TENZING™
Managed IT Services

Magento on AWS Performance Case Study

CONTENTS

EXECUTIVE SUMMARY

Tenzing Managed IT services has recently partnered with Amazon Web Services (AWS) enabling Magento Merchants to combine the flexibility and scalability of AWS with the expertise and managed services that Tenzing provides.

Prior to the official launch, Tenzing conducted beta testing of its reference architecture using 'real-world' customer data. Tenzing Managed IT Services partnered with several Magento Solution Partners for assistance in this phase of the Magento on AWS project.

This report details beta phase activities as well as summarizes findings.

1.1 Objective

The objective of the beta phase was to build a functional Magento store based on real-world sources (versus vendor sample data) on the Amazon Web Services (AWS) platform using Tenzing's optimized Magento reference architectures for AWS as well as common AWS components and services. Once built, Tenzing tested the performance of the store to ensure the AWS solution was comparable to other offerings.

The beta phase provided the participating systems integrator with details on how customer sites would be built on the platform, as well as access to review the 'demo' sites. This allowed the Sis the opportunity to recommend updates and refinements to the offering, based on customer needs.

1.2 Scope

This case study includes an outline of the build and testing that was done as part of the beta phase as well as presentation and discussion of results.

1.3 Methodology and Approach

The architecture was devised to leverage and fully exploit AWS services. Systems were built and tested using site and data provided by the SIs on the Magento Enterprise application software.

Both Tenzing and the SI conducted basic functional testing and review. Limited non-functional testing was conducted by Tenzing using Apache Jmeter, Blazemeter.com, and webpagetest.org.

Informal requirements[1] as they relate to traffic were provided by the systems integrator and used as a basis for analysis of non-functional test results.

The following configurations were tested:

- A single application and a single database server
- Two load-balanced application servers and a single active database with failover (active/passive).

1.4 Results

Section 3 of this document details the method used to estimate traffic and load requirements at average and at peak. Section 7 documents the origins of these requirements.

Table 1 details the results of performance testing. Results were the same in both configurations.

Test Detail	Test Result
Ability to satisfy average daily load of ~2500 unique site visitors.	Successful
Ability to satisfy historical peak monthly load of ~125000 unique site visitors.	Successful

Table 1: Results

1.5 Conclusions

Beta results show that a build of the Magento Store on AWS satisfied all outlined objectives and requirements and demonstrated improved performance to the current production environment.

[1] In a questionnaire distributed to the SI, it was indicated that while there are no formal requirements with the end-customer regarding load, traffic, etc., there is an expectation that the site should has been able to satisfy certain traffic. These details are documented in Section 7.

OBJECTIVES

Part of the beta testing phase for Tenzing's Magento on AWS service offering required test of the solution and configuration with 'real-world' data; partnering with Magento Solution Partners allowed Tenzing to accomplish this objective. In addition, this partnership seeks to illustrate the benefits of a public cloud platform as a potential alternative for a managed Magento Optimized hosted environment.

Table 2 summarizes the objectives for the beta test activities.

Objective	Description
1	Environment setup and site load.
2	Functional testing
3	Performance testing & tuning

Table 2: Beta Activities and Objectives

2.1 General Functional Requirements

The following table lists the required elements of the AWS hosted test site:

Requirement	Detail
1151-FR1	All content is available for navigation with no errors encountered.
1151-FR2	Applicable Magento features and functional are available (admin page, Full Page Cache, etc.).
1151-FR3	End-to-end transactions are possible (adding a product to the cart, view cart, checkout)

Table 3: Site Requirements Summary

2.2 General Non-Functional Requirements

In addition to functional requirements, several non-functional items were deemed required elements of an AWS hosted test site:

Requirement	Detail
1151-NFR1	Flexibility: the ability quickly to meet increased capacity and demand - quickly scaling up to meet increased customer traffic and then scaling down when traffic decreases in an automated fashion.
1151-NFR2	Agility: the ability to respond to unforeseen event, such as recovering from physical disasters, etc.
1151-NFR3	Performance: the ability to accomplish work required compared to the time and resources consumed. In this case, response times of the test site should be comparable to the production site.

Table 4: Site Requirements Summary

2.3 Specific Customer Requirements

A few informal requirements were gathered and generated based on discussions and feedback from the systems integrator. These items are detailed below[2]:

Assumption	Detail
1151-CR1	The site must be able to handle at least 2500 unique visitors per day.
1151-CR2	The site must be able to handle approximately 120000 visits per month at peak.

Table 5: Assumptions Summary

[2] See Section 7 for additional details

METHODOLOGY AND APPROACH

The site was configured to use 'small' and 'large' infrastructure definitions that allow alternatives for particular use-cases, which will be outlined in the respective sections.

All test sites utilized the following components:

Name	Function	Version
PHP-APC	PHP Opcode cache	3.1.15-0.3
Magento EE	Commerce Platform	1.12.0.2
Nginx	Web Server	1.4.2
PHP	Middleware	5.3.28-1.5
PHP-FPM	FastCGI Process Manager	5.3.28-1.5
MySQL	Database Server	MySQL 5.6.13 (RDS[3])

Table 6: Software Components

Additional tuning was performed on several components following Magento best practices and available data. The components above were tested with the following tools for purposes outlined below:

Tool	Function
Jmeter	Jmeter was used to test concurrency and response time of the test site under load. Testing with Jmeter utilized a single test node, 'local' to the test Magento site components (within the same AWS Availability Zone, VPC, and subnet).
Blazemeter.com	Blazemeter.com uses Jmeter scripts to simulate multiple users from multiple sources. The same parameters used to test 'locally' were used to test concurrency and load remotely.[4]
Webpagetest.org	As comparisons regarding concurrency and load were not available, Webpagetest.org provided a sample for comparison related to page load times, user experience, etc.

Table 7: Test Tools

3.1 'Small' Test Configuration

The 'small' configuration consists of a minimal number of components that would satisfy smaller, lower traffic sites that may not need high availability. Additionally, testing a minimal number of components provided baseline for core system capabilities prior to adding ancillary components (Elasticache, etc.). A single, more powerful server, or instance, was used in this configuration.

