ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

PROYECTO

FORMACIÓN EN INGENIERÍA DE SISTEMAS INFORMÁTICOS

"ESTUDIO ESTADÍSTICO DEL RENDIMIENTO DE SERVIDORES WEB, DE DISTRIBUCIÓN LIBRE, APLICADO A LOS SERVIDORES DE LA ESCUELA DE INGENIERÍA EN SISTEMAS (ESPOCH)"

TESIS DE GRADO

PREVIA LA OBTENCIÓN DEL TÍTULO DE:

Ingeniero en Sistemas Informáticos

Mónica Eulalia Poma Salazar

Riobamba – Ecuador

2009

AGRADECIMIENTO

"... A Dios, a mis Padres, mi Hijo, amigos, profesores que me acompañaron durante todo este proceso, no solo en la realización del Proyecto de Grado sino durante toda la carrera... gracias por su apoyo incondicional...."

Mónica

DEDICATORIA

"... A mi Padre que con su partida prematura no puede ver su sacrificio culminado con éxito, a mi Madre quien con abnegación y mucha paciencia me apoya en todo momento de mi vida.
A mi hijo que en silencio ha sabido apoyarme para concluir este trabajo, a mi hermana que su apoyo es fortaleza imperecedera para alcanzar mis metas ... "

Mónica

FIRMAS RESPONSABLES Y NOTAS

	FIRMAS	**FECHA**
Dr. Romeo Rodríguez **DECANO FACULTAD DE INFORMATICA Y ELECTRONICA**	……	……
Ing. Danilo Pástor **DIRECTOR PROFESIS**	……	……
Ing. Jorge Menéndez **DIRECTOR DE TESIS**	……	……
Ing. Diego Ávila **MIEMBRO DEL TRIBUNAL**	……	……
…… **MIEMBRO DEL TRIBUNAL**	……	……
Lic. Carlos Rodríguez **DIRECTOR CENTRO DE DOCUMENTACION**	……	……

NOTA DE LA TESIS ……

"Yo, Mónica Eulalia Poma Salazar, soy responsable de las ideas y resultados expuestos en esta tesis y el patrimonio intelectual de la tesis de grado pertenece a la Escuela Superior Politécnica de Chimborazo"

Mónica Poma Salazar

ÍNDICE DE ABREVIATURAS

ASP (Microsoft): Active Server

CGI Common Gateway Interface

DHTML Dynamic HyperText Markup Language

DNS El sistema de nombre de dominios.

FTP File Transfer Protocol. (Protocolo de Transferencia de Archivos).

GUI Graphical User Interface. (Interfaz Gráfica de Usuario).

HTML Hypertext Markup Language. (Lenguaje de Marcado de Hipertexto).

HTTP Hypertext Transfer Protocol. (Protocolo de Transferencia de Hipertexto).

HTTPS Secure HypterText Transfer Protocol

IP Internet Protocol.

IS Internet Service Provider

LDAP (*Lightweight Directory Access Protocol*

PHP Hypertext Preprocessor/Personal

SCGI **S***imple* **C***ommon* **G***ateway* **I***nterface*

SQL Structured Query Language

SSI Server Side Includes

SSL Secure Sockets Layer

SSH Secure Shell

URL Uniform Resourse Locator. (Localizador Uniforme de Recursos). Uniform

URN Uniform Resource Name. (Nombre Uniforme de Recursos).

XSS Cross-Site Scripting

WAP Wireless Application Protocol

WAR Archivos Web.

WWW World Wide Web

ÍNDICE

PORTADA

AGRADECIMIENTO

DEDICATORIA

ÍNDICE DE ABREVIATURAS

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

		Pág.
	CAPÍTULO I MARCO REFERENCIAL	**17**
1.1.	**Prefacio**	**17**
1.2.	**Formulación del Problema**	**17**
1.3.	**Objetivos**	**19**
	Objetivo General	19
	Objetivos Específicos	19
1.4.	**Justificación del Problema**	**19**
1.5.	**Planteamiento de la Hipótesis**	**20**
	CAPÍTULO II MARCO TEÓRICO	**21**
2.1.	Prefacio	21
2.2.	Breve historia de la WWW	21
2.2.1.	Fundamentos de la Web	21
2.2.1.1.	El protocolo HTTP	22
2.2.1.2.	El lenguaje HTML	24
2.3.	Historia de las aplicaciones Web	24
2.3.1.	Historia	25
2.3.2.	Desarrollo de Aplicaciones Web	26
2.3.2.1.	Arquitectura Web	27
2.3.2.2.	El Navegador Web (Browser)	27
2.3.2.3.	El Servidor http	28
2.3.2.4.	Aplicaciones Multinivel	29
2.3.2.5.	Ejemplos de Herramientas de Edición Web	31
2.4.	Conceptos básicos del servidor Web	32
2.5.	Servidor Web Apache	34

2.5.2. Instalación de Apache 35
2.5.2.1. Descargar 35
2.5.2.2. Descomprimir 36
2.5.3. Configuración de Apache 36
2.5.3.2. Inicio de Apache 37
2.6. Servidor Web Lighttpd 38
2.6.1. Lenguajes de servidor 39
2.6.2. Instalación de Lighttpd 40
2.6.3. Configuración de Lighttpd 40
2.7. Servidor Web Cherokee 42
2.7.2. Arquitectura 43
2.7.3. Instalación de Cherokee 44
2.7.4. Configuración de Cherokee 44
2.1. Prefacio 21
2.2. Breve historia de la WWW 21
2.2.1. Fundamentos de la Web 21
2.2.1.1. El protocolo HTTP 22
2.2.1.2. El lenguaje HTML 24
2.3. Historia de las aplicaciones Web 24
2.3.1. Historia 25
2.3.2. Desarrollo de Aplicaciones Web 26
2.3.2.1. Arquitectura Web 27

CAPÍTULO III COMPARATIVA DE SERVIDORES 46

3.1. Prefacio 46
3.2. ¿Por qué el benchmarking o comparativa es tan importante? 46
3.2.1. Consideraciones no válidas en la medida del rendimiento 47
3.2.2. Procedimientos de medida e interpretación de resultados 47
3.2.3. Entendiendo la elección de la herramienta 48
3.3. Comparativa de Servidores http 50
3.3.1. Tipos de servidores http 50
3.4. Hardware empleado 51
3.5. Características implementadas 52
3.5.1. Visión General 52
3.5.1.1. Características de Seguridad 52
3.5.1.1.1. Autentificación básica 54
3.5.1.1.2. Autentificación Digest 54

3.5.1.1.3.	Https (http Seguro)	54
3.5.1.2.	Contenido Dinámico	54
3.5.1.2.1.	CGI	55
3.5.1.2.2.	FastCGI	55
3.5.1.2.	Servlet	55
3.5.1.3.	SSI	55
3.5.1.4.	Lenguajes del lado del servidor	56
3.6.	Análisis Estadístico	57
3.6.1.	Caracteres y variables Estadísticas	57
3.6.2.	Métodos Estadísticos	58
3.6.2.1.	Método de Duncan	58
3.6.2.2.	Método de Scheffé	60
3.6.2.3.	Método de Tukey o Método de la Diferencia Significativa Honesta de Tukey (DSH)	61
3.6.2.4.	Graficas Estadísticas	62
3.6.2.4.1.	Diagrama de Cajas y Bigotes (Box and Whisker Plot)	63
3.6.3.	Diseño de Experimentos	63
3.6.3.1.	Definiciones/Conceptos	64
3.6.3.1.1.	Experimento	64
3.6.3.1.2.	Diseño de Experimentos	64
3.6.3.1.3.	Variable	65
3.6.3.1.4.	Sujeto o Unidad Experimental	65
3.6.3.1.4.1.	Observación	65
3.6.3.1.4.2.	Diferentes tipos de variables en el Diseño de un Experimento	65
3.6.3.1.4.3.	Repetición	66
3.6.3.1.4.4.	Aleatorización	67
3.6.4.	Elaboración del Experimento en nuestra Investigación	67
	Paso 1: Definir claramente la hipótesis a comprobar [9]	68
	Paso 2: Identificar la variable independiente	69
	Paso 3: Establecer los tratamientos	69
	Paso 4: Decidir el número de repeticiones para cada tratamiento	70
	Paso 5: Definir los sujetos sobre los que se va a realizar la medida	71
	Paso 6: Determinar la variable dependiente	72
	Paso 7: Explicitar el procedimiento de aleatorización	72
	Paso 8: Identificar posibles factores de "ruido" y/o variables de bloqueo	74

Paso 9: Asegurarse de la aptitud del Diseño del Experimento para contestar la pregunta inicial 75
Paso 10: Realización del Experimento 75

CAPÍTULO IV ANÁLISIS DE RESULTADOS 85

4.1. Prefacio 85
4.2. Evaluación de las características implementadas 86
4.3. Resultados numéricos de la comparativa 87
4.3.1. Evaluación de los Servidores http por prueba realizada 87
a) + REQUEST PER SECOND (PETICIONES POR SEGUNDO) 88
Páginas Estáticas – 1000 – 10000 - 100000 peticiones **88**
Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 90
Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 91
Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 92
b) - TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA) 93
Páginas Estáticas – 1000 – 10000 - 100000 peticiones 93
Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 95
Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 96
Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 97
c) + TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA 99
Páginas Estáticas – 1000 – 10000 - 100000 peticiones 99
Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 100
Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 101
Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 102
d) - TIME PER REQUEST (ms) - (TIEMPO POR PETICION) 104
Páginas Estáticas – 1000 – 10000 - 100000 peticiones 104
Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 105
Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 106

peticiones

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 108

e) - %CPU USER (Uso de CPU) 110

Páginas Estáticas – 1000 – 10000 - 100000 peticiones 110

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 111

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 112

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 113

f) - %CPU por sistema y Kernel 115

Páginas Estáticas – 1000 – 10000 - 100000 peticiones 115

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 116

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 117

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 119

g) + %CPU IDLE (CPU DISPONIBLE) 121

Páginas Estáticas – 1000 – 10000 - 100000 peticiones 121

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 122

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 123

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 124

h) - KBMEMUSED - %MEMORIA RAM UTILIZADA 126

Páginas Estáticas – 1000 – 10000 - 100000 peticiones 126

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 127

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 128

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones 129

i) + BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS – DISCO 131

Páginas Estáticas – 1000 – 10000 - 100000 peticiones 131

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones 132

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones 133

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 134

100000 peticiones
4.4. Resultados de la Comparativa. 136
4.5. Conclusiones de la comparativa. 140

Conclusiones
Recomendaciones
Bibliografía
Resumen
Summary
Glosario
Anexos

ÍNDICE DE TABLAS

Pág.

CAPÍTULO III COMPARATIVA DE SERVIDORES

Tabla III - 1.- Servidores Web y Licencias ---- 52
Tabla III - 2.- Plataformas de cada servidor http ---- 52
Tabla III - 3.- Seguridades por cada servidor ---- 53
Tabla III - 4.- Contenido Dinámico ---- 55
Tabla III - 5.- Lenguajes del lado del servidor ---- 56
Tabla III - 6.- Definición de variables en nuestra investigación – Variable Independiente ---- 69
Tabla III - 7.- Computadores que intervienen en la Arquitectura 1 ---- 71
Tabla III - 8.- Computadores que intervienen en la Arquitectura 2 ---- 71
Tabla III - 9.- Computadores que intervienen en la Arquitectura 3 ---- 72
Tabla III - 10.- Definición de variables en nuestra investigación – Variable Dependiente ---- 72
Tabla III - 11.- Codificación de los Laboratorios de la EIS ---- 73
Tabla III - 12.- Características Hardware Empleado – Arquitectura 1 ---- 76
Tabla III - 13.- Características Hardware Empleado – Arquitectura 2 ---- 76
Tabla III - 14.- Características Hardware Empleado – Arquitectura 3 ---- 76

CAPÍTULO IV ANÁLISIS DE RESULTADOS

Tabla IV- 1.- Características implementadas por cada Servidor Http ---- 114
Tabla IV- 2.- Resultado de la Comparativa ---- 164
Tabla IV- 3.- Asignación de Pesos para obtener el servidor apropiado a nuestra necesidad ---- 165
Tabla IV- 4.- Resultado Final

ÍNDICE DE ILUSTRACIONES

Pág.

CAPÍTULO II MARCO TEÓRICO

Ilustración II - 1.- Funcionamiento del Protocolo Http 23
Ilustración II - 2.- Esquema General de las tecnologías 25
Ilustración II - 3.- Arquitectura Web Básica 27
Ilustración II - 4.- Arquitectura Multinivel 30
Ilustración II - 5.- Arquitectura Web de tres niveles 31
Ilustración II - 6.- Logotipo Servidor http Apache 34
Ilustración II - 7.- Logotipo Servidor http Lighttpd 38
Ilustración II - 8.- Logotipo Servidor http Cherokee 42
Ilustración II - 9.- Página de resultado al configurar el servidor Web Cherokee 45

CAPÍTULO III COMPARATIVA DE SERVIDORES

Ilustración III. 1 Diagrama de Flujo Elaboración de Experimento 68

CAPÍTULO IV ANÁLISIS DE RESULTADOS

Ilustración IV- 1.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo 89
Ilustración IV- 2.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo 90
Ilustración IV- 3.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo 91
Ilustración IV- 4.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo 93
Ilustración IV- 5.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba 94
Ilustración IV- 6.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba 95
Ilustración IV- 7.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba 97
Ilustración IV- 8.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba 98
Ilustración IV- 9.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia 100

Ilustración IV- 10.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia 101
Ilustración IV- 11.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia 102
Ilustración IV- 12.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia 103
Ilustración IV- 13.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición 105
Ilustración IV- 14.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición 106
Ilustración IV- 15.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición 107
Ilustración IV- 16.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición 109
Ilustración IV- 17.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU 110
Ilustración IV- 18.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU 112
Ilustración IV- 19.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU 113
Ilustración IV- 20.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU 114
Ilustración IV- 21.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel 116
Ilustración IV- 22.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel 117
Ilustración IV- 23.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel 118
Ilustración IV- 24.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel 119
Ilustración IV- 25.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible 121
Ilustración IV- 26.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible 123
Ilustración IV- 27.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible 124
Ilustración IV- 28.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible 125
Ilustración IV- 29.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada 127

Ilustración IV- 30.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada 128
Ilustración IV- 31.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada 129
Ilustración IV- 32.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada 130
Ilustración IV- 33.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco 132
Ilustración IV- 34.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos – Disco 133
Ilustración IV- 35.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco 134
Ilustración IV- 36.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco 135

CAPÍTULO I

MARCO REFERENCIAL

1.1. Prefacio

Debido al auge que se encuentra viviendo el mundo entorno a la comunicación mediante el Internet, el florecimiento de la posibilidad irrestricta de compartir información mediante los sitios de la plataforma Web, se suscita una problemática que aún su discusión no se ha tomado de la manera que se debe. Los sitios Web, son de por si un medio de comunicación de masas, que pueden influir, educar o informar de una manera casi gratuita e irrestricta.

En la presente investigación se realizará una comparativa estadística de Servidores Web de distribución libre, tomando como parámetros de comparación sus características implementadas, pruebas de rendimiento, utilización de recursos, así como también sus fortalezas y debilidades. En base al análisis comparativo y a un conjunto de requerimientos necesarios para las pruebas de carga en cada servidor Web, lo cual permitirá saber cual es el más óptimo y el que va a cumplir con la hipótesis planteada.

1.2. Formulación del Problema

En la actualidad Internet se ha convertido en una herramienta necesaria para todas las personas ya que nos permite realizar diferentes actividades gracias a la difusión e incremento de sitios Web en los últimos años. En estos días los términos correo electrónico, foros de discusión, tiendas virtuales, etc. son muy comunes en nuestra sociedad y nos han hecho experimentar cambios significativos en el concepto que se tenía anteriormente de una computadora.

En las materias que son impartidas en la Escuela de Ingeniería en Sistemas, de la Escuela Superior Politécnica de Chimborazo existe un crecimiento significativo en la enseñanza de aplicaciones en entorno de red, por lo que aumenta la necesidad de servidores, los cuales son costosos en el mercado por sus características técnicas. Con este objetivo y el de optimizar los recursos existentes se han desarrollado otras investigaciones sobre la Virtualización de Servidores.

Un Servidor Virtual es en realidad una partición dentro de un servidor que habilita varias maquinas virtuales dentro de dicha máquina física, por medio de varias tecnologías. Un sistema operativo que soporta este tipo de tecnología es Linux. Así en esta investigación se utilizará software libre para reducir costos.

Entre los objetivos de la materia Aplicaciones Web está implementar, configurar y administrar información en un servidor Web, el cual es configurado por los estudiantes, lo que implicaría la necesidad de tener uno para cada estudiante o grupo de ellos, donde puedan desarrollar sus actividades académicas.

Dada la necesidad presentada y el consumo de recursos del servidor Web Apache, que se utiliza en la EIS, se propone un Estudio Estadístico del Rendimiento de Servidores Web, de distribución libre, aplicado a los servidores Web de la Escuela de Ingeniería en Sistemas, con el objetivo de crear servidores virtuales de páginas Web para las actividades prácticas de dicha materia y a la vez optimizar los recursos hardware existentes.

Para llevar a cabo este estudio se realizarán pruebas de carga a los servidores Web motivos del estudio, Apache Http Server, Lighttpd Server, Cherokee Http Server, evaluándose:

- Características implementadas
- Pruebas de rendimiento.
- Utilización de recursos.

El presente estudio analizará estadísticamente el grado de rendimiento existente entre los servidores Web de distribución libre a ser utilizado en la Escuela de Ingeniería en Sistemas, las observadas en los alumnos de la cátedra de aplicaciones Web de la misma escuela y las demandadas por el entorno laboral de la Facultad.

1.3. Objetivos

Objetivo General

Realizar un estudio estadístico del rendimiento de Servidores Web, de libre distribución, aplicado en los servidores de la Escuela de Ingeniería en Sistemas – FIE (ESPOCH), que contribuya a optimizar la plataforma de hardware disponible**.**

Objetivos Específicos

- Analizar las características de los servidores Web, de distribución libre, que permitan determinar parámetros de comparación y rendimiento
- Realizar un estudio estadístico del rendimiento de servidores Web como Apache, Cherokee, Lighttpd, de distribución libre, utilizando tecnología Linux, Distribución CentOS, aplicando scripts, que permitan someterlos a pruebas.
- Instalar y configurar el Servidor Web seleccionado y optimizado, en los servidores de la EIS, para mejorar así el servicio que se proporciona.

1.4. Justificación del Problema

Nuestro enfoque es hacia el análisis y comparativa del rendimiento de los servidores Web de distribución libre y así ofrecer un panorama de cuáles son las ventajas y desventajas en cada uno de ellos, para elaborar un conjunto de recomendaciones y consejos de acuerdo a su desempeño y a sus características cuantitativas y cualitativas. Ofreciendo un Servidor Web diferente al que está actualmente en uso, y además que cumpla con todos los requerimientos académicos y un mínimo consumo de recursos hardware. Incrementando la productividad y rendimiento de los equipos de computación en beneficio de la materia Aplicaciones Web, que es impartida en la EIS.

Cabe destacar el hecho de que la palabra servidor identifica tanto al programa como a la máquina en la que dicho programa se ejecuta. Para la realización de esta investigación se han elegido 3 servidores Web, de distribución libre, mismos que son:

1. Apache Http Server
2. Lighttpd Server
3. Cherokee Http Server

Un sistema operativo que soporta este tipo de tecnologías es **Linux**, del cual hemos escogido la Distribución CentOS por ser software libre, además por ser una tendencia institucional.

Las principales herramientas metodológicas usadas en esta investigación son: la observación, experimentación, métodos de la estadística descriptiva, Diseño de Experimentos, Histogramas, Diagramas de caja y bigote, visitas a sitios Web, entrevistas y la revisión y análisis de temas que complementan la información.

1.5. Planteamiento de la Hipótesis

Con el cumplimiento exitoso de los objetivos, debemos ahora plantear, por decirlo de alguna manera el resultado inesperado, puesto que es lo que arrojará la investigación:

La hipótesis planteada:

El estudio estadístico del rendimiento de servidores Web, de distribución libre, aplicado en los servidores de la Escuela de Ingeniería en Sistemas – FIE (ESPOCH), contribuirá a optimizar la plataforma de hardware disponible.

CAPÍTULO II

MARCO TEÓRICO

2.1. Prefacio

La Web ha dejado de ser una inmensa “biblioteca” de páginas estáticas para convertirse en un servicio que nos permite acceder a multitud de prestaciones y funciones, a infinidad de servicios, programas, tiendas, etc.

2.2. Breve historia de la WWW

World Wide Web (o la "Web") o **Red Global Mundial** es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet. Con un navegador Web, un usuario visualiza páginas Web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.

La Web fue creada alrededor de 1990 por el inglés Tim Berners-Lee y el belga Robert Cailliau mientras trabajaban en el CERN (Centro Europeo de Investigación Nuclear) en Ginebra, Suiza. Desde entonces, Berners-Lee ha jugado un papel activo guiando el desarrollo de estándares Web (como los lenguajes de marcado con los que se crean las páginas Web), y en los últimos años ha abogado por su visión de una Web Semántica.**[21]**

2.2.1. Fundamentos de la Web

El éxito espectacular de la Web se basa en dos puntales fundamentales: el protocolo HTTP y el lenguaje HTML. Uno permite una implementación simple y sencilla de un sistema de comunicaciones que nos permite enviar cualquier tipo de ficheros de una forma fácil, que simplifica el funcionamiento del servidor, permitiendo que servidores poco potentes atiendan miles de peticiones y simplificando los costes de despliegue. El otro nos proporciona un mecanismo de composición de páginas enlazadas simple y fácil, altamente eficiente y de uso muy simple. **[16]**

2.2.1.1. El protocolo HTTP

El protocolo HTTP (*Hypertext Transfer Protocol*) es el protocolo base de la WWW. Es un protocolo simple, orientado a conexión y sin estado. Es un protocolo orientado a conexión ya que emplea para su funcionamiento un protocolo de comunicaciones (TCP, Transport Control Protocol) de modo conectado, un protocolo que establece un canal de comunicaciones de extremo a extremo (entre el cliente y el servidor) por el que pasa el flujo de bytes que constituyen los datos a transferir, en contraposición a los protocolos de datagrama o no orientados a conexión que dividen los datos en pequeños paquetes (datagramas) y los envían, pudiendo llegar por vías diferentes del servidor al cliente. **[14]**

El protocolo no mantiene estado, es decir, cada transferencia de datos es una conexión independiente de la anterior, no manteniendo ninguna relación entre ellas. Esto es así hasta el punto de que para transferir una página Web debemos enviar el código HTML del texto así como las imágenes que la componen, pues en la especificación inicial de HTTP, la 1.0, se abrían y usaban tantas conexiones como componentes tenia la página, transfiriéndose por cada conexión un componente (el texto de la página o cada una de las imágenes).[1]

Existe una variante de HTTP llamada HTTPS (S por *secure*) que utiliza el protocolo de seguridad SSL (*Secure Socket Layer*) para cifrar y autenticar el tráfico entre cliente y servidor, siendo ésta muy usada por los servidores Web de comercio electrónico o que contengan información personal o confidencial.[2]

El funcionamiento esquemático de HTTP es el siguiente; el cliente establece una conexión TCP hacia el servidor, hacia el puerto HTTP (o el indicado en la dirección de conexión), envía un comando HTTP de petición de un recurso (junto con algunas cabeceras informativas) y por la misma conexión el servidor responde con los datos solicitados así como algunas cabeceras informativas.

---

[1] HTTP utiliza el puerto 80 (equivalente de alguna forma al identificador de conexión o de servicio TCP) para todas las conexiones por defecto (podemos utilizar otros puertos diferentes del 80).
[2] HTTPS utiliza por defecto el puerto 443.

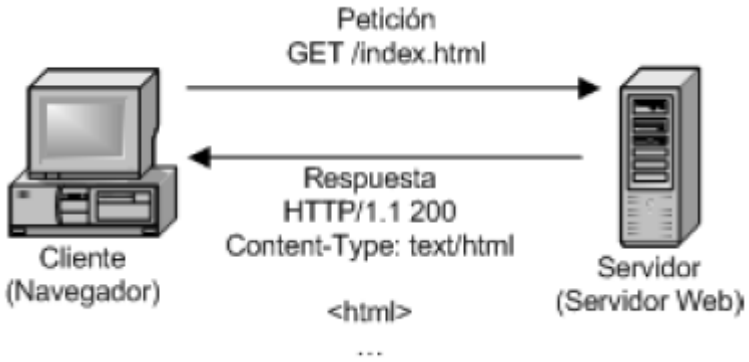


Ilustración II - 1.- Funcionamiento del Protocolo Http

El protocolo define además como codificar el paso de parámetros entre páginas, el tunelizar las conexiones (para sistemas de firewall), define la existencia de servidores intermedios de *cache*, etc.

Las directivas de petición de información que define HTTP 1.1 (la versión considerada estable y al uso) son:

GET Petición de recurso.

POST Petición de recurso pasando parámetros.

HEAD Petición de datos sobre recurso.

PUT Creación o envío de recurso.

DELETE Eliminación de recurso.

TRACE Devuelve al origen la petición tal como se ha recibido en el receptor, para depurar errores.

OPTIONS Sirve para comprobar las capacidades del servidor.

CONNECT Reservado para uso en servidores intermedios capaces de funcionar como túneles.

Detallaremos a continuación algunos de estos comandos, ya que su comprensión es fundamental para el desarrollo de aplicaciones Web.

Destacar que todos los recursos a ser servidos mediante HTTP deberían ser referenciados mediante una URL (Universal Resource Locators).

2.2.1.2. El lenguaje HTML

El otro puntal del éxito del WWW ha sido el lenguaje HTML (*HyperText Mark-up Language*). ´ Este es un lenguaje de marcas (se utiliza insertando marcas en el interior del texto) que nos permite representar de forma rica el contenido, así como referenciar otros recursos (imágenes, etc.), enlaces a otros documentos (la característica más destacada del WWW), mostrar formularios para luego procesarlos, etc.

El lenguaje HTML actualmente se encuentra en la versión 4.01 y empieza a proporcionar funcionalidades más avanzadas para crear páginas más ricas en contenido. Además se ha definido una especificación compatible con HTML, el XHTML (Extensible Hypertext Markup Language) que se suele definir como una versión XML validable de HTML, proporcionándonos un XML Schema contra el que validar el documento para comprobar si está bien formado, etc.

2.3. Historia de las aplicaciones Web

Una aplicación Web es un sistema informático que los usuarios utilizan accediendo a un servidor Web a través de Internet o de una intranet. Las aplicaciones Web son populares debido a la practicidad del navegador Web como cliente ligero. La facilidad para actualizar y mantener aplicaciones Web sin distribuir e instalar software en miles de potenciales clientes es otra razón de su popularidad. Aplicaciones como los webmails, wikis, weblogs, tiendas en línea son ejemplos bien conocidos de aplicaciones Web.

El modo de crear los documentos HTML ha variado a lo largo de la corta vida de las tecnologías Web pasando desde las primeras páginas escritas en HTML almacenadas en un fichero en el servidor Web hasta aquellas que se generan al vuelo como respuesta a una acción del cliente y cuyo contenido varía según las circunstancias.

Además, el modo de generar páginas dinámicas ha evolucionado, desde la utilización del CGI, Common Gateway Interface, hasta los servlets pasando por tecnologías tipo JavaServer Pages. Todas estas tecnologías se encuadran dentro de aquellas conocidas como Server Side, ya que se ejecutan en el servidor Web.

Otro aspecto que completa el panorama son las inclusiones del lado del cliente, Client Side, que se refieren a las posibilidades de que las páginas lleven incrustado código que se ejecuta en el cliente, como por ejemplo JavaScript y programas Java.

El esquema general de la situación se puede ver en la Ilustración II.2 en el cual podemos observar el Esquema general de las tecnologías Web, donde se muestran cada tipo de tecnología involucrada en la generación e interacción de documentos Web.

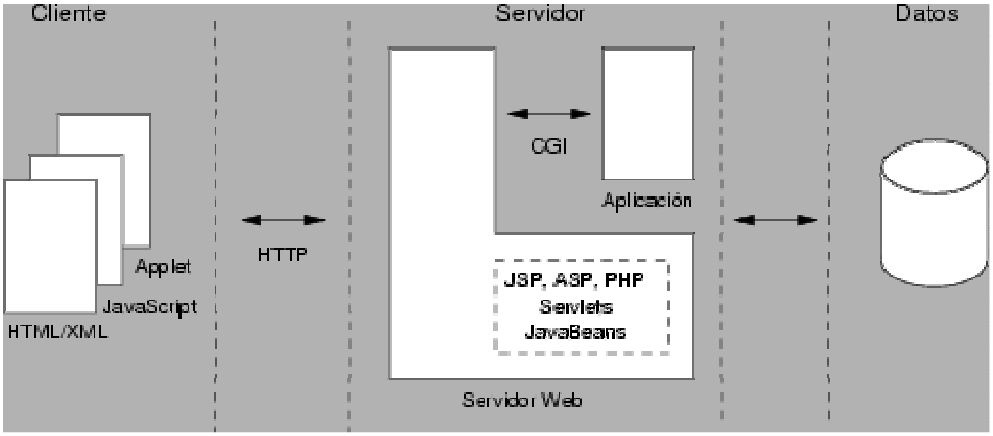


Ilustración II - 2.- Esquema General de las tecnologías

2.3.1. Historia

En los primeros tiempos de la computación cliente-servidor, cada aplicación tenía su propio programa cliente y su interfaz de usuario, estos tenían que ser instalados separadamente en cada estación de trabajo de los usuarios. Una mejora al servidor, como parte de la aplicación, requería típicamente una mejora de los clientes instalados en cada una de las estaciones de trabajo, añadiendo un costo de soporte técnico y disminuyendo la eficiencia del personal.

En contraste, las aplicaciones Web generan dinámicamente una serie de páginas en un formato estándar, soportado por navegadores Web comunes como HTML o XHTML. Se utilizan lenguajes interpretados del lado del cliente, tales como JavaScript, para añadir elementos dinámicos a la interfaz de usuario. Generalmente cada página Web individual es enviada al cliente como un documento estático, pero la secuencia de páginas provee de una experiencia interactiva.

2.3.2. Desarrollo de Aplicaciones Web

Con la introducción de Internet y del Web en concreto, se han abierto infinidad de posibilidades en cuanto al acceso a la información desde casi cualquier sitio. **[10]** Esto representa un desafío a los desarrolladores de aplicaciones, ya que los avances en tecnología demandan cada vez aplicaciones más rápidas, ligeras y robustas que permitan utilizar el Web.

El viejo CGI ha cumplido con el propósito de añadir interactividad a las páginas Web pero sus deficiencias en el desarrollo de aplicaciones y en la escalabilidad de las mismas ha conducido al desarrollo de APIs específicos de servidor como Active Server Pages, ASP, y PHP, que son más eficientes que su predecesor CGI.

Para aprovechar el potencial de estas tecnologías y ofertar una solución de servidor más extensible y portable, Sun ha desarrollado la tecnología llamada servlet. Los servlets Java son muy eficientes, debido al esquema de threads en el que se basan y al uso de una arquitectura estándar como la JVM, Java Virtual Machine.

Otra nueva tecnología viene a sumarse a las que extienden la funcionalidad de los servidores Web, llamada JavaServer Pages, JSP. Los JSP permiten unir HTML, aplicaciones Java, y componentes como las JavaBeans creando una página Web especial que el servidor Web compila dinámicamente en un servlet la primera vez que es llamada.

2.3.2.1. Arquitectura Web

Ante tal aluvión de posibilidades, conviene repasar algunos aspectos básicos de la arquitectura Web. **[5]**

Para abrir una página Web en un navegador, normalmente se teclea el correspondiente URL o se pica en el hiperenlace oportuno. Una vez que se solicita esta petición mediante el protocolo HTTP y la recibe el servidor Web, éste localiza la página Web en su sistema de ficheros y la envía de vuelta al navegador que la solicitó, según se muestra en la Ilustración II.3. Arquitectura Web básica

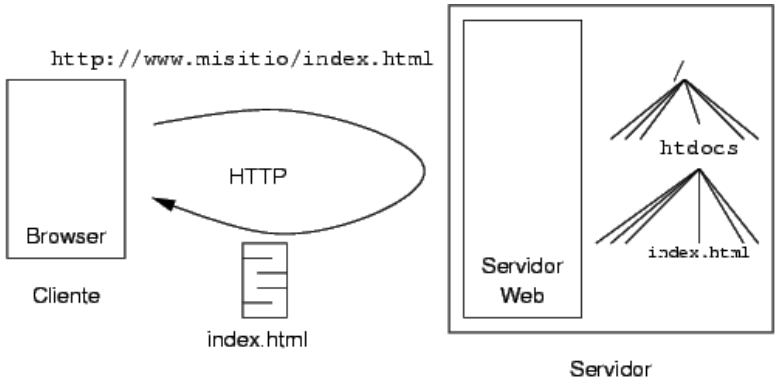


Ilustración II - 3.- Arquitectura Web Básica

2.3.2.2. El Navegador Web (Browser) [13]

El navegador puede considerarse como una interfaz de usuario universal. Dentro de sus funciones están la petición de las páginas Web, la representación adecuada de sus contenidos y la gestión de los posibles errores que se puedan producir.

Para todo esto, los fabricantes de navegadores les han dotado de posibilidades de ejecución de programas de tipo script, con modelos de objetos que permiten manipular los contenidos de los documentos. Estos lenguajes de programación son VBScript, JScript (ambas de Microsoft) y JavaScript (de Netscape), y proporcionan las soluciones llamadas del lado del cliente, client side y permiten realizar validaciones de datos recogidos en las páginas antes de enviarlos al servidor y proporcionan un alto grado de interacción con el usuario dentro del documento.

Otras de las posibilidades de los navegadores es la gestión del llamado HTML dinámico (Dinamic HTML, DHTML). Éste está compuesto de HTML, hojas de estilo en cascada, (Cascade Style Sheets, CSS), modelo de objetos y scripts de programación que permiten formatear y posicionar correctamente los distintos elementos HTML de las páginas Web, permitiendo un mayor control sobre la visualización de las páginas.

En esta línea, los navegadores han ido un poco más allá y permiten las visualizaciones de documentos XML (eXtensible Markup Language) después de haber sido transformado adecuadamente a HTML por las hojas de estilo extensibles (eXtensible Style Sheets, XSL). De

esta manera se puede elegir visualizar ciertos elementos y otros no, dependiendo de las circunstancias.

Además, los navegadores permiten la ejecución de aplicaciones dentro de los documentos mostrados. Las dos posibilidades más populares son la tecnología ActiveX y los applets Java. Los applets Java son pequeños programas que se descargan del servidor Web y se ejecutan en la JVM del navegador.

2.3.2.3. El Servidor http

El servidor http o servidor Web es un programa que corre sobre el servidor que escucha las peticiones HTTP que le llegan y las satisface. Dependiendo del tipo de la petición, el servidor Web buscará una página Web o bien ejecutará un programa en el servidor. De cualquier modo, siempre devolverá algún tipo de resultado HTML al cliente o navegador que realizó la petición. **[15]**

El servidor Web va a ser fundamental en el desarrollo de las aplicaciones del lado del servidor, server side applications, que vayamos a construir, ya que se ejecutarán en él.

2.3.2.4. Aplicaciones Multinivel

Al hablar del desarrollo de aplicaciones Web resulta adecuado presentarlas dentro de las aplicaciones multinivel. Los sistemas típicos cliente/servidor pertenecen a la categoría de las aplicaciones de dos niveles. La aplicación reside en el cliente mientras que la base de datos se encuentra en el servidor. En este tipo de aplicaciones el peso del cálculo recae en el cliente, mientras que el servidor hace la parte menos pesada, y eso que los clientes suelen ser máquinas menos potentes que los servidores. Además, está el problema de la actualización y el mantenimiento de las aplicaciones, ya que las modificaciones a la misma han de ser trasladada a todos los clientes. **[3]**

Para solucionar estos problemas se ha desarrollado el concepto de arquitecturas de tres niveles: interfaz de presentación, lógica de la aplicación y los datos.

La capa intermedia es el código que el usuario invoca para recuperar los datos deseados. La capa de presentación recibe los datos y los formatea para mostrarlos adecuadamente. Esta división entre la capa de presentación y la de la lógica permite una gran flexibilidad a la hora de construir aplicaciones, ya que se pueden tener múltiples interfaces sin cambiar la lógica de la aplicación.

La tercera capa consiste en los datos que gestiona la aplicación. Estos datos pueden ser cualquier fuente de información como una base de datos o documentos XML.

Convertir un sistema de tres niveles a otro multinivel es fácil ya que consiste en extender la capa intermedia permitiendo que convivan múltiples aplicaciones en lugar de una sola (véase la Ilustración II. Arquitectura Multinivel).

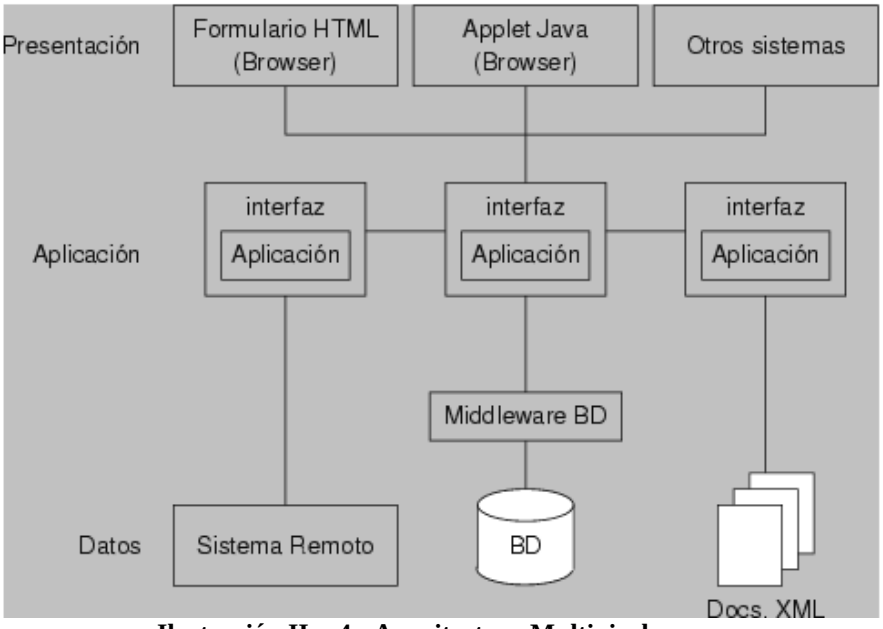


Ilustración II - 4.- Arquitectura Multinivel

La arquitectura de las aplicaciones Web suelen presentar un esquema de tres niveles (véase la Ilustración II.5. Arquitectura Web de tres niveles.). El primer nivel consiste en la capa de presentación que incluye no sólo el navegador, sino también el servidor Web que es el responsable de dar a los datos un formato adecuado. El segundo nivel está referido

habitualmente a algún tipo de programa o script. Finalmente, el tercer nivel proporciona al segundo los datos necesarios para su ejecución.

Una aplicación Web típica recogerá datos del usuario (primer nivel), los enviará al servidor, que ejecutará un programa (segundo y tercer nivel) y cuyo resultado será formateado y presentado al usuario en el navegador (primer nivel otra vez).

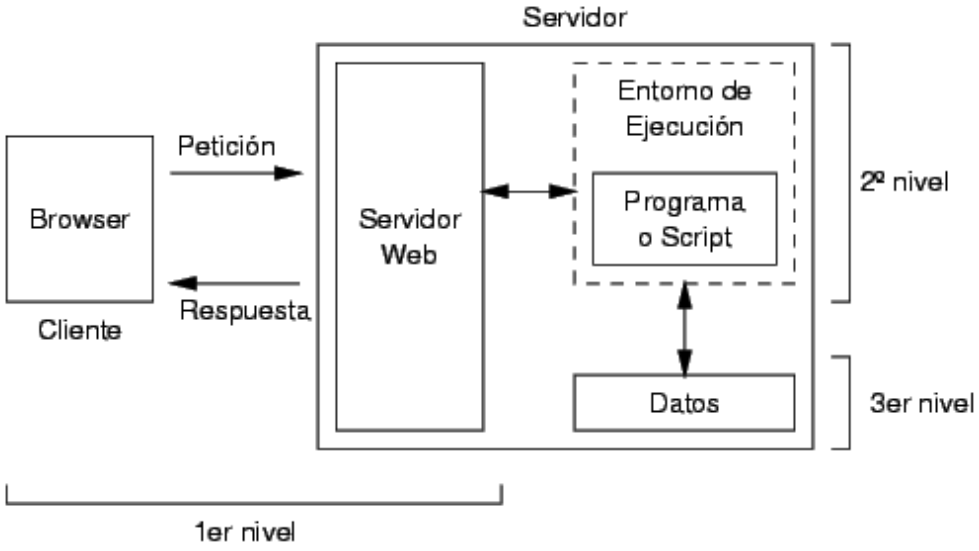


Ilustración II - 5.- Arquitectura Web de tres niveles

2.3.2.5. Ejemplos de Herramientas de Edición Web

Existen multitud de herramientas que permiten desarrollar portales Web con aspecto y prestaciones profesionales, aunque la mayoría de ellos requiere de una fase inicial de entrenamiento de duración variable según los casos.**[11]**

Entre los más utilizados están:

- Macromedia (DreamWeaver, Flash)
- Microsoft FrontPage
- Microsoft Studio .NET en todas las versiones conocidas
- PHP
- MapServer **[4]**

De los tres primeros hay que decir que integran en un entorno amigable casi todos los elementos relacionados con el desarrollo de páginas Web, además de aportar soluciones a la gestión y el mantenimiento de sitios Web grandes y complejos, y la interacción con bases de datos, El problema en su uso es que no resultan fáciles de manejar y requieren ciertos conocimientos y son de pago.

