini hanya bersifat sebagai *generator* untuk melayani semua permintaan *Sharing File Client.*

3.3. Rancangan Usulan

Alat bantu perancangan sistem yang digunakan adalah *layout* jaringan komputer, anggaran biaya, manajemen jaringan dan *security* jaringan.

1. Layout Jaringan Komputer

   Berikut di gambarkan *layout* jaringan komputer pada *server FTP* dengan *client* yang memiliki koneksi jaringan *Local Area Network* (*LAN*) yang terhubung melalui media kabel *UTP*, dimana *server FTP* di tempatkan pada ruangan yang sudah memiliki koneks jaringan internet dan *LAN* dan terdapat dua *client* komputer yang terhubung ke jaringan *LAN*.



Gambar 3.2 Layout Jaringan Komputer

Gambar diatas merupakan rancangan yang akan digunakan, berikut penjelasannya :

a. *Client* : merupakan pihak menggunakan layanan *server FTP*.

b. *Switch* : merupakan media jaringan untuk bisa mengakses *server FTP*.

c. *Server FTP* : merupakan tempat menyediakan layanan *server FTP*.

2. Anggaran Biaya

Berikut ini adalah rancangan anggaran biaya dalam pembuatan *server FTP* dan *client FTP*.

Tabel 3.1 Tabel Anggaran Biaya

No.	Nama Perangkat	Volume	Harga satuan	Total Harga
1	Laptop Asus X-441U	1 Unit	Rp. 6.000.000	Rp. 6.000.000
2	Laptop HP 15 AMD A10	1 Unit	Rp. 6.000.000	Rp. 6.000.000
3	Kabel UTP	10 Meter	Rp. 5000	Rp. 50.000
4	Konektor RJ 45	1 Plastik	Rp. 50.000	Rp. 50.000
5	Switch Hub Merk TP-Link	1 Unit	Rp. 100.000	Rp. 100.000
	Total Harga		**Rp. 12.200.000,-**	

3. Manajemen Jaringan

   Berikut ini adalah rancangan manajemen jaringan dalam pembuatan jaringan komputer *server FTP*.

Tabel 3.2 Manajemen Jaringan

No.	Nama Client	IP Adress	Subnet Mask	Gateway
1	Server FTP Ubuntu	192.168.100.9	255.255.255.0	192.168.100.1
2	Modem Internet	192.168.100.1	255.255.255.0	192.168.100.1
3	Client 1	192.168.100.10	255.255.255.0	192.168.100.1
4	Client 2	192.168.100.11	255.255.255.0	192.168.100.1

4. Security Jaringan

Security Jaringan yang digunakan dalam penelitian ini adalah *IPSec Tunnel* yang merupakan jenis *protocol* yang mengintegrasikan *fitur security* yang meliputi proses authentifikasi, integritas, dan kepastian ke dalam *Internet Protocol (IP)*. Dimana proses tersebut dilakukan pada *network layer* atau layer ketiga dalam model *OSI*. Dengan menggunakan *IPSec Tunnel*, kita dapat melakukan enkripsi dan atau membuat media komunikasi (*tunnel*) terautentifikasi tergantung pada kondisi protocol yang diinginkan oleh dua *peer* tersebut. *Protocol* tersebut antara lain:

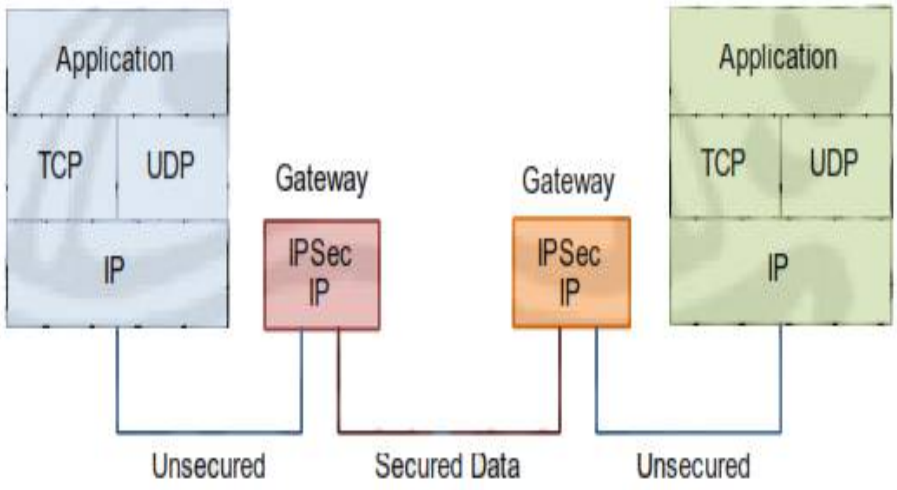
a. *Authentification Header (AH)*

Merupakan autentifikasi sumber data dan proteksi terhadap pencurian data. *Protocol AH* dibuat dengan melakukan enkapsulasi paket IP asli ke dalam paket baru yang mengandung *IP Header* yang baru yaitu *AH Header* disertai dengan *header* yang asli. Isi data yang dikirimkan melalui *protocol AH* bersifat *clear text* sehingga *tunnel* yang berdasar pada protocol *AH* ini tidak menyediakan kepastian data.

b. *Tunnel Mode*

Merupakan *Security Association* yang diimplementasikan pada dua *gateway IPSec Tunnel*. Pada mode ini terdapat *IP Header* tambahan di luar yang menspesifikasikan tujuan pemrosesan *IPSec* ditambah *IP Header* tambahan di dalam yang menunjukkan alamat tujuan paket yang sebenarnya. *Header protocol* keamanan akan tampak pada

bagian setelah *IP Header* tambahan di luar dan sebelum *IP Header* tambahan di dalam.



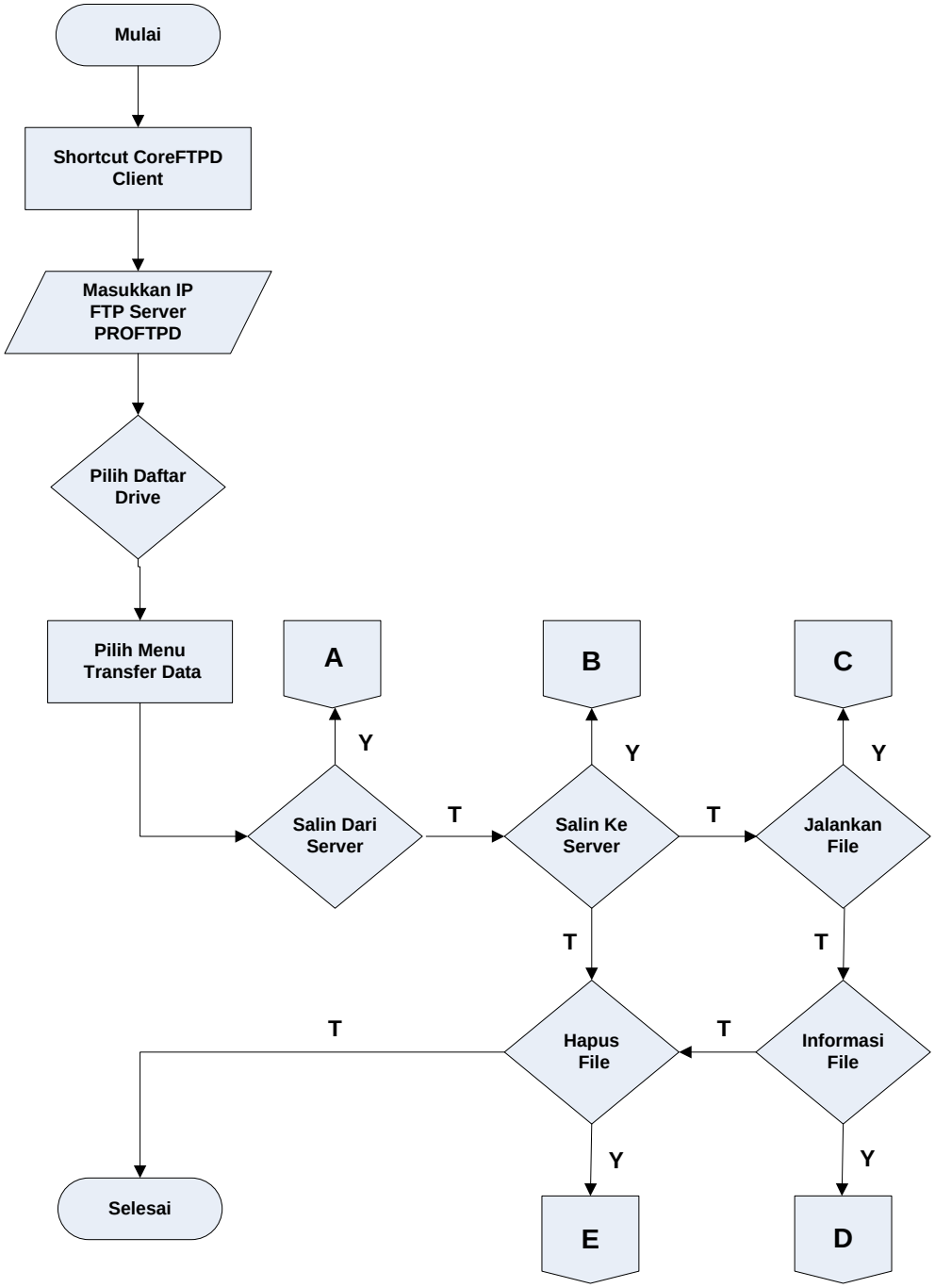
Gambar 3.3 Tunnel Mode IPSec.

