

Analisis Web Performance dan Load Test

Studi Kasus: Topologi Cloud Microsoft Azure Test Rig pada I-banking Bank XYZ

Guntoro¹, Dana Sulistyio Kusumo, Ph.D², Dr. Adiwijaya³

1103104185,02780291-1,00740196-1

^{1,2,3}*Teknik Informatika, Telkom School of Computing, Universitas Telkom, Bandung, Indonesia*

Abstrak – Dewasa ini peran Internet Banking memegang peranan penting dalam suatu perbankan. Bank XYZ adalah salah satu Bank yang sedang berkembang proses bisnisnya, begitu juga dengan segi teknologi yang digunakannya. Dalam upaya pencapaian tujuan Bank XYZ menjadi lebih baik, diperlukannya sistem I-banking untuk melayani semua kebutuhan dari nasabah. Sistem I-banking yang *powerfull* menjadi salah satu kunci untuk mendapat kepercayaan yang tinggi dari nasabah.

Untuk mengetahui kapasitas maksimum operasi I-banking Bank XYZ serta adanya kemacetan (*bottlenecks*) yang menyebabkan degradasi, sangatlah diperlukan untuk melakukan pengujian beban pada sistem I-banking. Maka dari itu diperlukannya pembangunan *Test Rig* dalam *Cloud Microsoft Azure* untuk melakukan *Web Performance* dan *Load Test* pada I-banking Bank XYZ, dengan skenario dan target user sesuai yang dibutuhkan. Sehingga dapat dengan mudah untuk mengidentifikasi faktor penyebab performansi dan skalabilitas *bottlenecks* dari web I-banking tersebut.

Kata Kunci: I-banking, *Test Rig*, *Cloud Microsoft Azure*, *Web Performance*, *Load Test*, *Bottlenecks*.

I. PENDAHULUAN

Internet Banking, untuk selanjutnya di sebut I-banking merupakan suatu layanan perbankan yang melalui jaringan Internet dengan menggunakan website milik Bank sebagai perantaranya. Penyelenggaraan I-banking merupakan penerapan aplikasi teknologi informasi yang terus berkembang dan dimanfaatkan untuk menjawab keinginan nasabah perbankan yang menginginkan pelayanan cepat, aman, nyaman, murah dan dapat

diakses dari mana saja baik itu dari HP, komputer, PDA dan sebagainya.

Pada awal tahun 2014 ini Bank XYZ baru saja selesai membangun sistem I-banking versi terbarunya. Maka dari itu sebelum digunakan oleh nasabah dan juga untuk mengetahui kapasitas maksimum operasi I-banking Bank XYZ serta adanya kemacetan (*bottlenecks*) yang menyebabkan degradasi, diperlukannya untuk melakukan pengujian beban pada sistem I-banking. Pengujian dilakukan dalam perangkat lunak maupun perangkat keras (*server*) dengan cara melakukan *stress test* pada sistem I-banking. Proses pengujian harus melalui *Cloud* atau melalui jaringan Internet karena untuk dapat mensimulasikan seolah-olah user asli saat membuka web I-banking tersebut. Hal ini berarti dibutuhkannya *Cloud Computing* yang mana memanfaatkan sumber daya teknologi komputer (komputasi) berbasis internet [1]. Dari data-data yang dihasilkan dapat dengan mudah untuk menganalisis faktor-faktor mana sajakah yang mempengaruhi performansi sistem I-banking Bank XYZ.

Hanya saja untuk mendukung proses pengujian di atas, dibutuhkannya infrastruktur yang sangat komplek seperti mesin virtual dengan spesifikasi yang tinggi, *bandwidth* konektifitas dengan jaringan yang besar, *Test Agent*, *Test Controller* dan lain-lain [2]. Maka dari itu diperlukannya layanan yang menyediakan infrastruktur di mana pengguna dapat membangun satu atau lebih mesin virtual sesuai yang dibutuhkan. Semua komponen di atas yang saling terkait dinamakan *Test Rig*.

Saat melakukan proses pengujian, dibutuhkan juga sistem pemantau performansi *Test Rig* seperti

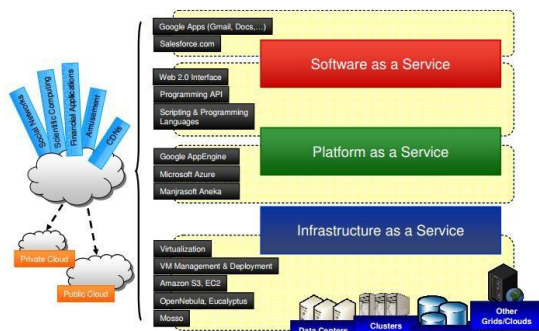
penggunaan *resource hardware* dan penggunaan konektivitas jaringan internet *Test Rig* berdasarkan *response time*, *throughput* dan *error rate* nya. Sehingga ketika menghasilkan laporan pengujian akan mendapatkan nilai yang terbaik tanpa ada faktor masalah internal dari *Test Rig*.

Semua layanan yang dibutuhkan untuk membangun *Test Rig* tersedia pada platform *Microsoft Azure* yang mendukung pemakaian Topologi *Cloud Microsoft Azure Test Rig*. Oleh karena itu berdasarkan permasalahan di atas, akan lebih mudah dan cepat jika pembangunan *Test Rig* dilakukan pada platform *Microsoft Azure* untuk melakukan proses pengujian pada sistem I-banking Bank XYZ.

II. TINJAUAN PUSTAKA

2.1. Cloud Computing

Cloud Computing merupakan tren teknologi terbaru yang mempunyai tujuan untuk memenuhi tuntutan sumber daya informasi teknologi dengan pemanfaatan teknologi komputer (komputasi) berbasis Internet [3].



Gambar 2-1 Arsitektur Cloud Computing [3]

2.2. Microsoft Azure

Microsoft Azure adalah sebuah platform yang menyediakan komputasi awan dan infrastruktur yang dibuat oleh Microsoft untuk membangun, menempatkan dan mengelola aplikasi dan layanan melalui jaringan global pusat data yang dikelola Microsoft dan mendukung kebutuhan layanan khusus dari pengembangan Tim, pelanggan dan pengguna [4].

2.3. Software Performance

Software Performance adalah kualitas dari sistem perangkat lunak yang mana semuanya saling mempengaruhi dari perangkat lunak itu sendiri untuk semua lapisan dasar seperti sistem operasi, *middleware*, perangkat keras, jaringan komunikasi, dan lain-lain [6].

2.4. Software Testing

Software Testing adalah proses mengeksekusi sistem perangkat lunak untuk menentukan apakah itu cocok spesifikasinya dan mengeksekusi ke dalam lingkup sistem yang ditargetkan [7].

2.5. Software Performance Testing

Software Performance Testing adalah suatu proses pengujian untuk mengukur kinerja perangkat lunak. Tes kinerja ini meliputi berbagai macam pengujian, masing-masing memiliki tujuan dan metode pengawasan.

2.5.1. Web Performance Test

Web Performance Test adalah serangkaian proses pengujian untuk mengukur kemampuan dari suatu website. Dalam Rekayasa Perangkat lunak, *Performance Testing* adalah suatu pengujian yang dilakukan untuk menentukan bagaimana sistem melakukan respon dan stabilitas di bawah beban kerja tertentu (degradasi kinerja), sehingga dengan melakukan pengujian akan dihasilkan data yang dapat di pakai untuk menyelidiki, mengukur, memvalidasi atau memverifikasi dari atribut kualitas sistem tersebut [8] [9].