El tercer caso es el editor de páginas Web que proporciona el navegador Netscape. No es más que un pequeño editor de páginas sin más posibilidades, pero lo fácil de su uso hace que sea una buena herramienta para iniciarse en la creación y en el mantenimiento de portales Web. El precio a pagar es que algunas cosas se han de seguir realizando a mano, como la conexión con bases de datos y la gestión de formularios.

El cuarto caso es PHP que se lo encuentra tanto para Linux como Windows y nos permite crear páginas Web sencillas, dinámicas y con acceso a Base de Datos, y MapServer es un ambiente de desarrollo de código abierto para construir aplicaciones Web espaciales construido sobre otros sistemas de código abierto o freeware y que corre tanto bajo plataformas UNIX/Linux como sobre plataforma Windows 95 o superior.

2.4. Conceptos básicos del servidor Web

En informática, un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. Servicios habituales son los servicios de archivos, que permiten a los usuarios almacenar y acceder a los archivos de un ordenador y los servicios de aplicaciones, que realizan tareas en beneficio directo del usuario final. Este es el significado original del término. Es posible que un ordenador cumpla simultáneamente las funciones de cliente y de servidor. El término servidor ahora también se utiliza para referirse al ordenador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar.

Este uso dual puede llevar a confusión. Por ejemplo, en el caso de un servidor Web, este término podría referirse a la máquina que almacena y maneja los sitios Web, y en este sentido es utilizada por las compañías que ofrecen hosting u hospedaje. Los archivos para cada sitio

de Internet se almacenan y se ejecutan en el servidor. Hay muchos servidores en Internet y muchos tipos de servidores, pero comparten la función común de proporcionar el acceso a los archivos y servicios. Un servidor sirve información a los ordenadores que se conecten a él. Cuando los usuarios se conectan a un servidor pueden acceder a programas, archivos y otra información del servidor.

Los servidores Web, servidores de correo y servidores de bases de datos son a lo que tiene acceso la mayoría de la gente al usar Internet. Algunos servidores manejan solamente correo o solamente archivos, mientras que otros hacen más de un trabajo, ya que un mismo ordenador puede tener diferentes programas de servidor funcionando al mismo tiempo.

Los servidores se conectan a la red mediante una interfaz que puede ser una red verdadera o mediante conexión vía línea telefónica o digital.

Un servidor Web es un programa que atiende y responde las diversas peticiones que le realizan los navegadores, proporcionándoles los recursos que solicitan mediante el protocolo HTTP o el protocolo HTTPS (la versión segura, cifrada y autenticada, de HTTP). Un servidor Web básico tiene un esquema de funcionamiento muy sencillo, ejecutando de forma infinita el bucle siguiente:

1. Espera peticiones en el puerto TCP asignado (el estándar para HTTP es el 80).
2. Recibe una petición.
3. Busca el recurso en la cadena de petición.
4. Envía el recurso por la misma conexión por donde ha recibido la petición.
5. Vuelve al punto 2.

Un servidor Web que siguiese el esquema anterior cumpliría los requisitos básicos de los servidores HTTP, aunque, eso sí, sólo podría servir ficheros estáticos.

A partir del esquema anterior se han diseñado y construido todos los programas servidores de HTTP que existen, variando sólo qué tipo de peticiones (páginas estáticas, CGI, Servlets, etc.) pueden atender, en si son o no multi-proceso, multi-hilados, etc.

A continuación detallaremos algunas de las características principales de los servidores Web, que extienden, obviamente el esquema anterior.

2.5. Servidor Web Apache

Ilustración II - 6.- Logotipo Servidor http Apache

Hoy en día, el servidor Web Apache es el servidor más usado de Internet, con una utilización del 67% aproximadamente.**[12]**

La primera aparición de Apache fue en Abril de 1995. Este servidor se sigue desarrollando “en Internet” como un proyecto de Software libre. Las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad del desarrollo distribuido. Todo el código fuente de Apache está escrito en C, con un total aproximado de 185.000 líneas de código.

El servidor HTTP Apache es un servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/4.3 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 4.3, pero más tarde fue reescrito por completo. Su nombre se debe a que originalmente Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, a patchy server (un servidor "parcheado").

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración. Apache tiene amplia aceptación en la red: en el 2005, Apache fue el servidor HTTP más usado, siendo el servidor HTTP del 70% de los sitios Web en el mundo y creciendo aún su cuota de mercado (estadísticas históricas y de uso diario proporcionadas por Netcraft) [3].

El núcleo 2.x de Apache tiene varias mejoras clave sobre el núcleo de Apache 1.x. Estas mejoras incluyen threads de UNIX, mejor soporte para plataformas no Unix (como Windows), un nuevo API, y soporte de IPv6.

Módulos

La arquitectura del servidor Apache es muy modular. El servidor consta de un sección core y mucha de la funcionalidad que podría considerarse básica para un servidor Web es provista por módulos.

2.5.2. Instalación de Apache

2.5.2.1. Descargar

Puede descargar Apache desde la sección de descargas del sitio Web de Apache el cual tiene varios mirrors. Para la mayoría de los usuarios de Apache que tienen sistemas tipo Unix. El proceso de compilación (descrito más abajo) es fácil, y permite adaptar el servidor Apache a sus necesidades. Además, las versiones disponibles en archivos binarios no están siempre actualizadas con las últimas modificaciones en el código fuente. Si se descarga un binario, siga las instrucciones contenidas en el archivo INSTALL.bindist incluido en la distribución

Después de la descarga, es importante que verifique que el archivo descargado del servidor HTTP Apache está completo y sin modificaciones. Esto puede hacerlo comparando el archivo descargado (.tgz) con su firma PGP. Instrucciones detalladas de cómo hacer esto están disponibles en la sección de descargas junto con un ejemplo de cómo usar PGP.

[3] NetCraf – Anexo 7

2.5.2.2. Descomprimir

Extraer el código fuente del archivo .tgz que acabada de descargar es muy fácil. Ejecute los siguientes comandos:

```
$ gzip -d httpd-2_1_NN.tar.gz
$ tar xvf httpd-2_1_NN.tar
```

Estos comandos crearán un nuevo directorio dentro del directorio en el que se encuentra y que contendrá el código fuente de la distribución. Debe cambiarse a ese directorio con cd para proceder a compilar el servidor Apache.

2.5.3. Configuración de Apache

El siguiente paso es configurar la estructura de directorios para su plataforma y sus necesidades personales. Esto se hace usando el script configure incluido en el directorio raíz de la distribución que acaba de descargar. (Los desarrolladores que se descarguen la versión del CVS de la estructura de directorios necesitarán tener instalados autoconf y libtool, y necesitarán ejecutar buildconf antes de continuar con los siguientes pasos. Esto no es preciso para las versiones oficiales.)

Para configurar la estructura de directorios a partir del código fuente usando las opciones por defecto, solo tiene que ejecutar ./configure. Para cambiar las opciones por defecto, configure acepta una serie de variables y opciones por la línea de comandos.

La opción más importante es --prefix que es el directorio en el que Apache va a ser instalado después, porque Apache tiene que ser configurado para el directorio que se especifique para que funcione correctamente. Es posible lograr un mayor control del lugar donde se van a instalar los ficheros de Apache con otras opciones de configuración.

En este momento, puede especificar que características o funcionalidades quiere incluir en Apache activando o desactivando módulos. Apache viene con una selección básica de módulos incluidos por defecto. Se pueden activar otros módulos usando la opción --enable-*module*, donde *module* es el nombre del módulo sin el mod_ y convirtiendo los guiones bajos

que tenga en guiones normales. También puede optar por compilar módulos como objetos dinámicos compartidos (DSOs) -- que pueden ser activados o desactivados al ejecutar -- usando la opción --enable-*module*=shared. De igual manera, puede desactivar alguno de los módulos que vienen por defecto en la selección básica con la opción --disable-*module*. Tenga cuidado cuando use estas opciones, porque configure no le avisará si el módulo que especifica no existe; simplemente ignorará esa opción.

Además, a veces es necesario pasarle al script configure información adicional sobre donde su compilador, librerías o ficheros de cabecera. Esto se puede hacer, tanto pasando variables de entorno, como pasándole opciones a configure a través de la línea de comandos. Para más información, consulte el Manual del script configure.

Para que se haga una idea sobre las posibilidades que tiene, aquí tiene un ejemplo típico que configura Apache para la ruta /sw/pkg/apache con un compilador y unos flags determinados, y además, con dos módulos adicionales mod_rewrite y mod_speling para cargarlos después a través del mecanismo DSO:

```
$ CC="pgcc" CFLAGS="-O2" \
./configure --prefix=/sw/pkg/apache \
--enable-rewrite=shared \
--enable-speling=shared
```

Cuando se ejecuta configure se comprueban que características o funcionalidades están disponibles en su sistema y se crean los Makefiles que serán usados luego para compilar el servidor. Esto tardará algunos minutos.

2.5.3.2. Inicio de Apache

Si el puerto especificado en la directiva Listen del fichero de configuración es el que viene por defecto, es decir, el puerto 80 (o cualquier otro puerto por debajo del 1024), entonces es necesario tener privilegios de usuario root (superusuario) para iniciar Apache, de modo que pueda establecerse una conexión a través de esos puertos privilegiados. Una vez que el servidor Apache se ha iniciado y ha completado algunas tareas preliminares, tales como abrir sus ficheros log, lanzarán varios procesos, procesos *hijo*, que hacen el trabajo de escuchar y

atender las peticiones de los clientes. El proceso principal, httpd continúa ejecutándose como root, pero los procesos hijo se ejecutan con menores privilegios de usuario. Esto lo controla el Módulo de MultiProcesamiento (MPM) seleccionado.

```
# /etc/init.d/httpd restart
```

Si todo va bien durante el arranque, la sesión de Terminal se suspenderá un momento y volverá a estar activa casi inmediatamente. Esto quiere decir que el servidor está activo y funcionando. Puede usar su navegador para conectarse al servidor y ver la página de prueba que hay en el directorio DocumentRoot y la copia local de esta documentación a la que se puede acceder desde esa página.

2.6. Servidor Web Lighttpd

Ilustración II - 7.- Logotipo Servidor http Lighttpd

Lighttpd es un servidor desarrollado inicialmente por Jan Kneschke, actualmente mantenido por la comunidad de desarrolladores de lighttpd junto con su desarrollador.

Lighttpd es un servidor Web diseñado para ser rápido, seguro, flexible, y fiel a los estándares. Está optimizado para entornos donde la velocidad es muy importante, y por eso consume menos CPU y memoria RAM que otros servidores. Por todo lo que ofrece, lighttpd es apropiado para cualquier servidor que tenga problemas de carga.

Seguridad, velocidad, cumplimiento, y la flexibilidad - todo esto describe a Lighttpd (pron. lighty), que es rápidamente la redefinición de la eficiencia de un servidor Web, ya que está

concebido y optimizado para entornos de alto rendimiento. Con una pequeña huella de memoria en comparación con otros servidores Web, gestión eficaz de la carga de cpu, y avanzado conjunto de características (FastCGI, SCGI, autentificación, compresión de salida, URL-reescritura y muchos más) Lighttpd es la solución perfecta para cada servidor que está sufriendo problemas de carga.

Lighttpd es software libre y se distribuye bajo la licencia BSD. Funciona en GNU/Linux y otros sistemas operativos tipo UNIX.

2.6.1. Lenguajes de servidor

Lighttpd permite comunicarse con programas externos mediante FastCGI o SCGI, que son mejoras al CGI original (también soportado). De esta forma, se pueden usar programas en prácticamente cualquier lenguaje de programación.

Tiene una importancia especial PHP, para el que se han hecho mejoras específicas. También es habitual combinarlo con Ruby on Rails

Características

- Virtual hosting (alojar varios dominios en la misma IP)
- CGI, SCGI y FastCGI
- Soporte para PHP, Ruby, y otros
- Entorno chroot
- Cifrado SSL
- Compresión (gzip, bzip2, ...)
- Autenticación (LDAP, htpasswd, otros)
- Server Side Includes
- Consumo de memoria constante
- Redirecciones HTTP, y reescrituras de URL
- Puede enviar partes de un fichero (rangos)
- Puede usar select() o poll()

- También permite otros sistema de *notificación de eventos* como kqueue y epoll
- Hace estadísticas mediante RRDtool
- Muestra un listado de ficheros cuando se entra a un directorio sin *index.html*
- Redirección condicional
- Permite módulos externos
- Cache Meta Language
- Acepta parte de WebDAV

2.6.2. Instalación de Lighttpd

Bajamos los archivos Lighttpd -**1.3.16-1.2.el4.rf.i386.rpm y lighttpd-fastcgi-1.3.16-1.2.el4.rf.i386.rpm**, luego empezamos la instalación:

```
# rpm -Uhv lighttpd-1.3.16-1.2.el4.rf.i386.rpm        (Instalador del Webserver)
# rpm -Uhv lighttpd-fastcgi-1.3.16-1.2.el4.rf.i386.rpm     (Instalador para soporte de php)
```

Una vez que tenemos instalados los dos paquetes, procedemos a arrancar nuestro Web Server para saber que está funcionando.

```
# /etc/init.d/lighttpd start     (Iniciamos nuestro Web Server Lighttp)
```

Para saber que arranco bien nuestro servidor, ponemos lo siguiente en un browser como el mozilla:

"http://localhost/" y nos tiene que salir la página principal de Lighttp.

2.6.3. Configuración de Lighttpd

Ahora procedemos a configurar nuestro Web Server de la siguiente manera:

vi /etc/lighttpd/lighttpd.conf

Con este comando editamos el archivo principal de lighttp, descomentamos la línea que contenga "mod_fastcgi" para dar soporte PHP a nuestro servidor. Algo así:

```
## modules to load
# at least mod_access and mod_accesslog should be loaded
# all other module should only be loaded if really neccesary
# - saves some time
# - saves memory
server.modules              = (
#                               "mod_rewrite",
#                               "mod_redirect",
#                               "mod_alias",
```

```
                                  "mod_access",
#                                 "mod_cml",
#                                 "mod_trigger_b4_dl",
#                                 "mod_auth",
#                                 "mod_status",
#                                 "mod_setenv",
                                 "mod_fastcgi",  #Linea a descomentar
#                                 "mod_proxy",
#                                 "mod_simple_vhost",
#                                 "mod_evhost",
#                                 "mod_userdir",
#                                 "mod_cgi",
#                                 "mod_compress",
#                                 "mod_ssi",
#                                 "mod_usertrack",
#                                 "mod_expire",
#                                 "mod_secdownload",
#                                 "mod_rrdtool",
                                  "mod_accesslog" )
```

También cambiamos la ubicación de nuestro DocumentRoot:

```
server.document-root        = "/srv/www/lighttpd/"     ----> la
cambiamos por
server.document-root        = "/var/www/html/"
```

También modificamos las siguientes líneas o mejor dicho, las descomentamos:

```
#### fastcgi module
## read fastcgi.txt for more info
## for PHP don't forget to set cgi.fix_pathinfo = 1 in the php.ini
fastcgi.server             = ( ".php" =>
                               ( "localhost" =>
                                 (
                                   "socket" => "/var/run/lighttpd/php-
fastcgi.socket",
                                   "bin-path" => "/usr/bin/php-cgi"
                                 )
                               )
                             )
```

Ahora modificamos el archivo php.ini:

```
# vi /etc/php.ini
```

y agregamos al final del archivo la siguiente línea:

```
cgi.fix_pathinfo = 1
```

Guardamos los archivos que se modificaron y reiniciamos nuestro servidor.

```
# /etc/init.d/lighttpd restart
```

Ya tenemos configurado nuestro Webserver con Lighttpd.

2.7. Servidor Web Cherokee

Ilustración II - 8.- Logotipo Servidor http Cherokee

Es software libre, publicado bajo la licencia GPL (General Public License). Escrito en C, unas 50.000 líneas de código. Es un proyecto que desarrolla una nueva implementación de este tipo de aplicaciones.

El fin último de Cherokee es hacer un servidor con unas características de las que Apache carece debido a su diseño original. Su diseño es un híbrido que combina las características de servidores basados en sockets no bloqueantes con las de servidores basado en hilos, en busca de obtener beneficios de ambos modelos y minimizar los aspectos negativos.

Básicamente, su funcionamiento es el de un servidor que procesa varias peticiones en cada uno de sus hilos. Estos hilos ni se crean ni se destruyen, se generan cuando arranca el servidor y permanecen vivos hasta que termina su ejecución. En su implementación, se ha puesto especial interés en la velocidad, flexibilidad y capacidad de ser empotrado.

Velocidad: en el último benchmark (técnica utilizada para medir el rendimiento de un sistema, frecuentemente en comparación con algún parámetro de referencia) realizado hasta el momento, Cherokee fue cinco veces más rápido que Apache.

Flexibilidad: Cherokee, igual que Apache, dispone de un sistema para la carga dinámica de módulos basado en plug-ins, tanto para manejadores (handlers) como para codificadores (encoders) y sistema de logging.

Capacidad de ser empotrado dentro de otras aplicaciones. Todo el código se encuentra en una librería dinámica (libcherokee) que puede utilizar cualquier aplicación. El API de esta librería es muy sencillo; básicamente permite crear, configurar y ejecutar diferentes formas objetos "servidor".

Al igual que Apache, Cherokee escala a servidores SMP (Symetric Multi-Processing. Sistemas con varios procesadores) y a sistemas multihilo. Es capaz de manejar más de un hilo y en cada uno de ellos, de nuevo, volver a procesar conexiones mediante compartición de tiempo.

2.7.2. Arquitectura

Hay tres grupos de módulos cargables: handlers, encoders y validators. Handlers. Son manejadores de peticiones. Cuando el servidor procesa una petición, decide qué clase de manejador debe utilizar. Dependiendo del módulo, la respuesta será una u otra.

Tipos de manejadores: files (servir ficheros al cliente), redir (redireccionar peticiones), Cgi's, etc. Encoders. Módulos que implementan una funcionalidad de conversión de la información que se puede enviar a los clientes si estos lo soportan. Cherokee puede enviar ciertos elementos de una página Web comprimidos para dotar de mayor rapidez de respuesta. Para comprimir archivos el más útil es el de GZip.

Cherokee HTTP Server es un servidor http gratuito y multiplataforma escrito enteramente en C., no depende más allá de un estándar de librerías C. Es un servidor muy rápido a la vez que implementa las funcionalidades más usadas por los servidores http. Es posible embeberlo en dispositivos hardware. Hace uso de plug-ins para extender sus funcionalidades. Es extensible con bloqueos y soporta altas configuraciones para lectura de archivos o cadenas de caracteres, TLS/SSL (vía GNUTLS or OpenSSL), virtual hosts, autentificaciones, rasgos de caches amigables, PHP, manejo de errores y mucho más.

El proyecto Cherokee nació en 2001, desarrollado inicialmente por Álvaro López. Actualmente es mantenido y mejorado por el propio autor y una comunidad de desarrolladores, al igual que muchos proyectos de software libre.

2.7.3. Instalación de Cherokee

Primero nos bajamos el archivo **"cherokee-0.6.0b700-14.3.i386.rpm"** y lo instalamos.

Una vez que tenemos nuestro archivo, lo instalamos de la siguiente manera:

```
# rpm -Uhv cherokee-0.6.0b700-14.3.i386.rpm
```

Luego modificamos el archivo cherokee.conf

```
# vi /etc/cherokee.conf
```

modificamos el apartado del DocumentRoot:

```
vserver!default!document_root = /svr/www/htdocs/       ---------
> Cambiamos esta línea por   vserver!default!document_root =
/var/www/html/
```

2.7.4. Configuración de Cherokee

Modificamos la línea que contiene el path de nuestro intérprete PHP:

```
vserver!default!extensions!php!handler = fcgi
vserver!default!extensions!php!handler!balancer = round_robin
vserver!default!extensions!php!handler!balancer!type =
interpreter
vserver!default!extensions!php!handler!balancer!local1!host =
localhost:1234
vserver!default!extensions!php!handler!balancer!local1!env!PHP_F
CGI_CHILDREN = 5
vserver!default!extensions!php!handler!balancer!local1!interpret
er = /usr/lib/cgi-bin/php -b 1234 --------> cambiar solo esta
linea por

vserver!default!extensions!php!handler!balancer!local1!interpret
er = /usr/bin/php-cgi -b 1234
```

Una vez que terminamos de modificar, arrancamos nuestro servidor para probar nuestra nueva

configuración ejecutando el comando:

```
# cherokee
```

y nos muestra la página principal de Cherokee

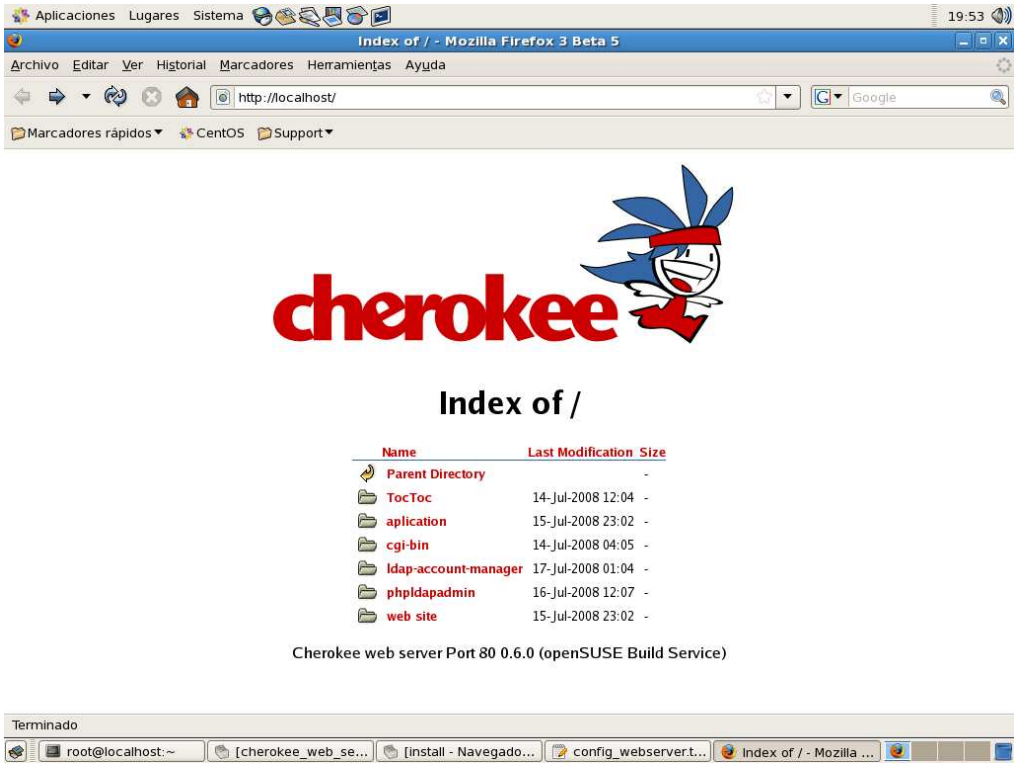


Ilustración II - 9.- Página de resultado al configurar el servidor Web Cherokee

CAPÍTULO III

COMPARATIVA DE SERVIDORES

3.1. Prefacio

La comparativa de Servidores nos encamina a utilizar el Benchmarking que significa medir la velocidad con la que un ordenador ejecuta una tarea, de forma que se puedan realizar comparaciones entre diferentes combinaciones de programas/componentes. Esta definición no tiene en cuenta la sencillez de utilización, estética o ergonomía o cualquier otro tipo de juicio subjetivo. **[24]**

El Benchmarking es una tarea repetitiva, tediosa, y hay debiéndose tener muy en cuenta como se lo realiza. Es muy normal que los resultados no sean los que uno espera y que estén sujetos a interpretación (puede que hoy en día ésta sea la parte más importante).

Para finalizar, el benchmarking trata con hechos y datos, no con opiniones ni aproximaciones.

3.2. ¿Por qué el benchmarking o comparativa es tan importante?

Cuando tenemos que hacernos las siguientes preguntas:

¿Cómo puedo maximizar el rendimiento con un presupuesto dado?

¿Cómo puedo minimizar costes manteniendo un nivel mínimo en el rendimiento?

¿Cómo puedo obtener la mejor relación calidad/precio (con un presupuesto o unas exigencias dadas)?

Tendremos que examinar, comparar o crear benchmarks. Minimizar costes sin tener que mantener un rendimiento en particular implica utilizar una máquina con partes desfasadas (aquel viejo 386SX-16 que está tirado en el garaje podría servir) y no necesita benchmarks, y

maximizar el rendimiento sin que importe el dinero no es una situación muy realista (a menos que quiera poner un Cray en su casa - la unidad de alimentación recubierta con cuero es bonita, ¿verdad?).

El *benchmarking* o *comparativa* de por si no tiene sentido, y es una pérdida de tiempo y dinero; solo tiene sentido como una parte de un proceso, esto es, si tiene que hacer una elección entre dos o más alternativas.

Normalmente otro parámetro a tener en cuenta en el proceso de decisión es el **coste**, pero también la disponibilidad, el servicio, la seguridad, estrategia o cualquier otra característica medible y racional que tenga que ver con un ordenador. Por ejemplo, cuando comparamos el rendimiento de diferentes versiones del núcleo de Linux, la **estabilidad** suele ser más importante que la velocidad. **[24]**

3.2.1. Consideraciones no válidas en la medida del rendimiento

Se pueden leer muy a menudo en los grupos de noticias y las listas de correo, pero aun así:

1. Reputación del fabricante (no se puede medir y es insensato).
2. Cuota de mercado del fabricante (insensato e irrelevante).
3. Parámetros irracionales (por ejemplo, supersticiones o prejuicios: ¿Compraría un procesador que se llame 131313ZAP pintado de rosa?)
4. Valor estimado (insensato, irracional y no se puede medir).
5. Cantidad de propaganda: creo que éste es la peor.

3.2.2. Procedimientos de medida e interpretación de resultados

Unas cuantas recomendaciones semi-obvias:

Primera y principal, **identifique el rendimiento objetivo**. ¿Qué es exactamente lo que trata de medir? ¿De qué forma la medida del rendimiento le ayudará después a tomar una decisión? ¿Cuánto tiempo y cuántos recursos está dispuesto a gastar en el proceso de medida? **[24]**

Utilice herramientas estándar. Use una versión, del núcleo, estable y actualizada, así como un gcc, libc, y una herramienta de medida del rendimiento actualizados y estándares.

Aislar una variable. Las medidas comparativas dan más información que las “absolutas”.

Verificar resultados. Ejecutar las pruebas unas cuantas veces y compruebe las fluctuaciones en los resultados, de haberlas. Las fluctuaciones inexplicables invalidarán sus resultados.

Si cree que el esfuerzo en la medición del rendimiento ha producido información útil, se recomienda **compartirla** de forma **breve** y **concisa**.

3.2.3. Entendiendo la elección de la herramienta

Las herramientas de medida sintéticas vs las de aplicaciones

Antes de perder el tiempo escogiendo entre todos los tipos de herramientas de medida, se debe hacer una elección básica entre las herramientas “sintéticas” y las de “aplicación”.

Las herramientas sintéticas están especialmente diseñadas para medir el rendimiento de un componente individual de un ordenador, normalmente llevando el componente escogido a su máxima capacidad. Un ejemplo de una herramienta sintética muy conocida es el **Whetstone**, programado originalmente en 1972 por Harold Curnow en FORTRAN y todavía ampliamente utilizada. El conjunto de herramientas Whetstone medirá el rendimiento de la unidad de punto flotante de la CPU.

La crítica principal que puede hacérsele a las medidas “sintéticas” es que no representan el rendimiento del sistema en las situaciones de la vida real.

Otro punto a tener en cuenta sobre las pruebas sintéticas es que, idealmente, deberían darnos información acerca de un aspecto **específico** del sistema que estamos comprobando, independientemente del resto de componentes: una prueba sintética sobre la E/S de las tarjetas Ethernet debería devolver el mismo resultado (o al menos similar) independientemente de si se ejecuta en un 386SX-16 con 4 MBytes de RAM o de si se ejecuta en un Pentium 200

MMX con 64 MBytes de RAM. Sin embargo, la prueba medirá el rendimiento global de la combinación CPU/placa base/Bus/tarjeta Ethernet/Subsistema de memoria/DMA: no es muy útil, ya que la variación en el procesador podría causar un impacto mayor en los resultados que la variación en la tarjeta de red Ethernet (naturalmente, esto es suponiendo que estamos utilizando la misma combinación de controlador/núcleo, que también pueden producir grandes cambios).

Para terminar, un error muy común es hacer la media de varias pruebas sintéticas y decir que esta media es una buena representación del rendimiento en la vida real de un sistema.

Por lo tanto, la tendencia en las pruebas de rendimiento es elegir las aplicaciones más comunes y utilizarlas para medir el rendimiento del sistema completo. Por ejemplo, **SPEC**, la organización sin ánimo de lucro que diseñó las herramientas SPECINT y SPECFP, ha lanzado un proyecto para un nuevo conjunto de herramientas basadas en aplicaciones. Pero de nuevo, sería muy raro que alguna herramienta comercial de medida del rendimiento, incluya código Linux.

Las pruebas sintéticas son válidas mientras comprenda sus propósitos y limitaciones. Las herramientas basadas en aplicaciones reflejarán mejor el rendimiento global del sistema, pero no hay ninguna disponible para Linux.

Pruebas de alto nivel vs. Pruebas de bajo nivel

Las pruebas de bajo nivel miden directamente el rendimiento de los componentes: el reloj de la CPU, los tiempos de la DRAM y de la caché SRAM, tiempo de acceso medio al disco duro, latencia, tiempo de cambio de pista, etc. Esto puede ser útil en caso de comprar un sistema y no se sabe con qué componentes viene, pero una forma mejor de comprobar estos datos es abrir la caja, hacer un listado con los nombres que pueda conseguir y obtener una hoja de características de cada componente encontrado (normalmente disponibles en la red). **[24]**

Otro uso de las pruebas de bajo nivel es comprobar que un controlador de núcleo ha sido correctamente instalado para un componente específico: si tiene la hoja de especificaciones del

componente, puede comparar los resultados de la prueba de bajo nivel con las especificaciones teóricas (las impresas).

Las pruebas de alto nivel están más enfocados a medir el rendimiento de la combinación componente/controlador/SO de un aspecto específico del sistema, como por ejemplo el rendimiento de E/S con ficheros, o el rendimiento de una determinada combinación de componentes/controlador/SO/aplicación, p.e. una prueba Apache en diferentes sistemas.

Por supuesto, todas las pruebas de bajo nivel son sintéticas. Las pruebas de alto nivel pueden ser sintéticas o de aplicación. A continuación se realiza en forma detallada la Comparativa de Servidores Web.

3.3. Comparativa de Servidores http

A la hora de instalar un servidor Web debe tenerse en cuenta una serie de consideraciones previas que tienen que ver con el soporte físico (hardware) sobre el que correrá el servidor: interfaces de red, sistema de almacenamiento SCSI con soporte RAID, memoria RAM de al menos 256 MB, procesador dependiente de si el contenido del sitio Web es más bien dinámico o estático, y sobre todo si tiene que acceder a diferentes bases de datos.

Y en cuanto al software, el sistema operativo será también una elección importante a la hora de montar un sitio Web, junto con el software servidor de Web.

3.3.1. Tipos de servidores http

Existen muchos parámetros para poder clasificar un servidor http, algunos de ellos son:

- **Coste del servidor**, existen aplicaciones comerciales, como por ejemplo Zeus Web Server y servidores gratuitos como por ejemplo Apache http Server.

- **Plataformas soportadas**, hay servidores diseñados para poder ejecutarse solo en un sistema operativo, como por ejemplo Internet Information Server para Microsoft Windows, o servidores multiplataforma como Apache http Server. También hay servidores diseñados

para una sola plataforma hardware (para Sparc de Solaris por ejemplo) o multiplataforma (para X86, PowerPC, Sparc,...).

- **Propósito del servidor**, podemos clasificar los servidores http según el uso al que se destine, hay servidores de propósito general dotados de muchas características, otros servidores están muy optimizados para alguna tarea en concreto (mínimo uso de recursos del sistema, gran rendimiento en contenido estático,...) a consta de renunciar a algunas características. Por último tenemos los servidores http embebidos, se suelen usar junto con hardware específico y están creados para ofrecer solamente lo necesario por ese hardware. Un ejemplo de este tipo de servidores http es la interfaz de configuración de muchos routers actuales.

- **Según su implantación**. Los servidores http suelen emplear concurrencia para servir a más de un cliente simultáneamente. Podemos clasificar los servidores http según la técnica que usen para implementar esta concurrencia: máquina de estados finitos, servidores basados en hebras, servidores basados en procesos, servidores mixtos.

3.4. Hardware empleado

Para el desarrollo de las distintas pruebas se ha procedido a la instalación y configuración de los tres servidores en tres tipos de Arquitecturas. ***Ver Anexo 3 - Arquitecturas***

3.5. Características implementadas

Comenzamos analizando detalladamente las características implementadas por cada uno de los servidores http. Esto nos servirá para conocer en profundidad a cada uno de los servidores http.

3.5.1. Visión General

	Creador	Coste	Código Abierto	Licencia
Apache2	Apache Software Fundation	0	Si	Apache License

Lighttpd	Jan Kneschke	0	Si	Variante de la licencia BSD
Cherokee	Álvaro López Ortega	0	Si	GPL

Tabla III - 1.- Servidores Web y Licencias
Fuente: Direcciones Web de cada servidor

Ahora pasamos al soporte para las distintas plataformas de cada servidor http.

	Windows	**MacOsX**	**Linux**	**BSD**	**Solaris**
Apache2	Si	Si	Si	Si	Si
Lighttpd	Si	Si	Si	Si	Si
Cherokee	Si	Si	Si	Si	Si

Tabla III - 2.- Plataformas de cada servidor http[4]
Fuente: Direcciones Web de cada servidor

Todos los servidores evaluados presentan compatibilidad con las plataformas más usadas, lo cual supone muchas ventajas, ya que podemos cambiar de sistema operativo y de plataforma sin tener que cambiar de servidor http.

3.5.1.1. Características de Seguridad

La siguiente tabla presenta las características de seguridad más destacadas y las implementadas por cada servidor.

	Autentificación Básica	**Autentificación Digest**	**Https**	**Virtual Host***
Apache2	Si	Si	Si	Si
Lighttpd	Si	Si	Si	Si
Cherokee	Si	Si	Si	Si

Tabla III - 3.- Seguridades por cada servidor[5]
Fuente: Direcciones Web de cada servidor

La seguridad es un elemento muy importante en nuestro servidor http, la posibilidad de autentificarse con el servidor de forma segura es muy deseable, sin tener que requerir de otras soluciones (programas escritos en lenguajes del lado del servidor por ejemplo).

El uso de https (http seguro) es imprescindible si en la Web se solicitan o se envían datos sensibles, cuya pérdida o manipulación constituya un riesgo importante. Se destaca el

[4] http://www.lighttpd.net , http://apache.org , http://cherokee.org
[5] http://www.lighttpd.net , http://apache.org , http://cherokee.org

servidor Apache2, principalmente por el hecho de su antigüedad con respecto a Lighttpd y Cherokee. También influye el gran número de programadores involucrados en su desarrollo y el hecho de que se emplee en muchos más sitios que los otros dos servidores mencionados. Un error de seguridad siempre se descubrirá antes si el programa es usado por más personas y se solucionará también antes ya que hay un mayor número de programadores en su desarrollo.

La posibilidad de ejecutar varios sitios Web en un mismo servidor no hace más seguro un servidor. Se podría argumentar lo contrario, es decir, a mayor número de sitios Web ejecutándose en un servidor es más probable que un fallo en alguno de los sitios Web pueda comprometer el sistema general. Sin embargo normalmente la ganancia de prestaciones y de aprovechamiento del sistema hace necesario alojar varios sitios en un mismo servidor para abaratar costes.

A continuación se describe brevemente cada una de las características de seguridad.

3.5.1.1.1. Autentificación básica

Es el método más fácil de implementar, la mayoría de servidores http lo implementan. Consiste en el uso de un formulario para introducir los datos de autentificación del cliente. Los datos viajan en forma de texto plano hacia el servidor, lo que hace que sean fácilmente interceptables.

Los datos viajan codificados en base64 pero no viajan cifrados, lo que hace que no sea especialmente difícil obtener su contenido por alguien ajeno a la transacción.

3.5.1.1.2. Autentificación Digest

Método de autentificación más complejo y seguro que el anterior, en esta ocasión los datos no viajan como texto plano, se codifican mediante una función md5 antes de enviarse al servidor.

El servidor en lugar de almacenar las contraseñas en formato plano puede almacenarlas codificadas con md5, si la cadena md5 enviada por el cliente coincide con la almacenada en el servidor el usuario consigue autentificarse.

3.5.1.1.3. Https (http Seguro)

El protocolo HTTPS es la versión segura del protocolo HTTP. El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP.

El servicio de https usa como puerto estándar el 443 del servidor en lugar del 80 usado por http.

3.5.1.2. Contenido Dinámico

No todos los servidores implementan la misma forma de acceder al contenido dinámico generado por los lenguajes de servidor.

	CGI	FastCGI	Servlet	SSI
Apache 2	Si	Si	No	Si
Lighttpd	Si	Si	No	Si
Cherokee	Si	Si	No	No

Tabla III - 4.- Contenido Dinámico[6]
Fuente: Páginas Web de cada Servidor http

3.5.1.2.1. CGI

Común Gateway Interface (en castellano «Interfaz Común de Pasarela», abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (explorador Web) solicitar datos de un programa ejecutado en un servidor Web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor Web y una aplicación externa.

[6] http://www.lighttpd.net , http://apache.org , http://cherokee.org

3.5.1.2.2. FastCGI

FastCGI es una alternativa al CGI estándar, cuya diferencia radica principalmente en el hecho de que el servidor crea un único proceso persistente por cada programa FastCGI en lugar de por cada solicitud del cliente.

3.5.1.2. Servlet

Los servlets son objetos que corren dentro del contexto de un servidor Web (ej.: Tomcat) y extienden su funcionalidad. También podrían correr dentro de un servidor de aplicaciones (ej.: OC4J Oracle) que además de contenedor para servlet tendrá contenedor para objetos más avanzados como son los EJB (Tomcat solo es un contenedor de servlets). El uso más común de los servlets es generar páginas Web de forma dinámica a partir de los parámetros de la petición que envíe el navegador Web.

3.5.1.3. SSI

Server Side Includes (SSI), es un lenguaje de scripting del lado del servidor. Su principal uso es el de incluir el contenido de uno o varios archivos en otro. Un posible uso es el de incluir una frase del día o aleatoria almacenada en un fichero .txt en un archivo html, o incluir un menú de navegación fácilmente en varias páginas del sitio Web. Los archivos que hacen uso de SSI han de llevar la extensión .shtml

3.5.1.4. Lenguajes del lado del servidor

Para poder usar programas creados en un lenguaje del lado del servidor es necesario un compilador o interprete de este lenguaje y algún método de comunicación con el servidor http. Existen dos metodologías totalmente distintas: usar un programa externo como compilador o intérprete y comunicarse con el servidor mediante CGI o FastCGI, o embeber el intérprete o compilador dentro del propio servidor en forma de módulo.

Normalmente el segundo método ofrece un rendimiento superior al primero, al no tener que depender de un programa externo. Por el contrario hace que el tamaño del servidor Web aumente, así como su complejidad y uso de recursos.

La siguiente tabla refleja los lenguajes de servidor más comunes y la forma de acceder a ellos de cada servidor http.

	Php	Perl	Python	Jsp	Ruby	Asp	ColdFusión (CFML)
Apache 2	Si (mod_php)	Si (mod_perl)	Si (mod_python)	No*	Si (mod_ruby)	Si (soporte parcial)	Si
Lighttpd	Si (CGI / FastCGI)	Si (CGI / FastCGI)	Si (CGI / FastCGI)	No	Si (CGI / FastCGI)	No	No
Cherokee	Si (CGI / FastCGI)	Si (CGI / FastCGI)	Si (CGI / FastCGI)	No	Si (CGI / FastCGI)	No	No

Tabla III - 5.- Lenguajes del lado del servidor[7]
Fuente: Direcciones Web de cada servidor

El soporte para Jsp en Apache se consigue usando el servidor Tomcat también desarrollado por la Fundación Apache.

3.6. Análisis Estadístico

La Estadística se ocupa del estudio de los experimentos aleatorios, es decir, aquellos cuyos resultados no pueden predecirse de antemano.

El experimento generalmente consiste en la observación de alguna característica sobre los elementos de cierto conjunto. Cuando una investigación va referida a un conjunto de elementos, estadísticamente a este conjunto se le llama población:

POBLACIÓN: es el conjunto de todos los individuos (o elementos) que poseen una determinada característica objeto de un estudio estadístico.

Cuando la población es muy grande no se suele hacer una observación exhaustiva de todos sus elementos (sería un proceso difícil, costoso, lento, o incluso imposible). Por ello, normalmente se elige una parte de la población, llamada muestra, y se realiza el estudio sobre los elementos de la misma, para posteriormente extender los resultados obtenidos sobre toda la población.

[7] http://www.lighttpd.net , http://apache.org , http://cherokee.org

MUESTRA: es la parte o subconjunto de la población que se selecciona para realizar un estudio estadístico.

