IMPLEMENTASI DAN ANALISIS SINGLE SIGN ON PADA SISTEM INFORMASI UNIVERSITAS UDAYANA

I Putu Agus Eka Darma Udayana[1)], Lie Jasa[2)]

[1)]Manajemen Sistem Informasi Dan Komputer Universitas Udayana
[2)]Teknik Elektro Universitas Udayana
Jl. PB. Sudirman, Denpasar, Bali 80232
Email : agus.ekadarma@gmail.com[1)] , liejasa@unud.ac.id[2)]

Abstrak

Universitas Udayana merupakan salah satu perguruan tinggi yang memanfaatkan luasnya pemakaian internet tersebut sebagai sarana berkomunikasi dan saling bertukar informasi untuk kepentingan akademis. Belakangan ini Lightweight Directory Access Protocol (LDAP) merupakan metode otentikasi yang sangat sering digunakan. Dengan menggunakan LDAP pengguna hanya akan memiliki sebuah username dan password untuk dapat login pada banyak aplikasi berbasis web. Tapi jika ingin melakukan otentikasi pada masing-masing aplikasi pengguna harus berulang kali mengisi username dan password. Untuk mempermudah pengguna dalam proses otentikasi tersebut maka dikembangkan mekanisme Single Sign On (SSO). Dengan menggunakan SSO, user hanya perlu melakukan satu kali login dalam mengakses semua aplikasi web yang terintegrasi. Untuk mengimplementasikan sistem SSO digunakan Central Authentication Service (CAS) sebagai pusat otentikasi dengan memanfaatkan struktur LDAP untuk manajemen user. Dalam penelitian ini ditunjukkan bahwa, sistem Single Sign On yang diintegrasikan dengan Simak, E-Learning dan Blog System berhasil diimplementasikan. SSO ini dapat diintegrasikan dengan aplikasi yang menggunakan basis data maupun aplikasi yang menggunakan LDAP. Dalam proses otentikasi yang dilakukan oleh user, sistem SSO dapat menangani proses otentikasi sebanyak 200 user secara bersamaan tanpa error dengan average response time sebesar 1.480,73 ms dan sistem SSO dapat menangani proses otentikasi sebanyak 400 user secara bersamaan dengan error sebesar 0,25% dengan average response time sebesar 2.909,10 ms.

Kata kunci:*Single Sign On, CAS, LDAP*

1. Pendahuluan

Dengan semakin luasnya pemakaian internet, pengguna biasanya mengakses beberapa layanan setiap harinya dan karena hal tersebut mereka harus memiliki banyak *username* dan *password* [1]. Dalam hal ini Universitas Udayana merupakan salah satu perguruan tinggi yang memiliki banyak layanan berbasis *web* sebagai sarana memudahkan proses berkomunikasi dan saling bertukaran informasi untuk kepentingan akademis. Layanan aplikasi berbasis *web* yang dimiliki Universitas Udayana diantaranya Simak, *E-Learning* serta layanan berbasis *Blog System.* Dengan pertumbuhan jumlah layanan yang semakin banyak, tentunya sangat tidak efisien jika setiap masuk pada sistem administrasi suatu layanan, pengguna harus melakukan proses *login* atau otentifikasi menggunakan bebrapa kombinasi *username* dan *password* yang mereka miliki. Salah satu metode otentikasi yang dapat digunakan adalah LDAP. LDAP merupakan singkatan dari *Lighweight Directory Access Protocol*, dimana LDAP itu sendiri digunakan untuk menyimpan dan mengambil informasi, yang mirip dengan relasional *database* [2]. Perbedaan yang utama antara LDAP dan *database* adalah LDAP menggunakan model pohon untuk mengatur informasi, hal ini membuat LDAP menyediakan layanan *query* yang lebih cepat daripada relasional *database*. Model ini membuat LDAP sangat mirip dengan hirarki yang sebenarnya pada suatu organisasi. Dengan menggunakan metode ini diharapkan dapat membantu dalam menyelesaikan permasalahan banyaknya *username* dan *password* serta memberikan kenyamanan yang lebih bagi para pengguna [3]. Akan tetapi pengguna tetap harus mengisikan *username* dan *password* pada saat akan melakukan otentikasi pada masing-masing layanan aplikasi. Metode LDAP ini juga memiliki kelemahan, dimana kelemahan tersebut adalah membutuhkan otentikasi berulang untuk setiap aplikasi ketika pengguna ingin masuk ke dalam sistem administrasi aplikasi berbasis *web* tersebut. Dengan otentikasi berulang pengguna masih harus memberikan *username* pengguna dan *password* ke masing-masing layanan aplikasi. Proses *login* atau otentikasi yang banyak atau dengan kata lain sebanyak jumlah layanan aplikasi yang tersedia, secara tidak langsung menjenuhkan pengguna.

