MIND Journal | ISSN (p): 2528-0015 | ISSN (e): 2528-0902 | Vol. 4 | No. 2 | Halaman 17 - 31
DOI : https://doi.org/10.26760/mindjournal Desember 2019

Sistem Otentikasi *Login* Dengan *Single Sign-On* Untuk Mengakses Banyak Sistem

THETA DINNARWATY PUTRI[1], WINARNO SUGENG[2], RESI KATRI[3]
[1,2,3]Program Studi Informatika Institut Teknologi Nasional Bandung
Email : theta@itenas.ac.id

ABSTRAK

SSO memungkinkan mengakses beberapa aplikasi dengan menggunakan satu akun melalui integrasi LDAP sebagai menajemen pengguna. Pada penerapan SSO ini digunakan layanan explicite FTPS sebagai layanan pertukaran dan penyimpanan data. Explicite FTPS merupakan pengembangan layanan FTP dengan fasilitas SSL. Protocol keamanan pada SSL pada saat pengiriman data bersifat privat antara klien dan server, sehingga pertukaran data menjadi aman karena sudah terenkripsi. Dalam sebuah jaringan, tolak ukur tingkat keamanan aplikasi yang saling terintegrasi tidak hanya dari otentikasi, tetapi juga otorisasi. Pada sistem ini menerapkan metode otentikasi dan otorisasi dengan fasilitas RADIUS sebagai protokol yang memberi izin akses setelah proses otentikasi dengan parameter akses diterima atau akses ditolak, sehingga tujuan akhir sistem ini akan menjadi efisien dan terlindungi.

***Kata kunci**: SSO, SSL, Explicite FTPS, RADIUS, LDAP*

ABSTRACT

SSO allows accessing multiple applications using a single account through the integration of LDAP as user management. In the implementation of this SSO used explicite FTPS as a service of exchange and data storage. Explicite FTPS is an FTP service development with SSL facility. The security protocol on SSL when sending data is private to clients and servers, so the exchange of data to be safe because it is encrypted. In a network, the benchmark application security level is integrated not only from authentication, but also authorization. This system applies authentication and authorization methods with RADIUS facility as that grants access permissions after authentication with access-accept or access-reject access, so the final goal this system will be efficient and protected.

***Keywords**: SSO, SSL, Explicite FTPS, RADIUS, LDAP*

1. PENDAHULUAN

Beberapa proses otentikasi pengguna umumnya menggunakan banyak akun untuk mengakses beberapa layanan aplikasi, yang tentunya akan mempersulit pengguna dalam mengingat akun untuk masing-masing aplikasi. Mekanisme SSO *(Single Sign-On)* dapat membantu pengguna dalam proses otentikasi (Aminuddin, 2014). Dengan menerapkan cara tersebut pengguna saat mengakses banyak aplikasi yang sudah terintegrasi cukup hanya satu kali *login*. Teknik tersebut diperoleh dengan memanfaatkan LDAP *(Lightweight Directory Access Protocol)* untuk memanajemen pengguna dan CAS *(Central Authentication Service)* sebagai pusat otentikasi. Penerapan dari SSO dapat diintegrasikan ke beberapa layanan jaringan seperti *web service, mail service*, dan *explicite* FTPS (*File Transfer Protocol explicite secure*) *service*.

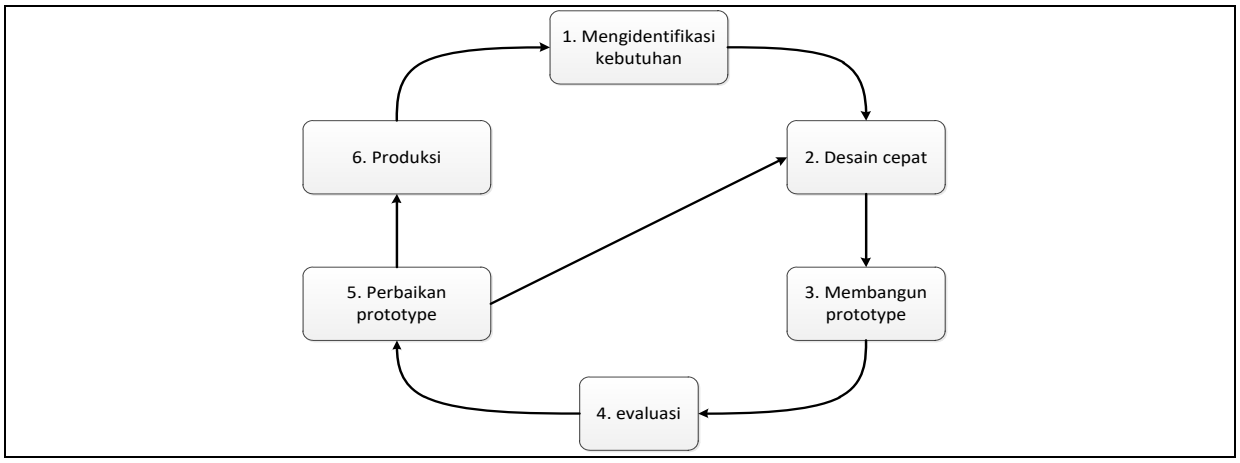
Pada dasarnya *explicite* FTPS merupakan layanan *file sever* yang digunakan dalam penyimpanan arsip atau pertukaran data. Kondisi normal proses pengiriman data dari klien ke server belum terenkripsi, sehingga diperlukan keamanan dari paket data tersebut untuk menunjang dalam proses perizinan akses (Rusmana, 2016). Proses enkripsi data dengan Protokol SSL *(Secure Socket Layer)* digunakan dalam hal proteksi paket data (Amiruddin, 2018). Kebutuhan autentikasi, otorisasi dan pendaftaran akun pengguna secara terpusat saat mengakses jaringan digunakan RADIUS *(Remote Authentication Dial-In User Service)* (Prajna, 2014) sehingga mampu membantu menyederhanakan proses *login*.

Protokol keamanan SSL saat pengiriman data antara klien dan server bersifat privat**.** Sedangkan RADIUS merupakan protokol yang memberi izin akses setelah proses otentikasi dengan parameter akses diterima (*access-accept*) atau akses ditolak (*access-reject*) (Agustian, 2013). Dengan menerapkan SSL dan RADIUS pada FTP (*File Transfer Protocol*) sistem akan lebih terproteksi dari usaha pencurian data dari pihak ketiga (Fernando, 2010). Dalam proses pembuktian keamanan yang menggunakan protocol SSL digunakan proses pengujian *sniffing* dengan cara menganalisa paket datanya (D.C Pramudita, 2014). Pada pengujian *sniffing* dilakukan analisa paket data dengan cara penyadapan pada jaringan yang diuji dalam memperoleh data atau informasi saat proses pertukaran data dari klien ke server (Juliharta, 2015) dari pengujian ini akan dibuktikan keberhasilan dari enkripsi data *login*.

2. METODE PENELITIAN

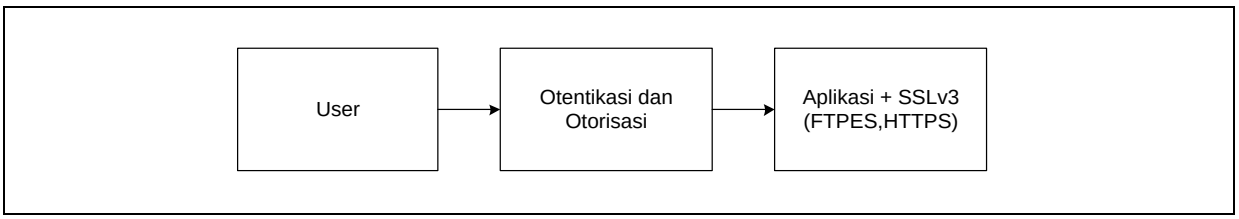
Dalam pengembangan perangkat lunak (*software development process*) dalam penelitian ini digunakan model prototipe seperti terlihat pada Gambar 1 sebagai penerapan struktur pada pengembangan perangkat lunak (*software*) dan sebagai panduan untuk menyukseskan proyek pengembangan sistem melalui tahapan-tahapan dari pengembangan perangkat lunak menggunakan model prototipe.

Pengembangan perangkat lunak menggunakan model prototipe dengan dasar satu metode siklus hidup sistem yang didasarkan pada konsep model bekerja, bertujuan pengembangan model menjadi sistem final. Artinya sistem akan dikembangkan lebih cepat dari pada metode tradisional dan biayanya menjadi lebih rendah. Sebagai pengenalan dari metode tersebut adalah baik dari sisi pengembang dan dari sisi pelanggan mampu melihat sekaligus melakukan pengerjaan dimulai dari bagian awal hingga akhir proses pengembangan.