Una vez recogidos los datos de la muestra, el proceso que sigue consta de dos fases:

1. En primer lugar hay que ordenar, organizar y simplificar la información recogida. De esto se ocupa la ESTADÍSTICA DESCRIPTIVA (objeto de esta a usarse en esta investigación).
2. En segundo lugar, hay que realizar un análisis de la información contenida en la muestra, y obtener conclusiones y predicciones que puedan extenderse a toda la población. De esta fase se ocupa la INFERENCIA ESTADÍSTICA.

3.6.1. Caracteres y variables Estadísticas

Como se mencionó anteriormente, cada uno de los individuos de una población o muestra pueden describirse según uno o varios caracteres.

Carácter Estadístico: es una propiedad que permite clasificar a los individuos de la población o muestra.

Se distinguen dos tipos generales de caracteres estadísticos:

- Cualitativos: son aquellos que no pueden ser medidos.
- Cuantitativos: son aquellos que se pueden medir o contar.

Variable Estadística: es el conjunto de todos los valores numéricos que puede tomar un carácter estadístico cuantitativo.

A su vez, las variables estadísticas pueden ser discretas o continuas:

- **Discretas**: son las que toman valores numéricos exactos (generalmente números enteros), y suelen ser resultado de un recuento. Por ejemplo: número de empleados de una fábrica, número de discos vendidos durante un día en una tienda, número de hijos en las familias de una ciudad, etc.
- **Continuas**: son las que pueden tomar, al menos teóricamente, todos los valores posibles dentro de un cierto intervalo de la recta real; suelen ser el resultado de una medición. Por ejemplo: velocidad de un coche en km/h, presión sanguínea de los enfermos de un

hospital, peso de los habitantes de Gijón, diámetro de las piezas producidas por una máquina, etc.

3.6.2. Métodos Estadísticos

Los métodos que se utilizaron en la realización de esta investigación son Duncan, Scheffé y Tukey.

3.6.2.1. Método de Duncan

Se utiliza para comparar todos los pares de medias. Fue desarrollado por primera vez por Duncan en 1951 pero posteriormente él mismo modificó su primer método generando el que ahora se denomina Nuevo método de Rango Múltiple de Duncan. Esta prueba no requiere de una prueba previa de F, como sucede con la DMS o sea que aún sin ser significativa la prueba F puede llevarse a cabo.

La estadística de Prueba es denotado, por q_p $q_p = \frac{Y_{i.} - Y_{i.'}}{\sqrt{CM_{error}/r}}$

Donde p es el número de medias inclusives entre las dos medias a comparar para diseños balanceados. Para aplicar esta prueba al nivel α se debe pasar por las siguientes etapas:

1. Determine el error estándar (desviación estandar) de cada promedio, $S_{\bar{y}_i}$, el cual es dado por la expresión: $S_{\bar{y}_i} = \sqrt{\frac{CM_{EE}}{r}}$

   Donde el CM EE es obtenido de la tabla Anova

2. Con los grados de libertad del error y el nivel de significancia α determinar los valores de r_p (intervalos o amplitudes estandarizadas significativos) utilizando las tablas de amplitudes estandarizadas de Duncan dadas por Harter (1960) y que se encuentran en el libro de Miller (1992). Para encontrar estos valores, se requieren los grados de libertad del error y el valor de $p = 2,3,\ldots,t$.

3. Determinar las amplitudes mínimas significativas denotadas por $R_p, p = 2,3,\ldots,t$ calculados por la expresión: $R_p = r_p s_{\bar{y}_i}$ para $p = 2,3,\ldots,t$

4. Se ordenan de manera creciente los resultados promedios del experimento $\mu_{(1)}, \mu_{(2)}, \cdots, \mu_{(t)}$

5. Se comparan las medias ordenadas $\mu_{(i)}(i = 1,2,3,\ldots t)$así comienza a comparar en el siguiente orden:

a) El promedio más alto, $\mu_{(t)}$con el más bajo, $\mu_{(1)}$comparando esta diferencia con el intervalo mínimo significativo R_t. Si esta diferencia es no significativa entonces todas las otras diferencias son no significantes. Si la diferencia es significativa se continúa con b)

b) Posteriormente se calcula la diferencia entre el valor más alto $\mu_{(t)}$y el penúltimo $\mu_{(2)}$y se compara con el intervalo mínimo significativo R_{t-1}

c) Este procedimiento se continúa hasta que todas las medias se han comparado con la media más grande $\mu_{(t)}$.

$$\mu_{(t)} \text{ vs } \mu_{(1)}$$
$$\vdots$$

d) A continuación se compara la se$\mu_{(t)} \text{ vs } \mu_{(t-1)}$ más grande $\mu_{(t-1)}$con la más pequeña $\mu_{(t-1)} \text{ vs } \mu_{(1)}$ y se compara con el intervalo mínimo significativo R_{t-1}.

$$\mu_{(t-1)} \text{ vs } \mu_{(1)}$$
$$\mu_{(t-1)} \text{ vs } \mu_{(2)}$$
$$\vdots$$
$$\mu_{(t-1)} \text{ vs } \mu_{(t-2)}$$

Este proceso continúa hasta que han sido comparadas las diferencias entre todos los $\frac{t(t-1)}{2}$ posibles pares.

Si una diferencia observada es mayor que el intervalo mínimo significativo, se concluye que la pareja de medias comparadas son significativamente diferentes. Para evitar contradicciones, ninguna diferencia entre una pareja de medias se considera significativamente diferentes, si éstas se encuentran entre otras dos que no difieren significativamente.

Cuando se conoce muy poco respecto a la naturaleza de los tratamientos y por ello es difícil proponer comparaciones con sentido, entonces se necesitan técnicas para probar efectos de tratamientos pero sugeridas por los datos. Estas técnicas se usan con frecuencia para comparar todos los posibles pares de medias, entre algunas de éstas se encuentran: el método de Bonferroni, t-multivariado, Scheffé, Tukey.

3.6.2.2. Método de Scheffé

En algunas situaciones no es fácil conocer las comparaciones que se deben realizar o es posible realizar más de $t-1$ comparaciones. En los experimentos exploratorios las comparaciones de interés se descubren sólo después de examinar los resultados. Scheffe´(1953) propuso un método para probar cualquier contraste C. Utiliza como regla de decisión la siguiente:

Rechazar $H_0 : C = 0$ si $\left|\hat{C}\right| \geq S_{Contraste}\sqrt{(t-1)F_{\alpha,t-1,g.l.,error}} = S_{\alpha,k}$

Donde $S_{Contraste} =$ es el error estándar (varianza estimada) del contraste y cuando se estima el contraste con media $\left(\hat{C} = \sum_{i=1}^{t} c_i \bar{y}_{i.}\right)$, está dado por $S_{Contraste} = \sqrt{CM_{EE}\sum_{i=1}^{t}\frac{c_i^2}{r_i}}$

Donde: $t =$ número de tratamientos

$F_{\alpha,t-1,glerror} =$ percentil $\alpha - ésimo$ de la distribución F con $t-1$ grados de libertad en el numerador y $glerror$ grados de libertad del error experimental.

Se pueden construir intervalos de confianza con este procedimiento para todos los posibles contrastes de las medias de tratamientos. Estos intervalos tienen como límites para la estimación del contraste C.

3.6.2.3. Método de Tukey o Método de la Diferencia Significativa Honesta de Tukey (DSH)

Este procedimiento fue propuesto por Tukey (1952) para probar la hipótesis $H_0 : \mu_j = \mu_k (j \neq k)$. Este método es muy similar en la aplicación al de DMS, salvo por el hecho de que en lugar de utilizar las distribuciones de t como base para realizar las comparaciones, se emplea la distribución del rango estandarizado o estudentizado

$$Q_{\alpha,t,r} = \frac{(\bar{y}_{i.} - \bar{y}_{i'.}) - (\mu_{i.} - \mu_{i'.})}{\sqrt{\frac{CM_{EE.}}{r}}}$$

Se rechaza $H_0 : \mu_i = \mu_i$ si

Nosotros comparamos $|\bar{y}_{i.} - \bar{y}_{i'.}|$ con $Q_{\alpha,t,gl,error}\sqrt{\frac{CM_{EE}}{r}}$

$$|\bar{y}_{i.} - \bar{y}_{i'.}| \geq Q_{\alpha,t,gl,error}\sqrt{\frac{CM_{EE}}{r}}$$

Donde $Q_{\alpha,t,r}$ es el $\alpha - ésimo$ percentil de la distribución rango estandarizado. Tablas para hallar los valores de $Q_{\alpha,t,gl,error}$ son dadas por Harter (1960), Hochberg and Tamhane (1987).

En esta prueba se utiliza un sólo valor con el cual se comparan todos los posibles pares de medias. El método de comparación de Tukey fue reformado por Kramer (1956) para casos en el que el número de réplicas no es igual. Este método es conocido como método de Tukey-Kramer. Este simplemente reemplaza la expresión dada en Tukey por:

$$|\bar{y}_{i.} - \bar{y}_{i'.}| \geq Q_{\alpha,t,gl,error}\sqrt{\frac{1}{2}\left(\frac{1}{r_{i.}} + \frac{1}{r_{i'.}}\right)CM_{EE}}$$

Donde $r = \sum r_i - t$ en un D.C.A.

Si el número de repeticiones no es demasiado desigual, Spotuall y Stoline (1973) dieron un método para probar la hipótesis $H_0 : \mu_i = \mu_{i'}$.

Rechazar H_0 si $|\hat{\mu}_i - \hat{\mu}_{i'}| \geq Q_{\alpha,t,gl,error}\frac{CM_{EE}}{\min(\sqrt{r_i}, \sqrt{r_{i'}}).}$

Cuando las réplicas son muy diferentes este método es menos sensible que el de Scheffé.

3.6.2.4. Graficas Estadísticas

Las distribuciones de frecuencias se presentan en tablas como las anteriores, o bien en gráficas. La representación gráfica se utiliza para facilitar al lector la comprensión de los resultados, pero no añade ninguna información sobre la que contendría una tabla de frecuencias; el objetivo de las gráficas es que la información “impacte” directamente al lector y que se exprese el “perfil” de la distribución, pero no debe olvidarse el rigor en aras de la estética: las gráficas deben reflejar fielmente lo que tratan de representar, fundamentalmente las frecuencias de cada modalidad o valor. Por ello la regla fundamental para la construcción de una gráfica es que:

Las áreas (o longitudes) han de ser proporcionales a las frecuencias, condición inexcusable para que una gráfica sea correcta.

Además, con carácter general puede recomendarse que el pie de la gráfica explique convenientemente de qué se trata, que no se intente representar demasiada información en una sola gráfica, que los detalles sean lo suficientemente visibles, etc.

Existen diversos tipos de gráficas, cada uno de ellos adecuado a un cierto tipo de variables, por lo que podemos clasificar las gráficas atendiendo a estos tipos.

3.6.2.4.1. Diagrama de Cajas y Bigotes (Box and Whisker Plot)

Presentación visual que describe al mismo tiempo varias características importantes de un conjunto de datos, tales como el centro, la dispersión, el alejamiento de la simetría, y la identificación de valores extremos (puntos atípicos), es decir, de valores que se alejan de una manera poco usual del resto de los datos.

Presenta los tres cuartiles, (y los valores mínimos y máximos) alineados sobre una caja vertical u horizontalmente.

3.6.3. Diseño de Experimentos

Definir las reglas básicas a seguir para el diseño, la realización y análisis de experimentos, resaltando las situaciones en que puede o debe ser utilizado. Es de aplicación a todos aquellos estudios y situaciones en las que se necesita ensayar hipótesis sobre una posible relación causa-efecto.

Su utilización será beneficiosa para el desarrollo y seguimiento de los proyectos abordados por los Equipos de Mejora y por todos aquellos individuos u organismos que estén implicados en proyectos de mejora de la calidad en las que concurran estas circunstancias.

Además, se recomienda su uso como herramienta de trabajo dentro de las actividades habituales de gestión.

3.6.3.1. Definiciones/Conceptos

3.6.3.1.1. Experimento

Definición.- El término "Experimento" se refiere a la creación y preparación de lotes de prueba que verifiquen la validez de las hipótesis establecidas sobre las causas de un determinado problema o defecto, objeto de estudio.

Concepto.- En un Experimento, el experimentador escoge ciertos factores para su estudio, los altera deliberadamente de forma controlada y después, observa el efecto resultante.
El Experimento puede realizarse bien en laboratorio o bien en el exterior: En la fábrica, en unos almacenes, en los locales del usuario, etc.

3.6.3.1.2. Diseño de Experimentos

Metodología estadística destinada a la planificación y análisis de un Experimento.

Concepto

El Diseño de un Experimento debe garantizar que este cumpla ciertos requisitos mínimos:

- Debe poder comprobar las hipótesis objeto de estudio, no dejándose confundir por variables insospechadas (=ruido), como errores de medida desproporcionados, etc.
- Debe poder revelar la existencia de cualquier causa importante de variación, aunque no haya sido adelantada como hipótesis.
- Debe mantener los costes de experimentación a un nivel razonable, en comparación con el problema objeto de estudio.
- Debe tener un alto grado de seguridad en las respuestas.
- Si el Experimento se realiza en un laboratorio, éste ha de ser, respecto a las variables estudiadas, un buen indicador de las pruebas que se obtendrían en el taller o "in situ".
- Si el Experimento se realiza durante el desarrollo normal del proceso en estudio, se tendrá además cuidado de interferir lo menos posible en el trabajo normal y protegerse de las interferencias no autorizadas o involuntarias en la prueba por parte del personal adepto.

3.6.3.1.3. Variable

Definición.- Característica de un objeto que puede ser observada y que puede tomar diferentes valores, tanto en el mismo objeto como entre diferentes objetos.

Concepto.- En base a la posibilidad de medida se distinguen dos tipos fundamentales de variables:

- ***Variables cualitativas*:** Son aquellas cuyos valores, de carácter nominal, sólo pueden ser comparados como diferentes entre sí.
- ***Variables continuas*:** Son aquellas cuyos valores, de carácter numérico, permiten realizar un mayor número de comparaciones. Una variable continua permite teóricamente, un infinito número de valores entre dos valores consecutivos. (En la realidad la precisión de los instrumentos de medida pone un límite operativo).

3.6.3.1.4. Sujeto o Unidad Experimental

El sujeto o unidad experimental es la unidad básica sobre la que se efectúa el proceso de medida.

3.6.3.1.4.1. Observación

Definición.- Una observación es una toma de medida de una variable y consta entonces de un valor de la misma. Dependiendo del tipo de Diseño, las observaciones pueden tomarse a diferentes sujetos o al mismo sujeto de manera secuencial.

3.6.3.1.4.2. Diferentes tipos de variables en el Diseño de un Experimento

En un Diseño de Experimento se distinguen los siguientes tipos de variables, según el contenido conceptual, o papel que tienen en el mismo.

Variable independiente

Es el factor (causa) que suponemos influye sobre la característica que medimos (defecto, error, etc). Para comprobar su influencia, el investigador la manipulará durante el Experimento, en el sentido que le asignará valores diferentes a cada observación.

Esta variable se llama también "tratamiento" y cada uno de los valores que se le asignarán "nivel de tratamiento". En este Procedimiento se consideran sólo Experimentos con una única variable independiente. Si se quiere o necesita manipular más de una variable independiente, será necesaria la colaboración de un experto en estadística, puesto que el Diseño del Experimento y la interpretación de los resultados se complican notablemente.

Variable dependiente

Es aquella variable que se mide en cada observación del Experimento, para establecer si la variable independiente efectivamente influye sobre sus valores.

Variables extrañas

Son todas aquellas que el investigador no puede manipular, pero influyen en la variable dependiente. Son la causa de que las observaciones en un mismo nivel de tratamiento no necesariamente arrojen el mismo valor de medida. El conjunto de variables extrañas se denomina generalmente en el Diseño y Análisis de Experimentos "ruido" o "error experimental".

Variable de bloqueo

Es una variable que sabemos puede intervenir en los niveles de la variable dependiente y decidimos eliminar su influencia mediante el control de la misma, creando bloques de observaciones, en los que esta variable, asume respectivamente un valor constante. A cada bloque se asignarán todos los niveles de tratamiento. Con la creación de bloques se persiguen dos objetivos:

- Aislar el efecto de los tratamientos, eliminando la influencia de la variable de bloqueo.
- Estimar los efectos de los bloques.

3.6.3.1.4.3. Repetición

Definición.- Reiteración de una observación o medida al mismo nivel de tratamiento. Proporciona una oportunidad para que los efectos de las variables extrañas, incontroladas se compensen y permite, además, medir el error experimental.

3.6.3.1.4.4. Aleatorización

Definición.- Técnica utilizada para reducir la influencia no predeterminable de variables extrañas sobre los resultados del Experimento.

Concepto.- La aleatorización consiste en asignar los sujetos a los distintos niveles de tratamiento al azar, con la esperanza de que los efectos extraños se contrarresten entre los distintos sujetos y observaciones que componen cada nivel de tratamiento (condición experimental). La aleatorización es fundamental en el Diseño de Experimentos ya que:

a) Previene la existencia de sesgo.

b) Evita la dependencia entre observaciones.

c) Confirma la adecuación de los procedimientos estadísticos para el análisis de los resultados del Experimento.

3.6.4. Elaboración del Experimento en nuestra Investigación

A continuación se puede observar un Diagrama de Flujo mismo que se debe realizar paso a paso para obtener un correcto Diseño de Experimentos.

Diagrama de Flujo

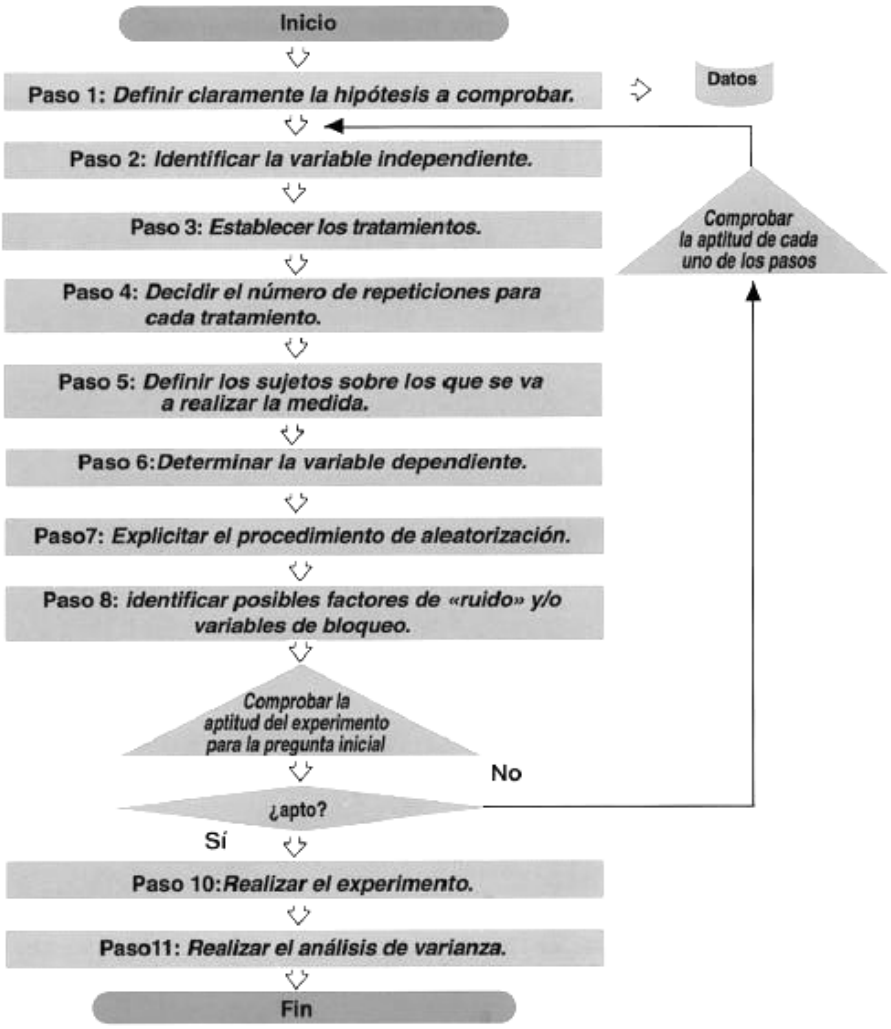


Ilustración III. 1 Diagrama de Flujo Elaboración de Experimento [9]

Paso 1: Definir claramente la hipótesis a comprobar [9]

Es de importancia fundamental identificar de forma muy específica el objetivo del Experimento, es decir, la pregunta exacta que se quiere contestar o la hipótesis que se necesita contrastar.

Hipótesis:	El estudio estadístico del Rendimiento de Servidores Web, de Distribución Libre, permitirá optimizar la plataforma hardware de la EIS

H_0: No existe diferencia significativa entre los servidores Apache, Cherokee, Lighttpd para cargas en 1000, 10000 y 100000 peticiones

H_1 : Existe diferencia significativa en el Rendimiento entre los servidores Web Apache, Cherokee, Lighttpd para cargas en 1000, 10000 y 100000 peticiones

La pregunta en nuestro caso será:

¿Qué permitirá optimizar la plataforma hardware de la EIS?

Paso 2: Identificar la variable independiente

La variable independiente representa la característica que, suponemos, influye sobre los valores de la variable dependiente. Puesto que, para la realización del Experimento, se le asignarán diferentes valores, hay que asegurarse que esté en nuestro poder manipularla.

VARIABLES	*CONCEPTO*
Independiente Servidor Web de Distribución gratuita	**Apache** Hoy en día, el servidor Web Apache es el servidor más usado de Internet. Las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad del desarrollo distribuido. **Cherokee** Es software libre. Es un proyecto que desarrolla una nueva implementación de este tipo de aplicaciones, es un servidor con unas características de las que Apache carece debido a su diseño original. **Lighttpd** es un servidor Web diseñado para ser rápido, seguro, flexible, y fiel a los estándares. Está optimizado para entornos donde la velocidad es muy importante, y por eso consume menos CPU y memoria RAM que otros servidores.

Tabla III - 6.- Definición de variables en nuestra investigación – Variable Independiente
Fuente: Autoría Propia

Paso 3: Establecer los tratamientos

En base a la naturaleza de la variable, las condiciones reales del proceso o situación y la pregunta específica que se quiere contestar, se identificarán los valores o el recorrido de valores de la variable independiente, relevantes para el Experimento y se establecerán los tratamientos a efectuar. Los tratamientos en esta investigación son:

1. Servidor Apache
2. Servidor Cherokee
3. Servidor Lighttpd

Paso 4: Decidir el número de repeticiones para cada tratamiento

Es absolutamente aconsejable realizar varias observaciones para cada nivel de tratamiento (condición experimental), para que los errores de medida e influencias no controladas de variables extrañas puedan contrarrestarse entre sí.

Así en esta investigación tomamos para cada nivel de tratamiento lo siguiente:

1. **Rendimiento del servidor en Páginas Estáticas** en donde se realizará un total de 1.000, 10.000, 100.000 peticiones (10 simultáneamente) de un archivo html estático, con imágenes, de un tamaño de 11500 bytes.

2. **Rendimiento del servidor sirviendo archivos,** en donde realizará un total de 1.000, 10.000 y 100.000 peticiones (10 simultáneamente) de un archivo de datos binario (un archivador .tar) de un tamaño aproximado de un megabyte (1024984 bytes).

3. **Rendimiento del servidor procesando archivos dinámicos (PHP)**, en donde se realizará un total de 1.000, 10.000, 100.000 peticiones (10 simultáneamente) de un documento generado dinámicamente mediante el lenguaje de lado del servidor PHP. El documento generado contiene información del entorno del servidor y de PHP.

4. **Rendimiento del servidor procesando archivos dinámicos (PHP) y acceso a bases de datos (MySql)**, en donde se realizará un total de 1.000, 10.000, 100.000 peticiones (10 simultáneamente) de un documento generado dinámicamente mediante el lenguaje de lado del servidor PHP. Se necesita acceder a una base de datos MySql para generar dicho documento. Aquí no solo se depende del servidor http, entra el juego también el servidor MySql. Dado el hecho de que se emplea la misma base de datos, el mismo servidor de base de datos y la misma máquina podemos suponer que el servidor MySql ofrezca un rendimiento similar para un servidor http que para otro.

El programa de pruebas y cargas de trabajo, llamada comparativa es un pequeño programa desarrollado en shell que realiza cuatro pruebas distintas. ***Ver Anexo 1.***

Midiendo el rendimiento del sistema con SAR[40]

El comando sar nos permite monitorizar un sistema en Linux, proporcionándonos el uso de la cpu, memoria, discos... El nombre sar proviene de las siglas de "system activity report" (informe de la actividad del sistema). En Linux se encuentra normalmente en el paquete sysstat. El paquete sysstat incluye programas y scripts para recopilar y mostrar información sobre el rendimiento del sistema, generando informes detallados. Este conjunto de programas

puede resultar de mucha utilidad a la hora de detectar cuellos de botella y para hacernos una idea de cómo se utiliza el sistema a lo largo del día. ***Ver Anexo 2***

Paso 5: Definir los sujetos sobre los que se va a realizar la medida

Los sujetos en donde se realizarán las pruebas necesarias en esta comparativa se encuentran clasificados en 3 grupos mismos que son:

Se debe indicar previamente que en cada Grupo se encuentran involucradas 3 Pc's que tienen la misma arquitectura como lo podemos ver en las **Tabla 7.- Computadores que intervienen en la Arquitectura 1, Tabla 8.- Computadores que intervienen en la Arquitectura 2, Tabla 9.- Computadores que intervienen en la Arquitectura 3**

Grupo 1 → Arquitectura 1

I6	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB
I10	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB MPROF
I8	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB M8

Tabla III - 7.- Computadores que intervienen en la Arquitectura 1
Fuente: Autoría Propia

Grupo 2 → Arquitectura 2

M10	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel Pentium 4 530J, 3000 MHz (15 x 200)
M20	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel West Branch D915GVWB (2 PCI
M7	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel Pentium 4 530J

Tabla III - 8.- Computadores que intervienen en la Arquitectura 2
Fuente: Autoría Propia

Grupo 3 → Arquitectura 3

P16	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)
P11	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)
P5	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)

Tabla III - 9.- Computadores que intervienen en la Arquitectura 3
Fuente: Autoría Propia

Paso 6: Determinar la variable dependiente

Sólo puede existir una única variable dependiente. Esta deberá tener, necesariamente, un nivel de medida continuo, o lo más próximo a ese extremo que sea posible. Cuantas más posibilidades de apreciar diferencias entre distintas observaciones ofrezca la variable dependiente, más se favorecerá la sensibilidad de la misma a los distintos tratamientos, esto lo podemos apreciar de mejor manera en la **Tabla 10.- Definición de variables en nuestra investigación = variable Dependiente**

VARIABLES	INDICADORES	INDICES
Dependiente Rendimiento	1. Parámetros del Servidor	• Petición por segundo • Tiempo empleado por cada petición • Tasa de Transferencia • Tiempo por petición • Número de Peticiones fallidas
	2. Uso del CPU	• %uso de CPU por usuario • %uso de CPU por sistema y kernel • %uso de CPU disponible
	3. Uso de Memoria	• %memoria RAM utilizada • %Memoria SWAP utilizada
	4. Uso de Disco	• Cantidad de datos leídos

Tabla III - 10.- Definición de variables en nuestra investigación – Variable Dependiente
Fuente: Autoría Propia

Paso 7: Explicitar el procedimiento de aleatorización

Esta es una parte muy importante del Diseño, ya que asegurará que las diferencias que se encuentren entre los tratamientos son debidas a ellos mismos y no a efectos laterales no deseados.

El tamaño de la muestra en esta investigación es de 3 computadores, ya que se obtendrán valores aproximados a la realidad, 3 arquitecturas para demostrar el nivel de influencia del Hardware sobre los servidores Web.

Se decidió tomar computadores de los laboratorios de:

- Investigación y Desarrollo: El hardware de este laboratorio es similar al servidor Web que se tiene en la Escuela de Sistemas

- Multimedia: Las computadoras de este laboratorio tienen un HW de uso medio, es decir que se puede encontrar en cualquier institución.
- Programación: Son computadoras con características bajas con memorias RIM y se puede decir que son computadores que se los puede encontrar en lugares donde la inversión económica no se orienta precisamente a actualizar equipos computacionales.

Cada uno de estos laboratorios se encuentra codificado de la siguiente manera:

Laboratorio		# de Codificación
Investigación y desarrollo	I	Número de PC (1 .. n)
Multimedia	M	Número de PC (1 .. n)
Programación	P	Número de PC (1 .. n)

Tabla III - 11.- Codificación de los Laboratorios de la EIS
Fuente: Autoría Propia

Un procedimiento muy fácil y común para asignar los sujetos a los diferentes tratamientos al azar es el siguiente:

Poner todos los papeles (con los nombres de las Pc's) en una bolsa y mezclarlos. Extraer, para cada tratamiento, 3 papeles "a ciegas" de cada arquitectura, así obtenemos las Pc's en las que se llevaran a cabo los benchmarks para esta investigación.

Nombre PC	DESCRIPCION
I6, I10, I8	Tecnología similar al servidor Web de la EIS
M10, M4, M7	Tecnología con características media
P6 ,P11 , P5	Tecnología con características básicas

Paso 8: Identificar posibles factores de "ruido" y/o variables de bloqueo

Analizar la futura situación experimental e identificar los factores que puedan, además de la variable independiente, influir sobre los valores de la variable dependiente. Según su relevancia y las posibilidades técnicas, la decisión a adoptar respecto a los factores de ruido y/o variables de bloqueo, será la siguiente:

- O bien tenerlos bajo control (constantes), a lo largo de todas las observaciones.
- O bien integrarlos en el Diseño, como variable de bloqueo.

- O bien transformarlos en una variable independiente. Esto será necesario, aunque se complique notablemente el Diseño de Experimento desde el punto de vista estadístico, cuando su influencia sobre la variable dependiente resulte ser relevante.
- O bien se espera que la aleatorización sea suficiente para que sus efectos se contrarresten en las repeticiones de cada tratamiento. Esta posibilidad será aceptable sólo si la variable en cuestión está fuera de nuestro control y se considera que su influencia es bastante limitada. Su efecto se englobará dentro del "error experimental", o "ruido".

En nuestro Experimento:

Los factores de ruido que se han considerado:

- Todas las Pc's fueron adquiridas en la misma empresa en un solo pedido, en la misma fecha, por lo tanto tienen las mismas características Hardware, factor de ruido controlado
- Virus en las Pcs, está controlado ya que se utiliza Software Libre lo que hace más segura la investigación.
- Desgaste del Hardware por uso constante, es controlable

Por lo tanto se puede decir que el ruido no afectara en la investigación.

Paso 9: Asegurarse de la aptitud del Diseño del Experimento para contestar la pregunta inicial

Esta investigación proporciona efectivamente la información necesitada para realizar la Comparativa de Servidores Web de Distribución Libre con lo que optaremos por el servidor que proporcione un óptimo servicio.

Paso 10: Realización del Experimento

A la hora de instalar un servidor Web debe tenerse en cuenta una serie de consideraciones previas que tienen que ver con el soporte físico (hardware) sobre el que correrá el servidor: interfaces de red, sistema de almacenamiento SCSI con soporte RAID, memoria RAM de al menos 256 MB, procesador dependiente de si el contenido del sitio Web es más bien dinámico o estático, y sobre todo si tiene que acceder a diferentes bases de datos.

Y en cuanto al software, el sistema operativo será también una elección importante a la hora de montar un sitio Web, junto con el software servidor de Web.

Se tomaron en cuenta ciertos parámetros que permiten clasificar un servidor http que son:

- **Coste del servidor**, servidores gratuitos como por ejemplo Apache http Server, Cherokee Server, Lighttpd Server.
- **Plataformas soportadas**, servidores multiplataforma como Apache http Server, Cherokee y Lighttpd.
- **Propósito del servidor**, podemos clasificar los servidores http según el uso al que se destine, hay servidores de propósito general dotados de muchas características, otros servidores están muy optimizados para alguna tarea en concreto (mínimo uso de recursos del sistema, gran rendimiento en contenido estático,...) a consta de renunciar a algunas características.
- **Según su implantación**. Los servidores http suelen emplear concurrencia para servir a más de un cliente simultáneamente. Podemos clasificar los servidores http según la técnica que usen para implementar esta concurrencia: máquina de estados finitos, servidores basados en hebras, servidores basados en procesos, servidores mixtos.

Para realizar esta comparativa se ha elegido como sistema operativo Linux distribución CentOs y 3 servidores http.

- Apache2 (versión 2.2.3)
- Lighttpd (versión 1.4.11)
- Cherokee (versión 0.6.0)

Para el desarrollo de las distintas pruebas se ha procedido a la instalación y configuración de los tres servidores en las siguientes máquinas:

Grupo 1 → Arquitectura 1

I6	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB
I10	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB (MPROF)
I8	Unknown, 2133 MHz (16 x 133) - Intel Corporation DG965SS -1005 MB (M8)

Tabla III - 12.- Características Hardware Empleado – Arquitectura 1

Grupo 2 → Arquitectura 2

M10	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel Pentium 4 530J, 3000 MHz (15 x 200)
M4	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel West Branch D915GVWB (2 PCI
M7	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video) Intel Pentium 4 530J

Tabla III - 13.- Características Hardware Empleado – Arquitectura 2

Grupo 3 → Arquitectura 3

P16	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)
P11	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)
P5	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio) - Intel Tehama i850 - 512 MB (RDRAM)

Tabla III - 14.- Características Hardware Empleado – Arquitectura 3

Para ver detalles completos acerca de las computadoras utilizadas en este experimento Ver ANEXO 2. Es necesario un servidor de bases de datos para algunas pruebas. Se ha empleado MySql 5 para dicho fin.

Características implementadas.

Comenzamos analizando detalladamente las características implementadas por cada uno de los servidores http.

Esto nos servirá para conocer en profundidad a cada uno de los servidores http. No todas las características reflejadas en esta parte cuentan para optar por el servidor Web que proporcione un óptimo servicio.

En la siguiente fase de nuestra comparativa procedemos con la evaluación de cada uno de los servidores Web, mediante el uso de un benchmark.

Una vez hemos hecho un repaso de las distintas características implementadas por cada servidor http vamos a pasar a medir su rendimiento de manera objetiva usando un benchmark creado para tal fin. **Ver Anexo 1**

El objetivo de este benchmark es obtener mediciones que reflejen el rendimiento de cada servidor http por separado. Para ejecutar cada benchmark, con el sistema recién arrancado, comprobamos que solo se está ejecutando el servidor http del que se va a realizar la medición (y no los demás) así como que funciona cualquier otro sistema o servidor empleado por el mismo. Lanzamos el benchmark y lo dejamos que se ejecute, sin utilizar el sistema para otro propósito durante la ejecución del mismo.

Se trata de medir **parámetros propios de un servidor http**, como son:

El **Rendimiento del Servidor** depende del mucho del propósito del sitio Web.

+ **Requests per second:** número de peticiones por segundo, indica el número de peticiones por segundo que el servidor es capaz de servir correctamente. Un número mayor es el mejor resultado que se desea.
- **Time taken for tests:** tiempo empleado por cada prueba. Cada servidor http se someterá a varios pruebas que se detallan más abajo. Esta medida indica el tiempo necesario que necesita un servidor para realizar una prueba completa. Un menor valor en esta medida es lo deseado.
+ **Transfer rate:** indica la tasa de transferencia que el servidor es capaz de mantener durante en desarrollo de la prueba. Se mide en Kbytes/sec. Un mayor valor en esta medida indica que es mejor.
- **Time per request:** tiempo empleado en atender a cada petición. Un menor valor de esta medida indica mayor rendimiento.

Preparando el Sistema.

Para realizar esta prueba hemos empleado los equipos descritos anteriormente (grupo1, Grupo 2, y Grupo 3), en los que hemos instalado y configurado todos los servidores de la siguiente forma.

1. **Apache2**, Se instala conjuntamente con el Sistema Operativo CentOs. Se configura mediante un archivo httpd.conf situado en /etc/apache2 Por defecto se instala

correctamente sin ser necesario modificar esta configuración por defecto. Si deseamos usar virtual hosts deberemos crear un archivo de configuración en /etc/apache2/sites-available y crear un enlace en /etc/apache2/sites-enabled para cada sitio que deseemos configurar. Por defecto instalamos Apache2 en el puerto 80 de nuestro sistema.

2. **Cherokee**, Primero nos bajamos el archivo **"cherokee-0.6.0b700-14.3.i386.rpm"** y lo instalamos. Una vez que tenemos nuestro archivo, lo instalamos de la siguiente manera:

```
# rpm -Uhv cherokee-0.6.0b700-14.3.i386.rpm
```

Luego modificamos el archivo cherokee.conf

```
# vi /etc/cherokee.conf
```

modificamos el apartado del DocumentRoot:

```
vserver!default!document_root = /svr/www/htdocs/       --------->
Cambiamos esta línea por   vserver!default!document_root =
/var/www/html/
```

Configuración de Cherokee

Modificamos la línea que contiene el path de nuestro intérprete PHP:

```
vserver!default!extensions!php!handler = fcgi
vserver!default!extensions!php!handler!balancer = round_robin
vserver!default!extensions!php!handler!balancer!type = interpreter
vserver!default!extensions!php!handler!balancer!local1!host =
localhost:1234
vserver!default!extensions!php!handler!balancer!local1!env!PHP_FCGI
_CHILDREN = 5
vserver!default!extensions!php!handler!balancer!local1!interpreter
= /usr/lib/cgi-bin/php -b 1234 --------> cambiar solo esta linea
por

vserver!default!extensions!php!handler!balancer!local1!interpreter
= /usr/bin/php-cgi -b 1234
```

Una vez que terminamos de modificar, arrancamos nuestro servidor para probar nuestra nueva configuración ejecutando el comando:

```
# cherokee
```

y nos muestra la pagina principal de Cherokee.

3. **Lighttpd**, Bajamos los archivos lighttpd -**1.3.16-1.2.el4.rf.i386.rpm y lighttpd-fastcgi-1.3.16-1.2.el4.rf.i386.rpm**, luego empezamos la instalación:

```
# rpm -Uhv lighttpd-1.3.16-1.2.el4.rf.i386.rpm        (Instalador del Webserver)
# rpm -Uhv lighttpd-fastcgi-1.3.16-1.2.el4.rf.i386.rpm     (Instalador para soporte de php)
```

Una vez que tenemos instalados los dos paquetes, procedemos a arrancar nuestro Web Server para saber que está funcionando.

```
# /etc/init.d/lighttpd start     (Iniciamos nuestro Web Server Lighttp)
```

Para saber que arranco bien nuestro servidor, ponemos lo siguiente en un browser como el mozilla:

"http://localhost/" y nos tiene que salir la página principal de Lighttp.

Ahora procedemos a configurar nuestro Web Server de la siguiente manera:

vi /etc/lighttpd/lighttpd.conf

Con este comando editamos el archivo principal de lighttp, descomentamos la línea que contenga "mod_fastcgi" para dar soporte PHP a nuestro servidor. Algo así:

```
## modules to load
# at least mod_access and mod_accesslog should be loaded
# all other module should only be loaded if really neccesary
# - saves some time
# - saves memory
server.modules              = (
#                               "mod_rewrite",
#                               "mod_redirect",
#                               "mod_alias",
                                "mod_access",
#                               "mod_cml",
#                               "mod_trigger_b4_dl",
#                               "mod_auth",
#                               "mod_status",
#                               "mod_setenv",
```

```
                                "mod_fastcgi",  #Linea a
descomentar
#                                "mod_proxy",
#                                "mod_simple_vhost",
#                                "mod_evhost",
#                                "mod_userdir",
#                                "mod_cgi",
#                                "mod_compress",
#                                "mod_ssi",
#                                "mod_usertrack",
#                                "mod_expire",
#                                "mod_secdownload",
#                                "mod_rrdtool",
                                 "mod_accesslog" )
```

También cambiamos la ubicación de nuestro DocumentRoot:

```
server.document-root        = "/srv/www/lighttpd/"     ----> la
cambiamos por
server.document-root        = "/var/www/html/"
```

También modificamos las siguientes líneas o mejor dicho, las descomentamos:

```
#### fastcgi module
## read fastcgi.txt for more info
## for PHP don't forget to set cgi.fix_pathinfo = 1 in the
php.ini
fastcgi.server             = ( ".php" =>
                               ( "localhost" =>
                                 (
                                   "socket" =>
"/var/run/lighttpd/php-fastcgi.socket",
                                    "bin-path" => "/usr/bin/php-
cgi"
                                 )
                               )
                             )
```

Ahora modificamos el archivo php.ini:

```
# vi /etc/php.ini
```

y agregamos al final del archivo la siguiente línea:

```
cgi.fix_pathinfo = 1
```

Guardamos los archivos que se modificaron y reiniciamos nuestro servidor.

```
# /etc/init.d/lighttpd restart
```

Ya tenemos configurado nuestro Webserver con Lighttpd.

No nos interesa que se ejecuten todos simultáneamente, para que el rendimiento de un servidor no pueda verse influenciado por la ejecución de otro. Para evitar que se carguen al inicio **procedemos a eliminar el permiso** de ejecución de cada uno de los demonios de los servidores, situados en /etc/init.d

Antes de realizar cada prueba procederemos a arrancar manualmente el servidor especificado. Para una de nuestras pruebas necesitamos hacer uso de una base de datos, procedemos a instalar y configurar MySql 5 como servidor de base de datos.

Uso de recursos del sistema

Una vez comprobado el rendimiento de cada servidor http, mediante el uso de un benchmark sintético vamos a evaluar el uso de los recursos del sistema que hace cada servidor http, estos resultados los encontramos en el Anexo 4, Anexo 5 y Anexo 6. Para ello hemos monitorizado el sistema mediante la utilidad *sar* recogiendo muestras del sistema cada 2 minutos. **Ver Anexo 2**

Nos centraremos principalmente en el uso de CPU, la ocupación de memoria RAM y el uso del subsistema de disco. Gracias a la información almacenada en el fichero log de la ejecución de cada benchmark podemos saber la hora exacta de inicio y fin de cada prueba. Usamos esta información para extraer la información precisa de los ficheros de salida del comando ***sar***.