3.1.1 Components

The following components and arrangement comprised the 'small' test site.

Service Name	Service Function	Sizing
CloudWatch	Monitoring	1 minute
EBS	Persistent block storage	25 GB (x1)
EC2	Compute	c34.xlarge (x1)[5]
RDS	Database services	db.m3.large (x1)[6] • 1000 provisioned IOPs[7] • 100 GB storage[8]

[3] See Section 6 for further details on RDS.
[4] Remote testing adds several variables and intermediaries that are not present during local tests. Thus, these tests are meant to be indicators versus definitive or extensive performance numbers, which would require additional resources to attain.
[5] AWS EC2 Instance Types
[6] AWS RDS Instance Types
[7] 1000 is the minimum number of provisioned IOPs available.
[8] 100GB is the minimum required amount of storage to utilize provisioned IOPs.

Route53	DNS services	N/A

Table 8: 'Small' Infrastructure Components

3.2 'Large' Test Configuration

The 'large' configuration consists of an expanded number of components that would satisfy larger, higher traffic sites that also may need high availability options. As load will be distributed, 2 x c32.xlarge[9] instances were used and provided comparable performance to a single c34.xlarge at the same price point and provide availability across 2 physical locations. Fail-over scenarios are tested.

3.2.1 Components

The following components and arrangement comprised the 'large' test site.

Service Name	Service Function	Sizing (count)
Auto Scaling	Facilitates automated horizontal scaling	N/A
CloudFront	Content Delivery Network	N/A
CloudWatch	Monitoring	1 minute
EBS	Persistent block storage	25GB (x1 per EC2 instance)
EC2	Compute	c32.xlarge (x2)[10]
ElastiCache	Caching services (memcached)	m1.medium (x1)
Elastic Load Balancer	Load balancing services	(x1)
RDS	Database services	db.m3.large (x1)[11] • 1000 provisioned IOPs[12] • 100 GB storage[13]
Route53	DNS services	N/A

Table 9: 'Large' Infrastructure Components

[9] Instance number and type can be changed with minimum effort.
[10] AWS EC2 Instance Types
[11] AWS RDS Instance Types
[12] 1000 is the minimum number of provisioned IOPs available.
[13] 100GB is the minimum required amount of storage to utilize provisioned IOPs.