3.4. Rancangan *Flowchart*

Flowchart program dimulai dengan menampilkan *Form* Utama *CoreFTP Client*. Pada *Form* Utama, pengguna dapat menginputkan *IP* dari *Server* yang dituju, kemudian pilih daftar *drive* untuk menyalin data baik dari *Clien*t ke *Server* (*Upload*) maupun *Server* Ke *Client* (*Download*).

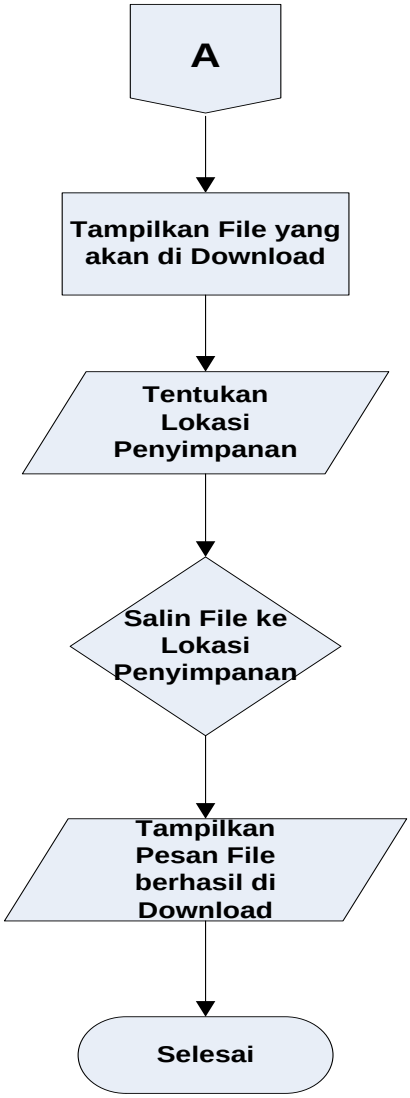
Berdasarkan Input *IP Server* yang dituju, maka akan menampilkan data baik dari *Server* maupun *Clien*t, kemudian sistem akan memproses perintah yang diberikan *User*, baik itu berupa penyalinan data dari *serve*r (*upload*) maupun penyalinan data ke *Client (download)*, menjalankan *file* pada Sisi *Server*, kemudian dapat melakukan penghapusan *file* baik dari *Server* maupun *Client* .

1. *Flowchart Menu* Utama



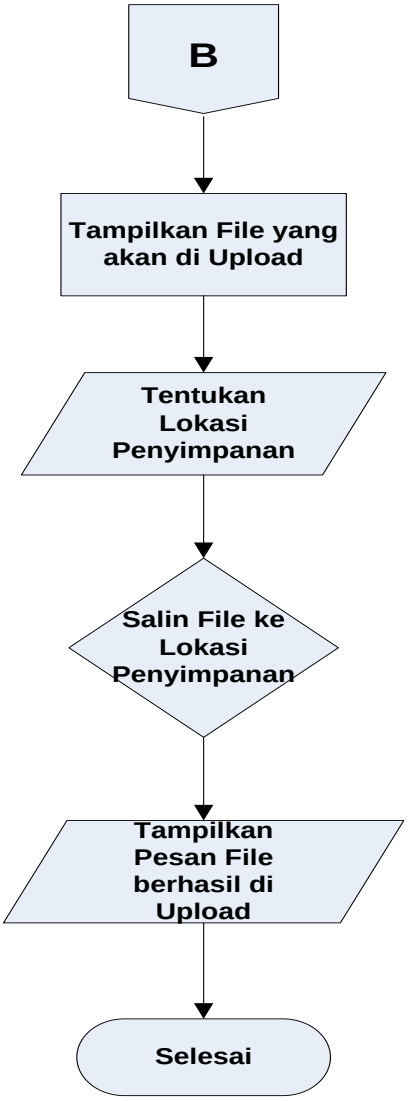
Gambar 3.4 *Flowchart Menu* Utama *CoreFTP Client*

2. *Flowchart* Proses *Download*



Gambar 3.5 Flowchart Proses Download.

3. *Flowchart* Proses *Upload*



Gambar 3.6 *Flowchart* Proses *Upload*.

3.5 Konfigurasi *TCP/IP Address*

Tujuan Konfigurasi *IP Address* adalah memberi alamat untuk sebuah *server* atau komputer dalam suatu jaringan. Secara sederhana agar komputer dalam jaringan dapat dikenali oleh semua *client* dan dirinya sendiri harus diberi alamat, alamat inilah yang dimaksud dengan *IP Address*. *IP Address* adalah nomor tertentu yang nantinya dijadikan patokan untuk memberi alamat pada *client* yang ada dalam suatu jaringan *LAN* berbasis *client server* ataupun *workgroup*.

3.5.1 Konfigurasi IP Address Pada Server

Agar komputer *server* bisa dikenali, maka harus diberi alamat berupa *IP Address*. Prosedur yang harus dilakukan adalah sebagai berikut :

1. Untuk membuat konfigurasi *IP Address* secara permanen, maka dilakukan pengeditan *file interfaces* yang lokasinya ada di "***/etc/network/interfaces***". Caranya, jalankan perintah nano untuk membuka file tersebut, lalu tambahkan konfigurasi yang diinginkan, seperti gambar 3.6 dibawah ini :

```
root@ubuntu:~# nano /etc/network/interfaces
```

Gambar 3.6 Perintah konfigurasi IP Address Pada Ubuntu Server.

