Daniel Sánchez González
Pedro García Santana
Gonzalo Bonilla Pérez
Eduardo Vega De La Fe
1ºDAWMA

Tarea final de sistemas informáticos

Introducción

Este informe tiene como objetivo describir la mayoría del proceso por el que ha pasado el equipo de Daniel, Gonzalo, Pedro y Eduardo al realizar la última tarea del módulo de sistemas informáticos, que consistió en realizar un tutorial sobre cómo securizar un Ubuntu Server utilizando la metodología “Scrum” durante todo el camino.

Desarrollo técnico

Organización a través de metodologías ágiles (Gonzalo)

Hemos intentado implementar la metodología ágil "Scrum", utilizando como herramienta organizativa TRELLO, para ello tuvimos que investigar sobre dicha metodología, y acto seguido planteamos una estructura inicial en el panel canvas, la cual consistía en las siguientes 7 columnas:

-Backlog: Todas las tareas a realizar para finalizar el producto.

-Sprint 1: Tareas planeadas durante este sprint

-Sprint 2: Tareas planeadas durante este sprint

-Pending: Tareas en la cola de espera.

-In Progress: Tareas en proceso.

-Done: Tareas terminadas.

-Bugs: Errores a resolver lo antes posible.

Después de depurar, eliminamos las 2 columnas de Sprint, para reemplazarlas por las etiquetas "sprint 1" y "sprint 2" dentro de cada una de las tareas.

También nos hemos apoyado en un sistema de pesos, esto es, un número en cada tarea el cual indica el número de horas estimadas para llevarla a cabo.

El sprint 1 lo completamos sin agregar "user stories". Aquí es donde se justifica el porqué de los requisitos. Ej: "Quiero crear un repositorio en GitHub para favorecer la paralelización del desarrollo". Finalmente agregamos un “user story” en cada tarea del Trello.

Nuestro principal inconveniente a la hora de trabajar, fue no enfocarnos en paralelizar el trabajo, y abusar de hacer "mob programming" durante la realización de las tareas.

Git y GitHub (Gonzalo)

Hemos utilizado git y gitHub para desarrollar nuestra página web, dado que es un estándar en la industria que favorece la paralelización y facilita el versionado del proyecto.

Esta tarea consto de crear el repositorio en la web de gitHub, y una vez configurada la clave SSH, hacer un commit inicial del proyecto.

Con esto hecho, se explicó cómo se trabaja con el gitFlow desde el repositorio local, repasando los comandos básicos necesarios:

-git add
-git commit
-git push

Finalmente, se trabajó sobre el repositorio remoto, apoyándonos en 3 ramas: "Main", "Development", y "Hotfix".

Una vez terminado el desarrollo, se configuró una clave SSH en el servidor ubuntu donde, a través del comando “git clone” se descargaron los códigos de la web para ser alojados en la máquina.

Creación de la web (Gonzalo)

Para crear la página web, en un inicio pensamos en utilizar librerías como "bootstrap", para facilitar algo tan importante como son los estilos, finalmente llegamos a la conclusión de que era suficiente utilizar CSS básico.

Lo primero que hicimos fue crear un fichero "index.html", que es el encargado de guardar el contenido de la web. Luego creamos el fichero "styles.css", que se ocupa de dar a nuestra web el formato que deseamos, concretamente, el de un artículo.

Antes de tener los contenidos de la web, fuimos creando en la rama de desarrollo los estilos para las etiquetas que íbamos a utilizar principalmente, como lo son el "body", "p", "li", etc.

Luego, en la misma rama de desarrollo, metimos los contenidos de la web, donde se explica paso a paso la realización de las tareas del ubuntu server.

Durante el proceso, no fue necesario recurrir a la rama “hotfix”, la cual es la encargada de hacer un “merge” urgente sobre la rama “main” con el objetivo de arreglar fallos existentes en producción.