Usando las mediciones recogidas durante la ejecución del benchmark tomamos los valores que presentan diferencias significativas para evaluar el rendimiento del sistema con cada uno de los servidores http.

Para medir la carga del sistema durante la ejecución de las distintas comparativas, se ha elegido los siguientes valores a monitorizar:

Uso de CPU

- **%user**, porcentaje del uso de la CPU empleado por los procesos del usuario. Mientras menos uso de CPU utilice un servidor, se lo tomará como el mejor.
- **%system,** porcentaje del uso de la CPU empleado por los procesos del kernel y del sistema. Mientras menos uso de procesos del kernel y del sistema utilice un servidor, se lo tomará como el mejor.
+ **%idle**, porcentaje de uso de CPU disponible para nuevos procesos. Mientras más CPU disponible presente un servidor, se lo tomará como el mejor.

Uso de Memoria

- **kbmemused**, Kilobytes de la memoria RAM ocupados por los distintos procesos que se están ejecutando. Mientras menos memoria RAM utilice en los distintos procesos un servidor, se lo tomará como el mejor.

Uso de Disco.

+ **bread/s**, cantidad total de datos leídos desde los dispositivos de almacenamiento masivo en bloques por segundos. Mientras más cantidad total de datos leídos desde los dispositivos de almacenamiento presente un servidor, se lo tomará como el mejor.

Los datos con los que se trabaja son con las medias, tomando en cuenta el nivel de significancia que rechaza la hipótesis nula, estos datos se los puede observar en el ANEXO 4 – DATOS DE LAS MEDIAS OBTENIDAS DE LAS PRUEBAS A LOS QUE FUERON SOMETIDOS LOS SERVIDORES HTTP Y QUE HAN PRESENTADO DIFERENCIAS SIGNIFICATIVAS – 1000 peticiones, tabla 1. Páginas Estáticas – 1000 peticiones, tabla 2. Transferencia de Archivos – 1000 peticiones, tabla 3. Páginas Dinámicas (PHP) – 1000 peticiones, tabla 4. Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL) – 1000 peticiones, ANEXO 5 – DATOS DE LAS MEDIAS OBTENIDAS DE LAS PRUEBAS A LOS QUE FUERON SOMETIDOS LOS SERVIDORES HTTP Y QUE HAN PRESENTADO DIFERENCIAS

SIGNIFICATIVAS – 10000 peticiones, tabla 1. Páginas Estáticas – 10000 peticiones, tabla 2. Transferencia de Archivos – 10000 peticiones, tabla 3. Páginas Dinámicas (PHP) – 10000 peticiones, tabla 4. Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL) – 10000 peticiones, y el ANEXO 6 – DATOS DE LAS MEDIAS OBTENIDAS DE LAS PRUEBAS A LOS QUE FUERON SOMETIDOS LOS SERVIDORES HTTP Y QUE HAN PRESENTADO DIFERENCIAS SIGNIFICATIVAS – 100000 peticiones, tabla 1. Páginas Estáticas – 1000 peticiones, tabla 2. Transferencia de Archivos – 100000 peticiones, tabla 3. Páginas Dinámicas (PHP) – 100000 peticiones, tabla 4. Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL) – 100000 peticiones, contienen los datos de las medias que se han obtenido al procesarlos en el Software SPSS[8].

Paso 11: Análisis de Varianza (ANOVA)

En este paso se realiza el cálculo de los valores que presentan diferencia significativa y para ello utilizamos el software SPSS ver 15.0 en donde los resultados arrojados nos indicarán si presentan o no diferencias significativas al comparar Apache, Cherokee y Lighttpd

[8] La información completa se encuentra en los archivos que se encuentran en el CD adjunto a esta documentación

CAPÍTULO IV

ANÁLISIS DE RESULTADOS

4.1. Prefacio

A continuación se realiza la evaluación de las características implementadas por cada servidor http cuyo peso asignado es del 30% del valor total de la comparativa y un análisis de los resultados de las pruebas realizadas a los servidores http Apache2, Cherokee y Lighttpd cuyos valores pueden ser apreciados en forma porcentual y el peso asignado a esta evaluación es del 70% del valor total. Dichas pruebas, fueron descritas en el Capítulo III – Comparativa de Servidores, siendo ordenadas ascendentemente por su nivel de complejidad como sigue: Prueba de Páginas Estáticas, Prueba de Transferencia de Archivos, Prueba de Páginas Dinámicas (PHP) y Prueba de Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL)**.**

Éste análisis de resultados fue desarrollado agrupando los resultados obtenidos en las diferentes pruebas a las que se sometieron los servidores http, por la cantidad de peticiones desarrolladas 1000, 10000, 100000 peticiones, por tipos de Páginas: Estáticas, Transferencia de Archivos, Dinámicas y Dinámicas con Base de Datos. Para obtener un resultado final cuya conclusión nos permitirá saber que servidor es el apropiado.

A pesar de que la investigación es más amplia se analizará con detenimiento la Arquitectura 3 la que está compuesta por equipos de características siguientes:

Intel Maryville D850MV Intel Pentium 4, 1600 MHz (16 x 100) (5 PCI, 1 AGP, 4 RIMM, Audio) – Intel Tehama i850 –

512 MB **(RDRAM)** cuya memoria es de una velocidad de procesamiento similar a los 256MB DDR que actualmente se utiliza

Disco Duro MAXTOR 4K040H2 (40 GB, 5400 RPM, Ultra-ATA/100)

En esta arquitectura los datos a analizar serán los obtenidos cuando se someten a una carga de 10.000 peticiones, dado que la carga a la que serán sometidos los mismos nunca superará el valor antes mencionado teniendo en cuenta que estos servidores tienen uso académico.

A continuación iniciamos la evaluación de características implementadas por cada Servidor http.

4.2. Evaluación de las características implementadas

En la **Tabla IV- 1.- Características Implementadas,** observamos las características que cada servidor dispone, donde el 30% del valor total de la comparativa es asignado al servidor(s) que cumplen con lo solicitado, cabe indicar que todos los servidores soportan conexión a Base de Datos.

	Apache	**Cherokee**	**Lighttpd**
Características generales			
Coste	Si	Si	Si
Código Abierto	**Si**	**Si**	**Si**
Windows	Si	Si	Si
MacOS	Si	Si	Si
Linux	**Si**	**Si**	**Si**
BSD	Si	Si	Si
Solaris	Si	Si	Si
Seguridad			
Autentificación básica	Si	Si	Si
Autentificación Digest	Si	Si	Si
Https	**Si**	**Si**	**Si**
Virtual Hosts*	**Si**	**Si**	**Si**

* La posibilidad de ejecutar varios sitios webs en un mismo servidor no hace más seguro un servidor. Se podría argumentar lo contrario, es decir, a mayor número de sitios webs ejecutandose en un servidor es más probable que un fallo en alguno de los sitios webs pueda comprometer el sistema general. Sin embargo normalmente la ganancia de prestaciones y de aprovechamiento del sistema hacen necesario alojar varios sitios en un mismo servidor para abaratar costes.

Contenido Dinámico			
CGI	**Si**	**Si**	**Si**
FastCGI	**Si**	**Si**	**Si**
Servlet	No	No	No
SSI	Si	No	Si
Lenguajes del lado del servidor			
Php	**Si**	**Si**	**Si**
Perl	Si	Si	Si
Python	Si	Si	Si
Jsp	No**	No	No
Ruby	Si	Si	Si
Asp	Si	No	No
Cold Fusion	Si	No	No

- **Tabla IV 1.-**

Características implementadas por cada servidor Http
Fuente: Páginas Web de cada servidor Web motivo de la investigación

Como se puede apreciar los servidores http Apache, Cherokee y Lighttpd cumplen con las características solicitadas, se debe hacer hincapié que las características que más se tomaron en cuenta son las que se encuentran sombreadas. Además que todos soportan conexión a Base de Datos. A los tres servidores se les ha asignado la puntuación del 30% del Valor total.

Ya obtenida una visión clara de cómo funcionan los servidores http de acuerdo a las características que presentan cada servidor, se procede al análisis de los resultados de las pruebas realizadas a los servidores http Apache2, Cherokee y Lighttpd.

4.3. Resultados numéricos de la comparativa

Se presentan los resultados de forma ordenada mediante tablas que se pueden observar en el Anexo 4, Anexo5, Anexo 6; cuyo contenido son las medias obtenidas de las pruebas a los que fueron sometidos los servidores http y que han presentado diferencias significativas. Y en el caso donde se encuentran casilleros en blanco significa que los servidores http no presentaron diferencias significativas

* El soporte para Jsp en Apache se consigue usando el servidor Tomcat también desarrollado por la Fundación Apache.

4.3.1. Evaluación de los Servidores http por prueba realizada

Las tablas con los datos de las medias obtenidas en las pruebas a los que fueron sometidos los servidores http se encuentran ubicadas en los Anexos: 4, 5 y 6 de acuerdo al número de peticiones y tipos de páginas.

Con el objetivo de analizar gráficamente esta información se sumaron las cantidades del servidor Apache, Cherokee y Lighttpd esto es el 100%; luego se toma cantidades individuales por servidor y se realiza una regla de tres para obtener el equivalente al 100% estipulado, esto solamente en las pruebas que tienen cantidades ya que hay pruebas en los Recursos del Sistema en donde todo es porcentual.

Ejemplo:

		1000 peticiones	
	Arquitectura 1	**Arquitectura 2**	**Arquitectura 3**
Apache	0	**19233,937**	0
Cherokee	0	4869,853	0
Lighttpd	0	7122,903	0
Esta cantidad se la toma como el 100%		**31226,693**	

Equivalencias porcentuales con la regla de tres

		1000 peticiones	
	Arquitectura 1	**Arquitectura 2**	**Arquitectura 3**
Apache	0	**(19233,937 * 100)/ 31226,693 = 61,6**	0
Cherokee	0	(19233,937 * 100)/ 31226,693 = 15,6	0
Lighttpd	0	(19233,937 * 100)/ 31226,693 = 22,8	0

Este procedimiento se lo realiza en todas las tablas y en el caso de las variables que componen el Uso de Recursos se mantienen los valores porcentuales no se hizo ninguna conversión, ya con todos estos datos se procede a realizar una evaluación gráfica de cada prueba realizada, iniciando con Peticiones por segundo.

a) REQUEST PER SECOND (PETICIONES POR SEGUNDO)

Requests per second: número de peticiones por segundo, indica el número de peticiones por segundo que el servidor es capaz de servir correctamente. Un número mayor es el mejor resultado que se desea. Los datos que cumplan con este requisito se encuentran subrayados.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 1.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) Peticiones por Segundo** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 61,5% a los servidores Cherokee y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera en un 55,5% a Apache y Cherokee, en la Arquitectura 2 con un 48,1%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 53,5% a Apache y Cherokee, en la Arquitectura 2 con un 49,8%, y en la Arquitectura 3 con un 57,8%.

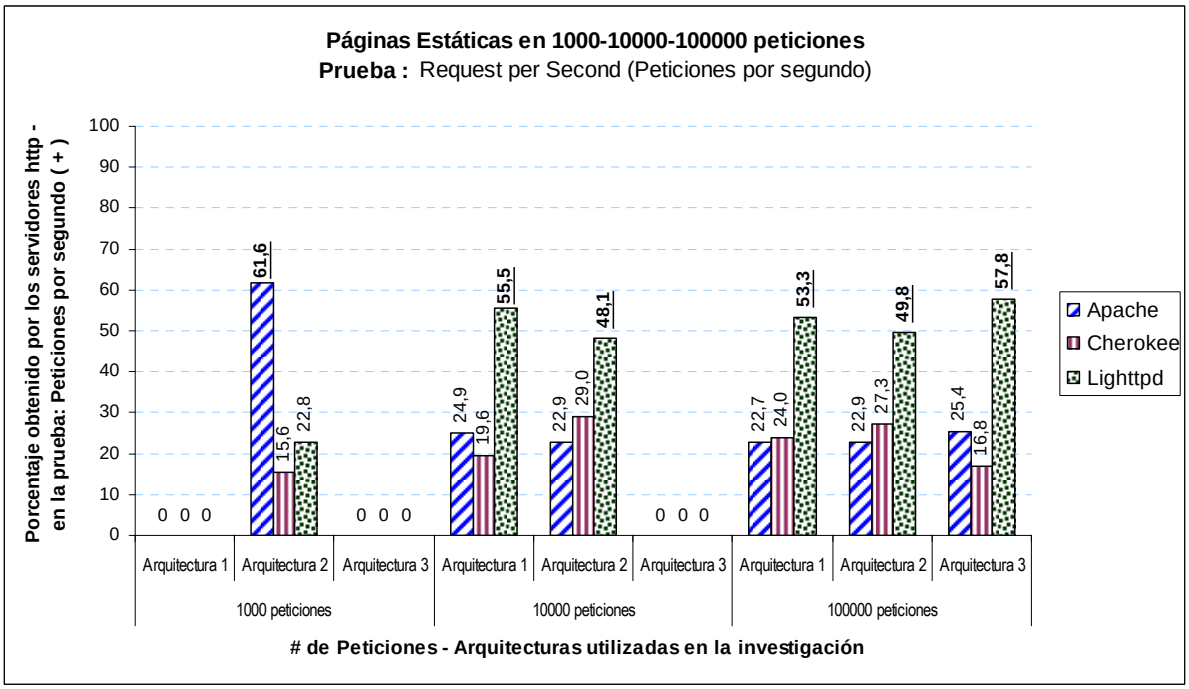


Ilustración IV- 1.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Peticiones por Segundo en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 2.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) Peticiones por Segundo** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 56,2% a los servidores Apache y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 40,6% a Apache y Cherokee, en la Arquitectura 2 Cherokee supera con un 59,3%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 2 supera con un 60,6% a Apache y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

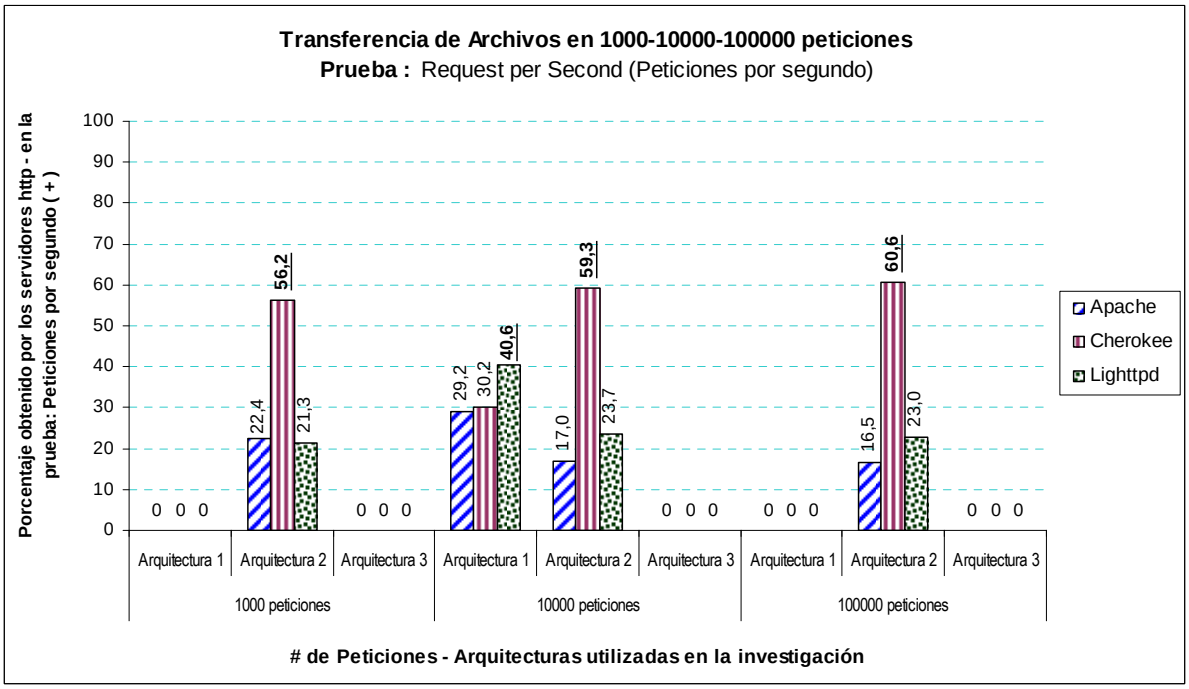


Ilustración IV- 2.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de

estos servidores su servicio no varía en lo que se refiere a Peticiones por Segundo en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 3.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (+) Peticiones por Segundo** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 61,5% a los servidores Cherokee y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 55,5% a Apache y Cherokee, en la Arquitectura 2 con un 48,1%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 53,5% a Apache y Cherokee, en la Arquitectura 2 con un 49,8%, y en la Arquitectura 3 en un 57,8%.

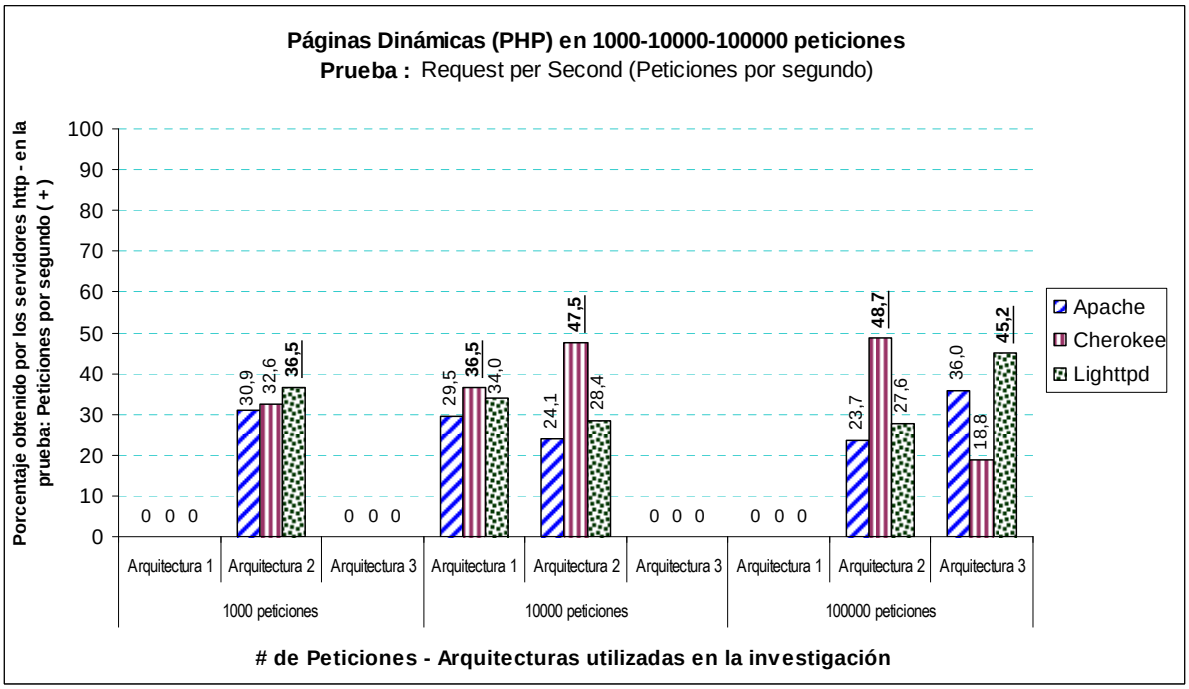


Ilustración IV- 3.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Peticiones por Segundo en Páginas Dinámicas (PHP).

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 4.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (+) Peticiones por Segundo** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 45,2% a los servidores Apache y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Cherokee en la Arquitectura 1 supera con un 36,7% a Apache y Lighttpd, en la Arquitectura 2 Cherokee supera con un 59,3% a Apache y Lighttpd, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 1 supera con un 37,1% a Apache y Lighttpd, en la Arquitectura 2 Cherokee supera con un 48,8% a Apache y Lighttpd, y en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

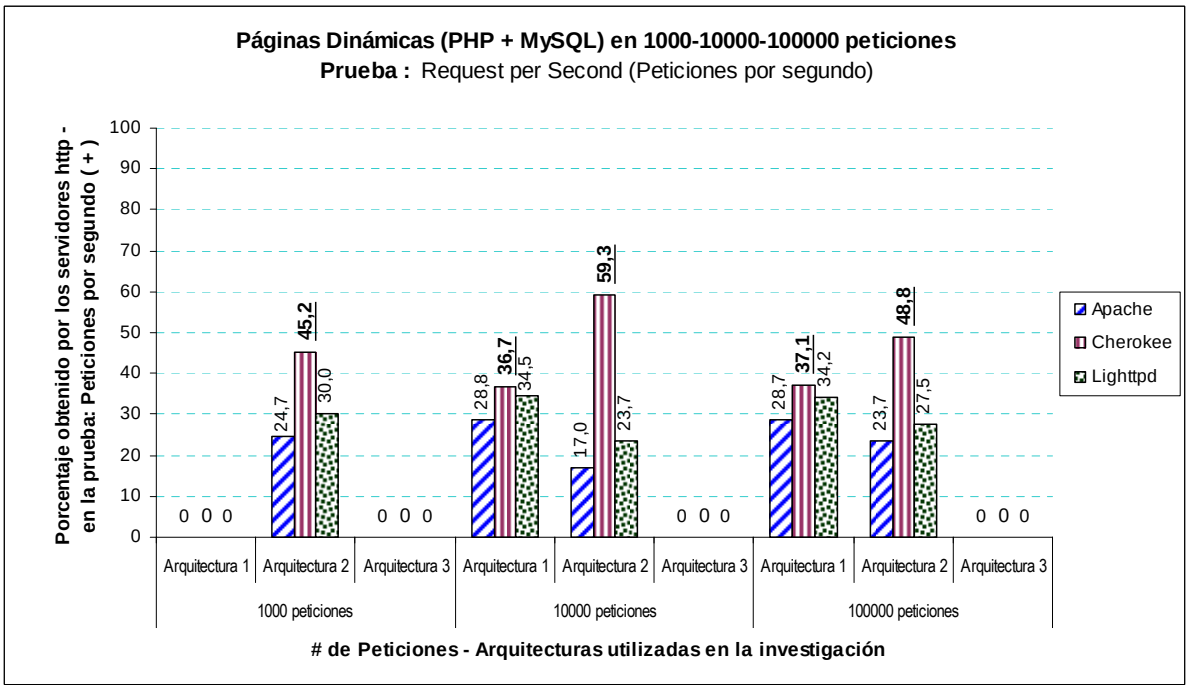


Ilustración IV- 4.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Peticiones por Segundo

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Peticiones por Segundo en Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL).

b) - TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)

Time taken for tests: tiempo empleado por cada prueba. Cada servidor http se someterá a varios pruebas que se detallan más abajo. Esta medida indica el tiempo necesario que necesita un servidor para realizar una prueba completa. Un menor valor en esta medida es lo deseado. Los datos que cumplan con este requisito se encuentran subrayados.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 5.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera en un 24,5% a los servidores Lighttpd y Apache en

la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Cherokee en la Arquitectura 1 supera en un 19,6% a Apache y Cherokee, en la Arquitectura 2 en un 22,9%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Apache en la Arquitectura 2 supera en un 22,2% a Lighttpd y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

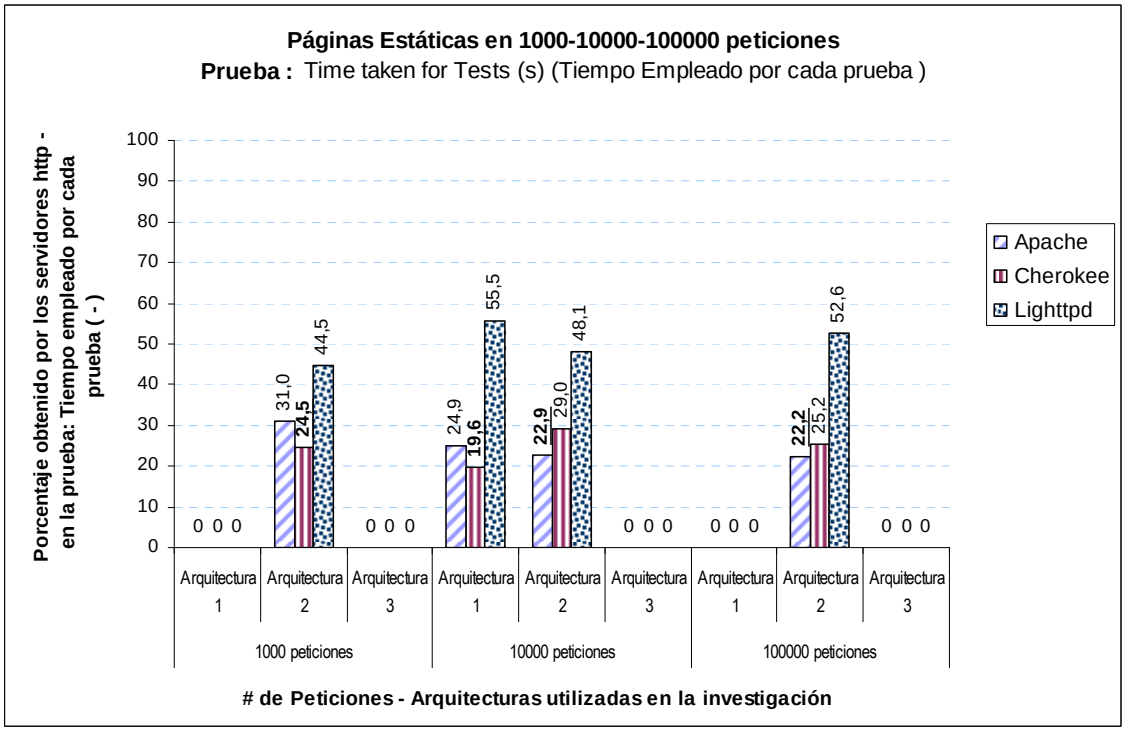


Ilustración IV- 5.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo Empleado por cada prueba en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 6.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba** cuyos datos numéricos se los puede

observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 25,9% a los servidores Apache y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Apache en la Arquitectura 2 supera con un 2,2% a Lighttpd y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Apache en la Arquitectura 2 supera con un 0,9% a Lighttpd y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

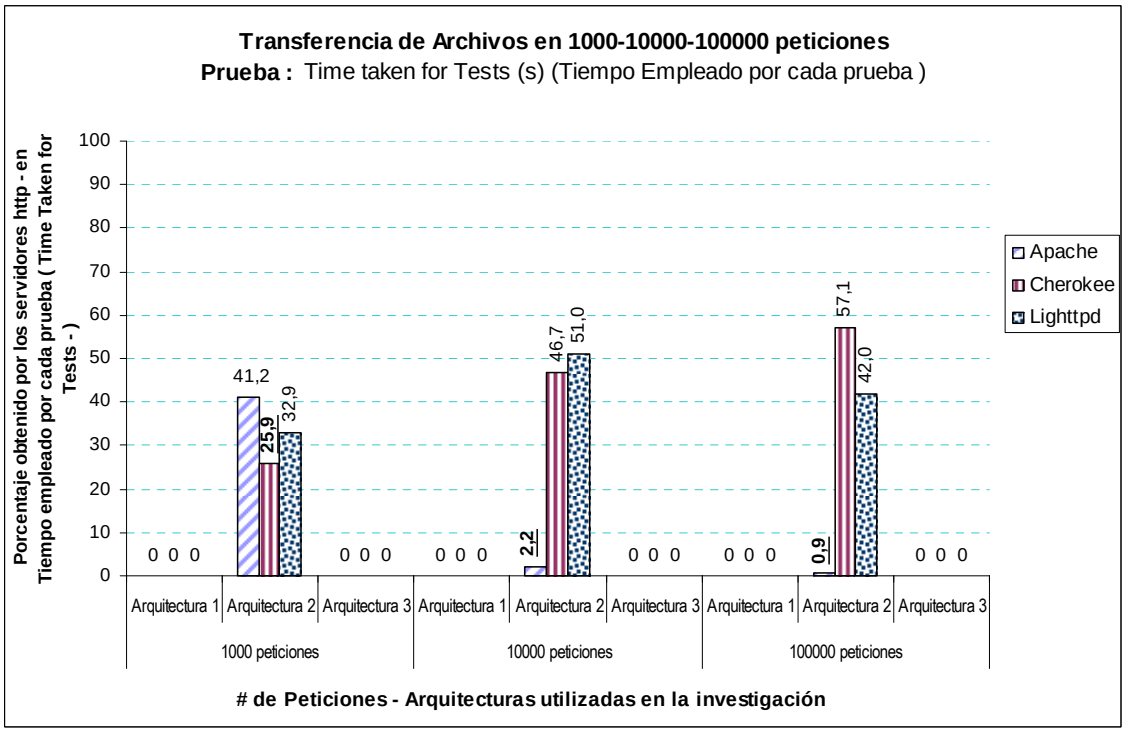


Ilustración IV- 6.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo Empleado por cada prueba en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 7.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd supera con un 2,2% a los servidores Cherokee y Apache en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Cherokee en la Arquitectura 2 supera con un 25,5% a Apache y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 1 supera con un 29,7% a Apache y Lighttpd, en la Arquitectura 2 Cherokee supera con un 9,4%, y en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

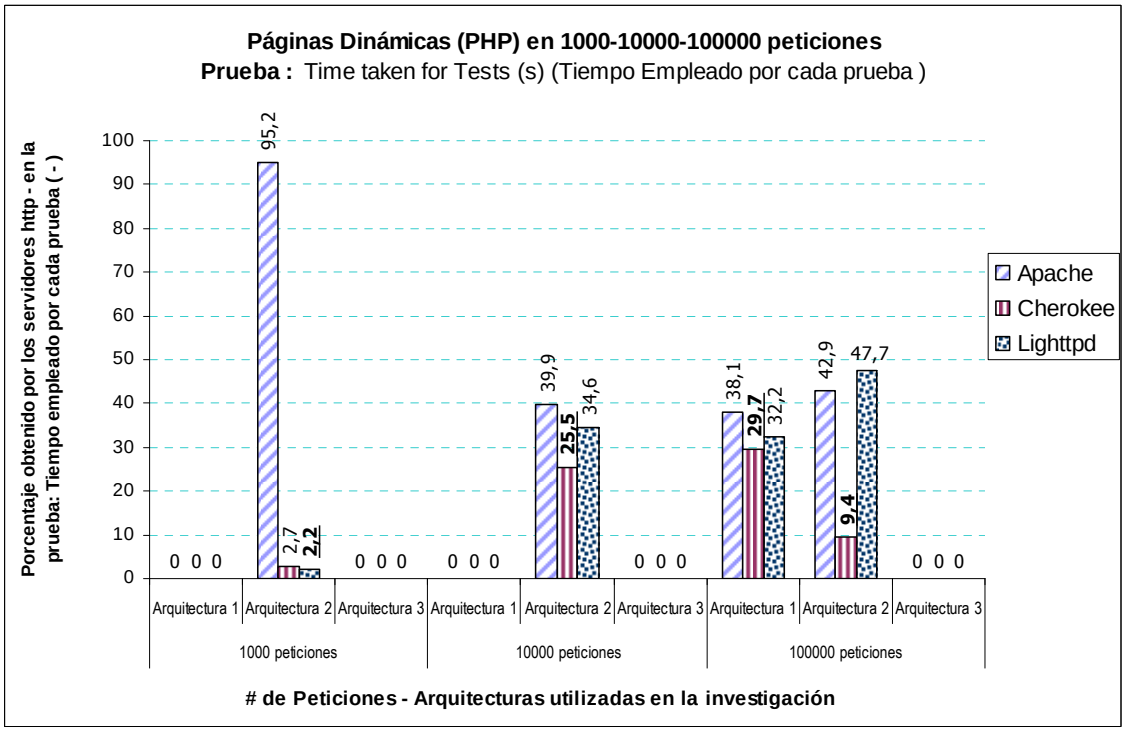


Ilustración IV- 7.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo Empleado por cada prueba en Páginas Dinámicas (PHP).

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 8.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 0,9% a los servidores Apache y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 2 Apache supera con un 2,2% a Apache y Lighttpd, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 2 supera con un 9,4% a Apache y Lighttpd, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

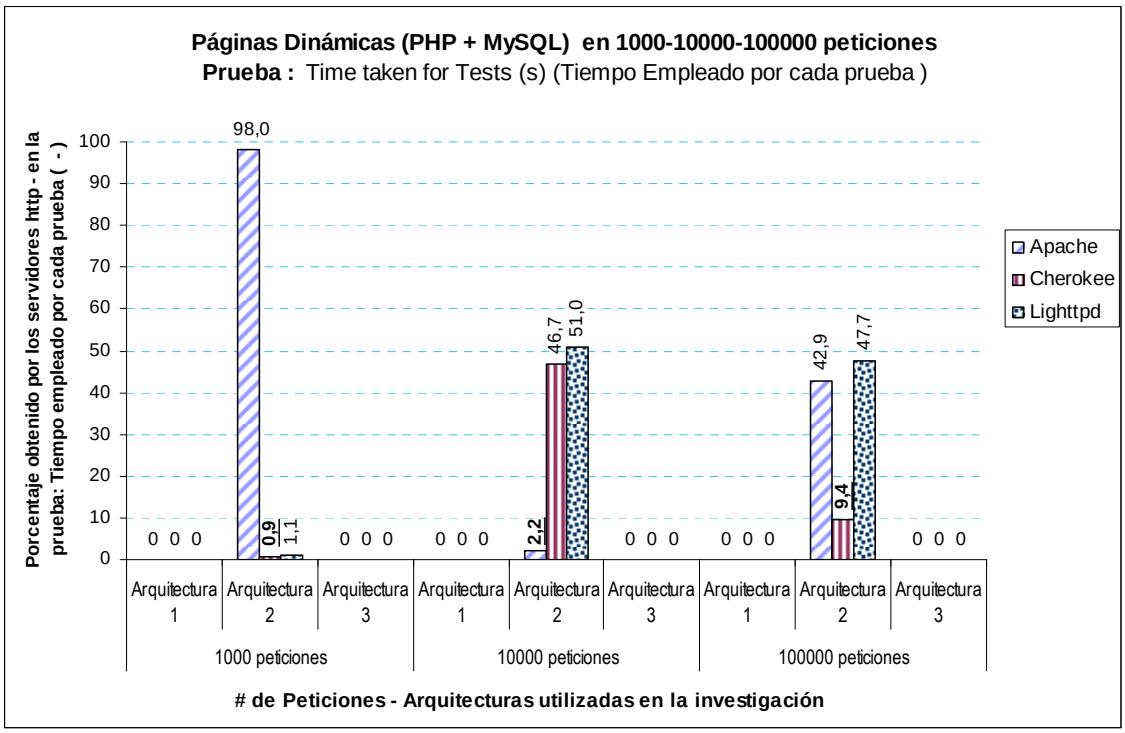


Ilustración IV- 8.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo Empleado por cada prueba

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo Empleado por cada prueba en Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL).

c) TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA

Transfer rate: indica la tasa de transferencia que el servidor es capaz de mantener durante en desarrollo de la prueba. Se mide en Kbytes/sec. Un mayor valor en esta medida indica que es mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 9.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) Tasa de Transferencia** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd supera con un 50,8% a los servidores Cherokee y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 51,9% a Apache y Cherokee, en la Arquitectura 2 en un 52,7%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 1 supera en un 53,3% a Apache y Cherokee, en la Arquitectura 2 en un 58,2%, y en la Arquitectura 3 en un 59,1%.

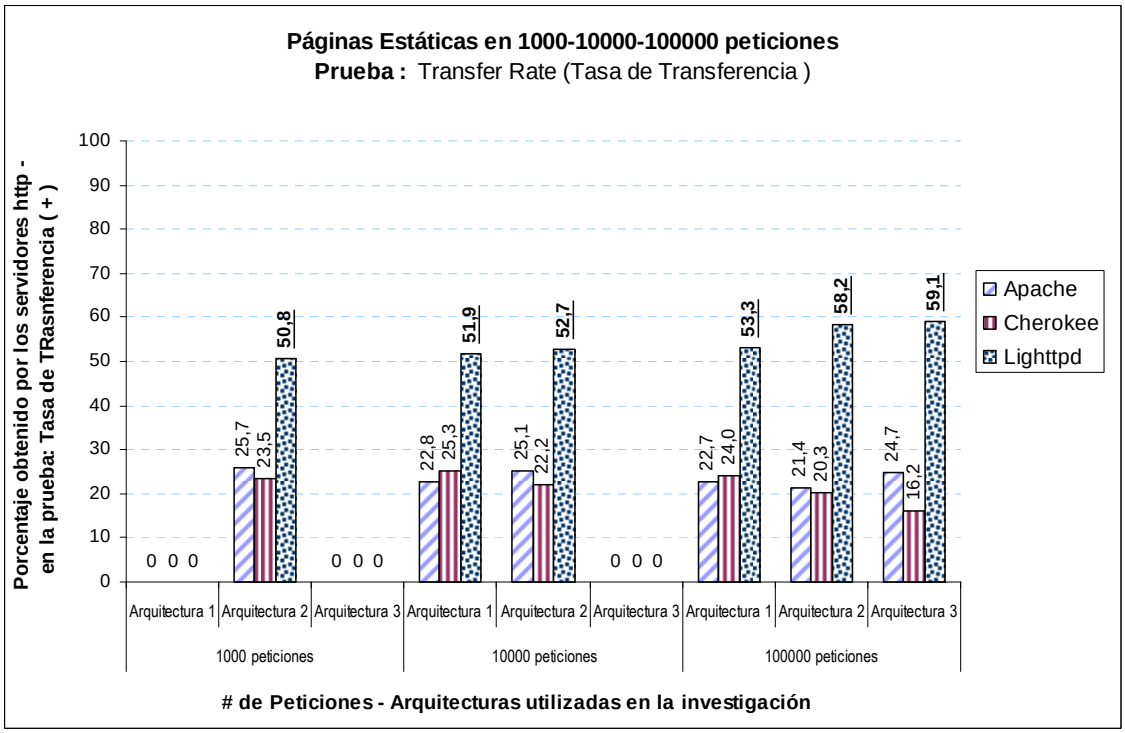


Ilustración IV- 9.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tasa de Transferencia en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 10.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) Tasa de Transferencia** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 25,9% a los servidores Apache y Lighttpd en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Apache en la Arquitectura 2 supera con un 2,2% a Cherokee y Cherokee, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 1 supera con un 26,5% a Apache y Cherokee, el servidor Apache supera con un 31,1% a Cherokee y Lighttpd, y en Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

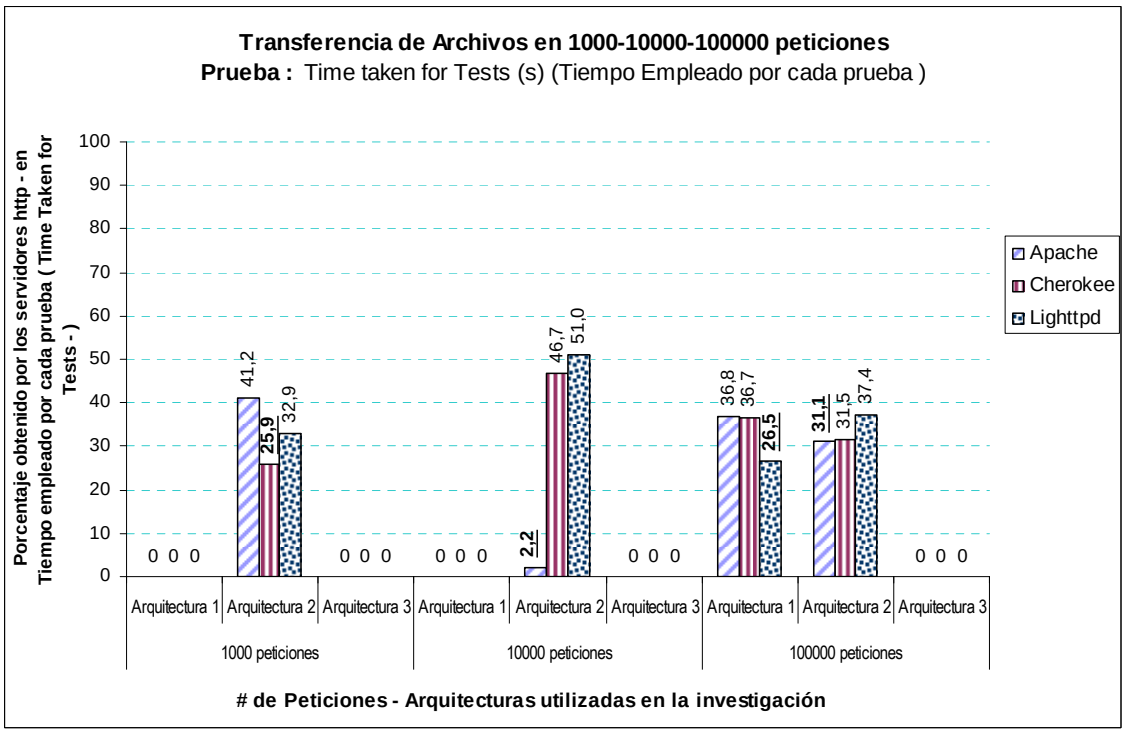


Ilustración IV- 10.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tasa de Transferencia en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 11.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (+) Tasa de Transferencia** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd supera con un 40,4% a los servidores Cherokee y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 36,2% a Apache y Cherokee, en la Arquitectura 2 con un 39,7%, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 51,1% a Apache y Cherokee, en la Arquitectura 2 con un 40,7%, y en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

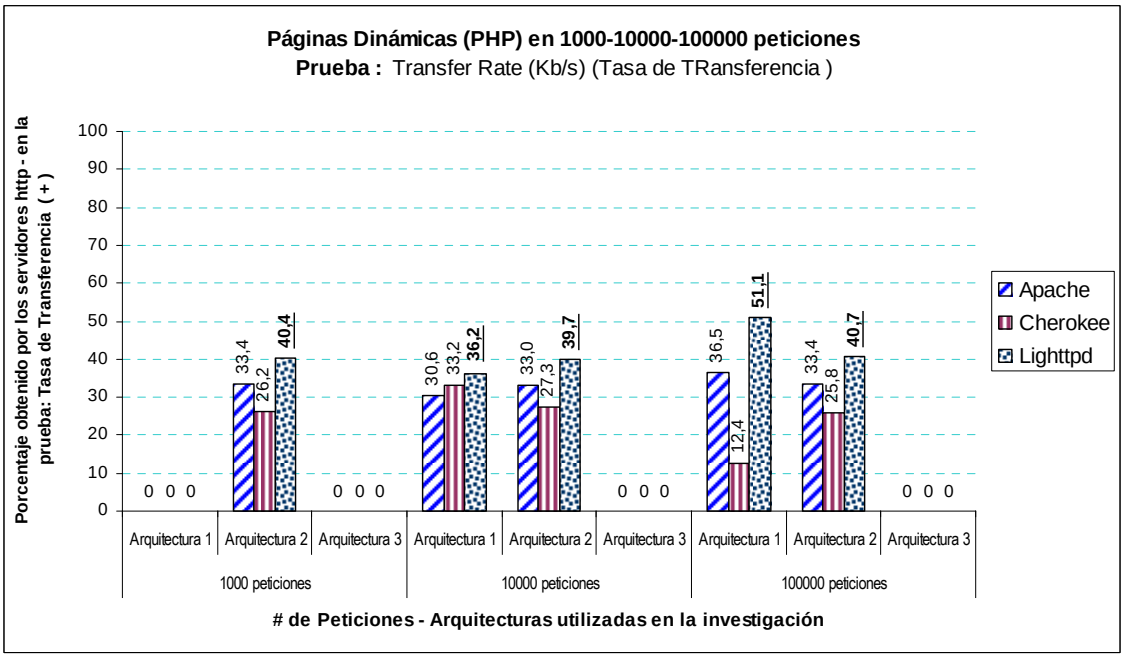


Ilustración IV- 11.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tasa de Transferencia en Páginas Dinámicas (PHP).

