May 2001

TESTING LABS

1001 Aviation Parkway, Suite 400 • Morrisville, NC 27560 • 919-380-2800 • Fax 919-380-2899
320 B Lakeside Drive • Foster City, CA 94404 • 650-513-8000• Fax 650-513-8099
www.etestinglabs.com • etesting_labs_info@ziffdavis.com • 877-619-9259 (toll free)

Internet Appliance INTERNET*pro*® Enterprise Stack™: Performance & failover testing

Test report prepared under contract from Internet Appliance

Executive summary

Internet Appliance commissioned eTesting Labs to conduct performance, scalability, and failover tests of the Internet Appliance INTERNET*pro* Enterprise Stack. The tests we conducted measured the maximum request per second, throughput, and latency performance of 6 Enterprise Stack configurations. eTesting Labs used the standard static and e-commerce test suites supplied with the 128-bit (US) version of WebBench 4.0.1 to performance test the Enterprise Stack configurations. Each test was run twice to verify repeatability. The results in this report are the average of both test runs. The servers and clients were rebooted after each test.

The Enterprise Stack is a completely self-contained redundant and scalable web server solution. The INTERNETpro Enterprise Stack consists of 3 components, a management component, a load balancing component, and a web server component. The management component, the INTERNETpro Enterprise Management Stack (or EMS) is the central controller for configuration, management, and monitoring of the Enterprise Stack. The EMS allowed us to configure and monitor the Enterprise Stack via a simple and easy to use web based JAVA application (Figure 3). The load balancing component, the INTERNETpro Enterprise Director Stack (or EDS) is the redundant central virtual IP address and access point on which all client requests are received and distributed to the web server components. The web server component, the INTERNETpro Enterprise Portal Stack (or EPS) is the redundant self-updating web server component that responds to the test client's HTTP requests. The self-updating aspect of the EPS is a clever use of the Coda file system, a network distributed file system that allows the Enterprise Stack to easily manage the content on all of the EPS units. The EMS allowed us to configure the EPS to act either as a dedicated Coda server, an Apache web server with a Coda client component, or an Apache web server with a Coda server and client component. For the performance tests we conducted, we changed the number of web servers and the distribution and type of Coda servers and clients.

Figure 1 shows the test results using the standard WebBench 4.01 e-commerce test suites. These results show perfect or near-perfect linear scaling at all test points ranging from 2 through 10 EPS components. Specifically, results generated using the 3 EPS fulfilled 1.5 times more requests per second than the 2 EPS

configuration, the 4 EPS configuration fulfilled 1.99 times more requests per second than the 2 EPS configuration, and the 10 EPS – 2 CPU EDS configuration fulfilled an amazing 4.65 times more requests per second than the 2 EPS configuration.

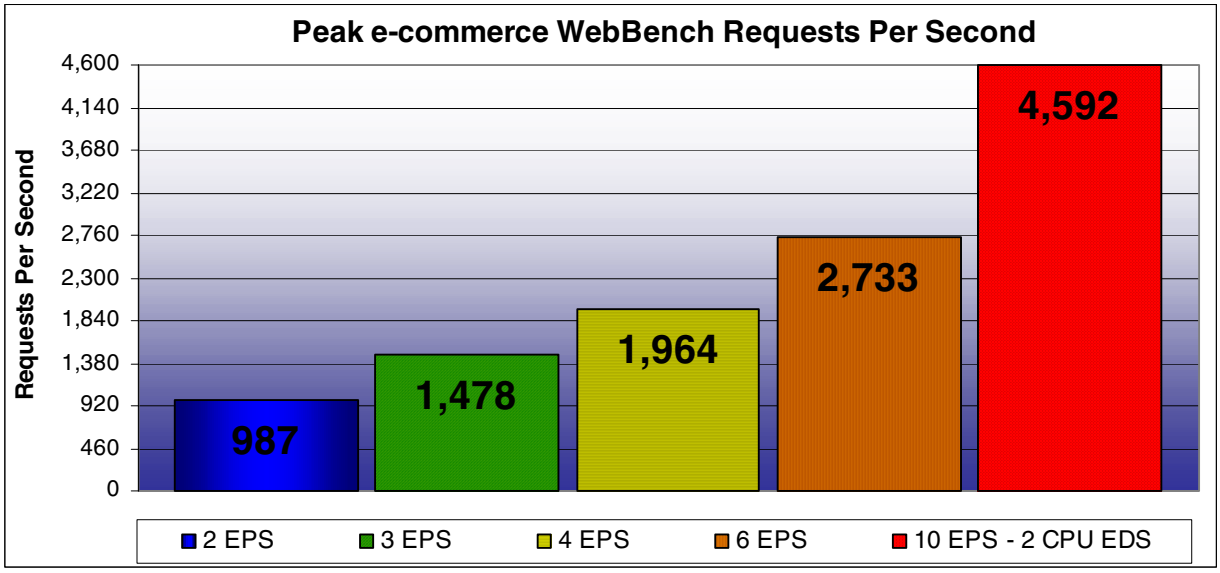

Figure 1: Comparison of peak e-commerce http performance when adding additional Enterprise Stack EPS

Figure 2 shows the test results using the standard WebBench 4.01 static test suites. These results show perfect linear scaling from 2 to 3 and from 3 to 4 EPS. Specifically, the 3 EPS configuration fulfilled 1.49 times more requests per second than the 2 EPS configuration and the 4 EPS configuration fulfilled 2 times more requests per second than the 2 EPS configuration. The 10 EPS – 2 CPU EDS configuration fulfilled 3.19 times more requests per second than the 2 EPS configuration. Although this is less than the 5 times improvement that would make the scaling perfect, this is still a very impressive demonstration of scaling. In addition to outstanding performance and scalability, the EDS failover tests showed that there was virtually no impact on performance. The EPS failover tests showed no unexpected impact on performance, and both exhibited excellent recovery capabilities.

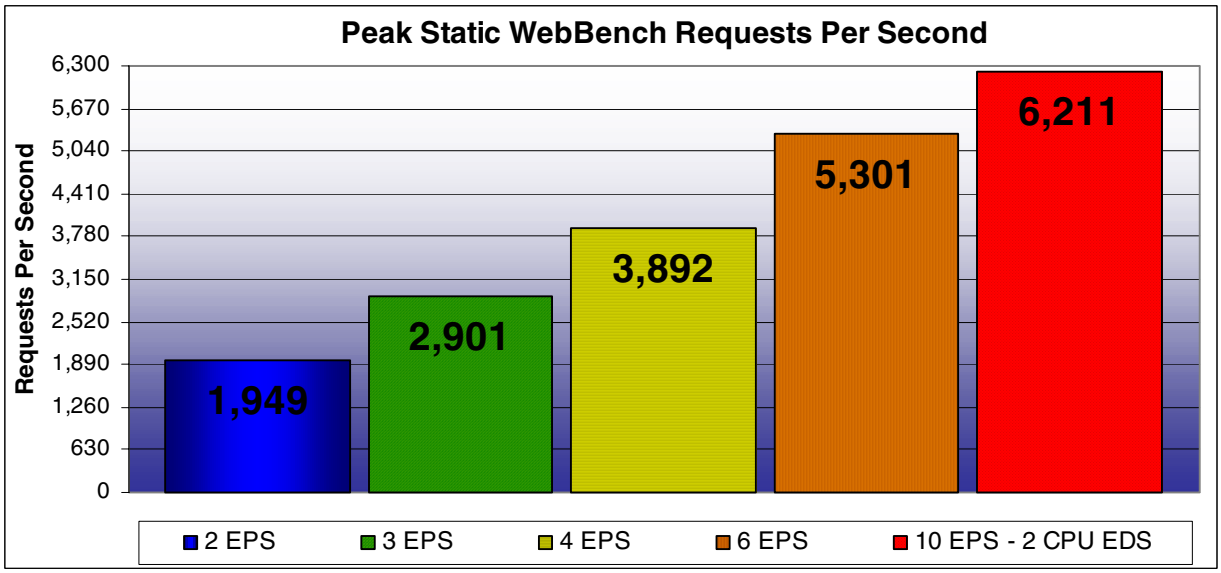

Figure 2: Comparison of peak static http performance when adding additional Enterprise Stack EPS

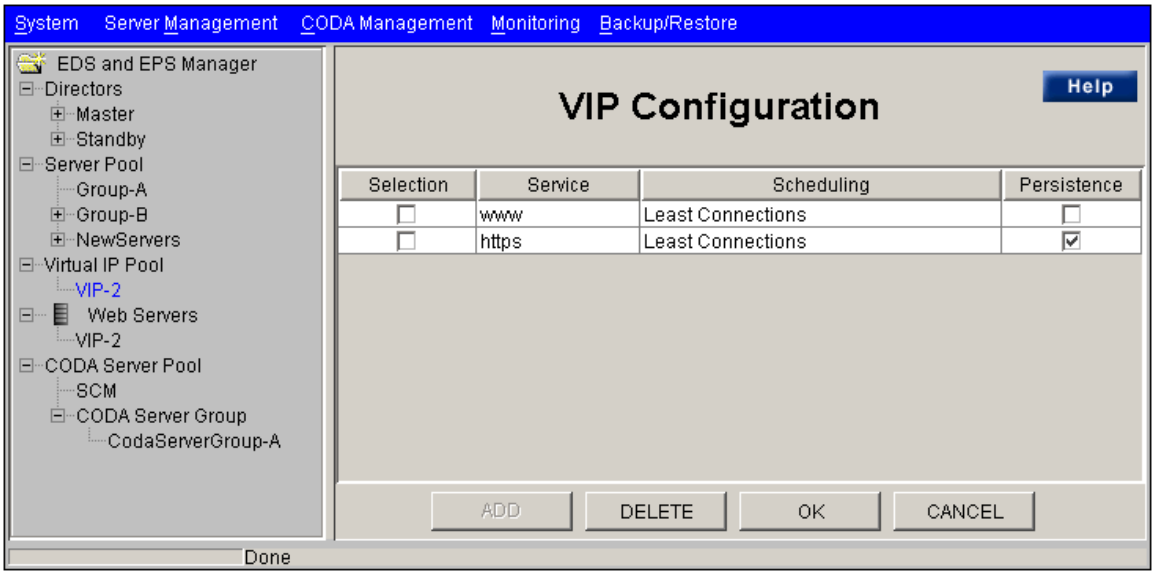

Figure 3: Example of EMS JAVA application.

Testing methodology

At the request of Internet Appliance we tested the following Enterprise Stack configurations:

1. 1 EMS, 2 EDS, 2EPS
2. 1 EMS, 2 EDS, 3EPS
3. 1 EMS, 2 EDS, 4EPS
4. 1 EMS, 2 EDS, 6EPS (2 dedicated Coda servers, 8EPS total)
5. 1 EMS, 2 EDS, 10EPS (2 dedicated Coda servers, 12EPS total)
6. 1 EMS, 2 EDS (1 additional CPU added to the primary EDS), 10EPS

For configurations 1, 2, 3, and 6 we configured two of the EPS components to host the Coda server as well as the Coda client. For configurations 4 and 5 we configured two additional EPS components as dedicated Coda servers. For all configurations we used two EDS, one primary EDS and 1 secondary (or failover) EDS both set to use a least connections load balancing rule. And for configuration 6 only, we added a second CPU to the Primary EDS. Since the second EDS was redundant, it was present only as a backup and remained at an idle but ready state during our performance tests. We used configuration 4 for both the EPS and EDS failover tests.

eTesting Labs used WebBench 4.0.1 (128 bit US version), an industry standard Web server performance benchmark that measures the performance of web server software and hardware by using multiple clients to make HTTP 1.0 GET requests to one or more web servers to simulate and generate Web traffic. For the performance tests in this report we tested with two of the default test suites provided by WebBench, the static suite and the UNIX dynamic e-commerce suite.

