



1. Iniciar una sesión de trabajo en GNU-Linux.

2. ¿Qué es el servicio IaaS de la ULL? (<https://iaas.ull.es/>)

El concepto de **Infraestructura como Servicio** (IaaS, *Infrastructure as a Service*) es uno de los tres modelos fundamentales en el campo de la computación en la nube (*cloud computing*), junto con el de **Plataforma como Servicio** (PaaS, *Platform as a Service*) y el de **Software como Servicio** (SaaS, *Software as a Service*). Al igual que todos los servicios en la nube, IaaS proporciona acceso a recursos informáticos situados en un entorno virtualizado, la “nube” (*cloud*), a través de una conexión pública, que suele ser Internet. En el caso de IaaS, los recursos informáticos ofrecidos consisten, en particular, en hardware virtualizado, o, en otras palabras, infraestructura de procesamiento. Físicamente, el repertorio de recursos de hardware disponibles procede de multitud de servidores y redes, generalmente distribuidos entre numerosos centros de datos, de cuyo mantenimiento se encarga el proveedor del servicio en la nube. El cliente, por su parte, obtiene acceso a los componentes virtualizados para construir con ellos su propia plataforma informática. Se ha solicitado una máquina virtual para cada alumno de manera que la puedan administrar.

3. Acceda al IaaS del *Servicio TIC* de la ULL.

Para ello abra en un navegador el sitio: (<https://iaas.ull.es/>)

4. En el panel de entrada de oVirt (*Open Virtualization Manager*) que se despliega:

- a) Introduzca como **Nombre de Usuario** su aluXXXXXXXXXX.
- b) Introduzca su contraseña institucional.
- c) Seleccione como perfil el de la ULL.

5. Cuando se despliegue el panel de administración de oVirt:

- a) Configure la consola de su máquina virtual.
 - 1) La máquina virtual que se le ha asignado tiene la etiqueta “LPP-2223”, 1GB de memoria y una CPU. Pulse el botón TOMAR UNA MÁQUINA.
 - 2) Seleccione el botón para ENCENDER para poner en marcha la máquina virtual.
 - 3) VNC (*Virtual Network Computing*) es una herramienta software que permite controlar un servidor remotamente a través de un cliente. También se le denomina software de escritorio remoto. Se activa si presiona el botón denominado “Consola VNC (explorador)”
- b) Obtener la dirección IP de la máquina virtual, a través de la consola que se ha desplegado en la ventana emergente.
 - 1) Introduzca como “*Nombre de usuario*”: **usuario** y como “*Contraseña*”: **usuario**

- 2) **Cambie la contraseña** de la cuenta **usuario**
- 3) Ejecute la orden: **ifconfig**
- 4) Copie la información asociada a la etiqueta **eth0** con nombre **Direc. inet**. Ha de tener la forma: **ddd.ddd.ddd.ddd**
- 5) Se recomienda NO utilizar la consola de la interfaz web (NO FUNCIONA).
- c) De manera opcional se puede obtener la dirección IP de la máquina virtual, a través de la página web (<https://iaas.ull.es/ovirtadmin/usuarios/mismaquinas>)
 Copie la información asociada a la columna **Direcciones IP** con nombre **nic1..** Ha de tener la forma: **ddd.ddd.ddd.ddd**
6. Abra una terminal.
7. Conéctese de forma remota a la máquina virtual asignada (**ssh usuario@ddd.ddd.ddd.ddd**)
 Si no es posible, tenga en cuenta que para poder establecer la conexión es necesario estar en la red privada de la ULL.
 Si la conexión es externa seguir las instrucciones que dan desde el *Servicio TIC* para crear una red privada virtual: VPN - *Virtual Private Network* (<http://www.ull.es/stic/tag/vpn/>).
 Desde el *Centro de cálculo* de la ESIT, a pesar de estar dentro de la red privada de la ULL, es NECESARIO abrir en un navegador <http://acceso.ull.es> y AUTENTICARSE con el '*Nombre de Usuario*' y la '*contraseña institucional*'.
8. Opcionalmente actualice el sistema operativo de la máquina virtual (puede tardar más de media hora).
9. Compruebe que **git** está instalado (**usuario@ubuntu:~\$ git --version**)
10. En el HOME cree un directorio de trabajo para la asignatura "Lenguajes y Paradigmas de Programación". (**cd; mkdir LPP**)
11. ¿Qué es **git**?. (**man git**)
 El control de versiones es el registro de los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que se puedan recuperar versiones específicas en un momento dado.
 Git es un software para control de versiones. Su principal característica es que es distribuido, además, permite el mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente de manera eficiente y fiable.
12. Configure **git** con el **nombre de usuario** de manera que pueda etiquetar de forma correcta las actualizaciones que este realice. (**git config --global user.name "Nombre Apellido"**)
13. Configure **git** con la dirección de correo electrónico para asociarla a las actualizaciones que se hagan en el repositorio git. (**git config --global user.email "aluXXXXXXXXXX@ull.edu.es"**)