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 12.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (+) Tasa de Transferencia** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd en la Arquitectura 1 supera con un 37,1% a los servidores Apache y Lighttpd, en la Arquitectura 2 supera con 40,6% a los servidores Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd en la Arquitectura 1 supera con un 36,7% a los servidores Apache y Lighttpd, en la Arquitectura 2 supera con 43,8% a los servidores Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 51,3% a Apache y Cherokee, en la Arquitectura 2 Lighttpd supera con un 40,7% a Apache y Cherokee, y en la Arquitectura 3 Lighttpd supera con 46,1% a Apache y Cherokee.

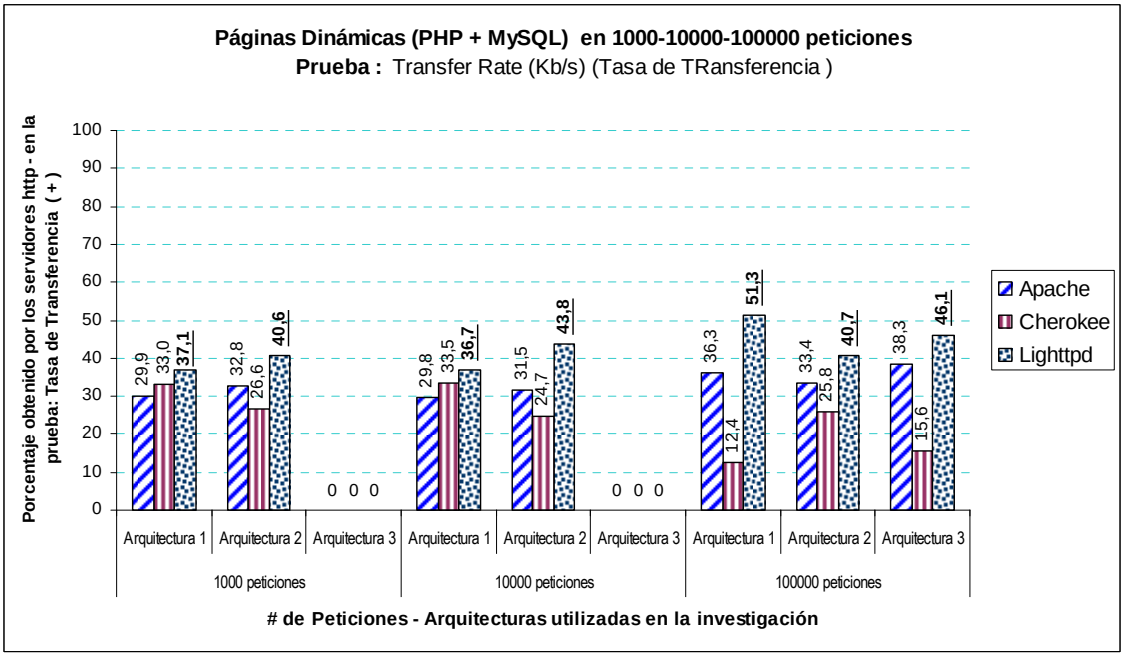


Ilustración IV- 12.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Tasa de Transferencia

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tasa de Transferencia en Páginas Dinámicas con Acceso a Base de Datos (PHP + MySQL).

d) TIME PER REQUEST (ms) - (TIEMPO POR PETICION)

Time per request: tiempo empleado en atender a cada petición. Un menor valor de esta medida es lo mejor**.**

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 13.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (-) Tiempo por Petición** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 23,5% a los servidores Cherokee y Apache en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Apache en la Arquitectura 1 supera con un 22,8% a Lighttpd y Cherokee, en la Arquitectura 2 el servidor Cherokee supera con un 22,2% a Apache y Lighttpd y en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Apache en la Arquitectura 1 supera con un 22,7% a Lighttpd y Cherokee, en la Arquitectura 2 el servidor Cherokee supera con un 20,3% a Apache y Lighttpd y en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

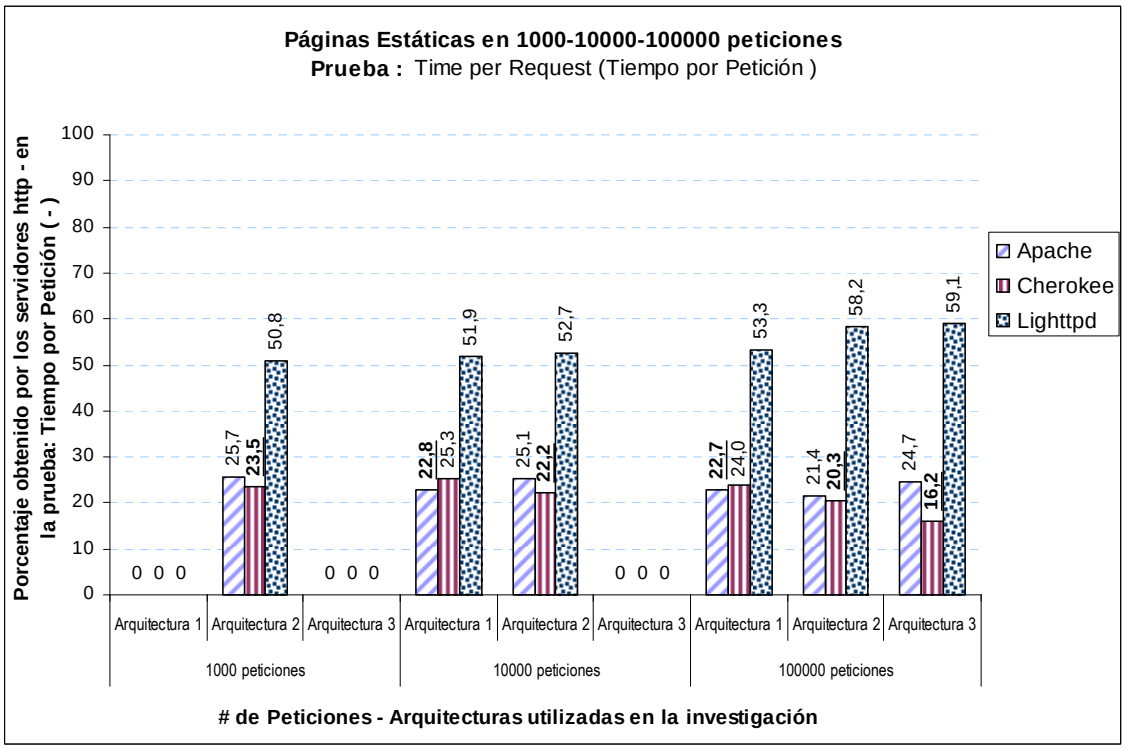


Ilustración IV- 13.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo por Petición en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 14.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) Tiempo por Petición** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 25,9% a los servidores Apache y Lighttpd en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Lighttpd en la Arquitectura 1 supera con un 26,9% a Apache y Cherokee, en la Arquitectura 2 Apache supera con 2,2% a Lighttpd y Cherokee y en la

Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 1 supera con un 26,5% a Apache y Cherokee, el servidor Apache supera con un 31,1% a Cherokee y Lighttpd, y en Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

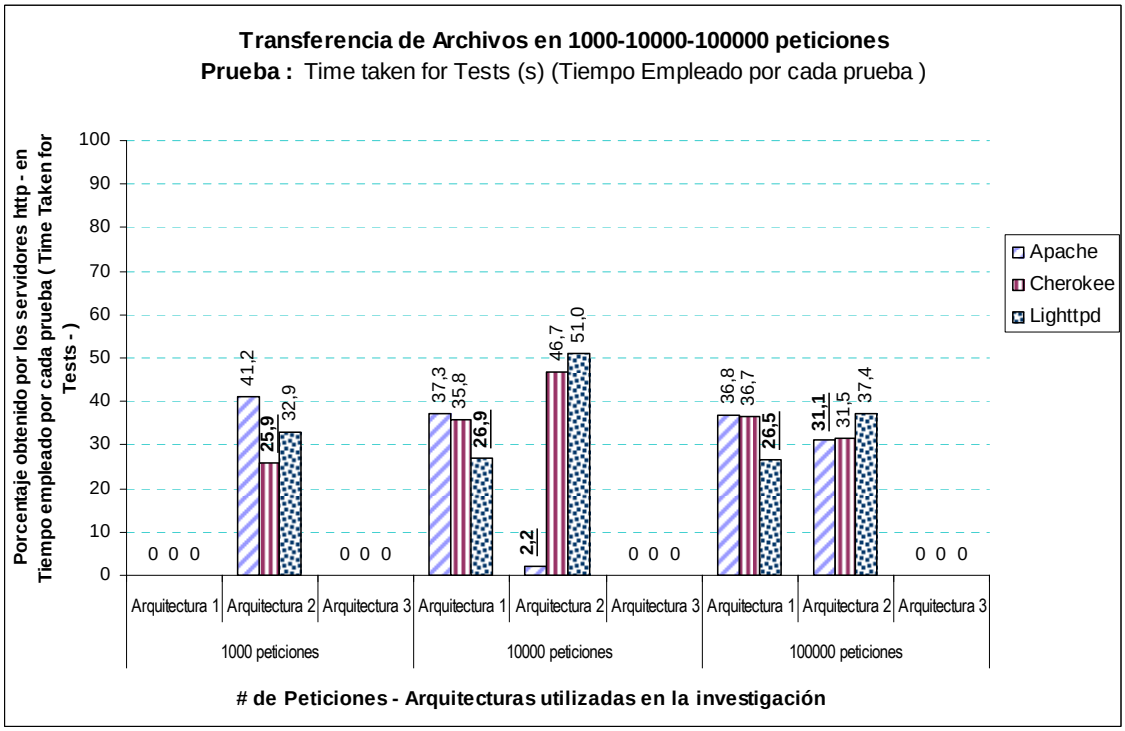


Ilustración IV- 14.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo por Petición en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 15.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (+) Tiempo por Petición** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones**

el servidor Lighttpd supera con un 29,1% a los servidores Cherokee y Lighttpd en la Arquitectura2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Cherokee en la Arquitectura 2 supera con un 25,5% a Apache y Cherokee, mientras que en la en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 2 supera con un 25,7% a Apache y Lighttpd, en la Arquitectura 3 Lighttpd supera con 21,3% supera a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

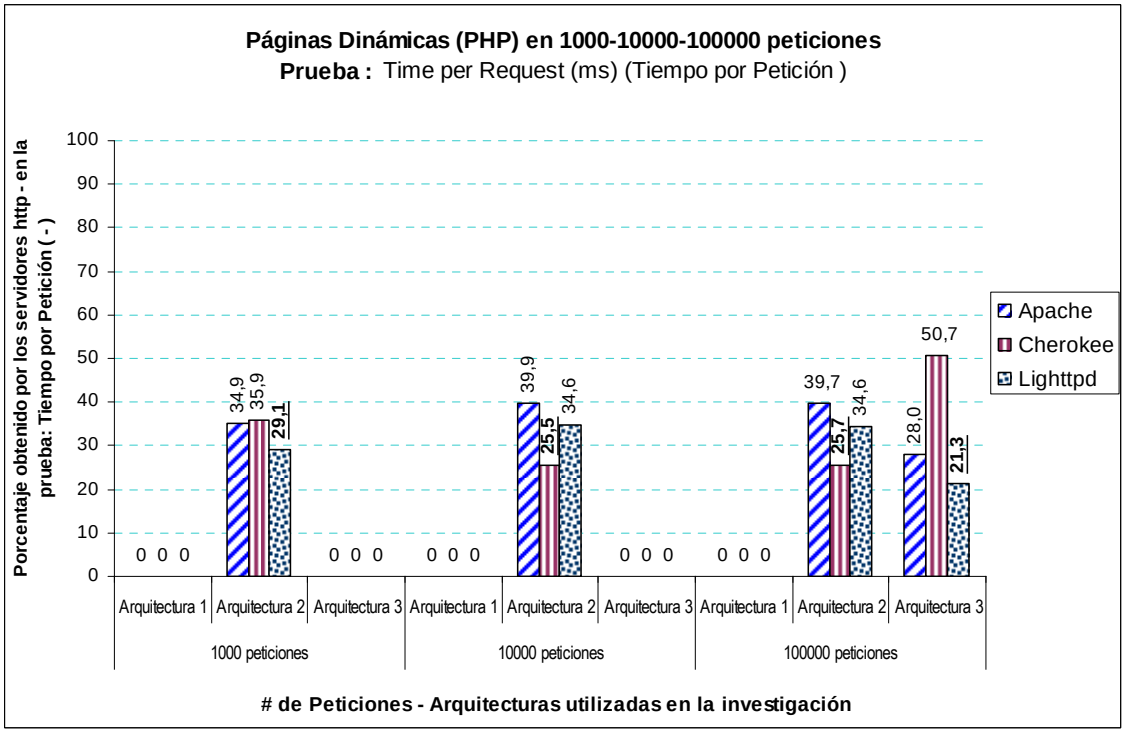


Ilustración IV- 15.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a Tiempo por Petición en Páginas Dinámicas (PHP).

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 16.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (+) Tiempo por Petición** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee en la Arquitectura 2 supera con un 26,5% a los servidores Apache y Lighttpd, en la Arquitectura 3 el servidor Apache supera con 23,4% a los servidores Lighttpd y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Cherokee en la Arquitectura 1 supera con un 30,3% a los servidores Apache y Lighttpd, en la Arquitectura 2 supera con 26,3% a los servidores Apache y Lighttpd, mientras que en la Arquitectura 3 Lighttpd supera con un 23,2% supera a Apache y Cherokee.

En **1000000 peticiones,** Cherokee en la Arquitectura 1 supera con un 29,9% a Apache y Lighttpd, en la Arquitectura 2 Cherokee supera con un 25,7% a Apache y Lighttpd, y en la Arquitectura 3 Lighttpd supera con 24,5% a Apache y Cherokee.

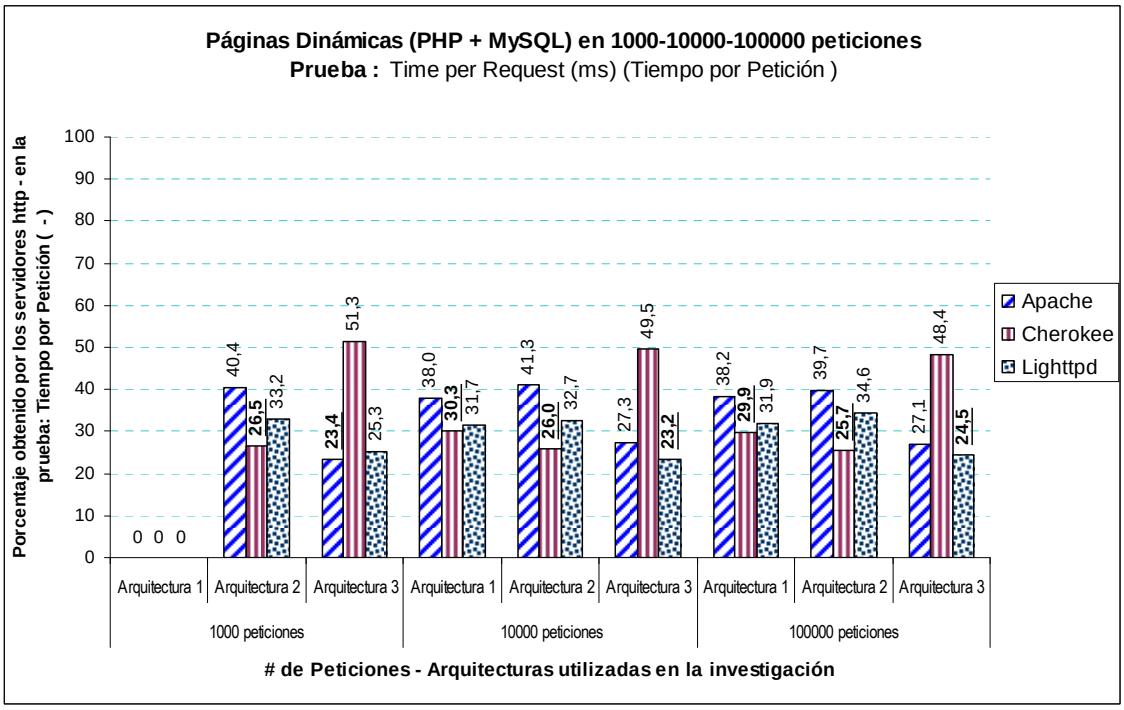


Ilustración IV- 16.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) Tiempo por Petición

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor Lighttpd supera en la prueba Páginas Dinámicas (PHP + MySQL) - **Tiempo por petición** con una cantidad de **77,685** ➔ 23,2% a Apache y Cherokee

e) %CPU USER (Uso de CPU)

%user, porcentaje del uso de la CPU empleado por los procesos del usuario. Mientras menos uso de CPU utilice un servidor, se lo tomará como el mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 16.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (-) %Uso de CPU** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 1,7% a los servidores Lighttpd y Cherokee en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

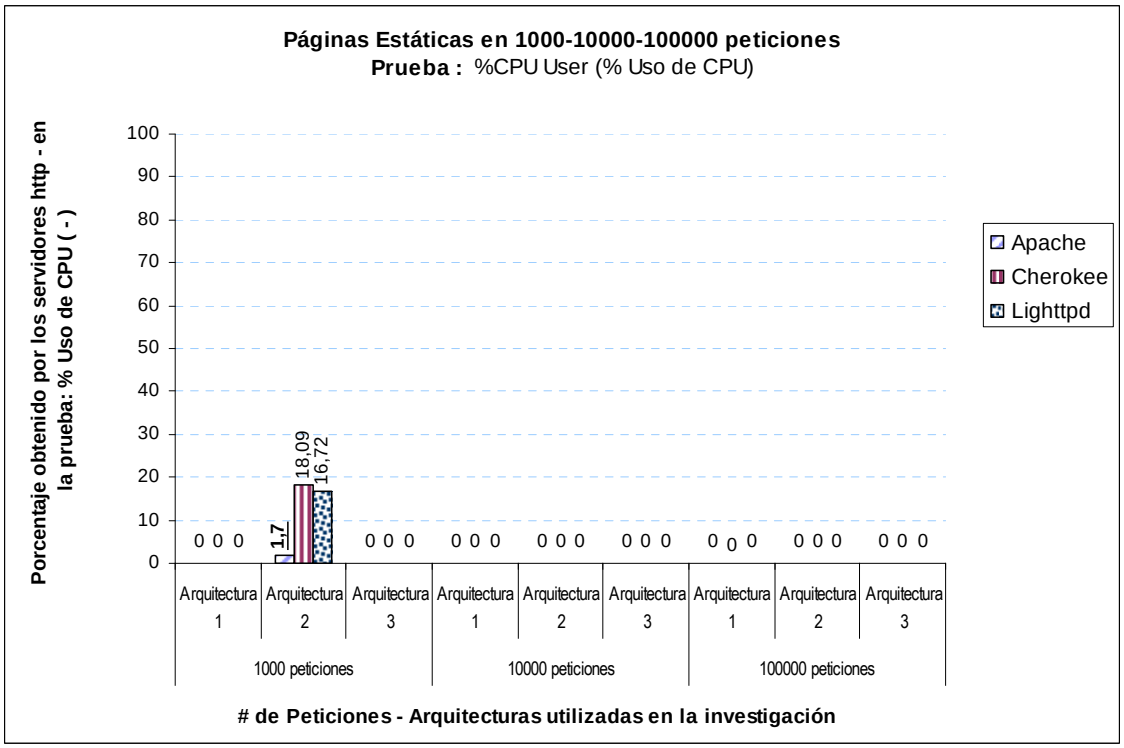


Ilustración IV- 17.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de

estos servidores su servicio no varía en lo que se refiere a %Uso de CPU en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 18.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (-) %Uso de CPU** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 29,55% a los servidores Cherokee y Lighttpd en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

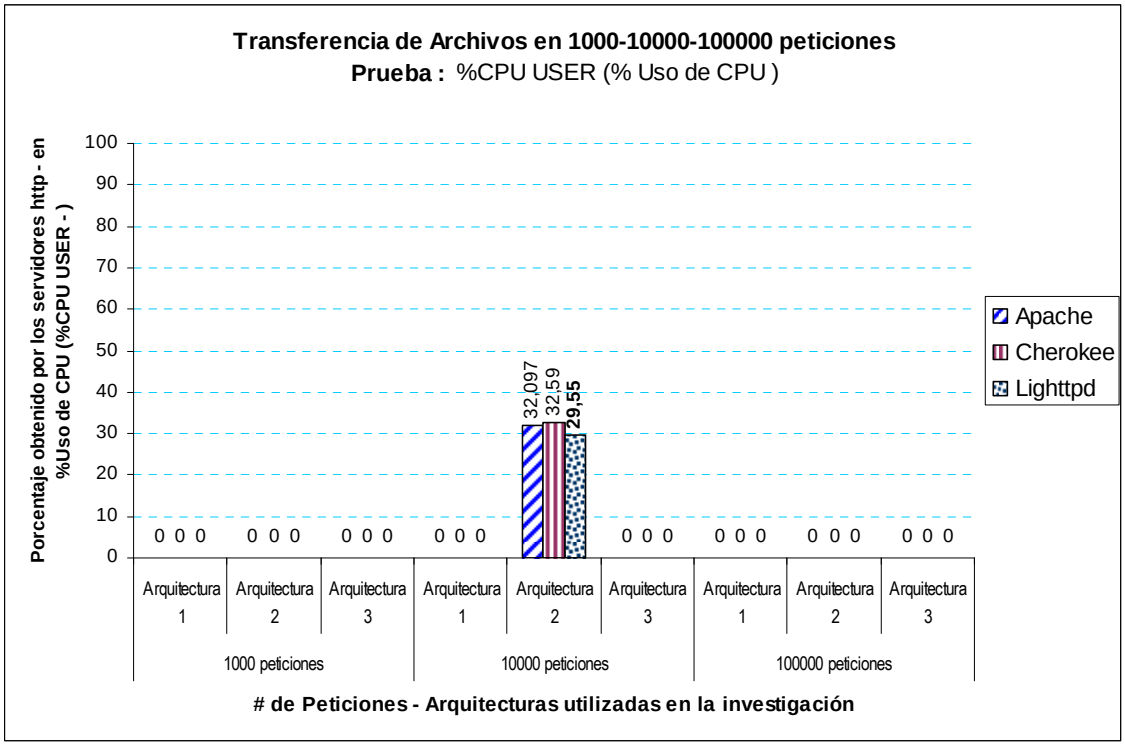


Ilustración IV- 18.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de

estos servidores su servicio no varía en lo que se refiere a %Uso de CPU en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 19.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (-) %Uso de CPU** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** en la Arquitectura 1, en la Arquitectura2, mientras que y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** Apache en la Arquitectura 1 supera con un 15,7% a Lighttpd y Cherokee, mientras que en la en la Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Lighttpd en la Arquitectura 2 supera con un 15,4% a Apache y Lighttpd, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

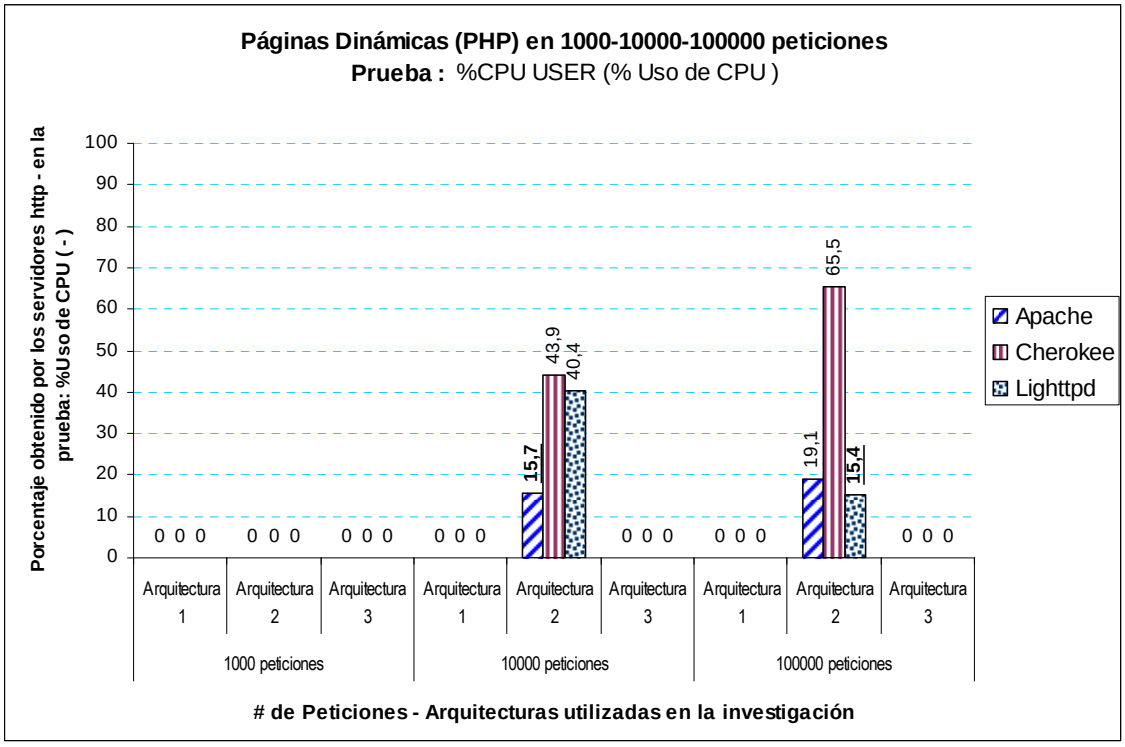


Ilustración IV- 19.- Páginas Dinámicas – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a %Uso de CPU en Páginas Dinámicas (PHP).

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 20.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (-) %Uso de CPU** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache en la Arquitectura 2 supera con un 16,5% a los servidores Cherokee y Lighttpd, en la Arquitectura 3 el servidor Lighttpd supera con 50,86% a los servidores Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd en la Arquitectura 2 supera con un 29,55% a los servidores Apache y Cherokee, en la Arquitectura 3 supera con 50,86% a los servidores Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** Cherokee en la Arquitectura 2 supera con un 20,43% a Apache y Lighttpd, en la Arquitectura 3 Lighttpd supera con un 50,86% a Apache y Cherokee, y en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

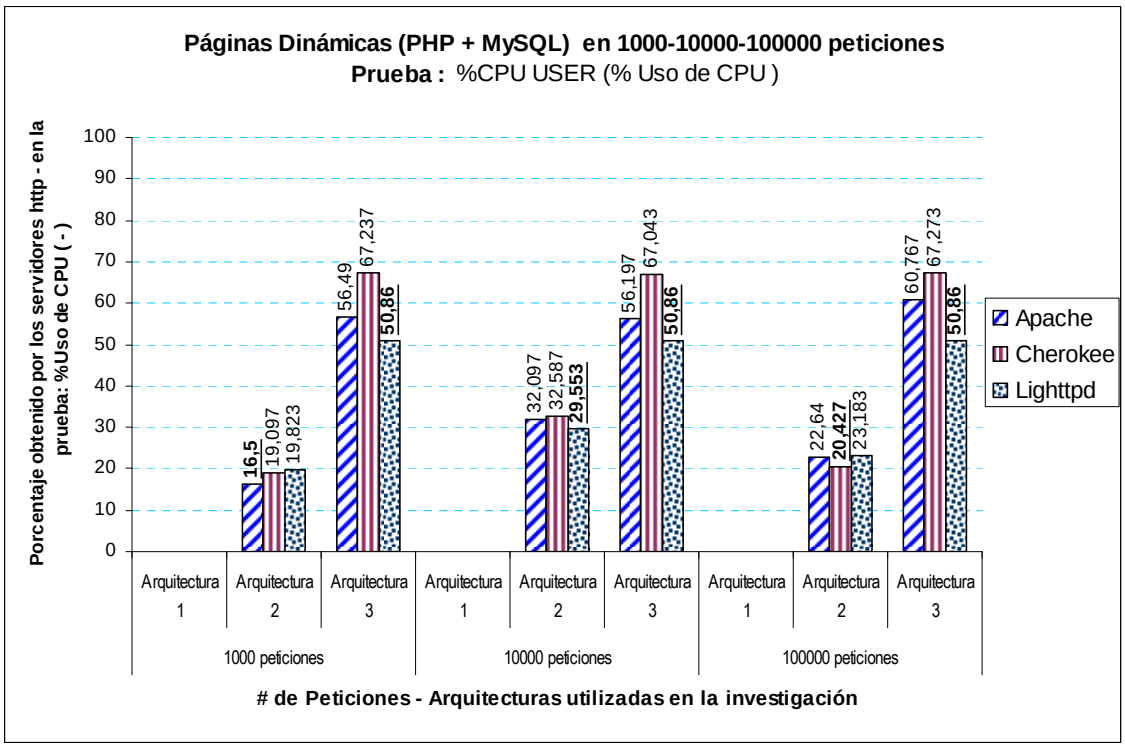


Ilustración IV- 20.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) %Uso de CPU

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP + MySQL) - **%Uso de CPU** con una cantidad de **50,86%** a Apache y Cherokee

f) CPU por sistema y Kernel

%system, porcentaje del uso de la CPU empleado por los procesos del kernel y del sistema. Mientras menos uso de procesos del kernel y del sistema utilice un servidor, se lo tomará como el mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 21.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU por Sistema y Kernel** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** el servidor Lighttpd supera en un 23,94% a los servidores Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 y Arquitectura 2 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

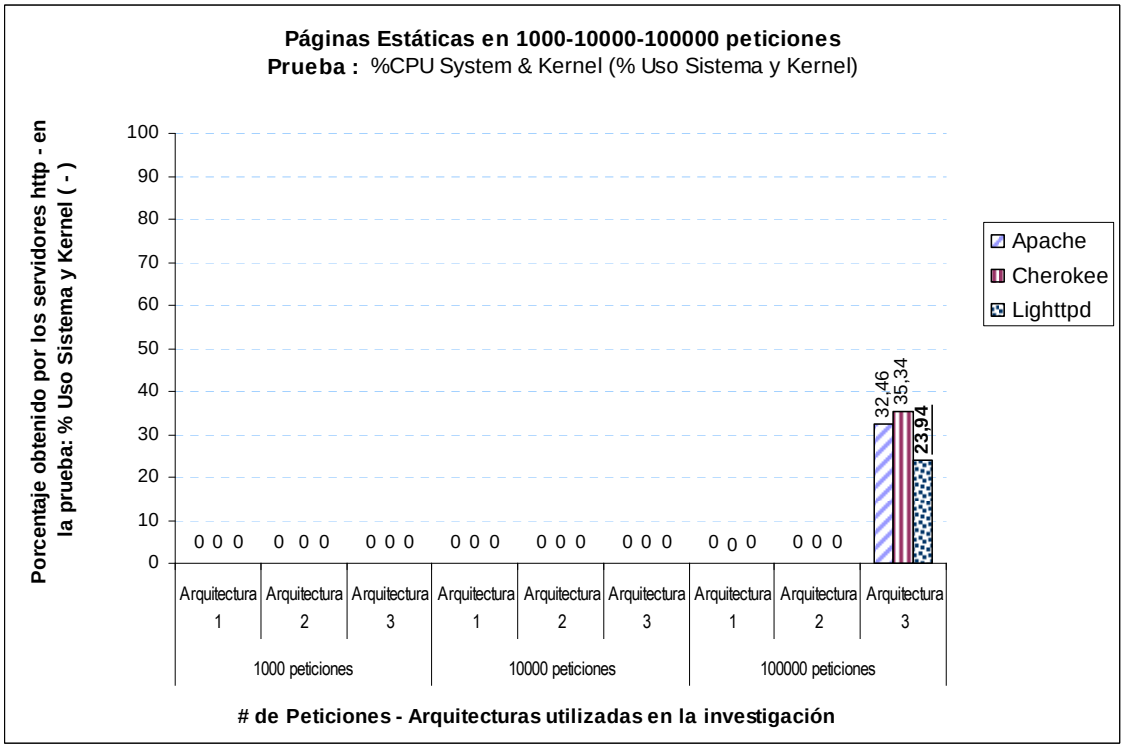


Ilustración IV- 21.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de estos servidores su servicio no varía en lo que se refiere a %CPU por Sistema y Kernel en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 22.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU por Sistema y Kernel** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 3,88% a los servidores Cherokee y Lighttpd en la Arquitectura 2, en la Arquitectura 3 el servidor Lighttpd supera con 23,94% a Apache y Cherokee, y Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 2 Apache supera con un 5,57% a Cherokee y Lighttpd, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Lighttpd supera con un 12,03% a Cherokee y Apache, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

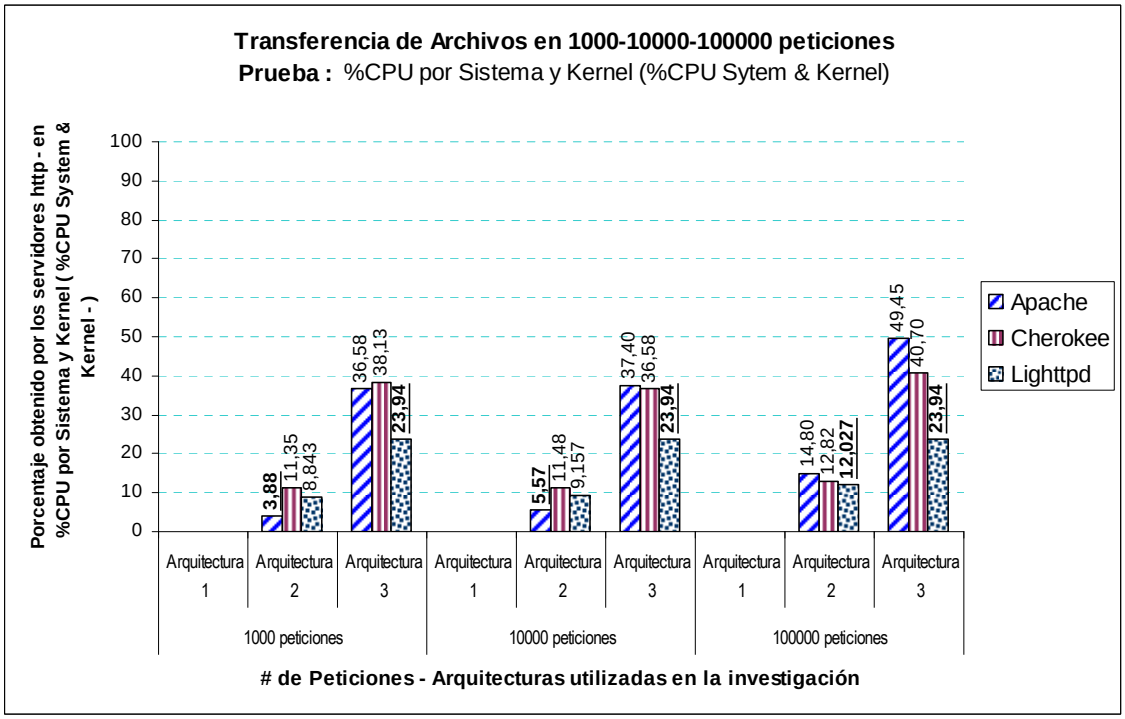


Ilustración IV- 22.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Transferencia de Archivos - **%CPU por Sistema y Kernel** con una cantidad de **23,94%** a Apache y Cherokee

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 23.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU por Sistema y Kernel** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd supera con un 12,03% a los servidores Cherokee y Apache en la Arquitectura 2, en la Arquitectura 3 el servidor Lighttpd supera con 23,94% a Apache y Cherokee, y Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 2 Lighttpd supera con un 12,10% a Cherokee y Lighttpd, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Lighttpd supera con un 12,79% a Cherokee y Apache, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

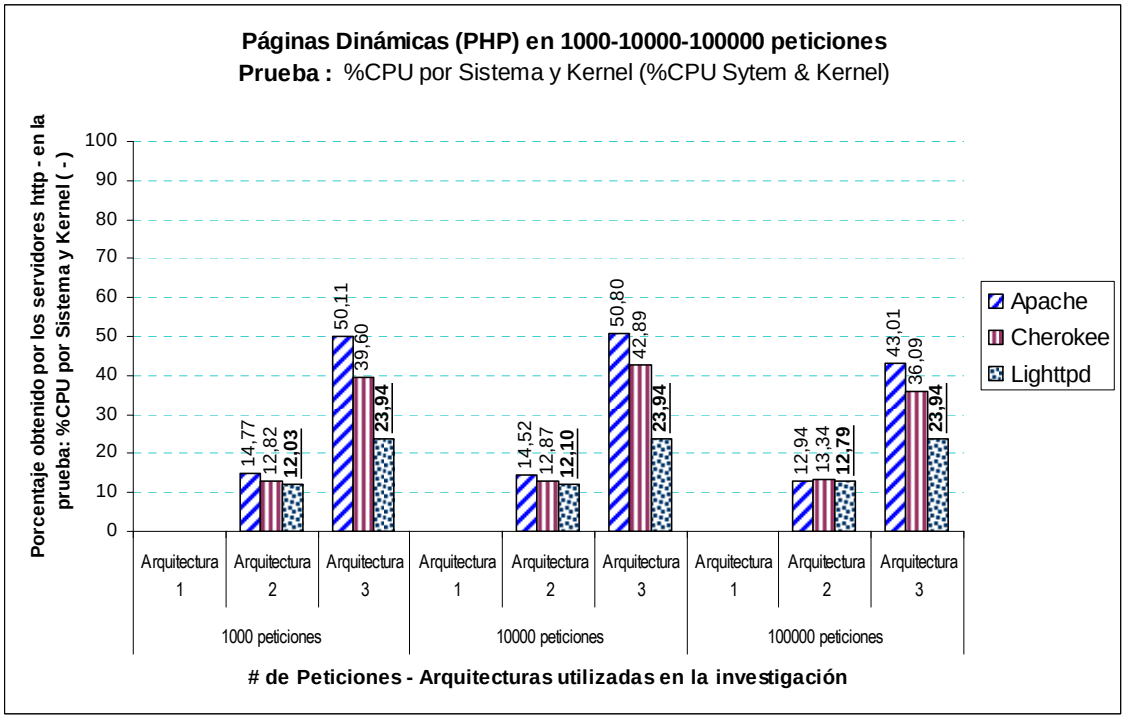


Ilustración IV- 23.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP) - **%CPU por Sistema y Kernel** con una cantidad de **23,94%** a Apache y Cherokee.