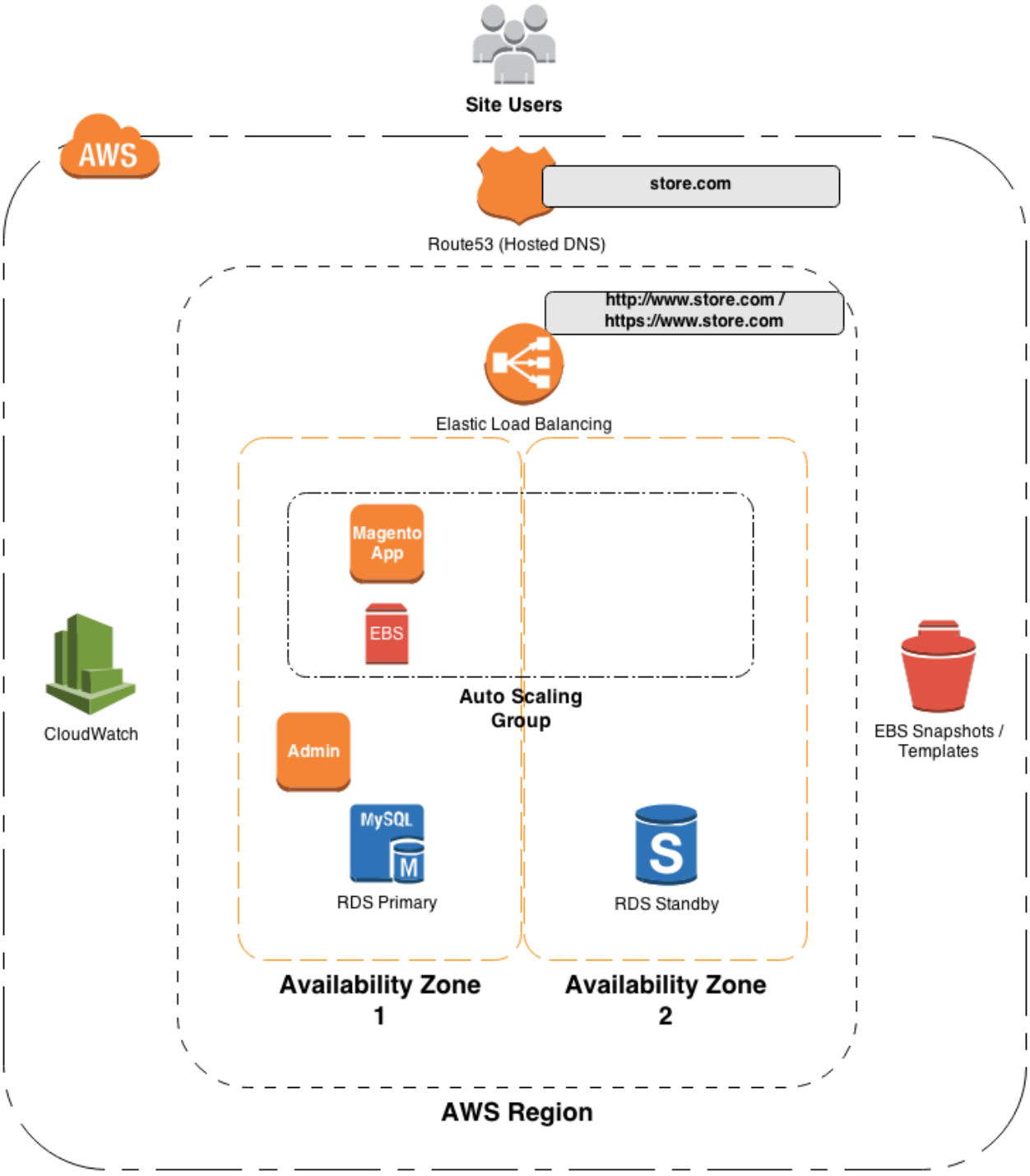


Figure 1: 'Small' Infrastructure Diagram

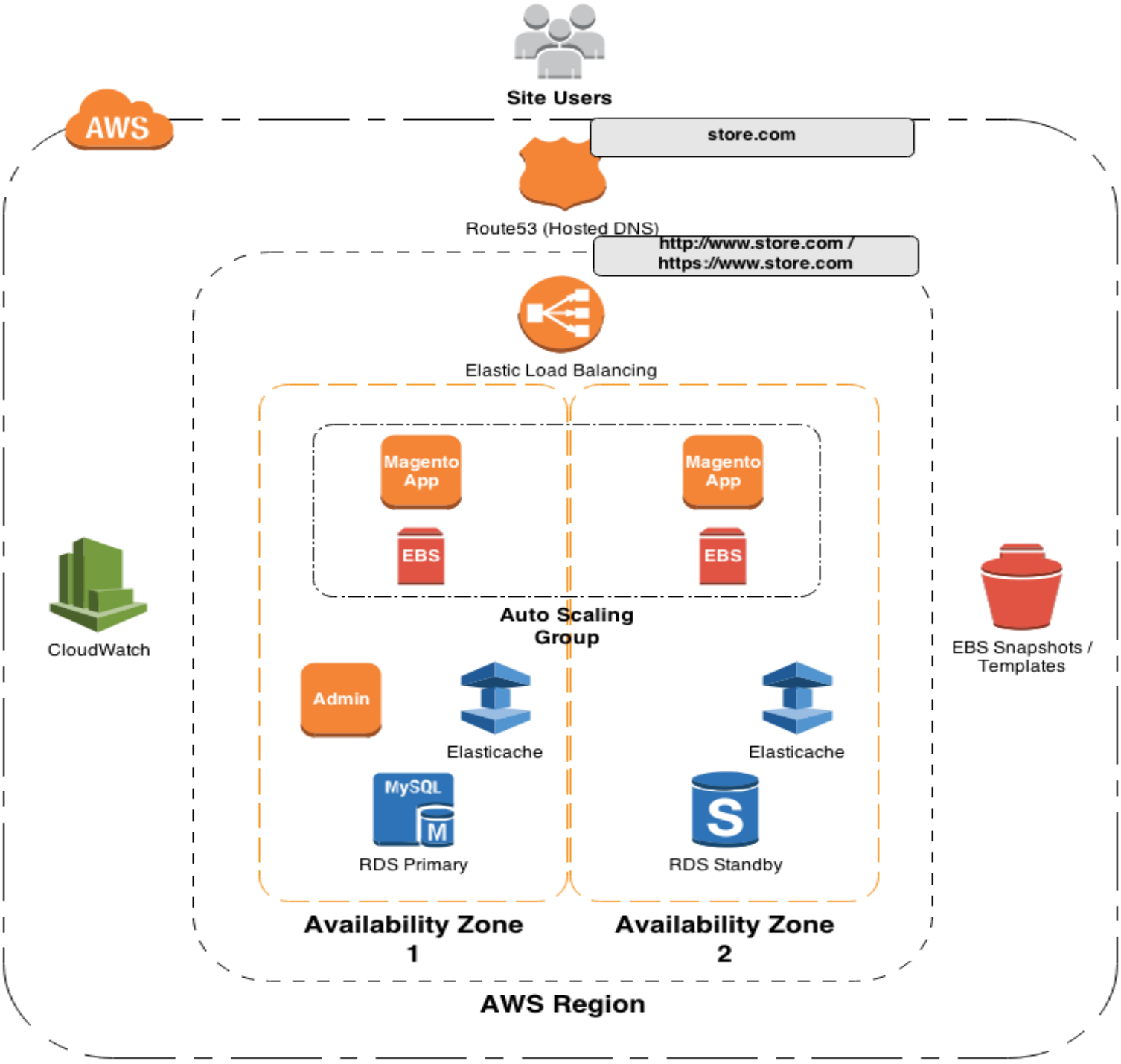


Figure 2: 'Large' Infrastructure Diagram

3.3 Requirements Testing

Requirement	Test Detail
1151-FR1	Manual site navigation: static content, catalog, and product pages.
1151-FR2	Ensure that admin page is accessible and all applicable EE features are enabled. Clear cache directories, browse site, verify that cache directories get populated.
1151-FR3	Add product to cart Checkout as Guest Checkout as logged in user Login to admin and invoice/ship orders

Requirement	Test Detail
1151-NFR1	Create appropriate AutoScaling Group, and Elastic Load Balancer and run test simulation Vertically scale template that includes fully configured customer data
1151-NFR2	Create applicable infrastructure in multiple zones; terminate active nodes to simulate failure and test failover to standby node in separate physical location.
1151-NFR3	Test and collect data on concurrency and response time, comparing results to the production site where possible.

Assumption	Test Detail
1151-CR1	Create and execute tests using Jmeter, Blazemeter.com, or webpagetest.org or a combination thereof based on details in Section 3.4 and Section 7.
1151-CR2	Create and execute tests using Jmeter, Blazemeter.com, or webpagetest.org or a combination thereof based on details in Section 3.4 and Section 7.

Table 10: Site Requirements Summary

3.4 Site Visitor Calculation

The test site is required to handle on average 2,500 visitors per day and 120,000 at peak per month[14]. Given that traffic would not be equal in a 24 period, a more realistic 4 hour distribution period was assumed.

Average Users

Users	Time	
0.173611	60	sec
10.41667	60	min /hour
625	4	hour /day
2500	31	day /month

Peak Users

Users	Time	
0.268817	60	sec
16.12903	60	min /hour
967.7419	4	hour /day
3870.968	31	day /month
120000	1	month /year

Table 11: Average and Peak Users

From the conversion outlined above, as peak users per second are less than 1, for test purposes 1 user per second was used; assuming 1 user per second distributed over a 4-hour period, the following table illustrates the number of users over various lengths of time.

Test Users

Users	Time	
1	60	sec
60	60	min /hour
14400	4	hour /day
446400	31	day /month

Table 12: Average and Peak Test Users

As a result of limited availability of site user behavior data, these simulations simply added 1 user every second for 60 seconds, held the 60 user traffic constant for a 30-60 seconds, and then reduced the number of simulated users; this cycle was then repeated one additional time. Average results for response time and concurrency were recorded.

