### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 9 de enero de 2014.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagsr y comprueba que esto ha creado los fiche- ros ~/lagsr.enero.14/parte.TULOGIN.py, y ~/lagsr.enero.14/practico.TULOGIN.txt, donde contestar´as las preguntas. (La cadena TULOGIN representa tu nombre de usuario

en el laboratorio)

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo u´nico que importa es lo que escribas en el examen, describiendo qu´e has hecho. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en el examen**

# Ejercicio 1 (6 puntos)

Edita el script con el nombre ~/lagsr.enero.14/parte.TULOGIN.py

de forma que, empleando la orden uptime, muestre por salida est´andar el tiempo que lleve encendido cada m´aquina del laboratorio en que trabajas ahora. El script debe funcionar aunque haya m´aquinas apagadas. El informe estar´a ordenado alfab´eticamente y tendr´a un aspecto similar a este:

kappa01 12:35

kappa02 12:33

kappa04 2:07

kappa05 2 days

Observaciones:

En el informe, los nombres de m´aquinas deben aparecer sin dominio

Es recomendable que reutilices c´odigo de tu pr´actica ~/lagsr/practica02/parte\_usuarios.py Para saber qu´e m´aquinas est´an encendidas, es recomendable que reutilices tu script

~/lagsr/practica02/parte02.py

El formato de salida de uptime cambia ligeramente cuando el tiempo es inferior o superior a 24 horas. Ejemplos:

mortuno@gsyc:~$ uptime

11:20:43 up 24 days, 1:54, 1 user, load average: 0.34, 0.32, 0.41

mortuno@kappa02:~$ uptime

11:21:42 up 3:27, 1 user, load average: 0.00, 0.01, 0.05

Para simplificar, toma la cadena entre la palabra up y la primera coma. Esto es, para tiempos superiores a 1 d´ıa, ignora las horas.

# Ejercicio 2 (1 punto)

Indica todos los pasos necesarios para montar por sshfs el directorio /var/tmp/

de la m´aquina del laboratorio cuyo nu´mero es uno m´as que el tuyo, en el directorio

/var/tmp/remota de tu m´aquina.

Ejemplo: si tu m´aquina es kappa07, monta el directorio /var/tmp/ de kappa08 en el directorio /var/tmp/remota de kappa07

Recuerda que si haces un paso, pero no cuentas en el examen que lo has hecho, es como si no lo hubieras hecho. Pega el resultado de la ejecuci´on, de forma similar a la memoria de pr´acticas.

# Ejercicio 3 (3 puntos)

Indica las o´rdenes necesarias para hacer lo descrito a continuaci´on, y pega el resul- tado de la ejecuci´on, de forma similar a la memoria de pr´acticas. Respeta los convenios habituales en el uso de git.

1. Toma el repositorio repo02 de tu pr´actica 4.2.2 y haz todo lo necesario para tener dos repositorios *esclavos* descendientes de ´el, llamados ~/lagsr.enero.14/repo.a y ~/lagsr.enero.14/repo.b
2. Crea en repo.a un fichero llamado examen.txt, escribe dentro la hora actual, m´etelo en el repositorio y haz todo lo necesario para propagar este cambio a repo.b
3. En repo.b, modifica el fichero examen.txt poniendo la hora actual (que sea distinta a la anterior, al menos un minuto m´as). Vuelve a meterlo en el repositorio. No lo propages a repo.a
4. Haz que repo.b vuelva al estado inmediatamente anterior a la modificaci´on que acabas de hacer en la pregunta 3.3. Consulta el estado del fichero examen.txt, comprueba que tiene la hora que escribiste en repo.a
5. Haz que repo.b vuelva a su estado m´as reciente, como si nunca hubieras hecho el apartado 3.4. Comprueba que el fichero examen.txt tiene el valor esperado.

Recuerda que si haces un paso, pero no cuentas en el examen que lo has hecho, es como si no lo hubieras hecho.

### Laboratorio de Administraci´on y Gestio´n de Sistemas y Redes Prueba escrita sobre la teor´ıa. 9 de enero de 2014

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

# Ejercicio 1 (2 puntos)

¿Qu´e es una *trap* de SNMP? ¿Qu´e pasar´ıa si no existiesen las traps ni nada parecido?

### Respuesta

Es un mensaje as´ıncrono desde un dispositivo administrado hasta un NMS. Las ope- raciones *get* y *set* son s´ıncronas, esto es, el NMS hace una solicitud y el dispositivo responde. Mientras que la operaci´on *trap* es as´ıncrona, la genera el dispositivo cuando sucede *algo importante*, sin que el NMS le *pregunte*. Por tanto, las *traps* resultan similares a las excepciones en los lenguajes de programaci´on

Si no hubiese este tipo de comunicaci´on as´ıncrona, el NMS no percibir´ıa las situaciones relevantes a menos que preguntase expl´ıcitamente por ellas al dispositivo.

# Ejercicio 2 (2 puntos)

Supongamos que creamos varios usuarios con la orden adduser, y posteriormente, por error, borramos el contenido del directorio /etc/skel. ¿Qu´e podr´ıa pasar?

### Respuesta

A los usuarios ya creados esto no les afecta, ya tienen su copia de los ficheros de configuraci´on por omisi´on. El problema lo tendr´ıan los usuarios creados posteriormente, a quienes les faltar´ıan estos ficheros.

# Ejercicio 3 (2 puntos)

La transparencia 12 del tema *shell II* dice *un int´erprete con bit SUID es muy peligroso, normalmente la activaci´on del SUID en un script no tiene efecto*

Explica esto.

### Respuesta

Para el usuario, ejecutar un script o ejecutar un ejecutable parece lo mismo, pero para el sistema es muy distinto. Un ejecutable se ejecuta sin m´as, pero un script es un fichero de texto plano que no se puede ejecutar por s´ı mismo. Al intentar ejecutarlo, el sistema operativo comprueba que el fichero no incluye el *nu´mero m´agico* 1 correspondiente a un ejecutable, pero s´ı empieza por los *nu´meros m´agicos* almohadilla admiraci´on, que indican que a continuaci´on el fichero especifica la ubicaci´on de un ejecutable (el int´erprete) que es el que realmente se ejecutar´a (cuando reciba el texto del script)2.

Por eso no tiene efecto activar el SUID en un script, porque el script no se ejecu- ta, se ejecuta el int´eprete (el int´erprete de shell bash, el int´erprete de python o el que corresponda)

Los int´erpretes normalmente pertencen al usuario root, como la mayor´ıa de ficheros *importantes* del sistema. Si un int´erprete perteneciente al root tuviera el bit SUID activo, cualquier script que se ejecutase en ´el tendr´ıa permisos de usuario root, con lo que podr´ıa hacer cualquier cosa en el sistema. Esto es muy peligroso. Activar el SUID en un fichero ejecutable *normal* puede tener cierto riesgo, pero muy reducido comparado con el caso anterior, porque podemos saber con bastante aproximaci´on qu´e va a hacer ese programa. Mientras que un int´erprete ejecutar´a cualquier o´rden que se le pase, es impredecible.

# Ejercicio 4 (4 puntos)

Deseamos que el ejecutable /usr/local/bin/ups sea un demonio que monitoriza el sistema de alimentaci´on ininterrumpida del ordenador, y que se ponga en marcha cada vez que se enciende el ordenador. Indica qu´e ficheros son necesarios y con qu´e nombre. Describe superficialmente cual ser´a su contenido. Debes seguir todos los convenios del arranque en System V.

### Respuesta

Para seguir correctamente todos los convenios, habr´a que renombrar

/usr/local/bin/ups para que pase a llamerse /usr/local/bin/upsd

Luego har´a falta un script que maneje el demonio. Se llamar´ıa /etc/init.d/ups, aceptar´ıa al menos los par´ametros START, STOP y RELOAD. Tambi´en podr´ıa aceptar otros como RESTART o STATUS. El demonio no tiene por qu´e manejarse de forma est´andar, el script es el que recibe ´ordenes est´andar y env´ıa al demonio ´ordenes que ya no son est´andar, el script *conoce* el interfaz particular de cada demonio concreto.

1Se llama nu´mero m´agico a aquel que, por convenio, tiene un significado especial

2Si no se indica la ubicaci´on del int´erprete, el sistema operativo usar´a el int´erprete por omisi´on del usuario

En tercer lugar, habr´a que hacer que el script se ponga en marcha cuando se encienda el ordenador, y que se detenga al apagarlo. Suponiendo que estemos en una distribuci´on basada en debian, crearemos un fichero dentro del directorio correspondiente al nivel de ejecuci´on por omisi´on, /etc/rc2.d, que ser´a un enlace simb´olico apuntando al script descrito en el p´arrafo anterior y cuyo nombre ser´a SNNups, donde NN ser´an dos d´ıgitos que indican el orden dentro de ese nivel, por ejemplo, S99ups.

Tambi´en ser´a necesario detener el servicio en los niveles 0 y 6 (*Halt* y *Reboot*, res- pectivamente). Para ello, en los directorios /etc/rc0.d y /etc/rc6.d crearemos enlaces simb´olicos apuntando al script, cuyo nombre ser´a KNNups, donde NN representan dos d´ıgi- tos, por ejemplo K99ups.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 17 de junio de 2014.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagrs y comprueba que esto ha creado los fiche-

ros ~/lagrs.junio.14/mayorfichero.TULOGIN.py, y ~/lagrs.junio.14/practico.TULOGIN.txt, donde contestar´as las preguntas. (La cadena TULOGIN representa tu nombre de usuario

en el laboratorio)

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo u´nico que importa es lo que escribas en el examen, describiendo qu´e has hecho. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en el examen**

# Ejercicio 1 (5 puntos)

Edita el script con el nombre ~/lagrs.junio.14/mayorfichero.TULOGIN.py

de forma que, tomando como base la salida de la orden de shell ls -l, muestre el nombre y el taman˜o del fichero de mayor taman˜o del directorio actual. El nombre del fichero debe incluir el *path* completo.

La salida tendr´a un aspecto similar a este:

/home/al-05-06/jperez/lagrs.junio.14/practico.jperez.txt 4823

Observaciones:

Obviamente, el script debe funcionar correctamente en cualquier directorio, no solo en el del examen

Es recomendable que reutilices c´odigo procedente de tu pr´actica mi\_ps.py

# Ejercicio 2 (2 puntos)

Monta en un directorio llamado ~/remoto de tu puesto del laboratorio, el directo- rio test del *home* del usuario *alumno* en la m´aquina 193.147.71.62. La contrasen˜a es *admin2014*

Escribe todos los pasos que has seguido. Cuando lo hayas hecho, avisa al profesor para que lo vea.

# Ejercicio 3 (3 puntos)

Indica las o´rdenes necesarias para hacer lo descrito a continuaci´on, y pega el resul- tado de la ejecuci´on, de forma similar a la memoria de pr´acticas. Respeta los convenios habituales en el uso de git.

1. Deseamos tener un repositorio maestro llamado ~/lagrs.junio.14/maestro.git

y un esclavo suyo llamado ~/lagrs.junio.14/esclavo Dentro del repositorio habr´a un fichero llamado prueba.txt Haz todo lo necesario para tener esto

1. Escribe dentro de prueba.txt el texto *hoy es viernes*. Haz un *commit* y prop´agalo.
2. Modifica el fichero para que diga *hoy es jueves*. Haz un *commit* y prop´agalo
3. Muestra un listado compacto de los *commit* que tienes
4. Vuelve el sistema al estado que ten´ıas en el paso 2
5. Modifica el fichero para que diga *hoy es viernes 20*. Haz un *commit* y prop´agalo

### Respuesta

mkdir abuelo; mkdir maestro.git; cd abuelo;

git init;

touch prueba.txt; git add prueba.txt;

git ca -m "Crea fichero vac´ıo"; cd ..;

git clone --bare abuelo maestro.git; mkdir esclavo;

git clone maestro.git esclavo; cd esclavo;

echo "hoy es viernes">prueba.txt; git ca -m "A~nade texto a fichero"; git push;

echo "hoy es jueves">prueba.txt;

git ca -m "Modifica texto en fichero"; git pull;

git log2;

git co e07e9e6;

echo "hoy es viernes 20">prueba.txt; git ca -m "A~nade dia del mes";

git push

### Laboratorio de Administraci´on y Gestio´n de Sistemas y Redes Prueba escrita sobre la teor´ıa. 17 de junio de 2014

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

# Ejercicio 1 (2.5 puntos)

La transparencia 57 del tema 2, que habla de los enlaces *duros*, dice lo siguiente: *Dado un fichero, se sabe cu´antos nombres tiene. Para saber cu´ales son sus nombres, habr´ıa que buscarlos*.

Explica esta afirmaci´on

### Respuesta

Dado un fichero llamado A, se le puede poner otro nombre, por ejemplo B. Decimos que B es un *enlace duro* de A. El nombre B no apunta al nombre A, sino al mismo fichero apuntado por A.

El sistema de ficheros almacena (en el inodo) el nu´mero de nombres que tiene cada fichero. Lo podemos ver con la orden ls -l, es el nu´mero que aparece tras los permisos. Cada nombre apunta al fichero, pero el fichero no apunta a sus nombres. Esto est´a res- presentado en la figura: hay una flecha desde cada nombre al fichero, pero la flecha va en un solo sentido. Por tanto, a partir del fichero, no es posible llegar a los nombres. El

fichero *sabe* que recibe dos flechas, pero no puede saber *de donde vienen*.

Para conocer todos los nombres del fichero A, habr´ıa que recorrer todo el a´rbol,

buscando otros nombres que apunten al mismo fichero. Por ejemplo usando la orden ls -i A, que nos indica el nu´mero de inodo del nombre A. Tendr´ıamos que buscar por todo el a´rbol otro nombre con el mismo nu´mero de inodo, esto es, otro nombre que apunte al mismo fichero.

# Ejercicio 2 (2.5 puntos)

¿Para qu´e sirve el *journal* en un sistema de ficheros?