2. Setelah file interfaces terbuka, berikan konfigurasi yang kita inginkan. Misalnya seperti dibawah ini..

```
# The loopback network interface
auto lo
iface lo inet loopback
auto eth1
iface eth1 inet static
```

```
address 192.168.1.1
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
gateway 192.168.1.254
dns-nameservers 8.8.8.8
```

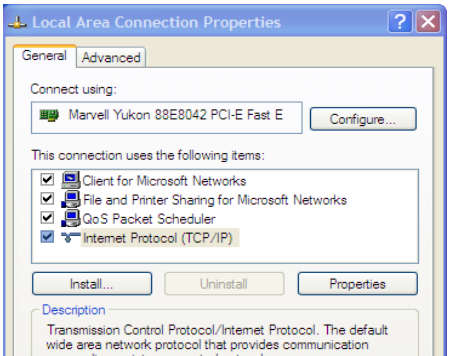
3. Setelah selesai memberikan konfigurasi, simpan perubahan dengan menekan tombol **Ctrl+O** (*save*). Lalu keluar dengan menekan tombol **Ctrl+X** (Exit). Kemudian restart *service* nya agar konfigurasi tadi bisa beroperasi. Perintahnya lihat di bawah ini

```
root@ubuntu:~# /etc/init.d/networking restart
* Running /etc/init.d/networking restart is deprecated
because it may not enable again some interfaces
* Reconfiguring network interfaces...
Ignoring unknown interface eth1=eth1.  [ OK ]
root@ubuntu:~#
```

3.5.2 Konfigurasi IP Address Pada Client

Agar komputer *client* bisa dikenali, maka harus diberi alamat berupa *IP Address*. Prosedur yang harus dilakukan adalah sebagai berikut :

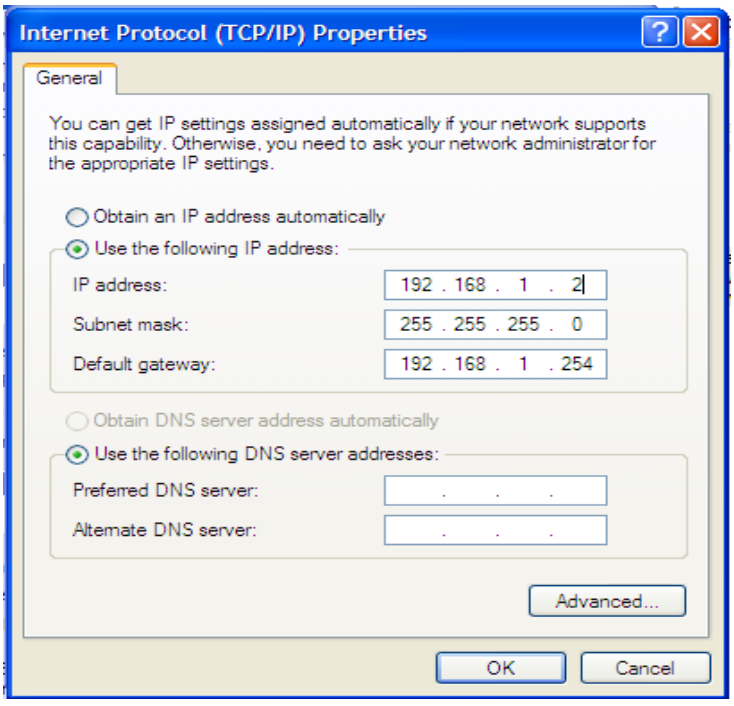
1. Dari Tombol *Start* kemudian pilih *Control Panel*, *Network Connection* kemudian Pilih *Local Area Connection*. Setelah itu akan tampil kotak dialog *Local Area Connection Status* seperti gambar 3.7 dibawah ini :



Gambar 3.7 Kotak Dialog Local Area Connection Properties Pada *Client*

2. Klik *Internet Protocol (TCP/IP).*

3. Klik Properties. Setelah itu akan tampil kotak dialog *Internet Protocol (TCP/IP) Properties* seperti gambar 3.8 berikut :



Gambar 3.8 Kotak dialog Internet Protocol (TCP/IP) Properties Pada *Client*.

6. Klik "*Use the following IP address*".

7. Ketikkan di kolom *IP Address* 192.168.1.2.

8. Klik tab di papan ketik.

9. Kolom *Subnet mask* tidak perlu diisi, dengan menekan tab *Subnet mask* 255.255.255.0 secara otomatis sudah terisi.
10. Ketikkan di Kolom *Default Gateway* 192.168.1.254.

3.6 Perancangan Aplikasi File Transfer Protocol (FTP) Server

Aplikasi *file transfer protocol* yang digunakan adalah menggunakan *service proftpd* yang sudah dilakukan penginstalan pada *protocol FTP* tersebut, sehingga pada *protocol FTP* sudah dibuat *shortcut* untuk melakukan penggunaan aplikasi *protocol FTP* tersebut. *Software* tambahan yang digunakan melakukan koneksi ke *server FTP* dan melakukan kegiatan *file transfer protocol* adalah menggunakan *CoreFTPD Client* versi 2.2.

Gambar 3.9 *Shortcut* Aplikasi *FTP Server*

3.6.1. Shortcut CoreFTP CLient

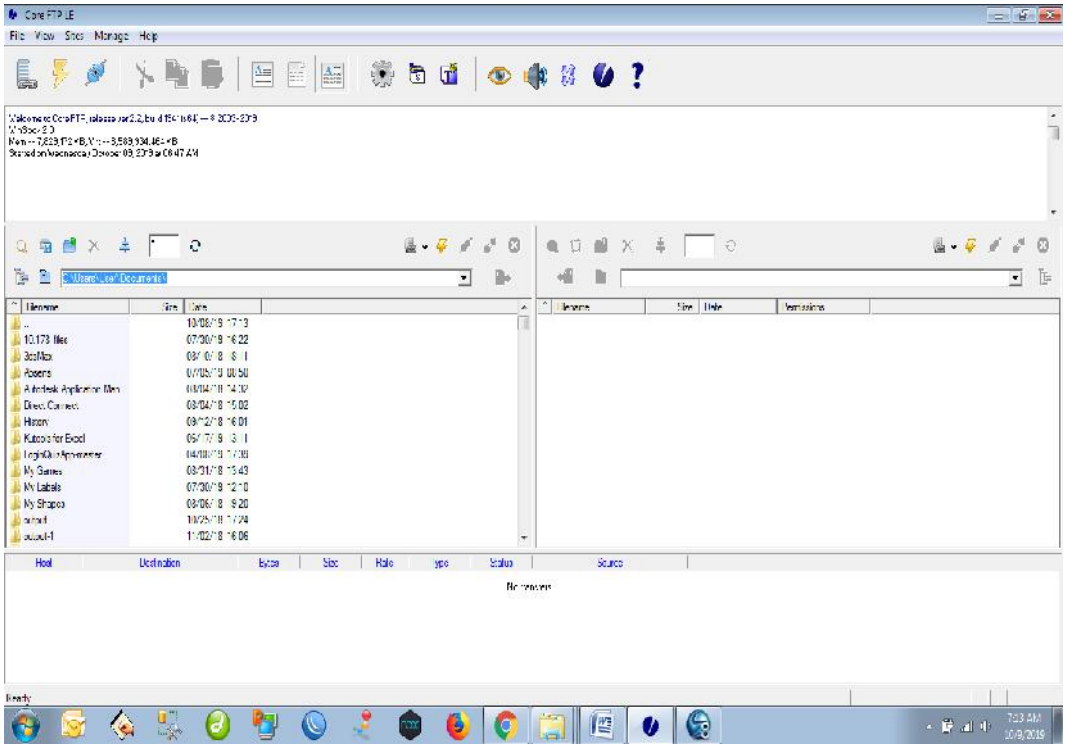
File Protocol : FTP

Hostname : 192.168.1.1

Port Number : 21

Username : pro

Password : pro123



Gambar 3.10 *Shortcut* Aplikasi *CoreFTP Client.*

BAB IV

HASIL DAN PEMBAHASAN

4.1 Kebutuhan Spesifikasi Mininum Hardware dan Software

Agar sistem perancangan yang telah kita kerjakan dapat berjalan baik atau tidak, maka perlu kiranya dilakukan pengujian terhadap sistem yang telah kita kerjakan. Untuk itu dibutuhkan beberapa komponen utama mencakup perangkat keras (*hardware*), perangkat lunak (*software*).