Gambar 1. Model Prototipe
(sumber:careerride.com, 2018)

Setelah melalui tahap awal yaitu melakukan identifikasi berkaitan dengan perangkat lunak dan semua kebutuhan sistem yang akan dibuat, tahap dilanjutkan ke tahap desain cepat atau rancangan dasar. Rancangan dasar dapat dilihat melalui Gambar 2 dari blok diagram umum. Tahapan penelitian dilanjutkan sesuai proses mengikuti metode prototipe, hanya proses produksi tidak dilakukan dikarenakan diperlukan pengujian lanjutan lebih dalam dan kompleks, dalam hala ini akan dilakukan penelitian lanjutkan tingkat hibah penelitian.



Gambar 2. Blok Diagram Umum

Sebagaimana yang telah disampiakan sebelumnya diperlukan proses otentikasi dan otorisasi dari layanan akun yang menggunakan modul protokol keamanan untuk masuk dalam banyak sistem secara sekaligus melalui satu pintu. Penerapan sebuah layanan dengan atau tanpa protokol keamanan SSL memungkinkan pemusatan otentikasi dan otorisasi dalam sebuah jaringan SSO yang terintegrasi pada RADIUS dan LDAP, sebagai akun tunggal untuk mengakses beberapa aplikasi.

Berdasarkan permasalahan otentikasi, otorisasi dan pengamanan paket data dari proses *sniffing* yang bertugas sebagai penganalisa paket data pada pertukaran dan penyimpanan data. Protokol SSL yang digunakan untuk proses pertukaran dan penyimpanan data menjadi terproteksi pada *explicite* FTPS, RADIUS, LDAP, dengan otentikasi dan otorisasi terpusat pada sistem SSO yang dibangun. Tahapan proses yang dilakukan adalah dengan menerapkan konfigurasi modul pam_RADIUS pada integrasi RADIUS (*freeradius*) ke LDAP (*ldapadmin*) sebagai otentikasi dan otorisasi layanan *explicite* FTPS. Selanjutnya menerapkan konfigurasi modul SSL pada layanan pihak ketiga, yaitu aplikasi proFTPd dengan merubah parameter layanan menjadi *explicite secure* seperti FTPES dan HTTPS. Dengan cara ini memungkinkan peningkatan proses pengamanan paket data dari usaha proses *sniffing* pihak penyusup. Tahapan kemudian adalah memfasilitasi layanan otentikasi akun tunggal terpusat SSO dengan memanfaatkan integrasi CAS dan LDAP. Dari penerapan metode otentikasi dan otorisasi, menjadikan sistem menjadi terproteksi dan menajemen akun pengguna menjadi sederhana.

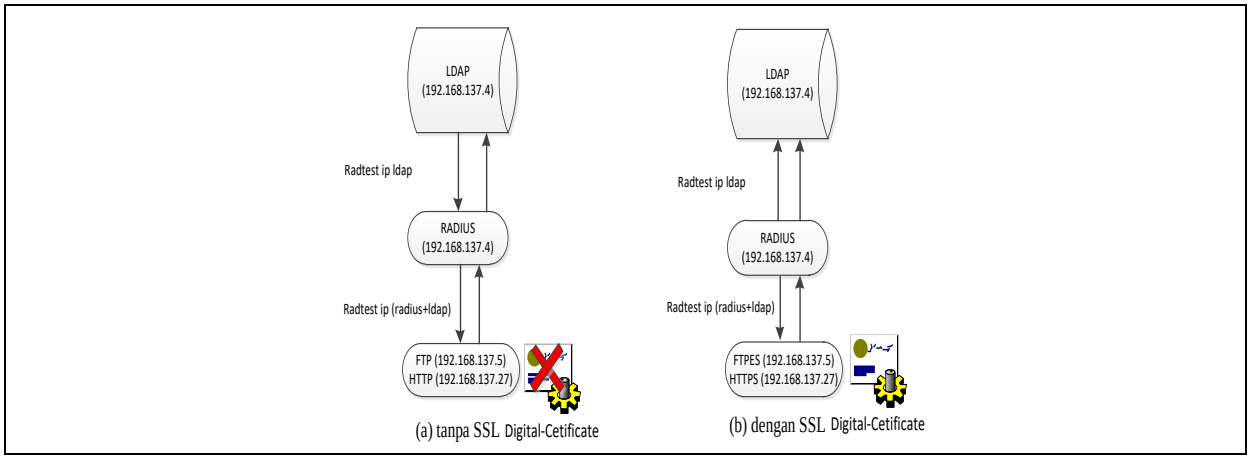
3. PERANCANGAN SISTEM

3.1. Mekanisme sistem tanpa dan dengan *Secure Socket Layer* pada pemusatan otentikasi FTPES

Pada penelitian ini aplikasi yang digunakan adalah FTP server dengan tolak ukur perbandingan sistem tanpa dan dengan menggunakan SSL. Pada Gambar 3 diilustrasikan bahwa FTP server yang merupakan aplikasi utama dalam penelitian ini merupakan aplikasi yang terdiri dari dua jenis otentikasi yaitu dengan *Anonymous Authentication* dan *User Authentication*. *Anonymous authentication* merupakan proses mengenali identitas pengguna secara acak atau tanpa menggunakan pengenal seperti akun dari pengguna, sedangkan *User Authentication* dapat dilakukan dengan menggunakan otentikasi langsung dari *root server* yang dibangun pada sistem operasi itu sendiri atau dengan menggunakan modul PAM (*Pluggable Authentication Module*) seperti mod_RADIUS dan mod_LDAP, sehingga penerapan otentikasi dan otorisasi terpusat pada LDAP dan RADIUS dapat dilakukan. Otentikasi dan otorisasi merupakan bagian penting dari aspek keamanan suatu sistem, tetapi layanan yang terhubung pada aspek keamanan tersebut juga harus diproteksi.

Dalam penelitian ini dilakukan proteksi untuk mencegah pengambilan paket data secara ilegal, dengan menggunakan mod_SSL/TLS, yang merupakan modul dari *Socket Secure Layer*, sehingga pada otentikasi layanan FTP tanpa SSL seperti diilustrasikan pada Gambar 3 (a) tidak akan melakukan proses *digital certicate request* pada *openssl* sebagai penyedia protokol keamanan, sedangkan pada otentikasi layanan FTP dengan SSL seperti diilustrasikan pada Gambar 3 (b) akan melakukan proses pembuatan *digital certicate* yang akan disimpan dalam format *cert* pada server dimana openssl digunakan.

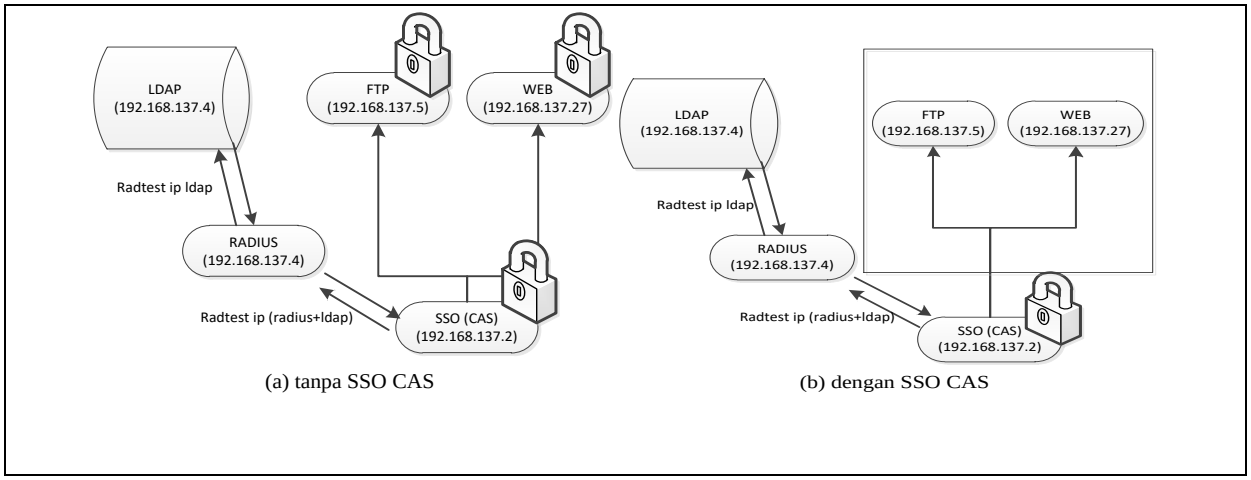
Hasil dari Gambar 3 (b), merupakan komunikasi FTPES, RADIUS, dan LDAP akan dihubungkan dengan *web service* menggunakan *Wordpress*, dan proses ini disebut pengembangan tahap satu, yang selanjutnya akan dihubungkan pada metode *Central Authentication Service* yang disebut *Single Sign On* sebagai proses pengembangan tahap kedua yang akan dijelaskan selanjutnya.