Several pages were included as part of testing[15]. Of these the first grouping simulated users visiting home and static pages with the next group in the test cycle visiting catalog and product pages.

NOTE: While incorporating checkouts and transactions were not part of the load test scripts, some manual checkouts were performed with no excessive degradation in response time.

[14] See Section 7.4 for details.
[15] See Section 7.1 for details.

RESULTS

The following section summarizes the results of the tests conducted and whether or not requirements detailed in Section 2 were successfully met and discusses additional implications.

Requirement	Detail	Results
1151-FR1	All content is available for navigation with no errors encountered.	Success
1151-FR2	Applicable Magento features and functional are available (admin page, Full Page Cache, etc.).	Success
1151-FR3	End-to-end transactions are possible (adding a product to the cart, view cart, checkout)	Success

Requirement	Detail	Results
1151-NFR1	Flexibility: the ability quickly to meet increased capacity and demand - quickly scaling up to meet increased customer traffic and then scaling down when traffic decreases in an automated fashion.	Success: AWS provides mechanisms such as Auto Scaling to accomplish this requirement. The site was able to scale beyond its original configuration.
1151-NFR2	Agility: the ability to respond to unforeseen event, such as recovering from physical disasters, etc.	Success: Testing of Multiple Availability Zone fail-over was successful with minimal service interruption (within minutes) for the 'Large' configuration.
1151-NFR3	Performance: the ability to accomplish work required compared to the time and resources consumed. In this case, response times of the test site should be comparable to the production site.	Success: Average of total response time with local and remote tests was minimal[16] and testing conducted via websitetest.org was comparable.

Assumption	Detail	Results
1151-CR1	The site must be able to handle at least 2500 unique visitors per day.	Success: Based on current test methodology, both configurations were successful.
1151-CR2	The site must be able to handle approximately 120000 visits per month at peak.	Success: Based on current test methodology, both configurations were successful.

Table 13: Assumptions Summary

[16] See Section 7.2 for full details

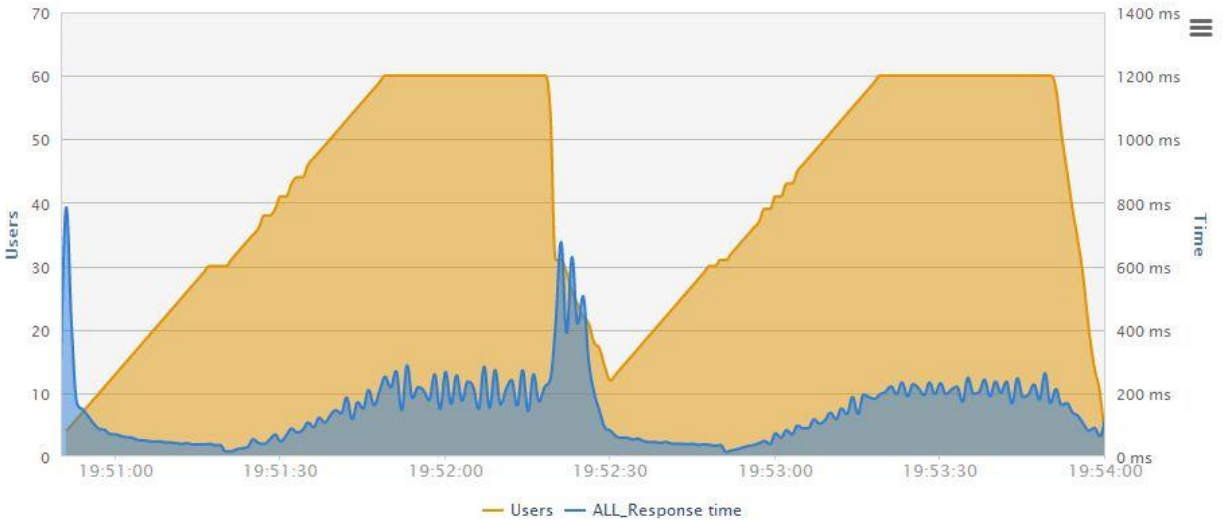


Figure 3: BlazeMeter.com
Average Response Times and User Distribution

Given that load testing and performance data were not available for the production site, online test tools were used for comparison. Below is an example of comparable page load times of the same page from both the production and test sites.

The first two examples are results from the test site. The third example shows results of the current production environment. In both instances, the test site outperformed the current production site.

Web Page Performance Test

Need help improving?

F	A	A	F	F	A	X
First Byte Time	Keep-alive Enabled	Compress Transfer	Compress Images	Progressive JPEGs	Cache static content	Effective use of CDN

From: Dulles, VA - **Chrome - Cable**
4/25/2014 1:10:46 PM

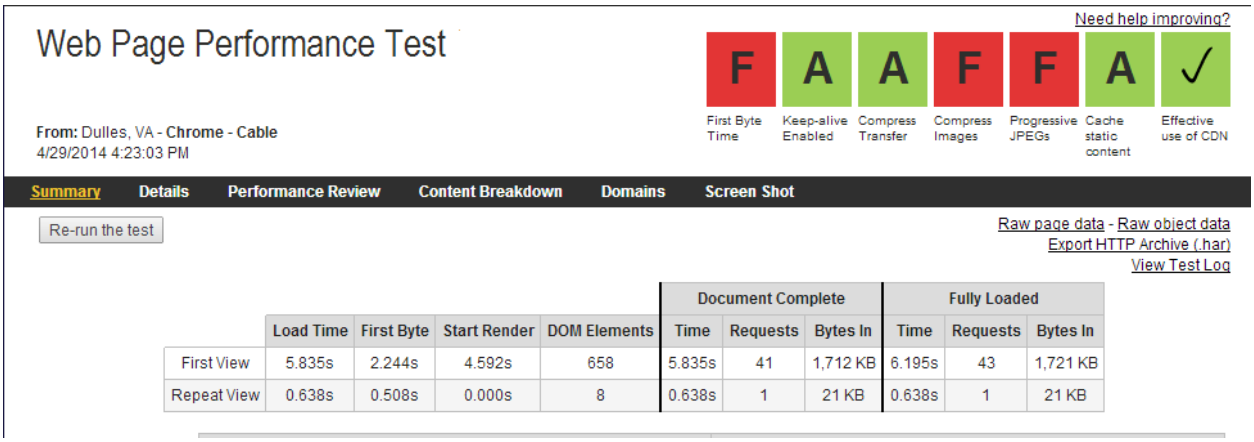
Summary | Details | Performance Review | Content Breakdown | Domains | Screen Shot