### Respuesta

Para garantizar que las operaciones sean at´omicas, esto es, que se hagan por completo o que no se hagan en absoluto, pero que nunca queden incompletas.

Un sistema de ficheros es una estructura bastante compleja. Si hay algu´n problema en el sistema en mitad de una operaci´on, por ejemplo un apag´on, el sistema podr´ıa quedar en un estado inconsistente, con una parte de las estructuras actualizadas y otra no. Ejemplo: mover un fichero consiste en copiarlo en la nueva ubicaci´on y borrar el original. Si la operaci´on queda a medias, podr´ıamos tener dos ficheros o ninguno.

Un sistema de ficheros con *journal* lo que hace es almacenar lo que va a hacer, y luego hacerlo realmente. Si algo falla, una herramienta de recuperaci´on puede comparar lo que dice el *journal* con lo que realmente hay en el sistema de ficheros, y si hay alguna inconsistencia, repararla.

# Ejercicio 3 (2.5 puntos)

Tenemos un sistema Unix. Queremos instalar cierto software y disponemos de una versi´on .tgz y otra .deb.

1. ¿Cu´al deber´ıamos instalar? Si la respuesta es *depende*, ¿de qu´e depende?
2. ¿Qu´e pasar´ıa instal´aramos *el otro*?

### Respuesta

El formato .tgz es general para Unix, el .deb es espec´ıfico de Debian y derivados, por tanto:

1. Depende. Si tenemos un Linux Debian o algu´n otro sistema basado en Debian, como Ubuntu, deber´ıamos instalar el .deb. Pero si se trata de otra variante de Linux, o bien otro Unix que no sea Linux, entonces normalmente deber´ıamos instalar el

.tgz, que es m´as general.

1. Si intentamos instalar un .deb en un sistema que no est´a basado en estos paquetes, simplemente no podremos, no funcionar´a nada porque no estar´an disponibles los gestores de paquetes.

Pero normalmente s´ı ser´a posible instalar un .tgz en un Debian. No es recomendable porque lo estamos haciendo *de espaldas* al sistema de paquetes, tendremos que resolver a mano la dependencias y las incompatibilidades que pueda haber. Pero normalmente ser´a posible.

# Ejercicio 4 (2.5 puntos)

Un compan˜ero que no ha cursado la asignatura te dice lo siguiente:

*No se si entiendo bien el protocolo SNMP. Veo que iso.org.dod.internet.mgmt.mib- 2.system.sysUpTime es una representaci´on textual que se corresponde con la represen- taci´on num´erica .1.3.6.1.2.1.1.3. Esto es muy parecido a los nombres de dominio y las*

*direcciones IP, esto es, la representaci´on en modo texto es una direcci´on web donde yo puedo consultar el valor correspondiente, en este caso el tiempo que lleva una m´aquina encendida. Y los nu´meros ser´ıan la direcci´on IP que me ofrece ese servicio. Pero la se- cuencia de nu´meros no tiene pinta de direcci´on IPv4. Parece una una direcci´on IPv6, as´ı que supongo que este es un ejemplo de SNMP funcionando sobre IPv6*

¿Qu´e le respondes?

### Respuesta

Es verdad que la idea se parece un poco. Pero esa representaci´on num´erica no tiene nada que ver con una direcci´on IP, ni v4 ni v6. Son dos representaciones del mismo objeto, pero no son direcciones de hosts.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 9 de enero de 2015.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagrs y comprueba que esto ha creado el direc- torio

~/lagrs.enero.15,

y dentro, los ficheros parte.TULOGIN.py y practico.TULOGIN.txt,

donde contestar´as las preguntas. (La cadena TULOGIN representa tu nombre de usuario en el laboratorio)

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo u´nico que importa es lo que escribas en el examen, describiendo qu´e has hecho. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en el examen**

# Ejercicio 1 (4 puntos)

Edita el script con el nombre ~/lagrs.enero.15/parte.TULOGIN.py

de forma que muestre por salida est´andar la direcci´on IP y la direcci´on MAC de la tarjeta de red de cada m´aquina que est´e funcionando en el laboratorio en que trabajas ahora. Para averiguar la MAC, usa la orden ifconfig

El script debe funcionar aunque haya m´aquinas apagadas. El informe estar´a ordenado alfab´eticamente y tendr´a un aspecto similar a este:

|  |  |  |
| --- | --- | --- |
| theta01 | 193.147.49.143 | 5c:f9:dd:77:c5:2a |
| theta02 | 193.147.49.144 | 5c:f9:dd:77:1a:c4 |
| theta04 | 193.147.49.146 | 5c:f9:dd:77:ba:b4 |
| theta05 | 193.147.49.147 | 5c:f9:dd:77:e1:c2 |

Observaciones:

En el informe, los nombres de m´aquinas deben aparecer sin dominio

Es recomendable que reutilices c´odigo de tus pr´acticas

En el caso de sistemas configurados en ingl´es, la orden ifconfig muestra la MAC tras la cadena HWaddr. En espan˜ol, este campo est´a identificado como direcci´onHW.

# Ejercicio 2 (2 punto)

1. Haz que cada vez que se encianda la m´aquina virtual pc01, env´ıe un paquete de ping a 193.147.71.64 y escriba en /var/tmp/log la salida est´andar de la orden (de forma que se conserven todas las salidas y no solo la u´ltima)

Hazlo de la forma m´as sencilla posible, sin respetar el est´andar de los niveles de ejecuci´on de System V

1. Describe br´evemente qu´e habr´ıa que hacer para cumplir el est´andar

Observa que en el apartado 1 tienes que dar una respuesta completa de todas las o´rdenes necesarias. En el 2, basta con que indiques qu´e ficheros utilizar, con qu´e nombre y en qu´e directorio, con una descripci´on de su comportamiento. Pero sin detallar su contenido.

Recuerda que no importa lo que hagas en el laboratorio, solo lo que escribas en el examen.

### Respuesta

1. Cualquier fichero cuyo nombre empiece por S y que est´e en el directorio /etc/rc2.d

se ejecutar´a al iniciar la m´aquina

Editamos un fichero llamado por ejemplo /etc/rc2.d/Sping y cuyo contenido sea

ping -c1 193.147.71.64 >> /var/tmp/log

Le damos permiso de ejecuci´on

chmod+x /etc/rc2.d/Sping

Esto basta para que se ejecuta cada vez que se reinicie el sistema, aunque siempre ser´ıa mejor an˜adir al principio #!/bin/bash

1. Para seguir el est´andar, el fichero deber´ıa llamarse /etc/rc2.d/SNNping siendo NN un nu´mero entre 00 y 99

Ademas, este fichero no deber´ıa tener directamente el contenido del script, deber´ıa ser un enlace simb´olico a ../init.d/ping

A su vez, el fichero /etc/init.d/ping deber´ıa ser un script que aceptase, al menos, los par´ametros START, STOP y RELOAD, y que llamase al verdadero fichero con el c´odigo del demonio, p.e. /usr/local/pingd

Obs´ervese que en ningu´n caso editamos bash\_profile, bashrc o similares porque la especificaci´on se refiere al inicio de la m´aquina, no al inicio de la sesi´on de ningu´n usuario.

# Ejercicio 3 (2 puntos)

Usando screen

1. Entra por ssh en otra m´aquina de este laboratorio
2. Lanza la orden top y d´ejala corriendo
3. Cierra la conexi´on con la m´aquina remota, sin que se interrumpa el top
4. Vuelve a entrar en la m´aquina remota y recupera la sesi´on

Recuerda que no importa lo que hagas en el laboratorio, solo lo que escribas en el examen.

### Respuesta

ssh theta01 screen

top

(creamos una nueva ventana con ctrl a c)

screen -d # nos desasociamos. Esto ya se ejecuta en la nueva ventana screen -ls # paso opcional, para comprobar que estamos desasociados exit

ssh theta01 # nos reconectamos

screen -ls # vemos el n´umero de sesi´on, p.e. 1234 screen -r 1234

(cambiamos a la ventana de top con ctrl a, los cursores e intro)

# Ejercicio 4 (2 puntos)

1. Crea un repositorio git llamado

~/lagrs.enero.15/mirepo

1. Mete dentro el fichero ~/lagrs.enero.15/mirepo/holamundo.txt cuyo contenido ser´a el texto *feliz 2014*. Etiqueta esto como *primer commit*
2. Haz que el fichero contenga el texto *feliz an˜o*. Etiqueta esto como *segundo commit*
3. Haz todo lo necesario para recuperar el fichero del *primer commit*
4. En el fichero del primer commit, reemplaza el texto por *feliz 2015* y haz un tercer commit etiquetado como *corrige la fecha del primer commit*
5. Muestra un log para comprobar que siguen siendo visibles los tres commits

Recuerda que no importa lo que hagas en el laboratorio, solo lo que escribas en el examen.

### Respuesta

mkdir mirepo cd mirepo git init

echo "feliz 2014">holamundo.txt git add holamundo.txt

git ca -m "primer commit"

echo "feliz a~no">holamundo.txt git ca -m "segundo commit"

git log

git co 9316a16bc1a # Vuelta al "presente" cp holamundo.txt ..

git co master

cp ../holamundo.txt .

echo "feliz 2015" > holamundo.txt

git ca -m "corrige la fecha del primer commit" git log

Hemos creado el repositorio, el primer commit y el segundo. Cuando volvemos al primer commit, copiamos el fichero en cualquier lugar, por ejemplo en el directorio padre. Tras esto, volvemos a la rama master con la o´rden git co master, lo que informalmente llamamos *volver al presente*. Si omitimos este paso, nos habremos quedado en la estado *detached head*, esto es, un estado sin rama. Y habremos perdido el segundo commit.

Tras recuperar el fichero, creamos el tercer commit y comprobamos que todo est´a bien.

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Prueba escrita sobre la teor´ıa. 9 de enero de 2015

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

# Ejercicio 1 (2.5 puntos)

La transparencia 10 del tema 7 dice: *En Unix y Linux se usan mucho otras herra- mientas de monitorizaci´on como Nagios y Zabbix, que ofrecen sus propios servicios pero tambi´en incluyen SNMP*

1. Explica este p´arrafo. ¿por qu´e se usan otras herramientas? ¿qu´e significa que ofrez- can sus servicios? ¿qu´e significa que incluyan SNMP?
2. ¿Va esto contra el est´andar?
3. Ventajas e inconvenientes de SNMP frente a herramientas como Nagios y Zabbix.

¿Cu´ando usar uno u otro?

### Respuesta

1. SNMP es un protocolo que no ha tenido demasiado ´exito, tiene muchas limitaciones. Como todo protocolo, pueden usarse desde muchas herramientas, como Nagios, Zabbix y similares. Por ejemplo, podemos tener una impresora que exporte su estado mediante SNMP, y monitorizarlo desde Zabbix.

Pero estas herramientas no solo soportan el protocolo est´andar de SNMP, sino que tambi´en ofrecen sus propios agentes, que usan su propio protocolo, m´as potente y flexible.

Si no hablamos de un host sencillo como una impresora, sino de otro m´as complejo, como un ordenador, tambi´en podremos exportar su estado mediante un agente (cliente) SNMP. Pero esto no es muy habitual, posiblemente preferiremos emplear el cliente espec´ıfico de nuestra herramienta de monitorizaci´on.

1. Si solo usamos agentes SNMP, estamos siguiendo el est´andar. Pero si se usan agentes particulares (cosa muy frecuente), s´ı, esto va contra el est´andar.
2. SNMP es universal, vamos a encontrar un agente SNMP en cualquier dispositivo con un m´ınimo de calidad y capaz de conectarse a Internet. Otras herramientas m´as avanzadas usan sus propios protocolos no universales. T´ıpicamente se usa SNMP cuando no hay otra opci´on disponible, por ejemplo impresoras, switches, c´amaras IP, dispositivos dom´oticos, etc. En estos dispositivos no vamos a encontrar sopor- te nativo para, por ejemplo, Zabbix. Cuando se trate de monitorizar ordenadores convencionales, s´ı podremos usar agentes propios de nuestra herramienta.

# Ejercicio 2 (2.5 puntos)

Un servidor de DHCP puede estar configurado en modo *authoritative* o en modo *non authoritative*

1. ¿Qu´e significa esto?
2. ¿Qui´en decide que un servidor est´e en un modo u otro?
3. ¿Qu´e garant´ıas tiene un cliente de que un servidor *authoritative* lo es realmente?

¿C´omo evitar que un servidor *non authoritative* conteste como *authoritative*?

1. ¿Qu´e problemas puede causar un servidor de DHCP que tenga este par´ametro mal configurado?

Observaci´on: La transparencia 8 del tema 6 explica el comportamiento de un servidor *authoritative* y *non authoritative* ante un DHCPREQUEST. Pero **no** es eso lo que debes contestar. C´ın˜ete al enunciado del ejercicio.

### Respuesta

1. Una respuesta con el flag *authoritative* activado significa que el servidor declara que su respuesta es *oficial*, el servidor est´a indicando que ´el es quien tiene autoridad en ese segmento de red.

Una respuesta *non authoritative*, el servidor no asegura ser la autoridad oficial.

Cuando un cliente hace una petici´on *DISCOVERY*, si hay varios servidores recibir´a varias respuestas *OFFER*, la norma permite que el cliente elija la respuesta m´as adecuada, segu´n su criterio. (Deber´ıa preferir una con autoridad frente a una sin autoridad, aunque muchas implementaciones se quedan con la primera en llegar)

Cuando un cliente pide una direcci´on y al servidor le parece incorrecta, si tiene autoridad se *atreve* a contradecirle. Si el servidor no tiene autoridad, no responde.