Gambar 3. Arsitektur dengan atau tanpa Sistem SSL

3.2. Mekanisme sistem tanda dan dengan Single Sign On

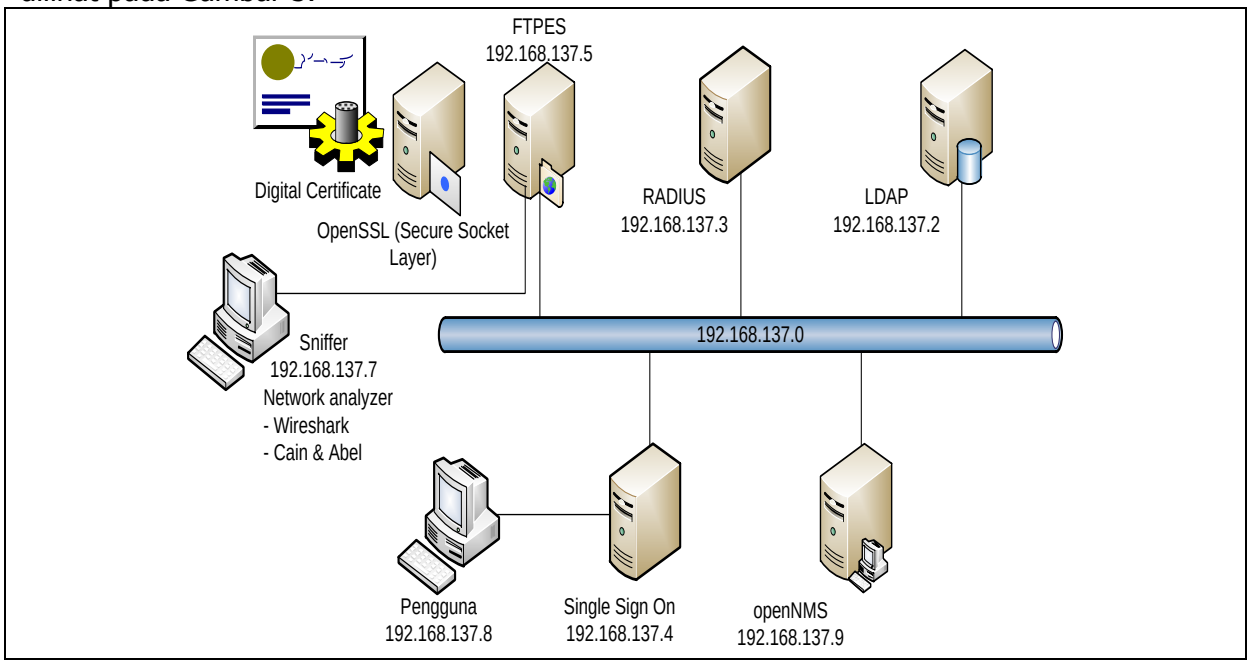
Pada penelitian ini sistem yang belum terintegrasi *Single Sing On* akan melakukan proses otentikasi pada semua portal dari masing-masing aplikasi. Dengan tolak ukur proses mengenali identitas pengguna terhadap aplikasi tujuan akan dilakukan berdasarkan berapa banyak aplikasi yang saling terhubung, secara garis besar digambarkan pada Gambar 4 (a), dengan menggunakan tiga aplikasi yaitu portal *Central Authentication Service*, FTP dan *Web service* (*Wordpress*) maka pada saat pengguna ingin mengakses, proses otentikasi akan dilakukan sebanyak tiga kali, sedangkan sistem yang telah terintegrasi *Single Sign On* akan melakukan otentikasi dan mengenali pengguna yang ingin mengakses tiga aplikasi dengan menggunakan satu akun seperti diilustrasikan pada Gambar 4 (b).



Gambar 4 Arsitektur tanda dan dengan Sistem SSO

3.3. Topologi Pengujian

Topologi pengujian untuk implementasi SSO pada Integrasi *Explicite* FTPS, RADIUS, dan LDAP untuk mencegah ancaman *sniffing* pada proses pertukaran dan penyimpanan data dapat dilihat pada Gambar 5.