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 24.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU por Sistema y Kernel** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Lighttpd supera con un 12,79% a los servidores Cherokee y Apache en la Arquitectura 2, en la Arquitectura 3 el servidor Lighttpd supera con 23,94% a Apache y Cherokee, y Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 2 Apache supera con un 5,57% a Cherokee y Lighttpd, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Apache supera con un 11,77% a Cherokee y Lighttpd, Arquitectura 3 Lighttpd supera con un 23,94% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

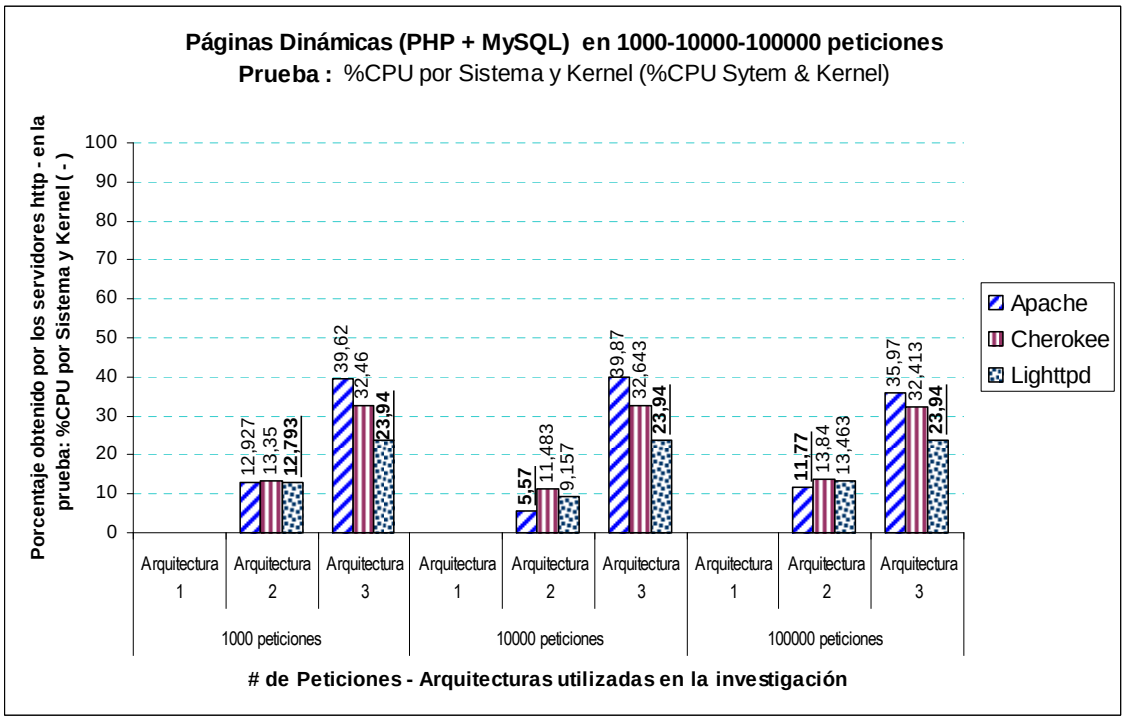


Ilustración IV- 24.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) %CPU por Sistema y Kernel

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP + MySQL) - **%CPU por Sistema y Kernel** con una cantidad de **23,94%** a Apache y Cherokee.

g) %CPU IDLE (CPU DISPONIBLE)

%idle, porcentaje de uso de CPU disponible para nuevos procesos. Mientras mas CPU disponible presente un servidor, se lo tomará como el mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 25.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) % CPU Disponible** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

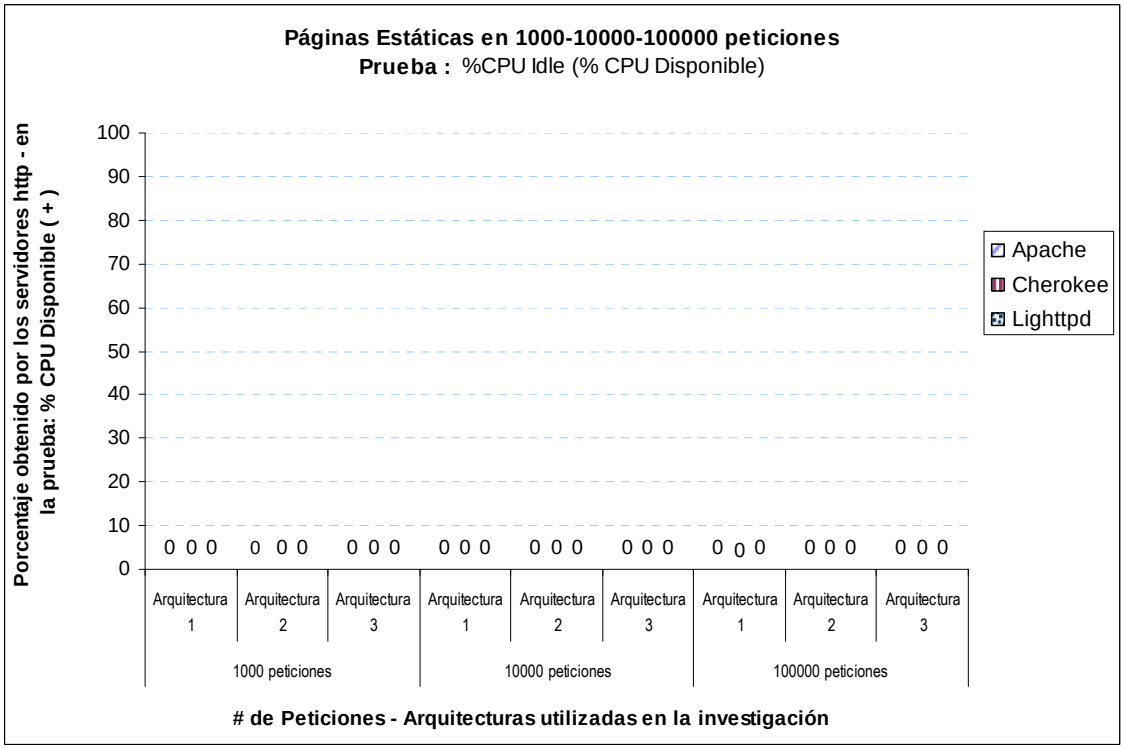


Ilustración IV- 25.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de

estos servidores su servicio no varía en lo que se refiere a % Uso CPU Disponible en Páginas Estáticas.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 26.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU Disponible** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 1, Arquitectura 2 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Lighttpd supera con un 78,69% a Cherokee y Apache, Arquitectura 3 Lighttpd supera con un 6,76% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

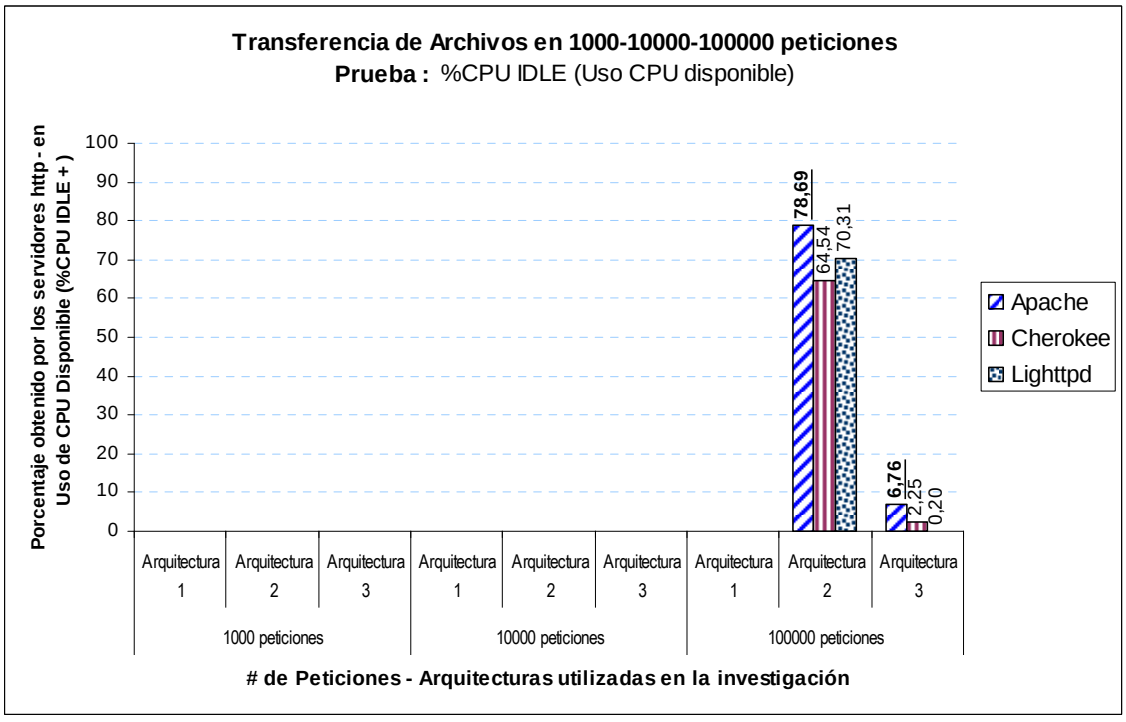


Ilustración IV- 26.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible

Concluyendo en la Arquitectura 3 – 10000 peticiones no existe diferencia significativa entre los servidores http, funcionan de forma similar esto significa que al instalar cualquiera de

estos servidores su servicio no varía en lo que se refiere a % Uso CPU Disponible en Transferencia de Archivos.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 27.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU Disponible** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 78,56% a los servidores Cherokee y Lighttpd en la Arquitectura 2, en la Arquitectura 3 el servidor Apache supera con 5,50% a Lighttpd y Cherokee, y Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 2 Apache supera con un 77,44% a Cherokee y Lighttpd, Arquitectura 3 Lighttpd supera con un 5,60% a Lighttpd y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Apache supera con un 35,0% a Cherokee y Lighttpd, Arquitectura 3 Apache supera con un 82,8% a Lighttpd y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

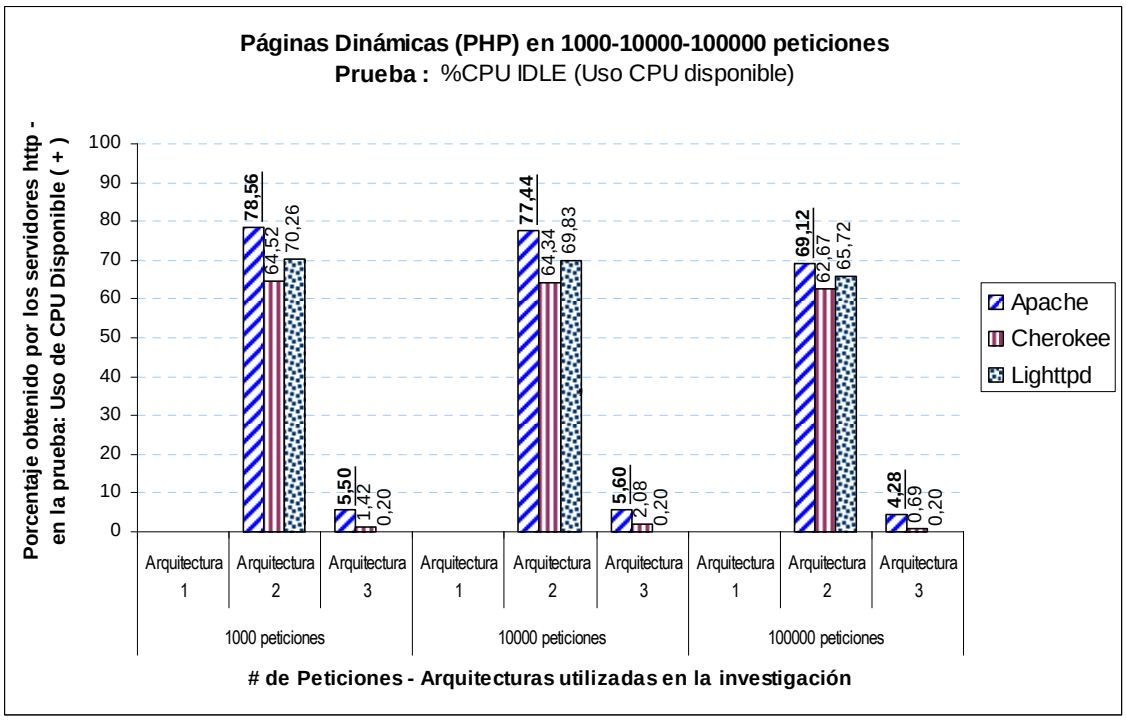


Ilustración IV- 27.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Apache** supera la prueba Páginas Dinámicas (PHP) - **%CPU Disponible** con una cantidad de **5,60%** a Lighttpd y Cherokee.

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 28.- Páginas Dinámicas (PHP) – 1000, 10000, 1000000 peticiones – Prueba: (-) %CPU Disponible** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Apache supera con un 69,05% a los servidores Cherokee y Lighttpd en la Arquitectura 2, en la Arquitectura 3 el servidor Apache supera con 3,72% a Lighttpd y Cherokee, y Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** en la Arquitectura 3 Apache supera con un 5,60% a Cherokee y Lighttpd, mientras que en la Arquitectura 1 y Arquitectura 2 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** en la Arquitectura 2 Apache supera con un 64,2% a Cherokee y Lighttpd, Arquitectura 3 Apache supera con un 4,28% a Lighttpd y Cherokee, mientras que

en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

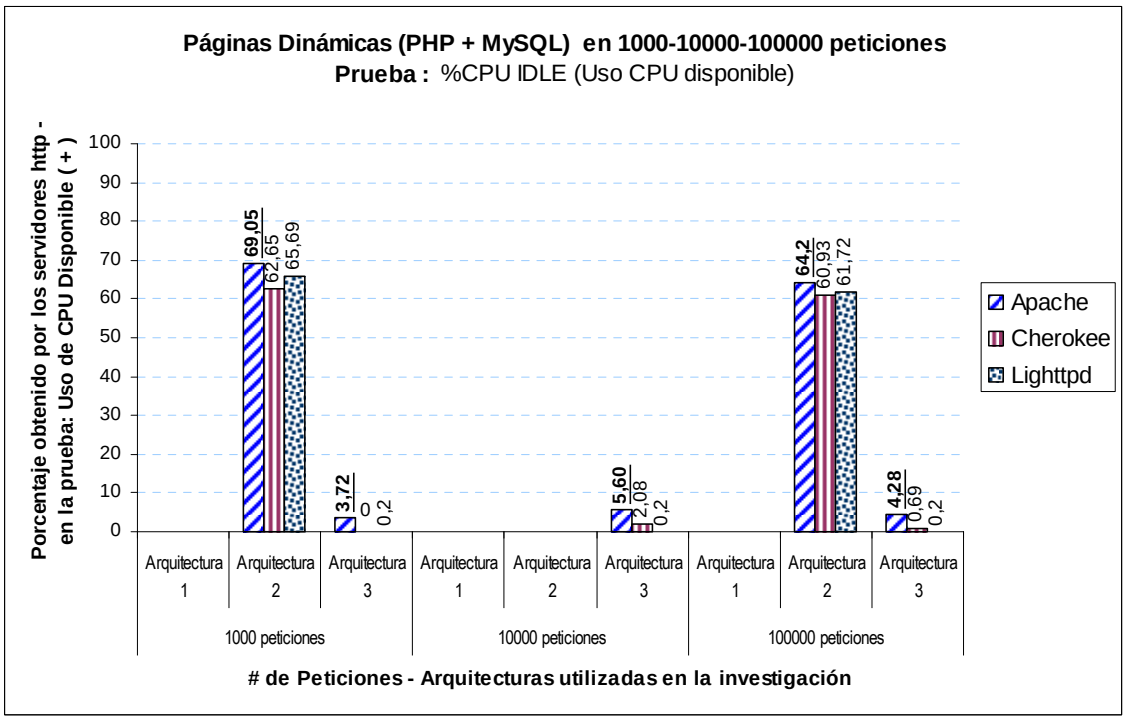


Ilustración IV- 28.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) % Uso CPU Disponible

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Apache** supera la prueba Páginas Dinámicas (PHP + MySQL) - **%CPU Disponible** con una cantidad de **5,60%** a Lighttpd y Cherokee.

h) KBMEMUSED - %MEMORIA RAM UTILIZADA

Kbmemused, Kilobytes de la memoria RAM ocupados por los distintos procesos que se están ejecutando. Mientras menos memoria RAM utilice en los distintos procesos un servidor, se lo tomará como el mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 29.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) RAM Utilizada** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones,** el servidor Lighttpd supera a en 33,3% a Apache y Cherokee en la Arquitectura 2, mientras que en la Arquitectura 1 y Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd supera a en 33,3% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera en 32,7% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **100000 peticiones,** el servidor Lighttpd supera a en 33,3% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera en 32,7% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

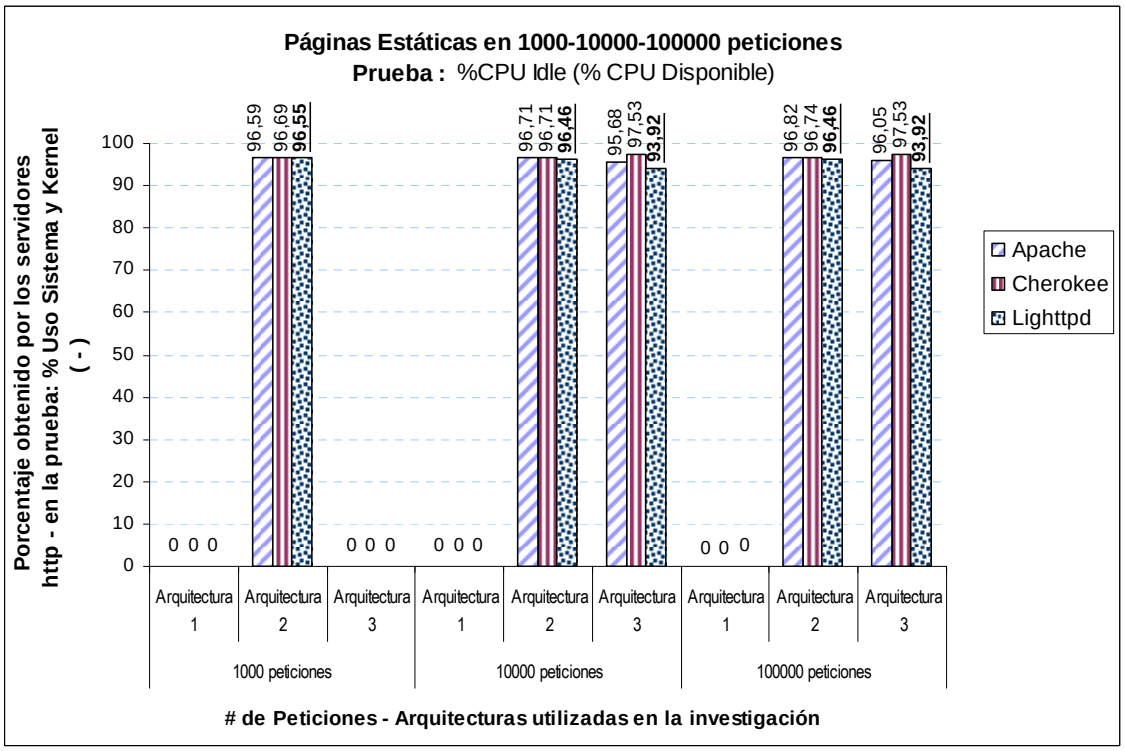


Ilustración IV- 29.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Estáticas - **RAM Utilizada** con una cantidad de **93,92%** a Apache y Cherokee.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 30.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) RAM Utilizada** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones,** el servidor Lighttpd supera a en 96,46% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd supera a en 96,49% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **100000 peticiones,** el servidor Lighttpd supera a en 96,54% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee,

mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

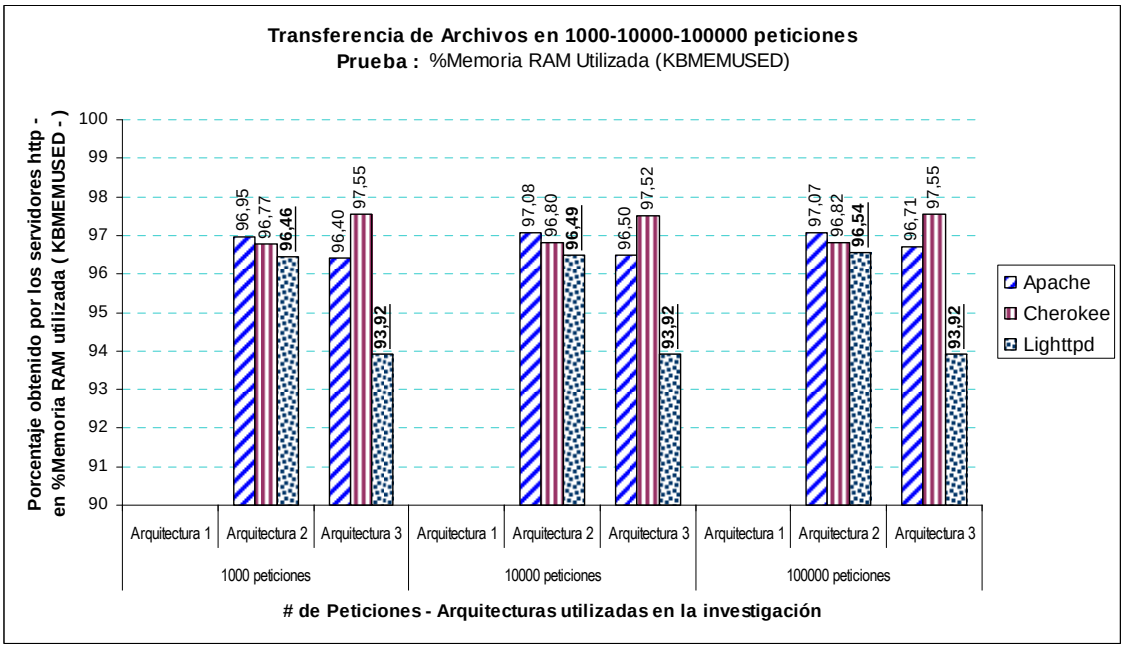


Ilustración IV- 30.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Transferencia de Archivos - **RAM Utilizada** con una cantidad de **93,92%** a Apache y Cherokee.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 31.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) RAM Utilizada** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones,** el servidor Lighttpd supera a en 96,59% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd supera a en 96,65% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **100000 peticiones,** el servidor Lighttpd supera a en 96,67% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

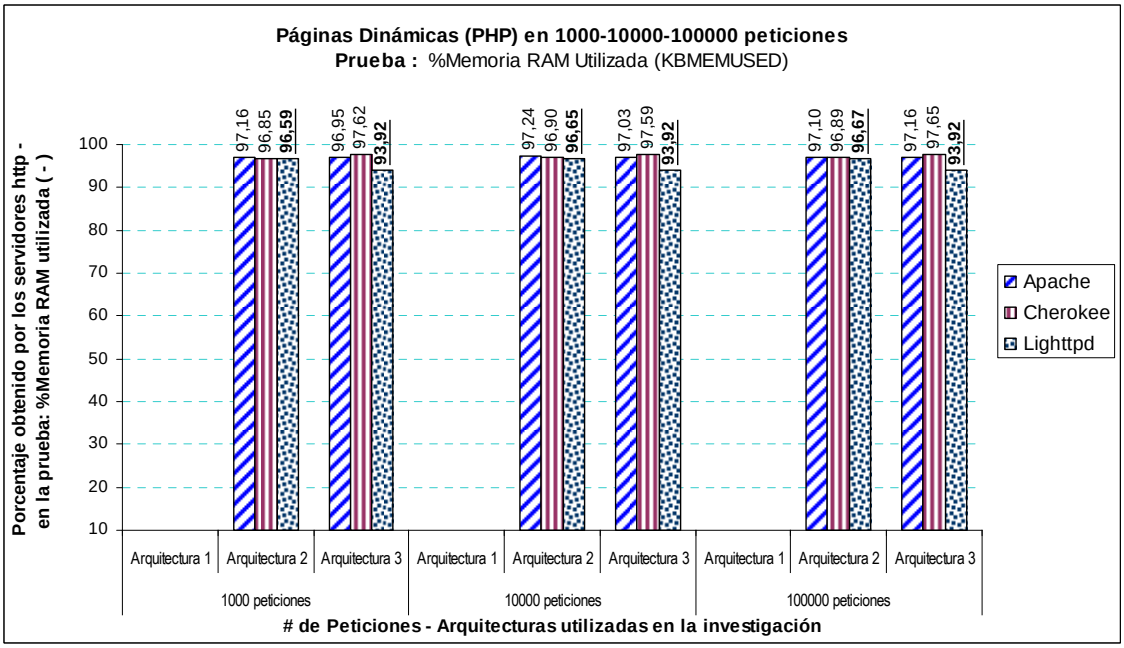


Ilustración IV- 31.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP) - **RAM Utilizada** con una cantidad de **93,92%** a Apache y Cherokee.

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 32.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) RAM Utilizada** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones,** el servidor Lighttpd supera a en 96,70% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Lighttpd supera a en 96,49% a Apache y Cherokee en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee,

mientras que en la Arquitectura 3 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **100000 peticiones,** el servidor Apache supera a en 96,73% a Cherokee y Lighttpd en la Arquitectura 2, en la Arquitectura 3 Lighttpd supera a en 93,92% a Apache y Cherokee, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

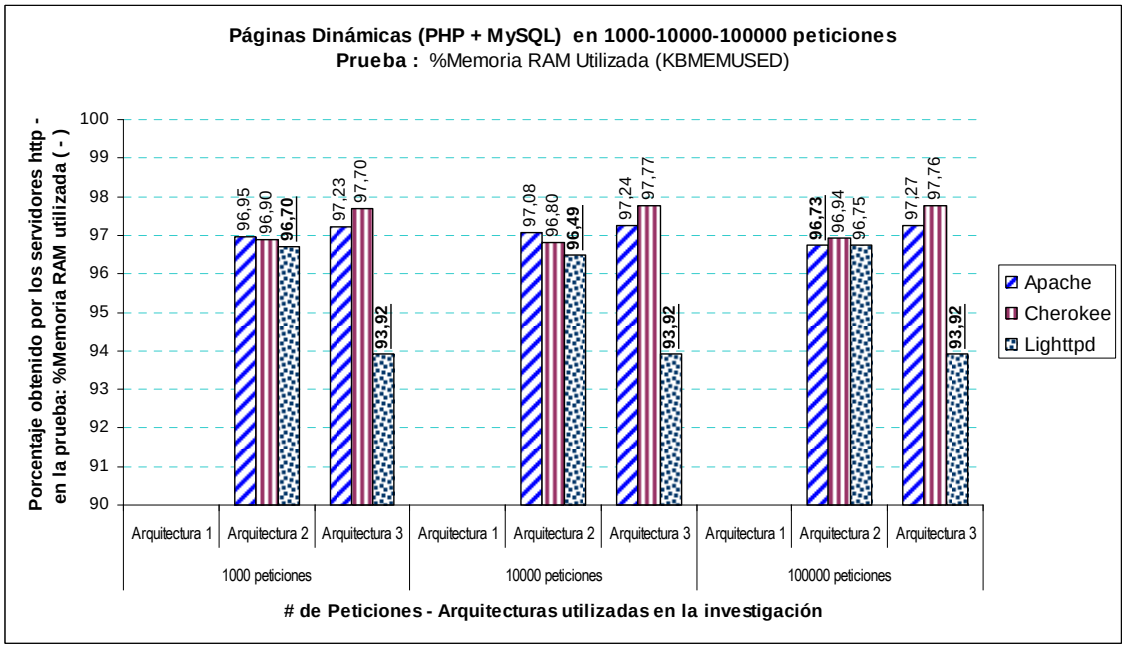


Ilustración IV- 32.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (-) % Memoria RAM utilizada

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP + MySQL) - **RAM Utilizada** con una cantidad de **93,92%** a Apache y Cherokee.

i) BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS – DISCO

Bread/s, cantidad total de datos leídos desde los dispositivos de almacenamiento masivo en bloques por segundos. Mientras más cantidad total de datos leídos desde los dispositivos de almacenamiento presente un servidor, se lo tomará como el mejor.

Páginas Estáticas – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 33.- Páginas Estáticas – 1000, 10000, 1000000 peticiones – Prueba: (+) BREDS** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 55,7% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 70,3% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Cherokee supera con un 55,8% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 72,2% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** el servidor Lighttpd supera con un 35,0% a los servidores Apache y Cherokee en la Arquitectura 2, Lighttpd supera con un 81,3% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

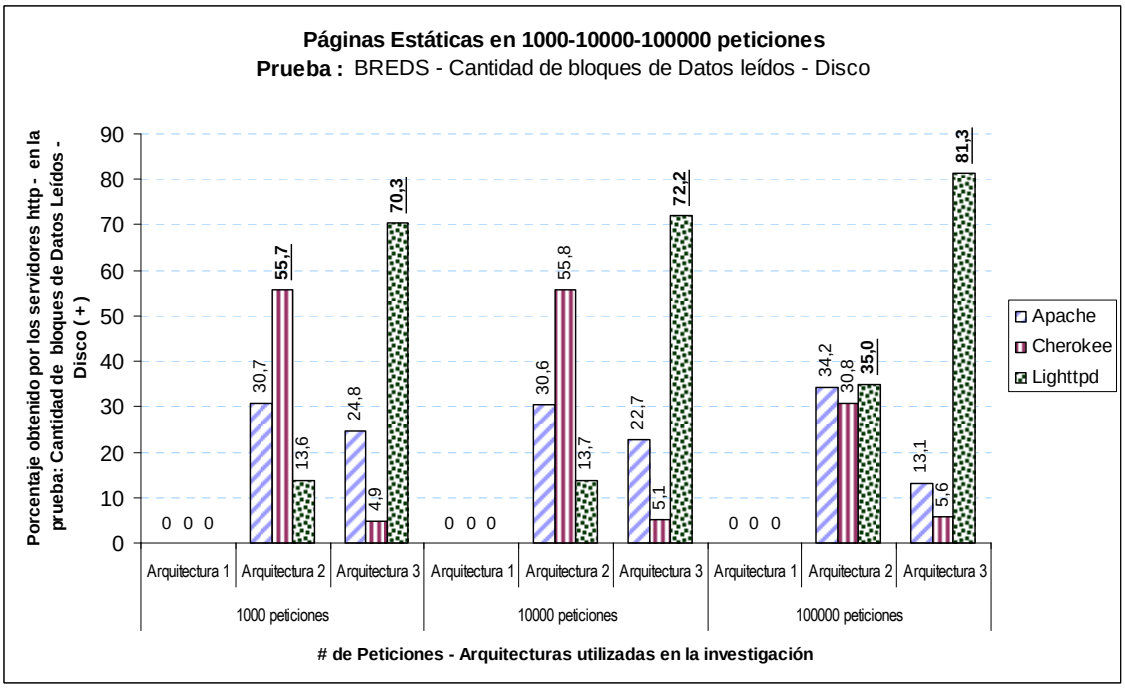


Ilustración IV- 33.- Páginas Estáticas – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Estáticas - **BREDS – Cantidad de Bloques de Datos leídos** con un 72,2% a Apache y Cherokee.

Páginas Transferencia de Archivos – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 34.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) BREDS** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 61,3% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 81,5% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Cherokee supera con un 57,5% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 83,0% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** el servidor Cherokee supera con un 62,5% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 89,0% a Apache y Cherokee

en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http

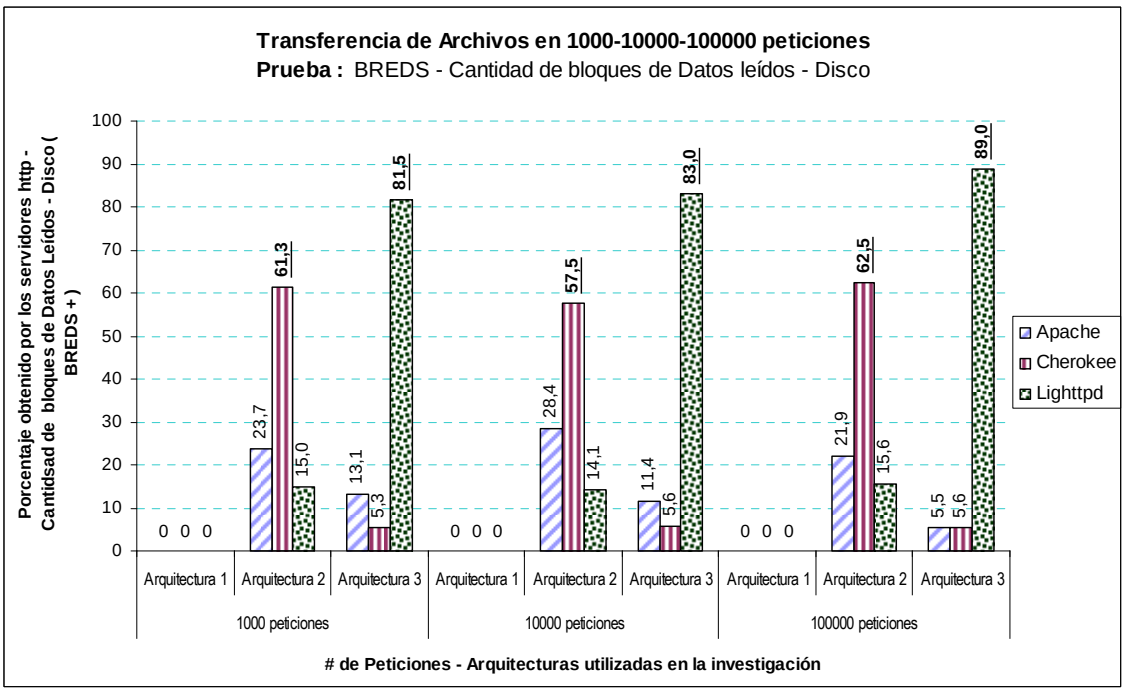


Ilustración IV- 34.- Transferencia de Archivos – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos – Disco

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Transferencia de Archivos - **BREDS – Cantidad de Bloques de Datos leídos** con un 83,0% a Apache y Cherokee.

Páginas Dinámicas (PHP) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 35.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) BREDS** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 62,6% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 91,2% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Cherokee supera con un 62,9% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 89,9% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** el servidor Cherokee supera con un 65,5% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 94,4% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http

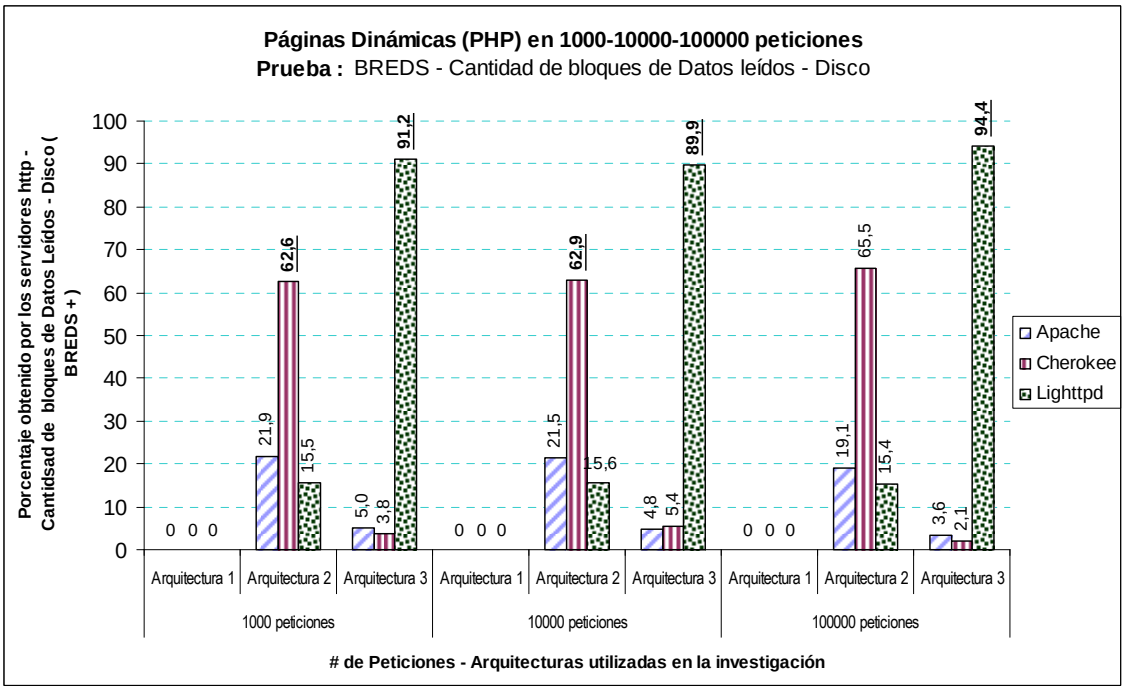


Ilustración IV- 35.- Páginas Dinámicas (PHP) – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP) - **BREDS – Cantidad de Bloques de Datos leídos** con un 89,9% a Apache y Cherokee.

Páginas Dinámicas (PHP + MySQL) – 1000 – 10000 - 100000 peticiones

En la **Ilustración IV- 36.- Transferencia de Archivos – 1000, 10000, 1000000 peticiones – Prueba: (+) BREDS** cuyos datos numéricos se los puede observar en el ANEXO 4, ANEXO 5 y ANEXO 6, se puede observar claramente que en **1000 peticiones** el servidor Cherokee supera con un 55,6% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 96,1% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **10000 peticiones,** el servidor Cherokee supera con un 57,5% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 96,4% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http.

En **1000000 peticiones,** el servidor Cherokee supera con un 67,3% a los servidores Apache y Lighttpd en la Arquitectura 2, Lighttpd supera con un 96,7% a Apache y Cherokee en la Arquitectura 3, mientras que en la Arquitectura 1 no existe diferencia significativa por tanto presentan un similar rendimiento los tres servidores http

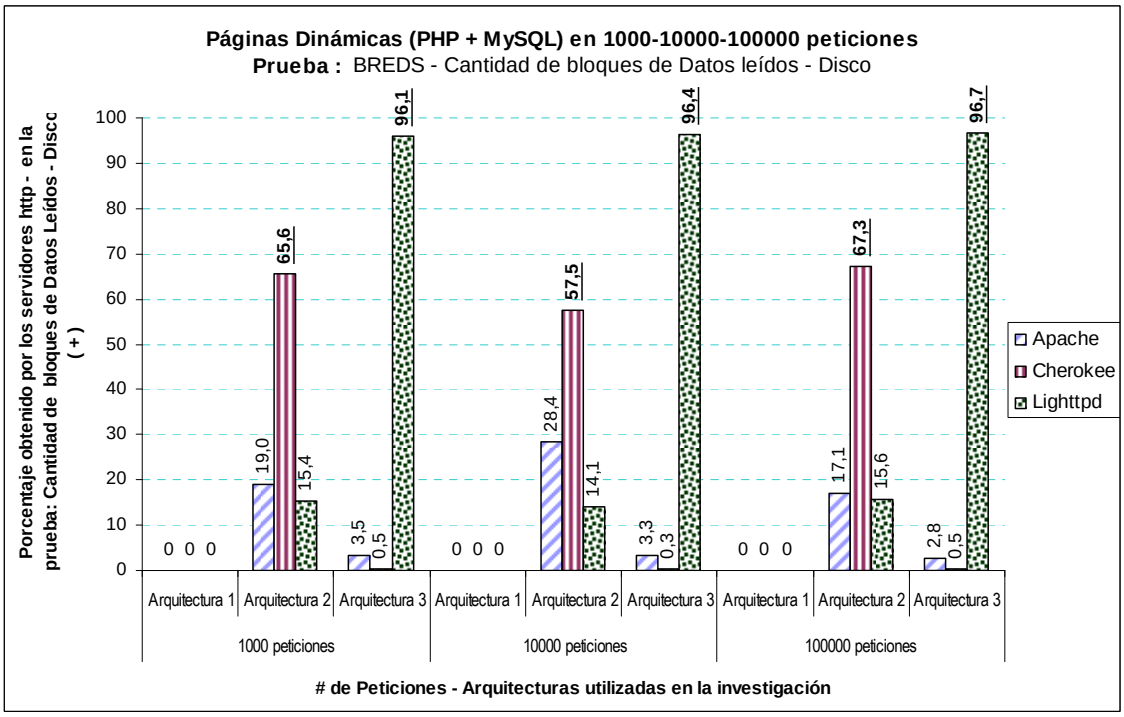


Ilustración IV- 36.- Páginas Dinámicas (PHP + MySQL) – 1000, 10000, 100000 peticiones – Prueba: (+) Cantidad de Bloques de Datos Leídos - Disco

Concluyendo en la Arquitectura 3 – 10000 peticiones el servidor **Lighttpd** supera la prueba Páginas Dinámicas (PHP + MySQL) - **BREDS – Cantidad de Bloques de Datos leídos** con un 96,4% a Apache y Cherokee.

4.4. Resultados de la Comparativa.

Los resultados de la comparativa se obtienen al tomar solo la Arquitectura 3 en 10000 peticiones en los tres servidores http Apache2, Cherokee y Lighttpd. Como se puede observar en la **Tabla IV-2.- Resultados de la Comparativa** se encuentran los datos resaltados con negrita de las pruebas a los que fueron sometidos los servidores http y en el servidor que supero en cada una de ellas.

	Arquitectura III		
	Apache 2	**Cherokee**	**Lighttpd**
Páginas Estáticas			
Requests per second (#/s)			
Time taken for tests (s)			
Transfer rate (Kbytes/s)			
Time per request (ms)			
USO DE CPU			
%user			
%system			
%idle			
USO DE MEMORIA			
kbmemused	95,683	97,53	**93,92**
USO DE DISCO			
bread/s	34,2	30,8	**81,3**
Transferencia de archivos			
Requests per second (#/s)			
Time taken for tests (s)			
Transfer rate (Kbytes/s)			
Time per request (ms)			
USO DE CPU			
%user			
%system	37,40	36,58	**23,94**
%idle			
USO DE MEMORIA			
kbmemused	96,497	97,52	**93,92**
USO DE DISCO			
bread/s	5,5	5,6	**89,0**
Páginas Dinámicas			

Requests per second (#/s)			
Time taken for tests (s)			
Transfer rate (Kbytes/s)			
Time per request (ms)			
USO DE CPU			
%user			
%system	50,8	42,893	**23,94**
%idle			
USO DE MEMORIA			
kbmemused	97,027	97,587	**93,92**
USO DE DISCO			
bread/s	4,8	5,4	**89,9**
Páginas Dinámicas + MySQL			
Requests per second (#/s)			
Time taken for tests (s)			
Transfer rate (Kbytes/s)			
Time per request (ms)			
USO DE CPU			
%user			
%system	39,87	32,64	**23,94**
%idle			
USO DE MEMORIA			
kbmemused	97,243	97,77	**93,92**
USO DE DISCO			
bread/s	3,3	0,3	**96,4**

Tabla IV - 2.- Resultado de la Comparativa
Fuente: Autoría Propia

El servidor que cumpla con las 36 pruebas obtendrá el 70%; valor que se le asigno a esta evaluación. Se aplicarán equivalencias de acuerdo al número de pruebas superadas por cada servidor, tomando como valor tope el 70%.