Re-run the test

Raw page data - Raw object data
Export HTTP Archive (.har)
View Test Log

						Document Complete			Fully Loaded		
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In
First View	5.180s	2.024s	3.593s	3783	658	5.180s	41	1,704 KB	5.454s	43	1,714 KB
Repeat View	2.521s	1.835s	2.494s	2500	658	2.521s	3	45 KB	2.798s	4	51 KB

Figure 4: webpagetest.org Results:
Test Site



Web Page Performance Test

Need help improving?

First Byte Time	Keep-alive Enabled	Compress Transfer	Compress Images	Progressive JPEGs	Cache static content	Effective use of CDN
F	A	A	F	F	A	✓

From: Dulles, VA - Chrome - Cable
4/29/2014 4:23:03 PM

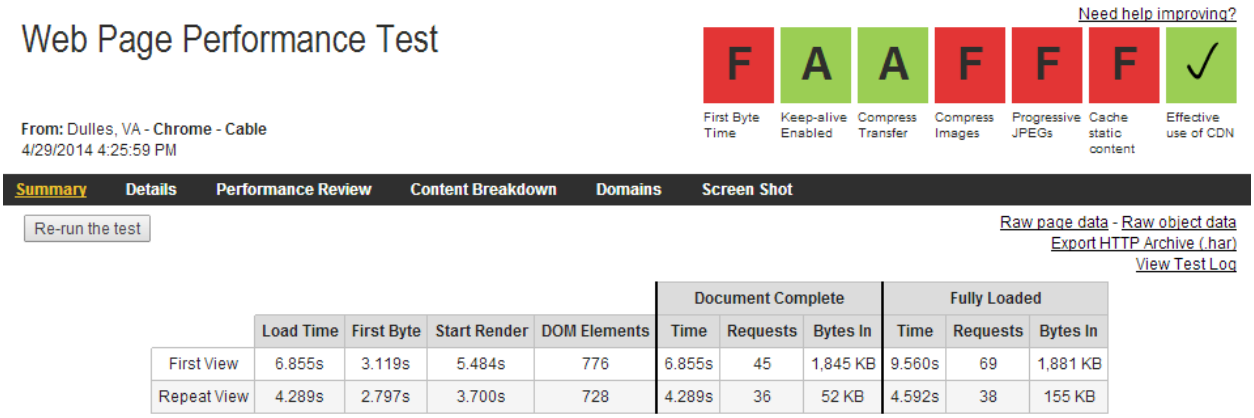
Summary | Details | Performance Review | Content Breakdown | Domains | Screen Shot

Re-run the test

Raw page data - Raw object data
Export HTTP Archive (.har)
View Test Log

					Document Complete			Fully Loaded		
	Load Time	First Byte	Start Render	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In
First View	5.835s	2.244s	4.592s	658	5.835s	41	1,712 KB	6.195s	43	1,721 KB
Repeat View	0.638s	0.508s	0.000s	8	0.638s	1	21 KB	0.638s	1	21 KB

Figure 5: webpagetest.org Results: Test Site [17]



Web Page Performance Test

Need help improving?

First Byte Time	Keep-alive Enabled	Compress Transfer	Compress Images	Progressive JPEGs	Cache static content	Effective use of CDN
F	A	A	F	F	F	✓

From: Dulles, VA - Chrome - Cable
4/29/2014 4:25:59 PM

Summary | Details | Performance Review | Content Breakdown | Domains | Screen Shot

Re-run the test

Raw page data - Raw object data
Export HTTP Archive (.har)
View Test Log

					Document Complete			Fully Loaded		
	Load Time	First Byte	Start Render	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In
First View	6.855s	3.119s	5.484s	776	6.855s	45	1,845 KB	9.560s	69	1,881 KB
Repeat View	4.289s	2.797s	3.700s	728	4.289s	36	52 KB	4.592s	38	155 KB

Figure 6: webpagetest.org Results: Production Site

[17] The AWS CloudFront CDN was used in this sample

4.1 Implications

Section 7.3 illustrates the approximate AWS cost for both small and large configurations used for testing. While both were able to satisfy the requirements as defined in previous sections, a public cloud platform further allows for dynamic and rapid infrastructure changes to accommodate evolving requirements while also introducing unique opportunities to reduce cost.

For instance, Section 7.2 illustrates the acceptable results of the identical load used in all tests on a single c32.xlarge (differing from the single instance testing in Section 4 for the 'small' configuration consisted of a single more powerful c34.xlage instance, while the 'large' configuration used 2 x c32.xlarge instances and distributed load between these generally halving the results shown in Section7.2: Table 14 per instance). In this case a smaller server, or instance type, can be substituted to meet these specific requirements at reduced pricing compared to the alternative. Nevertheless, should requirements change and demand more resources, a larger server, or instance type[18], can be rapidly put into service. Horizontal scaling is also an option where feasible[19]. Further, in Section 7.2: Table 17 illustrates that database resources for the instance type chosen[20] were far from exhausted during testing, even in testing without the use of caching mechanisms such as memcached.

Thus, multiple opportunities exist to reduce cost through the use of resources that can be 'right sized' but still flexible and dynamic. This eliminates the need to 'lock-in' to resources that are sized and priced at estimated peak usage, as is the case with traditional physical infrastructure.

[18] Section 7.2:
Figure 12 illustrates CPU resource usage when scaling from a c32.xlarge (the first peak) to a c34.xlarge (the plateau) under the same load. The final peak illustrates CPU consumption with 1.5x load on a c3.4.xlarge.
Figure 11 and Table 16 also show the corresponding response and transactions per second of the additional 1.5x load on the more powerful c43.xlarge instance.
[19] Magento licensing costs and models will directly affect horizontal scale.
[20] db.m3.large with 1000 provisioned IOPs

CONCLUSION

Based on the tests conducted, the AWS test site has successfully accomplished the established requirements. A build of the Magento Store on AWS satisfied all objectives and requirements and in preliminary comparisons showed better performance than the current production site.