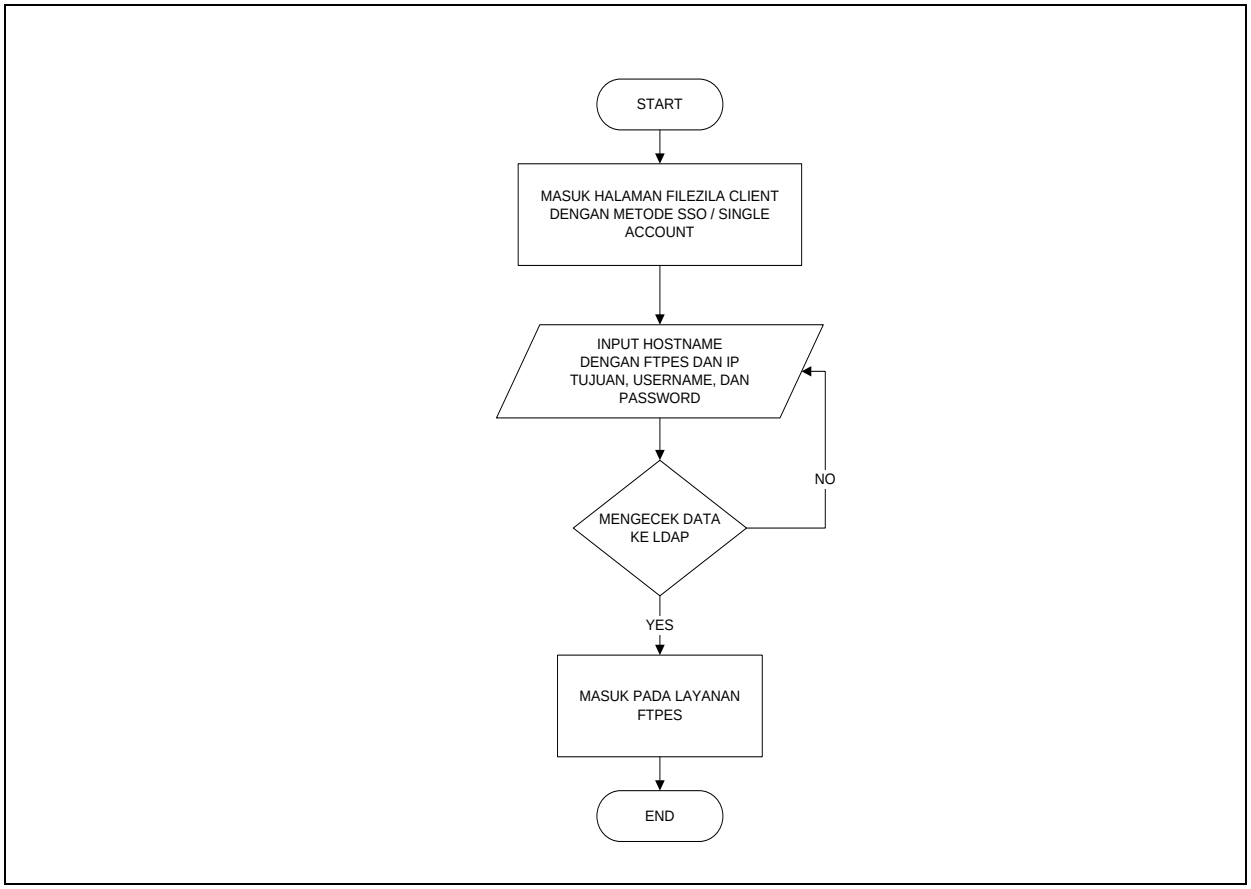


Gambar 5. Struktur dan Topologi

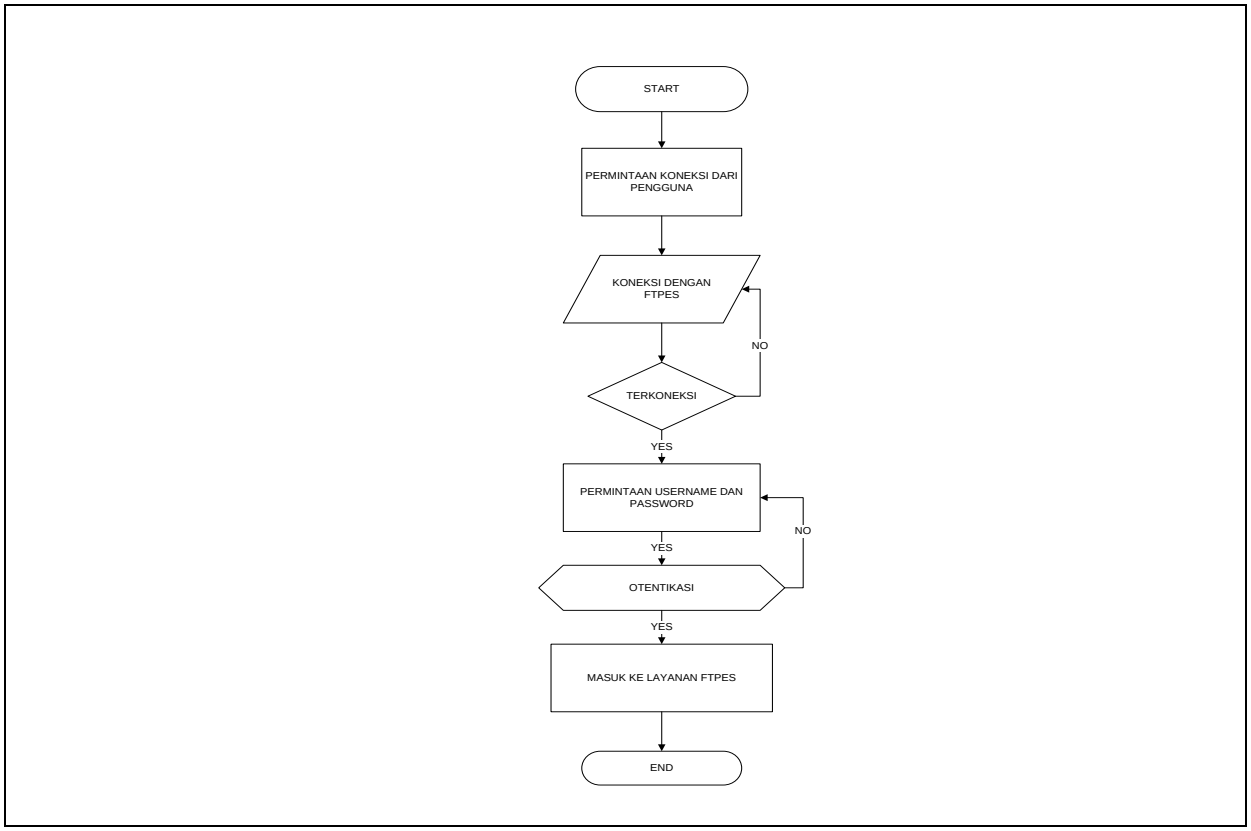
Pada Gambar 5 merupakan struktur topologi dimana sistem dimulai oleh pengaksesan data oleh klien dari satu komputer. Kemudian sistem aplikasi dihubungkan dengan server LDAP dan diintegrasikan dengan *freeradius* sebagai autentikasi. Sehingga jika klien ingin mengakases FTP server dengan autentikasi *freeradius* maka akan dihubungkan pada LDAP yang bertugas untuk manajemen pengguna, dimana pengguna dan *password* yang dimasukan oleh klien akan dibaca oleh server LDAP apakah sudah terdaftar di server LDAP, juga pada *mail* server untuk otentikasi dan manajemen pengguna akan dikelola oleh *freeradius* dan LDAP. Tugas untuk server adalah memonitor layanan server seperti LDAP, *mail* server dan *ftp* server layanan : *up* atau *down*, menampilkan trafik, dan memberikan informasi sistem yang berjalan.

Pada Gambar 6 merupakan diagram alir pada proses otentikasi SSO pada layanan *explicite* FTPS. Metode otentikasi secara terpusat pada direktori LDAP adalah melakukan pengecekan kecocokan akun tersebut yang akan menjadi parameter dari otentikasi izin akses dari RADIUS server.

Pada Gambar 7 menjelaskan komunikasi *explicite* FTPS dengan RADIUS dimulai dari permintaan koneksi dari pengguna. Jika *explicite* FTPS terkoneksi dengan RADIUS maka dilakukan proses otentikasi dilakukan, jika berhasil hak ases diberikan jika tidak proses akan dikembalikan ke proses permintaan akun.

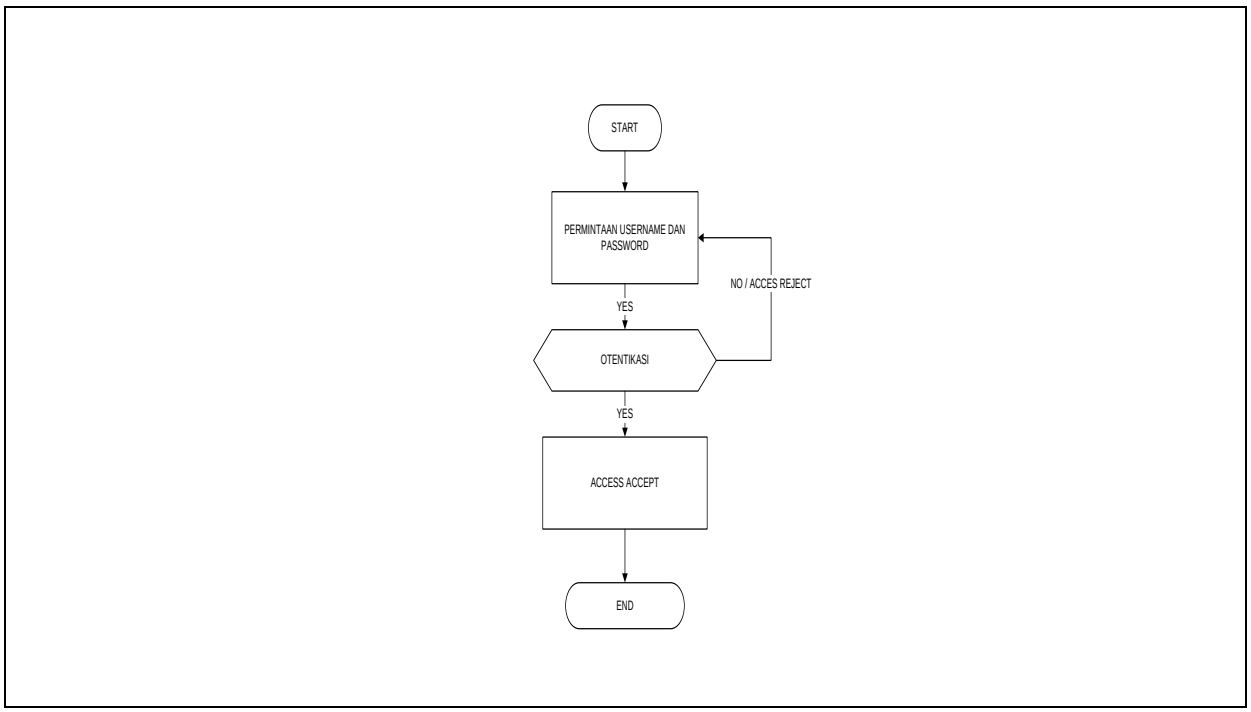


Gambar 6. Proses SSO terhadap layanan *explicite* FTPS



Gambar 7. Komunikasi *Explicite* FTPS dengan RADIUS *server*

Pada Gambar 8 menjelaskan diagram alir dari komunikasi RADIUS ke LDAP dengan parameter *access reject* atau *access accept*. Jika proses otentikasi dari metode SSO telah dilakukan maka akun tidak langsung digunakan untuk mengakses, melainkan diperiksa oleh server RADIUS apakah akun yang digunakan sesuai dengan akun yang terdaftar pada direktori LDAP.



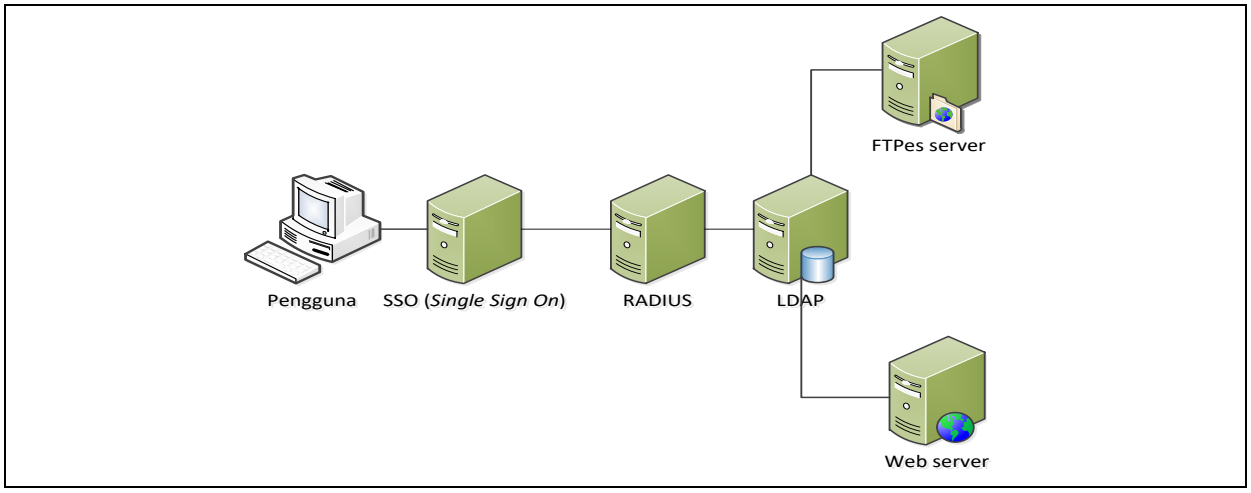
Gambar 8. Komunikasi RADIUS dan LDAP

3.4. Mekanisme SSO

Cara melakukan *login* pada sistem SSO pada sistem yang banyak aplikasi yang sudah terintegrasi dapat dilihat pada Gambar 9. Sistem terdiri dari CAS dan LDAP sebagai *server* serta aplikasi *explicite* FTPS. CAS merupakan tempat ditanamkannya sistem SSO. CAS diintegrasikan dengan server LDAP yang merupakan tempat diletakkan basis data pengguna dari aplikasi berbasis *explicite FTPS*.