The static test suite makes HTTP 1.0 GET requests for purely static content comprised files of various sizes and types including HTML, GIF, and binary. The e-commerce test suite generates a mixture of secure and un-secure static and dynamic HTTP 1.0 GET requests. The dynamic load is generated using a simple CGI script. For the e-commerce suite a small percentage of the requests of both the static and dynamic (CGI) content are made to a secure Web server using the Secure Socket Layer (SSL) protocol.

The standard WebBench suites start with a single client making requests and increment the client load by 4 until the number of clients making requests reaches 60 for a total of 16 different client load points over the

duration of the test. Because the standard 60 clients were not sufficient to stress all server configurations, we created a test suite utilizing 120 clients. This test suite increased the number of clients at each of the 16 different load points in increments of 8 instead of the usual 4. This resulted in significantly increased server load at all stages of the test and allowed us to completely saturate all server configurations tested.

For the failover tests we created a custom WebBench 4.01 test suite based on the results from the static performance testing. This custom test used 4 EPS devices and applied two types of steady loads to the Enterprise Stack; a medium load resulting in EPS CPU utilization of about 50% and a heavy load resulting in EPS CPU utilization of about 90%. We conducted failover testing for two of the three Enterprise Stack components, the EPS (web server component) and the EDS (load balancing component). To create a state of failure, we unplugged a single EPS's network cable and a single EDS's network cable in separate tests. Each test load spanned 7 test mixes. On the 4th mix, we unplugged the network cable of the device being tested . At the start of the 5th mix, we re-connected the network cable. We conducted two test runs for both the performance and failover tests in this report to verify repeatability of the results. The results in this report are an average of both test runs.

Figure 4 shows the basic components and network configuration we used for the tests in this report. We configured the 120 clients with Windows NT Workstation 4.0 Service pack 6. We connected all clients and Enterprise Stack components to a 100Mbps switched network and each of the switches in the test bed via Gigabit. We divided the switch connecting the Enterprise Stack into two simple VLAN's. This allowed us to segregate the HTTP traffic and the administration link for each of the Enterprise Stack components, a requirement for the Enterprise Stack architecture. Detailed information about the Enterprise Stack, web server, client, and WebBench settings can be found in the Appendix of this report.

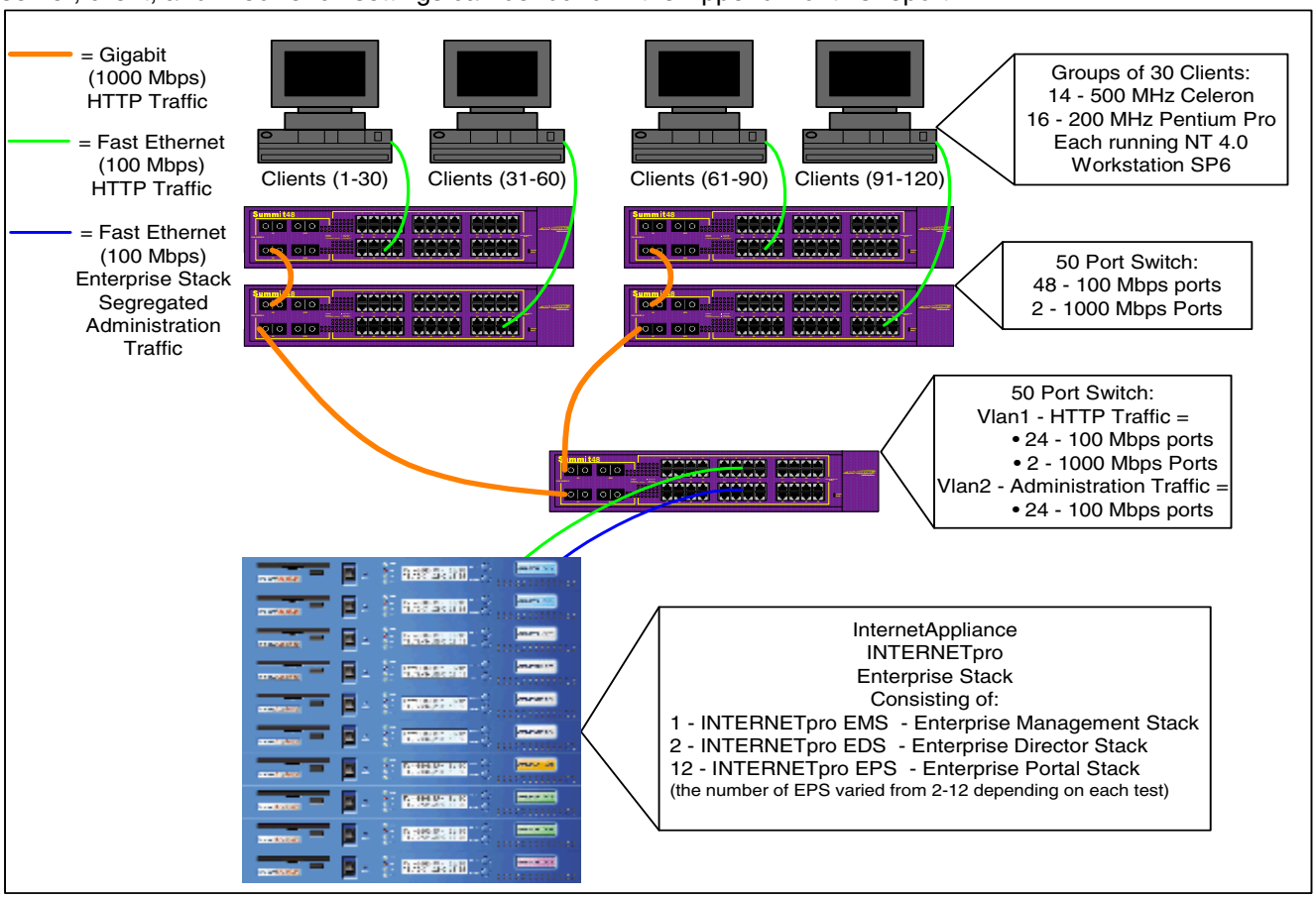

Figure 4: Basic network diagram. Each client and Enterprise Stack had dedicated 100Mbps links to the test network.

Test results

The test results section of this report contains results for performance and failover testing. We conducted performance testing on the following Enterprise Stack configurations:

1. 1 EMS, 2 EDS, 2EPS
2. 1 EMS, 2 EDS, 3EPS
3. 1 EMS, 2 EDS, 4EPS
4. 1 EMS, 2 EDS, 6EPS (2 dedicated Coda servers, 8EPS total)
5. 1 EMS, 2 EDS, 10EPS (2 dedicated Coda servers, 12EPS total)
6. 1 EMS, 2 EDS (1 additional CPU added to the primary EDS), 10EPS

For configurations 1, 2, 3, and 6 we configured two of the EPS components to host the Coda server as well as the Coda client. For configurations 4 and 5 we configured two additional EPS components as dedicated Coda servers. For all configurations we used two EDS, one primary EDS and 1 secondary (or failover) EDS. And for configuration 6 only, we added a second CPU to the EDS. Since the second EDS was redundant, it was present only as a backup and remained at an idle but ready state during our performance tests. We used configuration 3 for both the EPS and EDS failover tests.

The results in this report are organized in three sections; static, e-commerce, and failover results. The static and e-commerce results sections (figures 5 - 22) contain comparison graphs for average requests per second, average throughput, and average response time for all 6 test configurations as well as separate graphs of the CPU utilization for all components of each individual configuration. The failover section (figures 23 - 25) contains comparison graphs for average requests per second, average throughput, and average response time for configuration 3.

Static Results

The static results for the tests we conducted show perfect linear scaling from 2, 3 and 4 EPS for the number of requests per second. Using the maximum static request per second peak values in figure 2, the 3 EPS configuration was able to fulfill 1.49 times more requests per second than the 2 EPS configuration and the 4 EPS configuration was able to fulfill 2 times more requests per second than the 2 EPS configuration. The 10 EPS – 2 CPU EDS configuration was able to fulfill 3.19 times more requests per second than the 2 EPS configuration. Although this is less that the 5 times improvement that would make the scaling perfect, this is still a very impressive demonstration of scaling.

For the 10 EPS – 2 CPU EDS tests we improved the scaling capabilities by adding a second processor to the EDS. If you compare figures 8 – 13 and take note of the EDS1 CPU utilization, you'll notice that eventually the EDS1 becomes processor bound. In figure 8 the EDS1 CPU averages about 50%, and by the 4 EPS test (figure 10) the EDS1 processor quickly reaches 100% utilization. We conducted an additional test with 2 CPUs in the EDS. This test proved useful when compared test case 5, 10 EPS with 2 dedicated Coda servers. We experienced a 1.15 times peak requests per second improvement by simply adding a second CPU to the primary EDS. At the request of Internet Appliance we also tested the effects of using two additional EPS as dedicated Coda servers for the configurations with more EPS components. Our experience was that this configuration was more stable, but it was still possible to use the combined Coda client/server model used in configurations 1, 2, 3, and 6.

In addition to impressive request per second scaling results, the Enterprise Stack delivered 300Mbps (megabits per second) of static HTTP traffic. The peak throughput in bytes per second for configuration 6 was 37498375.69 bytes per second. Using the following formula, ((bytes per second X 8)/1,000,000 bits) we calculated the total throughput to be exactly 300 Mbps. In addition to its high throughput capabilities, the Enterprise Stack exhibited low latency numbers as well. At its worst the Enterprise Stack retuned a quick 135-millisecond response time. This was the worst case for the 2 EPS configuration at the 120-client load. Since all configurations reached maximum requests per second at or before the 32-client test mix, it would be more accurate to use the 32-client mix as the cut-off for the response time. Using the 32-client mix as the cut-off,

the Enterprise Stack responded to requests in under an impressively quick 16 milliseconds. The Enterprise Stack also did an excellent job of maintaining the peak performance over the duration of the test. With the exception of configuration 1 (2 EPS) all tests completed without degradation in requests per second performance even though the load continued to increase. And even the 2 EPS configuration did well up to the 104-client test mix.