Dengan *Web Performance Test* dapat dengan mudah membangun sebuah kerangka kerja tes untuk dilakukan secara berulang yang dapat membantu dalam menganalisis kinerja aplikasi website dan mengidentifikasi hambatan potensial.

2.5.2. Load Test

Load Test adalah bentuk sederhana dari *Performance Testing*, sebuah uji beban yang dilakukan untuk memahami perilaku sistem di bawah beban yang diharapkan secara spesifik [8].

Pengujian ini akan memberikan sebuah respon dan data dari semua transaksi yang dilakukan sesuai skenario, sehingga dapat dengan mudah mencari

kemacetan atau *bottlenecks* yang mempengaruhi performansi dari suatu sistem yang telah diuji.

Load Testing ini biasa digunakan bersamaan dengan *Web Performance Test* untuk melakukan kombinasi pengujian, beban, pengujian stress aplikasi website.

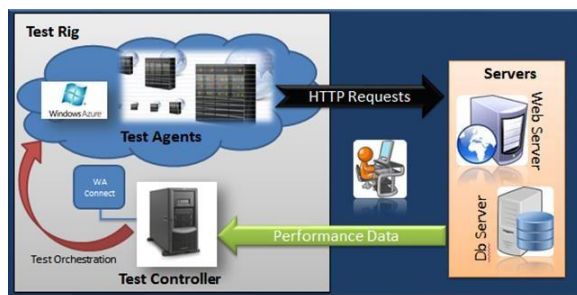
2.6. Test Agent dan Controller

Saat menjalankan pengujian jarak jauh (*remote*) pada beberapa mesin virtual, atau untuk mengumpulkan data dan mendiagnosa hasil dari pengujian jarak jauh diperlukannya *Test Agent* dan *Test Controller*. *Test Controller* berjalan sebagai sebuah layanan dan memberikan tugas kepada *Test Agent* untuk menjalankannya (skenario pengujian). Sedangkan *Test Agent* yang melakukan pengujian, mengumpulkan data hasil pengujian, dan menyimpannya pada *SQL* di *Test Controller* [10].

2.7. Topologi Cloud Microsoft Azure Test Rig

Ketika membangun *Test Rig* pada *Cloud Microsoft Azure*, perlu digunakannya Topologi *Test*

Rig, yang terdiri dari *Test Agent* dan *Test Controller*. Media komunikasi antara *Test Agent* dan *Test Controller* menggunakan *Windows Azure Connect*. Berikut gambaran Topologinya:



Gambar 2-2 Topologi Test Rig Menggunakan Cloud Microsoft Azure [2]

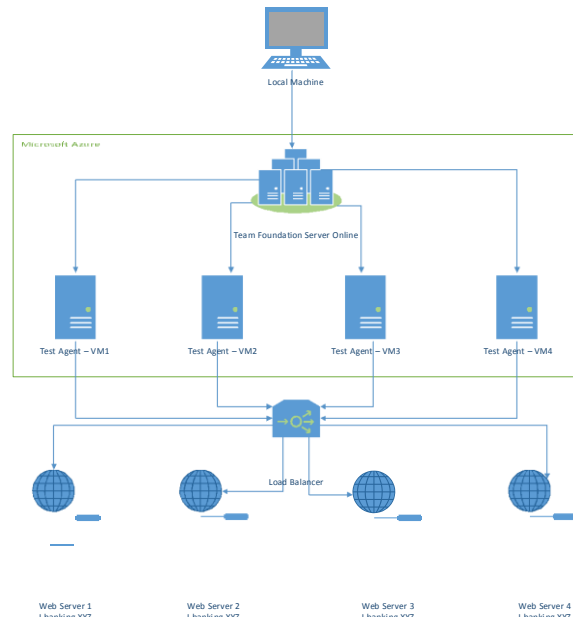
2.8. Team Foundation Server

Team Foundation Server adalah produk server terpisah yang dirancang khusus untuk tim rekayasa perangkat lunak seperti pengembang, penguji, arsitek, manajemen proyek, analis bisnis dan orang yang berkontribusi terhadap rilis pengembangan perangkat lunak/proyek [11].

III. PERANCANGAN SISTEM PENGUJIAN

3.1. Gambaran Umum Sistem

Gambaran umum sistem dengan menggunakan Topologi *Cloud Microsoft Azure Test Rig* untuk melakukan *Web Performance* dan *Load Test* pada *Web Server* I-banking Bank XYZ sebagai berikut:

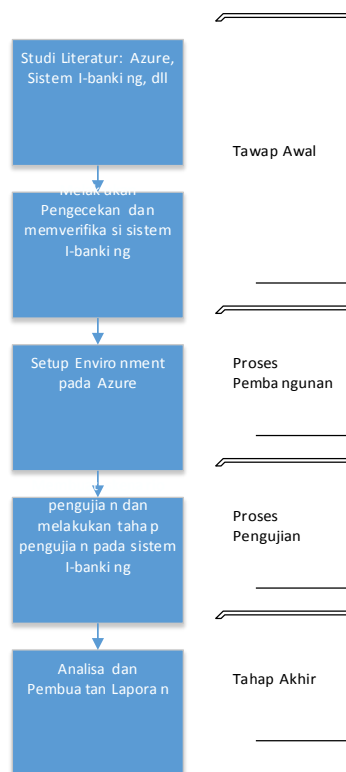


Gambar 3-1 Gambaran umum sistem ketika melakukan Testing

Pada gambar di atas terlihat ada 4 *Virtual Machines* yang digunakan dalam melakukan pengujian ini. Dengan tujuan agar *web server* yang digunakan oleh sistem I-banking akan berkerja/aktif semua. Sehingga mendapatkan hasil pengujian yang maksimal ketika melakukan *stress test* pada sistem I-banking Bank XYZ.

Selain itu komputer lokal digunakan untuk menjalankan semua konfigurasi seperti pembuatan skenario pengujian, kustomisasi jumlah user, tipe *load test*, dan lain-lain. *Test Controller* berfungsi untuk meneruskan permintaan dari pengguna pada *Test Agent* untuk melakukan *testing* sesuai dengan skenario yang telah di buat dan menyimpan semua hasil dari pengujian ke dalam *SQL*. *Test Agent* dapat disebut sebagai *end-point* dari *Test Rig*, yang berfungsi sebagai pembuat user virtual untuk melakukan request pada sistem I-banking sesuai skenario yang telah diatur.

3.2. Langkah Pengujian



Gambar 3-2 Diagram Blok Langkah Pengujian

Gambar 3-2 di atas adalah sebuah diagram blok yang menjelaskan langkah pengujian. Ada 4 tahapan/proses yang akan dilakukan di antaranya: Proses tahap awal (studi literature, *checking*), proses pembangunan (setup *environment* pada Azure), proses pengujian dan proses tahap akhir (Analisa dan pembuatan laporan).