Untuk mengatasi kelemahan yang dimiliki oleh metode LDAP, maka dikembangkanlah suatu mekanisme otentikasi yang disebut *Single Sign On* (SSO). SSO itu sendiri merupakan sebuah metode dimana pengguna hanya sekali melakukan *login* atau otentikasi untuk dapat mengakses semua layanan yang dimiliki oleh pengguna. Karena pada dasarnya otentikasi *Single Sign On* (SSO) hanyan melibatkan *credential user* untuk *login* ke bayak layanan aplikasi berbasis *web* setelah sebelumnya melakukan otentikasi pada salah satu aplikasi yang terintegrasi dengan sistem *Single Sign On* (SSO). Untuk mengimplementasikan sistem SSO pada penelitian ini akan digunakan *Central Authentication Service* (CAS)

sebagai otentikasi terpusat pada Simak, *E-Learning* dan *Blog System* di lingkungan Universitas Udayana. Dengan menggunakan CAS, semua layanan aplikasi yang terintegrasi dalam sistem portal SSO tidak perlu melakukan proses otentikasi secara independen, sebagai gantinya CAS yang akan melakukan otentikasi pengguna pada setiap aplikasi untuk menghindari proses otentikasi secara berulang dan untuk menjamin keamanan informasi, proses transmisi data pada sistem tersebut menggunakan teknologi SSL protokol enkripsi [4]. Selain itu, pemilihan metode *Single Sign On* (SSO) berbasis CAS yang digunakan pada pebelitian ini didasarkan bahwa CAS merupakan metode SSO yang mendukung *library* dari *client* untuk layanan aplikasi berbasis *web*. Tidak hanya melakukan implementasi, pada penelitian ini juga akan dilakukan analisis *Sistem Single Sign On* berbasis CAS dengan melakukan pengujian *black box, white box, response time* dan *load test* dari sistem SSO berbasis CAS.

2. Penelitian Pendahulu

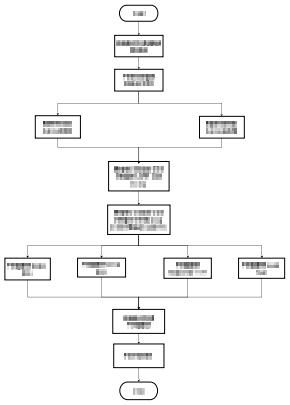
Dalam pengerjaan penelitian yang dilakukan oleh penulis terdapat beberapa penelitian pendahulu yang melatar belakangi penulis, sehingga penulis mengangkat penelitian mengenai SSO. Dimana penelitian terdahulu tersebut membahas bagaimana menerapkan SSO berbasis CAS pada jaringan LDAP [5]. Dimana hasil yang didapatkan dari penelitian tersebut adalah *server* SSO berhasil digunakan sebagai halaman *login* terpusat bagi layanan berbasis *web*. Keberhasilan ditentukan pada proses *login* dan *logout* salah satu aplikasi. Jika salah satu aplikasi *login* atau *logout* maka aplikasi lain akan *login* atau *logout*. Dalam pengembangannya disini akan dilakukan analisis kinerja sistem SSO berbasis CAS dalam menangani proses otentikasi *user*, sehingga akan diperoleh nilai dari kinerja SSO dalam menangani proses otentikasi *user*. Atau dapat juga dikatakan, penulis akan membahas mengenai QoS dari sistem SSO tersebut. Ketika berbicara suatu layanan pada lingkungan internet, *response time* merupakan sesuatu yang perlu diperhatikan untuk menentukan kualitas sistem [6].

Dalam penelitian lainnya diangkat bagaimana mengintegrasikan SSO dengan LDAP. Dimana dalam penelitian tersebut SSO diintegrasi hanya dengan menggunakan LDAP [3]. Hasil yang didapat dari penelitian tersebut adalah *user* terbantu dengan adanya SSO dapat mempermudah mereka karena tidak perlu menggunakan banyak *username* serta menghafal banyak *password*. Penggunaan SSO ini juga membantu dalam pengorganisasian *user* karena digunakannya LDAP sebagai *single* data *user*. Pada penelitian pendahulu ini sistem SSO hanya berbasis LDAP sehingga *user* harus tetap *login* berulang kali walaupun dengan *username* dan *password* yang sama. Pada penelitian yang akan dilakukan sistem SSO akan diterapkan berbasis CAS dengan tetap mengadopsi LDAP sebagai pusat manajemen *user*, sehingga *user* tidak perlu melakukan proses *login* berulang kali.