Instalación de Ubuntu Server y particionamiento RAID 1 (Pedro)

He instalado un Ubuntu Server 22.04 por medio de VirtualBox que tiene dos discos duros de 25 GB y -en el proceso- he hecho el particionamiento RAID 1.

El tipo de instalación fue “Ubuntu Server” (que contiene paquetes que permiten una administración más cómoda del servidor).

El formato del dispositivo RAID 1 es “ext4” y su tamaño es de aproximadamente 25 GB (el mismo que el de los dos discos).

En el proceso se instaló OpenSSH, que es necesario para próximas tareas.

La actualización a Ubuntu Pro no fue seleccionada, pero puede ser recomendable (con Ubuntu Pro las actualizaciones de seguridad se aplican a más paquetes). Tampoco se analizaron los “snaps” que la instalación ofrecía, pero podría haber sido interesante hacerlo para instalar alguno que nos conviniese.

Configuración del SSH (Pedro)

Creé una pareja de claves SSH en el Ubuntu Server con el objetivo de asociar la clave pública a mi cuenta de GitHub y así poder clonar el repositorio por medio de SSH. Se ha hecho porque el directorio del repositorio se usará como ejemplo al realizar las copias de seguridad.

En su momento tuve problemas al insertar la clave en GitHub:

1)El tamaño de una clave RSA (el tipo de clave que se me generaba) en GitHub debe ser de al menos 2048 bits para poder asociarla a tu cuenta y al principio no tuve esto en cuenta. Aún estableciendo en el comando generador de las claves que el tamaño fuera ese mínimo seguía teniendo el mismo error, por lo que creé una clave de 2500 bits.

2)Tuve dificultades para incluir la clave pública en mi cuenta de GitHub. Traté de copiar manualmente la clave en GitHub, pero no funcionó. Tampoco llegué a terminar de instalar las Guest Additions, que podían haber sido útiles. Lo que hice fue conectarme al servidor con SSH por medio de un Ubuntu Desktop, copiar la clave con Ctrl+Shift+C desde la terminal (mostrada en pantalla con “cat”) y pegarla en GitHub.

Sistema de copias de seguridad (Pedro)

El sistema de copias de seguridad fue creado para no tener que realizar en el servidor copias de seguridad del directorio del repositorio de GitHub manualmente. La forma en la que se hizo fue con dos scripts, uno para copias incrementales diarias y otro para copias completas.

“sudo apt update” y “sudo apt upgrade” fueron ejecutados previamente.

La herramienta usada para hacer las copias de seguridad fue y es “duplicity”. También se instaló “haveged” y “python3-boto” (aunque esto último fue innecesario). Además, hizo falta usar la herramienta “gpg”.

Hubo un inconveniente con el script de la copia incremental diaria, que consistió en que como no había una copia de seguridad en la que “duplicity” pudiera basarse para crear una copia incremental, el script no terminaba de ejecutarse correctamente. Para solucionarlo creé una copia de seguridad completa del directorio y con eso fue suficiente.

Instalación del software (Eduardo)