3.3. Testing Tools

Dalam melakukan pengujian pada sistem I-banking Bank XYZ ini dilakukan menggunakan beberapa tools di antaranya:

a. Microsoft Visual Studio 2013 Ultimate

Microsoft Visual Studio 2013 versi Ultimate merupakan sebuah perangkat lunak (*tools*) yang dapat digunakan untuk melakukan *Web Performance* dan *Load Test*. Versi ultimate adalah satu-satunya versi produk Visual Studio yang mendukung untuk melakukan *Web Performance* dan *Load Test*.

b. Team Foundation Server Online

Team Foundation Server Online digunakan untuk dapat melakukan *sharing file* pengujian pada setiap *Virtual Machines*.

c. Internet Explorer versi 9.0

Web browser yang digunakan adalah Internet Explorer versi 9.0. Internet Explorer merupakan satu-satunya web browser yang mendukung untuk dapat melakukan *record* skenario pada *web performance test* dalam Visual Studio, dengan versi yang paling rendah adalah 9.0.

3.4. Skenario Pengujian

Pada penelitian ini skenario yang akan dipakai hanya proses *Transaction History* dan *Balance Inquiry* karena pada skenario lainnya harus membutuhkan keamanan token I-banking untuk melakukan proses transaksi pengujian dan ada beberapa data yang bersifat privasi.

Dalam *web performance test* terdapat skenario pengujian yang akan dilakukan, berisi langkah-langkah *user* ketika mengakses website I-banking Bank XYZ dan terekam ke dalam file *.webtest* pada Microsoft Visual Studio. Maka dari itu akan dibuat 2 file *.webtest* (*web performance testing*) yang berisi skenario *Transaction History* dan skenario *Balance Inquiry*. Pengujian (*web performance test*) ini akan dilakukan jika *record* dari setiap skenario berhasil dilakukan.

Setelah selesai melakukan *web performance test* selanjutnya adalah tahap pengujian (*load testing*). Dalam tahap ini ada beberapa parameter pengujian yang akan dimasukan ke dalam *test case* (seperti yang sudah dijelaskan pada bagian Bab 3.2.3), diantaranya:

- Penggunaan *Think Time* selama 10 seconds. *Think Time* adalah waktu penundaan antara setiap permintaan (*request*). Hal ini merupakan salah satu pendekatan yang dapat mensimulasikan kebiasaan *user* asli ketika membuka web I-banking Bank XYZ.
- Pada tahap *Load Pattern* menggunakan tipe *constant load* atau *concurrent user* sebanyak 1000 user pada setiap *Virtual Machines*. Sehingga total *request* 4000 user dalam melakukan pengujian ini. Dengan

menggunakan tipe *concurrent user* ini akan membuat beban server menjadi berat, sehingga akan mudah terlihat *bottlenecks* yang terdapat pada sistem I-banking Bank XYZ.

- *Test Record* yang digunakan adalah file *TransactionHistory.webtest* dan *BalanceInquiry.webtest* yang sebelumnya di buat ketika melakukan *web performance test*.
- Penggunaan Internet dalam melakukan *load testing* ini menggunakan konektivitas dari *Virtual Machines* (LAN) yang terhubung dengan jaringan Internet pada Microsoft Azure ketika melakukan pengujian pada sistem I-banking Bank XYZ.
- Web browser yang digunakan dalam pengujian ini menggunakan Internet Explorer versi 9.0 yang merupakan satu-satunya web browser yang mendukung untuk dapat melakukan *record* skenario pada *web performance test* dalam Visual Studio, dengan versi yang paling rendah adalah 9.0.

3.5. Rencana Eksekusi Pengujian

Dalam melakukan pengujian pada sistem I-banking akan dilakukan setelah jam kerja Bank XYZ selesai yaitu pukul 16:00 WIB yang mana sistem I-banking tidak dipakai oleh karyawan Bank XYZ. Sehingga penggunaan *resource* sistem I-banking akan maksimal difokuskan untuk pengujian ini.

Pengujian (*Load Test*) pada kasus *Transaction History* akan dilakukan sebanyak 40.000 kali dengan maksimal 4000 user dan pada kasus *Balance Inquiry* akan dilakukan selama 20 menit dengan maksimal 4000 user juga. Dengan tujuan untuk mengetahui apakah ada hasil pengujian yang sangat berbeda antara batas penggunaan menggunakan waktu dan batas jumlah target transaksi pengujian.

Eksekusi pengujian dilakukan setelah dilakukannya *warming up* pada sistem I-banking Bank XYZ selama (minimal) 30 menit, untuk memastikan semua *Virtual Machines* dapat terhubung dengan baik pada sistem I-banking dalam melakukan pengujian ini.

IV. EKSEKUSI PENGUJIAN DAN ANALISIS HASIL

4.1. Eksekusi Pengujian

Hal yang perlu diperhatikan sebelum melakukan pengujian adalah penempatan *Virtual Machines* pada Microsoft Azure harus yang terdekat dengan *web server* pada sistem I-banking Bank XYZ, sehingga konektivitas jaringan internet tidak akan mempengaruhi hasil dari pengujian yang akan dilakukan. Pemilihan lokasi *cloud server* yang terdekat adalah pada zona Regional South East Asia - Singapore.

Sebelum melakukan pengujian dilakukan *warming up* terlebih dahulu dengan cara melakukan *load testing* pada sistem I-banking Bank XYZ selama 30 menit. Hal tersebut untuk memastikan semua *Virtual Machines* dapat terhubung dengan baik pada sistem I-banking dalam melakukan pengujian ini.

4.2. Web Performance Testing

Web Performance Test dilakukan untuk mengetahui seberapa responsif sistem I-banking Bank XYZ dan memastikan tidak ada kesalahan (*errors*) dalam skenario pengujian.

Pada pengujian ini dilakukan dengan cara menjalankan file *TransactionHistory.webtest* dan *BalanceInquiry.webtest* pada Microsoft Visual Studio dan selanjutnya menganalisa hasil datanya. Berikut hasil rata-rata nilai *Response Time* dari hasil *Web Performance Test* pada skenario *Transaction History* dan *Balance Inquiry* I-banking Bank XYZ, diantaranya:

4.2.1. Transaction History

Dalam pengujian (*web performance test*) pada skenario *Transaction History* ini dilakukan dengan 3 kali percobaan, berikut hasil datanya:

Table 4-1 Hasil Data Web Performance Test
(Transaction History)

Transaction History			
Pages	Min (/sec)	Max (/sec)	Avg (/sec)
Home	20.6	24	21.9
Login	12.6	18	15.5
IBTranscationHistory	6.1	18.8	11.5
Logout	1	11.5	6.1

Pada hasil di atas, dapat disimpulkan proses eksekusi yang memiliki nilai waktu pengujian paling tinggi skenario *Transaction History* adalah ketika proses masuk pada halaman *Home* dengan nilai 24 detik dan rata-ratanya 21.9 detik. Hal ini dapat disebabkan karena ketika membuka halaman depan terdapat berbagai macam tipe file seperti, gambar, css, js dan lain-lain. Proses logout merupakan proses yang memiliki nilai waktu transaksi paling kecil atau halaman yang paling ringan dalam proses pengujian ini, karena hanya menghilangkan *cookies* pada web browser klien.