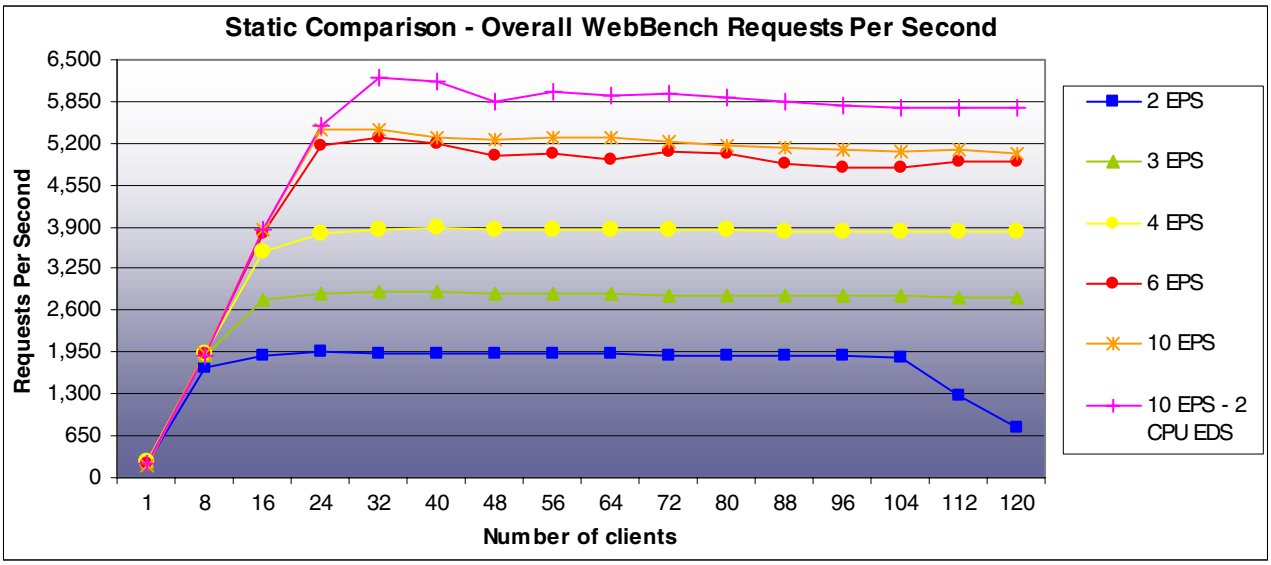

Figure 5: Static load requests per second comparison of the 6 configurations

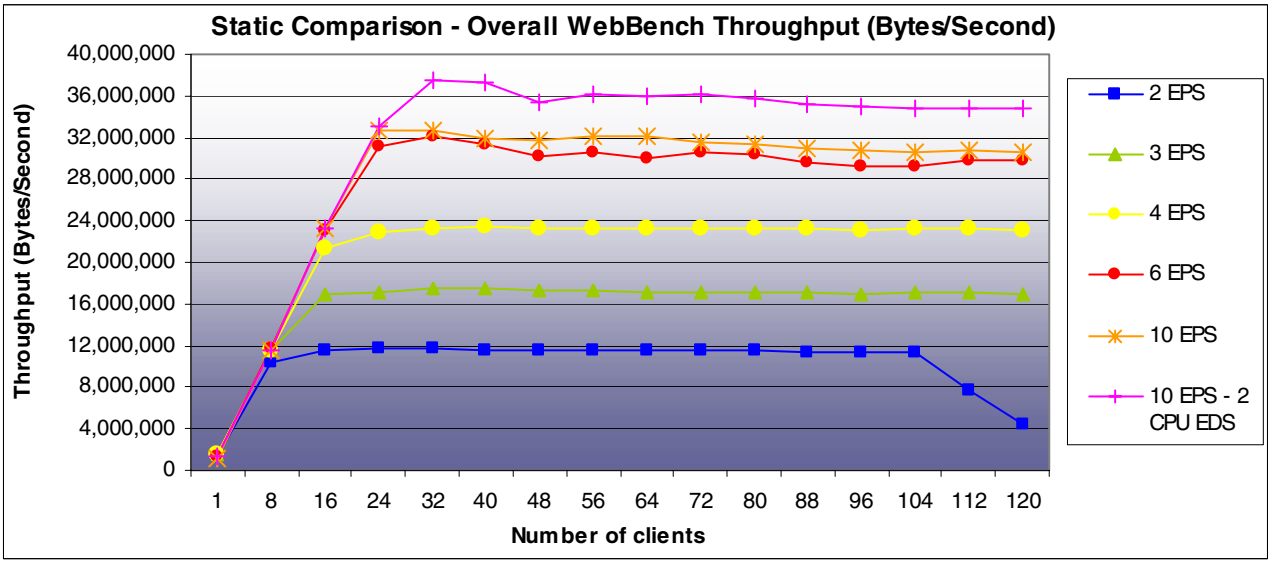

Figure 6: Static load throughput comparison of the 6 configurations

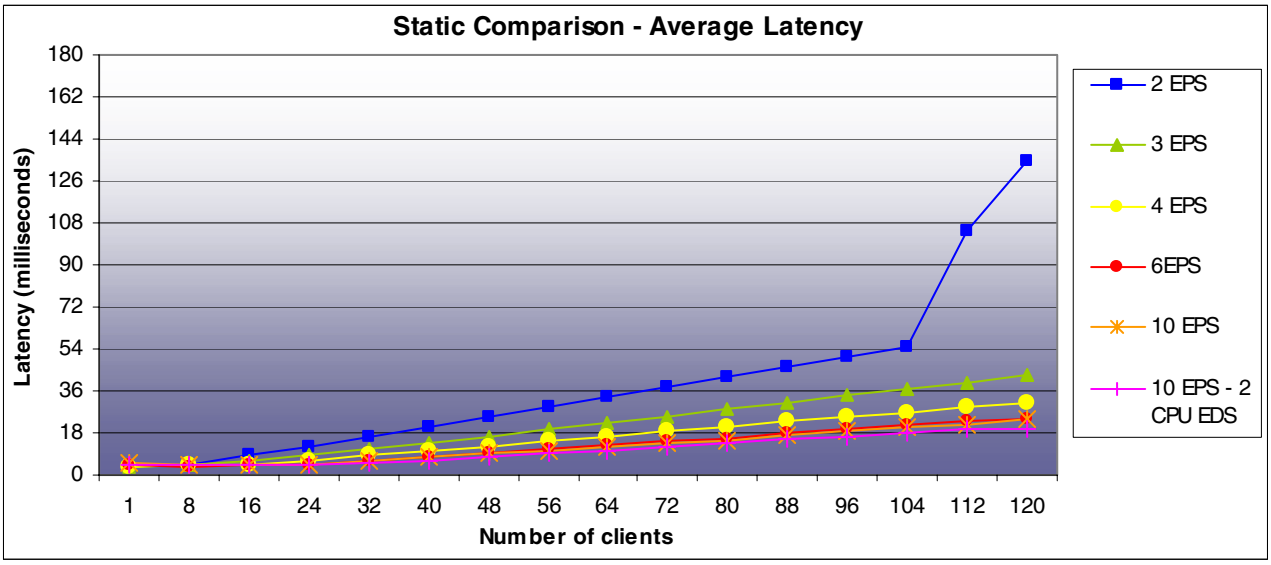

Figure 7: Static load latency comparison of the 6 configurations

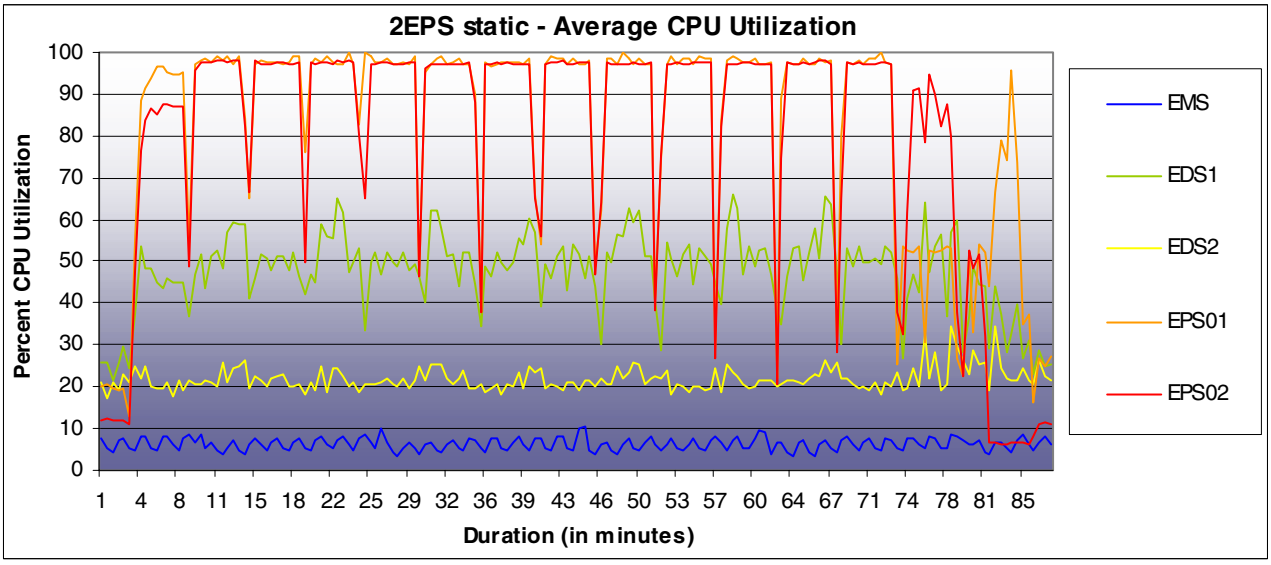

Figure 8: Configuration 1, CPU utilization static results for all Enterprise Stack components

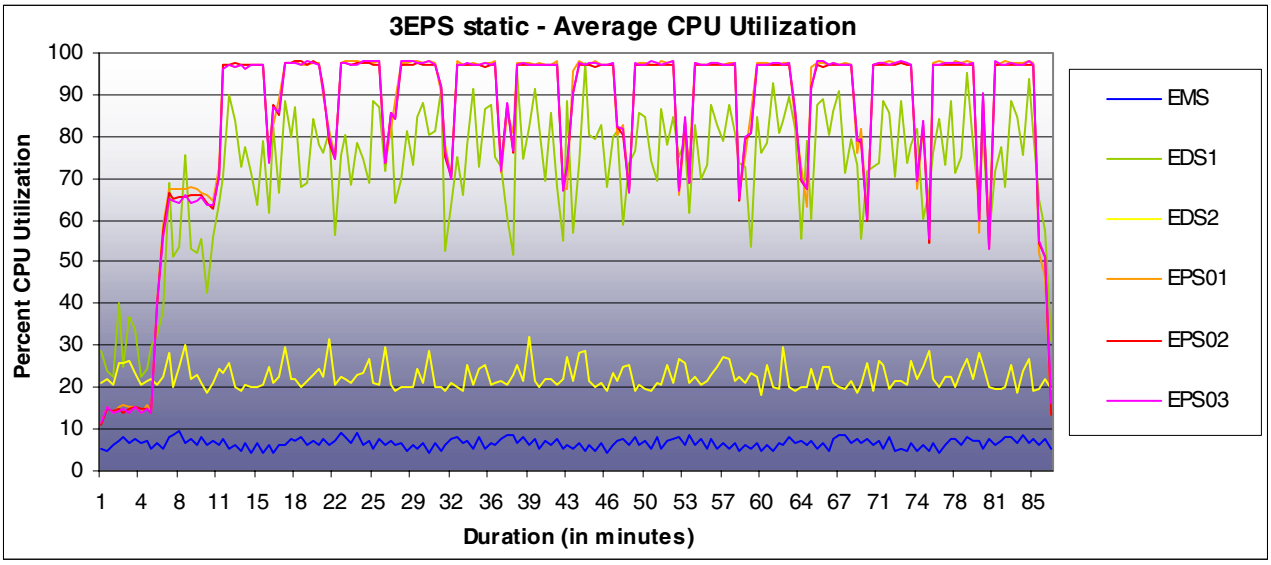

Figure 9: Configuration 2, CPU utilization static results for all Enterprise Stack components