1. El administrador de la m´aquina donde est´a el servidor de DHCP. Deber´ıa ser el administrador de la red, pero no tiene por qu´e serlo. Es imposible evitar que un servidor se atribuya la autoridad que no tiene.
2. Un servidor de DHCP mal configurado puede ser muy problem´atico. Supongamos un atacante que instala un servidor en su port´atil y lo lleva a una red ajena. El servidor puede empezar a repartir direcciones incorrectas, puede inutilizar la red con par´ametros incorrectos, o peor au´n, *infectar* la red proporcionando direcciones de servidores con malware.

Si las repuestas del atacante son *non authoritative* y el servidor de la red v´ıctima tambi´en env´ıa respuestas *non authoritative*, los clientes podr´ıan usar indistinta- mente una u otro. Al igual que el caso en que atacante y v´ıctima declaran ser *authoritative*.

El peor caso es el de una red v´ıctima donde el servidor tenga la configuraci´on por omisi´on (*non authoritative*), y el atacante est´e dando respuestas (*authoritative*). Las v´ıctimas preferir´an al atacante.

El caso menos dan˜ino es el de un atacante, posiblemente involuntario, que tenga un servidor por omisi´on (*non authoritative*). Si el administrador de la red ha teni- do la precauci´on de configurar el servidor oficial como (*authoritative*), los clientes preferir´an el oficial.

Esto es lo previsto en el est´andar, pero como hemos dicho, muchas implementaciones atienden la primera respuesta recibida, sea cual sea, por lo que aceptar la respuesta correcta o la del atacante acaba resultando aleatorio.

# Ejercicio 3 (2.5 puntos)

Probablemente conoces la *flamenca del whatsapp*, un *emoji* (icono) que representa una mujer vestida de rojo, bailando.

¿Ser´ıa posible escribir en Linux un texto que la incluya? ¿Qu´e har´ıa falta?

### Respuesta

Unicode es un est´andar de codificaci´on de caracteres. Es muy completo, incluye no solo todos los caracteres de pr´acticamente cualquier lenguaje humano, sino tambi´en iconos. Por ejemplo, desde la versi´on 6.0 (an˜o 2010), incluye la *flamenca*, que es el car´acter ’DANCER’ (U+1F483)

Toda herramienta que soporte unicode nos permitir´a manejar este y cualquier otro car´acter, aunque para poder visualizarlo, ser´a necesario tener instalado el *font* adecuado. Los fonts pueden cambiar de aspecto, por ejemplo este icono en algunos fonts representa un hombre de color amarillo.

# Ejercicio 4 (2.5 puntos)

Un script de shell bash normalmente empieza por #!/bin/bash

1. ¿Por qu´e?
2. ¿Qu´e sucede en caso contrario?
3. ¿Qu´e sucede con el intento de ejecuci´on de un script sin permiso de ejecuci´on?

### Respuesta

1. Para indicar expl´ıcitamente donde est´a el ejecutable que sabe interpretar ese texto. La secuencia almohadilla-admiraci´on es un *nu´mero m´agico*, un convenio que indica que si estos son los primeros caracteres de un fichero, lo que aparece a continuaci´on es el path del int´erprete.
2. Que el sistema no sabr´ıa cu´al es el int´erprete, intentar´ıa ejecutarlo con el int´erprete por omisi´on que haya especificado el usuario. Si ambos coinciden, todo funciona bien. En otro caso, para ejecutarlo ser´ıa necesario invocar al int´erprete (bash) y luego pasarle el script, bien por la entrada est´andar, bien como primer argumento.
3. Depende. Si se intenta ejecuta el script directamente, no funcionar´a. Pero si se invoca a la shell y se le pasa el script, se ejecutar´a normalmente.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 24 de junio de 2015.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagrs y comprueba que esto ha creado el direc- torio

~/lagrs.junio.15,

y dentro, los ficheros claves.tgz, info\_memoria.TULOGIN.py y practico.TULOGIN.txt, donde contestar´as las preguntas. (La cadena TULOGIN representa tu nombre de

usuario en el laboratorio)

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo u´nico que importa es lo que escribas en el examen, describiendo qu´e has hecho. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en el examen**

# Ejercicio 1 (4 puntos)

Tomando como punto de partida tu pr´actica 3.4, escribe el script

~/lagrs.junio.15/info\_memoria.TULOGIN.py que devolver´a el total de memoria instalada (en gigas) y el porcentaje de memoria libre en cada ordenador del laboratorio.

Extrae la informaci´on con la orden free -m

# Ejercicio 2 (3 puntos)

En la pr´actica 1.16 configuraste tu cuenta para poder entrar desde una m´aquina del laboratorio hasta otra sin teclear contrasen˜as.

1. Asegu´rate de que esto sigue funcionando correctamente
2. Entra por ssh con el usuario *examen1* en la m´aquina 193.147.71.62. Lo necesario para hacer esto lo encontrar´as en ~/lagrs.junio.15/claves.tgz (que tambi´en contiene ficheros sin ninguna utilidad)
3. Copia el fichero ~/prueba.txt que encontrar´as en la m´aquina del apartado 2.2 en el directorio ~/lagrs.junio.15/prueba.txt de tu host
4. Averigua qu´e codificaci´on emplea
5. Convi´ertelo a la codificacion por omisi´on empleada en tu host
6. Asegu´rate de que lo que hiciste en el apartado 2.1 sigue funcionando

# Ejercicio 3 (3 puntos)

En el directorio /home/examen2/mail de la m´aquina 193.147.71.62 hay un repositorio git. Es accesible para el usuario examen2, de contrasen˜a lagrs.junio.15

1. Clona este repositorio en el directorio ~/lagrs.junio.15/git de tu host
2. En principio no habr´a ningu´n fichero en el directorio de trabajo, pero en la revisi´on etiquetada como *corrige correo*, econtrar´as un script llamado mail\_amazon.py

Recup´eralo

1. Observa que en este script, parece que se declaran las variables smpt\_username y smpt\_passord , pero falta el contenido de las variables, no hay nada a la derecha del igual
2. Podr´as encontrar estos valores en la revisi´on etiquetada como *versi´on inicial*
3. Vuelve a la rama m´aster
4. Prepara una versi´on personalizada y completa del script, esto es, rellenando lo que falta en la versi´on de la etapa 3.3. con los datos de la versi´on 3.4
5. An˜ade tu nombre y login en la primera l´ınea del script. An˜ade al repositorio un fichero llamado ~/leeme.txt (cuyo contenido puede ser *blablabl´a*)
6. Crea en tu repositorio local una nueva revisi´on con todo lo que acabas de hacer, etiquetada como version\_jperez (donde jperez ser´ıa tu verdadero login)

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Prueba escrita sobre la teor´ıa. 24 de junio de 2015

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

# Ejercicio 1 (2.5 puntos)

Explica qu´e sucede al ejecutar en la shell las siguientes o´rdenes. Cuenta no solo qu´e sucede, sino c´omo sucede, esto es, c´omo funcionan en cada ejemplo los argumentos, las redirecciones, entrada est´andar, salida est´andar, salida de error, pipes, etc

1. cat
2. cat file1 file2 > file3
3. cat file1 | less
4. cat > file1

# Ejercicio 2 (2.5 puntos)

Supongamos que en nuestro sistema linux hemos montado el directorio raiz en /dev/sda1

1. Explica qu´e significa esto
2. Si ejecutamos ls /dev/sda1 no vemos los ficheros que contiene /. Si ejectutamos cd /dev/sda1 se produce un error

¿Por qu´e?

# Ejercicio 3 (2.5 puntos)

Alguien te dice *en mi empresa tenemos un mont´on de impresoras de diferentes marcas y queremos ahorrar t´oner, que las impresoras siempre est´en en modo ahorro, a menos que un usuario realmente necesite una calidad m´as alta. Pero que si el usuario que us´o la calidad alta deja esta configuraci´on en el panel de la impresora, algu´n sistema lo vuelva a configurar en modo ahorro. ¿Se puede hacer esto mediante SNMP? ¿C´omo?*

¿Qu´e respondes?

# Ejercicio 4 (2.5 puntos)

La transparencia 7 del tema 6 dice *El cliente puede renunciar a su lease anticipada- mente con un DHCPRELEASE*

¿Qu´e significa esto? ¿Por qu´e motivo el cliente har´ıa tal cosa?

### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 18 de diciembre de 2015.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagrs y comprueba que esto ha creado el direc- torio

~/lagrs.enero.16,

y dentro, los ficheros claves.tgz, info\_memoria.TULOGIN.py y practico.TULOGIN.txt, donde contestar´as las preguntas. (La cadena TULOGIN representa tu nombre de

usuario en el laboratorio)

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo u´nico que importa es lo que escribas en el examen, describiendo qu´e has hecho. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en el examen**

# Ejercicio 1 (3 puntos)

Tomando como punto de partida tu pr´acticas 3.4 y 3.5, escribe el script

~/lagrs.enero.16/info\_disco.TULOGIN.py

Que a partir de la orden df, mostrar´a, para cada m´aquina del laboratorio que ocupas actualmente, el porcentaje de uso de cada sistema de ficheros y la suma total.

De forma similar a esta:

delta02 15+0+1+1+0+1+1+21+1+20=61

delta04 19+0+2+1+0+4+1+21+8+20=76

delta05 ... [etc]

El programa debe tolerar que alguna m´aquina no responda correctamente.

# Ejercicio 2 (1.5 puntos)

Edita un fichero de texto llamado

~/lagrs.enero.16/origenes.txt

En su interior, escribe un texto que contenga en cada l´ınea un nombre de fichero, incluyendo su path completo. Por ejemplo esto:

/tmp/1

/tmp/2

/tmp/3

1. Empleando las o´rdenes xargs y touch, haz que si no existen los ficheros indicados en el fichero de texto origenes.txt, se creen. (Si ya existen no importa lo que suceda con ellos)
2. Empleando las o´rdenes xargs y touch, haz que en el directorio de trabajo actual se cree un enlace simb´olico a cada uno de los ficheros indicados en origenes.txt

### respuesta

1. cat origenes.txt|xargs touch
2. cat origenes.txt |xargs -I{} ln -s {} .

# Ejercicio 3 (2.5 puntos)

1. Crea un directorio llamado ~/lagrs.enero.16/base
2. Escribe los ficheros

~/lagrs.enero.16/base/f1.txt

~/lagrs.enero.16/base/f2.txt

Contendr´an un texto de ejemplo cualquiera, como *este es el fichero f1*

1. Haz todo lo necesario para que /tmp/r1 y /tmp/r2 y sean repositorios cuyo contenido sea el mismo que base
2. An˜ade un fichero llamado f3.txt al repositorio r1
3. Propaga este cambio a r2 Observaciones:

Es necesario que respetes las convenciones habituales en git

Recuerda que no importa lo que hagas en el ordenador del laboratorio, sino lo que escribas en el examen

### Respuesta

cd # vamos al directorio home mkdir lagrs.enero.16/base

cd lagrs.enero.16/base

echo "este es el fichero f1" > f1 echo "este es el fichero f2" > f2

git init # iniciamos el repositorio abuelo git add f1 f2

git ca -m "commit inicial" cd ..

mkdir padre.git

git clone --bare base padre.git # clonamos el abuelo al padre cd /tmp

mkdir r1 r2

git clone ~/lagrs.enero.16/padre.git r1 # clonamos el hijo r1 git clone ~/lagrs.enero.16/padre.git r2 # clonamos el hijo r2 cd r1

touch f3 git add f3

git ca -m "a~nade fichero f3"

git pull # este paso no hace falta, pero es buen h´abito git push # propagamos cambios del nieto al padre

cd ../r2

git pull # propagamos cambios del padre al nieto git push # este paso no hace falta

# Ejercicio 4 (2 puntos)

1. Sea TUHOST la m´aquina del laboratorio que est´as usando. Sea OTROHOST una m´aquina de laboratorio que est´e vac´ıa, funcionando y justo a tu derecha. (si no es posible, dile al profesor que te indique cu´al es OTROHOST). Indica el nombre de TUHOST y OTROHOST, para que conste en el examen
2. Haz todo lo necesario para que cuando lances romanclient.py contra el puerto 7777 de TUHOST, responda el romanserver.py del puerto 9999 de OTROHOST

### Respuesta

# Establecemos el t´unel

jperez@TUHOST ~/lagrs.enero.16 $ ssh -L 7777:OTROHOST:9999 jperez@OTROHOST

# Desde otra terminal entramos en otrohost y lanzamos romanserver jperez@TUHOST ~/lagrs.enero.16 $ ssh OTROHOST

jperez@OTROHOST ~ $ cd bin/

jperez@OTROHOST ~/bin $ ./romanserver.py TCP 9999

# Desde otra terminal lanzamos romancliente contra el puerto local jperez@TUHOST ~ $ cd bin/

jperez@TUHOST ~/bin $ romanclient.py localhost TCP 7777 23 XXIII

# Ejercicio 5 (1 puntos)

1. Indica qu´e es necesario para que tu script aguad de la pr´actica 5.3 se ejecute los lunes de septiembre, octubre y noviembre, cada 5 minutos
2. Indica qu´e es necesario para que ese mismo script se ejecute a las 9 de la man˜ana, de lunes a viernes

### Respuesta

Con la orden crontab -e creamos las siguientes tablas:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| #m | h | dom | mon | dow | command |
| \*/5 | \* | \* | 9-11 | 1 | /home/al-12-13/jperez/lagrs/pc01/aguad |
| #m | h | dom | mon | dow | command |
| 0 | 9 | \* | \* | 1-5 | /home/al-12-13/jperez/lagrs/pc01/aguad |

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Prueba escrita sobre la teor´ıa. 18 de diciembre de 2015

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

# Ejercicio 1 (2.5 puntos)

Un sistema operativo de software libre como Linux ¿se puede vender?

### Respuesta

S´ı, no es lo habitual pero es posible y se da en algunos casos. El software libre se caracteriza porque quien lo recibe tiene las *cuatro libertades*:

1. Usarlo para cualquier fin
2. Analizarlo y modificarlo
3. Distribuirlo tal y como est´a
4. Distribuir las modificaciones

Para esto, es necesario que quien lo reciba disponga del c´odigo fuente. Quien recibe el software puede venderlo, pero no puede evitar que el comprador pueda volver a revenderlo.