4.2.2. Balance Inquiry

Dalam pengujian (*web performance test*) pada skenario *Balance Inquiry* ini dilakukan dengan 3 kali percobaan juga, berikut hasil datanya:

Table 4-2 Hasil Data Web Performance Test (Balance Inquiry)

Balance Inquiry			
Pages	Min (/sec)	Max (/sec)	Avg (/sec)
Home	2.9	8.5	5.1
Login	13.2	19.7	16.0
AccountPortfolio	12	21.1	15.0
Logout	0.5	1.6	0.9

Pada hasil di atas, dapat disimpulkan proses eksekusi yang memiliki nilai waktu pengujian paling tinggi dalam skenario *Balance Inquiry*

adalah ketika proses *login* dengan nilai 19.7 detik dan rata-ratanya 16.0 detik. Berbeda dengan skenario sebelumnya (*Trasaction History*) yang mana halaman login tidak terlalu lama dibandingkan proses ketika membuka halaman *home*. Akan tetapi rata-ratanya waktunya tidak jauh berbeda dari kedua proses ini. Proses logout merupakan proses yang memiliki nilai waktu transaksi paling kecil juga dalam proses pengujian ini, karena hanya menghilangkan *cookies* pada web browser klien.

4.3. Load Testing

Sebelum melakukan pengujian ini (*load testing*) dilakukan *warming up* terlebih dahulu dengan menggunakan skenario yang telah di buat selama 30 menit. Tujuannya adalah untuk memastikan semua sistem bekerja dengan baik dan memastikan pengujian (*load testing*) siap dijalankan. Pengujian ini dilakukan setelah jam kerja Bank XYZ selesai (di atas jam 16:00), sehingga penggunaan *resource* sistem I-banking akan maksimal difokuskan untuk pengujian ini.

4.3.1. Skenario Transaction History

Ada beberapa kategori hasil pengujian utama yang akan didapatkan dari Visual Studio, di antaranya: (*Test Result, Error Details, Request Files* dan *Test Rig*). Berikut ini hasil data yang didapatkan ketika melakukan pengujian (*load testing*) pada skenario *Trasaction History*:

Test Run Information:

- Scenario : Transaction History
- Think Time : 10 seconds
- Load Test Type : *Conccurent Load*
- Total Users : 4000 users (1000 users / VM)
- Duration : 10.000 request / VM (+- 55 minutes)
- Start Time : 4:48:03 PM

1) Test Result

Hasil pengujian pada kategori Test Result dalam skenario *Transaction History* sebagai berikut:

Table 4-3 Hasil Data Load Testing (Transaction History)

	VM 1	VM 2	VM 3	VM 4
Total Req	10,000	10,000	10,000	10,000
Passed	4,105	3,975	2,406	2,387
Errors	5,895	6,025	7,594	7,613

2) Error Details

Hasil pengujian pada kategori *Error Details* dalam skenario *Transaction History* sebagai berikut:

Table 4-4 Informasi Error Details pada VM 1 (Transaction History)

VM 1	
Type	Count
ExtractHiddenFields	1,000
SocketException	1,000
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	940
WebException	38
No Information	917

Table 4-5 Informasi Error Details pada VM 2 (Transaction History)

VM 2	
Type	Count
ExtractHiddenFields	1,000
SocketException	815
ValidateResponseUrl	1,000
WebTestException	1000

Timeout	965
WebException	1
No Information	1,244

Table 4-6 Informasi Error Details pada VM 3 (Transaction History)

VM 3	
Type	Count
ExtractHiddenFields	1,000
SocketException	1,000
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	1,000
No Information	2,594

Table 4-7 Informasi Error Details pada VM 4 (Transaction History)

VM 4	
Type	Count
ExtractHiddenFields	1,000
SocketException	1,000
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	1,000
WebException	1
No Information	2,612

Informasi detail 'Type':

- ExtractHiddenFields: Redirect page <http://110.35.82.210/PaninWeb/Core/systemUnavailable.html>
- SocketException: Koneksi terputus di tengah pengujian (per Transaction)
- ValidateResponseURL: Hasil respon halaman selanjutnya tidak sesuai
- WebTestException: Respon post parameter data tidak ditemukan

- Timeout: Request Time Out
- WebException: Permintaan (request) dibatalkan

Dari 4 tabel di atas, hasil data *errors* pada skenario *Trasaction History* ini pada setiap *Virtual Machines* hampir memiliki nilai *errors* yang sama, akan tetapi kesalahan itu berasal dari sistem I-banking Bank XYZ semua, bukan berasal dari *Test Rig*. Hasil data *error details* yang dapat direkam oleh Microsoft Visual Studio 2013 maksimal 1000 data.

3) Request Files

Hasil pengujian pada kategori *Request Files* dalam skenario *Transaction History* sebagai berikut:

Table 4-8 Hasil Data Request Files saat Pengujian (Transaction History)

Request (Files)	Total	Failed
IBTransactionHistory.aspx{POST}	120,000	48,284
systemUnavailable.html	38,896	38,896
login.aspx{POST}	80,000	28,260
LandingPage.aspx{POST}	40,000	12,256
Login.aspx	72,252	6,456
ImageHandler.ashx	95,356	592
ScriptResource.axd	12,276	588
ajax-loader.gif	85,688	544
paninStyleNew.css	177,472	532
CombineScripts.axd	85,620	452
CategorizationRules.css	177,472	428
fg-menu.css	177,472	428
PMMActivity.css	177,472	416
Memorization.css	177,472	360
global.css	177,472	348
login.aspx{GET}	40,000	348
Calendar.gif	60,884	256

Core.DateExtensions.js	177,252	256
panin_logo.png	29,648	236
ArrowRight.gif	60,888	232
blank.gif	60,896	228
ArrowLeft.gif	60,816	212
Error.gif	60,836	204
DESGetFiles.aspx	4,080	188
Js	16,260	148
jquery-ui.css	4,144	144
WebResource.axd	8,136	136
TrueStampImageHandler.ashx	27,724	116
LandingPage.aspx{GET}	24,000	108
Logout.aspx	40,000	80
IBTransactionHistory.aspx{GET}	40,000	76
SystemUnavailable.aspx	72	72
timeout.js	95,584	64
GetUnreadMessagesCount	71,744	52
LandingPage.js	47,720	40
coolite.js	47,836	36
BasicDatePicker.js	47,836	20
date-id-ID.js	47,836	12
calendar.gif	24,892	8

Dari table di atas dapat di simpulkan, 5 *request files* yang memiliki *errors* paling banyak pada skenario *Transaction History* (selain request pada file “*systemUnavailable.html*”) disebabkan karena adanya transaksi yang melibatkan banyak perangkat keras (sistem), seperti penggunaan database (*post method*) ketika melakukan login atau mengambil data yang tersimpan dalam *database*.

Penggunaan file .css juga mempunyai banyak *errors* hal ini dapat di sebabkan karena tidak melakukan pengkompresan file, biasanya di sebut *minify css* atau disebabkan karena tidak melakukan teknik *caching* (*cache computing*). Teknik

pengkompresan file dan caching juga berlaku pada file bergambar atau javascript.

4) Test Rig

Hasil pengujian pada kategori *Test Rig* dalam skenario *Transaction History* sebagai berikut:

Table 4-9 Hasil Data *Throughput Rate* Setiap Detik (*Transaction History*)

Throughput Rate / Pages (Per Sec)			
VM1	VM2	VM3	VM4
31	34	27	26

Hasil data *Throughput Rate* pada skenario *Transaction History* paling kecil adalah 26 *pages/sec*. *Throughput Rate* di sini adalah banyaknya *pages* yang terseksekusi ketika melakukan pengujian dalam setiap detik. Semakin besar *Throughput* maka semakin baik.