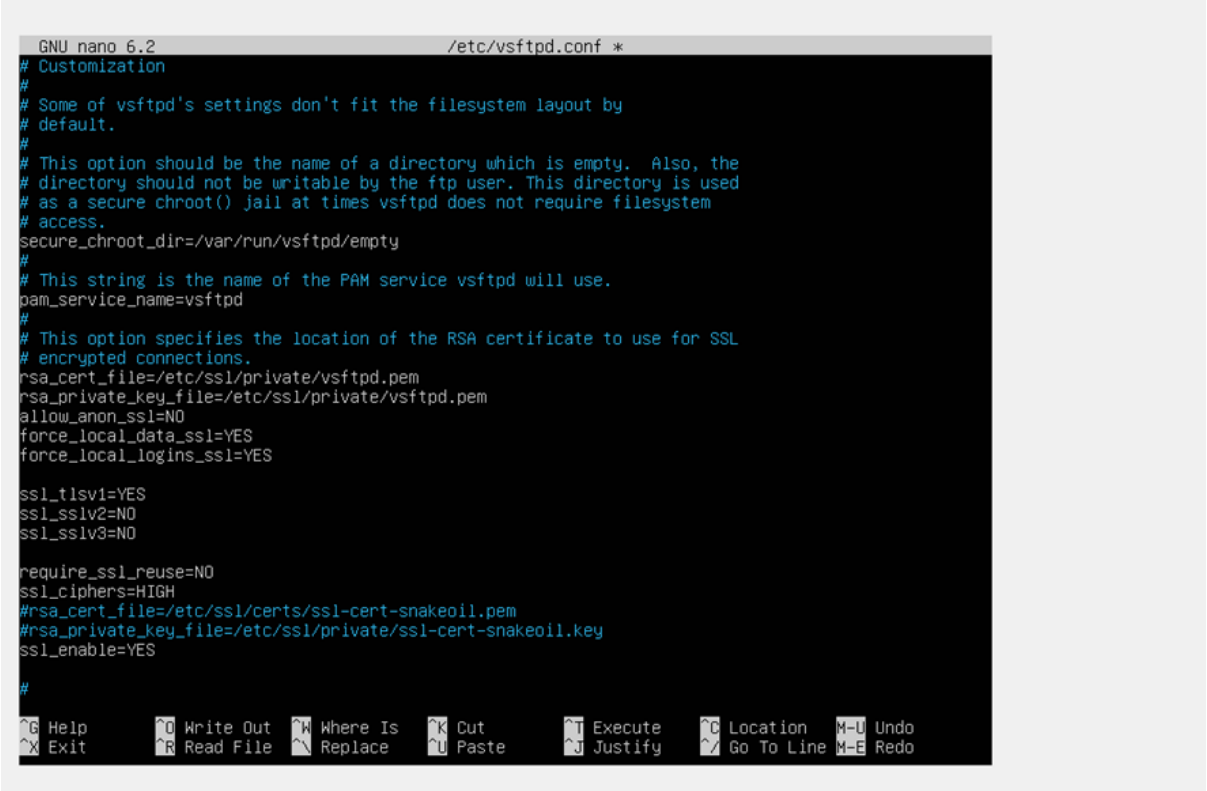
Descripción

Mi parte consistió en encargarme de la instalación del Software y la realización de la copia de seguridad del servidor. La instalación consistía exactamente en instalar los servidores SSH, FTP, MYSQL, Apache y su soporte PHP. Y la copia de seguridad en crear una copia incremental diaria y semanal de la carpeta donde se aloja nuestro servidor web para poder revertir cualquier cambio si hiciera falta.