1. Perangkat Keras *(Hardware)*

   a. Notebook Sebagai *Server*

   Merk : Asus X-441U

   Platform : *Notebook PC*

   Processor : *Intel Core i3 6006, 2.0 GHz*

   Memory : 4 GB *DDR3*

   HDD : 500 Gb *Seagate*

   Networking : *WLAN 802.11 b/g/n*

   Network Speed : 10 / 100 *Mbps*

   Interface Provided : *2x USB2.0, eSATA/USB, VGA, Audio, LAN*

   Operating System : *Linux Ubuntu Desktop 18.0*

   b. Notebook Sebagai *Client*

   Merk : HP

   Platform : *Notebook*

   Processor : *Intel AMD A10 2.0 GHz, T5870*

Memory : *8 GB DDR3*

HDD : 1000 Gb *Seagate*

Networking : WLAN 802.11 b/g/n

Network Speed : 10 / 100 Mbps

Interface Provided : *2x USB2.0, eSATA/USB, VGA, Audio, LAN*

Operating System : Windows 7

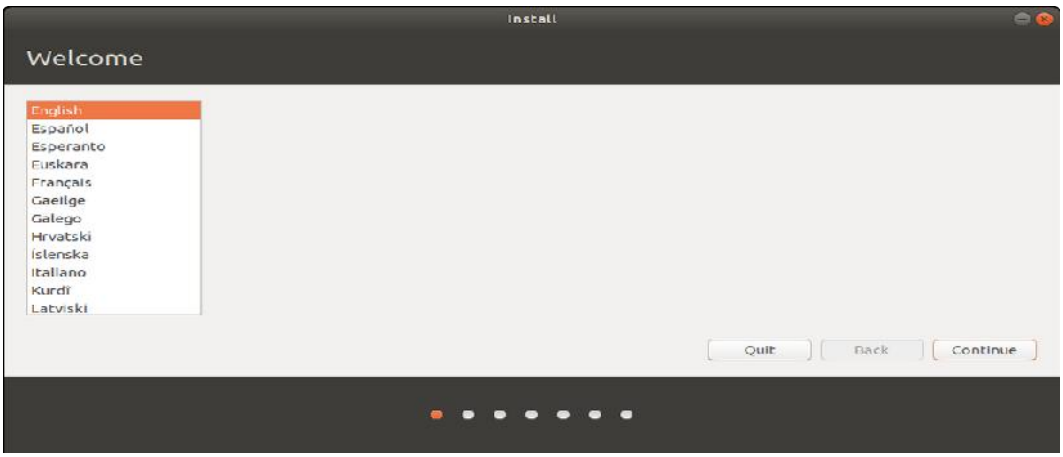
2. Perangkat Lunak *(Software)*
   a. Perangkat Lunak Pada *Server*
      1) Sistem operasi *Linux Ubuntu* versi 12.0 atau lebih tinggi
      2) *Modul FTP ProFTPD* merupakan aplikasi *FTP server* yang di-install pada *server Linux Ubuntu* yang digunakan untuk *file transfer*.
      3) Untuk pembuatan sertifikat *TLS* menggunakan *OpenSSL*.
   b. Perangkat Lunak Pada *Client*
      1) Sistem operasi *Windows 7* atau lebih tinggi
      2) *Software CoreFTP Client.*
      3) *Web Browser Mozilla Firefox atau Google Chrome*

3. Pengguna (*Brainwar*e)
   a. Mampu Mengoperasikan komputer dengan baik, dalam artian memiliki pengetahuan yang cukup mengenai komputer.
   b. Memiliki pengetahuan yang cukup tentang proses-proses yang akan dilakukan pada saat terkoneksi dengan *server FTP*.

4.2 Instalasi Ubuntu Desktop

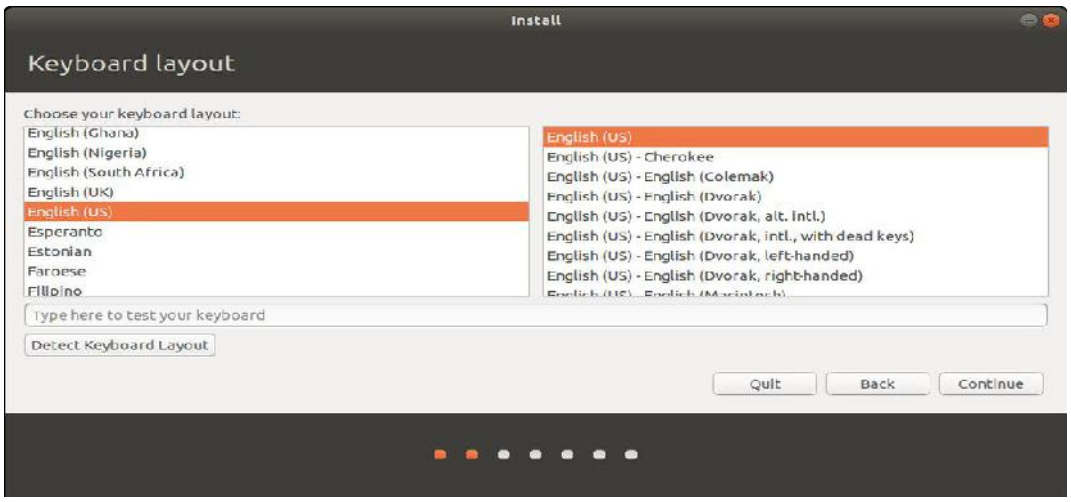
Untuk melakukan penginstallan sistem operasi *Linux Ubuntu Desktop* Versi *18.04* diperlukan langkah-langkah di bawah ini :

1. Pada saat proses instalasi dapat dilakukan melalui *USB flashdisk* atau *CD Room*, pada saat booting melalui *USB Flashdisk* dan akan langsung muncul tampilan pemilihan bahasa maka akan tampil sperti pada gambar 4.1.



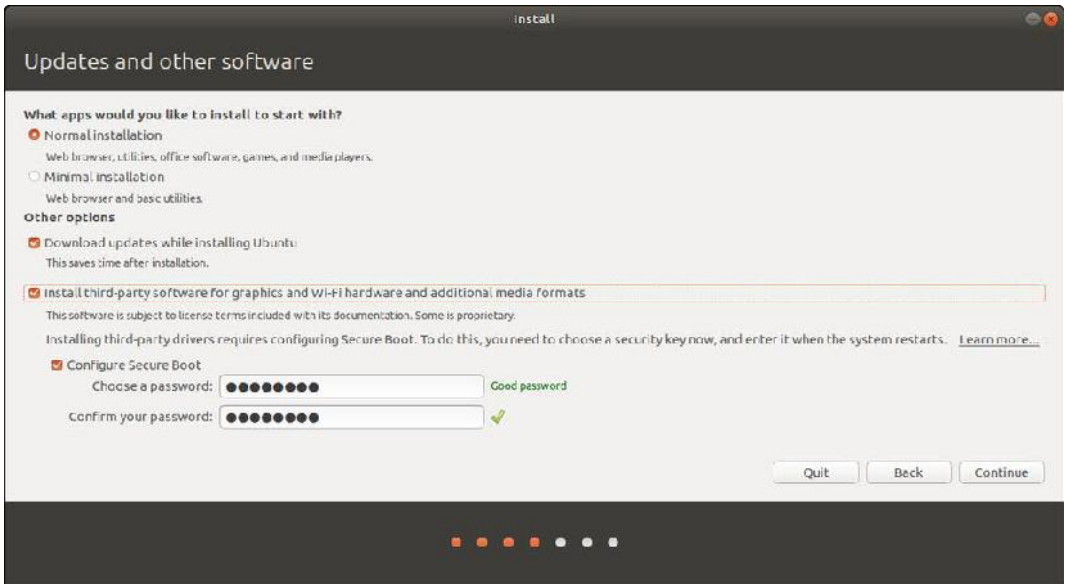
Gambar 4.1 Proses Pemilihan Bahasa.