Pada penelitian pendahulu yang ketiga diangkat mengenai penerapan *cross-domain* SSO pada *municipal portal*. Dimana dalam penelitian ini menghasilkan kesimpulan bahwa dengan menggunakan CAS, semua informasi otentikasi pengguna akan disimpan pada *server* CAS, sehingga menghasilkan akses *cross-domain* pada layanan aplikasi yang terintegrasi dengan CAS [4]. Selain penambahan-penambahan yang dilakukan penulis terhadap penelitian pendahulu bagian pertama dan kedua, penulis akan melakukan pengujian dari sisi *black box*, *white box* dan *load test* dari sistem yang telah dibangun, sehingga pengguna dapat menilai kualitas dari sistem SSO yang telah diimplementasikan tersebut.

3. Perancangan Sistem

Pada tahap perancangan sistem SSO yang diimplementasikan pada sistem informasi Uiversitas Udayana tentunya terdapat beberapa tahapan yang mengacu sehingga sistem tersebut nantinya dapat berjalan sesuai dengan harapan yang direncanakan. Adapun tahapan-tahapan pada bagian perancangan sistem ini di paparkan pada gambar 1 berikut :



Gambar 1. *Flowchart Analisa Dan Perancangan Sistem*

Seperti terlihat pada gambar 1, terdapat beberapa tahapan untuk dapat membangun dan memastikas sistem *single sign on* yang di bangun dapat berjalan sesuai dengan harapan. Adapun metode yang digunakan untuk membangun SSO tersebut adalah *Central Authentication Service* (CAS). CAS merupakan metode *authentication* yang sangat mengutamakan tingkat keamanan informasi pengguna. Berpedoman pada konsep *Central Authentication Service*, untuk melakukan pengamanan tersebut terdapat dua aspek yang diperhatikan oleh CAS. Dimana aspek tersebut adalah proses *authentication user* dan *accessing a protected web resource when authenticated*. Untuk *authenticating user*, ketika *username* dan *password* dari pengguna benar, maka *server* CAS akan mengirim *cookie* berupa TGC (*Ticket Granting Cookie*) pada *browser*. TGC merupakan paspor pengguna terhadap *server* CAS. Waktu hidup dari paspor tersebut tergantung definisi saat implementasi *server* CAS. Paspor ini berguna untuk meniadakan *re-authenticate* ketika pengguna mengakses aplikasi lain. Sedangkan untuk *accessing a protected web resource when authenticated*, ketika pengguna mengakses layanan yang terintegrasi dengan CAS, *server* CAS akan terlebih dahulu mengkonfirmasi TGC yang dimiliki oleh *browser*. Ketika TGC tersebut sudah sesuai, maka *server* CAS akan memberikan ST (*Service Ticket*) kepada layanan. ST tersebut merupakan *opaque ticket*, dimana tiket ini tidak

menyimpan informasi dari pengguna dan hanya dapat digunakan untuk satu layangan saja, sehingga dapat mengoptimalkan keamanan identitas pengguna saat terjadinya proses *authentication*. ST tersebut nantinya akan di validasi oleh CAS *Client* yang terpasang pada tiap-tiap layanan yang terintegrasi, ketika valid maka pengguna dapat mengakses layanan yang disediakan oleh layanan yang terintegrasi. Selain pemaparan mengenai CAS, tentunya terdapat beberapa tahapan utama untuk dapat menghasilkan sistem SSO yang mampu berjalan sesuai dengan harapan instansi, berikut adalah pembahasan dari beberapa tahapan utama dari pembangunan sistem tersebut :

A. Analisis Kebutuhan

Universitas Udayana merupakan sebuah institusi pendidikan yang memiliki bebrapa layanan sistem informasi berbasis *web.* Untuk mengintegrasikan SSO pada sistem informasi Universitas Udayana tentunya diperlukan suatu proses analisis pendahulu, sehingga nantinya SSO tersebut dapat terintegrasi dengan baik pada sistem informasi yang sudah ada sebelumnya. Dimana proses analisi ini akan dibagi menjadi dua bagian yaitu *baselining* dan *needs analysis.*

