Configuración de servidores en Andriod

Introducción y objetivos

En este trabajo vamos a analizar las diferentes herramientas que proporciona el sistema operativo android para la creación de servidores. Presentaremos las diferentes alternativas que podemos elegir y nos quedaremos con la mejor de ellas. Además haremos pruebas de carga en el servidor que montemos y monitorizaremos los recursos antes, durante y después de la prueba.

A menudo nos encontramos con información de todo tipo sobre servidores en Windows, Linux e incluso MAC, pero no resulta tan familiar el sistema operativo Android para la creación de servidores, pese a ser el sistema operativo más utilizado en dispositivos móviles.

El objetivo del trabajo es mostrar cómo son los servidores que podemos montar en el sistema operativo de Google y las herramientas que podemos utilizar para realizar el mantenimiento.

Buscando por internet podremos ver que no es sencillo encontrar información sobre este tema (de hecho no he encontrado nada de información ni ningún tutorial realmente bueno).

¿Un servidor en Android?

Realmente merece la pena analizar si Android es el sistema operativo ideal para montar un servidor. La respuesta es clara: **actualmente no**. Los dispositivos como smartphones o tablets actualmente no pueden competir con una computadora en términos de procesador o memoria RAM, y por tanto nunca pueden ser sustitutos de una granja web de computadoras bien montada.

En cambio, existe un entorno en que puede ser útil un servidor en Android, el entorno doméstico. Todos nosotros tenemos algún viejo smartphone que ya no utilizamos y en vez de tenerlo ocupando espacio, podemos darle un uso: usarlo como servidor. La principal ventaja de los smartphones o tablets son su reducido tamaño, por lo que resulta muy interesante la posibilidad de llevar un servidor contigo, para poder compartir archivos donde necesites: en casa, en el trabajo... Además, aunque como hemos dicho, estos dispositivos no pueden competir en prestaciones con las computadoras, sí que tienen potencia suficiente para montar un servidor casero que no tenga que atender demasiadas peticiones (lo veremos en las pruebas) e incluso pueden atender más de las que pensamos.

Sin duda una de las ventajas de montar un servidor en Android es, además de su portabilidad, su simplicidad. Hay que mencionar que para realizar el experimento no he utilizado ningún tutorial, de lo que se deduce lo sencillo que puede ser esta tarea.

Resumen de las herramientas utilizadas

Para la realización del trabajo se han empleado las siguientes herramientas:

- Tablet bq Edison:

  Sistema operativo: Android 4.4

  4 procesadores ARM 1.3 GHz

  RAM: 2048 MB

- Ubuntu server virtualizado con Virtual Box con 2048 MB de RAM en un equipo anfitrión Asus con procesador Intel Core i7-4510U de 3.1Ghz

Ahora, en cuanto al software necesario para montar los servidores, hemos elegido los siguientes:

- MyFTPServer
- ServersUltimate
- Server & Website Monitor
- Siege

Por hacer una reseña de ambos: MyFTPServer es gratuito y ServersUltimate cuesta unos 5 euros y te deja utilizar una versión gratuita durante 7 días. Por contra, MyFTPServer es mucho más limitado que ServersUltimate y además este último permite opciones como: acceder al servidor mediante terminal (para ello tienes que ser root en el dispositivo), hacer ping a una dirección, crear directivas de claves ssh...

MyFTPServer

Como su nombre indica, sirve únicamente para crear servidores FTP. Es muy sencillo de utilizar y simplemente indicaré cómo se usa sin profundizar en él. En la siguiente imagen podemos ver su pantalla de inicio:

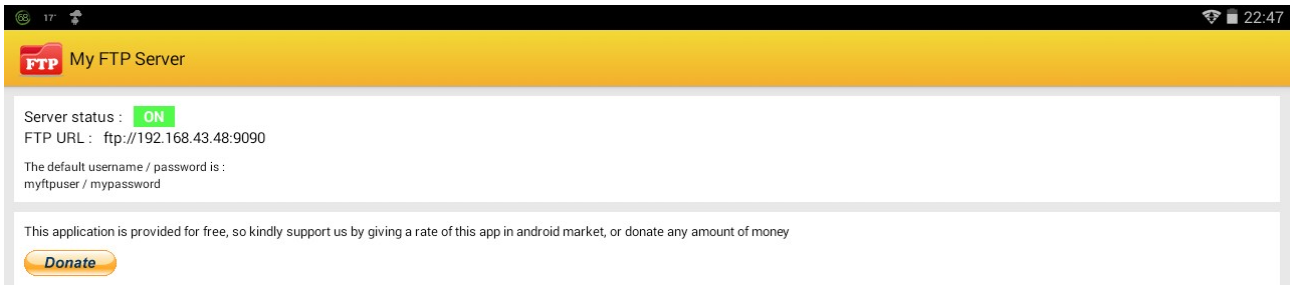

Ilustración 1: Pantalla de inicio de MyFTPServer

Si nos fijamos es realmente sencillo de utilizar; nos da una dirección IP privada que usaremos para acceder al servidor, mediante Filezilla por ejemplo, como podemos ver en la imagen siguiente:

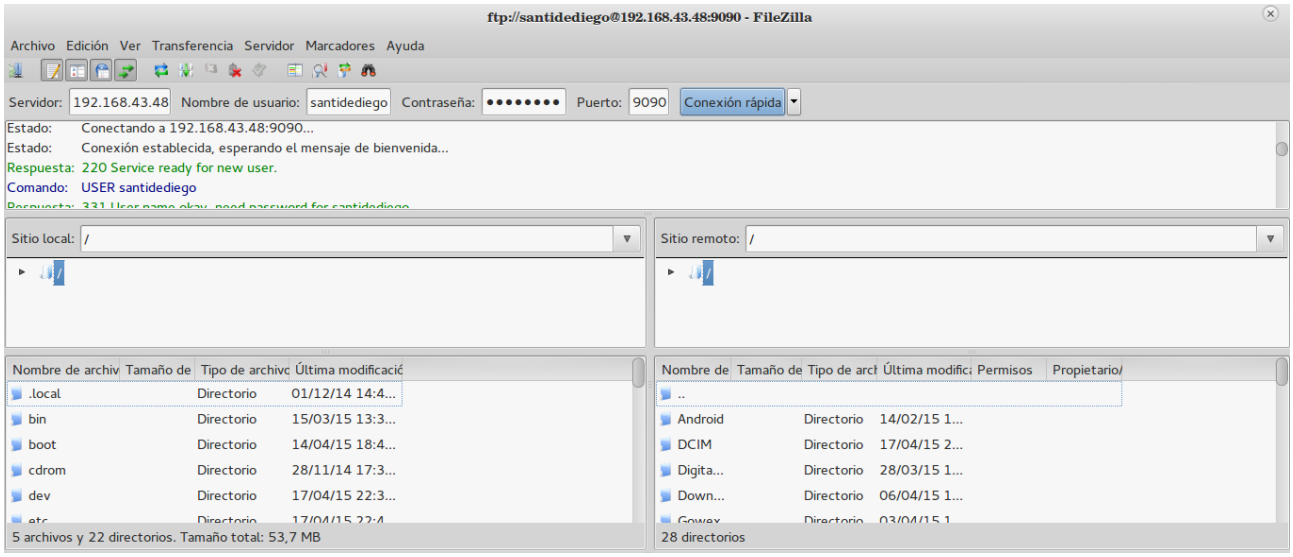

Ilustración 2: Accediendo al servidor FTP con Filezilla

En mi caso he cambiado el usuario y la contraseña por defecto. Esto se puede hacer en el botón ajustes que se encuentra en la pantalla de inicio de la aplicación.

No profundizaremos más en esta herramienta porque existen otras mejores para crear servidores como por ejemplo la que veremos a continución, que es la mejor alternativa para crear servidores.

ServersUltimate

Estamos ante la mejor y más completa herramienta para la creación de servidores en Android. Es muy sencilla de utilizar, pero no por ello poco potente porque, como ya veremos podemos hacer casi de todo lo que se nos ocurra.

Podemos crear una increíble variedad de servidores: FTP, Web, MySQL, SSH, Proxy... y todos los que nos imaginemos. Nosotros vamos a crear un servidor Web y a subir una pequeña página web en él. Empecemos.