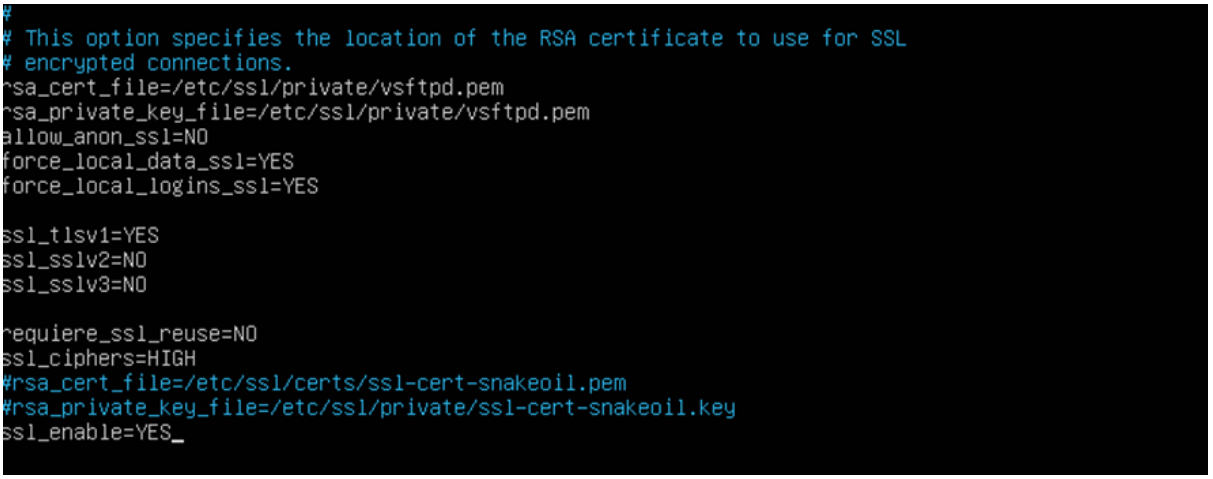
Proceso

Me tuve que documentar de antemano debido a la estructura lineal que sigue el proyecto que obliga a terminar una parte antes de empezar a trabajar en la siguiente. La primera parte de instalación del Software no fue muy compleja debido a que la gran mayoría de los pasos estaban en la actividad 6.2 realizada anteriormente. Hubo un pequeño percance con el archivo /etc/vsftpd.conf, que en vez de “inquire” puse “Inquiere” lo cual lo fastidió hasta que lo pude arreglar haciendo ese pequeño cambio, tras intentar hacer damage control del problema que surgió al dejar de funcionar el servidor FTP por un ajuste de configuración del servidor SSH que hizo Pedro .

Debería estar así

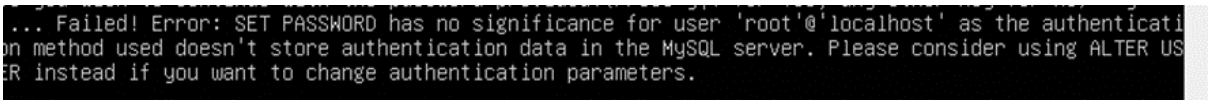


Cuando estaba así

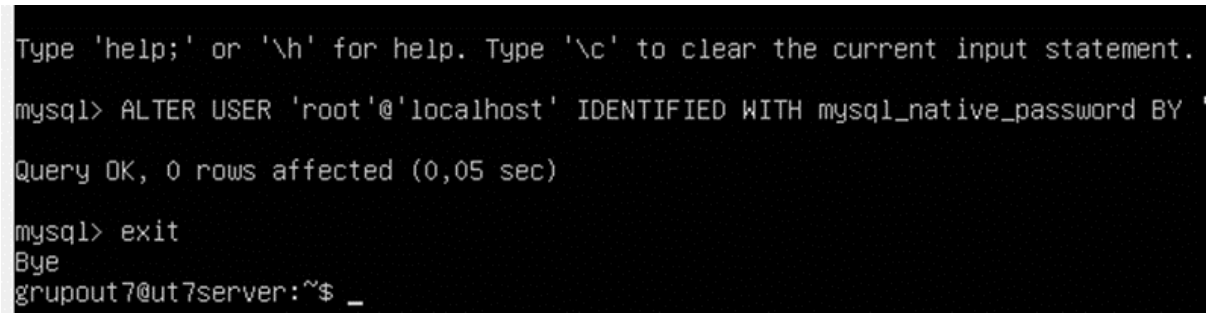


Luego procedí con la instalación del servidor MYSQL. Tuve problemas configurando la contraseña aunque ya estaba documentado de cuál era el fallo, lo cual tuve que realizar este paso primero de todo

Este error es el que surge



Y esto es lo primero que debí hacer



Es probable que debido a esto se me cambiara el acceso a la contraseña de SQL y diera error de que no requería contraseña asi que tuve que usar otro comando para entrar, que tuve que buscar el por que a veces daba el error que no requeria contraseña y otras veces, y di con el comando mysql -u root -p
https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-22-04