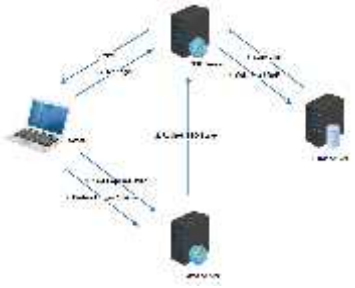
Dari sisi *baselining* layanan aplikasi berbasis *web* yang siap diintegrasikan dengan SSO adalah Simak, *E-Learning* dan *Blog System* Universitas Udayana. Dimana Simak merupakan sistem yang mengolah data dan melakukan proses kegiatan akademik yang melibatkan antara mahasiswa, dosen dan administrasi akademik. *E-Learning* merupakan media pembelajaran berbentuk *web* yang digunakan oleh Universitas Udayana, dimana aplikasi ini memungkinkan mahasiswa untuk masuk ke dalam ruang kelas digital untuk mengakses materi-materi pembelajaran. Tidak hanya mahasiswa, pengajarpun dapat mengunggah materi ajar, soal dan tugas pada aplikasi ini dan kemudian mahasiswa memilih kursus yang disediakan. Sedangkan *Blog System* Universitas Udayana merupakan suatu wadah yang dapat digunakan untuk berbagi informasi yang dimilikinya baik itu berupa ilmu–ilmu baru, penelitian yang dilakukan ataupun informasi akademik yang dimiliki pengguna *blog*. Dari sisi *needs analysis* ketiga aplikasi tersebut memiliki proses otentikasi dan basis data yang berbeda. Untuk memudahkan pengguna dalam melakukan otentikasi tentunya proses otentikasi dari ketiga aplikasi tersebut akan dikembangkan menjadi sistem SSO.

B. Perancangan Arsitektur Sistem Single Sign On

SSO merupakan layanan yang memungkinkan pengguna untuk mengakses ke semua sistem aplikasi yang terintegrasi oleh sistem SSO dengan hanya melakukan satu kali proses *login* [7]. Berikut adalah arsitektur dari sistem SSO yang diimplementasikan oleh penulis.

Gambar 2 merupakan penggambaran umum dari sistem yang dibangun oleh penulis, dimana komponen yang terlihat pada gambar 2 terdiri dari CAS *Server*, LDAP *Server* dan Aplikasi *Web Server*. Dimana CAS *Server* tersebut merupakan tempat ditanamkannya sistem SSO dengan menggunakan *Central Authentication Service* (CAS), sedangkan LDAP *Server* merupakan tempat peletakkan *database user* dari semua aplikasi berbasis *web* dan *Web Server* merupakan tempat menanamkan aplikasi berbasis *web* yang akan diintegrasikan dengan sistem SSO. Alur kerja dari sistem SSO tersebut dapat dijelaskan sebagai berikut :

1. *Client* akan mengakses aplikasi berbasis *web*
2. Ketika *client* menekan tombol *login*, aplikasi berbasis *web* akan *redirect browser* menuju SSO *Server*.
3. *Client* akan memasukkan *username* dan *password* untuk otentikasi.
4. *Username* dan *password client* akan dicocokkan pada LDAP.
5. Setelah melakukan pengecekan informasi data *client* akan dikirim kembali menuju SSO *Server.*
6. *Client* akan melewati proses otentikasi dan kembali ke aplikasi berbasis *web* dengan sebuah tiket.
7. *Client* yang sah dapat mengakses informasi yang terdapat pada aplikasi berbasis *web*.



Gambar 2. *Arsitektur Sistem SSO Berbasis CAS*

C. Implementasi dan Konfigurasi

Aplikasi utama yang harus diimplementasikan untuk membangun sistem SSO adalah CAS *Server*. Aplikasi ini nantinya akan memberikan layanan *Single Sign On*. Aplikasi CAS *Server* ini akan meneruskan ke halaman *web service* dan memberikan tiket sebagai bukti hak akses. Untuk mengimplementasikan *Single Sign On* otentikasi *user* CAS menggunakan LDAP. Langkah pertama yang harus disiapkan adalah *server* Ubuntu *Server* 12.04 LTS dengan *Apache2*, *Tomcat Java Server* dan aplikasi *Apache Maven* serta SSL. Lalu dilanjutkan dengan menanam aplikasi CAS *Server*.

Untuk membuat agar CAS *Server* otentikasinya menggunakan LDAP maka perlu melakukan editing pada berkas *deployerConfigContext.xml*, *cas-servlet.xml* dan *pom.xml*. Karena aplikasi *web* yang diintegrasikan berbasis PHP, maka perlu diinstall *apache web server* dan *php5*. *Apache web server* bertugas menerima dan membalas permintaan situs yang datang dari klien di *web server*. Php5 berfungsi mendukung bahasa pemrograman php pada *web server*.