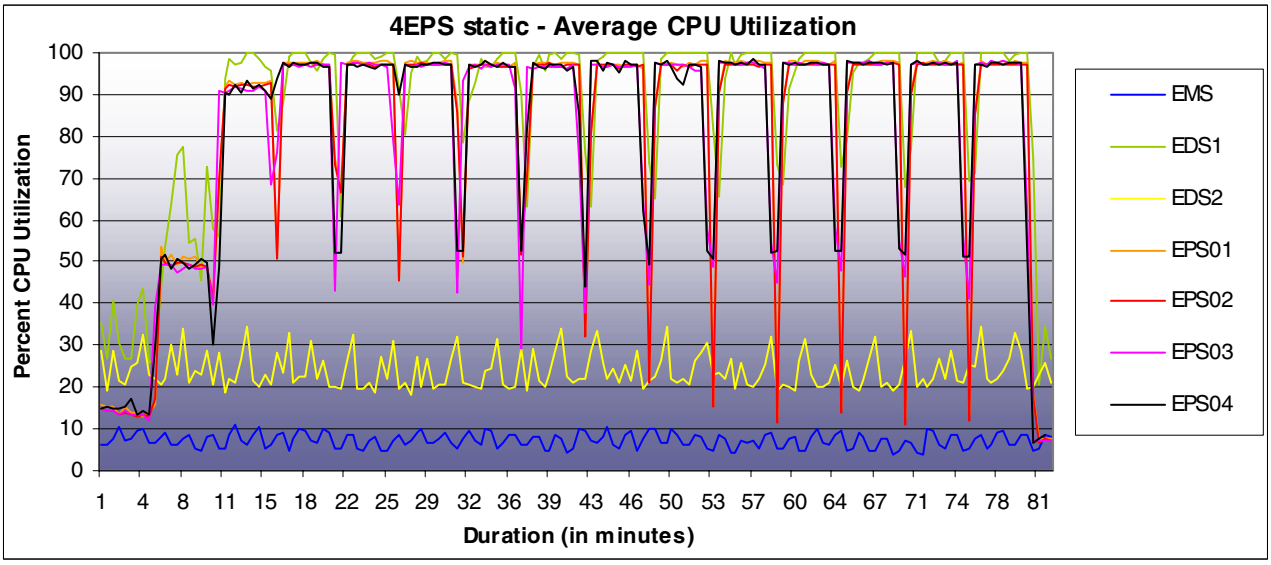

Figure 10: Configuration 3, CPU utilization static results for all Enterprise Stack components

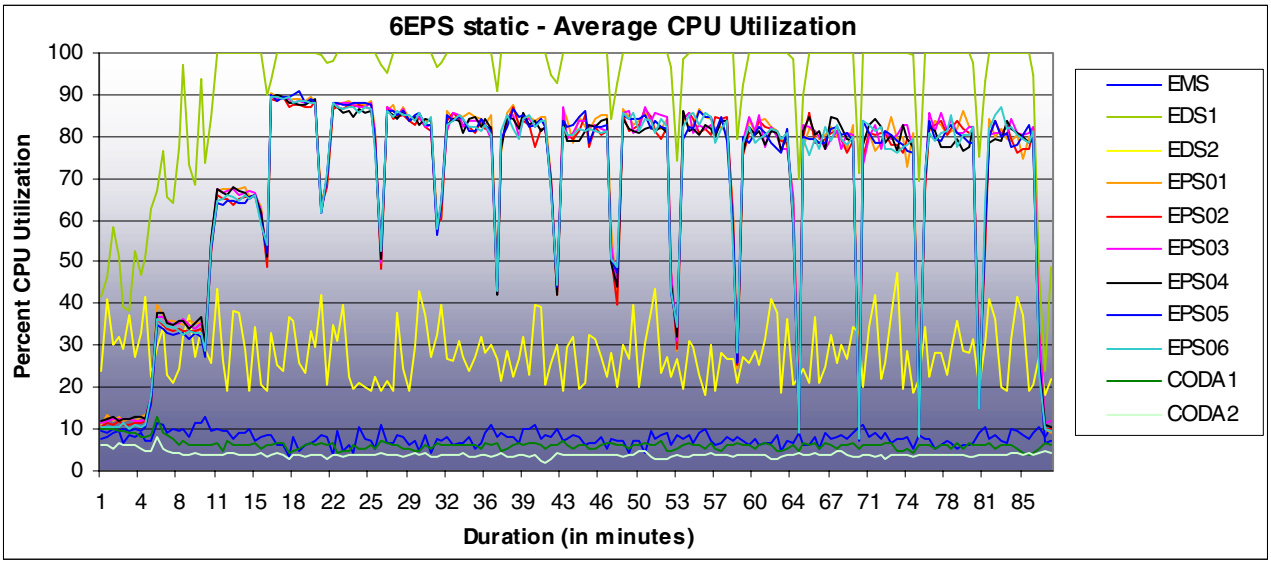

Figure 11: Configuration 4, CPU utilization static results for all Enterprise Stack components

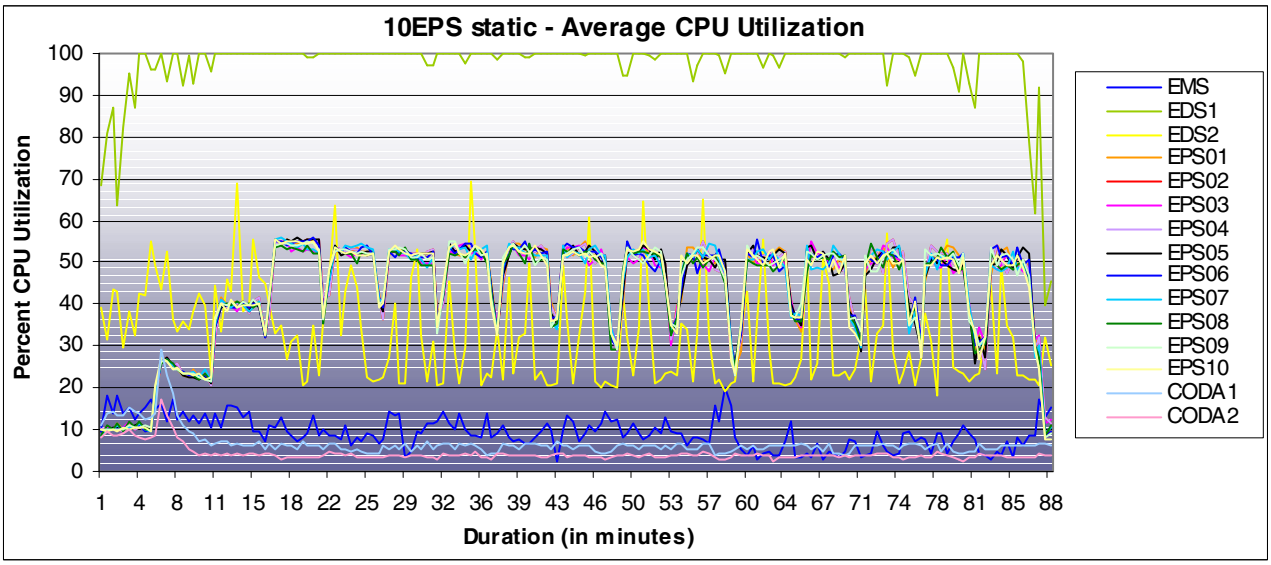

Figure 12: Configuration 5, CPU utilization static results for all Enterprise Stack components

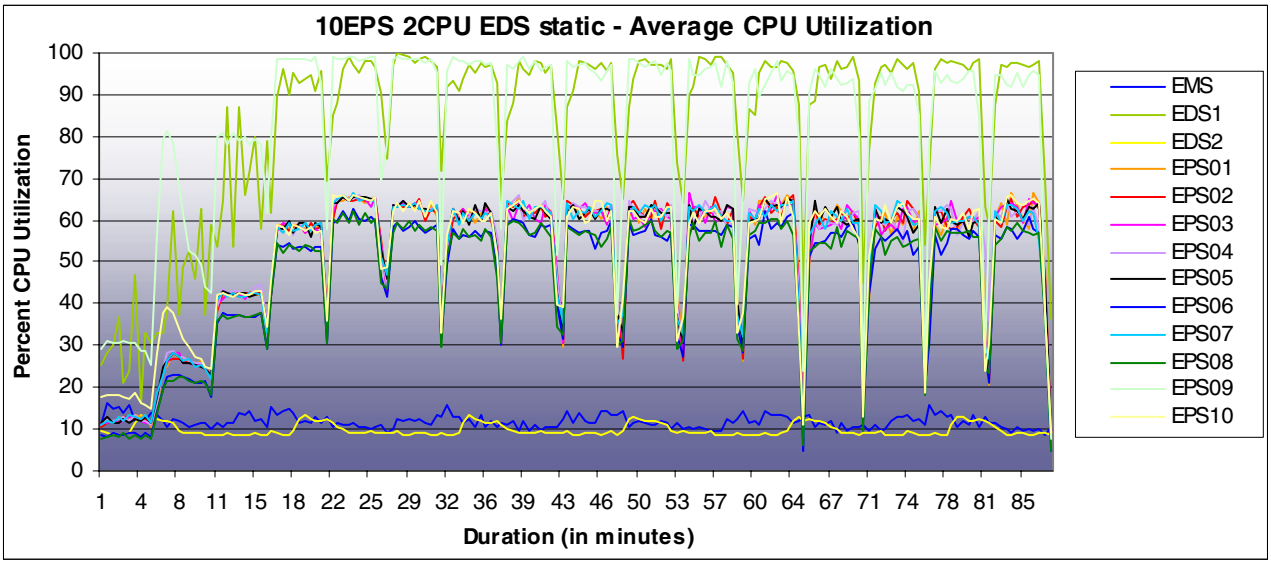

Figure 13: Configuration 6, CPU utilization static results for all Enterprise Stack components

E-commerce Results

Like the static results, the e-commerce results for the tests we conducted show perfect linear scaling from 2, 3 and 4 EPS for the number of requests per second. Using the maximum e-commerce request per second peak values in figure 1, the 3 EPS configuration was able to fulfill 1.5 times more requests per second than the 2 EPS configuration and the 4 EPS configuration was able to fulfill 1.99 times more requests per second than the 2 EPS configuration. The 10 EPS – 2 CPU EDS configuration was able to fulfill an amazing 4.65 times more requests per second than the 2 EPS configuration. The scaling results are better for the e-commerce tests because the SSL and CGI components take more web server processing power.

If you look at the 10 EPS – 2 CPU static results, you can see that the EPS CPU utilization (figure 13) peaks at about 60%. However, the web servers performance does not increase because the EDS has reached its maximum CPU utilization. If you compare this to the e-commerce 10 EPS – 2 CPU EDS CPU utilization chart (figure 22) you will notice that the EDS is not getting as much of a workout and that the EPS CPU utilization has reached 100%. So with this workload we did not see the same improvement in performance with the additional EDS CPU. If you compare the results from configurations 5 and 6, the overall ramp-up is different but the peak performance remains virtually the same. At the request of Internet Appliance we also tested the effects of using two additional EPS as dedicated Coda servers for the configurations by adding more EPS components. Our experience was that this configuration was more stable, but it was still possible to use the combined Coda server/client model used in configurations 1, 2, 3, and 6.

For the e-commerce tests the Enterprise Stack throughput was 150Mbps (megabits per second). This is less than with the static suite because there is less data transferred and more server side processing being conducted with the e-commerce suite. The peak throughput in bytes per second for configuration 6 was 147981118.8 bytes per second. Using the following formula, ((bytes per second X 8)/1,000,000 bits) we have calculated the total throughput to be exactly 150 Mbps. Again the Enterprise stack performed exceptionally well regarding latency. At its worst the Enterprise Stack returned a respectable 210-millisecond response time. This was the worst case for the 2 EPS configuration at the 120-client load. Again, since all configurations reached maximum requests per second at or before the 32-client test mix, it would be more accurate to use the 32-client mix as the cut-off for the response time. Using the 32-client mix as the cut-off, the Enterprise Stack responded to requests in under an impressively quick 33 milliseconds. The Enterprise Stack also did an excellent job of maintaining a high load for a long duration. With the exception of

configuration 1 (2 EPS) all tests completed without degradation in requests per second performance even though the load continued to increase. And even the 2 EPS configuration did well up to the 96-client test mix.