# Ejercicio 2 (2.5 puntos)

¿Cu´al es la diferencia entre kill <pid> y kill -9 <pid> ?

¿Cu´ando debe usarse uno y cu´ando debe usarse el otro?

### Respuesta

kill <pid>

Env´ıa al proceso <pid> la sen˜al por omisi´on, SIGTERM, que solicita al proceso que concluya. Normalmente el proceso aceptar´a la orden y concluir´a de forma ordenada, aunque tambi´en puede ignorar la orden, por haber sido programado as´ı o por haberse colgado.

kill -9 <pid>

Env´ıa al proceso la sen˜al SIGKILL. En caso de que el emisor sea el duen˜o del proceso o el usuario root, esto causa que el proceso concluye de manera brusca, e inevitable.

Lo correcto es en primer lugar intentar finalizar un proceso con SIGTERM, y si falla, forzarlo con SIGKILL.

# Ejercicio 3 (2.5 puntos)

Explica brevemente la diferencia entre trabajar sobre un VPS, *virtual private server*

y trabajar sobre un PaaS, *platform as a service*.

(Como seguramente sabr´as, ejemplos de PaaS son Heroku, Google App Engine y Microsoft Azure Web Sites. Un ejemplo de VPS es una instancia de EC2 de Amazon, esto es, una m´aquina en la *nube de amazon*)

### Respuesta

Un VPS es una m´aquina virtual, completa, que un proveedor de *hosting* alquila a un usuario. Se administra pr´acticamente igual que una m´aquina f´ısica, con la u´nica diferencia de que el hardware est´a virtualizado y controlado por el proveedor. En esta m´aquina, se puede ejecutar cualquier combinaci´on de servicios.

Un PaaS es un tipo de computaci´on en la nube. Un proveedor ofrece a sus clientes una plataforma para usar un servicio concreto, como pueda ser un servidor web, una base de datos, almacenamiento, etc. La administraci´on de la m´aquina es responsabilidad del proveedor, el usuario solo usa y configura el servicio que haya contratado.

# Ejercicio 4 (2.5 puntos)

Explica con tus palabras las diferencias y similitudes entre *unicode* y *UTF-8*.

### Respuesta

Unicode es una norma definida en 1991 que enumera la pr´actica totalidad de los caracteres de los lenguajes humanos, naturales o artificiales, adem´as de otros s´ımbolos como los emoticonos. Informalmente podr´ıamos decir que es un conjunto de tablas que asocia a cada s´ımbolo un nu´mero.

UTF-8 es una de las formas posibles de codificar ese nu´mero. Detalla qu´e secuencia de bytes en concreto deben emplearse para indicar un s´ımbolo unicode. Es relativamente reciente, del an˜o 2006, pero se ha convertido en la codificaci´on UTF-8 recomendada en

Unix y la m´as popular en internet. Microsoft Windows no usa UTF-8 de forma nativa pero muchas herramientas s´ı lo soportan.

Cualquier s´ımbolo unicode puede codificarse en UTF-8, si bien es posible que en una plataforma concreta no pueda representarse gr´aficamente de forma correcta, porque no est´e disponible la fuente tipogr´afica adecuada.

### Laboratorio de administración y gestión de redes y sistemas Prueba escrita - 15 de enero de 2017

Grado en Ingeniería Telemática, Universidad Rey Juan Carlos

Instrucciones:

Crea el directorio

~/lagrs.enero.17

y dentro de él, el fichero

~/lagrs.enero.17/respuestas.TULOGIN.txt

donde contestarás las preguntas. (La cadena TULOGIN representa tu nombre de usuario en el laboratorio, es decir, si tu login es pepito, el fichero se llamará ejercicio.pepito.txt).

El enunciado te pedirá que hagas una serie de cosas en tu ordenador, es conveniente que lo hagas realmente, aunque en realidad lo único que importa es lo que escribas en el examen, describiendo qué has hecho. Dicho de otro modo: el ordenador te servirá para comprobar que lo estás haciendo bien, pero es irrelevante lo que hagas en la shell, lo único que importa es lo que escribas en el examen.

## Ejercicio 1 (1 punto)

Indica todos los pasos necesarios para montar por sshfs el directorio

/var/tmp/

de la máquina del laboratorio cuyo número es uno más que el tuyo, en el directorio

/var/tmp/remota

de tu máquina.

Ejemplo: si tu máquina es kappa07, monta el directorio /var/tmp/ de kappa08 en el directorio /var/tmp/remota de kappa07

Recuerda que si haces un paso, pero no cuentas en el examen que lo has hecho, es como si no lo hubieras hecho. Pega el resultado de la ejecución, de forma similar a la memoria de prácticas.

## Ejercicio 2 (1.5 puntos)

Edita un fichero de texto llamado

~/lagrs.enero.17/origenes.txt

En su interior, escribe un texto que contenga en cada línea un nombre de fichero, incluyendo su path completo. Por ejemplo esto:

/tmp/1

/tmp/2

/tmp/3

1. Empleando las órdenes xargs y touch, haz que si no existen los ficheros indicados en el fichero de texto origenes.txt, se creen. (Si ya existen no importa lo que suceda con ellos)
2. Empleando las órdenes xargs y touch, haz que en el directorio de trabajo actual se cree un enlace simbólico a cada uno de los ficheros indicados en origenes.txt

## Ejercicio 3 (2.5 puntos)

Crea un directorio llamado

~/lagrs.enero.17/base

Crea los ficheros

~/lagrs.enero.17/base/f1.txt

~/lagrs.enero.17/base/f2.txt

de forma que contengan un texto de ejemplo cualquiera, como *este es el ﬁchero f1*

1. Haz todo lo necesario para que /tmp/r1 y /tmp/r2 y

sean repositorios git cuyo contenido sea el mismo que base

1. Añade un fichero llamado f3.txt al repositorio r1
2. Propaga este cambio a r2

Observaciones:

Es necesario que respetes las convenciones habituales en git

Recuerda que no importa lo que hagas en el ordenador del laboratorio, sino lo que escribas en el examen

## Ejercicio 4 (2 puntos)

1. Sea TUHOST la máquina del laboratorio que estás usando. Sea OTROHOST una máquina de laboratorio que esté vacía, funcionando y justo a tu derecha. (si no es posible, dile al profesor que te indique cuál es OTROHOST). Indica el nombre de TUHOST y OTROHOST, para que

conste en el examen

1. Haz todo lo necesario para que cuando lances romanclient.py contra el puerto 7777 de TUHOST, responda el romanserver.py del puerto 9999 de OTROHOST

## Ejercicio 5 (1 puntos)

1. Indica qué es necesario para que tu script aguad de la práctica 5.3 se ejecute los lunes de septiembre, octubre y noviembre, cada 5 minutos
2. Indica qué es necesario para que ese mismo script se ejecute a las 9 de la mañana, de lunes a viernes

## Ejercicio 6 (2 puntos)

Usando screen

1. Entra por ssh en otra máquina de este laboratorio
2. Lanza la orden top y déjala corriendo
3. Cierra la conexión con la máquina remota, sin que se interrumpa el top
4. Vuelve a entrar en la máquina remota y recupera la sesión

Recuerda que no importa lo que hagas en el laboratorio, solo lo que escribas en el examen.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Prueba escrita sobre las pr´acticas. 20 de diciembre de 2017.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Ejecuta ~mortuno/prepara\_examen\_lagrs y comprueba que esto ha creado los siguientes ficheros

\verb|~/lagrs.diciembre.17/practico.jperez.txt|

\verb|~/lagrs.diciembre.17/contenedores\_de.py|

\verb|~/lagrs.diciembre.17/cex/construye.sh|

\verb|~/lagrs.diciembre.17/cex/lanza\_jpercex01.sh|

\verb|~/lagrs.diciembre.17/cex/context/Dockerfile|

\verb|~/lagrs.diciembre.17/cex/context/entrypoint.sh|

\verb|~/lagrs.diciembre.17/cex/context/holamundo|

Como siempre, la cadena *jperez* representa tu nombre de usuario en el laboratorio y *jper*, las primeras 4 letras de tu nombre de usuario.

Todos los ficheros estar´an vac´ıos, excepto el script python, que contiene una funci´on auxiliar.

El enunciado te pedir´a que hagas una serie de cosas en tu ordenador, es conveniente

que lo hagas realmente, aunque lo u´nico que importa es lo que escribas en los

diferentes ficheros que indica el enunciado. Dicho de otro modo: el ordenador te servir´a para comprobar que lo est´as haciendo bien, pero **es irrelevante lo que hagas en la shell, lo u´nico que importa es lo que escribas en los ficheros del examen**.

# Ejercicio 1 (3 puntos)

En tu cuenta encontrar´as un fragmento de script, con el siguiente nombre

~/lagrs.diciembre.17/contenedores\_de.py

Completa el script, de forma que:

Reciba un argumento por linea de comandos, que ser´a un nombre de usuario. Basandose en la orden docker images, mostrar´a todas las im´agenes de docker disponibles en el sistema, cuyo prefijo sea esa nombre de usuario. Tambi´en sumar´a el total del taman˜o de esas im´agenes.

Ejemplo:

mgarcia@alpha:~/lagrs.diciembre.17$ ./contenedores\_de.py jperez

Contenedores con el prefijo jperez: caa, cbb, cac

Tama~no total: 694 MB

Observaciones

El nombre de usuario que se debe mostrar es el que se pasa por linea de comandos, no tu nombre de usuario ni el nombre del usuario que invoca el script.

En el listado de im´agenes que mostrar´a tu script, se omite el prefijo con el nombre de usuario. El nombre de usuario solo aparece 1 vez, al principio del informe.

Dicho de otro modo: en el listado no debe decir *jperez/caa, jperez/cbb* sino *caa, cbb*. Tal y como se ve en el ejemplo.

Para obtener los argumentos de l´ınea de comandos, puedes usar sys.argv[], no hace falta que uses la librer´ıa optparse.

El script deber recibir un argumento, exactamente uno. En otro caso, el script debe mostrar un mensaje de error y finalizar sin hacer nada m´as. Si quieres, puedes levantar una excepci´on.

En la salida de docker images (que es la entrada de tu script), el taman˜o de cada imagen podr´a estar expresado en kB, MB o GB. Pero la salidad de tu script siempre tendr´a que estar expresada en MB. Para facilitarte esta parte, dentro del script ya tienes programada una funci´on auxiliar, any\_size\_to\_mb(), que recibe el taman˜o expresado en cualquiera de las tres unidades y devuelve el taman˜o expresado en MB. En el fuente del script tienes los detalles.

En caso de que no haya ninguna imagen con ese nombre de usuario como prefijo, tu programa puede mostrar lo que quieras: un mensaje especial, una salida vac´ıa, un taman˜o de 0, un taman˜o en blanco, etc. Lo u´nico que no es admisible es que el programa falle, muestre algo sin sentido o una excepci´on.

Cuando lo tengas funcionando, ens´en˜aselo al profesor.

# Ejercicio 2 (4 puntos)

En este ejercicio preparar´as una imagen de un contenedor segu´n la siguiente especifi- caci´on:

* 1. Una vez lanzando, tendr´a corriendo un servidor de ssh.
  2. Tendr´a creado un usuario llamado *alumno*.
  3. Ese usuario tendr´a asignada la contrasen˜a *xx99*. Esto no lo has hecho en ninguna pr´actica, pero la transparencia 17 del tema de administraci´on de usuarios indica c´omo cambiar una contrasen˜a desde un script.
  4. Cuando el alumno entre en el contenedor, podr´a ejecutar un script llamado

/usr/local/bin/holamundo

que mostrar´a por la salida est´andar un mensaje similar a "¡Hola, mundo!"

* 1. La imagen se llamar´a jpercex, donde jper representa las primeras 4 letras de tu nombre de usuario en el laboratorio.
  2. El contenedor se llamar´a jpercex01
  3. Para preparar este contenedor, edita los ficheros

~/lagrs.diciembre.17/cex/construye.sh

~/lagrs.diciembre.17/cex/lanza\_jpercex01.sh

~/lagrs.diciembre.17/cex/context/Dockerfile

~/lagrs.diciembre.17/cex/context/entrypoint.sh

~/lagrs.diciembre.17/cex/context/holamundo

Como es habitual, *jper* representa las primeras 4 letras de tu nombre de usuario en el laboratorio.

* 1. Se valorar´a positivamente que instales solo los paquetes necesarios y que uses solo los ficheros necesarios. Esto es, si el ejercicio funciona, tendr´as buena nota. Pero para tener la m´axima nota tendr´as que instalar solo los ficheros y paquetes impres- cindibles (y no otros que hayan sido necesarios para otros ejercicios)
  2. Cuando lo tengas funcionando, ens´en˜aselo al profesor.

Sugerencia: para probar el servidor de ssh, entra por ssh desde el propio contenedor hasta el contenedor, usando la direcci´on *localhost*.

# Ejercicio 3 (3 puntos)

Asegu´rate de que est´as sentado en un puesto con nu´mero de m´aquina superior a 5.

Por tanto, no admite conexiones del exterior. Haz todo lo necesario para:

1. Lanzar un servidor de vnc en ese puesto. No elijas puerto, usa el que asigne el servidor. Pon el taman˜o del escritorio y la profundidad de color que prefieras.
2. Configurar un tu´nel inverso de SSH, usando el mismo proxy y el mismo puerto que en tu pr´actica de tu´nel inverso (no el puerto de tu pr´actica de vnc en dockerserver), para que un cliente de vnc (vinagre), pueda acceder a tu servidor de vnc, a trav´es de la direcci´on pu´blica del proxy.
3. Ejecutar el cliente vnc, en el puesto de tu derecha, de forma que se conecte al servidor de vnc.