Proses kerja sistem SSO yang dibangun sebagai berikut:

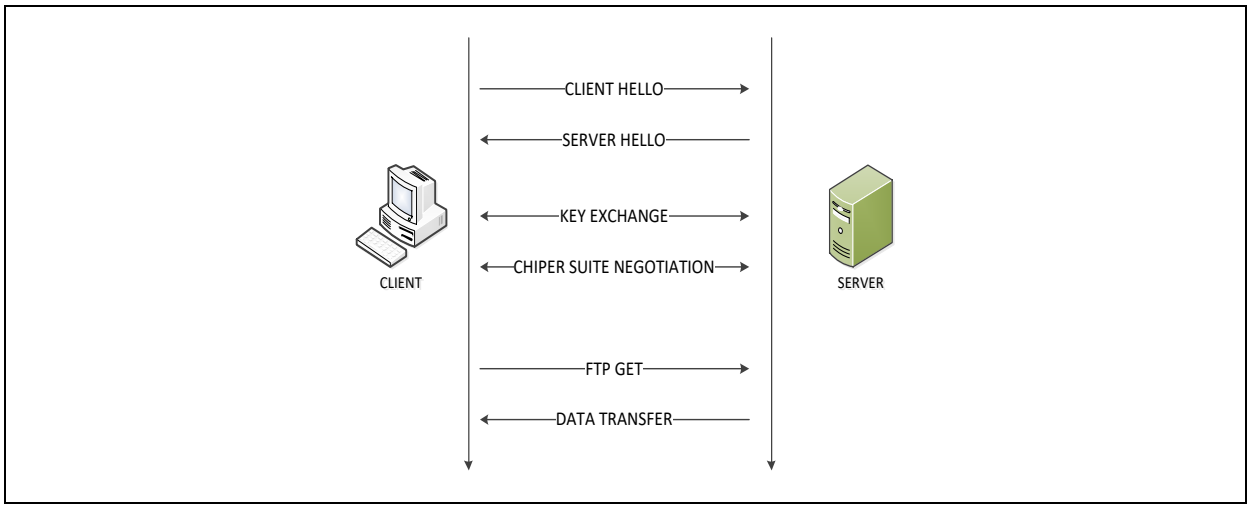
1. Aplikasi berbasis *Explicite FTPS* diakses oleh pengguna
2. Pada saat berhasil *login*, pengguna di*redirect browser* menuju SSO*Server*.
3. Pengguna harus memasukkan *username* dan *password* untuk proses otentikasi.
4. Sistem mencocokan *username* dan *password* pengguna pada LDAP.
5. Setelah dilakukan pengecekan informasi, data pengguna dikirim kembali menuju server SSO.
6. Pengguna yang berhasil melewati proses otentikasi, diarahkan kembali ke aplikasi berbasis *explicite FTPS* dengan sebuah tiket.
7. Pengguna yang sudah sah dapat mengakses informasi yang terdapat pada aplikasi tersebut.



Gambar 9. Mekanisme SSO

3.5. Mekanisme SSL

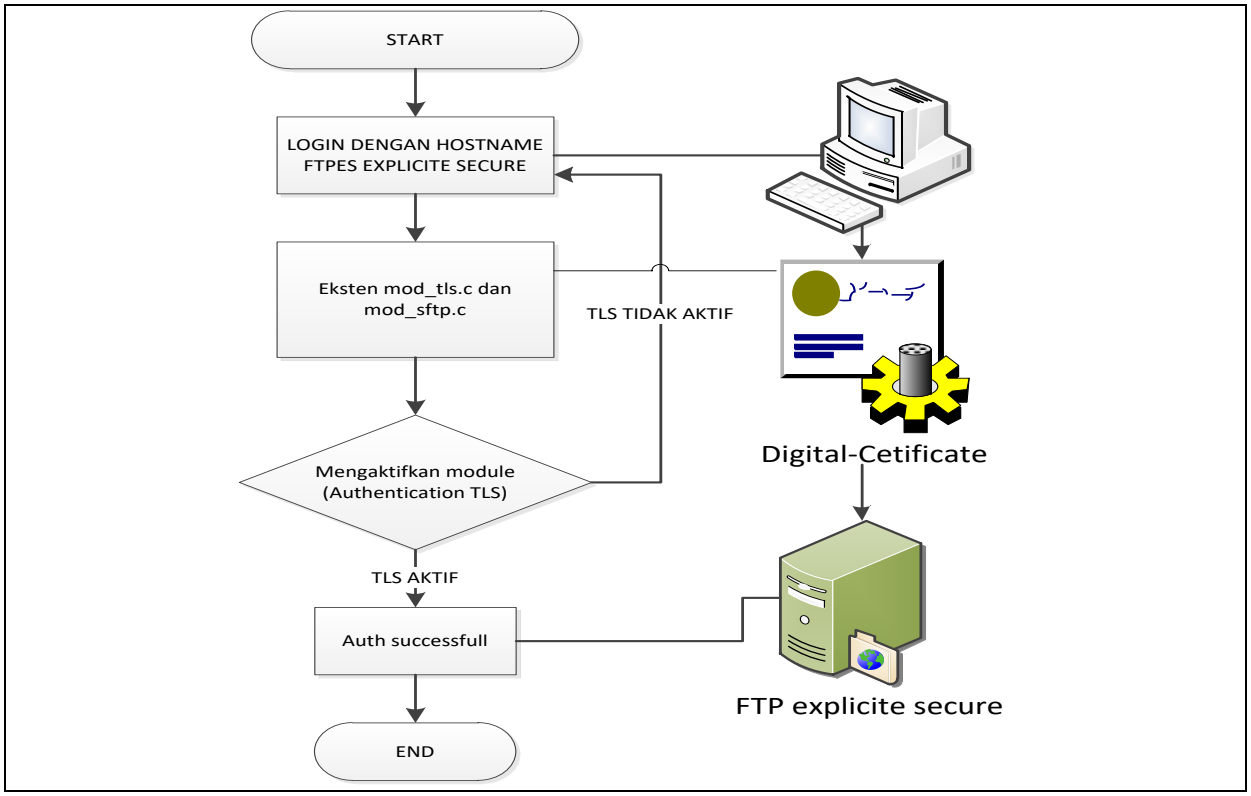
Pada Gambar 10 menjelaskan teknologi SSL menggunakan konsep yang mirip dengan proses enkripsi pada kriptografi *public*, agar sistem komunikasi antara pengguna dan server menjadi aman. Pengguna dan server berinteraksi mengirimkan data dalam hal ini data disamarkan supaya tidak mudah terbaca dan untuk membacanya digunakan sandi dan kunci yang hanya dimiliki kedua pihak yang berkomunikasi tersebut, sehingga pihak lain yang mencoba menyadap data yang dikirim tersebut tidak akan bisa membacanya karena sandi dan kunci yang dibutuhkan tersebut hanya terdapat pada pihak yang melakukan komunikasi yaitu pengguna dan server.



Gambar 10. Mekanisme SSL

3.6. Mekanisme *Explicite* FTPS

Pada Gambar 11 menjelaskan proses penerapan modul TLS sebagai dasar dari pembuatan sertifikasi digital, dengan memodifikasi layanan menjadi terfasilitasi oleh protokol kemanan dengan parameter *Explicite secure* dengan mengacu pada port 21/990. Dalam pengujian dengan aplikasi penganalisa jaringan packet data tidak dapat disadap dan akun pengguna dapat pengamanan saat proses otentikasi dilakukan. Parameter keberhasilan dari proses ini adalah jika *module* TLS dan SSL dapat diaktifkan maka *authentication successful*, jika *module* TLS dan SSL tidak dapat diaktifkan maka *digital certificate* dan *authentication failed.*