2. Kemudian akan tampil pemilihan tata letak *keyboard*, pada proses ini pilih *default* atau pilih seperti pada gambar 4.2.



Gambar 4.2 Kotak dialog proses pemilihan tata letak *keyboard*.

3. Kemudian pilih tipe instalasi Instalasi Normal untuk mendapatkan fitur-fitur dari *Ubuntu* seperti pada gambar 4.3.

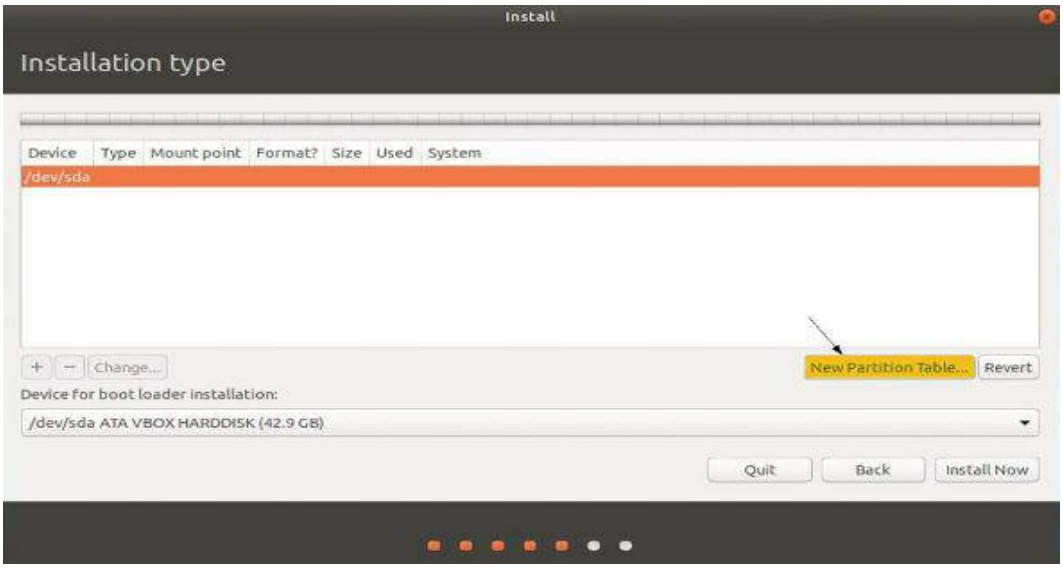


Gambar 4.3 Kotak dialog proses pemilihan tipe instalasi *Ubuntu*.

4. Kemudian akan disajikan beberapa opsi instalasi, kemudian pilih Something Else – jika ingin secara manual membuat partisi sendiri dan ingin menginstal *Ubuntu* bersama dengan *OS* yang ada atau membuat *Dual Booting (Windows* dan *Ubuntu*) seperti pada gambar 4.4.

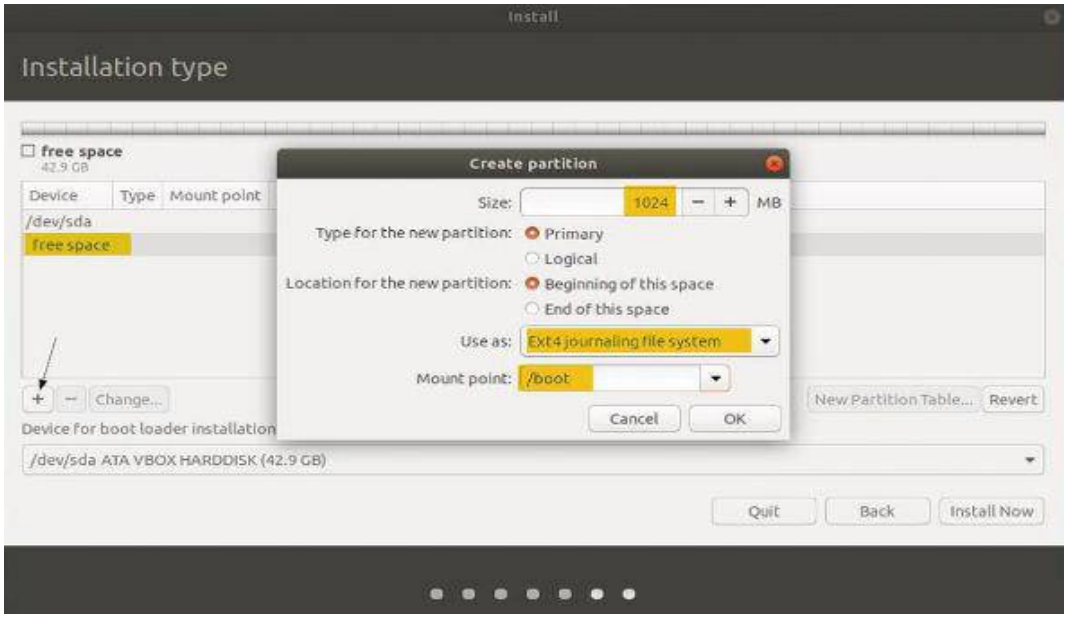
Gambar 4.4 Kotak dialog proses pemilihan opsi instalasi *Ubuntu*.

5. Kemudian untuk membuat partisi sendiri, klik "*New Partitions Table*" seperti pada gambar 4.5.



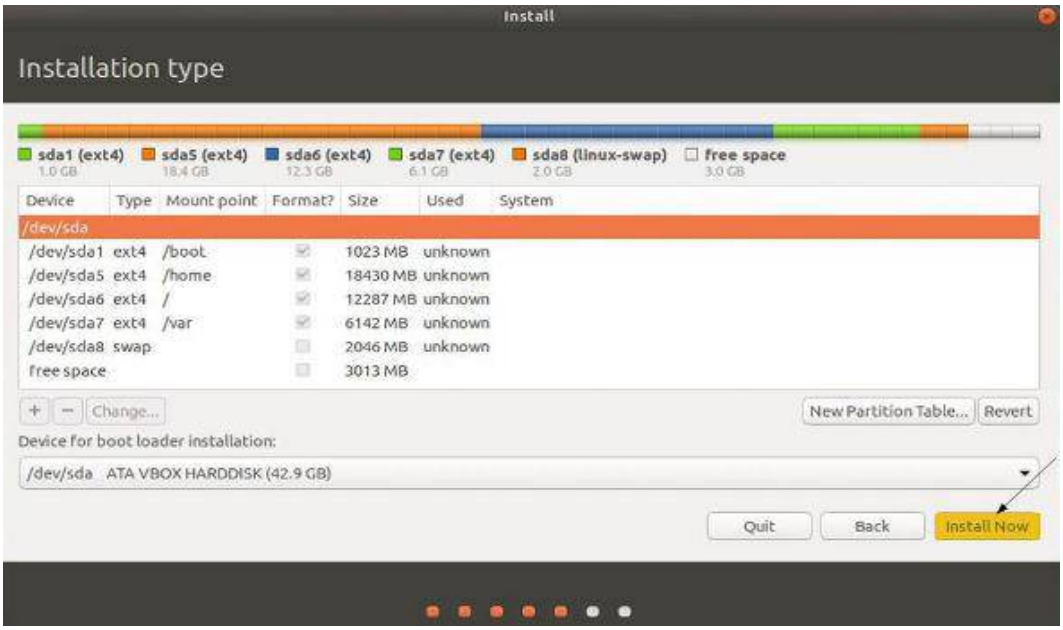
Gambar 4.5 Kotak dialog proses pembuatan partisi.

6. Kemudian pembuatan /boot partisi ukuran 1GB, Pilih ruang kosong lalu Klik pada simbol "+" untuk membuat partisi baru seperti pada gambar 4.6.