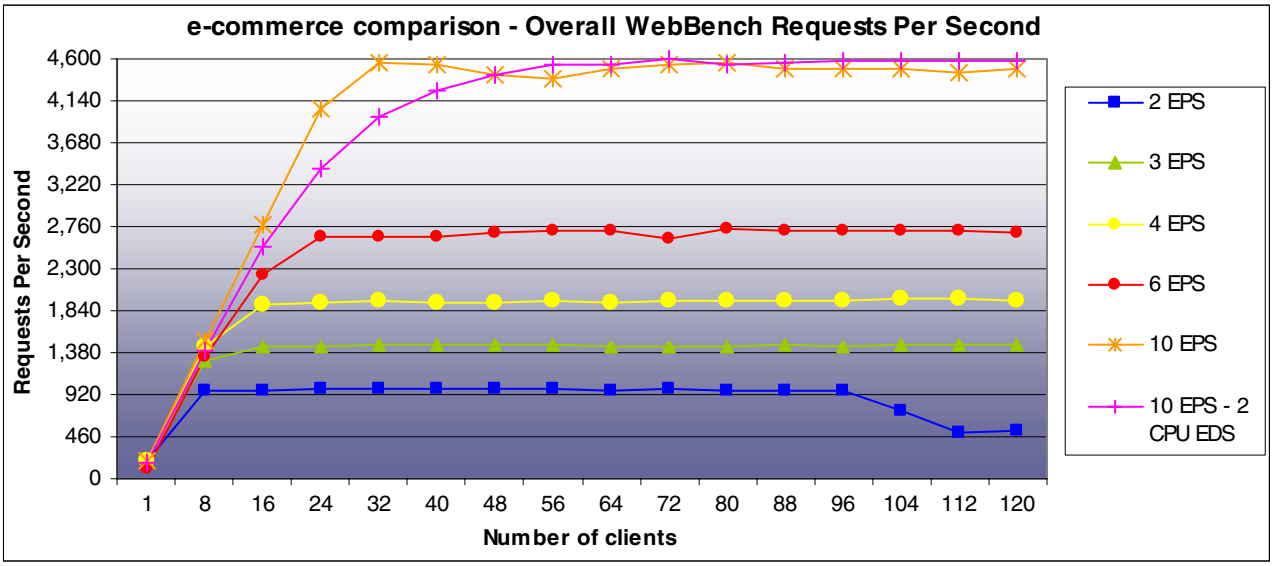

Figure 14: E-commerce load requests per second comparison of the 6 configurations

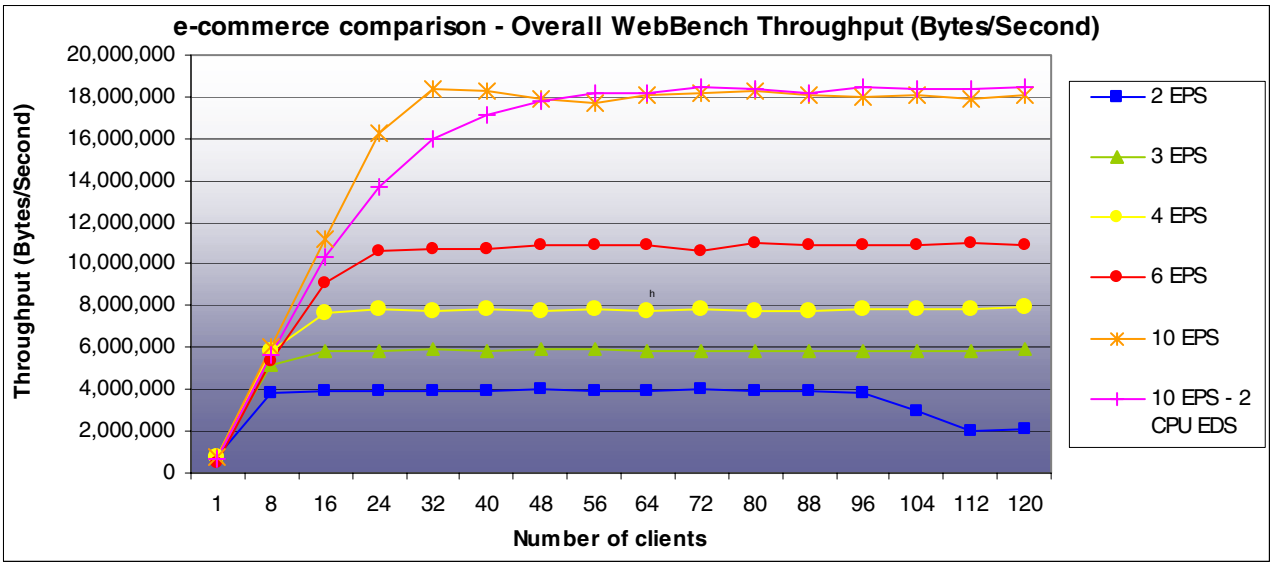

Figure 15: E-commerce load throughput comparison of the 6 configurations

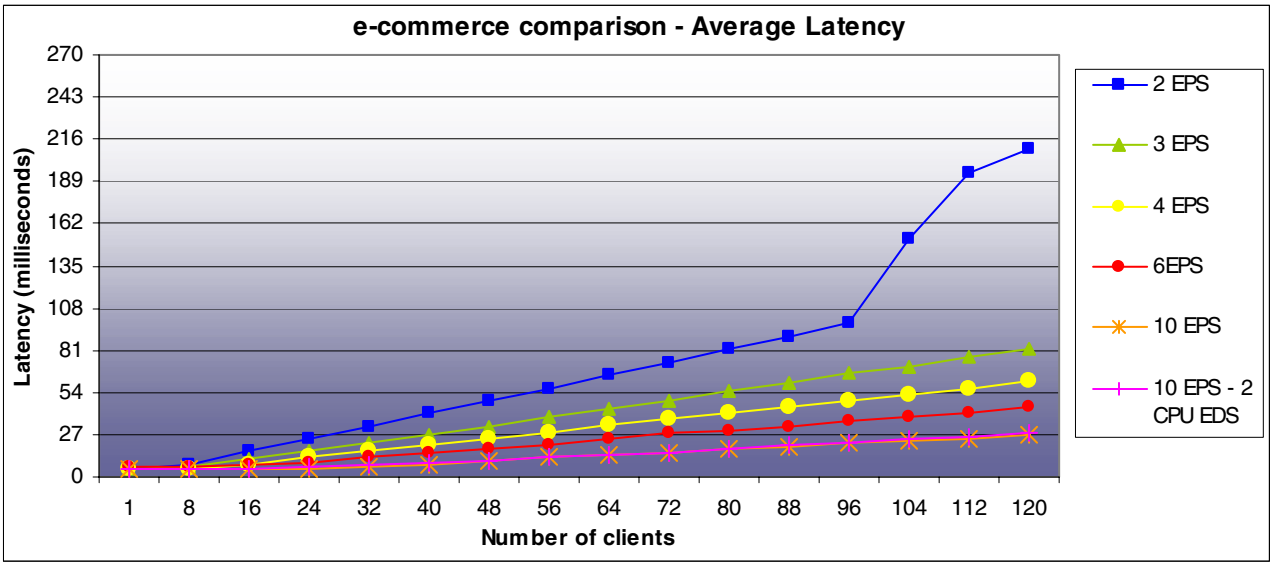

Figure 16: E-commerce load latency comparison of the 6 configurations

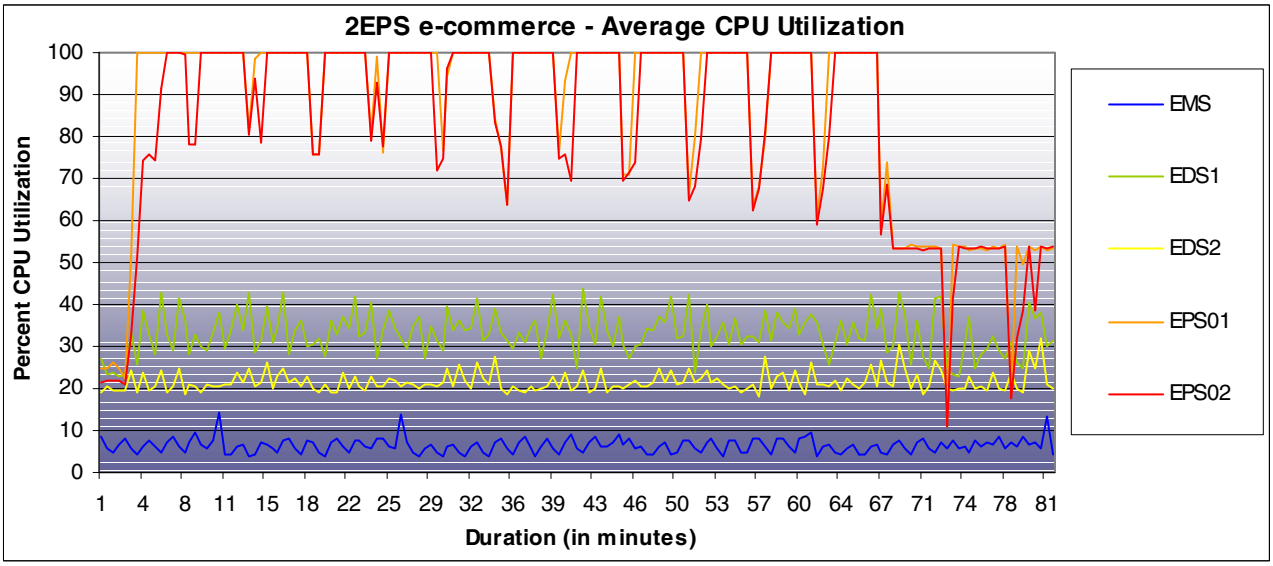

Figure 17: Configuration 1, CPU utilization e-commerce results for all Enterprise Stack components

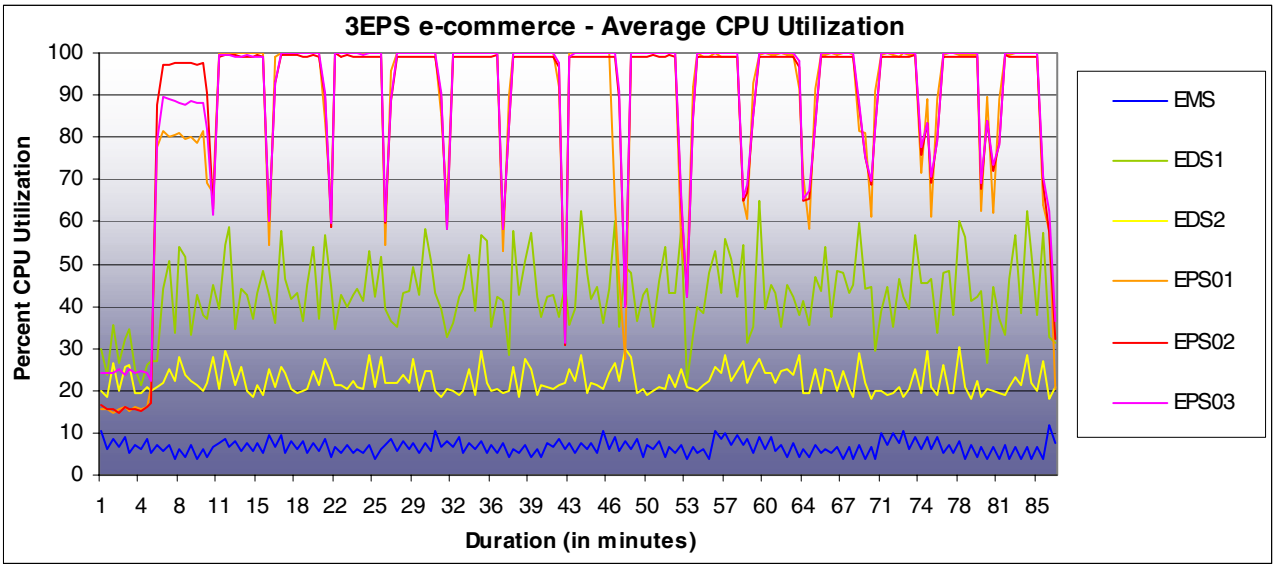

Figure 18: Configuration 2, CPU utilization e-commerce results for all Enterprise Stack components