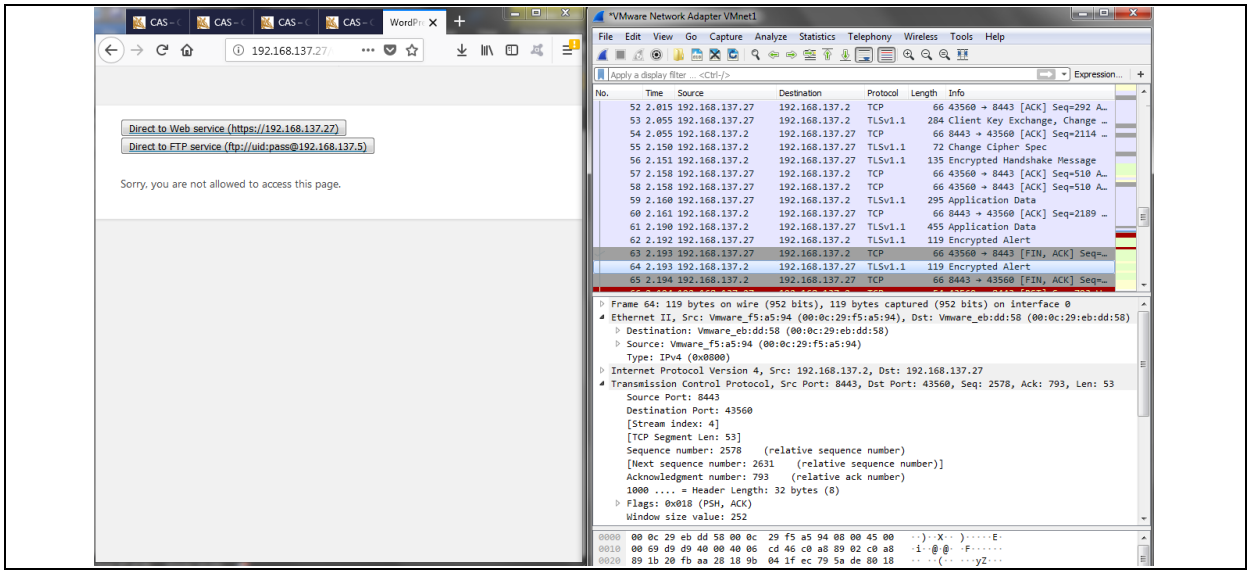


Gambar 11. Mekanisme *Explicite* FTPS

4. IMPLEMENTASI DAN PENGUJIAN

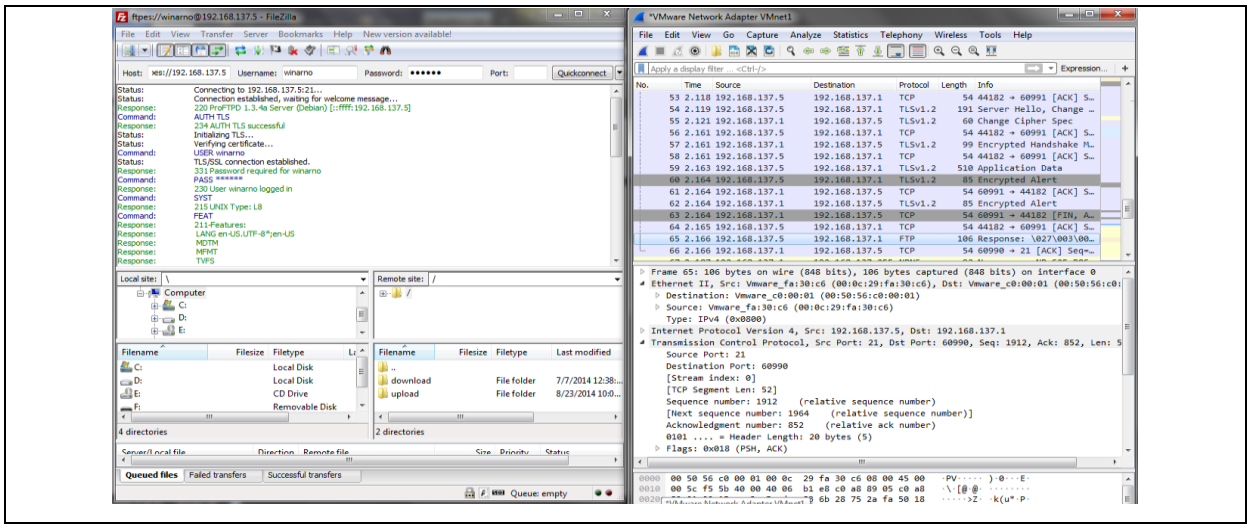
4.1. Pengujian *response time*

Pengujian *response time* dilakukan untuk memperoleh nilai waktu tanggap suatu sistem yang sedang terintegrasi ataupun tidak terintegrasi, dengan parameter pengujian dilakukan mulai saat proses otentikasi sampai informasi pada layanan dapat diakses.



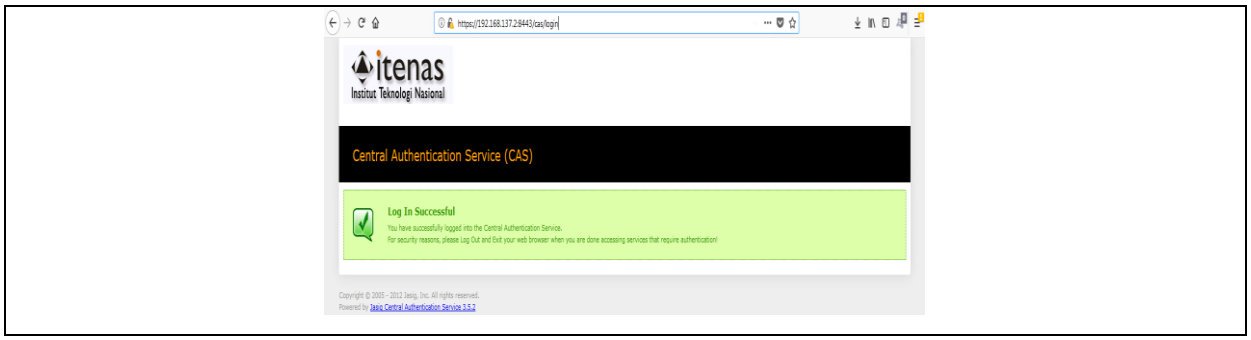
Gambar 12. Pengujian *response time* keseluruhan

Gambar 12. Pengujian analisis *response time* pada sistem keseluruhan mengambil parameter *time* dengan cara yang sama pada HTTPS karena SSO CAS menggunakan akses protokol yang sama dengan HTTPS.



Gambar 13. Pengujian *response time* SSL pada FTP (FTPS)

Gambar 13. Pengujian analisis *response time* pada layanan dengan SSL pada FTP mengambil parameter *time* pada *capture protocol* TLS dari awal melakukan proses sampai pada batas akhir *protocol* TLS yang direkam.

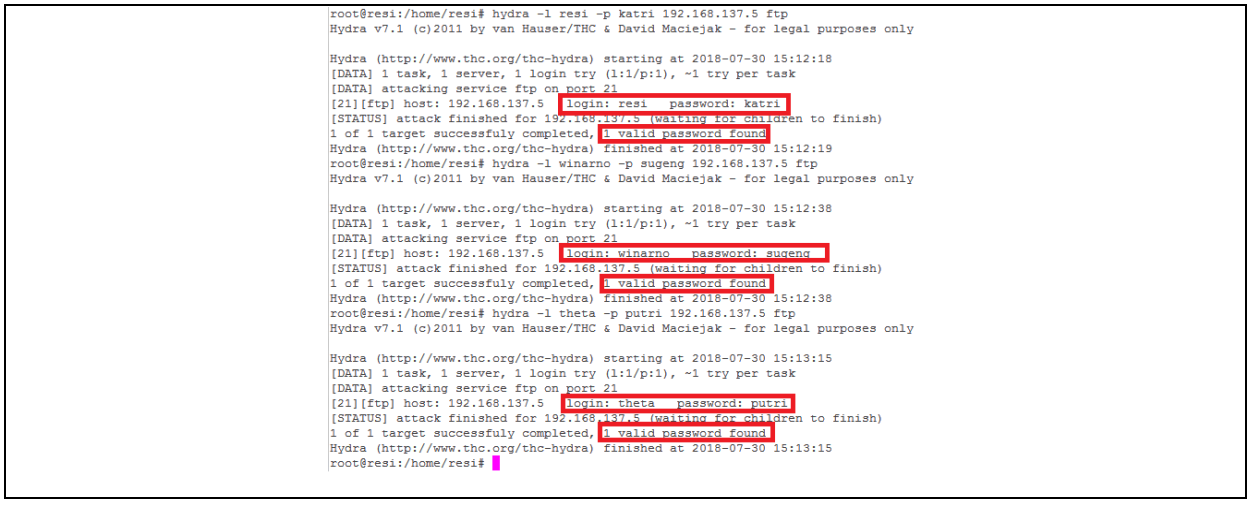


Gambar 14. Pengujian *response time* SSL pada HTTP (HTTPS)

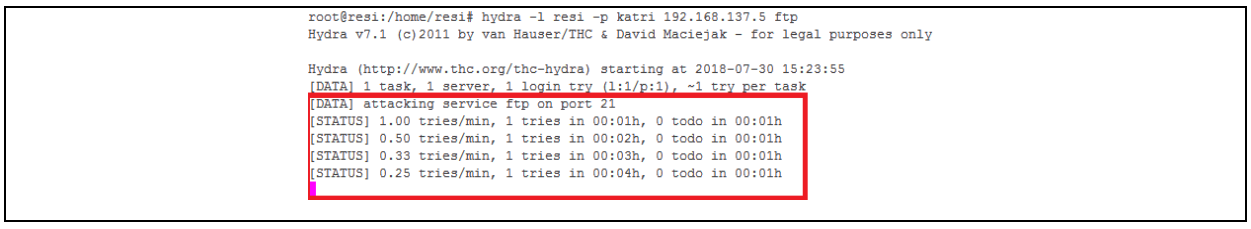
Gambar 14. Pengujian analisis *response time* pada layanan dengan SSL pada HTTP mengambil parameter *time* dengan cara yang sama pada FTPES. Pengujian dilakukan 10 kali dengan browser, di setiap server. Waktu tanggap rata-rata adalah dihitung dengan menambahkan waktu respons keseluruhan dibagi dengan jumlah percobaan dilakukan.