Gambar 4.6 Kotak dialog proses pembuatan *folder* penyimpanan *ftp*.

7. Kemudian setelah selesai pembuatan partisi, lalu klik opsi "*Install Now*" untuk melanjutkan instalasi seperti pada gambar 4.7.

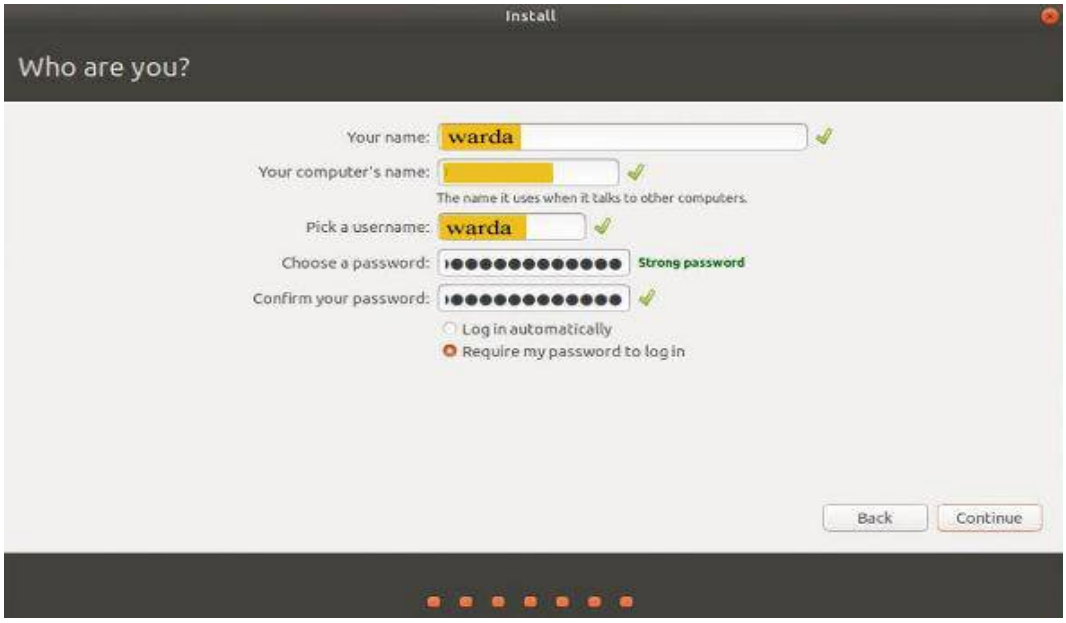


Gambar 4.7 Kotak dialog proses instalasi Ubuntu.

8. Kemudian Pilih zona waktu (Jika Indonesia maka pilih GMT+7 Jakarta) dan kemudian klik "*Continue*"seperti pada gambar 4.8.

Gambar 4.8 Kotak dialog pemilihan zona waktu.

9. Kemudian memberikan kredensial pengguna, pada tampilan ini berikan nama, nama komputer, nama pengguna, dan kata sandi Anda untuk masuk ke *Ubuntu* 18.04 *LTS* dan kemudian klik "*Continue*" seperti pada gambar 4.9.

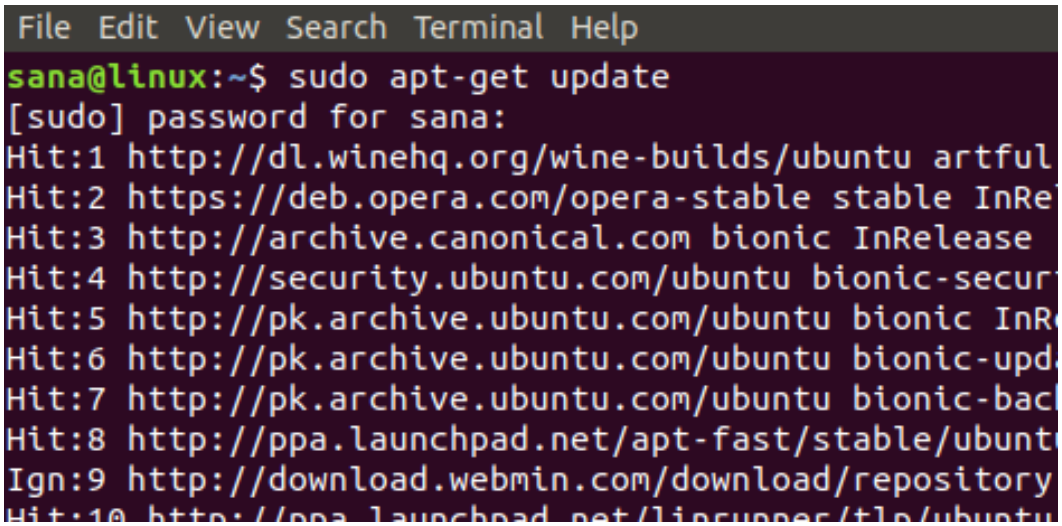


Gambar 4.9 Kotak dialog pengisian *username* dan *password*.

10. Kemudian Klik "*Continue*" untuk memulai proses instalasi seperti pada gambar 4.10.

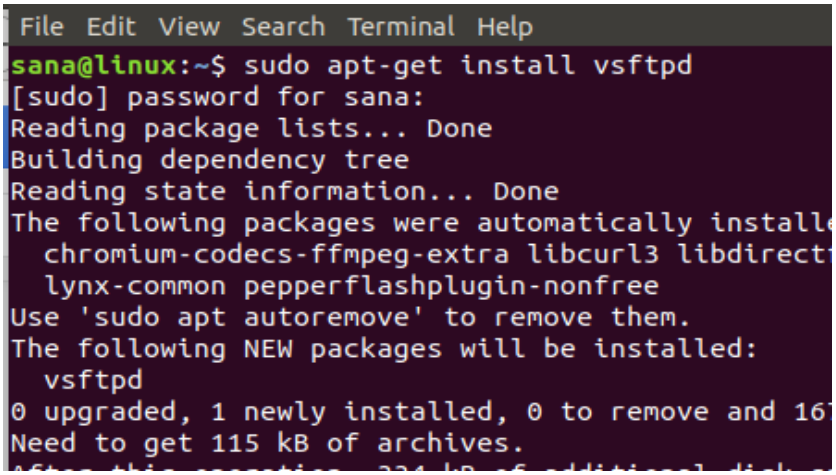
Gambar 4.10 Kotak dialog *Login* untuk membuka *ftp* dengan *windows explorer*.

11. Setelah proses instalasi *Ubuntu* selesai, maka tahap berikutnya adalah *update* sistem operasi *Ubuntu*, seperti pada gambar 4.11.



Gambar 4.11 Kotak dialog proses *update ubuntu.*

12. Kemudian tahap berikutnya adalah instalasi *protocol FTP VSFTPD*, seperti pada gambar 4.12.



Gambar 4.12 Kotak dialog proses *install VSFTPD.*

13. Kemudian dilanjutkan dengan proses konfigurasi *VSFTPD* seperti pada gambar 4.13.

```
sana@linux:~$ sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.orig
sana@linux:~$
```

Gambar 4.13 Kotak dialog proses konfigurasi *VSFTPD.*

14. Kemudian dilanjutkan dengan proses konfigurasi *VSFTPD* seperti pada gambar 4.14.

```
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                  /etc/vsftd.conf                  Modifie

write_enable=YES
local_umask=022
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
chroot_local_user=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=NO
pasv_enable=Yes
pasv_min_port=10000
pasv_max_port=10100
allow_writeable_chroot=YES



^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
```