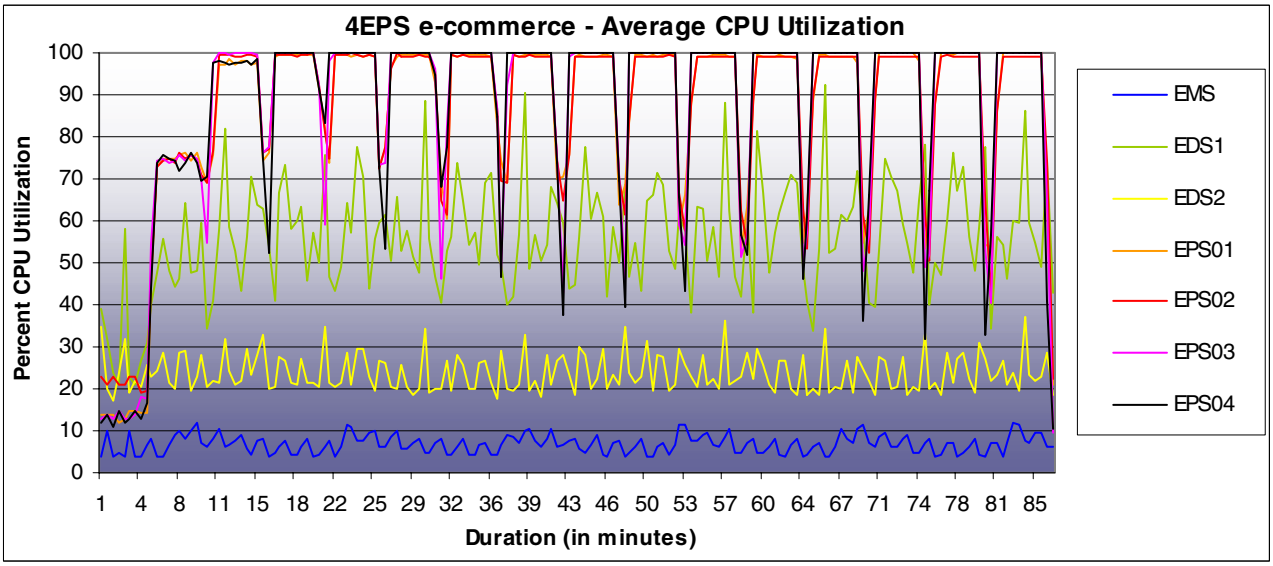

Figure 19: Configuration 3, CPU utilization e-commerce results for all Enterprise Stack components

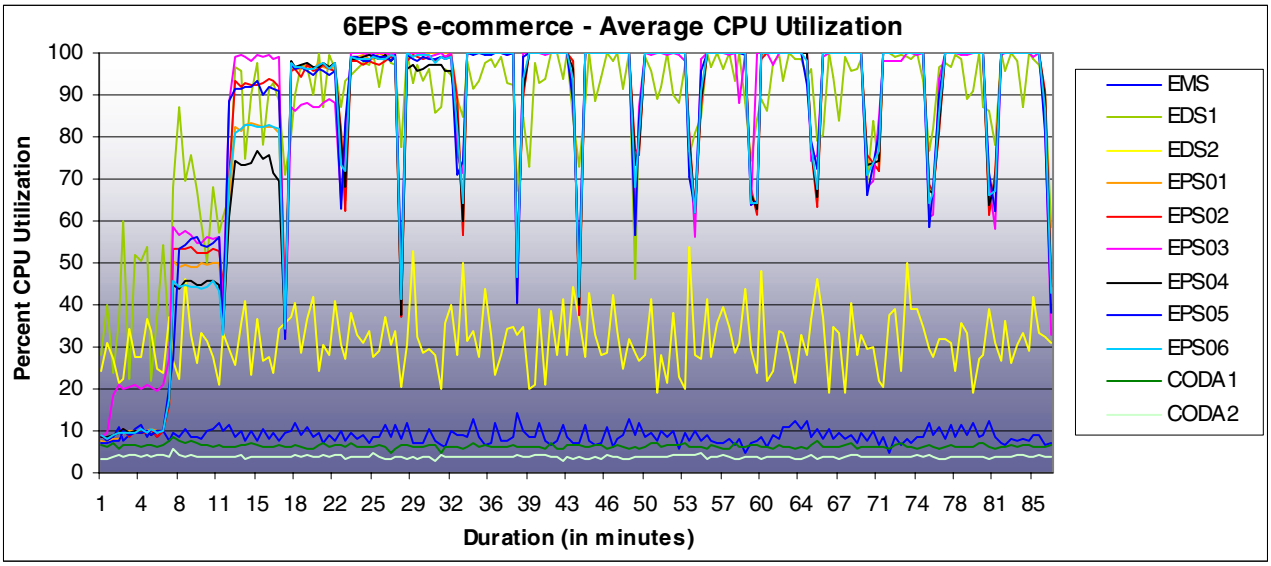

Figure 20: Configuration 4, CPU utilization e-commerce results for all Enterprise Stack components

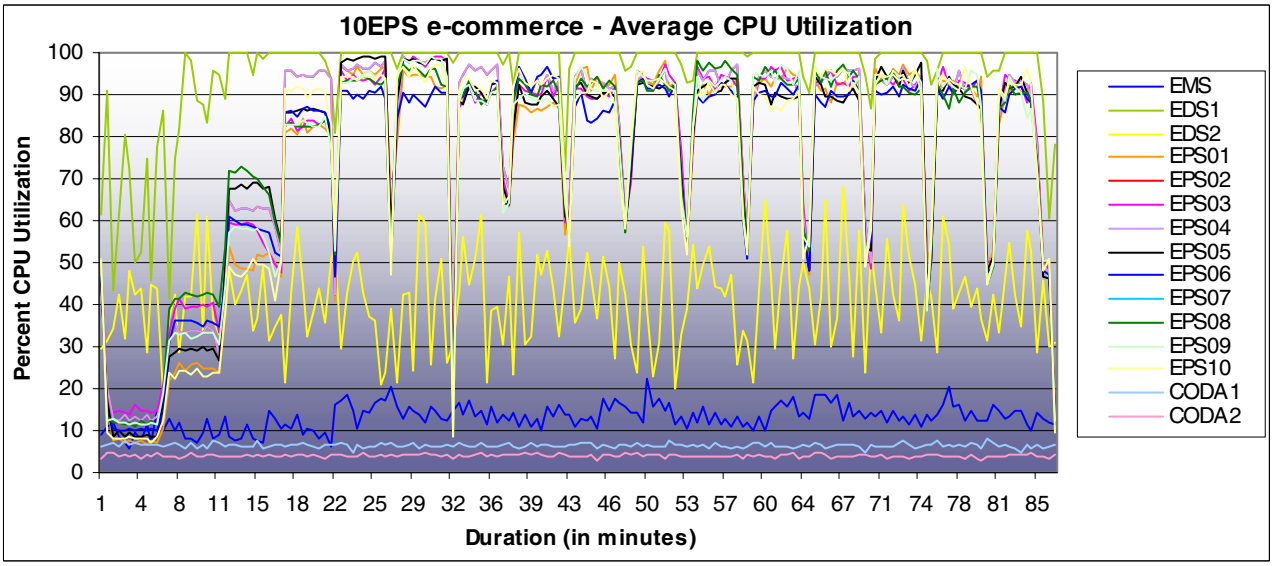

Figure 21: Configuration 5, CPU utilization e-commerce results for all Enterprise Stack components

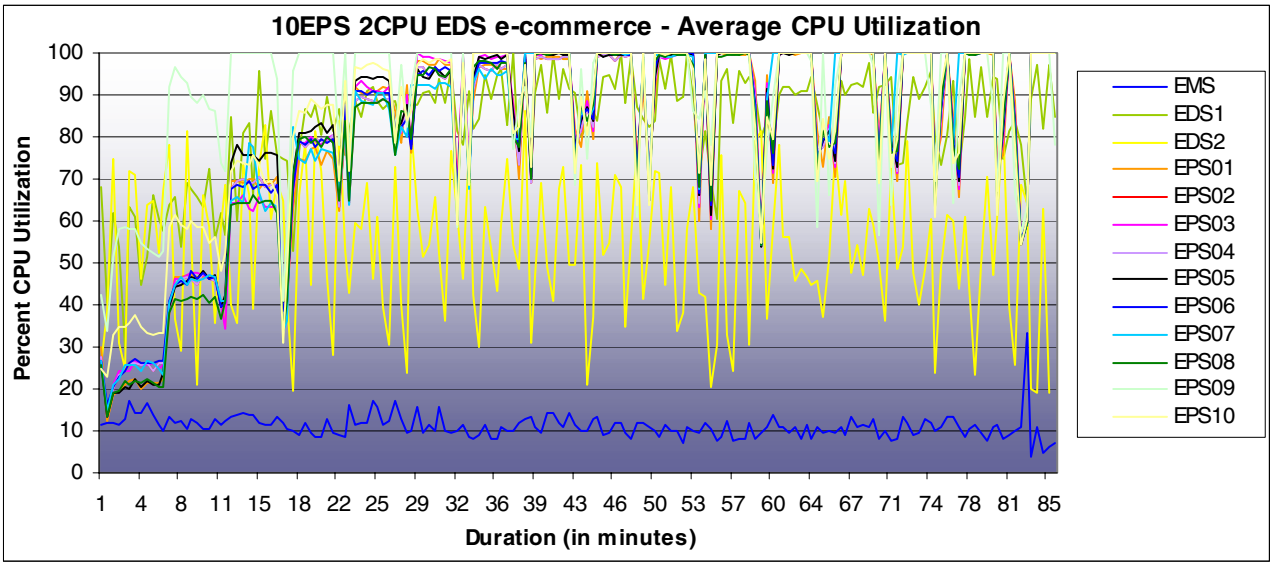

Figure 22: Configuration 6, CPU utilization e-commerce results for all Enterprise Stack components

Failover Results

Here is another area where we found that the Enterprise Stack performed exceptionally well. The EDS failover tests showed that there was virtually no impact on performance when a node was failed over. This is quite impressive, when we completely disconnected the load-balancing piece from the network, the backup unit took control immediately. Not only did it regain and maintain control, but also the primary EDS regained control just as fast when we re-connected the network cable. The EPS failover test worked just as well. There was an expected dip in performance during the failover as fewer web servers were available to service requests. However, the performance returned to its previous state after we re-connected the network cable.