Dibuja en papel un diagrama de tu configuraci´on. Escribe (en el fichero practico.TULOGIN.txt) los pasos que has seguido y cuando lo tengas funcionando, ens´en˜aselo todo al profesor.

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Prueba escrita sobre la teor´ıa. 20 de diciembre de 2017

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Encontrar´as en el *home* del puesto del laboratorio el fichero el fichero ~/teoria.tulogin.txt.

C´ambiale el nombre, reemplazando *tulogin* por tu verdadero login. Por ejemplo, si eres *jperez*, debes hacer

mv teoria.tulogin.txt teoria.jperez.txt

Revisa que est´e bien hecho. Si te equivocas en este paso tan sencillo, suspender´as.

Dentro del fichero teoria.jperez.txt, escribe tu login, nombre y apellidos y con- testa al examen.

Recuerda que si te limitas a repetir la informaci´on contenida en las transparencias, tu respuesta no tendr´a ningu´n valor

# Ejercicio 1

¿Que es VT-x? ¿En qu´e consiste?

### Respuesta

Es una tecnolog´ıa presente en los procesadores Intel desde hace unos 10 an˜os, que permite la virtualizaci´on nativa. Varios nu´cleos de varios sistemas operativos pueden correr sobre el mismo procesador, de forma que cada uno de ellos percibe estar en el *ring 0*, esto es, cada uno percibe un acceso exclusivo e ilimitado a la CPU, sin acceso a los recursos de otros nu´cleos.

En tecnolog´ıas anteriores es necesaria una capa de software adicional para distribuir la CPU entre los nu´cleos de cada sistema operativo.

# Ejercicio 2

Supongamos que estoy en una m´aquina Unix/Linux/MacOS y hago lo siguiente:

Copio un script holamundo.py en mi directorio ~/bin.

Le doy permisos de ejecuci´on para mi usuario.

Escribo en la terminal holamundo.py pero no se ejecuta.

Me doy cuenta de que mi varible no incluye ~/bin as´ı que escribo en el terminal

PATH=$PATH:$HOME/bin

Vuelvo a escribir holamundo.py

1. ¿Qu´e sucede? ¿Por qu´e?
2. Ahora abro un terminal nuevo, en otra ventana. ¿Qu´e sucede? ¿Por qu´e?

Observaciones

Contesta solamente las preguntas 1 y 2. Los puntos anteriores son el contexto del ejercicio, no se te pide que escribas esas o´rdenes.

### Respuesta

1. Hemos an˜adido a la variable de entorno PATH el directorio donde est´a el fichero. La shell buscar´a el fichero en todos los directorios indicados en PATH, por tanto ahora lo encontrar´a y ejecutar´a.
2. En un terminal nuevo la shell ser´a un proceso distinto, con sus propias variables de entorno. En esta shell, no hemos cambiado la variable PATH, as´ı que la shell no encontrar´a el script y no lo ejecutar´a.

Para ello deber´ıamos haber an˜adido la sentencia export PATH=$PATH:$HOME/bin al fichero que ejecuta cada shell antes de iniciarse. Lo m´as recomendable es el fichero ~/.bashrc, teniendo en cuenta que el fichero .bash\_profile deber´ıa llamar a ~/.bashrc, de lo contrario, el ~/.bashrc no ser´a invocado en las shell interactivas no de login, como por ejemplo un acceso por ssh.

# Ejercicio 3

¿Cu´al es la diferencia entre los siguientes directorios, cu´al es su prop´osito?

/lib

/usr/lib

/usr/local/lib

### Respuesta

/lib

Contiene librer´ıas esenciales para ejecutables del sistema.

/usr/lib

Contiene librer´ıas para ejecutables de menor importancia, no esenciales.

/usr/local/lib

Contiene librer´ıas de programas no incluidos de forma est´andar en la distribuci´on, sino que cada administrador local an˜ade.

# Ejercicio 4

Reproducimos aqu´ı una oferta real de empleo vista recientemente. A partir de tus conocimientos sobre el significado del t´ermino *DevOps*, com´entala. Si tienes alguna duda con el ingl´es puedes consultar al profesor.

DEVOPS ENGINEERS

Your activities will mainly focus on the design and development of a software application and/or embedded systems. It is crucial that you have a good overview of possible software design met- hodologies, operating systems and programming languages and that you are able to motivate this to others.

WE ARE LOOKING FOR

You have a B. Sc. or M.Sc. degree in Computer Sciences, Informatics or similar education with several years of experience in software. During your M.Sc. or through professional experience you have gained expertise in one or more of the following topics:

Work experience as System Engineer supporting web application architectures (Java, Tomcat, Apache, JBoss, Drools, etc)

*·*

Experience working in Agile projects applying methodologies (Jira, Confluence, BitBucket, Bam- boo etc)

*·*

* Continuous Integration tools (Maven, Jenkins, etc.)
* Experience with automation/configuration management using Puppet and/or other tools
* Virtualization technologies (VMware, Docker, AWS)
* Strong scripting skills in order to automate common tasks (bash, Unix tools)
* Knowledge of best practices and IT operations in an always-up, always-available service
* Experience developing logical model for BRMS
* Fluency in English and Spanish

### Respuesta

Segu´n todos los especialistas en *DevOps*, los libros, los congresos, etc, no tiene sentido hablar de *Ingeniero DevOps*. Por ejemplo en el libro *DevOps for developers* (Michael Huttermann, Ed, Apress) podemos leer expl´ıcitamente *DevOps is not a new job. If you see a job advertisement that asks for a DevOps expert, please point the author of the ad to this book*.

Pero en la industria inform´atica general, entre los no especialistas en *DevOps* es relativamente frecuente usar el t´ermino de forma distinta, como vemos por ejemplo en este anuncio. Se solicita un desarrollador con conocimientos en metodolog´ıas ´agiles, integraci´on continua, administraci´on con herramientas de configuraci´on de sistemas, etc, esto, es, herramientas que son habituales en *DevOps*. Pero ni son imprescindibles esas herramientas para hacer *DevOps* ni su uso implica estar siguiendo metodolog´ıas *DevOps*.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen de pr´acticas. 14 de diciembre de 2018.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Instrucciones

Abre una sesi´on en el laboratorio con tu usuario y contrasen˜a habituales. Ejecuta el script prepara\_lagrs

Esto crear´a en tu cuenta el directorio ~/lagrs.diciembre.18

y los ficheros ~/lagrs.diciembre.18/practico.tulogin.txt,

~/lagrs.diciembre.18/claves.tgz y

~/lagrs.diciembre.18/vigila\_puertos.py

# Ejercicio 1. (2 puntos)

En el fichero ~/lagrs.diciembre.18/claves.tgz

tienes todo lo necesario para abrir una sesi´on ssh en la m´aquina 193.147.79.2 con el usuario *alumno*. Tambi´en ficheros que *sobran*. Entra en esa m´aquina, escribe en practico.tulogin.txt los pasos que has seguido y muestra la sesi´on al profesor.

# Ejercicio 2. (4 puntos)

En este ejercicio escribir´as un script en python que, usando la order netstat, vigilar´a ciertos puertos y enviar´a un mensaje de telegram si algu´n proceso escucha peticiones en ellos. Edita el fichero ~/lagrs.diciembre.18/vigila\_puertos.py para que se corresponda con la siguiente especificaci´on:

1. El programa leer´a de un fichero de texto con nombre id\_usuario.txt el identificador de usuario de telegram que recibir´a las alarmas. No especifiques ningun *path*, de forma que se busque en el directorio actual.
2. El programa leer´a el token del bot de telegram en el fichero token.txt. De nuevo, no especifiques ningu´n trayecto, que el script lea el fichero desde el directorio actual.
3. En las primeras l´ıneas del script define una lista parecida a esta:

PUERTOS\_TCP = [6666, 443]

1. Cada vez que se ejecute el programa, revisar´a todos los puertos TCP especificados en PUERTOS TCP, y si alguno de ellos est´a ocupado, en cualquier estado (ya sea *ESCU- CHAR*, *ESTABLECIDO*, o cualquier otro en cualquier otro idioma: *LISTEN*, *ESTA- BLISHED*... ), enviar´a una alarma a trav´es de telegram describiendo toda la informaci´on relevante proporcionada por netstat.
2. La alarma solo se disparar´a si el puerto est´a ocupado en alguna direcci´on local, sea cual sea, pero no en ninguna direcci´on remota.
3. Usa romanserver.py para probar el programa. Cuando funcione, ens´en˜aselo al profesor.

# Ejercicio 3. (4 puntos)

En este ejercicio preparar´as y lanzar´as dos contenedores Docker, uno con *romanserver.py* y otro con *romanclient.py*, segu´n la siguiente especificaci´on:

1. Ambos contenedores estar´an conectados por un segmento privado de red dentro de Docker.
2. Como todos los contenedores del laboratorio comparten segmento de red virtual, cada alumno necesita su propio puerto. El profesor te indicar´a un *puerto alumno*, usa el puerto TCP 12000 + *puerto alumno*.
3. El contenedor con romanserver.py estar´a basado en una imagen llamada exa. Si tu nom- bre de usuario fuera *jperez*, el contenedor se llamar´ıa jperexa01. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.

Al ejecutarlo, lanzar´a el programa romanserver.py en el puerto reservado para t´ı y a continuaci´on abrir´a una shell.

1. El contenedor con romanclient.py estar´a basado en una imagen llamada exb.

Al ejecutarlo, lanzar´a un terminal, donde el usuario podr´a invocar a romanclient.py con los par´ametros adecuados. El usuario deber´a podar lanzar romanclient.py sin especificar ningu´n path.

1. El contenedor exa tambi´en tambi´en tendr´a disponible romanclient.py, para poder probar el servicio en local.
2. Ambos contenedores deber´an poderse hacer *ping* entre s´ı. Tambi´en podr´an usar *ifconfig*.
3. Usa el mismo convenio empleado en clase para nombrar los ficheros de configuraci´on de los contenedores, creaci´on de im´agenes y ejecuci´on. Aunque guardando todo en el directorio

~/lagrs.diciembre.18. Esto es, usa los siguientes ficheros:

~/lagrs.diciembre.18/exa/construye.sh

~/lagrs.diciembre.18/exa/lanza\_jperexa01.sh

~/lagrs.diciembre.18/exa/context/Dockerfile

~/lagrs.diciembre.18/exa/context/entrypoint.sh

~/lagrs.diciembre.18/exa/context/romanserver.py

~/lagrs.diciembre.18/exa/context/romanclient.py

Y para el cliente:

~/lagrs.diciembre.18/exb/construye.sh

~/lagrs.diciembre.18/exb/lanza\_jperexb01.sh

~/lagrs.diciembre.18/exb/context/Dockerfile

~/lagrs.diciembre.18/exb/context/entrypoint.sh

~/lagrs.diciembre.18/exb/context/romanclient.py

1. Dentro de los contenedores, los ficheros romanserver.py y romanclient.py deben estar en el sitio *correcto* segu´n la norma FHS.
2. Instala solo los paquetes necesarios, usa solo los ficheros necesarios (y no otros que hayan hecho falta en las pr´acticas).
3. Cuando lo tengas funcionando, ens´en˜aselo al profesor.

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Examen de teor´ıa. 14 de diciembre de 2018

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

### Instrucciones:

Entra en el puesto del laboratorio con el nombre de usuario *examen* y la contrasen˜a

*examen*.

Ejecuta el script prepara\_lagrs

Esto dejar´a en el ordenador el fichero ~/lagrs/TULOGIN/teoria.TULOGIN.18.txt, donde debes escribir tus respuestas. TULOGIN ser´a tu nombre de usuario en el laboratorio.

# Ejercicio 1

Explica brevemente qu´e hacen las siguientes ´ordenes:

1. su
2. su xxx
3. sudo xxx
4. sudo su

### Respuesta

1. Solicita la contrasen˜a de root. Si es correcta, la shell actual (la sesi´on) pasa a ser de root.
2. Solicita la constrasen˜a del usuario xxx. Si es correcta, la shell actual (la sesi´on) pasa a ser de ese usuario. Si esta orden se invoca con privilegios de root, no solita la contrasen˜a del usuario xxx.
3. Solicita al usuario su contrasen˜a. Si es correcta y el usuario est´a autorizado en el fichero *sudoers*, la orden xxx se ejecutar´a con privilegios de root. Si el usuario introdujo su contrasen˜a recientemente, no se le vuelve a pedir.
4. Igual que el paso anterior, pero la orden que se ejecuta es *su*, por tanto, la shell pasa a ser de root.

# Ejercicio 2

En la metodolog´ıa *scrum*, explica brevemente la diferencia entre un *sprint* y una *historia de usuario*.

### Respuesta

Un *sprint* es el periodo de tiempo en el que se desarrola una nueva versi´on del producto software. Su duraci´on suele ser de entre 2 y 4 semanas.

Una *historia de usuario* es una descripci´on informal en lenguaje natural de una nueva caracter´ıstica en el software, visto desde el punto de vista del usuario.

# Ejercicio 3

Si editamos algo en el fichero .bash\_profile o en el fichero .bashrc, los efectos no son inmediatos. La soluci´on del principiante es *salir y entrar*. Esto funciona, pero hay una forma mejor.

1. ¿Por qu´e funciona *salir y entrar* ?
2. ¿Cu´al es la soluci´on mejor?
3. ¿Por qu´e funciona la *soluci´on mejor* ?

### Respuesta

1. Porque si cerramos la sesi´on y la volvemos a abrir, se leen de nuevo los ficheros de invo- caci´on de la shell (*.bashrc*, *.bash profile*, etc). Por supuesto, si apagamos el ordenador y encendemos de nuevo, tambi´en.
2. Ejecutar expl´ıcitamente los ficheros de invocaci´on de la shell con la orden *source*.
3. Porque ejecuta estos ficheros en el contexto de la misma shell.