Esta es la página que seguí de guía para la instalación de php y mysql

Finalmente en el tema de Copia de seguridad, me documenté a su vez gracias a estas dos páginas

http://somebooks.es/copias-de-seguridad-en-ubuntu-server-20-04-lts-con-duplicity-parte-i/

http://somebooks.es/copias-de-seguridad-en-ubuntu-server-20-04-lts-con-duplicity-parte-i/

Para no pasar por el lastre que es descargar y subir la máquina virtual constantemente, que es algo muy lento, di mis apuntes y notas a Pedro, que solo tuve que seguir los pasos que le puse en la guía, más algunos apuntes de la página específicos. El único problema que le surgió es que tuvo que crear una copia de seguridad manual para que pudiera crear la incremental, pero ningún problema.

Securización del servidor (Daniel)

Para securizar el servidor, tuve que buscar mediante diferentes fuentes de información diversas formas de securizar tanto Ubuntu Server como las aplicaciones instaladas. Existen muchas formas de securizar Ubuntu Server y las aplicaciones que tuvimos que instalar, sin embargo, decidí utilizar los métodos que consideré más rápidos y efectivos para securizar, ya que hacerlos todos no solo sería imposible debido a la falta de tiempo, sino que, además, hay algunos métodos que, si bien pueden ser efectivos, también pueden ser muy arriesgados. Por ejemplo, una forma de securizar Ubuntu Server es desinstalar algunas aplicaciones que no se estén utilizando. A primera vista podría parecer que no es gran cosa, pero lo cierto es que, si desinstalas sin querer algo que no deberías, podría acabar mal y tendríamos que rehacer todo el trabajo, de forma que el riesgo es mucho mayor que el beneficio de tener un poco más de seguridad. Al final, si bien no fue una parte

especialmente fácil, lo cierto es que pensaba que sería mucho más difícil de lo que fue al final. El único inconveniente más o menos grave es que al principio no pude acceder a MySql para securizarlo simplemente porque no tenía la contraseña de dicha aplicación, un pequeño fallo de coordinación que, por suerte, se pudo solucionar fácilmente.

Inconvenientes generales

Nuestra organización ha sido pésima durante todo el proyecto. No valoramos bien cuánto duraba cada paso, y se nos descolocó la organización de los Sprint cuando tardamos mucho en particionar la máquina virtual, más el resolver el error que hacía que no funcionara el servidor FTP el lunes hizo que no se aprovecharan tan bien como podían haberse aprovechado las dos horas del lunes 12. No tener un story para ver los progresos de las tareas hizo que fuera más complicado tener seguimiento de cómo de avanzada estaban las mismas, pues solo se sabían cuándo estaban acabadas o si estaban pendientes.

Fuera de Gonzalo, la mayoría no estábamos muy acostumbrados a usar Git y Github, igual si hubiéramos estado más familiarizados con el mismo, podríamos haber avanzado más rápido en la máquina virtual, así que es algo a tener en cuenta para nuestros próximos proyectos.

Y algunos fallos tontos respecto a las instalaciones pudieron haberse solucionado si me hubiera fijado bien en que tenía todo bien escrito justo como me pedía, o en cuando me saltaría el error de lo de la contraseña, lo cual se resuelve haciendo más instantáneas para ver cual es el proceso que seguí y exactamente en qué paso lo hice mal, en vez de tomarlas justo tras terminar un paso entero.

Conclusiones

Aunque el trabajo ha podido salir bien, hemos utilizado de manera muy subóptima las metodologías ágiles y Scrum, aunque nuestro trabajo en sí de cada tarea no haya sido malo. La mejora consistiría en planificar de forma aproximada las duraciones de cada tarea y paso, trabajar más de forma remota y no tanto en Mob programming todos detrás de un ordenador valiéndonos de Git y Github. Con una mejor organización, creo que habríamos tenido más clases para poder trabajar de forma eficiente y no esperar todos a que uno terminara su paso para seguir con lo suyo.