Table 4-10 Hasil Data Avg. Response Time Setiap Halaman (*Transaction History*)

Avg. Response Time (Sec)			
VM1	VM2	VM3	VM4
4.43	4.29	6.59	6.67

Hasil data pada table di atas terlihat bahwa *response time* setiap halaman pada skenario *Transaction History* paling besar adalah 6.67 *sec/pages*. *Response Time* di sini adalah selisih waktu antara permintaan *request* dengan respon yang di berikan oleh server pada sistem I-banking. Semakin kecil *Response Time* maka semakin baik.

Table 4-11 Hasil Data *Error Rates* dari Pengujian (*Transaction History*)

Error Rates (%)			
VM1	VM2	VM3	VM4
58.95	60.25	75.94	76.13

Hasil data *error rates* ini didapatkan dari table “*error details*” sebelumnya (Table 4.4). Dari data di atas terlihat *error rates* paling besar pada skenario *Transaction History* adalah 76.13%. *Error rates* di sini adalah jumlah transaksi yang gagal selama pengujian berlangsung. Akan tetapi data *errors* di atas, semuanya disebabkan oleh pihak sistem I-banking, yang mana telah dipaparkan pada bagian hasil data “*error details*” sebelumnya. Semakin kecil *errors* maka semakin baik.

Table 4-12 Informasi Kapasitas Memori yang Tersedia Saat Pengujian (*Transaction History*)

Average (Available Memory)	
VM1	1.3GB
VM2	1.3GB
VM3	1.4GB
VM4	1.5GB

Ketika melakukan pengujian, penggunaan *memory* pada *virtual machines* akan semakin besar. Ini disebabkan karena *virtual machines* dalam *test rig* mencoba untuk membuat user-user virtual untuk melakukan pengujian terhadap sistem I-banking. Data di atas merupakan rata-rata jumlah *resource memory* yang tersisa ketika melakukan pengujian pada skenario *Trasaction History*. Selama pengujian penggunaan *resource memory* pada setiap *Virtual Machines* masih tersisa, berarti penggunaan *memory* sebesar 7GB sudah layak untuk melakukan pengujian.

Table 4-13 Informasi Threshold Penggunaan Processor > 75% (*Transaction History*)

Threshold Processor			
VM 1		VM 2	
Times	Warning > 75 %	Times	Warning > 75 %
0:02:30	76.28	0:00:15	97.89
0:03:15	81.73	0:00:15	95.32
0:03:15	76.87	0:00:30	94.71
0:03:45	79.43	0:00:30	97.92

0:04:15	77.67	0:04:15	77.24
0:04:30	87.22	0:04:30	78.18
0:04:30	82.6	0:04:45	81.89
0:04:45	84.23	0:04:45	77.03
0:04:45	81.15	0:05:00	80.1
0:05:45	83.75	0:05:30	91.78
0:05:45	79.74	0:05:30	86.35
0:08:30	78.24	0:05:45	80.62
0:09:00	80.55	0:05:45	85.2
0:09:00	75.84	0:08:45	80.05
0:09:15	80.26	0:09:00	95
0:09:15	75.49	0:09:00	86.19
0:09:30	79.73	0:09:15	80.64
0:09:45	88.69	0:09:15	89.76
0:09:45	84.42	0:09:30	82.36
0:10:00	84.58	0:09:30	75.57
0:10:00	80.99	0:09:45	75.35
0:14:30	77.41	0:09:45	81.97
0:14:45	75.6	0:10:00	90.27
0:15:00	82.87	0:10:00	84.16
0:15:00	77.39	0:10:15	75.86
0:19:45	77.85	0:14:45	79.41
0:20:00	97.2	0:15:15	75.48
0:20:00	93.21	0:20:30	79.33
0:20:15	93.54	0:20:30	75.59
0:20:15	89.06	0:20:45	83.97
		0:20:45	89.55
		0:21:00	84.99
		0:21:00	78.16
		0:25:00	76.05
		0:25:15	88.23
		0:25:15	82.61
		0:25:30	77.05

VM 3		VM 4	
Times	Warning > 75 %	Times	Warning > 75 %
0:10:00	80.11	0:08:00	81.77
0:10:15	85.13	0:08:00	76.25
0:10:15	77.6	0:08:15	80.15
0:10:30	83.01	0:08:15	87.26
0:10:30	78.9	0:08:30	92.13
0:10:45	89.81	0:08:30	87.78
0:10:45	83.57	0:09:15	88.72
0:11:00	90.38	0:09:15	97.59
0:11:00	83.85	0:09:30	96.07
0:11:15	87.26	0:09:30	87.56
0:11:15	79.91	0:09:45	85
		0:09:45	97.28
		0:10:00	98.33
		0:10:00	88.75
		0:10:15	88.54
		0:10:15	97.17
		0:10:30	97.92
		0:10:30	91.44
		0:10:30	9.998
		0:10:45	92.98
		0:10:45	96.25
		0:11:15	79.13
		0:11:15	75.03
		0:11:30	75.1
		0:12:00	78.93

Ketika melakukan pengujian, sama halnya penggunaan *processor* pada *virtual machines* akan semakin besar juga. Ini disebabkan karena *virtual machines* dalam *test rig* mencoba untuk membuat user-user virtual untuk melakukan pengujian terhadap sistem I-banking. Pada pengujian ini, terpasang peringatan (warning) yang akan di rekam oleh *test rig* ketika penggunaan *resource processor*

lebih dari 75%. Data di atas merupakan informasi penggunaan resource *processor* ketika melebihi ambang batas 75% dalam melakukan pengujian pada skenario *Trasaction History*. Selama pengujian penggunaan resource *processor* pada setiap *Virtual Machines* tidak pernah mencapai nilai 100%, sehingga penggunaan 4 *cores* pada *Virtual Machines* dalam Microsoft Azure sudah cukup untuk melakukan pengujian.

4.3.2. Skenario Balance Inquiry

Ada 5 kategori hasil pengujian utama yang akan didapatkan dari Visual Studio, di antaranya: (*Test Result*, *Error Details*, *Request Files*, *Pages* dan *Test Rig*). Berikut ini hasil data yang didapatkan ketika melakukan pengujian (*load testing*) pada skenario *Balance Inquiry*:

Test Run Information:

- Scenario : Balance Inquiry
- Think Time : 10 seconds
- Load Test Type : *Conccurent Load*
- Total Users : 4000 users (1000 users / VM)
- Duration : 20 minutes
- Start Time : 5:35:35 PM

1) Test Result

Hasil pengujian pada kategori *Test Result* dalam skenario *Balance Inquiry* sebagai berikut:

Table 4-14 Hasil Data Load Testing (Balance Inquiry)

	VM 1	VM 2	VM 3	VM 4
Total Req	5,642	5,966	5,041	5,803
Passed	418	699	446	326
Errors	5,224	5,267	4,595	5,477

2) Error Details

Hasil pengujian pada kategori *Error Details* dalam skenario *Balance Inquiry* sebagai berikut:

Table 4-15 Informasi Error Details pada VM 1 (Balance Inquiry)

VM 1	
Type	Count
ExtractHiddenFields	1,000
SocketException	1,000
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	325
WebException	5
No Information	894

Table 4-16 Informasi Error Details pada VM 2 (Balance Inquiry)