4.2. Pengujian tanpa dan dengan SSL

Pengujian dengan metode *sniffing bruteforce* menggunakan *hydra* dapat dilihat pada Gambar 15, pengujian *sniffing* paket data yang dilakukan pada CLI linux dengan menggunakan *hydra*, dimana pada hasil pengujian diperoleh informasi *uid* dan *password* untuk layanan tanpa SSL. Sedangkan pada Gambar 16 merupakan pengujian yang dilakukan dengan menggunakan *hydra* pada kondisi layanan mengaktifkan modul SSL, dengan hasil status pengujian pada CLI linux dimana aplikasi *hydra* terpasang, tidak diperoleh informasi *uid* dan *password* yang digunakan oleh pengguna.



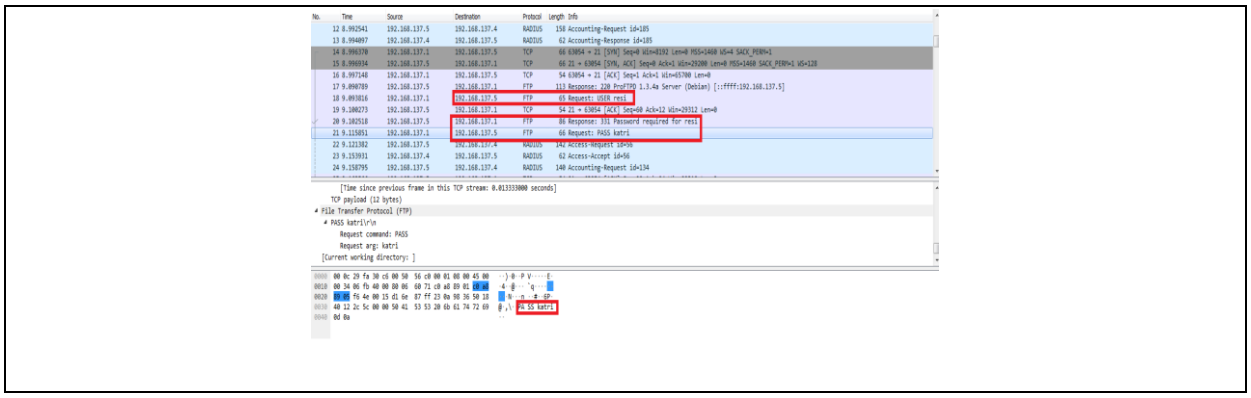
Gambar 15. pengujian *hydra* layanan tanpa SSL



Gambar 16. pengujian *hydra* layanan dengan SSL

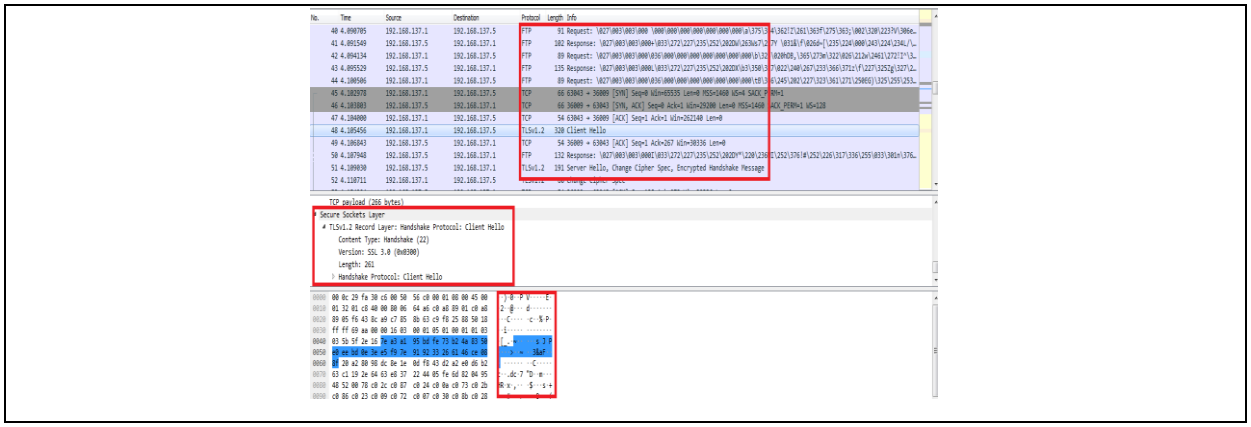
4.3. Proses *sniffing* menggunakan *wireshark*

Pengujian FTP tanpa protokol SSL Gambar 17 dilakukan dengan cara melakukan pengujian proses *sniffing* paket data pada layanan FTP tanpa SSL untuk mengetahui apakah *username* dan *password* terindentifikasi sebagai pengguna yang sedang mengakses suatu layanan pada saat pengujian proses *capturing* pada jaringan yang terhubung dengan pengguna *wireshark* disisi klien.



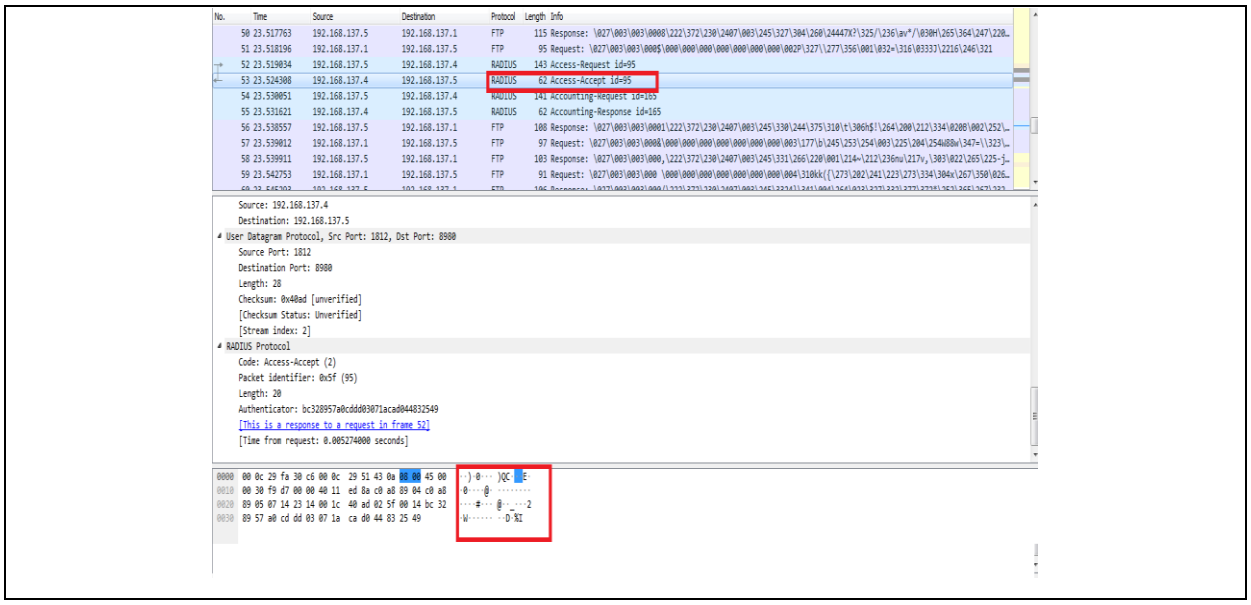
Gambar 17. Pengujian jaringan FTP tanpa SSL

Pengujian FTP dengan protokol SSL (FTPES) Gambar 18 dilakukan dengan cara pengujian FTP dengan protokol SSL untuk megidentifikasi *uid* dan *password* sentralisasi ke basis data phpLDAPadmin, dengan hasil analisa jaringan FTPES *username* dan *password* tidak dapat ditemukan pada saat *capture* semua protokol yang sedang berjalan.