Untuk membangun aplikasi berbasis *web* yang mendukung SSO perlu dilakukan modifikasi pada berkas yang berbeda pada masing–masing aplikasi *web*. Tombol *login* maupun *logout* pada aplikasi *web* diarahakan menuju halaman *login* dan *logout* CAS *Server*. Halaman pemrosesan *login* harus memuat fungsi GET tiket yang dikirimkan CAS *Server* dengan validasi *login* berupa *username*. Halaman *logout* tidak perlu memuat fungsi khusus selain fungsi pemusnahan *session* atau *cookie* lokal aplikasi *web*.

D. Pengujian Sistem

Dalam penelitan ini terdapat empat jenis pengujian yang dilakukan oleh penulis, dimana jenis–jenis pengujian tersebut adalah pengujian *black box*, pengujian *white box*, pengujian *response time* dan pengujian *load test*. Untuk lebih jelasnya mengenai jenis–jenis desain pengujian tersebut akan dibahas lebih lengkap pada pembahasan di bawah :

1. Pengujian *black box* yang digunakan dalam penelitian ini adalah pengujian *input* dan *output* dari sistem *Single Sign On*. Pengujian ini ditujukan untuk mengetahui kebenaran *output* dari perintah yang dimasukkan
2. Pengujian *white box* yang digunakan dalam penelitian ini adalah pengujian basis *path*. Notasi yang digunakan untuk menggambarkan jalur eksekusi adalah notasi diagram alir (grafik program), yang menggunakan notasi lingkaran (*node*) dan anak panah (*link*). Pengujian basis *path* pada penelitian ini difokuskan pada proses *login*.
3. *Response time* merupakan pengujian yang dilakukan untuk mengetahui waktu yang dibutuhkan oleh *server* SSO dalam melayani *user* untuk melakukan otentikasi. Dalam hal ini, *response time* merupakan komponen yang sangat penting terhadap layanan aplikasi *web*. Manajemen yang efisien dari waktu *response time* menandakan ketersediaan layanan *web* tersebut [6]. Pengujian *response time* dilakukan dengan melakukan proses otentikasi pada *server Single Sign On*. Jumlah *user* yang melakukan peroses otentikasi secara bersamaan yaitu satu *user*, dua *user* dan empat *user*.
4. *Load test* merupakan proses penilaian prilaku dari suatu sistem saat berada pada kondisi tertekan atau terbebani [8]. Pengujian ini dilakukan untuk mengetahui kinerja *server* SSO saat terbebani, sehingga nantinya akan diketahui bagaimana kinerja dari *server* SSO dalam menangani otentikasi dalam jumlah *user* yang banyak dalam waktu yang bersamaan. Pengujian *load test* dilakukan dengan melakukan proses otentikasi pada *server* SSO dengan jumlah *user* yang bervariasi.

4. Hasil Pengujian

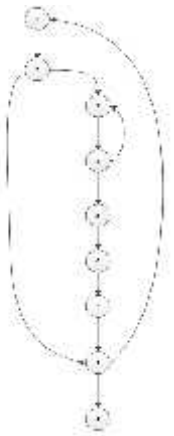
Hasil dari setiap jenis pengujian sistem SSO tentunya akan menghasilkan sebuah informasi yang sangat dibutuhkan untuk mengevaluasi sistem SSO tersebut. Adapun hasil dari setiap jenis pengujian yang telah dilakukan adalah :

A. Pengujian Black Box

Dari hasil pengujian *black box* yang telah dilakukan dalam penelitian ini sistem SSO telah dapat memenuhi harapan *output* yang diharapkan pada saat melakukan pengujian. Dimana ketika *user* mengakses pada aplikasi Simak, *output* yang dihasilkan berupa sistem redirek menuju SSO *server* dan melakukan proses *login* hingga terotentikasi. Setelah terotentikasi pada Simak, *user* selanjutnya mengakses aplikasi *E-Learning*, *output* yang dihasilkan berupa *user* secara otomatis terotentikasi pada aplikasi *E-Learning*. Hal ini juga berlaku pada *Blog System* setelah melakukan *login* pada salah satu aplikasi terintegrasi, maka *user* tidak perlu lagi melakukan proses *login* ketika mengakses aplikasi lainnya.