VM 2	
Type	Count
ExtractHiddenFields	1,000
SocketException	834
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	292
WebException	4
No Information	1,137

Table 4-17 Informasi Error Details pada VM 3 (Balance Inquiry)

VM 3	
Type	Count
ExtractHiddenFields	1,000
SocketException	1,000
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	423

WebException	4
No Information	168

Table 4-18 Informasi Error Details pada VM 4 (Balance Inquiry)

VM 4	
Type	Count
ExtractHiddenFields	1,000
SocketException	905
ValidateResponseUrl	1,000
WebTestException	1,000
Timeout	465
WebException	14
No Information	1

Informasi detail ‘Type’:

- ExtractHiddenFields: Redirect page <http://110.35.82.210/PaninWeb/Core/system/Unavailable.html>
- SocketException: Koneksi terputus di tengah pengujian (per Transaction)
- ValidateResponseURL: Hasil respon halaman selanjutnya tidak sesuai
- WebTestException: Respon post parameter data tidak ditemukan
- Timeout: Request Time Out
- WebException: Permintaan (request) dibatalkan

Dari 4 tabel di atas, hasil data *errors* pada skenario *Balance Inquiry* ini pada setiap *Virtual Machines* hampir memiliki nilai *errors* yang sama, akan tetapi kesalahan itu berasal dari sistem I-banking Bank XYZ semua, bukan berasal dari *Test Rig*. Hasil data *error details* yang dapat direkam oleh Microsoft Visual Studio 2013 maksimal 1000 data.

3) Request Files

Hasil pengujian pada kategori *Request Files* dalam skenario *Balance Inquiry* sebagai berikut:

Table 4-19 Hasil Data Request Files saat Pengujian (Balance Inquiry)

Request (Files)	Total	Failed
systemUnavailable.html	8,389	8,389
login.aspx{POST}	12,214	6,486
LandingPage.aspx{POST}	5,955	3,111
Login.aspx	12,498	2,381
ScriptResource.axd	3,224	271
SystemUnavailable.aspx	171	171
panin_logo.png	4,810	133
ajax-loader.gif	11,078	128
CombineScripts.axd	10,835	117
Core.DateExtensions.js	15,386	116
Error.gif	9,274	111
blank.gif	9,274	103
Calendar.gif	9,275	103
ArrowLeft.gif	9,277	99
paninStyleNew.css	15,480	97
ArrowRight.gif	9,277	91
fg-menu.css	15,479	88
PMMActivity.css	15,474	86
CategorizationRules.css	15,485	84
DESGetFiles.aspx	1,061	84
Memorization.css	15,476	80
js	11,264	79
WebResource.axd	1,071	71
global.css	15,502	62
AccountPortfolio.aspx	5,827	11
GetUnreadMessagesCount	3,647	11
login.aspx{GET}	6,532	5
Logout.aspx	5,690	4

timeout.js	5,512	3
ImageHandler.ashx	7,372	2
LandingPage.js	3,695	1
TrueStampImageHandler.ashx	3,359	1
errorNew.png	1,484	0
jquery-ui.css	966	0
LandingPage.aspx(GET)	1,952	0

Dari table di atas dapat disimpulkan, 5 *request files* yang memiliki *errors* paling banyak pada skenario *Balance Inquiry* (selain request pada file “*systemUnavailable.html*”) disebabkan karena adanya transaksi yang melibatkan banyak perangkat keras (sistem), seperti penggunaan database (*post method*) ketika melakukan login atau mengambil data yang tersimpan dalam *database*.

Penggunaan file .css juga mempunyai banyak *errors* hal ini dapat disebabkan karena tidak melakukan pengkompresan file, biasanya disebut *minify css* atau disebabkan karena tidak melakukan teknik *caching (cache computing)*. Teknik pengkompresan file dan *caching* juga berlaku pada file bergambar atau javascript.

4) Test Rig

Hasil pengujian pada kategori *Test Rig* dalam skenario *Balance Inquiry* sebagai berikut:

Table 4-20 Hasil Data Throughput Rate Setiap Detik (Balance Inquiry)

Throughput Rate / Pages (Per Sec)			
VM1	VM2	VM3	VM4
29	31	30	26

Hasil data *Throughput Rate* pada skenario *Balance Inquiry* paling kecil adalah 26 *pages/sec*. *Throughput Rate* di sini adalah banyaknya *pages* yang terseksekusi ketika melakukan pengujian dalam setiap detik. Semakin besar *Throughput* maka semakin baik.

Table 4-21 Hasil Data Avg. Response Time Setiap Halaman (Balance Inquiry)

Avg. Response Time (Sec)			
VM1	VM2	VM3	VM4
6.53	6.29	6.02	6.67

Hasil data pada table di atas terlihat bahwa *response time* setiap halaman pada skenario *Balance Inquiry* paling besar adalah 6.67 *sec/pages*. *Response Time* di sini adalah selisih waktu antara permintaan *request* dengan respon yang diberikan oleh server pada sistem I-banking. Semakin kecil *Response Time* maka semakin baik.

Table 4-22 Hasil Data Error Rates dari Pengujian (Balance Inquiry)

Error Rates (%)			
VM1	VM2	VM3	VM4
92.59	88.28	91.15	94.38

Hasil data *error rates* ini didapatkan dari table “*error details*” sebelumnya (Table 4.20). Dari data di atas terlihat *error rates* paling besar pada skenario *Balance Inquiry* adalah 95.38%. *Error rates* di sini adalah jumlah transaksi yang gagal selama pengujian berlangsung. Akan tetapi data *errors* di atas, semuanya disebabkan oleh pihak sistem I-banking, yang mana telah dipaparkan pada bagian hasil data “*error details*” sebelumnya. Semakin kecil *errors* maka semakin baik.

Table 4-23 Informasi Kapasitas Memori yang Tersedia Saat Pengujian (Balance Inquiry)

Avarage (Available Memory)	
VM1	1.4GB
VM2	1.4GB
VM3	1.4GB
VM4	1.5GB

Ketika melakukan pengujian, penggunaan *memory* pada *virtual machines* akan semakin besar. Ini disebabkan karena *virtual machines* dalam *test rig* mencoba untuk membuat user-user virtual untuk melakukan pengujian terhadap sistem I-banking. Data di atas merupakan rata-rata jumlah *resource memory* yang tersisa ketika melakukan pengujian pada skenario *Balance Inquiry*. Selama pengujian penggunaan *resource memory* pada setiap *Virtual Machines* masih tersisa, berarti penggunaan *memory* sebesar 7GB sudah cukup untuk melakukan pengujian.