Como se puede apreciar casilleros vacíos en la **tabla IV-2.- Resultado de la Comparativa** significa que no se encontró diferencias significativas lo cual indica que los servidores Apache, Cherokee y Lighttpd funcionan de forma similar.

	Apache 2	Cherokee	Lighttpd
Páginas Estáticas			
Requests per second (#/s)	1	1	1
Time taken for tests (s)	1	1	1
Transfer rate (Kbytes/s)	1	1	1
Time per request (ms)	1	1	1
USO DE CPU			
%user	1	1	1
%system	1	1	1
%idle	1	1	1
USO DE MEMORIA			
kbmemused	0	0	**1**
USO DE DISCO			
bread/s	0	0	**1**
Transferencia de archivos			
Requests per second (#/s)	1	1	1
Time taken for tests (s)	1	1	1
Transfer rate (Kbytes/s)	1	1	1
Time per request (ms)	1	1	1
USO DE CPU			
%user	1	1	1
%system	0	0	**1**
%idle	**1**	**1**	1
USO DE MEMORIA			
kbmemused	0	0	**1**
USO DE DISCO			
bread/s	0	0	**1**
Paginas Dinámicas			
Requests per second (#/s)	1	1	1
Time taken for tests (s)	1	1	1
Transfer rate (Kbytes/s)	1	1	1
Time per request (ms)	1	1	1
USO DE CPU			
%user	1	1	1
%system	0	0	**1**
%idle	1	1	1
USO DE MEMORIA			
kbmemused	0	0	**1**
USO DE DISCO			
bread/s	0	0	**1**
Páginas Dinámicas + MySQL			
Requests per second (#/s)	1	1	1

Time taken for tests (s)	1	1	1
Transfer rate (Kbytes/s)	1	1	1
Time per request (ms)	1	1	1
USO DE CPU			
%user	1	1	1
%system	0	0	**1**
%idle	1	1	1
USO DE MEMORIA			
kbmemused	0	0	**1**
USO DE DISCO			
bread/s	0	0	**1**
Total # Pruebas acertadas	**24**	**24**	**36**

Tabla IV - 3.- Asignación de Pesos para obtener el servidor apropiado a nuestra necesidad
Fuente: Autoría Propia

Como se puede observar en la Tabla IV- 2.- Asignación de pesos para obtener el servidor adecuado a nuestra necesidad se les asigna un valor de 1 a los casilleros en blanco; ocurriendo esto en 24 pruebas del total de ésta parte de evaluación, también se le da el peso de 1 al servidor que cumple con los requerimientos de la prueba y 0 a los servidores que no.

Como en 24 pruebas los tres servidores presentan un funcionamiento similar su equivalente es 46,67%, y para obtener los otros equivalentes procedemos a realizar la siguiente regla de tres:

36 ➔ 70%

24 X = (24*70)/36 = 46,67%

Una vez calculada la equivalencia procedemos a realizar a sumar los valores que se obtuvieron en: Características del Servidor (30%), Rendimiento del Servidor y Recursos del Sistema (70%) resultado que puede llegar al 100% de la comparativa.

	Apache	**Cherokee**	**Lighttpd**
	%	**%**	**%**
Características del Servidor	30,00	30,00	30,00
Rendimiento del Servidor y Recursos del sistema	46,67	46,67	**70,00**
TOTAL	**76,67%**	**76,67%**	**100,00%**

Tabla IV - 4.- Resultado Final

Fuente: Autoría Propia

Siendo el servidor **Lighttpd** con un **100%** obtenido en la comparativa quien supera a Apache que alcanzó un 76,67% y Cherokee con un 76,67%.

Ahora pasaremos a la parte final de este capítulo en donde realizaremos las conclusiones de la comparativa obtenidas a lo largo de esta investigación.

4.5. Conclusiones de la comparativa.

Una vez analizado las características implementadas por cada servidor, el rendimiento del servidor y Recursos del sistema por cada uno llegamos a la conclusión que en aquellas variables medidas de las 36 pruebas en total, en donde existieron diferencias significativas en sus valores, Lighttpd supero en 12 de ellas a Apache, Cherokee y en las que no existió diferencias significativas los tres servidores presentan una similar funcionalidad totalizando Lighttpd el 70,00%, Apache 46,67% y Cherokee 46,67%.

Dado que el hardware de los servidores con los que cuenta la escuela de Ingeniería en Sistemas para la materia de Aplicaciones Web son Hp ProLaint ML37 G5; multiprocesador 1866Mhz con 2048MB en RAM y SCSI 136GB HP en Disco Duro, se pueden crear máquinas virtuales para prácticas en el manejo de servidores Web con características básicas 256MB en RAM y Disco Virtual de máximo 40GB ofreciendo de esta manera un amplio servicio sin problemas de procesos.

CONCLUSIONES

La presente investigación ha llevado a cabo un estudio estadístico del rendimiento de servidores web utilizando Apache, Cherokee y Lighttpd para determinar cuál de ellos utiliza menos recursos Hardware. De donde se obtiene las siguientes conclusiones:

- El estudio estadístico del rendimiento de Servidores Web, de libre distribución, aplicado en los servidores de la Escuela de Ingeniería en Sistemas – FIE (ESPOCH), contribuirá a optimizar la plataforma de hardware disponible.

- El análisis de las características de los servidores Web de distribución libre, y la utilización de pruebas creadas utilizando comandos propios de Linux permitió someter a Apache, Cherokee y Lighttpd a ensayos de Rendimiento y Uso de Recursos de Sistema.

- Las variables medidas en esta investigación son 36, de las cuales en 12 se presentaron diferencias significativas donde Lighttpd superó en todas a Apache y Cherokee, mientras que en las 24 pruebas restantes los tres servidores presentan un rendimiento similar, obteniendo Lighttpd un total de 70%, Apache 46,67% y Cherokee 46,67%.

- Lighttpd se muestra como una alternativa real, fiable, escalable, segura y veloz frente a otros servidores ya establecidos como Apache o Cherokee.

- Al utilizar Lighttpd nos garantizamos que una aplicación Web funcione más rápido sobre todo cuando existe cargas altas, recomendándose su uso en este tipo de situaciones frente a Apache por ser el más utilizado u otros servidores http.

- Apache2 es el gran servidor de esta comparativa, grande debido a que se encuentra mucho más extendido que el resto de servidores. Muy recomendado al igual que el ganador. Necesita más recursos del sistema para funcionar. Especialmente recomendado cuando se necesitan características adicionales que no implementan sus competidores.

- Cherokee es casi igual de eficiente que Lighttpd en cuanto a recursos del sistema. Muy rápido a la hora de servir páginas estáticas, aunque se debe mejorar el rendimiento en páginas dinámicas para ponerse al mismo nivel que Apache2 y Lighttpd.

RECOMENDACIONES

Las siguientes recomendaciones constituyen una recopilación de las principales actividades que se deben ejecutar para un correcto manejo del servidor Lighttpd:

- Se recomienda Instalar y configurar el Servidor Web Lighttpd, creando máquinas virtuales en los servidores de la Escuela de Ingeniería en Sistemas, para mejorar así el servicio que se proporciona a los estudiantes de la materia de Aplicaciones Web en sus prácticas de servidores Web.

- Continuar incrementando esta investigación analizando el funcionamiento de cada servidor http (Apache, Cherokee y Lighttpd) en función de la plataforma hardware, con el fin de poner a disposición del público mayor información.

- Dado que el hardware de los servidores con los que cuenta la escuela de Ingeniería en Sistemas para la materia de Aplicaciones Web son Hp ProLaint ML37 G5; multiprocesador 1866Mhz con 2048MB en RAM y SCSI 136GB HP en Disco Duro, es muy recomendable crear máquinas virtuales para prácticas en el manejo de servidores Web con características básicas 256MB en RAM y Disco Virtual de máximo 40GB ofreciendo de esta manera un amplio servicio sin problemas de procesos.

- Se recomienda utilizar Software Libre ya que es una posibilidad concreta y realizable, que nos abre una nueva ventana como ingenieros y profesionales de Educación Superior frente al Ecuador y al mundo. Por ley todas las personas tenemos derecho a acceder a la información y a los documentos públicos, hecho que se puede realizar gracias al software libre.

BIBLIOGRAFÍA

Libros

1.- **Bowen, R.;** López Ridruejo, D.; Liska, A. (2002). Apache Administrator's Handbook. 3a. ed. Canadá: SAS.

2.- **De B. Scott Burkett, y otros**. (2002). Guía Linux de Programación». 4a. ed. Canadá: O'Reilly.

3.- **Gibbs Mark**. (1997). Redes Para Todos. 1a. ed. México: Prentice Hall

4.- **Gundavaram, S.** (1996). CGI programming. 2a. ed.: EE.UU: O'Reilly.

5.- **Hernández Sampieri Roberto, Fernández Collado Carlos, Baptista Lucio Pilar**. (1996). Metodología de la investigación. 2a. ed. Colombia: Mc Graw Hill.

http://www.ispjae.cu/eventos/colaeiq/Cursos/Curso12.doc 20080326

6.- **López Cano José Luis.** (1984). Métodos e hipótesis científicas. 2da. ed. México: Prentice Hall

7.- **Matt Welsh, y otros.** (2000). Linux Installation and Getting Started**.** «Linux Instalación y Primeros Pasos». 3a. Ed. EE.UU.: O'Reilly.

8.- **Meyer, Eric A.** (2000). Cascading Style Sheets: The Definitive Guide. . 2a. ed. EE.UU: O'Reilly.

9.- **Rosenfeld, L.; Morville, P**. (1998). Information Architecture for the World Wide Web. 2a. ed. EE.UU.: O'Reilly.

10.- **Tackett Jack, Gunter David & Brown Lance.** (1996). Edición Especial Linux. 1a. ed. México: Prentice Hall.

12.- **Steel, RGD; Torrie, JH.** (1985). Bioestadística Principios y procedimientos. 1a. ed. México: Mcgraw-Hill.

13.- **G.E.P. Box, W.G. Hunter, J.S. Hunter**.(1989). Estadística para experimentadores. 1a. ed. Barcelona: Editorial Reverté.

Direcciones electrónicas

1.- **Apache Foundation (2003).** Apache HTTP Server Versión 2.0 Documentation. http://httpd.apache.org/docs-2.0/: Apache Foundation 20080323

2.- **Aplicaciones Web Espaciales con Software Libre** **http://mapa.buenosaires.gov.ar/sig/AplicacionesWebEspacialesCon SoftLibre.html** 20091803

3.- **Aplicaciones Multinivel** http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node21.html

20080320

4.- **Aplicaciones Web Espaciales con Software Libre**

http://mapa.buenosaires.gov.ar/sig/AplicacionesWebEspacialesConSoftLibre.html

20091803

5.- **Arquitectura Web**

http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node18.html

20080320

6.- **Benchmark**

http://es.wikipedia.org/wiki/Benchmark

20080425

7.- **Configurar Lighttpd**

http://trac.lighttpd.net/trac/wiki/TutorialConfiguration

20080425

8.- **Diccionario académico**

http://www.rae.es/NIVEL1/buscon/AUTORIDAD2.HTM

20080320

9.- **Diseño de Experimentos**

http://www.fundibeq.org/metodologias/herramientas/diseno_de_experimentos.pdf

20080425

10.- **Desarrollo de Aplicaciones Web**

http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node17.html

20080320

11.- **Ejemplos de Herramientas de Edición Web**

http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node22.html

20091803

12.- **Estadísticas de Uso**

http://www.irontec.com/~aktor/dokuwiki/doku.php?id=wiki:apache-

roadmap&s=recomendad

20091803

13.- **El Navegador Web, Browser**

http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node19.html

20080320

14.- **El protocolo HTTP**

http://www.cibernetia.com/manuales/introduccion_aplicaciones_web/2_1_

fundamentos_web.php

20080320

15.- **El Servidor HTTP**

http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node20.html

20080320

16.- **Fundamentos de la web**

http://www.cibernetia.com/manuales/introduccion_aplicaciones_web/2_1_fundamentos_web.php

20080320

17.- **Glosario Básico Inglés-Español para usuarios de Internet**

http://www.ati.es/novatica/glosario/glosario_internet.txt

20080320

18.- **Glosario de Términos**

http://www.uco.es/webuco/si/ccc/glosario/glosario.html

20080320

19.- **Informática Forense - Análisis de Apache y WebERP**

http://labs.dragonjar.org/laboratorio-informatica-forense-analisis-de-apache-y-weberp

20080319

20.- **La gran Red PHP**

http://php.apsique.com/contenido/mejorar_rendimiento_tu_sitio_php_parte_i_hardware_servidores_http_y_bases_datos

20080320

21.- **La Web de Wikipedia, la enciclopedia libre**

http://es.wikipedia.org/wiki/World_Wide_Web

20091803

22.- **Lighttpd**

http://www.lighttpd.net/download/lighttpd-1.5.0-r1992.tar.gz

20080320

23.- **Linux**

http://www.linuxparatodos.com

20080425

24.- **Linux Benchmarking CÓMO**

http://es.tldp.org/COMO-INSFLUG/es/pdf/Benchmarking-COMO.pdf

20080315

25.- **Manual de Desarrollo Web**

www.desarrolloweb.com/manuales/41/

20080322

26.- **Manual de Squid.**

www2.idesoft.com/squid/manual.htm

20080312

27.- **Mejorar rendimiento tu sitio php parte I hardware, servidores http y base de datos**

http://php.apsique.com/contenido/mejorar_rendimiento_tu_sitio_php_parte

_i_hardware_servidores_http_y_bases_datos

20080318

28.- **Open source firewall**
http://www.opensourcefirewall.com
20080318

29.- **php español**
http://www.phpnuke-espanol.org
20080430

30.- **¿Qué es un servidor? - Definición de servidor**

http://www.masadelante.com/faq-servidor.htm

20080418

31.- **¿Que es un benchmark?**

http://www.monografias.com/trabajos/benchmark/benchmark.shtml

20080425

32.- **Requerimientos de hardware para servidor**.

http://www.mail-archive.com/linux-plug@linux.org.pe/msg00553.html

20080325

33**.-** **Servidor Apache**

www.htmlpoint.com/apache/

20080321

34.- **Servidores**

http://www.w3.org/People/Raggett/book4/ch02.html

20080322

35.- **Servidores Web**

http://www.w3.org/Consortium/

20080321

36.- **Servidores Web - Características**

http://www.ciens.ucv.ve/ciencias/servidores_web.htm

20080325

37.- **Servidores Web**

http://es.wikipedia.org/wiki/Servidor_web

20080415

38.- **Statistical methods in psychology journals: Guidelines and Explanations.**
http://www.apa.org/journals/amp/amp548594.htm
20080425

39.- **Vulnerabilidad en el servidor Apache**

http://www.dragonjar.org/vulnerabilidad-en-el-servidor-apache-20.xhtml

20080315

40.- **Midiendo el rendimiento del sistema con SAR**

http://mundogeek.net/traducciones/midiendo-el-rendimiento-del-sistema-con-SAR.htm

20080315

RESUMEN

El presente trabajo investigativo tuvo como objetivo seleccionar mediante un estudio Estadístico entre **Apache, Cherokee y Lighttpd** de distribución libre**,** el servidor más adecuado para optimizar el hardware existente en la Escuela de Ingeniería en Sistemas - ESPOCH y demostrar que su utilización permite mejorar las prácticas en el manejo de servidores Web a los estudiantes de la materia de Aplicaciones Web.

La investigación estuvo basada en el Método Experimental más una planeación de recursos y actividades, el Estudio Estadístico consistió en: pruebas de Rendimiento y Uso de Recursos del Sistema a cada servidor utilizando equipos Intel Pentium 4 1600 MHz (16 x 100); 256 MB en RAM y 40GB 5400 RPM Ultra-ATA/100 en disco duro, y un Análisis de Resultados.

Como resultados se obtuvieron: Los tres servidores cumplen con lo requerido en características implementadas, mientras que en el Rendimiento del Servidor y Uso de Recursos del Sistema; Lighttpd totalizó un valor de **70%,** Apache **46,67%** y Cherokee con **46,67%**, determinándose con estos resultados el más adecuado para los servidores de la institución.

Se concluye: el servidor **Lighttpd** es superior a Apache y Cherokee, en uso de CPU, Memoria y Disco, ya que estos aspectos optimizan el hardware, recomendándose su uso para crear Máquinas Virtuales con características básicas (256MB en RAM y Disco de 40GB) ofreciendo así más equipos computacionales para prácticas en el manejo de servidores Web.

SUMMARY

The present investigative work had as objective to select by means of a Statistical study between **Apache, Cherokee and Lighttpd** of free distribution, the servant more adapted to optimize the existing hardware in the Engineering in Systems School - ESPOCH and to demonstrate that its use allows to improve the practices in the handling of Web servers to the students of the matter of Applications Web.

The investigation was based on the Experimental Method plus a planning of resources and activities, the Statistical Study consisted of: tests of Yield and Use of Resources of the System to each servant using equipment Intel Pentium 4 1600 MHz (16 x 100); 256 MB RAM and 40GB 5400 RPM Ultra-ATA/100 in hard disk, and an Analysis of Results.

As results were obtained: The three servants fulfill the required thing in implemented characteristics, whereas in the Yield of the Servant and Use of Resources of the System; Lighttpd totalized a value of **70,00%**, Apache **46,67%** and Cherokee with **46,67%,** determining itself with these results the most adapted for the servants of the institution.

One concludes: the **Lighttpd** servant is superior to Apache and Cherokee, in use of CPU, Memory and Disc, since these aspects optimize the hardware, recommending themselves their use to create Virtual machines with basic characteristics (256MB RAM and Disc of 40GB) thus offering more computational equipment for practices in the handling of Web servers.

ANEXOS

ANEXO 1

El programa de pruebas y cargas de trabajo, llamada comparativa es un pequeño programa desarrollado en shell que realiza cuatro pruebas distintas.

Apache

```
echo "Test de Apache 2 Web Server"

rm apache2/apache_log.txt
rm apache2/apache_test_1_1000.txt
rm apache2/apache_test_1_10000.txt
rm apache2/apache_test_1_100000.txt
rm apache2/apache_test_2_1000.txt
rm apache2/apache_test_2_10000.txt
rm apache2/apache_test_2_100000.txt
rm apache2/apache_test_3_1000.txt
rm apache2/apache_test_3_10000.txt
rm apache2/apache_test_3_100000.txt
rm apache2/apache_test_4_1000.txt
rm apache2/apache_test_4_10000.txt
rm apache2/apache_test_4_100000.txt

rm apache2/apache_sar_test1_1000.txt
rm apache2/apache_sar_all_test1_1000.txt
rm apache2/apache_sar_test1_10000.txt
rm apache2/apache_sar_all_test1_10000.txt
rm apache2/apache_sar_test1_100000.txt
rm apache2/apache_sar_all_test1_100000.txt

rm apache2/apache_sar_test2_1000.txt
rm apache2/apache_sar_all_test2_1000.txt
rm apache2/apache_sar_test2_10000.txt
rm apache2/apache_sar_all_test2_10000.txt
rm apache2/apache_sar_test2_100000.txt
rm apache2/apache_sar_all_test2_100000.txt

rm apache2/apache_sar_test3_1000.txt
rm apache2/apache_sar_all_test3_1000.txt
rm apache2/apache_sar_test3_10000.txt
rm apache2/apache_sar_all_test3_10000.txt
rm apache2/apache_sar_test3_100000.txt
rm apache2/apache_sar_all_test3_100000.txt

rm apache2/apache_sar_test4_1000.txt
rm apache2/apache_sar_all_test4_1000.txt
rm apache2/apache_sar_test4_10000.txt
rm apache2/apache_sar_all_test4_10000.txt
rm pache2/apache_sar_test4_100000.txt
rm apache2/apache_sar_all_test4_100000.txt

echo "Benchmark del Servidor Apache 2 Web Server"
>>apache2/apache_log.txt
echo "----------------------------------------"
>>apache2/apache_log.txt
echo "Test de Apache 2 Web Server" >>apache2/apache_log.txt
echo "Fecha y hora de comienzo del test">>apache2/apache_log.txt
date >>apache2/apache_log.txt
```

##Test 1

```
echo " ">>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor en paginas
estaticas">>apache2/apache_log.txt
echo "Realizo una prueba de 1000 peticiones (10 concurrentes) de un
documento html">>apache2/apache_log.txt
echo ""
echo "Fecha y hora del comienzo de esta prueba:
">>apache2/apache_log.txt
date >>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor en paginas estaticas"

#Realizo una llamada a sar para que mida el rendimiento antes de
comenzar
/usr/lib/sa/sa1 1 1

#Ahora realizo la prueba con Apache Benchmark
ab -n 1000 -c 10 http://localhost/index.html
>>apache2/apache_test_1_1000.txt

#Realizo una llamada a sar para que mida el rendimiento despues de
finalizar
/usr/lib/sa/sa1 1 1
echo "Fecha y hora del fin de esta prueba: ">>apache2/apache_log.txt
date >>apache2/apache_log.txt
sar >> apache2/apache_sar_test1_1000.txt
sar -A >> apache2/apache_sar_all_test1_1000.txt
```

##Test 2

```
echo " ">>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor sirviendo
archivos">>apache2/apache_log.txt
echo "Realizo una prueba de 1000 peticiones (10 concurrentes) de un
documento tar de 1MB">>apache2/apache_log.txt
echo ""
echo "Fecha y hora del comienzo de esta
prueba">>apache2/apache_log.txt
date >>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor sirviendo archivos"

#Realizo una llamada a sar para que mida el rendimiento antes de
comenzar
/usr/lib/sa/sa1 1 1

#Ahora realizo el test con Apache Benchmark
ab -n 1000 -k -c 10 http://localhost/datos.tar.gz
>>apache2/apache_test_2_1000.txt

#Realizo una llamada a sar para que mida el rendimiento despuÃ©s de
finalizar
/usr/lib/sa/sa1 1 1
echo "Fecha y hora del fin de esta prueba">>apache2/apache_log.txt
date >>apache2/apache_log.txt
sar >> apache2/apache_sar_test2_1000.txt
sar -A >> apache2/apache_sar_all_test2_1000.txt
```

##Test 3

```
echo " ">>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor procesando PHP"
>>apache2/apache_log.txt
echo "Realizo una prueba de 1000 peticiones (10 concurrentes) de un
documento php" >>apache2/apache_log.txt
echo ""
echo "Fecha y hora del comienzo de esta prueba"
>>apache2/apache_log.txt
date >>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor procesando PHP"

#Realizo una llamada a sar para que mida el rendimiento antes de
comenzar
/usr/lib/sa/sa1 1 1

#Ahora realizo el test con Apache Benchmark
ab -n 1000 -c 10 http://localhost/phpinfo.php
>>apache2/apache_test_3_1000.txt

#Realizo una llamada a sar para que mida el rendimiento despuÃ©s de
finalizar
/usr/lib/sa/sa1 1 1
echo "Fecha y hora del fin de esta prueba">>apache2/apache_log.txt
date >>apache2/apache_log.txt
sar >> apache2/apache_sar_test3_1000.txt
sar -A >> apache2/apache_sar_all_test3_1000.txt
```

##Test 4

```
echo " ">>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor procesando PHP y
MySql">>apache2/apache_log.txt
echo "Realizo una prueba de 1000 peticiones (10 concurrentes) de un
documento php con consultas a una base de datos
SQL">>apache2/apache_log.txt
echo ""
echo "Fecha y hora del comienzo de esta
prueba">>apache2/apache_log.txt
date >>apache2/apache_log.txt
echo "Midiendo el rendimiento del servidor procesando PHP y MySql"

#Realizo una llamada a sar para que mida el rendimiento antes de
comenzar
/usr/lib/sa/sa1 1 1

#Ahora realizo el test con Apache Benchmark
ab -n 1000 -c 10 http://localhost/phpinfo.php
>>apache2/apache_test_4_1000.txt

#Realizo una llamada a sar para que mida el rendimiento despuÃ©s de
finalizar
/usr/lib/sa/sa1 1 1
echo "Fecha y hora del fin de esta prueba">>apache2/apache_log.txt
date >>apache2/apache_log.txt
sar >> apache2/apache_sar_test4_1000.txt
sar -A >> apache2/apache_sar_all_test4_1000.txt
```

El programa que se usa finalmente para realizar cada prueba es Apache Benchmark también de la Fundación Apache, en su versión 2.

Los resultados de cada prueba se almacenan en archivos de texto, uno por cada prueba. Para cada servidor se almacena un fichero de log, con detalles como la hora de inicio y fin de cada prueba.

ANEXO 2

Comando SAR. **Midiendo el rendimiento del sistema con SAR**

El comando sar nos permite monitorizar un sistema en Linux, proporcionándonos el uso de la cpu, memoria, discos...

El nombre sar proviene de las siglas de "system activity report" (informe de la actividad del sistema). En Linux se encuentra normalmente en el paquete sysstat. El paquete sysstat incluye programas y scripts para recopilar y mostrar información sobre el rendimiento del sistema, generando informes detallados. Este conjunto de programas puede resultar de mucha utilidad a la hora de detectar cuellos de botella y para hacernos una idea de cómo se utiliza el sistema a lo largo del día

Recopilar datos sobre el rendimiento

El programa que se ocupa de recopilar la información se llama sadc (system activity data collector o recolector de datos de la actividad del sistema). Obtiene su información, principalmente, del kernel, a través del sistema de archivos virtual en /proc. Después guarda los datos en un fichero (uno por día con nombre /var/log/sa/saDD donde DD es el día del mes.

El paquete incluye dos shell scripts. El primer script, sa1, recopila datos de forma regular, mientras que el script sa2 se utiliza para crear los informes resumidos (uno por día en /var/log/sa/sarDD. Ambos scripts se ejecutan usando cron. En Red Hat Enterprise estos scripts se añaden automáticamente a las tareas de cron de la siguiente forma:

```
# ejecuta la herramienta de recopilación de datos cada 10 minutos
*/10 * * * * root /usr/lib/sa/sa1 1 1
# genera un informe diario del rendimiento de los procesos a las 23:53
53 23 * * * root /usr/lib/sa/sa2 –A
```

Con la configuración por defecto los datos se recopilan cada 10 minutos y se genera un informe justo antes de la media noche. Si se sospecha que puede existir un problema de rendimiento con un programa en particular, hay que utilizar sadc para recopilar datos sobre ese proceso (-x), o sus hijos (-X), pero se necesita crear su propio script que use esos flags.

Como demostró el Dr. Heisenberg, el simple hecho de realizar una medida cambia los datos medidos. Cualquier herramienta dedicada a recopilar datos sobre rendimiento tiene un cierto impacto negativo en el rendimiento del sistema, pero con sar, este parece ser mínimo. Se realizó una prueba en la que cron lanzaba el programa sa1 para obtener datos cada minuto (en un servidor que no estaba demasiado ocupado) y no causó ningún problema serio. Lo cual puede no resultar cierto en un sistema con mucha carga.

Creando informes

Si los informes diarios creados por el script sa2 no son suficientes, se puedes crear los propios utilizando sar. Por defecto este programa obtiene la información del archivo de datos del día actual, a menos que se especifique lo contrario. Para hacer que sar obtenga los datos de un archivo en particular, utilizamos el flag -f /var/log/sa/saDD. Se puede seleccionar múltiples archivos usando varios flags -f. Dado que muchos de los informes que crea sar tienen un tamaño considerable, puede que se quiera redirigir la salida a un archivo.

Para crear un informe básico que muestre el uso de CPU y el porcentaje de tiempo gastado esperando E/S, ejecutamos sar sin ningún argumento. Generará un informe similar al siguiente:

```
01:10:00 PM       CPU     %user     %nice   %system   %iowait     %idle
01:20:00 PM       all      7.78      0.00      3.34     20.94     67.94
01:30:00 PM       all      0.75      0.00      0.46      1.71     97.08
01:40:00 PM       all      0.65      0.00      0.48      1.63     97.23
01:50:00 PM       all      0.96      0.00      0.74      2.10     96.19
02:00:00 PM       all      0.58      0.00      0.54      1.87     97.01
02:10:00 PM       all      0.80      0.00      0.60      1.27     97.33
02:20:01 PM       all      0.52      0.00      0.37      1.17     97.94
02:30:00 PM       all      0.49      0.00      0.27      1.18     98.06
Average:          all      1.85      0.00      0.44      2.56     95.14
```

Si %idle (desocupado) está cerca de cero, tu CPU está sobrecargada. Si el valor %iowait (espera por e/s) es grande, sus discos están sobrecargados.

Para comprobar cómo se comporta el fichero de paginación ejecuta sar -B para obtener un informe similar al siguiente:

```
11:00:00 AM  pgpgin/s pgpgout/s   fault/s  majflt/s
11:10:00 AM      8.90     34.08      0.00      0.00
11:20:00 AM      2.65     26.63      0.00      0.00
11:30:00 AM      1.91     34.92      0.00      0.00
11:40:01 AM      0.26     36.78      0.00      0.00
11:50:00 AM      0.53     32.94      0.00      0.00
```

```
12:00:00 PM      0.17     30.70      0.00      0.00
12:10:00 PM      1.22     27.89      0.00      0.00
12:20:00 PM      4.11    133.48      0.00      0.00
12:30:00 PM      0.41     31.31      0.00      0.00
Average:       130.91     27.04      0.00      0.00
```

Puede que el número de veces que se ha tenido que recurrir a la paginación no sea determinante, pero el que exista un alto número de fallos mayores de página (majflt/s) indica que el sistema necesita más memoria. (nota: majflt/s sólo es válido para versiones del kernel >= 2.5).

Para obtener estadísticas de red, utilizamos sar -n DEV. Este comando genera un informe que muestra estadísticas con los datos transmitidos y recibidos para cada interfaz de red. Veamos una versión abreviada de este informe:

```
11:00:00 AM     IFACE   rxpck/s   txpck/s   rxbyt/s   txbyt/s
11:10:00 AM        lo      0.62      0.62     35.03     35.03
11:10:00 AM      eth0     29.16     36.71   4159.66  34309.79
11:10:00 AM      eth1      0.00      0.00      0.00      0.00
11:20:00 AM        lo      0.29      0.29     15.85     15.85
11:20:00 AM      eth0     25.52     32.08   3535.10  29638.15
11:20:00 AM      eth1      0.00      0.00      0.00      0.00
```

Para ver los errores de red, prueba con el comando sar -n EDEV.

Informes en tiempo real

También podemos utilizar sar para ver que está pasando en este preciso momento con un componente del sistema. Si se le pasa un intervalo de tiempo (en segundos) y el número de informes a producir, podemos obtener información inmediata sobre un posible cuello de botella.

Por ejemplo, para ver el informe básico cada segundo durante los próximos 10 segundos, usaríamos sar 1 10. Cualquiera de las combinaciones de flags anteriores se puede ejecutar con estos parámetros para obtener resultados cercanos al tiempo real.

Pruebas de rendimiento

Incluso si contamos con potencia de sobra para correr nuestras aplicaciones, sar siempre puede ser útil para registrar cambios en el volumen de trabajo a lo largo del tiempo. Para hacer esto, guardaremos los informes (sar sólo guarda siete) en un directorio diferente durante un periodo de tiempo de unas pocas semanas o un mes.

Este grupo de informes pueden servir como ejemplo del volumen de trabajo normal del sistema. Se pueden comparar los siguientes informes con estos resultados previos para ver cómo cambia el volumen de trabajo con el tiempo. Podemos incluso automatizar estos informes usando awk o nuestro lenguaje de programación favorito.

En la gestión de grandes sistemas, las pruebas de rendimiento son importantes para predecir qué hardware actualizar y cuándo hacerlo. También nos proporciona datos con los que justificar estas actualizaciones.

Profundizando

La mayor parte de los problemas de rendimiento de hardware están relacionados con los discos, la memoria, o la CPU. Otras veces errores al programar las aplicaciones o bases de datos mal diseñadas pueden producir problemas de rendimiento realmente serios. De cualquier forma, sar y sus amigos nos pueden dar una visión detallada de cómo se comporta nuestro sistema y ayudarnos a detectar y arreglar problemas. Los ejemplos anteriores no hacen más que arañar la superficie de lo que se puede hacer con sar. Se pueden crear informes adicionales muy detallados con los programas iostat y mpstat incluidos en el mismo paquete. Si echas un vistazo a la documentación debería ser bastante sencillo crear un conjunto de informes adaptados a tus necesidades.

ANEXO 3 - Arquitecturas Empleadas en la comparativa

ARQUITECTURA 1

Ordenador No.1: I6

--------[EVEREST Professional (c) 2003, 2004 Lavalys, Inc.]----------

Versión	**EVEREST v1.51.195/es**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**InvesPC6**
Generador	**linvestigacion**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-29**
Hora	**15:10**

--------[Resumen]----------

Ordenador :

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 2
Internet Explorer	6.0.2900.2180 (IE 6.0 SP2)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	**InvesPC6**
Nombre de usuario	linvestigacion
Nombre de dominio	**InvesPC6**
Fecha / Hora	2008-08-29 / 15:10

Placa base :

Tipo de procesador	Unknown, 2133 MHz (16 x 133)
Nombre de la Placa Base	Intel Corporation DG965SS
Chipset de la Placa Base	Desconocido
Memoria del Sistema	1005 MB
Tipo de BIOS	Insyde (07/16/06)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor :

Tarjeta gráfica	Intel(R) G965 Express Chipset Family (128 MB)

Almacenamiento :

Controlador IDE	Controladora estándar PCI IDE de doble canal
Controlador IDE	Controladora estándar PCI IDE de doble canal
Controlador IDE	Controladora estándar PCI IDE de doble canal
Controlador SCSI/RAID	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	ST3160811AS (149 GB , IDE)
Disco duro	GENERIC USB DISK DEVICE USB Device (1945 MB , USB)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Estado de los discos duros SMART	OK

Particiones :

C: (NTFS)	39997 MB (24537 MB libre)
E: (FAT32)	42615 MB (37448 MB libre)

Dispositivos de entrada :

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red :

Dirección IP principal	172.30.104.68
Dirección MAC principal	00-19-D1-02-59-2C
Tarjeta de Red	Intel(R) 82566DC Gigabit Network Connection (172.30.104.68)

Dispositivos :

Impresora	Microsoft Office Document Image Writer
Impresora	Microsoft XPS Document Writer
Controlador FireWire	Texas Instruments TSB43AB22 1394A-2000 OHCI PHY/Link-Layer Controller
Dispositivos USB	Dispositivo de almacenamiento masivo USB

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MQ96510J.86A.0816.2006.0716.2308
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	DA672BB6-5ADB11DB-931500E0-18811F39
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	DG965SS
DMI Versión de la Placa Base	AAD41678-304
DMI Número de serie de la Placa Base	BQSS64100J4G
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

Ordenador No.2: MPROF

--------[EVEREST Professional (c) 2003, 2004 Lavalys, Inc.]----------

Versión	**EVEREST v1.51.195/es**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**MULTIPROF**
Generador	**LMultimedia**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-29**
Hora	**15:25**

--------[Resumen]----------

Ordenador :

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 3
Internet Explorer	7.0.5730.13
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	MULTIPROF
Nombre de usuario	LMultimedia
Nombre de dominio	MULTIPROF
Fecha / Hora	2008-08-29 / 15:25

Placa base :

Tipo de procesador	Unknown, 2133 MHz (16 x 133)
Nombre de la Placa Base	Intel Corporation DG965SS
Chipset de la Placa Base	Desconocido
Memoria del Sistema	1005 MB
Tipo de BIOS	Insyde (07/16/06)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor :

Tarjeta gráfica	Intel(R) G965 Express Chipset Family (128 MB)

Multimedia :

Tarjeta de sonido	SigmaTel Audio

Almacenamiento :

Controlador IDE	Controladora estándar PCI IDE de doble canal
Controlador IDE	Intel(R) ICH8 2 port Serial ATA Storage Controller - 2825
Controlador IDE	Intel(R) ICH8 4 port Serial ATA Storage Controller - 2820
Controlador SCSI/RAID	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	ST3160811AS (149 GB , IDE)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	HL-DT-ST CD-RW GCE-8527B
Estado de los discos duros SMART	OK

Particiones :

C: (NTFS)	45002 MB (30115 MB libre)
D: (NTFS)	35000 MB (34869 MB libre)
E: (FAT32)	34983 MB (31436 MB libre)

Dispositivos de entrada :

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red :

Dirección IP principal	172.30.104.58
Dirección MAC principal	00-16-76-C8-6C-28
Tarjeta de Red	Intel(R) 82566DC Gigabit Network Connection (172.30.104.58)

Dispositivos :

Impresora	Adobe PDF
Controlador FireWire	Texas Instruments TSB43AB22 1394A-2000 OHCI PHY/Link-Layer Controller
Controlador USB1	Intel(R) ICH8 Family USB Universal Host Controller - 2830 [NoDB]
Controlador USB1	Intel(R) ICH8 Family USB Universal Host Controller - 2831 [NoDB]
Controlador USB1	Intel(R) ICH8 Family USB Universal Host Controller - 2832 [NoDB]
Controlador USB1	Intel(R) ICH8 Family USB Universal Host Controller - 2834 [NoDB]
Controlador USB1	Intel(R) ICH8 Family USB Universal Host Controller - 2835 [NoDB]
Controlador USB2	Intel(R) ICH8 Family USB2 Enhanced Host Controller - 2836 [NoDB]
Controlador USB2	Intel(R) ICH8 Family USB2 Enhanced Host Controller - 283A [NoDB]

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MQ96510J.86A.0816.2006.0716.2308
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	D3ACC8D0-3C0911DB-932600E0-188FF8D8
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	DG965SS
DMI Versión de la Placa Base	AAD41678-304
DMI Número de serie de la Placa Base	BQSS6360017T
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

Ordenador No.3: M8

--------[EVEREST Ultimate Edition]----------

Versión	**EVEREST v4.50.1330/es**
Benchmark Module	**2.3.224.0**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**MULTIPC8**
Generador	**LMultimedia**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-29**
Hora	**14:55**

--------[Resumen]----------

Ordenador:

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 2
Internet Explorer	7.0.5730.13 (IE 7.0)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	MULTIPC8
Nombre de usuario	LMultimedia
Nombre de dominio	MULTIPC8
Fecha / Hora	2008-08-29 / 14:55

Placa base:

Tipo de procesador	DualCore Intel Core 2 Duo E6400, 2133 MHz (8 x 267)
Nombre de la Placa Base	Intel Shrewsbury DG965SS (2 PCI, 1 PCI-E x1, 1 PCI-E x16, 4 DDR2 DIMM, Audio, Video, Gigabit LAN, IEEE-1394)
Chipset de la Placa Base	Intel Broadwater G965
Memoria del Sistema	1005 MB
DIMM1:	Corsair Value Select VS1GB667D2 1 GB DDR2-667 DDR2 SDRAM (5-5-5-15 @ 333 MHz) (4-4-4-12 @ 266 MHz) (3-3-3-9 @ 200 MHz)
Tipo de BIOS	Intel (07/16/06)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor:

Tarjeta gráfica	Intel GMA X3000
Acelerador 3D	Intel GMA X3000

Multimedia:

Tarjeta de sonido	SigmaTel STAC9227X @ Intel 82801HB ICH8 - High Definition Audio Controller [B-0]

Almacenamiento:

Controlador IDE	Controladora estándar PCI IDE de doble canal
Controlador IDE	Intel(R) ICH8 2 port Serial ATA Storage Controller - 2825
Controlador IDE	Intel(R) ICH8 4 port Serial ATA Storage Controller - 2820
Storage Controller	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	GENERIC USB DISK DEVICE USB Device (1945 MB, USB)
Disco duro	ST3160811AS (160 GB, 7200 RPM, SATA-II)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	Unidad de CD-ROM
Estado de los discos duros SMART	OK

Particiones:

C: (NTFS)	41998 MB (30788 MB libre)
D: (NTFS)	37001 MB (28360 MB libre)
E: (FAT32)	34992 MB (31444 MB libre)
Tamaño total	111.3 GB (88.5 GB libre)

Dispositivos de entrada:

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red:

Dirección IP principal	172.30.104.43
Dirección MAC principal	00-19-D1-02-58-E6
Tarjeta de Red	Intel(R) 82566DC Gigabit Network Connection (172.30.104.43)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet1 (192.168.199.1)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet8 (192.168.245.1)

Dispositivos:

Controlador FireWire	Texas Instruments TSB43AB22 1394A-2000 OHCI PHY/Link-Layer Controller (PHY: TI TSB41AB1/2)
Controlador USB1	Intel 82801HB ICH8 - USB Universal Host Controller [B-0]
Controlador USB1	Intel 82801HB ICH8 - USB Universal Host Controller [B-0]
Controlador USB1	Intel 82801HB ICH8 - USB Universal Host Controller [B-0]
Controlador USB1	Intel 82801HB ICH8 - USB Universal Host Controller [B-0]
Controlador USB1	Intel 82801HB ICH8 - USB Universal Host Controller [B-0]
Controlador USB2	Intel 82801HB ICH8 - USB2 Enhanced Host Controller [B-0]
Controlador USB2	Intel 82801HB ICH8 - USB2 Enhanced Host Controller [B-0]
Dispositivos USB	Dispositivo de almacenamiento masivo USB