14. `git` permite almacenar la configuración global de un usuario en el archivo `.gitconfig`. Este archivo se encuentra en el directorio `HOME` del usuario. `git` almacena el remitente y el nombre del autor de un cambio en cada confirmación en el repositorio. Esta información se puede almacenar en el fichero de configuración global de manera que no se solicite cada vez que se haga una confirmación. Muestre el contenido del fichero `.gitconfig` (`cat .gitconfig`)
15. Configure `git` de manera que no sea necesario introducir la contraseña cada vez que se hace una actualización. (`git config --global credential.helper cache`)
16. Configure `git` de manera que todos los cambios se empujen siempre en el repositorio `git`. (`git config --global push.default "matching"`)
17. Configure `git` de manera que se eviten las confirmaciones (*commits*) innecesarias. (`git config --global branch.autosetuprebase always`)
18. Muestre las configuraciones globales de `git`. (`git config --list`)
19. Sitúese en el **directorio** de la asignatura “Lenguajes y Paradigmas de Programación” esto es en el directorio *LPP* (`cd LPP`).
20. Muestre el contenido del directorio actual (`ls -la`).
21. Cree un nuevo directorio denominado *prct01* (`mkdir prct01`). Este será el **directorio de trabajo** durante la realización de esta práctica.
22. Sitúese en el directorio *prct01* y cree la estructura de directorios que le permita tener subcarpetas para el código y los documentos, es decir:
 - un subdirectorio *src*
 - un subdirectorio *docs*
23. Guarde el fichero PDF que contiene el enunciado de esta práctica en el directorio *docs*.
24. Situado en el directorio de trabajo, inicialícelo para que sea un repositorio `git`. (`git init`)
25. Compruebe que se crea el directorio `.git` (`ls -la`).
26. Cree contenidos en el directorio de trabajo. Ejecute los comandos:

```
touch test01
touch test02
touch src/src01
touch src/src02
ls > test01
```
27. Compruebe el contenido del directorio de trabajo (`ls -la`).

28. Antes de realizar una confirmación en un repositorio Git es necesario marcar qué cambios se deben confirmar añadiendo los nuevos ficheros y los ficheros cambiados al **índice del repositorio git**, esto es, al área de preparación. Esto crea una instantánea de los ficheros afectados. Si después de la instantánea, se cambia uno de los ficheros antes de confirmarlos, es necesario añadir el fichero de nuevo al índice para registrar los nuevos cambios. Añada todos los ficheros y subdirectorios del directorio actual al *índice del repositorio git*. (`git add .`)
29. Confirme (*commit*) los cambios del índice en el repositorio git local.
(`git commit -m "Primera confirmación - vacíos"`)
30. Muestre el fichero con las confirmaciones realizadas en el repositorio hasta el momento.
(`git log`)
31. Modifique los ficheros del directorio de trabajo. (

 `echo "Hola desde el fichero test01" > test01`
 `echo "Hola desde el fichero test02" > test02`

)
32. Compruebe las diferencias entre los ficheros anteriores y los nuevos. (`git diff`)
33. Confirme (*commit*) los cambios del índice en el repositorio git local. La opción `-a` permite registrar los cambios de los ficheros modificados, pero no añade ficheros nuevos automáticamente al índice.
(`git commit -a -m "Diciendo hola"`)
34. Modifique los ficheros del directorio de trabajo actual. (

 `echo "Adios desde el fichero test01" > test01`
 `echo "Adios desde el fichero test02" > test02`

)
35. Muestre el estado del repositorio git local, esto es, qué ficheros han cambiado, cuáles son nuevos y cuáles han sido borrados. (`git status`)
36. Muestre las diferencias entre los ficheros sin confirmar y los de la última confirmación. (`git diff`)
37. Añada los cambios al *índice del repositorio git* y confírmelos.
(`git add . && git commit -m "Maaaas cambios - con un error sintáctico en el mensaje"`)
38. Muestre la historia de las distintas confirmaciones (*commits*) en la rama actual. (`git log`)
39. Arregle el error en el mensaje de la última confirmación (*commit*) del apartado 37.
(`git commit --amend -m "Más cambios - ahora sin errores"`)
40. Cree un fichero y póngalo bajo el control de versiones. (

 `touch sinsentido.txt`
 `git add . && git commit -m "se ha creado un nuevo fichero sin sentido"`

)