NEXT STEPS

A public cloud platform such as AWS provides a number of alternative configuration options to satisfy dynamic requirements as well as cost-effective options for availability and disaster recovery. Given the positive results of the beta testing Tenzing teams will continue to investigate the potential applications and benefits of AWS infrastructure for Magento merchants.

Tenzing also aims to combine its commerce expertise to further improve performance on AWS. One example would be by incorporating advanced managed services into the monitoring program for AWS.

APM Assure is a current Tenzing product and provides tremendous benefit by breaking down an entire transaction stack into each procedure call. It then orders by method time allowing the reviewer to simply scroll down the list and view anything that may be out of order.

Using APM Assure on the test site, 2 potential performance concerns were revealed:

1. The following procedure may need review due to the number of times the procedure is called. The following seems to indicate that images are being opened and/or saved while just browsing the site.

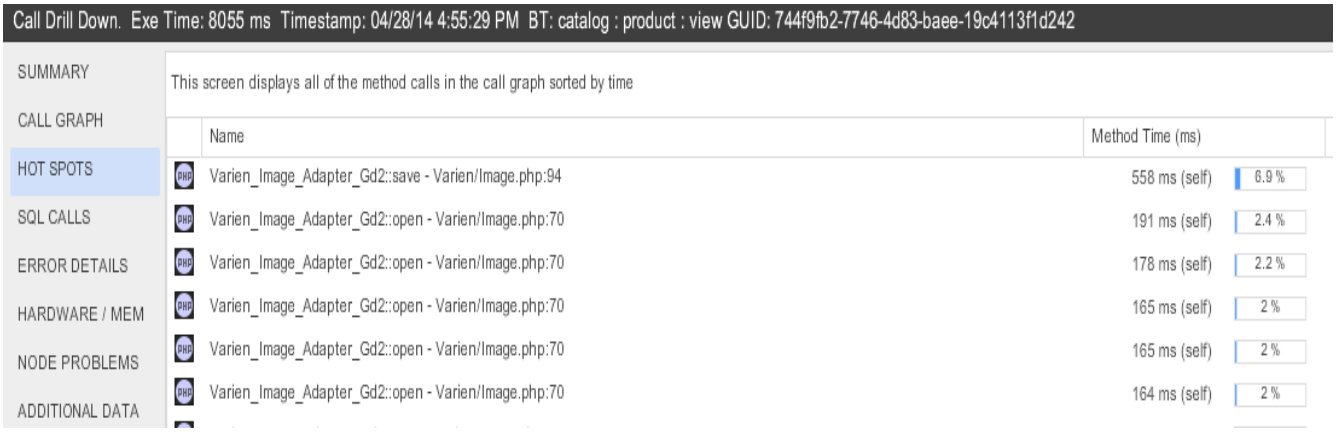


Figure 7: APM Assure Results: 'HOT SPOT' Example 1

NOTE: Upon further investigation, the first 'Hot Spot' disappeared once Compilation was enabled within the Magento Admin site. The second 'Hot Spot' remains.

2. The procedure labelled GetSortAttributes is being called multiple times per transaction and may need optimization.

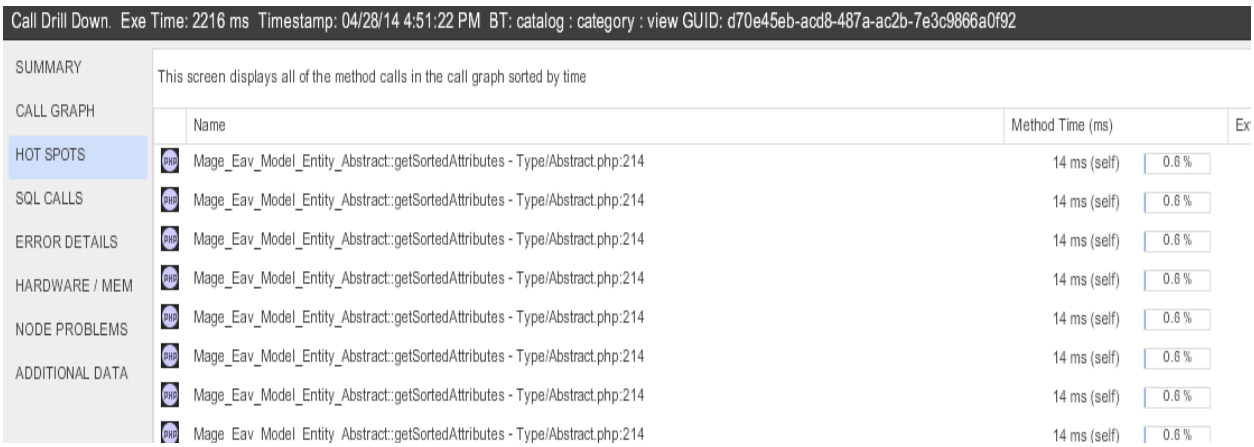


Figure 8: APM Assure Results: 'HOT SPOT' Example 2

While just an example from a test site, this data attempts to demonstrate of the level of visibility and added value that Tenzing can provide.

GLOSSARY

Auto Scaling

Auto Scaling allows scaling of Amazon EC2 capacity up or down automatically according to defined conditions. With Auto Scaling, the number of Amazon EC2 instances increases seamlessly during demand spikes to maintain performance, and decreases automatically during demand lulls to minimize costs.

EBS: Elastic Block Storage

Amazon Elastic Block Store (Amazon EBS) provides persistent block level storage volumes for use with Amazon EC2 instances in the AWS Cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect you from component failure, offering high availability and durability.

EC2: Elastic Cloud Compute

EC2 allows users to rent virtual computers on which to run their own computer applications. EC2 allows scalable deployment of applications by providing a Web service through which a user can boot an Amazon Machine Image to create a virtual machine, which Amazon calls an "instance", containing any software desired. A user can create, launch, and terminate server instances as needed, paying by the hour for active servers, hence the term "elastic". EC2 provides users with control over the geographical location of instances that allows for latency optimization and high levels of redundancy.