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MQ96510J.86A.0816.2006.0716.2308
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	0F624E00-5ADB11DB-931500E0-18811F39
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	DG965SS
DMI Versión de la Placa Base	AAD41678-304
DMI Número de serie de la Placa Base	BQSS64100HZH
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

ARQUITECTURA 2

Ordenador No.1: M7

--------[EVEREST Ultimate Edition]----------

Versión	**EVEREST v4.50.1330/es**
Benchmark Module	**2.3.224.0**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**MULTIPC7**
Generador	**LMultimedia**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-28**
Hora	**16:15**

--------[Resumen]----------

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 3
Internet Explorer	7.0.5730.13 (IE 7.0)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	**MULTIPC7**
Nombre de usuario	LMultimedia
Nombre de dominio	MULTIPC7
Fecha / Hora	2008-08-28 / 16:15

Placa base:

Tipo de procesador	Intel Pentium 4 530J, 3000 MHz (15 x 200)
Nombre de la Placa Base	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video)
Chipset de la Placa Base	Intel Grantsdale-G i915GV
Memoria del Sistema	501 MB (PC3200 DDR SDRAM)
DIMM1:	Samsung M3 68L3223ETM-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
DIMM3:	Samsung M3 68L3223FTN-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
Tipo de BIOS	Intel (02/16/05)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor:

Tarjeta gráfica	Intel GMA 900
Acelerador 3D	Intel GMA 900

Multimedia:

Tarjeta de sonido	Realtek ALC880(D) @ Intel 82801FB ICH6 - High Definition Audio Controller [B-2]

Almacenamiento:

Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB - 2651
Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB/FBM - 266F
Disquetera de 3 1/2	Unidad de disquete
Disco duro	SAMSUNG SP0802N (80 GB, 7200 RPM, Ultra-ATA/133)
Lector óptico	HL-DT-ST CD-ROM GCR-8525B (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones:

C: (NTFS)	29996 MB (20802 MB libre)
D: (FAT32)	19992 MB (19992 MB libre)
E: (FAT32)	19992 MB (18731 MB libre)
Tamaño total	68.3 GB (58.1 GB libre)

Dispositivos de entrada:

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red:

Dirección IP principal	172.30.104.72
Dirección MAC principal	00-0A-5E-59-C5-37
Tarjeta de Red	NIC PCI 3Com EtherLink XL 10/100 PCI para administración completa del equipo (3C905C-TX) (172.30.104.72)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet1 (192.168.255.1)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet8 (192.168.126.1)

Dispositivos:

Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB2	Intel 82801FB ICH6 - Enhanced USB2 Controller [B-2]

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	WB91X10J.86A.1037.2005.0216.1050
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	7583BDDA-DC7011D9-A680000E-A68F73D0
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D915GVWB
DMI Versión de la Placa Base	AAC82205-104
DMI Número de serie de la Placa Base	FCWB52510963
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

Ordenador No.2: M10

--------[EVEREST Ultimate Edition]----------

Versión	**EVEREST v4.50.1330/es**
Benchmark Module	**2.3.224.0**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**MULTIPC10**
Generador	**LMultimedia**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-28**
Hora	**16:15**

--------[Resumen]----------

Ordenador:

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 3
Internet Explorer	7.0.5730.13 (IE 7.0)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	**MULTIPC10**
Nombre de usuario	LMultimedia
Nombre de dominio	MULTIPC10
Fecha / Hora	2008-08-28 / 16:15

Placa base:

Tipo de procesador	Intel Pentium 4 530J, 3000 MHz (15 x 200)
Nombre de la Placa Base	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video)
Chipset de la Placa Base	Intel Grantsdale-G i915GV
Memoria del Sistema	501 MB (PC3200 DDR SDRAM)
DIMM1:	Samsung M3 68L3223ETM-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
DIMM3:	Samsung M3 68L3223ETM-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
Tipo de BIOS	Intel (02/16/05)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor:

Tarjeta gráfica	Intel GMA 900
Acelerador 3D	Intel GMA 900

Multimedia:

Tarjeta de sonido	Realtek ALC880(D) @ Intel 82801FB ICH6 - High Definition Audio Controller [B-2]

Almacenamiento:

Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB - 2651
Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB/FBM - 266F
Storage Controller	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	SAMSUNG SP0802N (80 GB, 7200 RPM, Ultra-ATA/133)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	HL-DT-ST CD-ROM GCR-8525B (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones:

C: (NTFS)	29996 MB (20797 MB libre)
D: (FAT32)	19992 MB (19992 MB libre)
E: (FAT32)	19992 MB (18103 MB libre)
Tamaño total	68.3 GB (57.5 GB libre)

Dispositivos de entrada:

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red:

Dirección IP principal	172.30.104.24
Dirección MAC principal	00-0A-5E-59-CA-0D
Tarjeta de Red	NIC PCI 3Com EtherLink XL 10/100 PCI para administración completa del equipo (3C905C-TX) (172.30.104.24)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet1 (192.168.255.1)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet8 (192.168.126.1)

Dispositivos:

Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB2	Intel 82801FB ICH6 - Enhanced USB2 Controller [B-2]

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	WB91X10J.86A.1037.2005.0216.1050
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	6124EAEC-DC9511D9-AE5D000E-A6292A1D
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D915GVWB
DMI Versión de la Placa Base	AAC82205-104

DMI Número de serie de la Placa Base	FCWB52512142
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

Ordenador No.3: M20

--------[EVEREST Ultimate Edition]----------

Versión	**EVEREST v4.50.1330/es**
Benchmark Module	**2.3.224.0**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**MULTIPC20**
Generador	**LMultimedia**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2008-08-29**
Hora	**14:59**

--------[Resumen]----------

Ordenador:

Tipo de ordenador	Equipo multiprocesador ACPI
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 3
Internet Explorer	7.0.5730.13 (IE 7.0)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	**MULTIPC20**
Nombre de usuario	LMultimedia
Nombre de dominio	MULTIPC20
Fecha / Hora	2008-08-29 / 14:59

Placa base:

Tipo de procesador	Intel Pentium 4 530J, 3000 MHz (15 x 200)
Nombre de la Placa Base	Intel West Branch D915GVWB (2 PCI, 1 PCI-E x1, 4 DDR DIMM, Audio, Video)
Chipset de la Placa Base	Intel Grantsdale-G i915GV
Memoria del Sistema	501 MB (PC3200 DDR SDRAM)
DIMM1:	Samsung M3 68L3223FTN-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
DIMM3:	Samsung M3 68L3223FTN-CCC 256 MB PC3200 DDR SDRAM (3.0-3-3-8 @ 200 MHz) (2.5-3-3-7 @ 166 MHz)
Tipo de BIOS	Intel (02/16/05)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de impresora ECP (LPT1)

Monitor:

Tarjeta gráfica	Intel GMA 900
Acelerador 3D	Intel GMA 900

Multimedia:

Tarjeta de sonido	Realtek ALC880(D) @ Intel 82801FB ICH6 - High Definition Audio Controller [B-2]

Almacenamiento:

Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB - 2651
Controlador IDE	Controladoras de almacenamiento Ultra ATA Intel(R) 82801FB/FBM - 266F
Disquetera de 3 1/2	Unidad de disquete
Disco duro	SAMSUNG SP0802N (80 GB, 7200 RPM, Ultra-ATA/133)
Lector óptico	HL-DT-ST CD-ROM GCR-8525B (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones:

C: (NTFS)	29996 MB (20851 MB libre)
D: (FAT32)	19992 MB (19849 MB libre)
E: (FAT32)	19992 MB (11623 MB libre)
Tamaño total	68.3 GB (51.1 GB libre)

Dispositivos de entrada:

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red:

Dirección IP principal	172.30.104.42
Dirección MAC principal	00-0A-5E-59-D9-CC
Tarjeta de Red	NIC PCI 3Com EtherLink XL 10/100 PCI para administración completa del equipo (3C905C-TX) (172.30.104.42)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet1 (192.168.152.1)
Tarjeta de Red	VMware Virtual Ethernet Adapter for VMnet8 (192.168.172.1)

Dispositivos:

Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB1	Intel 82801FB ICH6 - USB Universal Host Controller [B-2]
Controlador USB2	Intel 82801FB ICH6 - Enhanced USB2 Controller [B-2]

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	WB91X10J.86A.1037.2005.0216.1050
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	2D2F281E-DC6E11D9-AE5D000E-A6292A1D
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D915GVWB
DMI Versión de la Placa Base	AAC82205-104
DMI Número de serie de la Placa Base	FCWB52511629
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	

ARQUITECTURA 3

Ordenador No.1: P16

--------[EVEREST Professional (c) 2003, 2004 Lavalys, Inc.]----------

Versión	**EVEREST v1.51.195/es**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**progPC16**
Generador	**lprogramacion**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2007-12-17**
Hora	**14:23**

--------[Resumen]----------

Ordenador :

Tipo de ordenador	Monoprocesador ACPI de PC
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 2
Internet Explorer	6.0.2900.2180 (IE 6.0 SP2)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	progPC16
Nombre de usuario	lprogramacion
Nombre de dominio	progPC16
Fecha / Hora	2007-12-17 / 14:23

Placa base :

Tipo de procesador	Intel Pentium 4, 1600 MHz (16 x 100)
Nombre de la Placa Base	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio)
Chipset de la Placa Base	Intel Tehama i850
Memoria del Sistema	512 MB (RDRAM)
Tipo de BIOS	AMI (03/28/02)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de comunicaciones (COM2)
Puerto de comunicación	Puerto de impresora (LPT1)

Monitor :

Tarjeta gráfica	Trident Video Accelerator 9440 (1 MB)

Multimedia :

Tarjeta de sonido	Intel 82801BA(M) ICH2 - AC'97 Audio Controller [B-4]

Almacenamiento :

Controlador IDE	Controladora de almacenamiento Ultra ATA Intel(R) 82801BA - 244B
Controlador SCSI/RAID	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	MAXTOR 4K040H2 (40 GB , 5400 RPM, Ultra-ATA/100)
Disco duro	USB 2.0 Flash Disk USB Device (117 MB , USB)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	SAMSUNG CD-ROM SC-152L (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones :

C: (NTFS)	38177 MB (30276 MB libre)

Dispositivos de entrada :

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red :

Dirección IP principal	127.0.0.1
Dirección MAC principal	00-00-00-00-00-00

Dispositivos :

Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC11
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC12
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC7
Impresora	Microsoft Office Document Image Writer
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 1 [B-4]
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 2 [B-4]
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Dispositivo de almacenamiento masivo USB

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MV85010A.86A.0025.P10.0203282158
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	54C5C5AF-6C0211D6-ACE90010-83B8DC13
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D850MV
DMI Versión de la Placa Base	AAA56420-500
DMI Número de serie de la Placa Base	IUMV22100822
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	
DMI Sockets de memoria (Total/Libres)	4 /0

Ordenador No.2: P17

--------[EVEREST Professional (c) 2003, 2004 Lavalys, Inc.]----------

Versión	**EVEREST v1.51.195/es**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**
Ordenador	**progPC17**
Generador	**lprogramacion**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2007-12-17**
Hora	**14:23**

--------[Resumen]----------

Ordenador :

Tipo de ordenador	Monoprocesador ACPI de PC
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 2
Internet Explorer	6.0.2900.2180 (IE 6.0 SP2)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	progPC17
Nombre de usuario	lprogramacion
Nombre de dominio	progPC11
Fecha / Hora	2007-12-17 / 14:23

Placa base :

Tipo de procesador	Intel Pentium 4, 1600 MHz (16 x 100)
Nombre de la Placa Base	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio)
Chipset de la Placa Base	Intel Tehama i850
Memoria del Sistema	512 MB (RDRAM)
Tipo de BIOS	AMI (03/28/02)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de comunicaciones (COM2)

Puerto de comunicación	Puerto de impresora (LPT1)

Monitor :

Tarjeta gráfica	Trident Video Accelerator 9440 (1 MB)

Multimedia :

Tarjeta de sonido	Intel 82801BA(M) ICH2 - AC'97 Audio Controller [B-4]

Almacenamiento :

Controlador IDE	Controladora de almacenamiento Ultra ATA Intel(R) 82801BA - 244B
Controlador SCSI/RAID	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	MAXTOR 4K040H2 (40 GB , 5400 RPM, Ultra-ATA/100)
Disco duro	USB 2.0 Flash Disk USB Device (117 MB , USB)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	SAMSUNG CD-ROM SC-152L (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones :

C: (NTFS)	38177 MB (30276 MB libre)

Dispositivos de entrada :

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red :

Dirección IP principal	127.0.0.1
Dirección MAC principal	00-00-00-00-00-00

Dispositivos :

Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC11
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC12
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC7
Impresora	Microsoft Office Document Image Writer
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 1 [B-4]
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 2 [B-4]
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Dispositivo de almacenamiento masivo USB

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MV85010A.86A.0025.P10.0203282158
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	54C5C5AF-6C0211D6-ACE90010-83B8DC13
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D850MV
DMI Versión de la Placa Base	AAA56420-500
DMI Número de serie de la Placa Base	IUMV22100822
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	
DMI Sockets de memoria (Total/Libres)	4 /0

Ordenador No.3: P5

--------[EVEREST Professional (c) 2003, 2004 Lavalys, Inc.]----------

Versión	**EVEREST v1.51.195/es**
Sitio Web	**http://www.lavalys.com/**
Tipo de informe	**Asistente de informes**

Ordenador	**progPC5**
Generador	**lprogramacion**
Sistema operativo	**Microsoft Windows XP Professional 5.1.2600 (WinXP Retail)**
Fecha	**2007-12-17**
Hora	**14:23**

--------[Resumen]----------

Ordenador :

Tipo de ordenador	Monoprocesador ACPI de PC
Sistema operativo	Microsoft Windows XP Professional
Service Pack del Sistema Operativo	Service Pack 2
Internet Explorer	6.0.2900.2180 (IE 6.0 SP2)
DirectX	4.09.00.0904 (DirectX 9.0c)
Nombre del sistema	progPC5
Nombre de usuario	lprogramacion
Nombre de dominio	progPC5
Fecha / Hora	2007-12-17 / 14:23

Placa base :

Tipo de procesador	Intel Pentium 4, 1600 MHz (16 x 100)
Nombre de la Placa Base	Intel Maryville D850MV (5 PCI, 1 AGP, 4 RIMM, Audio)
Chipset de la Placa Base	Intel Tehama i850
Memoria del Sistema	512 MB (RDRAM)
Tipo de BIOS	AMI (03/28/02)
Puerto de comunicación	Puerto de comunicaciones (COM1)
Puerto de comunicación	Puerto de comunicaciones (COM2)
Puerto de comunicación	Puerto de impresora (LPT1)

Monitor :

Tarjeta gráfica	Trident Video Accelerator 9440 (1 MB)

Multimedia :

Tarjeta de sonido	Intel 82801BA(M) ICH2 - AC'97 Audio Controller [B-4]

Almacenamiento :

Controlador IDE	Controladora de almacenamiento Ultra ATA Intel(R) 82801BA - 244B
Controlador SCSI/RAID	A347SCSI SCSI Controller
Disquetera de 3 1/2	Unidad de disquete
Disco duro	MAXTOR 4K040H2 (40 GB , 5400 RPM, Ultra-ATA/100)
Disco duro	USB 2.0 Flash Disk USB Device (117 MB , USB)
Lector óptico	AXV CD/DVD-ROM SCSI CdRom Device (Virtual DVD-ROM)
Lector óptico	SAMSUNG CD-ROM SC-152L (52x CD-ROM)
Estado de los discos duros SMART	OK

Particiones :

C: (NTFS)	38177 MB (30276 MB libre)

Dispositivos de entrada :

Teclado	Teclado estándar de 101/102 teclas o Microsoft Natural PS/2 Keyboard
Ratón	Mouse PS/2 de Microsoft

Red :

Dirección IP principal	127.0.0.1
Dirección MAC principal	00-00-00-00-00-00

Dispositivos :

Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC11
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC12
Impresora	Detectando automáticamente Microsoft Office Document Image Writer en progPC7
Impresora	Microsoft Office Document Image Writer
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 1 [B-4]
Controlador USB1	Intel 82801BA(M) ICH2 - USB Controller 2 [B-4]
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Concentrador USB genérico
Dispositivos USB	Dispositivo de almacenamiento masivo USB

DMI:

DMI Distribuidor de la BIOS	Intel Corp.
DMI Versión de la BIOS	MV85010A.86A.0025.P10.0203282158
DMI Fabricante del Sistema	
DMI Nombre del Sistema	
DMI Versión del sistema	
DMI Número de serie del Sistema	
DMI UUID del Sistema	54C5C5AF-6C0211D6-ACE90010-83B8DC13
DMI Fabricante de la Placa Base	Intel Corporation
DMI Nombre de la Placa Base	D850MV
DMI Versión de la Placa Base	AAA56420-500
DMI Número de serie de la Placa Base	IUMV22100822
DMI Fabricante del chasis	
DMI Versión del chasis	
DMI Número de serie del chasis	
DMI Identificador del chasis	
DMI Tipo de chasis	
DMI Sockets de memoria (Total/Libres)	4 /0

ANEXO 4 – DATOS DE LAS MEDIAS DE LOS SERVIDORES WEB OBTENIDOS DEL SPSS VER 15.0 – 1000 peticiones

Páginas Estáticas – 1000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS ESTATICAS	**Apache**	**Cherokee**	**Lighttpd**	**PAGINAS ESTATICAS**	**Apache**	**Cherokee**	**Lighttpd**	**PAGINAS ESTATICAS**	**Apache**	**Cherokee**	**Lighttpd**
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	**REQUEST PER SECOND (PETICIONES POR SEGUNDO)**										
1000 peticiones				1000 peticiones	19233,937	4869,853	7122,903				
-	**TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)**										
1000 peticiones				1000 peticiones	0,3018	0,2379	0,4332				
+	**TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA**										
1000 peticiones				1000 peticiones	41474,567	37873,913	81794,157				
-	**TIME PER REQUEST (ms) - (TIEMPO POR PETICION)**										
1000 peticiones				1000 peticiones	3,018	2,379	1,62				
-	**%CPU USER (Uso de CPU)**										
1000 peticiones				1000 peticiones	1,7	18,09	16,72	1000 peticiones			
-	**%CPU por sistema y Kernel**										
1000 peticiones				1000 peticiones				1000 peticiones			
+	**%CPU IDLE (USO CPU DISPONIBLE)**										
1000 peticiones											
-	**KBMEMUSED - %MEMORIA RAM UTILIZADA**										
1000 peticiones				1000 peticiones	96,587	96,69	96,553	1000 peticiones			
+	**BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO**										
1000 peticiones				1000 peticiones	151,457	274,523	67,167	1000 peticiones	133,58	26,527	379,05

Tabla 2.- Páginas Estáticas – 1000 peticiones

FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Transferencia de Archivos – 1000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
TRANSFERENCIA DE ARCHIVOS	Apache	Cherokee	Lighttpd	TRANSFERENCIA DE ARCHIVOS	Apache	Cherokee	Lighttpd	SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$				
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
1000 peticiones				1000 peticiones	590,887	1480,755	560,911	1000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
1000 peticiones				1000 peticiones	1,854	1,168	1,481	1000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
1000 peticiones				1000 peticiones	591604,343	458817,637	753619,933	1000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
				1000 peticiones	18,54	11,677	14,81	1000 peticiones			
-	%CPU USER (Uso de CPU)										
1000 peticiones				1000 peticiones				1000 peticiones			
-	%CPU por sistema y Kernel										
1000 peticiones				1000 peticiones	3,883	11,347	8,843	1000 peticiones	36,577	38,13	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
1000 peticiones				1000 peticiones				1000 peticiones			
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
1000 peticiones				1000 peticiones	96,95	96,767	96,463	1000 peticiones	96,403	97,547	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
1000 peticiones				1000 peticiones	105,61	273,423	67,113	1000 peticiones	61,033	24,843	379,05

Tabla 1.- Transferencia de Archivos – 1000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas (PHP) – 1000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
1000 peticiones				1000 peticiones	613,743	649,087	725,44	1000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
1000 peticiones				1000 peticiones	67,318	1,874	1,521	1000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
1000 peticiones				1000 peticiones	30252,367	23711,833	36648,323	1000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
1000 peticiones				1000 peticiones	18,228	18,736	15,205	1000 peticiones			
-	%CPU USER (Uso de CPU)										
1000 peticiones				1000 peticiones				1000 peticiones			
-	%CPU por sistema y Kernel										
1000 peticiones				1000 peticiones	14,767	12,823	12,033	1000 peticiones	50,113	39,597	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
1000 peticiones				1000 peticiones	78,557	64,517	70,263	1000 peticiones	5,5	1,417	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
1000 peticiones				1000 peticiones	97,16	96,85	96,587	1000 peticiones	96,947	97,62	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
1000 peticiones				1000 peticiones	93,833	268,317	66,633	1000 peticiones	20,603	15,75	379,05

Tabla 2.- Páginas Dinámicas (PHP) – 1000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas con acceso a Base de Datos (PHP+MySQL) – 1000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
1000 peticiones	939,777	1178,197	1129,057	1000 peticiones	614,607	1123,673	746,047	1000 peticiones	141,863	62,04	133,72
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
1000 peticiones				1000 peticiones	129,585	1,187	1,487				
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
1000 peticiones	47130,173	51988,72	58344,887	1000 peticiones	30387,473	24706,23	37661,237				
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
1000 peticiones				1000 peticiones	18,109	11,871	14,869	1000 peticiones	73,619	161,461	79,553
-	%CPU USER (Uso de CPU)										
1000 peticiones				1000 peticiones	16,5	19,097	19,823	1000 peticiones	56,49	67,237	50,86
-	%CPU por sistema y Kernel										
1000 peticiones				1000 peticiones	12,927	13,35	12,793	1000 peticiones	39,62	32,46	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
1000 peticiones				1000 peticiones	69,053	62,647	65,687	1000 peticiones	3,723	0	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
1000 peticiones				1000 peticiones	96,95	96,897	96,697	1000 peticiones	97,23	97,7	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
1000 peticiones				1000 peticiones	76,217	262,467	61,683	1000 peticiones	13,673	1,827	379,05

Tabla 3.- Páginas Dinámicas con acceso a Base de Datos (PHP+MySQL) – 1000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

ANEXO 5 – DATOS DE LAS MEDIAS DE LOS SERVIDORES WEB OBTENIDOS DEL SPSS VER 15.0 – 10000 peticiones

Páginas Estáticas – 10000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd	PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd	PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
10000 peticiones	6714,403	5290,767	15001,907	10000 peticiones	4517,08	5727,98	9483,023	10000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
10000 peticiones	6714,403	5290,767	15001,907	10000 peticiones	4517,08	5727,98	9483,023	10000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
10000 peticiones	76832,78	85442,717	174971,287	10000 peticiones	51907,163	45820,58	108699,647	10000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
10000 peticiones				10000 peticiones	2,209	1,974	1,223	10000 peticiones			
-	%CPU USER (Uso de CPU)										
10000 peticiones				10000 peticiones				10000 peticiones			
-	%CPU por sistema y Kernel										
10000 peticiones				10000 peticiones				10000 peticiones			
+	%CPU IDLE (USO CPU DISPONIBLE)										
10000 peticiones				10000 peticiones				10000 peticiones			
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
10000 peticiones				10000 peticiones	96,71	96,71	96,457	10000 peticiones	95,683	97,53	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
10000 peticiones				10000 peticiones	150,587	274,583	67,213	10000 peticiones	119,047	26,75	379,05

Tabla 4.- Páginas Estáticas – 10000 peticiones

FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Transferencia de Archivos – 10000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd	SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd	SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
10000 peticiones	1019,057	1053,937	1415,153	10000 peticiones	612,827	2131,497	851,943	10000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
10000 peticiones				10000 peticiones	11,464	239,219	261,359	10000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
10000 peticiones				10000 peticiones	99528,347	78215,2	138503,36	10000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
10000 peticiones	9,875	9,493	7,135	10000 peticiones	17,764	11,161	14,076	10000 peticiones			
-	%CPU USER (Uso de CPU)										
10000 peticiones				10000 peticiones	32,097	32,587	29,553	10000 peticiones			
-	%CPU por sistema y Kernel										
10000 peticiones				10000 peticiones	5,57	11,483	9,157	10000 peticiones	37,397	36,583	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
10000 peticiones				10000 peticiones				10000 peticiones			
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
10000 peticiones				10000 peticiones	97,077	96,8	96,49	10000 peticiones	96,497	97,52	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
10000 peticiones				10000 peticiones	134,963	272,853	67,03	10000 peticiones	51,867	25,537	379,05

Tabla 5.-Transferencia de Archivos – 10000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas (PHP) – 10000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
10000 peticiones	1017,783	1255,97	1171,63	10000 peticiones	642,557	1269,653	758,19	10000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
10000 peticiones				10000 peticiones	17,498	11,198	15,185	10000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
10000 peticiones	51028,56	55402,76	60513,503	10000 peticiones	31681,987	26222,313	38201,63	10000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
10000 peticiones	9,83	7,963	8,539	10000 peticiones	17,498	11,198	15,185	10000 peticiones			
-	%CPU USER (Uso de CPU)										
10000 peticiones				10000 peticiones	6,353	17,78	16,33	10000 peticiones			
-	%CPU por sistema y Kernel										
10000 peticiones				10000 peticiones	14,52	12,873	12,103	10000 peticiones	50,8	42,893	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
10000 peticiones				10000 peticiones	77,443	64,34	69,827	10000 peticiones	5,603	2,08	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
10000 peticiones				10000 peticiones	97,237	96,897	96,647	10000 peticiones	97,027	97,587	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
10000 peticiones				10000 peticiones	91,65	267,773	66,33	10000 peticiones	20,077	22,593	379,05

Tabla 6.- Páginas Dinámicas (PHP) – 10000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas con acceso a Base de Datos (PHP+MySQL) - 10000

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
10000 peticiones	991,607	1264,633	1186,217	10000 peticiones	612,827	2131,497	851,943	10000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
10000 peticiones				10000 peticiones	11,464	239,219	261,359	10000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
10000 peticiones	49704,987	55784,883	61232,247	10000 peticiones	99528,347	78215,2	138503,36	10000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
1000 peticiones	10,647	8,489	8,865	10000 peticiones	17,764	11,161	14,076	10000 peticiones	91,136	165,512	77,685
-	%CPU USER (Uso de CPU)										
10000 peticiones				10000 peticiones	32,097	32,587	29,553	10000 peticiones	56,197	67,043	50,86
-	%CPU por sistema y Kernel										
10000 peticiones				10000 peticiones	5,57	11,483	9,157	10000 peticiones	39,87	32,643	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
10000 peticiones				10000 peticiones				10000 peticiones	5,603	2,08	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
10000 peticiones				10000 peticiones	97,077	96,8	96,49	10000 peticiones	97,243	97,77	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
10000 peticiones				10000 peticiones	134,963	272,853	67,03	10000 peticiones	13,133	1,16	379,05

Tabla 7.- Páginas Dinámicas con Acceso a Base de Datos (PHP+MySQL) – 10000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

ANEXO 6 – DATOS DE LAS MEDIAS DE LOS SERVIDORES WEB OBTENIDOS DEL SPSS VER 15.0 – 100000 peticiones

Páginas Estáticas – 100000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd	PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd	PAGINAS ESTATICAS	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
100000 peticiones	7077,743	7486,163	16639,283	100000 peticiones	4612,367	5491,207	10004,427	100000 peticiones	750,057	494,863	1707,12
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
100000 peticiones				100000 peticiones	2279,978	2593,983	5413,915	100000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
100000 peticiones	81328,307	85682,02	190710,16	100000 peticiones	43353,99	41067,02	117700,333	100000 peticiones	8619,247	5664,18	20656,71
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
100000 peticiones	1,182	1,336	0,601	100000 peticiones	950,38	2,294	1,25	100000 peticiones	15,527	20,3	5,94
-	%CPU USER (Uso de CPU)										
100000 peticiones				100000 peticiones				100000 peticiones			
-	%CPU por sistema y Kernel										
100000 peticiones				100000 peticiones				100000 peticiones	32,463	35,34	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
100000 peticiones				100000 peticiones				100000 peticiones			
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
100000 peticiones				100000 peticiones	96,823	96,737	96,46	100000 peticiones	96,047	97,53	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
100000 peticiones				100000 peticiones	22,64	20,427	23,183	100000 peticiones	60,983	26,273	379,05

Tabla 8.- Páginas Estáticas – 100000 peticiones

FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Transferencia de Archivos – 100000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd	SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd	SIRVIENDO ARCHIVOS	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
100000 peticiones	1011,347	1006,623	1402,49	100000 peticiones	619,427	2275,667	863,033	100000 peticiones			
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
100000 peticiones				100000 peticiones	1,989	130,858	96,275	100000 peticiones			
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
100000 peticiones	99,675	99,343	71,938	100000 peticiones	156032,313	157718,903	187174,593	100000 peticiones			
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
100000 peticiones	4,984	9,934	7,194	100000 peticiones	14,14	11,146	13,945	100000 peticiones			
-	%CPU USER (Uso de CPU)										
100000 peticiones				100000 peticiones				100000 peticiones			
-	%CPU por sistema y Kernel										
100000 peticiones				100000 peticiones	14,797	12,817	12,027	100000 peticiones	49,453	40,703	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
100000 peticiones				100000 peticiones	78,69	64,543	70,307	100000 peticiones	6,763	2,25	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
100000 peticiones				100000 peticiones	97,073	96,82	96,543	100000 peticiones	96,713	97,553	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
100000 peticiones				100000 peticiones	93,907	267,58	66,627	100000 peticiones	23,22	23,75	379,05

Tabla 9.- Transferencia de Archivos – 100000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas (PHP) – 100000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
				100000 peticiones	648,55	1330,93	756,077	100000 peticiones	135,97	71,213	170,79
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
100000 peticiones	99,619	77,561	84,331	100000 peticiones	82,065	18,067	91,276				
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
100000 peticiones	8214,763	2796,02	11493,233	100000 peticiones	11546,26	8929,023	14077,033				
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
100000 peticiones				100000 peticiones	17,405	11,262	15,142	100000 peticiones	78,958	143,159	60,142
-	%CPU USER (Uso de CPU)										
100000 peticiones				100000 peticiones	76,337	262,51	61,677	100000 peticiones			
-	%CPU por sistema y Kernel										
100000 peticiones				100000 peticiones	12,937	13,343	12,79	100000 peticiones	43,01	36,087	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
100000 peticiones				100000 peticiones	69,123	62,667	65,723	100000 peticiones	4,277	0,69	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
100000 peticiones				100000 peticiones	97,1	96,893	96,673	100000 peticiones	97,163	97,65	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
100000 peticiones				100000 peticiones	76,337	262,51	61,677	100000 peticiones	14,317	8,25	379,05

Tabla 10.- Páginas Dinámicas – 100000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

Páginas Dinámicas con acceso a Base de Datos (PHP+MySQL) – 100000 peticiones

ARQUITECTURA 1				ARQUITECTURA 2				ARQUITECTURA 3			
PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd	PAGINAS DINAMICAS - PHP+MYSQL	Apache	Cherokee	Lighttpd
	$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$		$\overline{X}$	$\overline{X}$	$\overline{X}$
+	REQUEST PER SECOND (PETICIONES POR SEGUNDO)										
100000 peticiones	996,827	1288,54	1189,523	100000 peticiones	643,513	1327,34	748,433				
-	TIME TAKEN FOR TESTS (s) (TIEMPO EMPLEADO POR CADA PRUEBA)										
100000 peticiones	100,361	77,607	84,154	100000 peticiones	82,065	18,067	91,276				
+	TRANSFER RATE (Kb/s) - TASA DE TRANSFERENCIA										
100000 peticiones	8155,48	2794,343	11512,297	100000 peticiones	11546,26	8929,023	14077,033	100000 peticiones	2458,8	1004,167	2958,44
-	TIME PER REQUEST (ms) - (TIEMPO POR PETICION)										
10000 peticiones	10,089	7,907	8,439	100000 peticiones	17,405	11,262	15,142	100000 peticiones	89,325	159,619	80,819
-	%CPU USER (Uso de CPU)										
100000 peticiones				100000 peticiones	22,64	20,427	23,183	100000 peticiones	60,767	67,273	50,86
-	%CPU por sistema y Kernel										
100000 peticiones				100000 peticiones	11,77	13,84	13,463	100000 peticiones	35,97	32,413	23,94
+	%CPU IDLE (USO CPU DISPONIBLE)										
100000 peticiones				100000 peticiones	64,2	60,93	61,72	100000 peticiones	4,277	0,69	0,2
-	KBMEMUSED - %MEMORIA RAM UTILIZADA										
100000 peticiones				100000 peticiones	96,73	96,94	96,75	100000 peticiones	97,267	97,763	93,92
+	BREDS - CANTIDAD DE BLOQUES DE DATOS LEIDOS - DISCO										
100000 peticiones				100000 peticiones	65,34	257,23	59,683	100000 peticiones	11,04	1,907	379,05

Tabla 11.- Páginas Dinámicas con Acceso a Base de Datos (PHP+MySQL) – 100000 peticiones
FUENTE: Datos obtenidos al ser procesados en el programa SPSS 15.0

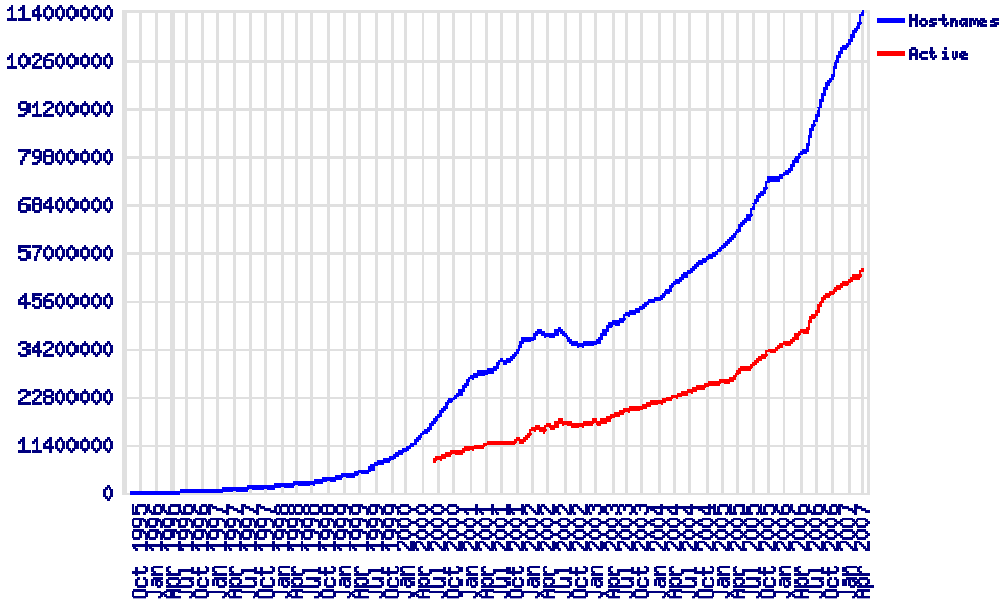
ANEXO 7 – ESTADISTICAS USO DE SERVIDORES HTTP

Abril 2007 servidor Web encuesta

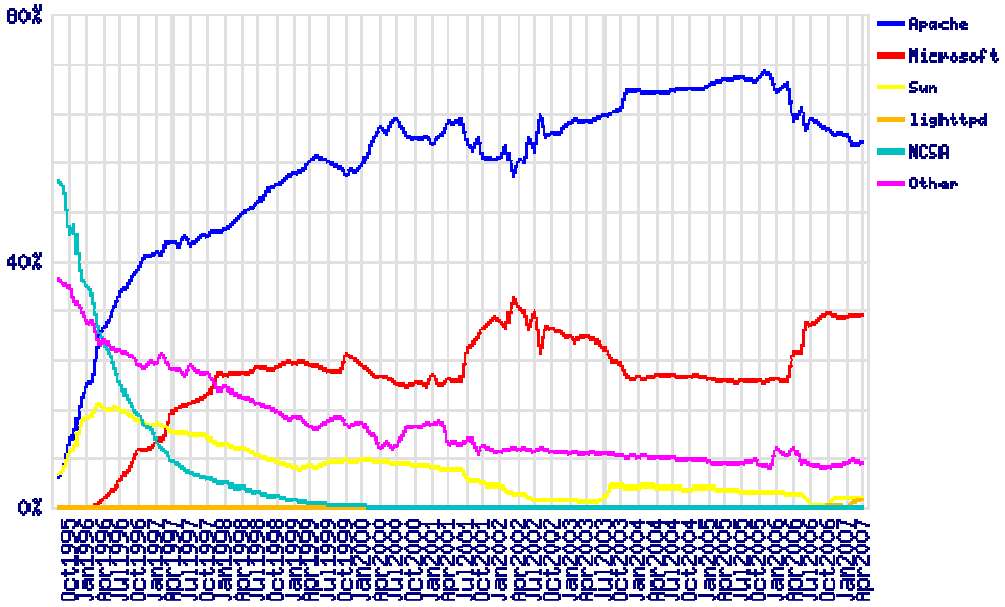
En **abril** la **encuesta de 2007** que hemos recibido las respuestas de los sitios **113658468,** un aumento de 3,2 millones de sitios el mes pasado la encuesta. Apache continúa siendo el más ampliamente utilizado servidor web, la alimentación de más de 66,9 millones de sitios, en comparación con 35,3 millones de sitios usando software de servidor de Microsoft.

This month the Web Server Survey adds public tracking of lighttpd , an open source server designed for high-performance sites that has been gaining popularity in recent months.Lighttpd is currently detected on 1.38 million sites for a 1.2% share of the web server market, well ahead of Zeus and moving up quickly on Sun.Lighttpd has a relatively small memory footprint and is optimized for a large number of parallel connections, which has made it popular on sites using applications based on AJAX or Ruby on Rails, or hosting environments for virtual private servers. Sitios populares lighttpd utilizando como su principal servidor de incluir el SourceForge comunidad de desarrolladores, los medios de comunicación social el centro de Reddit y el servicio de mensajería instantánea Meebo, mientras que YouTube y Wikipedia utilizarlo para servir partes de sus sitios.

Total de sitios en todos los dominios de agosto de 1995 - abril 2007



Cuota de mercado de los servidores Comienzo de la página en todos los dominios de agosto de 1995 - abril 2007



Comienzo de la página de Desarrolladores

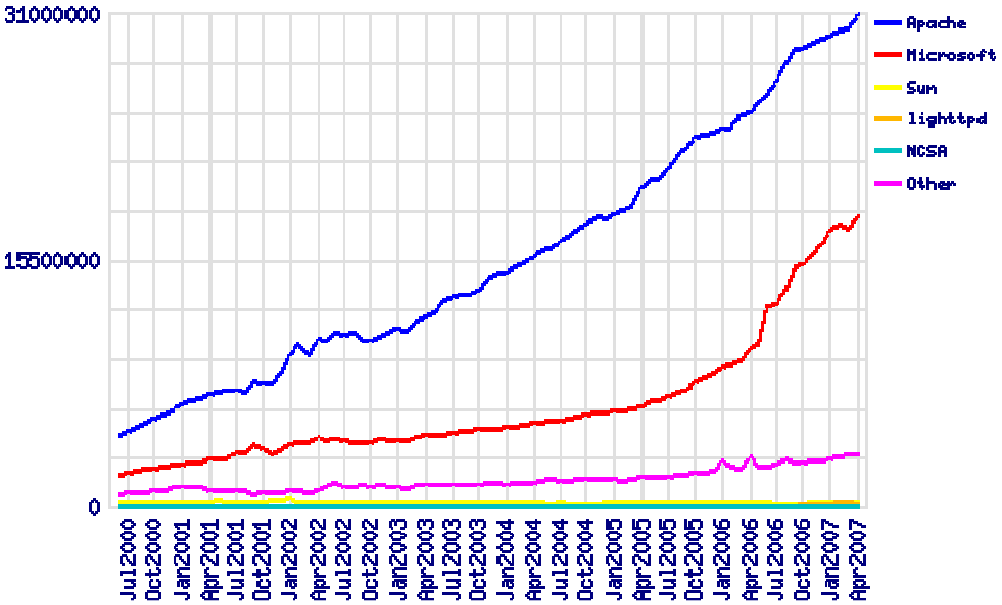
Desarrolladores	Marzo 2007	Por ciento	Abril 2007	Por ciento	Cambiar
Apache	64747516	58,62	66899485	58,86	0,24
Microsoft	34265321	31,02	35380121	31,13	0,11
Domingo	1851269	1,68	1907610	1,68	0,00
lighttpd	1399786	1,27	1382843	1,22	-0,05
Zeus	525405	0,48	488838	0,43	-0,05

Sitios activos

Desarrolladores	Marzo 2007	Por ciento	Abril 2007	Por ciento	Cambiar
Apache	30049510	58,58	30882069	58,50	-0,08
Microsoft	17430711	33,98	18181813	34,44	0,46
Zeus	220936	0,43	218410	0,41	-0,02
Domingo	189938	0,37	196334	0,37	0,00
lighttpd	234389	0,46	166751	0,32	-0,14

Activos totales de servidores en todos los dominios

De junio de 2000 - abril 2007



Sun es la suma de los sitios corriendo SunONE, iPlanet-Empresa, Netscape-Enterprise, Netscape-FastTrack, Netscape-Commerce, Netscape-Comunicaciones, Netsite-Commerce & Netsite-Comunicaciones.

Microsoft es la suma de los sitios que ejecutan Microsoft Internet Información Server, Microsoft-IIS, Microsoft-IIS-W, Microsoft-PWS-95, & Microsoft-PWS.