-)
41. Elimine el fichero del directorio. (`rm sinsentido.txt`)
 42. Añada los cambios al *índice del repositorio git* y confírmelos.
(`git add . && git commit -m "se ha eliminado el fichero sinsentido.txt"`)
 43. ¿Qué es GitHub? (<http://github.com>)
GitHub es una plataforma para la organización y almacenamiento de código con funcionalidades que permiten el control de versiones y la colaboración en la realización de proyectos.
 44. Cree una cuenta en *GitHub*.
 - a) Abra en el navegador el sitio de GitHub: <http://github.com>
 - b) Pulse el botón verde que aparece en pantalla
 - c) Introduzca como Nombre de Usuario su aluXXXXXXXXXX
 - d) Introduzca su dirección de correo electrónico institucional: aluXXXXXXXXXX@ull.edu.es
 - e) Introduzca su contraseña
 - f) Pulse el botón verde para crear la cuenta
 - g) Acceda al correo electrónico institucional y verifique la cuenta creada
 45. Muestre el árbol de directorios de su HOME (`tree`).
 46. Compruebe si existe el directorio `.ssh` (`cd ~/.ssh`).
 47. Si la respuesta es “No existe tal directorio” continúe por el ejercicio 50.
 48. Si la respuesta es afirmativa, cree un directorio con nombre *copia* dentro del directorio `.ssh`
(`mkdir ~/.ssh/copia`).
 49. Mueva la pareja de clave-pública clave-privada al directorio *copia*
(`mv ~/.ssh/id_rsa* ~/.ssh/copia/`).
 50. Genere una nueva pareja de clave-pública clave-privada en el directorio `.ssh` (`ssh-keygen -t rsa`). Para usar las opciones por defecto, a cada pregunta responda pulsando la tecla de retorno de carro.
 51. Muestre por la consola la clave-pública que ha generado (`cat ~/.ssh/id_rsa.pub`)
 52. Copie en el almacenamiento temporal la clave-pública. Para ello, márkela con el ratón y pulse las teclas **Ctrl+C**.
 53. Añada su clave-pública a *GitHub*.
 - a) En la barra de usuario, en la esquina superior derecha de la página, haga clic en el icono de “Configuración de la cuenta” (*Account Settings*).
 - b) En la barra de opciones que aparece en la izquierda haga clic en la etiqueta “Claves SSH” (*SSH Keys*).
 - c) Haga clic en el botón “Add SSH key” que aparece a la derecha.

- d) En el campo de texto Título (*Title*) escriba *Centro de Cálculo*
 - e) En el área de texto Clave (*Key*) pegue (Ctrl+V) la clave que copió en el ejercicio 52.
 - f) Pulse el botón verde para crear la clave (*Add key*).
54. Cree un repositorio en *GitHub*
- a) En la barra de usuario, en la esquina superior derecha de la página, haga clic en el icono de “Crear un repositorio nuevo” (*Create a New Repo*).
 - b) Introduzca el nombre `prct01`
 - c) Seleccione que quiere hacer el repositorio público.
 - d) NO seleccione la casilla de crear el fichero README.md.
 - e) Pulse el botón para crear el repositorio (*Create repository*)
 - f) Copie la dirección “ssh” del repositorio (Ctrl+C)
55. Cree un repositorio remoto con nombre corto *ghp01*
- ```
(git remote add ghp01 git@github.com:aluXXXXXXXX/prct01.git)
```
56. Empuje los cambios en el repositorio remoto denominado *ghp01*.
- ```
( git push -u ghp01 master )
```
57. Muestre los detalles del repositorio remoto denominado *ghp01*. (`git remote show ghp01`)
58. Muestre los repositorios remotos que están definidos. (`git remote -v`)
59. Acepte la invitación para realizar la tarea en GitHub Classroom.
- ```
(https://classroom.github.com/a/G_wkHQ_J)
```
60. Cree un grupo con un único miembro con nombre: Nombre-Apellido-aluXXXXXXXXXX
- ```
( Nombre-Apellido-aluxxxxxxxxxx )
```
61. Cree un enlace al repositorio remoto privado creado en la organización LPP2223 denominado *rpp01*.
- ```
(git remote add rpp01 git@github.com:ULL-ESIT-LPP-2223/fundamentos-git-github-Nombre-Apellido-aluxxxxxxxxx.git)
```
62. Muestre los repositorios remotos que están definidos. ( `git remote -v` )
63. Empuje los cambios en el repositorio remoto denominado *rpp01*.
- ```
( git push -u rpp01 master )
```
64. Muestre los detalles del repositorio remoto denominado *rpp01*. (`git remote show rpp01`)
65. Muestre los repositorios remotos que están definidos. (`git remote -v`)
66. Escriba las direcciones HTTP de los repositorios público y privado que ha creado en la tarea habilitada en el campus virtual.
67. Cierre la sesión.