Table 4-24 Informasi Threshold Penggunaan Processor > 75% (Balance Inquiry)

Threshold Processor			
VM 1		VM 2	
Times	Warning > 75 %	Times	Warning > 75 %
0:00:15	93.65	0:00:15	87.37
0:00:15	90.1	0:00:15	89.42
0:00:30	97.08	0:00:30	96.86
0:00:30	92.56	0:00:30	95.71
0:05:30	81.1	0:00:45	97.74
0:08:15	77.98	0:00:45	95.89
0:08:30	92.07	0:05:15	88.93
0:08:30	87.68	0:05:15	91.03
		0:05:30	79.95
		0:05:30	86.76
		0:05:45	78.59
		0:05:45	85.16
		0:06:00	96.14
		0:06:00	95.83
		0:08:15	89.74
		0:08:15	85.65
		0:08:30	79.93
		0:08:30	76.65

VM 3		VM 4	
Times	Warning > 75 %	Times	Warning > 75 %
0:02:30	77.57	0:00:15	78.01
0:04:15	83.45	0:01:15	83.6
0:04:15	78.38	0:01:15	81.81
0:07:00	34.08	0:05:45	39.33
0:08:45	78.97	0:05:45	97.38
0:08:45	76	0:05:45	95.55
0:09:45	88.64	0:07:00	35.01
0:09:45	84.72	0:09:00	85.31
		0:09:00	82.82
		0:09:45	76.61
		0:10:00	76.14

Ketika melakukan pengujian, sama halnya penggunaan *processor* pada *virtual machines* akan semakin besar juga. Ini disebabkan karena *virtual machines* dalam *test rig* mencoba untuk membuat user-user virtual untuk melakukan pengujian terhadap sistem I-banking. Pada pengujian ini, terpasang peringatan (warning) yang akan di rekam oleh *test rig* ketika penggunaan *resource processor* lebih dari 75%. Data di atas merupakan informasi penggunaan *resource processor* ketika melebihi ambang batas 75% dalam melakukan pengujian pada skenario *Balance Inquiry*. Selama pengujian penggunaan *resource processor* pada setiap *Virtual Machines* tidak pernah mencapai nilai 100%, sehingga penggunaan 4 *cores* pada *Virtual Machines* dalam Microsoft Azure sudah cukup untuk melakukan pengujian.

4.4 Analisi Hasil Pengujian

Berdasarkan hasil data dari pengujian yang telah dilakukan di atas, dapat dijadikan acuan untuk menjawab pertanyaan *Research Question* yang ada di Bab pertama pada bagian perumusan masalah (1.2) yang berisi sebagai berikut:

RQ 1: Bagaimana membangun Test Rig yang optimum untuk dapat melakukan Web Performance dan Load Test pada I-banking Bank XYZ?

Sebagaimana yang telah diterangkan untuk dapat membangun *Test Rig* dalam *Cloud* Microsoft Azure dan dapat melakukan *web performance* dan *load test* diperlukannya melakukan pengaturan/pembangunan pada 4 hal utama, di antaranya: Team Foundation Server Online, *Virtual Machine*, *Test Agent/Test Controller* dan penggunaan *Static IP Address*.

Team Foundation Server Online akan digunakan untuk menyimpan *file testing* yang terdapat dalam Microsoft Visual Studio dan saling berbagi (*sharing*) sesama 4 *Virtual Machine* lainnya.

Virtual Machine yang digunakan saling terhubung dengan jaringan lokal Microsoft Azure dan terhubung dengan Team Foundation Server Online melalui jaringan Internet. Setiap *Virtual Machine* menggunakan sistem operasi Windows Server 2012 R2, terdapat Visual Studio 2013 Ultimate, dan browser Internet Explorer versi 9.0.

Berdasarkan yang sudah dijelaskan pada bagian Bab 3.2.2, digunakannya 4 *Virtual Machines* dengan menggunakan paket A3 (4 *cores processor* dan 7 GB *memory*) pada *Cloud Microsoft Azure*. Karena batas jumlah maksimal *cores processor* yang dapat dipakai dalam 1 *User ID* adalah 20 *cores*. Maka dari itu paket A3 yang akan dipakai untuk melakukan pengujian ini karena penguji/*tester* hanya mempunyai 1 *User ID* dan membutuhkan 4 *Virtual Machines*. Selain itu juga membutuhkan spesifikasi yang tinggi, sehingga performansi VM tidak akan mempengaruhi pada hasil pengujian.

Digunakannya 4 *virtual machines* mempunyai tujuan lain yaitu agar *web server* yang digunakan oleh sistem I-banking akan berkerja/aktif semua. Sehingga mendapatkan hasil pengujian yang maksimal ketika melakukan *stress test* pada sistem I-banking Bank XYZ.

Test Agent dan *Test Controller* terpasang di setiap *Virtual Machine* dan saling terhubung dengan SQL Server Agent di setiap *Virtual Machines* untuk dapat menyimpan hasil data pengujian yang telah dilakukan. Selain itu berfungsi untuk menghasilkan user-user virtual yang seolah-olah dapat

mensimulasikan kebiasaan user dalam melakukan permintaan (*request*) dan mendapatkan respon dari sistem I-banking pada tahap pengujian

Hal yang perlu diperhatikan dalam penempatan *Virtual Machine*, harus dekat dengan Web Server sistem I-banking Bank XYZ, sehingga konektifitas jaringan internet tidak akan mempengaruhi hasil pengujian. Pemilihan lokasi *cloud server* yang terdekat adalah pada zona Regional South East Asia - Singapore.

Selain itu setiap *Virtual Machine* harus mempunyai *Static IP Address* yang terdaftar pada *Firewall Load Balancer* sistem I-banking Bank XYZ, sehingga dapat melakukan *multiple/bulking request* dan mengijinkan *multiple cookies* melalui *HTTP Port*.

RQ 2: Bagaimana performansi Test Rig saat melakukan proses pengujian pada I-banking Bank XYZ, berdasarkan penggunaan processor, memory dan parameter evaluasi response time, throughput, error rate?

Dari hasil pengujian (*Load Testing*) pada bagian “*Test Rig*” (Bab 4.3.1) dengan menggunakan 4 *Virtual Machines* dapat disimpulkan sebagai berikut:

Penggunaan *processor* di setiap *Virtual Machine* tidak pernah mencapai 100%, walaupun dalam beberapa waktu *processor* terpakai lebih dari 75% dan Penggunaan *memory* (RAM) masih selalu tersedia lebih dari 1 GB, sehingga penggunaan *hardware* dengan spesifikasi 4 *cores processor* dan 7 GB *memory* RAM sudah layak untuk melakukan pengujian pada sistem I-banking Bank XYZ sesuai dengan yang diharapkan.

Sedangkan berdasarkan parameter evaluasi *response time*, *throughput* dan *error rate* didapatkan hasil sebagai berikut: Besar *response time* pada setiap *Virtual Machines* dalam melakukan pengujian maksimal sebesar 6 detik, dengan (minimal) *throughput* sebanyak 6 *pages* per detik dan *error rates* (terbesar) adalah 94% (akan tetapi semua *errors* ini disebabkan oleh sistem I-banking Bank XYZ, seperti yang dipaparkan dalam Bab 4.3.1 dan 4.3.2 pada bagian *Error Details*), sehingga penggunaan konektifitas jaringan internet pada setiap *Virtual Machines* dengan kecepatan *download* sebesar +289 Mbps dan kecepatan

upload sebesar +-82 Mbps sudah layak untuk melakukan pengujian dan tidak mempengaruhi kualitas hasil pengujian.

RQ 3: Apa saja faktor yang mempengaruhi performansi (bottlenecks) dari web I-banking Bank XYZ?