# Ejercicio 4

Explica brevemente las diferencias entre una instalaci´on basada en clonaci´on y una instala- ci´on basada en los ficheros *preseed* de Debian y derivados.

### Respuesta

Clonaci´on

El administrador configura una m´aquina. El disco duro de esa m´aquina se copia en las dem´as. Esto exige que el hardware sea id´entico

Ficheros *preseed*

El sistema operativo se instala desde cero en cada m´aquina. Este fichero contiene las respuestas a las preguntas que se le hacen al administrador en las instalaciones manuales ordinarias. Tiene la ventaja de que la instalaci´on se adapta a cada hardware particular.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico. 14 de junio de 2019.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Preparativos:

Crea un directorio ~/lagrs.junio.19/, donde escribir´as todo lo indicado a continuaci´on.

# Ejercicio 1 (5 puntos)

En tu pr´actica 3.5 escribirte un programa que revisaba todos los directorios indicados en la constante DIRECTORIOS para comprobar que ni el taman˜o ni el nu´mero de ficheros contenidos excediera un l´ımite prefijado. Si se exced´ıa algu´n l´ımite, se enviaba un mensaje de alarma mediante telegram. Ahora modificar´as ese programa para que adem´as de hacer lo que ya hac´ıa, compruebe que los directorios y su contenido pertenezca a ciertos usuarios previstos.

* 1. Copia el script de tu pr´actica 3.5 en ~/lagrs.junio.19/monitor.py.
  2. An˜ade al principio del script una l´ınea similar a esta:

USUARIOS\_PERMITIDOS = ["jperez", "mgarcia"]

* 1. Tu programa tiene que comprobar que todos los directorios indicados en la constante DIRECTORIOS pertenezcan a alguno de estos usuarios permitidos. Si alguno de estos di- rectorios pertenece a otro usuario, enviar´a un mensaje de telegram.
  2. Del mismo modo, tu programa tambi´en tiene que avisar si alguno de los ficheros o direc- torios contenidos dentro de estos directorios pertenecen a un usuario que no est´a entre los permitidos.
  3. No es necesario que tu programa recorra recursivamente los subdirectorios de DIRECTORIOS revisando los duen˜os de ficheros y directorios.
  4. Si el script se lanza con la opci´on -f o --force-telegram, siempre enviar´a un mensaje de telegram, ya sea para indicar que hay problemas o para indicar que todo est´a bien (taman˜o de los directorios, nu´mero de ficheros y duen˜os de los directorios y ficheros).
  5. Ens´en˜aselo al profesor cuando acabes.

# Ejercicio 2 (5 puntos)

En este ejercicio preparar´as un contenedor docker donde el usuario *root* tendr´a el editor vim listo para usar, con un fichero de configuraci´on de vim personalizado. Para ello, solo tienes que instalar vim dentro del contedor y copiar en el directorio personal del usuario *root* (esto es, en el directorio /root del contenedor) un fichero oculto con nombre .vimrc y con el siguiente contenido

syntax on

set tabstop=4 set nu

El contenedor estar´a construido a partir de ubuntu 18.04, con los paquetes actualizados a la u´ltima versi´on disponible. Estar´a configurado en espan˜ol y tendr´a vim instalado. (Nada m´as, si an˜ades paquetes o configuraci´on adicional, tendr´as una penalizaci´on en la nota). Los ficheros necesarios para esto seguir´an el mismo convenio que hemos venido usando durante toda la asignatura.

1. El contenedor basado en una imagen llamada exa. Si tu nombre de usuario fuera *jperez*, el contenedor se llamar´ıa jperexa01. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
2. Al ejecutar el contenedor, se abrir´a autom´aticamente una sesi´on de vim. Y despu´es, cuando se cierre este vim, se abrir´a autom´aticamente una sesi´on de shell bash.
3. Los nombres de los ficheros que debes preparar para esto ser´an:

~/lagrs.junio.19/exa/construye.sh

~/lagrs.junio.19/exa/lanza\_jperexa01.sh

~/lagrs.junio.19/exa/context/Dockerfile

~/lagrs.junio.19/exa/context/entrypoint.sh

~/lagrs.junio.19/exa/context/vimrc

(Reemplazando *jper* por las primeras 4 letras de tu login)

1. Observa que el fichero de configuraci´on de vim es un fichero oculto, empieza por pun- to. Pero cuando lo prepares en el directorio contexto, no debe estar oculto (su nom- bre no empieza por punto). En otras palabras: tienes que preparar un fichero llamado

~/lagrs.junio.19/vimrc, que se copie dentro de la imagen del contenedor con nombre

/root/.vimrc

1. Ens´en˜aselo al profesor cuando acabes.

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Examen de teor´ıa. 14 de junio de 2019

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Preparativos

Ejecuta el script prepara

Esto crear´a en tu ordenador el fichero ~/lagrs/TULOGIN/teoria.txt, donde TU- LOGIN es tu nombre de usuario en el laboratorio. Escribe tus respuestas en este fichero.

# Ejercicio 1 (2.5 puntos)

La orden *sudo*. ¿Para qu´e sirve? ¿C´omo se usa? ¿C´omo se configura?

# Ejercicio 2 (2.5 puntos)

Explica las diferencias entre las m´aquinas virtuales *de proceso* y las m´aquinas virtuales

*de sistema*. Pon algu´n ejemplo de cada una.

# Ejercicio 3 (2.5 puntos)

Un tu´nel de ssh remoto, tambi´en llamado tu´nel inverso, sirve para acceder a un ser- vidor tras un NAT. Explica, a nivel conceptual, qu´e hace, por qu´e es u´til, cu´ales ser´ıan t´ecnicas alternativas para acceder al servidor. No es especialmente importante que indi- ques los par´ametros concretos para configurar el tu´nel.

# Ejercicio 4 (2.5 puntos)

Explica brevemente las diferencias fundamentales entre el desarrollo de software *en cascada* y el desarrollo de software *´agil*.

1

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico. 13 de enero de 2020.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Preparativos:

Crea un directorio ~/lagrs.enero.20/, donde escribir´as todo lo indicado a continuaci´on.

# Ejercicio 1 (5 puntos)

En este ejercicio escribir´as un script en python 3 que, usando la order netstat, vigilar´a ciertos puertos y enviar´a un mensaje de telegram si algu´n proceso escucha peticiones en ellos. Edita el fichero ~/lagrs.enero.20/vigila\_puertos.py para que se corresponda con la siguiente especificaci´on:

1. El programa leer´a de un fichero de texto con nombre id\_usuario.txt el identificador de usuario de telegram que recibir´a las alarmas. No especifiques ningun *path*, de forma que se busque en el directorio actual.
2. El programa leer´a el token del bot de telegram en el fichero token.txt. De nuevo, no especifiques ningu´n trayecto, que el script lea el fichero desde el directorio actual.
3. En las primeras l´ıneas del script define una lista parecida a esta:

PUERTOS\_TCP = [6666, 443]

1. Cada vez que se ejecute el programa, revisar´a todos los puertos TCP especificados en PUERTOS TCP, y si alguno de ellos est´a ocupado, en cualquier estado (ya sea *ESCU- CHAR*, *ESTABLECIDO*, o cualquier otro en cualquier otro idioma: *LISTEN*, *ESTA- BLISHED*... ), enviar´a una alarma a trav´es de telegram describiendo toda la informaci´on relevante proporcionada por netstat.
2. La alarma solo se disparar´a si el puerto est´a ocupado en alguna direcci´on local, es irrele- vante lo que pase en direcciones remotas.
3. Usa romanserver.py para probar el programa. Cuando funcione, ens´en˜aselo al profesor.

# Ejercicio 2 (5 puntos)

En este ejercicio preparar´as un contenedor docker que al ejecutarse:

1. El usuario root tendr´a una sesi´on de shell.
2. Este usuario tendr´a un directorio ~/bin.
3. Tendr´a en ~/bin/tictac el siguiente script

#!/bin/bash while true do

sleep 1

echo -n "tic" sleep 1

echo " tac" done

1. La variable de entorno PATH del usuario root ser´a la que prepare normalmente la distri- buci´on, pero an˜adiendo el directorio *bin* del *home* del usuario root. De tal forma que se podr´a lanzar el script anterior tecleando simplemente tictac, desde cualquier directorio, sin an˜adir ningu´n *path*.

El contenedor estar´a construido a partir de ubuntu 18.04. No actualices los paquetes y d´ejalo en ingl´es. (Si an˜ades paquetes o configuraci´on adicional, tendr´as una penalizaci´on en la nota). Los ficheros necesarios para esto seguir´an el mismo convenio que hemos venido usando durante toda la asignatura.

1. El contenedor estar´a basado en una imagen llamada exa. Si tu nombre de usuario fuera *jperez*, el contenedor se llamar´ıa jperexa01. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
2. Los nombres de los ficheros que debes preparar para esto ser´an:

~/lagrs.enero.20/exa/construye.sh

~/lagrs.enero.20/exa/lanza\_jperexa01.sh

~/lagrs.enero.20/exa/context/Dockerfile

~/lagrs.enero.20/exa/context/entrypoint.sh

~/lagrs.enero.20/exa/context/bashrc

~/lagrs.enero.20/exa/context/tictac

(Reemplazando *jper* por las primeras 4 letras de tu login)

1. Observa que el fichero de configuraci´on .bashrc es un fichero oculto, empieza por punto. Pero cuando lo prepares en el directorio contexto, no debe estar oculto (su nombre no empieza por punto). En otras palabras: tienes que preparar un fichero llamado bashrc, que dentro del contenedor tendr´a por nombre .bashrc
2. Ens´en˜aselo al profesor cuando acabes.

2

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Examen de teor´ıa. 13 de enero de 2020

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Preparativos

Ejecuta el script prepara

Esto crear´a en tu ordenador el fichero ~/lagrs/TULOGIN/teoria.txt, donde TU- LOGIN es tu nombre de usuario en el laboratorio. Escribe tus respuestas en este fichero.

# Ejercicio 1 (2.5 puntos)

* 1. Un usuario de Unix ¿puede pertenecer a m´as de un grupo? En caso afirmativo ¿Es habitual? ¿Qu´e sentido tiene esto?
  2. Explica brevemente que es el *grupo primario*.
  3. Explica brevemente que es el *grupo primario por omisi´on*.

# Ejercicio 2 (2.5 puntos)

Explica con detalle esta sesi´on de Unix: qu´e es cada linea, cada o´rden, qu´e hace, qu´e sucede y por qu´e.

01 koji@mazinger:~$ echo hola > fichero.txt 02 koji@mazinger:~$ file fichero.txt

1. fichero.txt: ASCII text
2. koji@mazinger:~$ echo ni~no >> fichero.txt 05 koji@mazinger:~$ file fichero.txt

06 fichero.txt: UTF-8 Unicode text

(Evidentemente los nu´meros iniciales no forman parte de las o´rdenes, solo aparecen para facilitar la referencia)

# Ejercicio 3 (2.5 puntos)

*Continuous Integration*, *Continuous Delivery* y *Continuous Deployment* :

1. Explica brevemente en qu´e consiste cada una de estas tres t´ecnicas.
2. Cuando no exist´ıan ¿Cu´al era la manera habitual de trabajar? ¿Qu´e problemas ten´ıa?

1

# Ejercicio 4 (2.5 puntos)

Para administrar un conjunto de m´aquinas Unix podemos usar dos tipos de herra- mientas:

Scripts desarrollados por nosotros, en bash, en python o en algu´n lenguaje similar. Accediendo a las m´aquinas por ssh o m´etodos parecidos.

Software de administraci´on como *Ansible*, *Puppet* o *Chef*

Contesta brevemente

1. ¿Cu´ales son las diferencias principales entre trabajar con t´ecnicas del primer grupo o con herramientas del segundo grupo?
2. Si usamos herramientas del segundo grupo ¿Cu´ales son las diferencias principales entre *Ansible* y *Puppet* ?

2

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen teorico-pr´actico parcial. 25 de noviembre de 2020.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Entra en tu cuenta del laboratorio y ejecuta

~mortuno/prepara

Esto crear´a el fichero ~/parcial.2020.TULOGIN.txt (donde TULOGIN es tu nombre de usuario). Contesta las dos preguntas en este fichero, solo aqu´ı, en ningu´n otro sitio.

# Ejercicio 1 (3 puntos)

Supongamos que tienes que tienes que disen˜ar el sistema inform´atico basado en Linux para un colegio con unos pocos cientos de ordenadores. La soluci´on podr´ıa basarse tanto en virtua- lizaci´on como en clonaci´on. Explica brevemente en qu´e consiste cada enfoque, cu´ales son sus ventajas y cu´ales sus inconvenientes. Sin duda en tu respuesta habr´a *dependes*. En esos casos, considera las opciones posibles. Ejemplo: Si los puestos son relativamente potentes entonces ..., en otro caso .... Si son muy heterog´eneos, ...., y si no, ....

# Ejercicio 2 (7 puntos)

Deseamos un par de contenedores con la siguiente especificaci´on:

Estar´an basados en la misma imagen.

Tendr´an ambos el servicio *iperf* listo para atender peticiones TCP en el puerto 9999.

Montar´an ambos por **sshfs** en el directorio */media/(lab)* tu *home* de la maquina *f-l2108- pc02*. Supodremos que est´a funcionando bien, en un sistema m´as robusto habr´ıa que verificarlo, pero aqu´ı lo omitimos). Observa que **no** se te pide un montaje *bind*.

La autenticaci´on ser´a manual, esto es, al arrancarse el contenedor, te pedir´a tu contrasen˜a en el laboratorio.

Esto no lo has hecho en pr´acticas (usar sshfs dentro de un contenedor). Pero s´ı lo vimos en clase de teor´ıa. Recuerda que la orden docker run necesitar´a ciertos par´ametros, documentados en las transparencias sobre Docker.