Gambar 4.14 Kotak dialog proses konfigurasi *VSFTPD.*

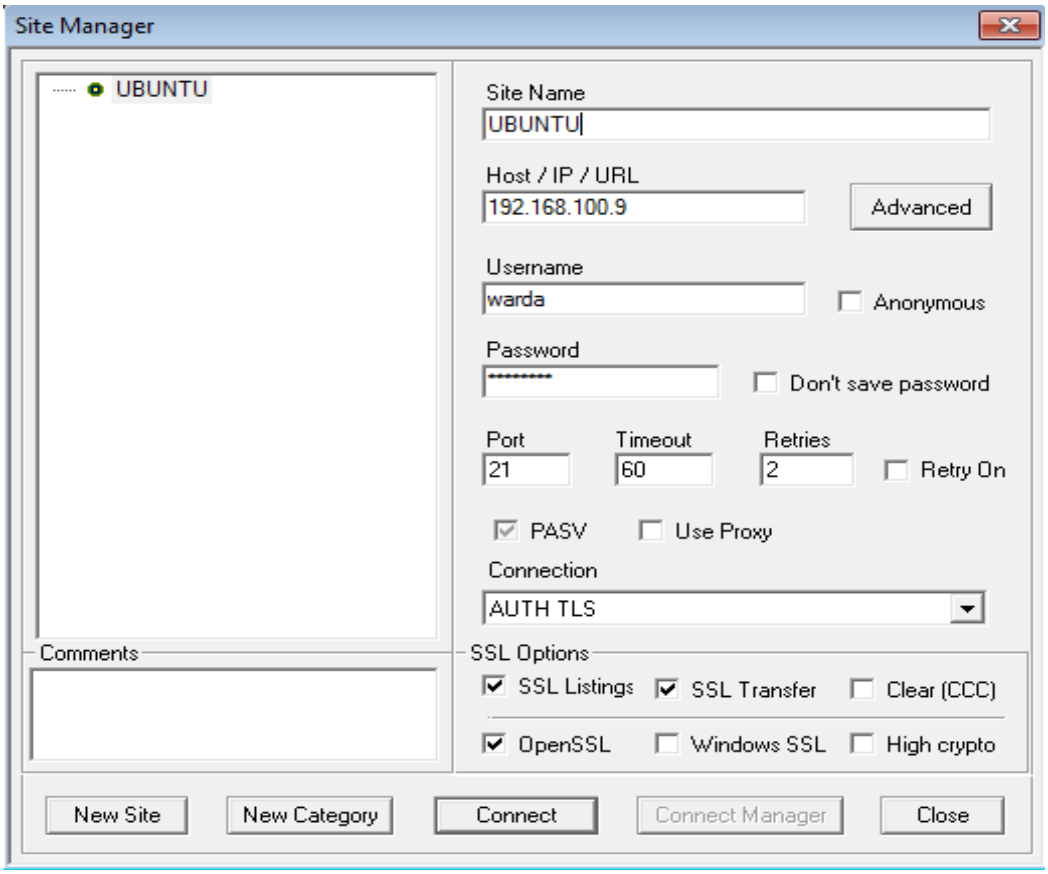
15. Kemudian dilanjutkan dengan proses *restart service VSFTPD* seperti pada gambar 4.15.

```
linuxuser@ubuntu16-hostadvice:~$ sudo systemctl restart vsftpd
[sudo] password for linuxuser:
linuxuser@ubuntu16-hostadvice:~$ sudo systemctl status vsftpd
● vsftpd.service - vsftpd FTP server
   Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset: e
   Active: active (running) since Thu 2018-04-19 15:39:48 UTC; 8s ago
  Process: 6339 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, s
 Main PID: 6343 (vsftpd)
    Tasks: 1
   Memory: 552.0K
      CPU: 3ms
   CGroup: /system.slice/vsftpd.service
           └─6343 /usr/sbin/vsftpd /etc/vsftpd.conf

Apr 19 15:39:48 ubuntu16-hostadvice systemd[1]: Starting vsftpd FTP server...
Apr 19 15:39:48 ubuntu16-hostadvice systemd[1]: Started vsftpd FTP server.
lines 1-13/13 (END)
```

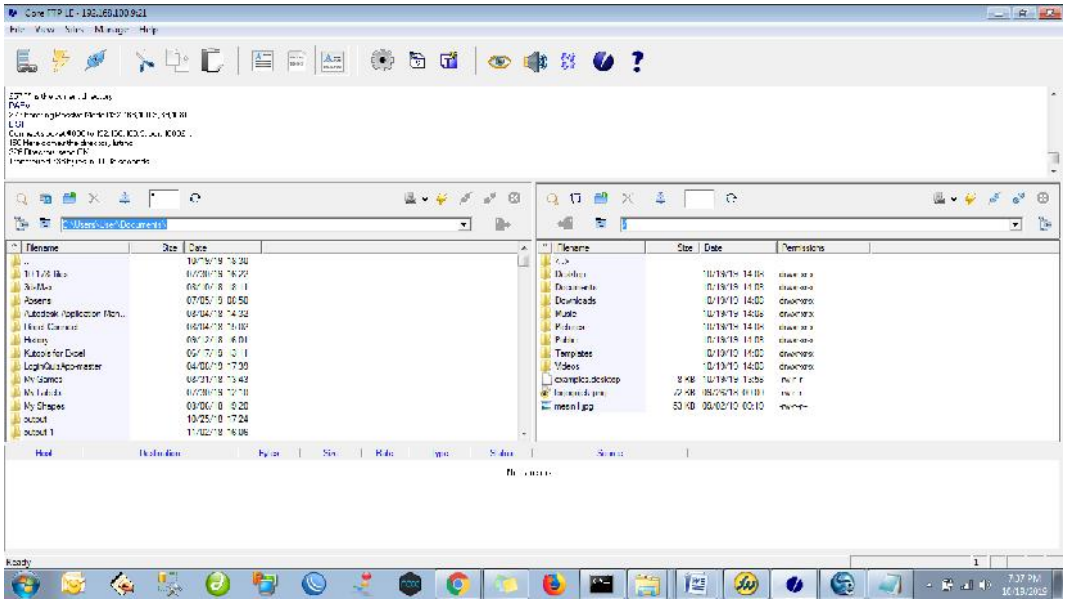
Gambar 4.15 Kotak dialog proses *restart service VSFTPD.*

16. Proses penambahan koneksi *server FTP* menggunakan *software CoreFTP* seperti pada gambar 4.16.



Gambar 4.16 Kotak dialog proses penambahan *server* menggunakan *CoreFTP*.

17. Tahap berikutnya adalah melakukan uji coba *transfer file* baik itu proses *download file* dari *server* maupun *upload file* ke *server* menggunakan *software CoreFTP* seperti pada gambar 4.17.



Gambar 4.17 Kotak dialog proses uji coba *transfer file* menggunakan *CoreFTP*.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil rancang bangun keamanan *FTP Server* Menggunakan *Enkripsi AES-128* pada *Transport Layer Security* menggunakan sistem operasi *Linux Ubuntu* ini, penulis menarik kesimpulan sebagai berikut :

1. Bahwa dengan menggunakan sistem operasi *Linux Ubuntu* dan *protocol VSFTPD* dan *security Transport Layer Security OpenSSL* yang diinstall pada sisi *server* dan aplikasi *CoreFTP* pada *client* dapat dihasilkan *server* aplikasi *File Transfer Protocol (FTP)* yang aman karena *file* sudah dikirim dapat terenkripsi dengan baik.
2. Bahwa dalam prinsipnya dalam proses pengiriman data baik pengiriman data *download* maupun *upload*, *file* yang dikirim harus sama dengan *file* yang diterima.
3. *TCP/IP* (*Transmission Control Protocol* / *Internet Protocol*) adalah standar komunikasi data yang digunakan oleh komunitas *internet* dalam proses tukar-menukar data dari satu komputer ke komputer lain di dalam jaringan *Internet.*
4. Bahwa dalam melakukan pengiriman data antara *client* dengan *server* dapat menggunakan fasilitas *File Transfer Protocol (FTP)* berbasis *TCP/IP*.
5. Kecepatan *transfer* berbanding lurus dengan besaran ukuran *file* yang di *transfer*, baik itu proses *download* maupun *upload*.