Dari hasil pengujian (*Load Testing*) pada bagian Bab 4.3.1 dan 4.3.2 terutama pada hasil data “*Error Details*”, “*Request Files*” dan “*Pages*” dapat disimpulkan 5 faktor yang paling mempengaruhi performansi dari web I-banking Bank XYZ adalah:

- a. Server tidak sanggup menerima banyak request (dalam pengujian ini ada 4000 request secara bersamaan dalam target waktu tertentu) akan tetapi proses request sudah sampai pada server, tidak *timeout*. Hal tersebut terlihat lebih dari 4000 pengujian yang di arahkan ke halaman *systemUnavailable.html* atau error page “503”.
- b. Ada lebih dari 8000 request yang terputus dan tidak mendapatkan respon dari server ketika melakukan pengujian. Kasus ini bisa terjadi karena beban server yang tinggi ketika merespon dari permintaan (*request*) sebelumnya atau konektifitas jaringan Internet server terlalu kecil atau kurang cukup untuk menampung semua request dari user-user virtual.
- c. Ada lebih dari 4000 request yang hasil respon parameternya berbeda (*WebTestException Type*) dari hasil *record*. Hal ini disebabkan karena ketika melakukan proses request metode *post*, sistem I-banking tidak memberikan respon parameter yang valid atau bisa saja tidak ada respon sama sekali yang diberikan. Hal tersebut juga dapat terjadi karena terdapat (*bottlenecks*) pada sistem *database* Bank XYZ, karena metode *post* yang dilakukan dalam pengujian ini selalu melibatkan pengambil data pada *database* sistem I-banking Bank XYZ.
- d. Pada bagian hasil data *Request Files* (Tabel 4-8 dan 4-24) ada 5 hal yang paling sering menimbulkan *errors* respon dan semuanya selalu berkaitan dengan metode “POST”, yang melibatkan penggunaan *database* (*post method*). Sehingga dapat disimpulkan terdapat (*bottlenecks*) pada sistem *database* Bank XYZ dalam kasus tersebut.
- e. Selain itu ada banyak file .css, .js dan file gambar yang menimbulkan banyak *errors*, ini bisa

terjadi karena tidak melakukan teknik kompres *files* (*minify*) dan teknik caching (*cache computing*).

V. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Adapun kesimpulan yang dapat ditarik dari hasil penelitian dan pengujian ini adalah sebagai berikut:

1. Hal yang paling utama dalam membangun *Test Rig* dalam *Cloud Microsoft Azure* untuk dapat melakukan *web performance* dan *load testing* pada sistem I-banking Bank XYZ adalah melakukan pengaturan/pembangunan pada: *Team Foundation Server Online*, *4 Virtual Machine*, *Test Agent/Test Controller* dan penggunaan *Static IP Address*. Paket *Virtual Machines* yang di ambil dalam *Microsoft Azure* adalah A3 (4 cores processor dan 7 GB memory). Paket *resource VM* ini cukup untuk dapat membangun 1000 user virtual (per VM), terbukti tidak adanya penggunaan *processor* atau *memory* yang melebihi *threshold* lebih dari 100% ketika melakukan pengujian. Penggunaan *static IP Address* dalam setiap *Virtual Machines* harus dilakukan untuk dapat dimasukan kedalam *whitelist* atau diijinkan untuk menembus *firewall load balancer* sistem I-banking. Terutama untuk melakukan *spamming/bulking request* dan *multiple cookies*. Penempatan *Virtual Machine* juga harus dekat dengan Web Server sistem I-banking Bank XYZ, sehingga konektifitas jaringan internet tidak akan mempengaruhi hasil pengujian.
2. Performansi *Test Rig* dalam melakukan pengujian yang menggunakan *resource 4 cores Processor* dan 7 GB memory dalam setiap *Virtual Machines* untuk membangun 1000 user virtual, dengan kecepatan Internet download +- 289 Mbps / upload +- 82 Mbps dalam mengakses sistem I-banking, sudah layak untuk melakukan pengujian pada sistem I-banking sesuai yang diharapkan. Hal ini terbukti tidak adanya penggunaan *processor* atau *memory* yang melebihi *threshold* lebih dari 100% ketika melakukan pengujian.
3. Hal yang paling banyak menyebabkan *errors* dalam pengujian ini adalah faktor dari server I-

banking, seperti: konektivitas jaringan Internet yang dipakai server I-banking yang tidak dapat menampung semua *request* dari user, *resource server* yang kurang baik (kecil) dan masih banyaknya file-file website yang tidak dikompres (*minify*) atau tidak menggunakan teknik *caching*. Sehingga nilai *errors* dalam pengujian ini sangat besar.

5.2. Saran

Di akhir laporan tugas akhir ini, penulis memberikan saran kepada pihak yang membaca penelitian ini sebagai berikut:

Penggunaan *Test Agent* yang dipasang langsung pada pihak sistem yang akan diuji, seperti penempatan pada *Database server* atau *Web Server* dalam sistem I-banking belum diimplementasikan pada penelitian ini, pengujian yang dibantu dengan pemasangan *Test Agent* pada target pengujian (dalam kasus di sini sistem I-banking) akan dapat menghasilkan parameter hasil uji yang lebih detail, sehingga akan mengetahui lebih dalam faktor yang mempengaruhi performansi (*bottlenecks*) dari sistem tersebut.

REFERENSI

- [1] C. Hewitt, "ORGs for Scalable, Robust, Privacy-Friendly Client Cloud Computing," *IEEE Internet Computing*, 2008.
- [2] T. Arora, "Part 1 - Load Testing In The Cloud," 29 June 2013. [Online]. Available: <http://geekswithblogs.net/TarunArora/archive/2011/11/20/part-1---load-testing-in-the-cloud.aspx>. [Accessed 18 October 2014].
- [3] C. Vecchiola, X. CHU and R. Buyya, "Aneka: a Software Platform for .NET based Cloud Computing," *High Speed and Large Scale Scientific Computing*, 2009.
- [4] Tata Consultancy Services, "Windows Azure - The Cloud Computing Platform," *High Tech*, 2011.
- [5] Azure Team Support, "What is Azure?," Microsoft, 2014. [Online]. Available: <http://azure.microsoft.com/en-us/overview/what-is-azure>. [Accessed 1 October 2014].
- [6] M. Woodside, G. Franks and D. C. Petriu, "The Future of Software Performance Engineering," *Future of Software Engineering*, 2007.
- [7] J. A. Whittaker, "What is Software Testing? And Why is It So Hard," *Software, IEEE*, 2000.
- [8] M. Gousset, B. Keller and M. Woodward, *Professional Application Lifecycle Management with Visual Studio 2012*, Indiana USA: John Wiley & Sons, Inc., 2012.
- [9] E. J. Weyuker, S. Member, IEEE and F. I. Vokolos, "Experience with Performance Testing of Software System: Issues, an Approach, and Case Study," *Software Engineering*, 2000.
- [10] Microsoft Developer Network Support, "Setting Up Test Machines to Run Tests or Collect Data," Microsoft, 2013. [Online]. Available: <http://msdn.microsoft.com/en-us/library/dd293551.aspx>. [Accessed 18 October 2014].
- [11] E. Blanker, M. Woodward, G. Holliday and B. Keller, *Professional Team Foundation Server 2012*, Indianapolis: John Wiley & Sons, Inc., 2013.
- [12] J.-L. David, M. Gousset and E. Gunvaldson, *Professional Team Foundation Server*, Indianapolis: Wiley Publishing, Inc., 2007.
- [13] Microsoft Developer Network, "Setting Up Test Controllers in Lab Environments," Microsoft, 2013. [Online]. Available: <http://msdn.microsoft.com/en-us/library/hh546460.aspx>. [Accessed 24 11 2014].