Especifica los ficheros necesarios para conseguir esto en Docker (una versi´on moderna, la incluida Ubuntu 20.04):

1. Indica qu´e ficheros habr´a en el directorio *context* y con qu´e contenido.
2. Al igual que en pr´acticas, queremos un script para preparar la imagen. Indica qu´e nombre tendr´ıa y qu´e contenido.
3. Al igual que en pr´acticas, queremos un script para lanzar cada contenedor. Indica qu´e nombre tendr´ıan y qu´e contenido.
4. Al lanzar los contenedores, el usuario escribir´a su contrasen˜a, a continuaci´on deber´a ave- riguar la IP del *otro* contenedor y lanzar el cliente de iperf. Indica c´omo hacer estos pasos.

### Soluci´on

Fichero context/Dockerfile

FROM ubuntu:20.04

RUN apt update && apt upgrade -y

RUN apt-get install -y net-tools iputils-ping sshfs iperf COPY entrypoint.sh /

EXPOSE 9999

CMD ["/entrypoint.sh"]

Fichero context/entrypoint.sh.

#!/bin/bash

iperf -s -p 9999 & mkdir /media/lab

sshfs [mortuno@f-l2108-pc02.aulas.etsit.urjc.es:](mailto:mortuno@f-l2108-pc02.aulas.etsit.urjc.es) /media/lab

/bin/bash

Script para crear la imagen, construye.sh

#!/bin/bash

docker build -t jperez/exa context

Script para lazar el primer contenedor, lanza\_jperexa01.sh

#!/bin/bash PREFIJO=jper IMAGEN=exa USUARIO=jperez

CONTENEDOR=$PREFIJO${IMAGEN}01

docker run --rm -it -h $CONTENEDOR --name $CONTENEDOR \

--cap-add SYS\_ADMIN --device /dev/fuse \

--security-opt apparmor:unconfined \

$USUARIO/$IMAGEN

El script para lanzar el segundo contenedor, lanza\_jperexa02.sh es id´entico, excepto la l´ınea CONTENEDOR=$PREFIJO${IMAGEN}02

Todos los scripts necesitan permiso de ejecuci´on

\verb|chmod ugo+x context/entrypoint.sh|

\verb|chmod ugo+x \*.sh|

Una vez lanzados los contendores, introducimos la contrasen˜a, averiguamos la direcci´on IP de cada uno con la orden *ifconfig* y ejecutamos

iperf -c <IP\_SERVIDOR> -p 9999

Donde <IP\_SERVIDOR> es la direcci´on IP del contenedor donde est´a el servidor iperf.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen del primer parcial. 4 de febrero de 2021.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Entra en tu cuenta del laboratorio y ejecuta

~mortuno/prepara

Esto crear´a el fichero ~/lagrs.feb.21/parcial.TULOGIN.txt (donde TULOGIN es tu nombre de usuario). Contesta la pregunta en este fichero, solo aqu´ı, en ningu´n otro sitio.

# Ejercicio u´nico

En este ejercicio indicar´as todo lo necesario para preparar un contenedor docker segu´n la siguiente especificaci´on:

1. En el contenedor habr´a un usuario con el mismo nombre que tu cuenta del laboratorio.
2. Al poner en marcha el contenedor, en el terminal habr´a una sesi´on de shell de este usuario, que tendr´a el multiplexor de terminales *tmux* listo para usar.
3. *tmux* estar´a personalizado de forma que la tecla *bind* no sea Ctrl b sino Ctrl a. Para ello, solo hay que instalar el paquete *tmux* dentro del contenedor y copiar en el directorio personal del usuario un fichero oculto con nombre .tmux.conf y con el siguiente contenido:

set -g prefix C-a bind C-a send-prefix unbind C-b

1. El contenedor estar´a construido a partir de ubuntu 20.04, con los paquetes actualizados a la u´ltima versi´on disponible. Estar´a configurado en espan˜ol y tendr´a *tmux* instalado. (Nada m´as, si an˜ades paquetes o configuraci´on adicional, tendr´as una penalizaci´on en la nota).
2. El contenedor estar´a basado en una imagen llamada exa. Si tu nombre de usuario fuera *jperez*, el contenedor se llamar´ıa jperexa01. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
3. Siguiendo el mismo convenio que hemos venido usando durante toda la asignatura, los nombres de los ficheros que habr´ıa que preparar ser´ıan.

~/lagrs.feb.21/exa/construye.sh

~/lagrs.feb.21/exa/lanza\_jperexa01.sh

~/lagrs.feb.21/exa/context/Dockerfile

~/lagrs.feb.21/exa/context/entrypoint.sh

~/lagrs.feb.21/exa/context/tmux.conf

(Reemplazando *jper* por las primeras 4 letras de tu login). Pero no escribas en esos ficheros,

**Escribe solamente en parcial.TULOGIN.txt** indicando qu´e habr´ıa en cada fichero.

P.e. *Contenido del fichero X: blabla. Contenido del fichero Y: blabla*

1. Observa que el fichero de configuraci´on de *tmux* es un fichero oculto, empieza por pun- to. Pero cuando lo prepares en el directorio contexto, no debe estar oculto (su nom- bre no empieza por punto). En otras palabras: tienes que preparar un fichero llamado

~/lagrs.feb.21/tmux.conf, que se copie dentro de la imagen del contenedor, en el *home*

del usuario, con nombre .tmux.conf

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico (segundo parcial)

**4 de febrero de 2021.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Ejercicio unico (10 puntos)

En tu cuenta encontrar´as

El fichero ~/lagrs.feb.21/ckcron.TULOGIN.py, que debes completar para que sea un programa que revise una serie de tablas de cron. Adema´s de detectar errores, dar´a avisos sobre aspectos de la tabla que, siendo correctos, pueden ser peligrosos.

Una serie de ficheros de texto ~/lagrs.feb.21/tablaNN.txt con tablas de cron, que tambi´en aparecen impresas a continuaci´on. Estas tablas ser´an los casos de prueba para tu programa. Algunas son correctas, otras tienen errores o situaciones que consideramos pe- ligrosas. Tu programa debe detectar todos los problemas que puedan tener estos ejemplos. Puedes ignorar los posibles errores no previstos aqu´ı.

Tu programa:

Recibir´a una lista de argumentos por l´ınea de comandos, que ser´an nombres de ficheros. Usa *sys.argv*, es suficiente para este caso. Si te sobra tiempo, puedes reemplazarlo por *optparse*.

Ejemplo

ckcron.jperez.py tabla01.txt tabla02.txt tabla03.txt

Para cada una de esas tablas, tu programa deber´a decir si les ha detectado algu´n error o no. (El programa no deber´ıa decir que son *correctas* porque no har´a una comprobaci´on exhaustiva). En caso de error, deber´a describirlo brevemente.

El programa mostrar´a un *aviso* si algu´n comando de la tabla est´a indicado como un nombre de fichero, sin especificar *path*.

El programa mostrar´a un *aviso* si algu´n comando (con *path* completo) no existe. Tal vez no sea un error porque tal vez s´ı exista cuando se ejecute, pero consideramos que esta situaci´on merece al menos un aviso. (No importa si un comando sin trayecto existe o no, simplemente notifica el aviso por falta de trayecto)

El programa mostrar´a un *aviso* si alguna variable de entorno no est´a definida. Tal vez no sea un error, tal vez se definir´a en su momento, pero merece un aviso.

Naturalmente, el programa no solo debe funcionar para esas tablas en particular, sino que debe admitir cualquier otra tabla con problemas semejantes.

Deja claro en el c´odigo fuente qu´e tipo de error est´as buscando. Ejemplo: *#Ahora detec- tamos si hay texto en klingon como en tabla12.txt, que es incorrecto.*

Observaciones

Encontrar´as en el esqueleto de fichero una funci´on sencilla que, a partir de una l´ınea de texto, devuelve una lista de lo que parezcan ser una variable de shell.

Las tablas de cron reales normalmente definen variables de entorno antes de enumerar los comandos. Ignora esto.

No preguntes al profesor qu´e errores debe detectar tu programa o qu´e errores hay en los ejemplos, eso forma parte de la materia del examen.

Tu programa deber´ıa indicar todos los errores y avisos que encuentre. Sugerencia: escri- be funciones para detectar posibles problemas sobre l´ıneas de texto, y aplica todas las funciones a cada l´ınea.

Un enfoque diferente ser´ıa que en cuanto el programa encuentre un error o un aviso, lo muestre y deje de buscar. Esto es claramente peor. Procura evitarlo.

### Tablas de ejemplo

# tabla01.txt

# m h dayofmonth month dow command

\* \* \* \* \* /usr/bin/touch /tmp/probando.txt

\*/5 \* \* \* \* $HOME/bin/holamundo.sh

# tabla02.txt

# m h dayofmonth month dow command

\* \* \* \* \* \* /usr/bin/touch /tmp/probando.txt

# tabla03.txt

# m h dayofmonth month dow command

\* \* \* \* \* /NADA/NADA/touch /tmp/probando.txt

# tabla04.txt

# m h dayofmonth month dow command

* \* \* 14 9 /usr/bin/touch /tmp/probando.txt

# tabla05.txt

# m h dayofmonth month dow command

* \* \* 1 3 touch /tmp/probando.txt

# tabla06.txt

# m h dayofmonth month dow command

\*/5 \* 4 \* $CASA/bin/holamundo.sh

// tabla07.txt

// m h dayofmonth month dow command

* 8 \* \* 1-5 /usr/bin/touch /tmp/probando.txt

# tabla08.txt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | m | h dayofmonth month | dow | command |
|  | \* | 22-7 \* \* | 1-5 | /usr/bin/touch /tmp/probando.txt |

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen de teor´ıa

**4 de febrero de 2021.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Ejercicio 1. (3 puntos)

Describe brevemente c´omo se puede aplicar el multicast en el clonado de m´aquinas, cu´ales son sus ventajas y cu´ales sus inconvenientes.

# Ejercicio 2. (3 puntos)

¿En qu´e se parece UTF-8 a Unicode? ¿En qu´e se diferencian?

# Respuesta

Son dos t´erminos muy relacionados, con frecuecia se consideran sin´onimos pero no lo son. Unicode es un estandar que aparece en la d´ecada de 1990 para representar las letras y algunos otros s´ımbolos como jerogl´ıficos o emoticonos de todos los lenguajes del mundo, naturales o artificiales. Podemos decir que Unicode es una gran tabla con algo menos de 150 000 entradas, a cada caracter le asigna un nu´mero.

Hay diversas formas de codificar Unicode, esto es, de representar ese nu´mero en un orde- nador. UTF-8 es una de ellas, la m´as usada actualmente. Es retrocompatible con ASCII, el nu´mero de bytes que ocupa cada caracter es variable (menos bytes los m´as frecuentes, m´as bytes los m´as raros), y se auto-sincroniza (tiene un mecanismo para evitar que en un flujo de bytes haya ambigu¨edades, para saber si cierto byte pertenece a un nuevo s´ımbolo o al s´ımbolo anterior)

# Ejercicio 3. (4 puntos)

Como seguramente sabes, chmod u+s y chmod u-s permiten quitar y poner el bit SUID a un fichero. ¿Qu´e significa esto? ¿Para qu´e se usa? ¿Por qu´e puede suponer un problema de seguridad?

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico

**30 de junio de 2021.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Entra en tu cuenta del laboratorio y ejecuta

~mortuno/prepara

Esto crear´a los ficheros

~/lagrs.julio.21/final.txt

~/lagrs.julio.21/conexiones.py

Contesta las preguntas en estos dos ficheros, en ningu´n otro sitio.

# Ejercicio 1 (6 puntos)

En este ejercicio indicar´as todo lo necesario para preparar un contenedor docker segu´n la siguiente especificacion:

* 1. Deseamos que al ponerse en marcha el contenedor, el usuario *root* tenga una sesi´on de shell abierta.
  2. El usuario podr´a ejecutar la orden maquina. Ser´a un script de shell que simplemente escriba en la salida est´andar el mensaje *Est´as en TU-HOST*, donde *TU-HOST* ser´a el nombre del *host*, obtenido de la variable de entorno correspondiente 1.
  3. El *path* completo del script *maquina* ser´a /opt/examen/bin/maquina.
  4. Como seguramente sabes, para que el usuario pueda ejecutar la orden tecleando maquina ( y no un trayecto completo como /opt/examen/bin/maquina), tendr´as que an˜adir cierta l´ınea al fichero .bashrc del usuario *root*.

Haz esto de forma similar a lo que hiciste en la pr´actica 1.20 para an˜adir una l´ınea a

/etc/host. Esto es:

* + 1. Prepara, en el directorio contexto, un fichero delta\_bashrc que contenga la l´ınea adecuada.
    2. Haz que este fichero aparezca en el directorio /tmp/ de la imagen.
    3. Haz que cuando se prepare la imagen del contenedor, se an˜ada el contenido de

delta\_bashrc al fichero .bashrc del usuario *root*.

* 1. El contenedor estar´a construido a partir de ubuntu 20.04, con los paquetes actualizados a la u´ltima versi´on disponible. No tendr´a ningu´n otro paquete ni ninguna configuraci´on adicional.

1En el caso particular de este examen, el nombre del host lo conoces a priori, ser´a jperex2021-1. Pero no lo escribas como una cadena constante en el script. Obtenlo de la variable de entorno

* 1. La imagen del contenedor se llamar´a ex2021. Los contenedores basados en esta imagen (en este caso solo uno), tendr´an como prefijo parte de tu nombre de usuario, y como sufijo, un nu´mero. Si tu nombre de usuario fuera *jperez*, el prefijo ser´ıa *jper*. El sufijo, *-1*. Por tanto el contenedor se llamar´ıa jperex2021-1. El nombre de *host* ser´a este mismo. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
  2. Siguiendo un convenio como el que hemos venido usando durante toda la asignatura, los nombres de los ficheros ser´ıan:

~/lagrs.jul.21/ex2021/construye.sh

~/lagrs.jul.21/ex2021/lanza\_jperex2021-1.sh

~/lagrs.jul.21/ex2021/context/Dockerfile

~/lagrs.jul.21/ex2021/context/entrypoint.sh

~/lagrs.jul.21/ex2021/context/maquina

~/lagrs.jul.21/ex2021/context/delta\_bashrc

(Reemplazando *jper* por las primeras 4 letras de tu login). Pero no escribas en esos ficheros, **escribe solamente en final.txt** indicando qu´e habr´ıa en cada fichero. P.e. *Contenido del fichero X: blabla. Contenido del fichero Y: blabla, etc*.