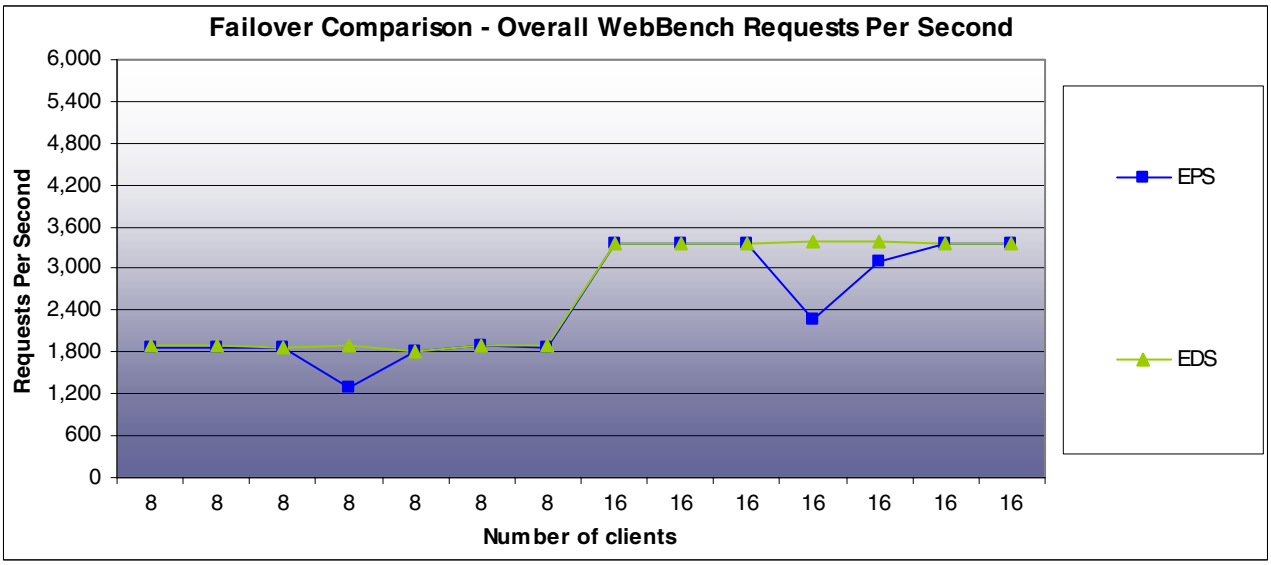

Figure 23: Static load requests per second failover comparison

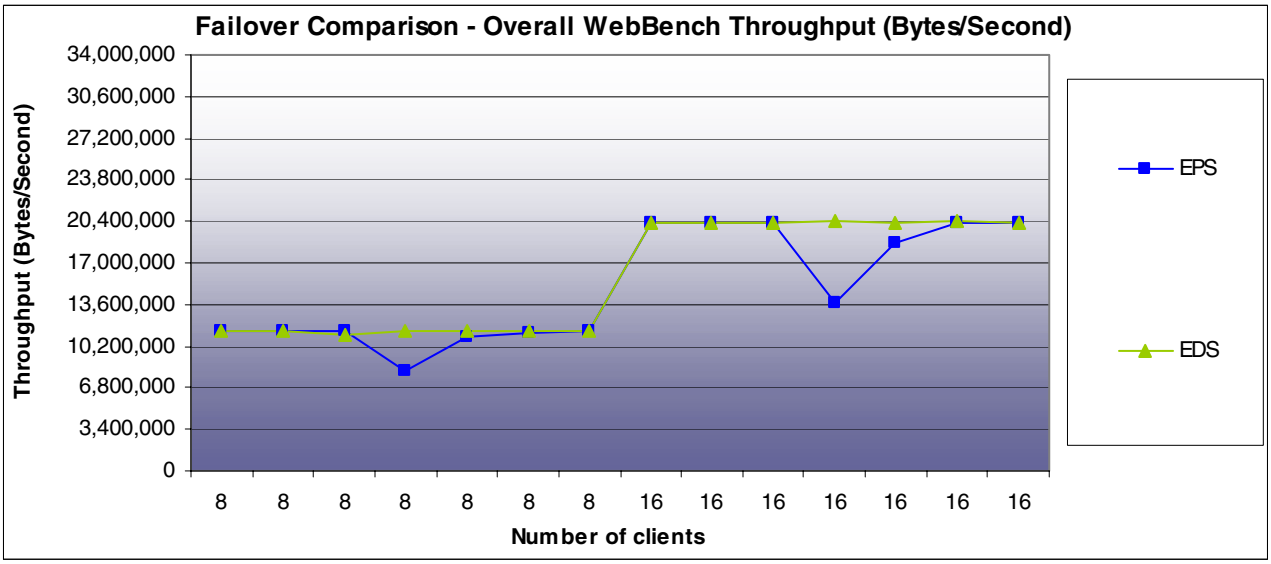

Figure 24: Static load throughput failover comparison

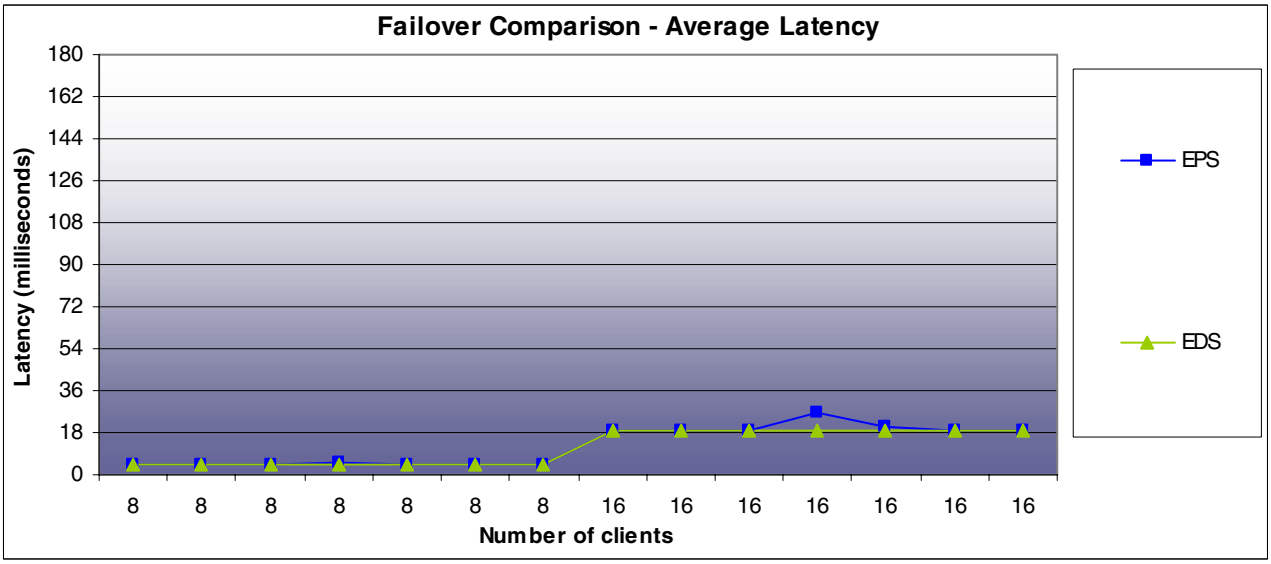

Figure 25: Static load response time failover comparison

Appendix

A. INTERNETpro Enterprise Stack Configuration Information

We used an Enterprise Stack configuration for the testing in this report that consisted of the following devices: 1-INTERNETpro - Enterprise Management Stack (EMS), 2-INTERNETpro - Enterprise Director Stacks (EDS), and 12-INTERNETpro - Enterprise Portal Stacks (EPS). The number of EMS and EDS remained consistent for all of the tests in this report. The number of EPS varied from 2-12 depending on each test. Each set of results in this report discloses the number of EPS participating in the test. Specific information about the configuration of each Enterprise Stack device is listed below.

Enterprise Stack EMS configuration	
Operating System	Embedded Linux version 2.2.18, Coda Kernel/Venus communications v4.6.0 Apache web server 1.3.17, Apache Jserv 1.1.2 with gnujsp 1.0.1, mod-ssl 2.8.0 with openssl 0.9.6, PHP 4.0.4pl1 with Zend Engine v1.0.4 and Zend Optimizer v1.0.0, mod_perl 1.25, mod_fastcgi 2.2.10
Vendor/Model	Internet Appliance/INTERNET*pro* Enterprise Management Stack (EMS)
CPU	1 - 850 MHz Pentium III – 256K L2 Cache
System bus speed	100 MHz
RAM	128 MB

Enterprise Stack EDS configuration	
Operating System	Embedded Linux version 2.2.18, Coda Kernel/Venus communications v4.6.0 Apache web server 1.3.17, Apache Jserv 1.1.2 with gnujsp 1.0.1, mod-ssl 2.8.0 with openssl 0.9.6, PHP 4.0.4pl1 with Zend Engine v1.0.4 and Zend Optimizer v1.0.0, mod_perl 1.25, mod_fastcgi 2.2.10
Vendor/Model	Internet Appliance/ INTERNET*pro* Enterprise Director Stack (EDS)
CPU	1 or 2 - 850 MHz Pentium III – 256K L2 Cache (There was 1 CPU for most of the tests conducted for this report. The tests results that contained an additional CPU are disclosed in the test results section of this report).
System bus speed	100 MHz
RAM	128 MB

Enterprise Stack EPS configuration	
Operating System	Embedded Linux version 2.2.18, Coda Kernel/Venus communications v4.6.0 Apache web server 1.3.17, Apache Jserv 1.1.2 with gnujsp 1.0.1, mod-ssl 2.8.0 with openssl 0.9.6, PHP 4.0.4pl1 with Zend Engine v1.0.4 and Zend Optimizer v1.0.0, mod_perl 1.25, mod_fastcgi 2.2.10
Vendor/Model	Internet Appliance/ INTERNET*pro* Enterprise Portal Stack (EPS)
CPU	2 - 850 MHz Pentium III – 256K L2 Cache
System bus speed	100 MHz
RAM	1024 MB (960 available to Operating System)

B. Apache configuration information

Each Enterprise Stack EPS we used for these tests used Apache 1.3.17 as a web server. The httpd.conf file, Apache's configuration file, is below. In the interest of saving space, we have deleted all commented-out lines and blank spaces and reduced the font size to 6pt. so that we could display the file in two columns.