Una vez dentro del programa, entramos en “Simple” y nos encontramos lo siguiente:

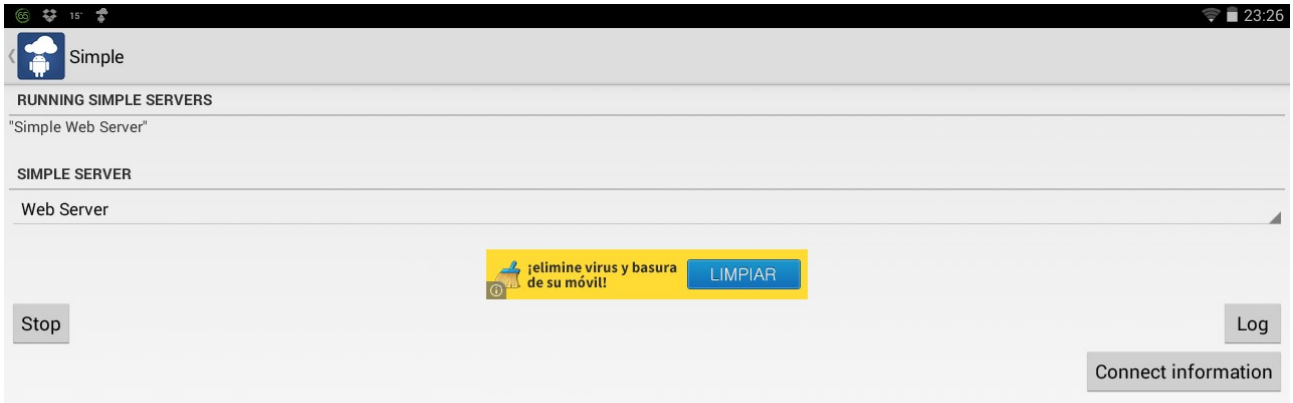

Ilustración 3: Creando un servidor Web

Haciendo click en Conectar ya lo tenemos conectado y en línea. Realmente sencillo; ahora vamos a ver algunas capturas para mostrar las opciones y el entorno del programa:

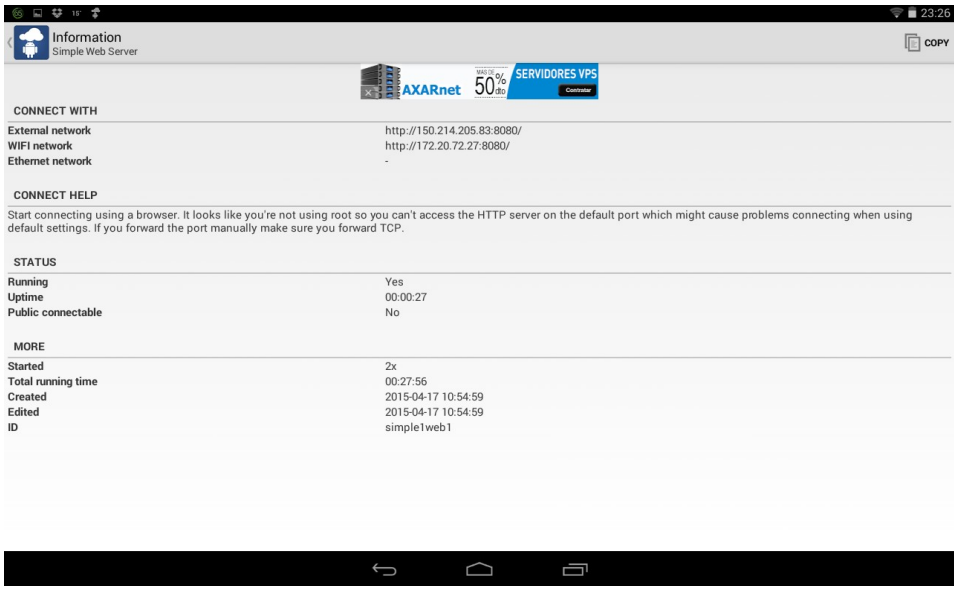

Ilustración 4: Pantalla de información del servidor

En la imagen anterior podemos ver la información de nuestro servidor recién creado, como su dirección IP, el tiempo que lleva online, etc. Ahora en la siguiente imagen podemos ver la pantalla de los logs. Ahí podemos ver los intentos de conexión a nuestro servidor. A la izquierda podemos ver una barra con todas las pantallas disponibles:

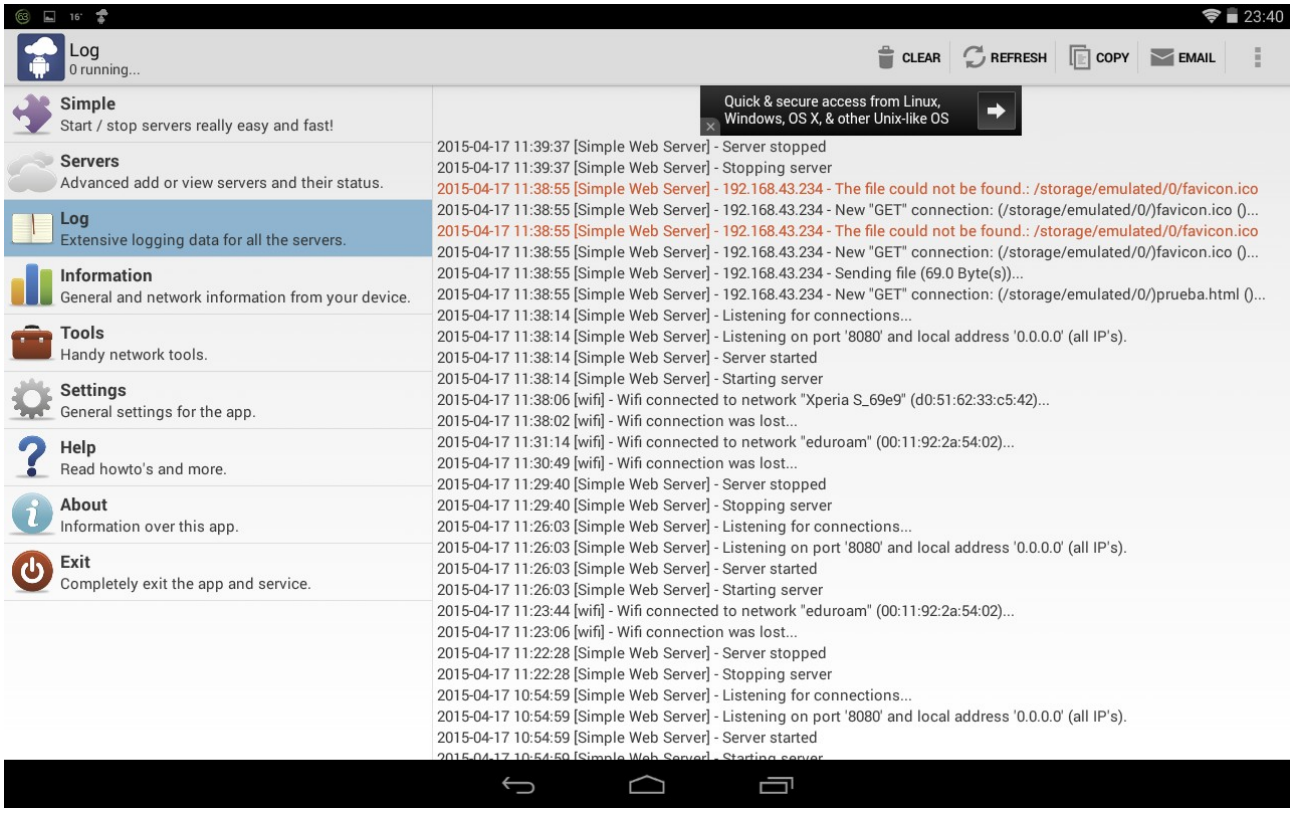

Ilustración 5: Logs de ServersUltimate

Una vez visto el entorno del programa, vamos a subir una página web sencilla. En la siguiente imagen podemos ver la ruta donde el programa aloja nuestras páginas web:

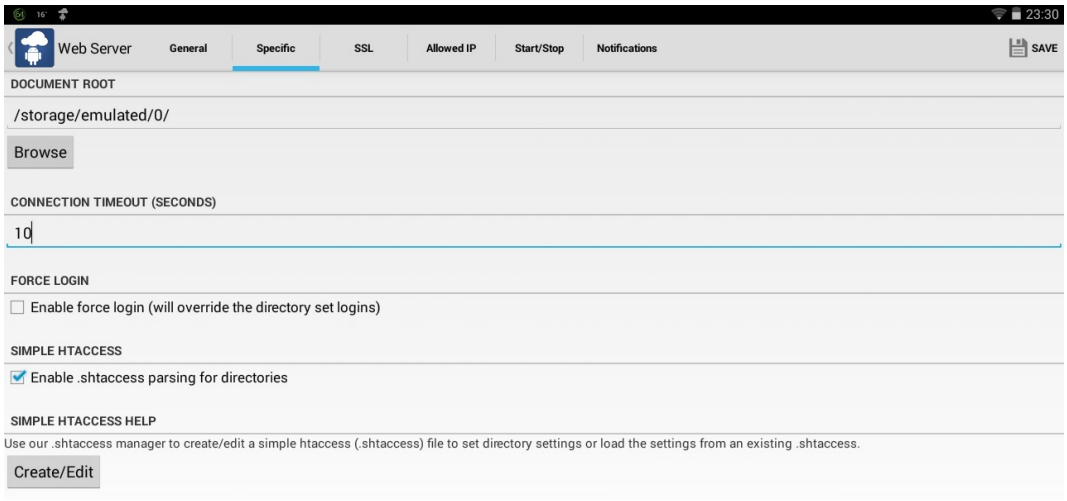

Ilustración 6: Configuración del alojamiento

Creamos una sencilla página web en esa ubicación con el editor DroidEditor de Android y después desde el navegador de nuestro ordenador vemos que efectivamente la página está en línea. Podemos encontrarla en la dirección vista antes en el puerto 8080 (también podemos elegir otro):

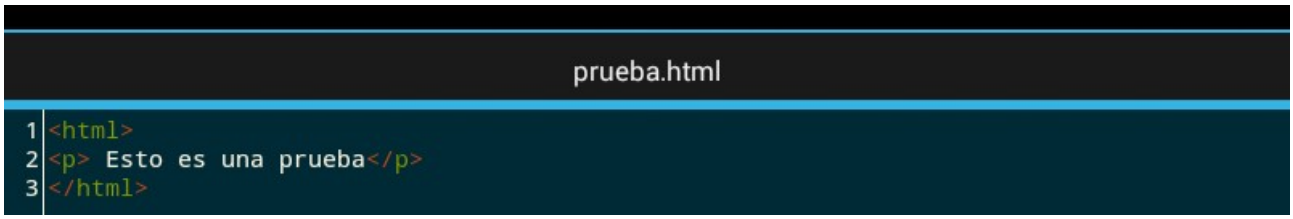

Ilustración 7: Archivo HTML

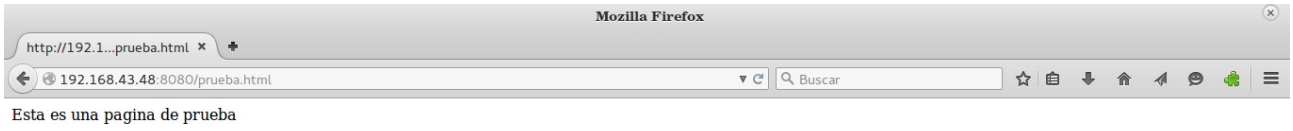

Ilustración 8: Página web vista desde el navegador

Pruebas de carga y monitorización

Una vez hecho esto, vamos a realizar pruebas de carga a nuestro servidor para hacernos una idea de la demanda de peticiones que puede soportar. Para ello utilizaremos el programa siege, que se utiliza principalmente para realizar pruebas de carga. En mi caso voy a lanzar siege desde mi ordenador, ya que uso Ubuntu como sistema operativo. En este caso las prestaciones de mi equipo son irrelevantes ya que lo único que va a hacer es lanzar ataques contra el servidor y recoger los datos.

Además, durante la carga, monitorizaremos el estado del sistema mediante el software para Android **Server & Website Monitor** que podemos encontrar gratuitamente en Play Store. Este software de monitorización es realmente sencillo de utilizar. Una vez instalado, añadimos la dirección del servidor que queremos monitorizar y le damos a "ON", así de simple. En mi caso lo he instalado en el mismo servidor y por tanto el servidor a monitorizar sería *localhost*.

Para instalar siege, basta escribir en la termina *sudo apt install siege* y ya lo tenemos. Una vez hecho esto, vamos a aplicar una serie de cargas a nuestro servidor Android a ver cómor responde. La primera es una carga muy pequeña, para tomar contacto con el servidor:

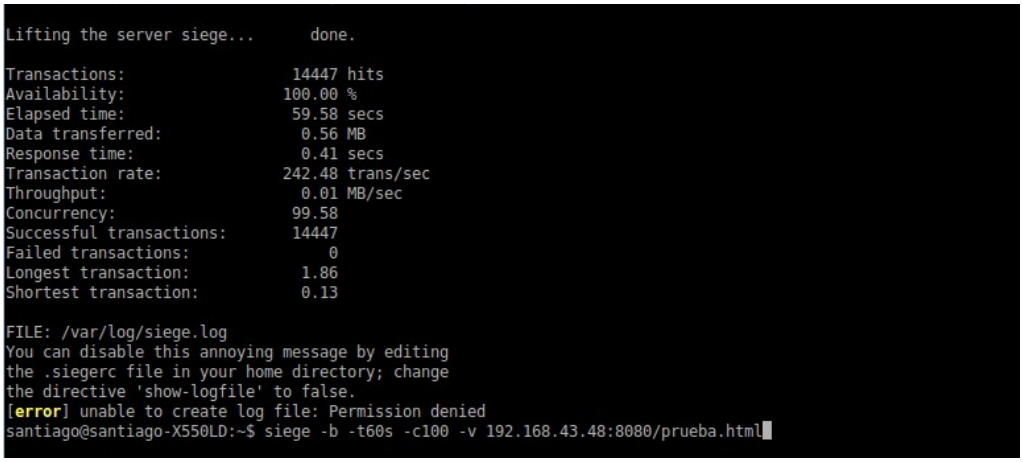

Ilustración 9: Carga de prueba

Como explicación, hemos enviado peticiones ininterrumpidamente durante 1 minuto simulando una concurrencia de 100 usuarios a la vez. Como podemos ver, el servidor las sirve sin ningún problema, de hecho, no ha hecho falta ni enseñar los resultados de la monitorización, ya que apenas se aprecia un incremento del tráfico. En la parte inferior de la imagen podemos ver la sintaxis de siege.

Vamos a probar ahora con una concurrencia un poco mayor (200 usuarios) y una duración de dos minutos:

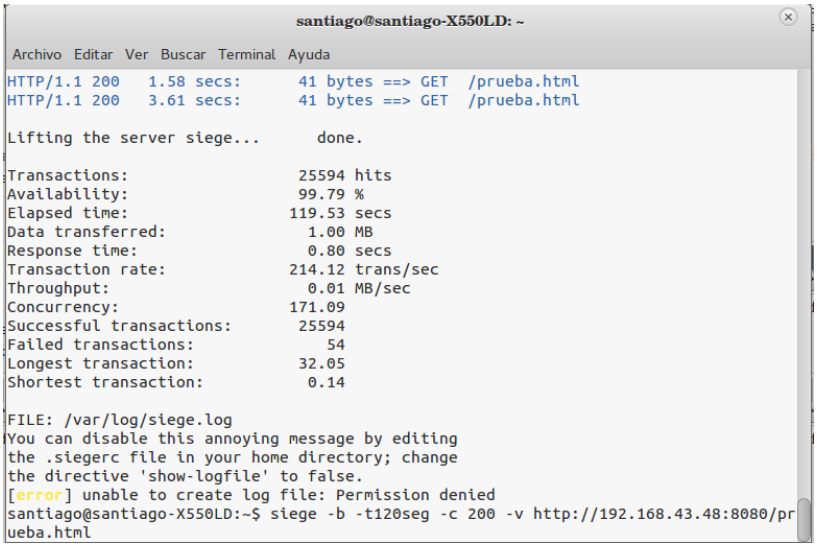

Ilustración 10: Segunda prueba de carga

Como podemos ver, ya al servidor le cuesta servir todas las peticiones, no ha podido procesar 54 de ellas. El software de monitorización nos revela picos de uso en el servidor:

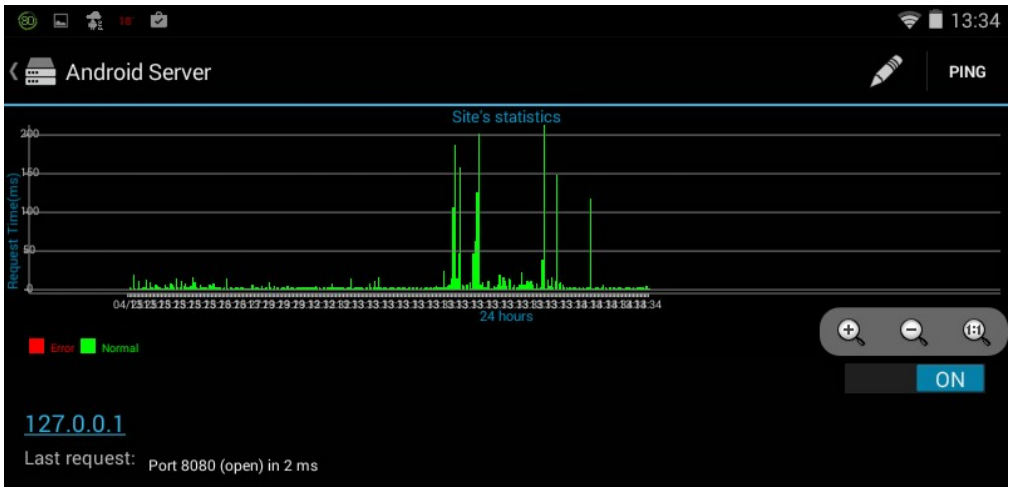

Ilustración 11: Picos de carga durante la prueba

Como podemos ver, el tiempo de respuesta aumenta considerablemente durante la prueba de carga.

Ahora vamos a forzar al servidor a que falle, para ello hacemos una carga de 4 minutos con 1000 usuarios de concurrencia:

```
HTTP/1.1 200   3.22 secs:      41 bytes ==> GET  /prueba.html
HTTP/1.1 200   2.22 secs:      41 bytes ==> GET  /prueba.html
HTTP/1.1 200  11.46 secs:      41 bytes ==> GET  /prueba.html
HTTP/1.1 200  11.46 secs:      41 bytes ==> GET  /prueba.html

Lifting the server siege...      done.

Transactions:                  56422 hits
Availability:                  98.91 %
Elapsed time:                 239.76 secs
Data transferred:               2.21 MB
Response time:                  3.95 secs
Transaction rate:             235.33 trans/sec
Throughput:                     0.01 MB/sec
Concurrency:                  930.71
Successful transactions:       56422
Failed transactions:             621
Longest transaction:           43.56
Shortest transaction:           0.02

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
[error] unable to create log file: Permission denied
santiago@santiago-X550LD:~$ siege -b -t240s -c1000 -v 192.168.43.48:8080/prueba.html
```

Ilustración 12: Prueba de carga severa

Como podemos ver, el servidor ya tiene dificultades para procesar todas esas peticiones, se ha dejado 621 peticiones por procesar. Además, si vemos el resultado que nos muestra el software de virtualización, podemos distinguir la zona de la prueba anterior, en forma de dos picos de mediana estatura y un último pico mucho más alto que corresponde a esta última prueba. Apenas se aprecia

el gráfico debido a la reducción de zoom que ha habido que hacer para que se aprecie por completo este último pico:

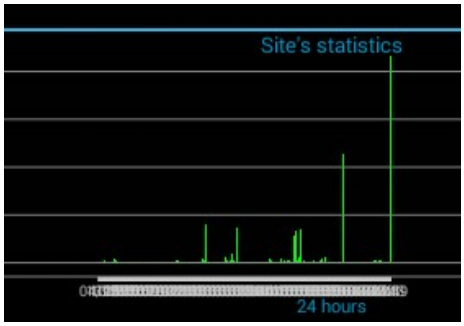

Ilustración 14: Monitorización durante la última carga

Mediante esta otra vista que nos ofrece el software de monitorización podemos ver de forma más limpia el último pico, que eclipsa a todas las pruebas anteriores:

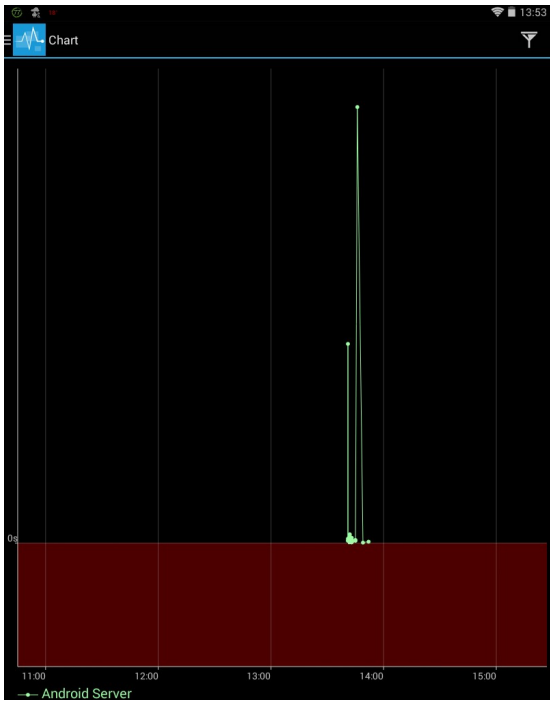

Ilustración 13: Monitorización durante la última carga con otra vista

Para concluir,a modo de comparativa, vamos a hacer la misma prueba contra un servidor Ubuntu Server que hemos montado en una máquina virtual. Los resultados los podemos ver aquí:

```
HTTP/1.1 200   31.05 secs:      51 bytes ==> GET  /prueba.html
HTTP/1.1 200   31.08 secs:      51 bytes ==> GET  /prueba.html
HTTP/1.1 200   31.08 secs:      51 bytes ==> GET  /prueba.html
done.
siege aborted due to excessive socket failure; you
can change the failure threshold in $HOME/.siegerc

Transactions:                  32903 hits
Availability:                  95.48 %
Elapsed time:                  42.80 secs
Data transferred:               1.60 MB
Response time:                  0.52 secs
Transaction rate:             768.76 trans/sec
Throughput:                     0.04 MB/sec
Concurrency:                  396.14
Successful transactions:       32903
Failed transactions:            1559
Longest transaction:           35.24
Shortest transaction:           0.05

FILE: /var/log/siege.log
You can disable this annoying message by editing
the .siegerc file in your home directory; change
the directive 'show-logfile' to false.
[error] unable to create log file: Permission denied
santiago@santiago-X550LD:~$ siege -b -t240s -c1000 -v 192.168.56.2/prueba.html
```

Ilustración 15: Misma prueba de carga contra Ubuntu Server

Como podemos ver, nuestro servidor Android ha respondido mucho mejor a las pruebas que la máquina virtual con Ubuntu Server. Además, en esta segunda prueba, el servidor se ha venido abajo

antes de que se llegase a la concurrencia máxima, como podemos ver la concurrencia real ha sido de 396.14 mientras que el servidor con Android ha llegado a soportar 930.71 de concurrencia; por no hablar de muchas más peticiones aceptadas y menos rechazadas.

NOTA: Otra aplicación para monitorizar todo tipo de servidores desde nuestro dispositivo Android puede ser Monyt. No la he escogido porque simplemente buscada algo rápido y fácil de configurar que arrojase gráficas visibles, pero Monyt supone una mejor alternativa que la aplicación que hemos usado en cuanto a monitorización de servidores.

Conclusión

Como se ha visto en las pruebas, un servidor montado en un dispositivo Android como es una tablet, tiene potencia suficiente como para aguantar pruebas de carga bastante severas, incluso mucho mejor que un Ubuntu virtualizado, con lo que supone una excelente alternativa a otros sistemas operativos para servidores, siempre y cuando estemos en un ámbito doméstico. En cambio, un servidor Android no puede competir a día de hoy con un computador y menos con una granja web completa y bien montada.

Además, sin más que crear un dominio propio (por ejemplo con DynDNS) podemos dar de alta nuestra web en internet para poder servir contenidos, igual que si lo hiciésemos con un servidor montado en otro sistema operativo.

Sea como fuere, parece ser que ya no es necesario tener un excesivo conocimiento en administración de sistemas para montar un servidor de una forma fiable y más que suficiente para según qué fines, y lo mejor de todo, el servidor irá con nosotros allí donde lo necesitemos.

Referencias

- Blog donde he descubierto la aplicación Servers Ultimate, no incorpora ningún manual, es de carácter divulgativo.

  http://www.redeszone.net/2013/08/14/convierte-tu-android-en-un-centro-de-servidores-con-servers-ultimate/

- Comando *man siege* en la terminal de Ubuntu, para ver como se usa el programa *siege*
- Play Store
- Concepto de prueba de carga:

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

Anexo: Herramienta siege

Siege es una herramienta que se utiliza principalmente para realizar pruebas de carga a servidores.

Podemos instalarla en linux sin más que escribir *sudo apt-get install siege* y una vez instalada es realmente sencillo utilizarla. Aquí podemos ver varias opciones del comando (las más importantes:

- -t: permite indicar el tiempo que durará la prueba
- -c: permite especificar la cantidad de usuarios concurrentes que simularemos, a mayor concurrencia, más carga recibirá el servidor
- -v: muestra el número de versión de siege
- -b: evita retrasos entre usuarios concurrentes

Existen otras alternativas a Siege, como pueden ser Httperf, Apache Benchmark, OpenWebLoad...

Ejemplo de uso:

Siege -b -t60 -c200 http://mipagina.com

Estamos realizando peticiones ininterrumpidamente durante 60 segundos con 200 usuarios solicitando el servicio a la vez.