# Ejercicio 2 (4 puntos)

Escribe un programa en python llamado ~/lagrs.jul.21/conexiones.py segu´n la siguien- te especificaci´on:

1. El programa usar´a la librer´ıa *optparse* para aceptar un nu´mero de puerto, con la opci´on

-p. Ejemplo: conexiones.py -p 631

1. Partiendo de la informaci´on proporcionada por la orden de shell netstat -tupan, el programa mostrar´a en la salida est´andar todas las conexiones tcp v4 y todas las *pseudo- conexiones* udp v4 que usen dicho puerto en la direcci´on local. El programa ignorar´a las conexiones tcp v6 y las udp v6.
2. Tambi´en indicar´a el nu´mero de conexiones totales que muestra.

En otras palabras, el programa invocar´a la orden netstat -tupan (con estas opciones y no otras), ignorar´as las l´ıneas correspondientes a *tcp6* y *udp6*, mostrar´a todas aquellas l´ıneas cuyo puerto de la columna *Direcci´on local* sea el indicado, e indicar´a cu´antas l´ıneas son.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen de teor´ıa

**30 de junio de 2021.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Entra en tu cuenta del laboratorio y ejecuta

~mortuno/prepara

Esto crear´a el fichero ~/lagrs.julio.21/teoria.TULOGIN.txt (donde TULOGIN es tu nombre de usuario). Contesta las cuatro preguntas en este fichero.

# Ejercicio 1. (2.5 puntos)

En administraci´on de sistemas Unix, cuando usamos la palabra *root* podemos referirnos a tres cosas distintas. Por escrito queda un poco m´as claro, de palabra puede haber confusiones.

¿Qu´e tres significados tiene?

# Ejercicio 2. (2.5 puntos)

Explica brevemente en qu´e consiste el m´etodo *secret sharing* para la gesti´on de contrasen˜as, cu´ales son sus ventajas y qu´e herramienta podemos usar en Linux con este prop´osito. No es necesario que indiques los detalles del uso de la herramienta.

# Ejercicio 3. (2.5 puntos)

Explica brevemente el concepto *desarrollo de software ´agil*.

# Ejercicio 3. (2.5 puntos)

Cuando usamos *cron* hay que tener cuidado con las variables de entorno, es frecuente que el principiante cometa errores relacionados con esto. Expl´ıcalo brevemente.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico

**20 de enero de 2022.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Ejercicio 1 (6 puntos)

En este ejercicio preparar´as un contenedor Docker segu´n la siguiente especificaci´on:

1. Deseamos que al ponerse en marcha el contenedor, el usuario *root* tenga una sesi´on de shell abierta.
2. En el contenedor, habr´a un script de shell llamado /opt/examen/bin/maquina
3. Este script maquina escribir´a en la salida est´andar el mensaje *Est´as en TU-HOST*, donde

*TU-HOST* ser´a el nombre del *host*, obtenido de la variable de entorno correspondiente. Naturalmente, *TU-HOST* ser´a siempre el nombre de la m´aquina donde se ejecuta el script

maquina. Cuando se ejecute en tu puesto del laboratorio, devolver´a el nombre del puesto

del laboratorio. Cuando se ejecute en el servidor de contenedores, devolver´a el nombre del servidor de contenedores. Cuando se ejecute en un contenedor, devolver´a el nombre de ese contenedor.

1. Dentro del contenedor, con la sesi´on de shell de root abierta, el usuario podr´a ejecutar la orden maquina, *a secas*, sin necesidad de escribir el trayecto completo

(/opt/examen/bin/maquina).

Como seguramente sabes, para poder conseguir esto, tendr´as que an˜adir cierta l´ınea al fichero .bashrc del usuario *root*.

Hazlo de forma similar a lo que hiciste en la pr´actica 1.23 para an˜adir una l´ınea a

/etc/host. Esto es:

* 1. Prepara, en el directorio contexto, un fichero delta\_bashrc que contenga la l´ınea adecuada.
  2. Haz que este fichero aparezca en el directorio /tmp/ de la imagen.
  3. Haz que cuando se prepare la imagen del contenedor, se an˜ada el contenido de

delta\_bashrc al fichero .bashrc del usuario *root*.

1. El contenedor estar´a construido a partir de ubuntu 20.04, con los paquetes actualizados a la u´ltima versi´on disponible. No tendr´a ningu´n otro paquete ni ninguna configuraci´on adicional.
2. La imagen del contenedor se llamar´a ex2022. Los contenedores basados en esta imagen (en este caso solo uno), tendr´an como prefijo parte de tu nombre de usuario, y como sufijo, un nu´mero. Si tu nombre de usuario fuera *jperez*, el prefijo ser´ıa *jper*. El sufijo, *-1*. Por tanto el contenedor se llamar´ıa jperex2022-1. El nombre de *host* ser´a este mismo. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
3. Siguiendo un convenio como el que hemos venido usando durante toda la asignatura, los nombres de los ficheros ser´an:

~/lagrs.ene.22/ex2022/construye.sh

~/lagrs.ene.22/ex2022/lanza\_jperex2022-1.sh

~/lagrs.ene.22/ex2022/context/Dockerfile

~/lagrs.ene.22/ex2022/context/entrypoint.sh

~/lagrs.ene.22/ex2022/context/maquina

~/lagrs.ene.22/ex2022/context/delta\_bashrc

(Reemplazando *jper* por las primeras 4 letras de tu login).

Cuando acabes, ensen˜a el resultado al profesor.

# Ejercicio 2 (4 puntos)

Escribe un programa en python llamado ~/lagrs.ene.22/conexiones.py segu´n la siguien- te especificaci´on:

1. El programa usar´a la librer´ıa *optparse* para aceptar un nu´mero de puerto, con la opci´on

-p. Ejemplo: conexiones.py -p 631

1. Partiendo de la informaci´on proporcionada por la orden de shell netstat -tupan, el programa mostrar´a en la salida est´andar todas las conexiones tcp v4 y todas las *pseudo- conexiones* udp v4 que usen dicho puerto en la direcci´on local. El programa ignorar´a las conexiones tcp v6 y las udp v6.
2. Tambi´en indicar´a el nu´mero de conexiones totales que muestra.

En otras palabras, el programa invocar´a la orden netstat -tupan (con estas opciones y no otras), ignorar´as las l´ıneas correspondientes a *tcp6* y *udp6*, mostrar´a todas aquellas l´ıneas cuyo puerto de la columna *Direcci´on local* sea el indicado, e indicar´a cu´antas l´ıneas son.

Cuando acabes, ensen˜a el resultado al profesor.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen de teor´ıa

**20 de enero de 2022.**

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Entra en tu cuenta del laboratorio y ejecuta

~mortuno/prepara

Comprueba que esto deja en tu cuenta el fichero ~/lagrs.enero.22/teoria.TULOGIN.txt (donde TULOGIN es tu nombre de usuario).

Contesta ah´ı las cuatro preguntas siguientes:

# Ejercicio 1. (2.5 puntos)

En administraci´on de sistemas Unix, cuando usamos la palabra *root* podemos referirnos a tres cosas distintas. Por escrito queda un poco m´as claro, de palabra puede haber confusiones.

¿Qu´e tres significados tiene?

# Ejercicio 2. (2.5 puntos)

Explica brevemente en qu´e consiste el m´etodo *secret sharing* para la gesti´on de contrasen˜as, cu´ales son sus ventajas y qu´e herramienta podemos usar en Linux con este prop´osito. No es necesario que indiques los detalles del uso de la herramienta.

# Ejercicio 3. (2.5 puntos)

Explica brevemente el concepto *desarrollo de software ´agil*.

# Ejercicio 4. (2.5 puntos)

Cuando usamos *cron* hay que tener cuidado con las variables de entorno, es frecuente que el principiante cometa errores relacionados con esto. Expl´ıcalo brevemente.

### Laboratorio de administraci´on y gesti´on de redes y sistemas Examen pr´actico. 18 de junio de 2022.

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

Preparativos:

Si no has hecho el examen de teor´ıa, crea un directorio ~/lagrs.junio.22/, donde escri- bir´as todo lo indicado a continuaci´on.

# Ejercicio 1 (5 puntos)

En este ejercicio escribir´as un script en python 3 que har´a lo mismo que tu pr´actica 2.8,

~/lagrs/practica02/recientes.py, con la siguiente diferencia:

1. El programa se llamar´a ~/lagrs.junio.22/bot.py
2. El programa leer´a de un fichero de texto con nombre id\_usuario.txt el identificador de usuario de telegram que recibir´a las alarmas. No especifiques ningu´n *path*, de forma que se busque en el directorio actual.
3. El programa leer´a el token del bot de telegram en el fichero token.txt. De nuevo, no especifiques ningu´n trayecto, que el script lea el fichero desde el directorio actual.
4. El intervalo de tiempo no se expresar´a en d´ıas, sino en horas.
5. En tu pr´actica probablemente habr´as usado la funci´on os.path.getmtime. En este caso, modif´ıcala para que se base en la orden de shell ls -l --full-time. Y viceversa: si en tu pr´actica usaste ls, ahora usa os.path.getmtime.
6. Los par´ametros de entrada (nombre de directorio e intervalo de tiempo) no se los pregun- tar´a el bot al usuario, sino que los marcar´a el administrador como argumentos de la l´ınea de comandos, al lanzar el script *bot.py*. Usa la librer´ıa *optparse*, como en tu pr´actica 2.4.

# Ejercicio 2 (5 puntos)

En este ejercicio preparar´as un contenedor docker segu´n la siguiente especificaci´on

1. En la imagen del contenedor habr´a un fichero llamado practicas.tgz. La ubicaci´on de este fichero dentro de la imagen ser´a la que tu decidas. Procura que sea *adecuada*. Este fichero contendr´a tus ficheros de de memoria de pr´acticas:

~/lagrs/practica01.md

~/lagrs/practica03.txt

~/lagrs/practica04.txt

1. Cada vez que se inicie el contenedor, estos ficheros deben estar disponibles, descomprimi- dos, dentro del directorio

/root/practicas

1. En el contenedor deber´a estar disponible el editor *vim*
2. Cuando tengas todo listo, ens´en˜aselo al profesor. El profesor te pedir´a que, usando vim, edites cada uno de los ficheros de de memoria de pr´acticas escribiendo en la primera l´ınea tu nombre, apellidos y la fecha de hoy.

El contenedor estar´a construido a partir de ubuntu 20.04, con la configuraci´on de idioma por omisi´on (ingl´es). Si an˜ades paquetes, ficheros o configuraci´on adicional, tendr´as una penalizaci´on en la nota. Los ficheros necesarios para esto seguir´an el mismo convenio que hemos venido usando durante toda la asignatura.

1. El contenedor estar´a basado en una imagen llamada exa. Si tu nombre de usuario fuera *jperez*, el contenedor se llamar´ıa jperexa01. Sustituye *jper* por los primeros 4 caracteres de tu nombre de usuario.
2. Los nombres de los ficheros que debes preparar para esto ser´an, entre otros:

~/lagrs.junio.22/exa/construye.sh

~/lagrs.junio.22/exa/lanza\_jperexa01.sh

~/lagrs.junio.22/exa/context/Dockerfile

~/lagrs.junio.22/exa/context/entrypoint.sh

(Reemplazando *jper* por las primeras 4 letras de tu login) Esta lista no est´a completa, para cumplir con la especificaci´on puedes necesitar algu´n fichero adicional

### Laboratorio de Administraci´on y Gestio´n de Redes y Sistemas Examen de teor´ıa. 18 de junio de 2022

Grado en Ingenier´ıa Telem´atica, Universidad Rey Juan Carlos

# Preparativos

Ejecuta el script ~mortuno/prepara

Esto crear´a en tu ordenador el fichero ~/lagrs.junio.22/teoria.TULOGIN.txt, donde TULOGIN es tu nombre de usuario en el laboratorio. Escribe tus respuestas en este fichero.

# Ejercicio 1 (2.5 puntos)

¿Cu´al es la diferencia entre las o´rdenes Unix *sudo* y *su*?

# Ejercicio 2 (2.5 puntos)

Explica con detalle esta sesi´on de Unix: qu´e es cada linea, cada o´rden, qu´e hace, qu´e sucede y por qu´e.

01 koji@mazinger:~$ echo hola > fichero.txt 02 koji@mazinger:~$ file fichero.txt

1. fichero.txt: ASCII text
2. koji@mazinger:~$ echo ma~nana >> fichero.txt 05 koji@mazinger:~$ file fichero.txt

06 fichero.txt: UTF-8 Unicode text

(Evidentemente los nu´meros iniciales no forman parte de las o´rdenes, solo aparecen para facilitar la referencia)

# Ejercicio 3 (2.5 puntos)

Describe brevemente qu´e es el *despliegue frecuente* y cu´ales son sus principales ven- tajas.

# Ejercicio 4 (2.5 puntos)

Describe brevemente qu´e es y para qu´e sirve un *tu´nel ssh inverso*. No es necesario que indiques detalles sobre par´ametros y sintaxis, basta una descripci´on conceptual.

1

© 2022 Miguel Angel Ortuño Pérez.

Algunos derechos reservados. Este documento se distribuye bajo la licencia *Atribución-CompartirIgual 4.0 Internacional* de Creative Commons disponible en

<https://creativecommons.org/licenses/by-sa/4.0/deed.es>

<http://hdl.handle.net/10115/20116>