6. Untuk tingkat keamanan pada *protocol FTP VSFTPD* masih memiliki celah dalam proses *transfer* filenya, sehingga dibutuhkan aplikasi tambahan yaitu *openssl* sebagai aplikasi tambahan *security Transport Layer Security (TLS).*

5.2 Saran

Adapun saran yang ingin penulis berikan berdasarkan hasil implementasi perancangan perangkat lunak ini adalah sebagai berikut :

1. Untuk meningkatkan keamanan dalam proses transfer diwajibkan untuk menggunakan aplikasi tambahan yaitu *Transport Layer Security*.
2. Sebaiknya dilakukan proses pemecahan terhadap file yang akan ditransfer jika file tersebut berukuran besar (dalam satuan GB). Hal ini untuk mengurangi waktu proses *transfer file* tersebut. Selain itu, hal ini dapat menghemat pemakaian memori sistem sehingga tidak mengakibatkan komputer menjadi *hang* pada saat melakukan transfer file.
3. Menggunakan *password* dengan tingkat kekuatan tinggi, yaitu dengan panjang minimal 6 karakter dan terdiri dari huruf dan angka.
4. Sebaiknya dilakukan pembaharuan perangkat lunak sistem operasi *Linux* dan *service Protocol FTP* secara rutin.
5. Untuk pengembangan selanjutnya, dapat ditambahkan fasilitas untuk mengenkripsi suatu *file* yang akan di *transfer* dan dapat melakukan penghapusan *folder*.

DAFTAR PUSTAKA

Ade Hendini, 2016. *"Pemodelan UML Sistem Informasi Monitoring Penjualan Dan Stok Barang (Studi Kasus: Distro Zhezha Pontianak)"*. Jurnal Mahasiswa Program Studi Manajemen Informatika AMIK BSI Pontianak.

Angga, Aditya Permana, Dkk. 2018. *"Rancangan Aplikasi Pengamanan Data Dengan Algoritma Advanced Encyption Standard (AES)"*. Jurnal Mahasiswa Program Studi Teknik Informatika Universitas Muhammadiyah Tangerang.

Anthony, A., Tanaamah, A. R., & Wijaya, A. F. (2017). *"Analisis Dan Perancangan Sistem Informasi Penjualan Berdasarkan Stok Gudang Berbasis Client Server (Studi Kasus Toko Grosir "Restu Anda"). Jurnal Teknologi Informasi Dan Ilmu Komputer*, *4*(2), 136. https://doi.org/10.25126/jtiik.201742321.

Anti, Ulan Ari, Dkk. 2017. *"STEGANOGRAFI PADA VIDEO MENGGUNAKAN METODELEAST SIGNIFICANT BIT (LSB) DAN END OF FILE (EOF)"*. Jurnal Mahasiswa Program Studi Ilmu Komputer Universitas Mulawarman.

Arman, M. (2017). *"Rancang Bangun Pengamanan FTP Server dengan Menggunakan Secure Sockets Layer". Jurnal Integrasi*, *9*(1), 16. https://doi.org/10.30871/ji.v9i1.272.

Fachri, barany, agus perdana windarto, and ikhsan parinduri. "penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik." jepin (jurnal edukasi dan penelitian informatika) 5.2 (2019): 202-208.

Fachri, b., windarto, a. P., & parinduri, i. (2019). Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 5(2), 202-208.

Fachri, barany; windarto, agus perdana; parinduri, ikhsan. Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 2019, 5.2: 202-208.

Fadhli, M., Munshi, F. A., & Wicaksono, T. A. (2016). Ancaman Keamanan pada Transport Layer Security. *Jurnal ULTIMA Computing*, *7*(2), 70–75. https://doi.org/10.31937/sk.v7i2.234.

Hamdi, nurul. "model penyiraman otomatis pada tanaman cabe rawit berbasis programmable logic control." jurnal ilmiah core it: community research information technology 7.2 (2019).

Indrajani, 2015. "*Perencanaan Basis Data dalam All in 1"*. Jakarta : Elex Media Komputindo.

K, G. G. P. U., & Erlanshari, A. (2016). *Implementasi Metode A Dvanced Encryption Standard (AES) Dan Message Digest 5 (MD5) Pada Enkripsi Dokumen (Studi Kasus LPSE UNIB)*. *4*(3), 277–287.

Ngatmono, D., Riasti, B. K., & Sasongko, D. (2015). Membangun Sistem Operasi Mandiri Berbasis Open Source Dengan Metode Remaster. *Indonesian Journal on Networking and Security*, *4*(3), 39–47.

Pabokory, F. N., Astuti, I. F., & Kridalaksana, A. H. (2016). Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File

Dokumen Menggunakan Algoritma Advanced Encryption Standard. *Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer*, *10*(1), 20. https://doi.org/10.30872/jim.v10i1.23

Permana, aminuddin indra. "kombinasi algoritma kriptografi one time pad dengan generate random keys dan vigenere cipher dengan kunci em2b." (2019).

Putra, randi rian. "sistem informasi web pariwisata hutan mangrove di kelurahan belawan sicanang kecamatan medan belawan sebagai media promosi." jurnal ilmiah core it: community research information technology 7.2 (2019).

Putra, randi rian, et al. "decision support system in selecting additional employees using multi-factor evaluation process method." (2019).

Putra, randi rian. "implementasi metode backpropagation jaringan saraf tiruan dalam memprediksi pola pengunjung terhadap transaksi." jurti (jurnal teknologi informasi) 3.1 (2019): 16-20.

Rahadjeng, I. R., & Puspitasari, R. (2018). ANALISIS JARINGAN LOCAL AREA NETWORK (LAN) PADA PT. MUSTIKA RATU Tbk JAKARTA TIMUR. *Prosisko*, *5*(1), 53–60.

Ratumurun, S. (2015). Sistem Informasi Akuntansi Permintaan Barang dari Gudang pada PT. Mauwasa Sejahtera Ambon. *Cita Ekonomika, Jurnal Ekonomi*, *IX*(1), 57–64.

Saputra, muhammad juanda, and nurul hamdi. "rancang bangun aplikasi sejarah kebudayaan aceh berbasis android studi kasus dinas kebudayaan dan pariwisata aceh." journal of informatics and computer science 5.2 (2019): 147-157

Sidik, a. P., efendi, s., & suherman, s. (2019, june). Improving one-time pad algorithm on shamir's three-pass protocol scheme by using rsa and elgamal algorithms. In journal of physics: conference series (vol. 1235, no. 1, p. 012007). Iop publishing.

Sitepu, n. B., zarlis, m., efendi, s., & dhany, h. W. (2019, august). Analysis of decision tree and smooth support vector machine methods on data mining. In journal of physics: conference series (vol. 1255, no. 1, p. 012067). Iop publishing.

Syafrizal, Melwin. 2015. "*Pengantar Jaringan Komputer*". Yogyakarta : Penerbit ANDI.

Tasril, v., wijaya, r. F., & widya, r. (2019). Aplikasi pintar belajar bimbingan dan konseling untuk siswa sma berbasis macromedia flash. Jurnal informasi komputer logika, 1(3).

Wongkar, Stefen, Dkk. 2015. *"Analisa Implementasi Jaringan Internet Dengan Menggabungkan Jaringan LAN Dan WLAN Di Desa Kawangkoan Bawah Wilayah Amurang II"*. Jurnal Mahasiswa Program Studi Teknik Informatika Universitas Sam Ratulangi.

Yuza, Reswan, Dkk. 2018. *"Implementasi Kompilasi Algoritma Kriptografi Transposisi Columnar Dan Rsa Untuk Pengamanan Pesan Rahasia"*. Jurnal Mahasiswa Program Studi Teknik Informatika Universitas Muhammadiyah Bengkulu.