ElastiCache

ElastiCache is a web service that makes it easy to deploy, operate, and scale an in-memory cache in the cloud. The service improves the performance of web applications by allowing you to retrieve information from fast, managed, in-memory caches, instead of relying entirely on slower disk-based databases. ElastiCache supports two open-source caching engines: memcached and Redis.

ELB: Elastic Load Balancer

Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables greater levels of fault tolerance in applications, seamlessly providing the required amount of load balancing capacity needed to distribute application traffic. Elastic Load Balancing detects unhealthy instances and automatically reroutes traffic to healthy instances until the unhealthy instances have been restored.

RDS: Relational Database Service

Amazon Relational Database Service is a distributed relational database service by Amazon.com. It is a web service running "in the cloud" and provides a relational database for use in applications. It is aimed at simplifying the setup, operation, and scaling a relational database.

APPENDIX

7.1 Test Results

Label	# Samples	Average	Median	90% Line	Min	Max	Error %
OP Watches	3607	23	19	39	12	343	0.00%
OP /accessories.html	3584	43	19	41	12	2014	0.00%
OP accessories/bags.html	3565	22	18	37	11	347	0.00%
OP Home	1076	23	19	37	11	134	0.00%
OP Culture	1067	597	599	912	229	1316	0.00%
OP Culture /campaign	1049	22	18	37	10	135	0.00%
OP Blog /skate	1031	590	600	873	220	1363	0.00%
TOTAL	14979	108	20	455	10	2014	0.00%

Table 14: Response in Milliseconds
Single c32.xlarge

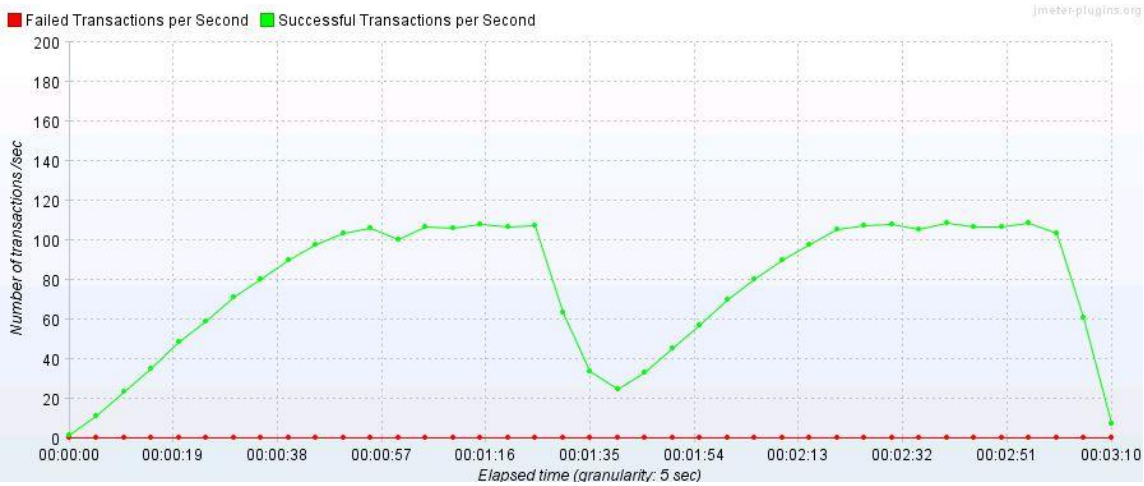


Figure 9: Transactions per second
Single c32.xlarge

Label	# Samples	Average	Median	90% Line	Min	Max	Error %
OP Watches	3702	13	12	16	11	33	0.00%
OP /accessories.html	3683	31	12	16	10	1291	0.00%
OP accessories/bags.html	3663	12	11	15	10	41	0.00%
OP Home	1390	12	12	15	10	28	0.00%
OP Culture	1377	283	280	325	229	424	0.00%
OP Culture /campaign	1358	11	11	14	9	27	0.00%
OP Blog /skate	1346	275	272	311	230	372	0.00%
TOTAL	16519	60	13	269	9	1291	0.00%

Table 15: Response in Milliseconds
Single c34.xlarge

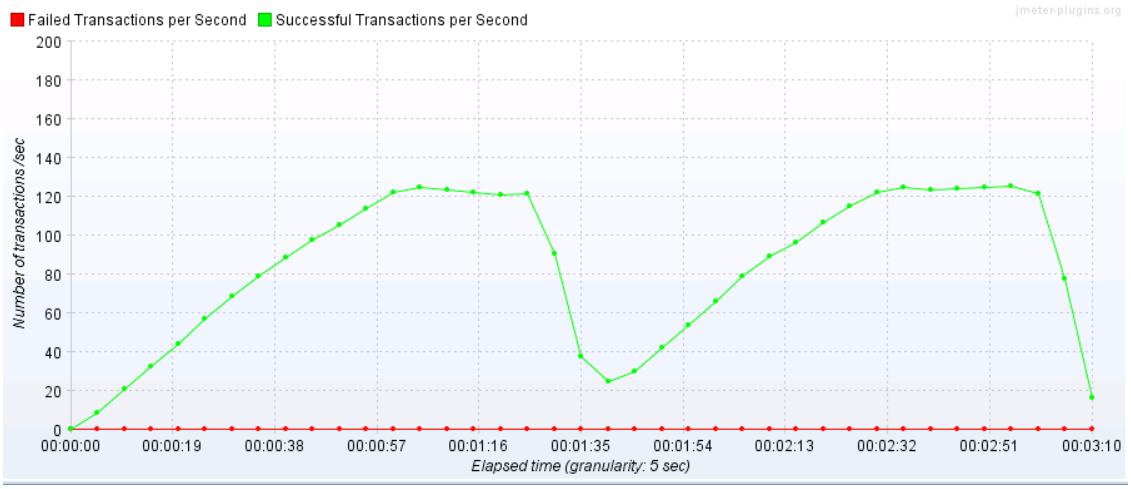


Figure 10: Transactions per second
Single c34.xlarge

Label	# Samples	Average	Median	90% Line	Min	Max	Error %
OP Watches	5535	15	15	18	11	162	0.00%
OP /accessories.html	5504	34	15	18	10	1831	0.00%
OP accessories/bags.html	5472	13	14	16	10	142	0.00%
OP Home	1924	14	14	16	10	145	0.00%
OP Culture	1910	373	343	535	233	767	0.00%
OP Culture /campaign	1880	13	13	15	9	176	0.00%
OP Blog /skate	1868	367	334	532	228	768	0.00%
TOTAL	24093	74	15	317	9	1831	0.00%