B. Pengujian White Box

Pengujian tersebut ditujukan untuk melihat *path-path* yang dilalui saat program dijalankan. Pengujian basis *path* pada penelitian ini difokuskan pada proses *login*. Gambar 3 adalah penggambarn *flowgraph* pada pengujian proses *login* :



Keterangan
1 : Pengaksesan aplikasi *web*
2 : Cek tiket
3 : Proses *login* SSO *server*
4 : Validasi *username* dan *password*
5 : Redirek aplikasi *web*
6 : Validasi tiket
7 : *Get username*
8 : Validasi *username* pada *database* aplikasi
9 : *Login* pada aplikasi *web*

Gambar 3. *Flowgraph Login Aplikasi*

V(G) = E – N + 2
V(G) = 11 – 9 + 2
V(G) = 4
Jalur pengujian :

1) Jalur 1 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 - 9
2) Jalur 2 : 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 - 1
3) Jalur 3 : 1 – 2 – 8 – 9
4) Jalur 4 : 1 – 2 – 8 - 1

Nilai V(G) yang didapat adalah 4, dimana terdapat 4 jalur pengujian yang didapat berdasar perhitungan dari *flowgraph* tersebut. Dari pengujian tersebut nilai *cyclomatic complexity* yang dimiliki adalah 4. Dimana *cyclomatic complexity* tersebut merupakan suatu nilai yang menentukan jumlah jalur dalam basis suatu program dan memberikan batas atas jumlah uji coba yang harus dikerjakan untuk menjamin bahwa seluruh perintah sekurang-kurangnya telah dikerjakan sekali.

Dari hasil pengujian *white box* yang telah dilakukan, sistem SSO yang telah dibangun dapat diintegrasikan dengan aplikasi yang menggunakan basis data selain LDAP. Hal ini dilakukan dengan mengambil nilai tiket yang didapatkan setelah melakukan otentikasi pada *server* SSO. *Username* dari tiket yang telah dibuat pada *server* SSO tersebut akan diambil pada node 7 dan dilakukan pencocokan pada node 8, ketika *username* pada tiket sesuai dengan *username* pada aplikasi yang terintegrasi maka *user* tersebut dapat menikmati layanan yang disediakan pada aplikasi web tersebut.

C. Pengujian Response Time

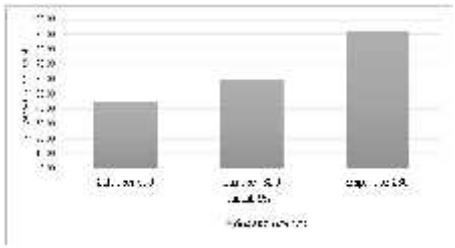
Response time merupakan pengujian yang dilakukan untuk mengetahui waktu yang dibutuhkan oleh *server* SSO dalam melayani *user* untuk melakukan otentikasi.

Berikut adalah hasil dari pengujian *response time* dari sistem SSO yang terlihat pada tabel 1 :

Tabel 1. *Pengujian Response Time*

Jumlah User	*Response Time (ms)*
Satu *User* SSO	44,57
Dua *User* SSO	59,47
Tiga *User* SSO	92,10

Sesuai dengan tabel 1, terdapat peningkatan nilai *response time* dari pengujian yang menggunakan satu *user*, dua *user* dan empat *user*.



Gambar 4. *Grafik Pengujian Response Time*

Berdasarkan grafik pada gambar 4, terlihat antara pengujan *response time* menggunakan satu *user* dan dua *user* memiliki peningkatan *response time* sebesar 14,90 ms. Sedangkan peningkatan *response time* antara pengujan menggunakan satu *user* dan dua *user* sebesar 32,63 ms.

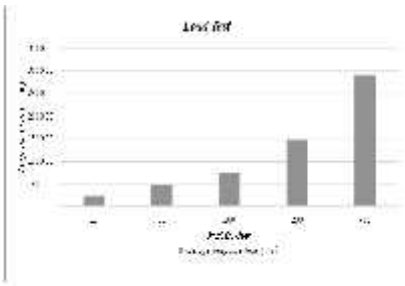
D. Pengujian Load Test

Load test merupakan pengujian yang dilakukan untuk mengetahui kinerja *server* SSO saat diberikan beban berlebih, sehingga nantinya akan diketahui bagaimana kinerja dari *server* SSO dalam menangani otentikasi dalam jumlah *user* yang banyak dalam waktu yang bersamaan. Berikut adalah hasil dari pengujian *load test* dari sistem SSO :

Tabel 2. *Pengujian Load Test*

Jumlah User	*Average Response Time (ms)*	*Error (%)*
10	231,73	0,00
50	480,73	0,00
100	762,73	0,00
200	1.480,73	0,00
400	2.909,10	0,25

Sesuai dengan tabel 2 di atas pada pengujian dengan jumlah 200 *user* yang melakukan otentikasi *average response time* sebesar 1.480,73 ms dan pada 400 *user* yang melakukan otentikasi *average response time* sebesar 2.909,10 ms. Untuk lebih jelas dalam mengamati perubahan *average response time* dan *average error* dari pengujian *load test*, hasil pengujian tersebut disajikan dalam bentuk grafik. Gambar 5 di bawah adalah grafik pengujian *load test*.