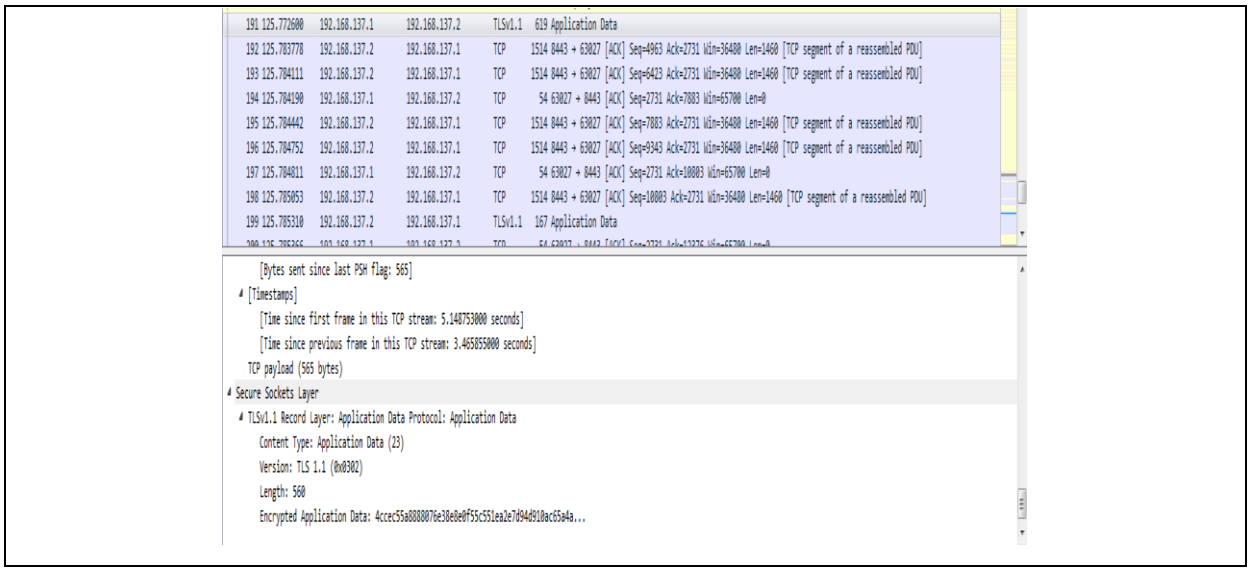
Gambar 18. mengalisa jaringan FTP dengan SSL

Pengujian RADIUS *request-access accept* pada layanan dengan protokol SSL Gambar 19 dilakukan pengujian pada proses perizinan access pada RADIUS, dimaksudkan menganalisa aktifitas jaringan pada pengembangan tahap satu dengan melakukan *capture* aktifitas jaringan RADIUS yang merupakan bagian dari integrasi sistem yang digunakan pada konsep SSO. Cara yang dilakukan dengan *radtest* (RADIUS testing) dengan meminta informasi *uid* dan *password* pada LDAP server. Hasil yang diperoleh dari *capturing* tidak terdapat informasi mengenai *uid* dan *password* dari RADIUS test ke LDAP server.



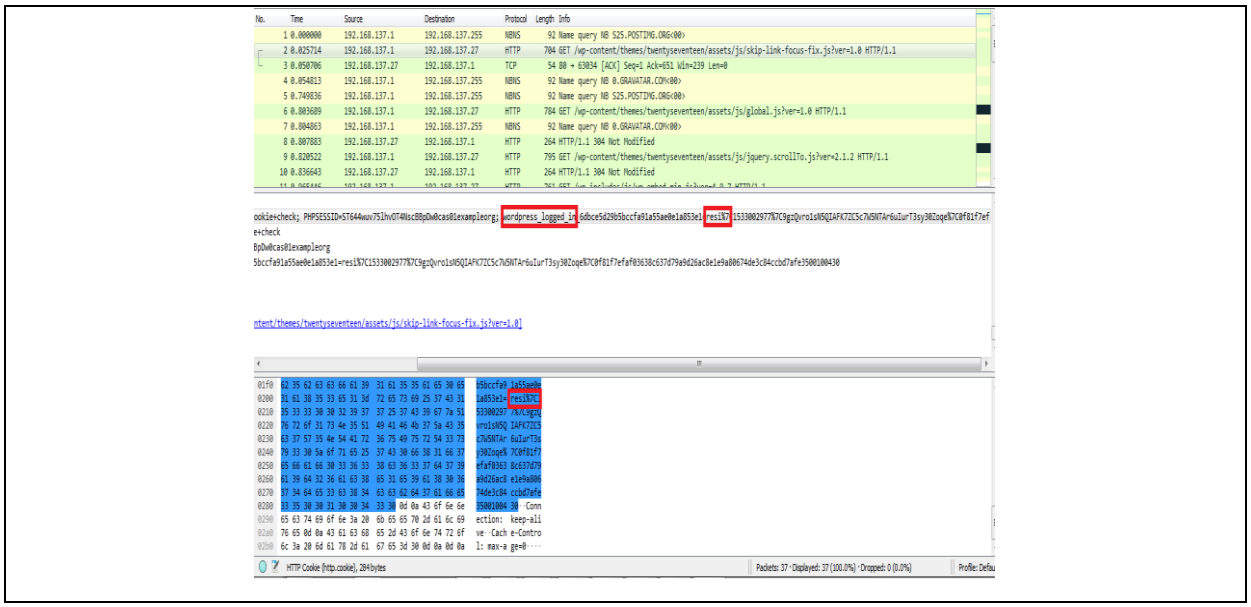
Gambar 19. Pengujian jaringan RADIUS

Pengujian protokol HTTPS Gambar 20 pengujian menganalisa jaringan pada protokol HTTPS, dimaksudkan untuk menganalisa aktifitas jaringan *Web service* dengan protokol SSL. Dari hasil pengujian *capturing* tidak dapat diperoleh informasi *uid* dan *password* pada aktifitas jaringan pengembangan tahap satu untuk layanan *web*.



Gambar 20. Pengujian jaringan *Web* dengan protokol SSL (HTTPS)

Pengujian protokol HTTP Gambar 21 pengujian mengenalisa jaringan pada protokol HTTP, dimaksudkan untuk menganalisa aktifitas jaringan *Web service* tanpa protokol SSL. Dari hasil pengujian *capturing* dapat diperoleh informasi *uid* dan *password* pada aktifitas jaringan pengembangan tahap satu untuk layanan *web*.



Gambar 21. Pengujian jaringan *Web* tanpa protokol SSL (HTTP)

5. KESIMPULAN

Sistem otentikasi *login* untuk mengakses banyak sistem dengan menerapkan konsep *Single Sign-On* metode *Central Authentication Service* pada dua layanan yaitu FTPES dan *Web service* terintegrasi RADIUS, dan LDAP, disimpulkan bahwa dengan metode tersebut proses menajemen dan otentikasi pengguna menjadi lebih efisien dan proses pertukaran data pada sebuah layanan data *storage* akan menjadi terproteksi.

Kesimpulan dari pengujian sistem keseluruhan berdasarkan konsep SSO CAS, jika konsep ini diterapkan pada dua layanan jaringan seperti FTP dan Web secara bersamaan maka waktu tanggap (*response time*) pada proses otentikasi akan lebih lambat, tetapi dari segi penggunaan akun, aktifitas otentikasi dan otorisasi akan menjadi efisien dibandingkan tanpa menerapkan konsep SSO CAS, karena pada proses menganalisa dua aplikasi layanan jaringan akan melakukan proses capture dua kali, tergantung berapa jumlah aplikasi yang digunakan, sementara beberapa aplikasi dengan konsep SSO CAS proses *capture* aktifitas jaringan pada saat otentikasi dilakukan ke semua server yang terhubung diantaranya CAS, LDAP, RADIUS, FTP, Web, dan SSL.

DAFTAR RUJUKAN

Agustian, A. F. (2013). Implementasi server RADIUS pada Cloud Computing berbasis GNU/Linux.

Aminuddin. (2014). Implementasi Single Sign On untuk Mendukung Interaktivitas Aplikasi E-Commerce Menggunakan Protocol Oauth.

Amiruddin, A. (2018). Implementasi Security pada Load Balancing Layanan Web Multidomain dengan SSL.

D.C Pramudita, P. ,. (2014). Otentikasi dan Manajemen Pengguna Hotspot Router Mikrotik Menggunakan RADIUS dan PHP-MySQL.

Fernando, H. (2010). *Studi dan Implementasi Sistem Keamanan Berbasis Web dengan Protokol SSL di Server Students Informatika ITB.* Bandung: Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung.

Juliharta, I. G. (2015). Bussiness Impact Analysis Aplikasi Jaringan Komputer dengan Teknik Packet Sniffing.

Prajna, I. K. (2014). *Implementasi LDAP Server Integrasi Radius Auntetikasi Service dan Layanan OpenNMS sebagai Monitoring Server*. Diambil kembali dari http://repository.taas.telkomuniversity.ac.id/index.php/Proyek-Akhir-Mahasiswa/TK/Implementasi LDAP Server Integrasi RADIUS Autentikasi Service Dan Layanan OpenNMS Sebagai Monitoring Server

Rusmana, M. U. (2016). Sistem Monitoring Jaringan Menggunakan OpenNMS Berbasis Smartphone Android.