Table 16: Response in Milliseconds
Single c34.xlarge (@ 1.5x load)

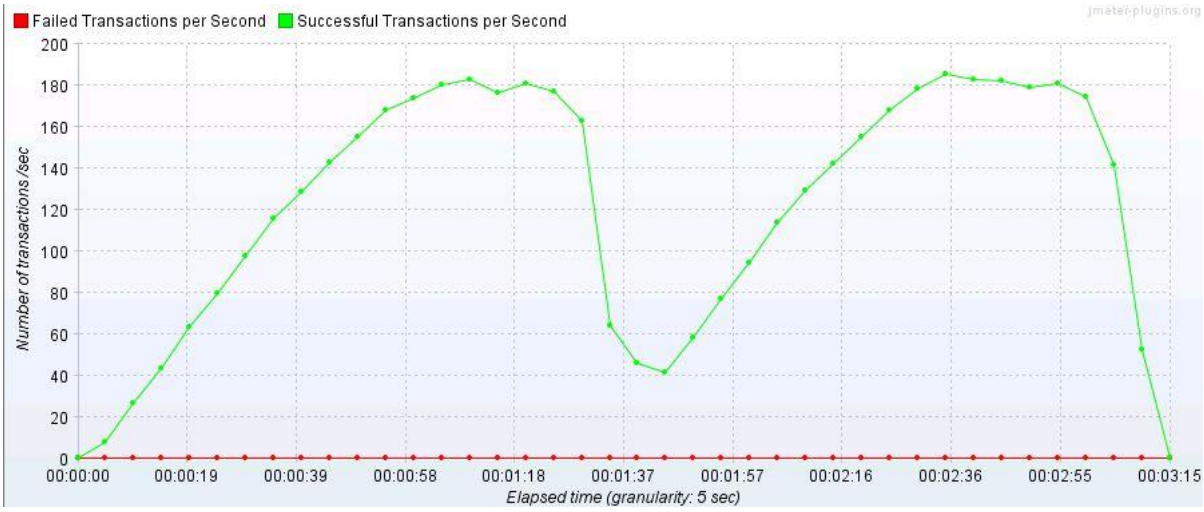


Figure 11: Transactions per second
Single c34.xlarge (@ 1.5x load)

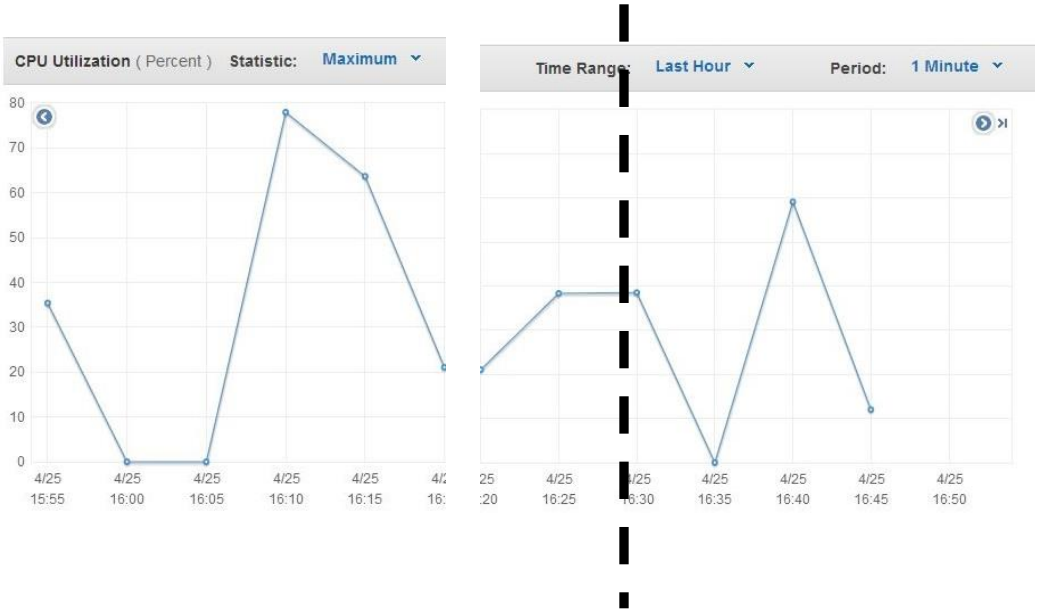


Figure 12: CPU Consumption
(c32.xlarge / c34.xlarge)

Database / RDS Resource Name	Average Used	Maximum Used
CPU	4%	9%
Memory	2GB	2GB
Read IOPs	0.5	1.75
Write IOPs	100	220

Table 17: Database Resource Consumption

7.3 AWS Simple Calculator / Pricing

'Small' Configuration: $1086.21 /month

http://calculator.s3.amazonaws.com/index.html#r=IAD&key=calc-E89794A5-36F9-4FE0-8C44-565D7D0114E8

'Large' Configuration: $1669.59 /month

http://calculator.s3.amazonaws.com/index.html#r=IAD&key=calc-567D80DA-CB5F-4747-9316-4D2E33917811

7.4 Traffic Metrics / Assumptions

Traffic Metrics/ Assumptions
Note: If statistics are not available, please provide any estimates or projections.
How many total unique visitors do you expect the current site can accommodate at peak?
I am unsure, but GA says the site averages about 2,500 unique visitors/day with no issues
How many total transactions / orders do you expect the current site can accommodate at peak?
I am unsure, but GA says the site averages about 40 orders/day with no issues

Figure 14: Excerpt from page 3

Magento Beta Questionnaire.doc