Gambar 5. *Grafik Average Response Time*

Berdasarkan grafik pada gambar 5 dapat dilihat bahwa semakin banyak jumlah *request* yang melakukan otentikasi pada *server* SSO maka nilai *response time* semakin besar. Dimana nilai *response time* tertinggi terjadi pada 400 jumlah *user* melakukan otentikasi dengan *average response time* sebesar 2,909,10 ms.

Gambar 6. *Grafik Persentase Error*

Gambar 6 menunjukkan bahwa semakin banyak jumlah *user* yang melakukan otentikasi pada *server* SSO akan menyebabkan terjadinya *error*. Pada pengujian yang sudah dilakukan, *server* SSO yang diimplementasikan pada penelitian ini mampu melayani otentikasi maksimal sebanyak 200 *user* tanpa *error*. *Error* terjadi pada saat otentikasi sebanyak 400 *user* dengan persentase *error* sebesar 0,25%. *Error* yang dimaksud dalam pengujian ini adalah adanya kegagalan *user* dalam melakukan proses otentikasi terhadap sistem *Single Sign On*. Berikut ini adalah jumalah *error* yang terjadi berdasarkan pengujian *load test* :

Tabel 3. *Jumlah Error*

Jumlah User	*Jumlah Error*
10	0
50	0
100	0
200	0
400	1

Tabel 3 memperlihatkan jumlah *error* yang terjadi pada pengujian *load test*. Dimana dalam pengujian *load test* sebanyak 400 *user* yang melakukan proses otentikas terdapat *error* sebanyak satu buah. *Error* yang terjadi pada pengujian tersebut memiliki pesan *error rc="500" rm="Internal Server Error*". Pesan *error* tersebut didapatkan dari *log* pengujian yang didapat pada *JMeter*. Pesan *error rc="500" rm="Internal Server Error*" tersebut dapat diartikan bahwa *apache* tidak dapat menangani proses yang masuk sehingga *apache* menjadi *timeout*.

E. Analisa Keseluruhan

Dalam penelitian ini, sistem *Single Sign On* telah telah mampu menangani otentikasi secara terpusat pada sistem informasi Universitas Udayana. Dimana sistem informasi tersebut adalah Simak, *E-Learning* dan *Blog System* Universitas Udayana. Hal ini dapat dilihat dari pengujian yang telah dilakukan dimana setelah melakukan otentikasi pada *server* SSO maka ketika mengakses Simak, *E-Learning* dan *Blog System* tidak perlu melakukan proses otentikasi untuk yang kedua kalinya. Hal ini tidak hanya berlaku jika sistem informasi yang pertama kali dikases adalah Simak, sistem informasi apapun yang diakses pertama kali dengan ketentuan sebelumnya sudah melakukan otentikasi maka disaat mengakses sistem informasi yang lain *user* tidak perlu lagi untuk melakukan otentikasi.

Sistem *Single Sign On* yang telah diimplementasikan pada penelitian ini telah dapat menangani proses otentikasi dengan jumlah *user* yang mengakses sebanyak satu *user* memerlukan waktu *response time* sebesar 44,57 ms, sedangkan untuk proses otentikasi dua *user* memerlukan waktu *response time* sebesar 59,47 ms dan untuk proses otentikasi empat *user* memerlukan waktu *response time* sebesar 92,10 ms, hal ini berarti sistem yang telah dibangun sudah dapat menangni otentikasi dengan baik.

Dalam pengujian *load test*, sistem SSO yang telah diimplementasikan diberikan beban *user* yang melakukan otentikasi secara bersamaan. Dimana jumlah *user* tersebut adalah dimulai dari 10 *user*, 50 *user*, 100 *user*, 200 *user* dan 400 *user*. Dalam penelitian ini sistem SSO dapat menangani proses otentikasi sebanyak sebanyak 200 *request* tanpa *error* dengan *average response time* sebesar 1.480,73 ms. *Error* terjadi saat otentikasi sebanyak 400 *request* dengan persentase *error* sebesar 0,25% dan *average response time* sebesar 2.909,10 ms. Rata–rata besaran *response time* yang terjadi semakin bertambah besar seiring dengan bertambahnya *request* yang melakukan proses otentikasi, hal ini dapat dilihat ketika hanya 10 *request* yang melakukan otentikasi secara bersamaan *average response time* yang dihasilkan sebesar 231,73 ms dan sampai dengan 200 *request* yang melakukan otentikasi secara bersamaan *average response time* yang dihasilkan sebesar 1.480,73 ms. Dalam pengujian *load test* yang telah dilakukan terdapat *error* otentikasi yang terjadi pada pengujian *load test* dengan menggunakan 400 *user*. Dimana pesan *error* yang terjadi pada pengujian itu adalah *rc="500" rm="Internal Server Error*". Pesan *error* tersebut dapat diartikan bahwa *error* tersebut terjadi karena *apache* tidak dapat menangani proses yang masuk sehingga *apache* menjadi *timeout*.