Apache httpd.conf

```
ServerType standalone
ResourceConfig /dev/null
AccessConfig /dev/null
ServerRoot "/usr/local/apache"
PidFile /usr/local/apache/logs/httpd.pid
ScoreBoardFile /usr/local/apache/logs/httpd.scoreboard
LoadModule vhost_alias_module libexec/mod_vhost_alias.so
LoadModule env_module        libexec/mod_env.so
LoadModule config_log_module  libexec/mod_log_config.so
LoadModule mime_module       libexec/mod_mime.so
LoadModule negotiation_module libexec/mod_negotiation.so
LoadModule includes_module    libexec/mod_include.so
LoadModule autoindex_module   libexec/mod_autoindex.so
LoadModule dir_module        libexec/mod_dir.so
LoadModule cgi_module        libexec/mod_cgi.so
LoadModule access_module      libexec/mod_access.so
LoadModule setenvif_module    libexec/mod_setenvif.so
LoadModule ssl_module        libexec/libssl.so
ClearModuleList
AddModule mod_so.c
AddModule mod_vhost_alias.c
AddModule mod_env.c
AddModule mod_log_config.c
AddModule mod_mime.c
AddModule mod_negotiation.c
AddModule mod_include.c
AddModule mod_autoindex.c
AddModule mod_dir.c
AddModule mod_cgi.c
AddModule mod_alias.c
AddModule mod_access.c
AddModule mod_setenvif.c
AddModule mod_ssl.c
ServerName localhost
Group nobody
ServerAdmin root@localhost
User nobody
Listen 443
Port 80
Listen 80
Timeout 60
HostnameLookups off
KeepAlive on
MinSpareServers 100
MaxSpareServers 100
MaxRequestsPerChild 10000
KeepAliveTimeout 30
MaxClients 100
StartServers 100
MaxKeepAliveRequests 100
ListenBackLog 2000
RLimitCPU 60 120
RLimitMEM 10485760 20971520
RLimitNPROC 512 512
DocumentRoot "/usr/local/apache/htdocs"
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
<Directory "/usr/local/apache/htdocs">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
<IfModule mod_userdir.c>
    UserDir public_html
</IfModule>
<IfModule mod_dir.c>
    DirectoryIndex index.html index.htm index.php index.jsp
</IfModule>
AccessFileName .htaccess
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
UseCanonicalName On
<IfModule mod_mime.c>
    TypesConfig /usr/local/apache/conf/mime.types
</IfModule>
    AddIcon /icons/back.gif ..
    AddIcon /icons/hand.right.gif README
    AddIcon /icons/folder.gif ^^DIRECTORY^^
    AddIcon /icons/blank.gif ^^BLANKICON^^
    DefaultIcon /icons/unknown.gif
    ReadmeName README
    HeaderName HEADER
    IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
</IfModule>
<IfModule mod_mime.c>
    AddEncoding x-compress Z
    AddEncoding x-gzip gz tgz
    AddLanguage da .dk
    AddLanguage nl .nl
    AddLanguage en .en
    AddLanguage et .ee
    AddLanguage fr .fr
    AddLanguage de .de
    AddLanguage el .el
  AddLanguage he .he
    AddCharset ISO-8859-8 .iso8859-8
    AddLanguage it .it
    AddLanguage ja .ja
    AddCharset ISO-2022-JP .jis
    AddLanguage kr .kr
    AddCharset ISO-2022-KR .iso-kr
    AddLanguage no .no
    AddLanguage pl .po
    AddCharset ISO-8859-2 .iso-pl
    AddLanguage pt .pt
    AddLanguage pt-br .pt-br
    AddLanguage ltz .lu
    AddLanguage ca .ca
    AddLanguage es .es
    AddLanguage sv .se
    AddLanguage cz .cz
    AddLanguage ru .ru
    AddLanguage tw .tw
    AddCharset Big5        .Big5    .big5
    AddCharset WINDOWS-1251 .cp-1251
    AddCharset CP866       .cp866
    AddCharset ISO-8859-5   .iso-ru
    AddCharset KOI8-R      .koi8-r
    AddCharset UCS-2       .ucs2
    AddCharset UCS-4       .ucs4
    AddCharset UTF-8       .utf8
    <IfModule mod_negotiation.c>
        LanguagePriority en da nl et fr de el it ja kr no pl pt pt-br ru ltz ca es sv tw
    </IfModule>
    AddType application/x-httpd-php .php
    AddType application/x-httpd-php-source .phps
    AddType application/x-tar .tgz
</IfModule>
<IfModule mod_setenvif.c>
    BrowserMatch "RealPlayer 4\.0" force-response-1.0
    BrowserMatch "Java/1\.0" force-response-1.0
    BrowserMatch "JDK/1\.0" force-response-1.0
</IfModule>
<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
</IfDefine>
<IfModule mod_ssl.c>
SSLPassPhraseDialog  builtin
SSLSessionCache        dbm:/usr/local/apache/logs/ssl_scache
SSLSessionCacheTimeout  300
SSLMutex  file:/usr/local/apache/logs/ssl_mutex
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
SSLLog      "|/usr/local/apache/bin/rotatelogs /usr/local/apache/logs/ssl_engine_log
86400"
SSLLogLevel info
</IfModule>
Include /usr/local/apache/conf/jserv/jserv.conf
Include /usr/local/apache/conf/httpsd_local.conf
NameVirtualHost 200.4.1.240:80
NameVirtualHost 200.4.1.240:443
<VirtualHost 200.4.1.240:80>
  DocumentRoot "/coda/etl.com/htdocs"
  ServerName etl.com
```

```
DefaultType text/plain
<IfModule mod_mime_magic.c>
    MIMEMagicFile /usr/local/apache/conf/magic
</IfModule>
ErrorLog "|/usr/local/apache/bin/rotatelogs /usr/local/apache/logs/error_log 86400"
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog "|/usr/local/apache/bin/rotatelogs /usr/local/apache/logs/access_log 86400"
common
ServerSignature On
ServerTokens Min
<IfModule mod_alias.c>
   Alias /icons/ "/usr/local/apache/icons/"
    <Directory "/usr/local/apache/icons">
       Options Indexes MultiViews
       AllowOverride None
       Order allow,deny
       Allow from all
    </Directory>
   ScriptAlias /cgi-bin/ "/usr/local/apache/cgi-bin/"
    <Directory "/usr/local/apache/cgi-bin">
       AllowOverride None
       Options None
       Order allow,deny
       Allow from all
    </Directory>
</IfModule>
    IndexOptions FancyIndexing
    AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
    AddIconByType (TXT,/icons/text.gif) text/*
    AddIconByType (IMG,/icons/image2.gif) image/*
    AddIconByType (SND,/icons/sound2.gif) audio/*
    AddIconByType (VID,/icons/movie.gif) video/*
    AddIcon /icons/binary.gif .bin .exe
    AddIcon /icons/binhex.gif .hqx
    AddIcon /icons/tar.gif .tar
    AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
    AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
    AddIcon /icons/a.gif .ps .ai .eps
    AddIcon /icons/layout.gif .html .shtml .htm .pdf
    AddIcon /icons/text.gif .txt
    AddIcon /icons/c.gif .c
    AddIcon /icons/p.gif .pl .py
    AddIcon /icons/f.gif .for
    AddIcon /icons/dvi.gif .dvi
    AddIcon /icons/uuencoded.gif .uu
    AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
    AddIcon /icons/tex.gif .tex
    AddIcon /icons/bomb.gif core
  ServerAdmin webmaster@etl.com
  ErrorLog "|/usr/local/apache/bin/rotatelogs /usr/local/apache/logs/etl.com_error_log
86400"
  CustomLog "|/usr/local/apache/bin/rotatelogs
/usr/local/apache/logs/etl.com_access_log 86400" common
  <Directory "/coda/etl.com/htdocs">
    Options Indexes Includes FollowSymLinks ExecCGI MultiViews
    AllowOverride None
  </Directory>
  ScriptAlias "/cgi-bin/" "/coda/etl.com/cgi-bin/"
  <Directory "/coda/etl.com/cgi-bin/">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
  </Directory>
</VirtualHost>
<VirtualHost 200.4.1.240:443>
  DocumentRoot "/coda/etl.com/htdocs"
  ServerName etl.com
  ServerAdmin webmaster@etl.com
  <Directory "/coda/etl.com/htdocs">
    Options Indexes Includes FollowSymLinks ExecCGI MultiViews
    AllowOverride None
  </Directory>
  ScriptAlias "/cgi-bin/" "/coda/etl.com/cgi-bin/"
  <Directory "/coda/etl.com/cgi-bin/">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
  </Directory>
  SSLEngine on
  SSLCipherSuite
ALL:!ADH:!EXP56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
  SSLCertificateFile /usr/local/apache/conf/ssl.crt/200.4.1.240.crt
  SSLCertificateKeyFile /usr/local/apache/conf/ssl.key/200.4.1.240.key
  #SSLVerifyClient require
  #SSLVerifyDepth  10
  <Files ~ "\.(cgi|shtml|phtml|php3?|jsp)$">
    SSLOptions +StdEnvVars
  </Files>
  <Directory "/coda/etl.com/cgi-bin/">
    SSLOptions +StdEnvVars
  </Directory>
  SetEnvIf User-Agent ".*MSIE.*" \
         nokeepalive ssl-unclean-shutdown \
         downgrade-1.0 force-response-1.0
  CustomLog "|/usr/local/apache/bin/rotatelogs
/usr/local/apache/logs/etl.com_ssl_request_log 86400" \
         "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>
```

C. WebBench client information

We used a test bed of 120 clients to conduct the tests in this report. The client test bed contained two machine types, 56-500 MHz Celeron, and 64-200 MHz Pentium Pro based systems. Specific information about the configuration of each client type is listed below.

Client configuration	
Operating System	Microsoft Windows NT Workstation 4.0 Service Pack 6
Vendor/Model	Dell Optiplex GX100
CPU	500 MHz Celeron
RAM	128 MB

Client configuration	
Operating System	Microsoft Windows NT Workstation 4.0 Service Pack 6
Vendor/Model	Dell Dimension XPS Pro200n
CPU	200 MHz Pentium Pro
RAM	64 MB

D. WebBench workload configuration

WebBench 4.0.1 ships with standard workloads based on profiles of popular static, dynamic, and secure content web sites. For this project we used the standard static and the standard Unix e-commerce cgi workloads. Brief file type descriptors of the file types that appear in the workload definition file are provided below. The numbers that end each line are the percentage that each file type is used during the test. The sum of the percentages equals 100. We left the workload for all tests unmodified. The workload provided by WebBench consists of approximately 60 MB of files in a complex nested file directory structure.

static Workload	e-commerce workload
DEFINE_CLASSES CLASS_223.gif: 20 CLASS_735.gif: 8 CLASS_1522.gif: 12 CLASS_2895.gif: 20 CLASS_6040.htm: 14 CLASS_11426.htm: 16 CLASS_22132.htm: 7 CLASS_FRACTIONAL: 1 CLASS_404: 2	DEFINE_CLASSES CLASS_SSL_DYNAMIC: 2 CLASS_SSL_STATIC: 6 CLASS_DYNAMIC: 16 CLASS_223.gif: 14 CLASS_735.gif: 13 CLASS_1522.gif: 15 CLASS_2895.gif: 11 CLASS_6040.htm: 9 CLASS_11426.htm: 8 CLASS_FRACTIONAL: 4 CLASS_404: 2

E. Network hardware information

The test bed consisted of five Extreme Summit 48 Layer 3 switches. Each switch was set to its factory default settings prior to testing. The switch that the INTERNET*pro* Enterprise Stack was connected to was divided into two vlans. Vlan1 was visible by the clients and contained 24 10/100 ports as well as both Gigabit ports. Vlan2 contained the remaining 24 ports and was not accessible by the clients. Vlan2 was used by the second NIC in each of the Enterprise Stack devices for administration purposes. A network topology of the test bed can be found in Figure 4, in the testing methodology section of this report.

Network Hardware	
Operating System	4.1.19b2
Vendor/Model	Extreme Summit 48
10/100 Ports	Auto Detect Speed/Duplex (verified to be operating at 100Mbps/Full Duplex)
Gigabit Ports	Auto Detect Speed/Duplex (verified to be operating at 1000Mbps/Full Duplex)

eTesting Labs Inc. (www.etestinglabs.com), a Ziff Davis Media company, leads the industry in Internet and technology testing. In June 2000, ZD Labs changed its name to eTesting Labs to better reflect the breadth of its testing services. Building on Ziff Davis Media's history of leadership in product reviews and benchmark development, eTesting Labs brings independent testing, research, development, and analysis directly to publications, Web sites, vendors, and IT organizations everywhere.

For more information email us at etesting_labs_info@ziffdavis.com or call us toll free at 877-619-9259.

Disclaimer of Warranties; Limitation of Liability:

eTESTING LABS INC. HAS MADE REASONABLE EFFORTS TO ENSURE THE ACCURACY AND VALIDITY OF ITS TESTING, HOWEVER, eTESTING LABS SPECIFICALLY DISCLAIMS ANY WARRANTY, EXPRESSED OR IMPLIED, RELATING TO THE TEST RESULTS AND ANALYSIS, THEIR ACCURACY, COMPLETENESS OR QUALITY, INCLUDING ANY IMPLIED WARRANTY OF FITNESS FOR ANY PARTICULAR PURPOSE. ALL PERSONS OR ENTITIES RELYING ON THE RESULTS OF ANY TESTING DO SO AT THEIR OWN RISK, AND AGREE THAT eTESTING LABS, ITS EMPLOYEES AND ITS SUBCONTRACTORS SHALL HAVE NO LIABILITY WHATSOEVER FROM ANY CLAIM OF LOSS OR DAMAGE ON ACCOUNT OF ANY ALLEGED ERROR OR DEFECT IN ANY TESTING PROCEDURE OR RESULT.

IN NO EVENT SHALL eTESTING LABS BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH ITS TESTING, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL eTESTING LABS' LIABILITY, INCLUDING FOR DIRECT DAMAGES, EXCEED THE AMOUNTS PAID IN CONNECTION WITH eTESTING LABS' TESTING. CUSTOMER'S SOLE AND EXCLUSIVE REMEDIES ARE AS SET FORTH HEREIN.