Dari hasil penelitian yang telah diperoleh, sistem *Single Sign On* pada sistem informasi Universitas Udayana dalam hal ini Simak, *E-Learning* dan *Blog System* telah mampu memberikan layan otentikasi secara terpusat, sehingga *user* hanya perlu melakukan satu kali proses *login* untuk mendapatkan layanan dari Simak, *E-Learning* dan *Blog System.*

5. Kesimpulan

Berdasarkan pada penelitian yang telah dilakukan, maka dapat diambil kesimpulan bahwa sistem *Single Sign On* yang diintegrasikan dengan sistem informasi Universitas Udayana dalam hal ini Simak, *E-Learning* dan *Blog System* telah mampu menangani otentikasi secara terpusat dan dapat diintegrasikan dengan aplikasi yang tidak menggunakan *Lighweight Directory Access Protocol* (LDAP) sebagai manajemen *user*. Dalam proses otentikasi yang dilakukan oleh *user*, sistem *Single Sign On* dapat menangani proses otentikasi sebanyak 200 *user* secara bersamaan tanpa terjadi *error* dengan *average response time* sebesar 1.480,73 ms dan dapat menangani proses otentikasi sebanyak 400 *user* secara bersamaan dengan *error* sebesar 0,25% serta *average response time* sebesar 2.909,10 ms.

Daftar Pustaka

[1] J. Wang, G. Wang, and W. Susilo, "Secure Single Sign-On Schemes Constructed from Nominative Signatures," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2013, pp. 620–627.

[2] J. Hu, Q. Sun, and H. Chen, "Application of Single sign-on (SSO) in Digital Campus," in *2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*, 2010, pp. 725–727.

[3] N. Dian, N. K. Yesi, and S. Rusmala, "Single Sign On (Sso) Dengan Menggunakan Lightweight Directory Access Protocol (Ldap) Studi Kasus Di Universitas Bina Darma," *J. Mhs. TI S1*, no. JURNAL-TIS1-UBD-DIAN, 2013.

[4] H. Hu and Z. Guo, "The application of cross-domain single sign-on in municipal portal," in *TENCON 2013 - 2013 IEEE Region 10 Conference (31194)*, 2013, pp. 1–4.

[5] K. I. S. Muhammad Yanuar Ary . S., "Implementasi Sistem Single Sign On / Single Sign Out Berbasis Central Authentication Service Protocol Pada Jaringan Lightweight Directory Access Protocol Universitas Diponegoro," vol. 1, no. 3, 2012.

[6] B. Gueye, I. Niang, B. Gueye, M. O. Deye, and Y. Slimani, "Constraints-based response time for efficient QoS in Web Services Composition," in *2011 7th International Conference on Next Generation Web Services Practices (NWeSP)*, 2011, pp. 141–146.

[7] T. Yulin and Z. Feng, "The analysis and design for single sign-on in the mobile Application Data Center," in *2012 3rd International Conference on System Science, Engineering Design and Manufacturing Informatization (ICSEM)*, 2012, vol. 2, pp. 258–260.

[8] Z. Jiang and A. Hassan, "A Survey on Load Testing of Large-Scale Software Systems," *IEEE Trans. Softw. Eng.*, vol. PP, no. 99, pp. 1–1, 2015.

Biodata Penulis

I Putu Agus Eka Darma Udayana, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika Universitas Udayana tahun 2014. Saat ini sedang menempuh studi Magister pada Jurusan Manajemen Sistem Informasi Dan Komputer Universitas Udayana.

Lie Jasa, memperoleh gelar Ir Teknik Elektro Institut Teknologi Sepuluh Nopember pada tahun 1990, gelar MT Teknik Elektro Institut Teknologi Sepuluh Nopember pada tahun 1998 dan gelar Dr Teknik Elektro Institut Teknologi Sepuluh Nopember pada tahun 2015.