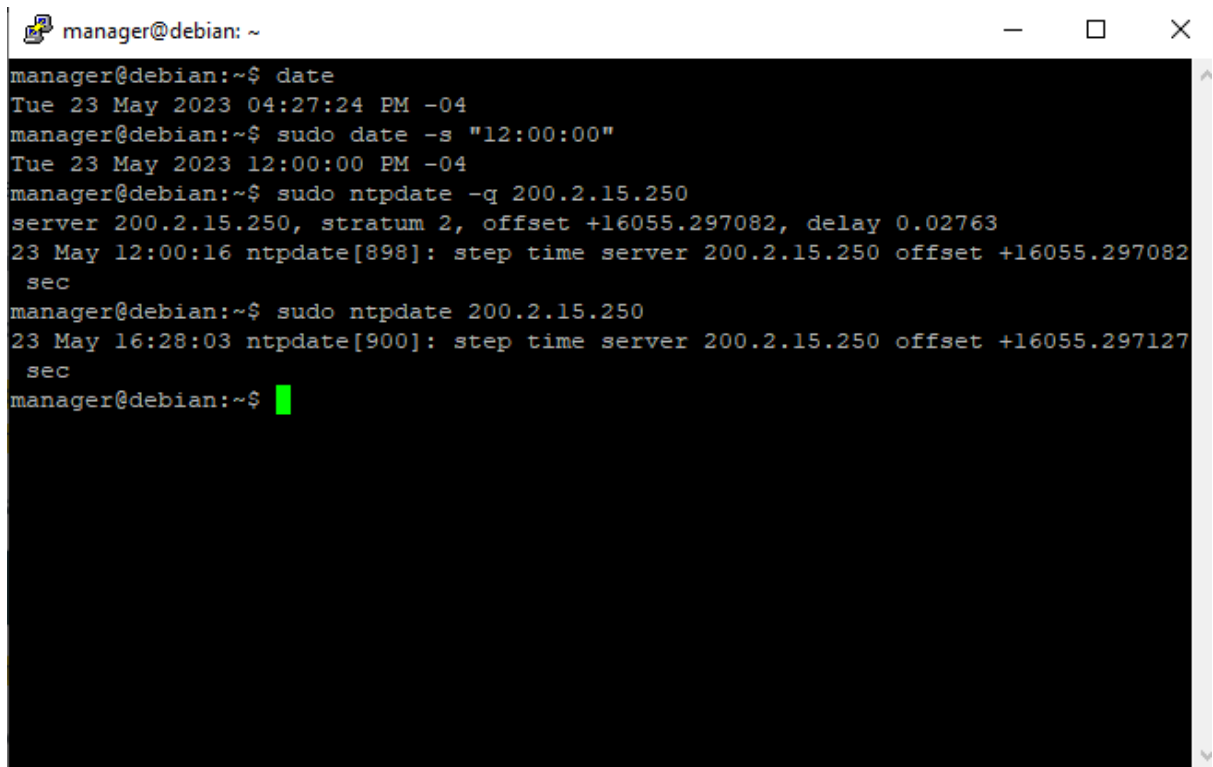


Universidad Católica Andrés Bello  
Ingeniería Informática  
Sistemas Distribuidos  
William Gandino, C.I: 23630951

### PRÁCTICA #3

- 1) La diferencia entre las dos salidas es que se encuentran en zonas horarias distintas. La primera salida sin UTC muestra la fecha y hora en la zona horaria local (configuración del equipo) y la diferencia de -04 con respecto al meridiano 0. La segunda salida muestra la zona horaria de referencia utilizada (UTC) que se encuentra en el meridiano de Greenwich.
- 2) Siendo el formato "Caracas, [día de la semana] [día del mes] de [mes del año letra] de [año]" sería: `date +"Caracas, %A %d de %B de %Y"`
- 3) `sudo date -s "01/01/2023"` o `sudo date -s "$(date +%Y)-01-01"`
- 4)

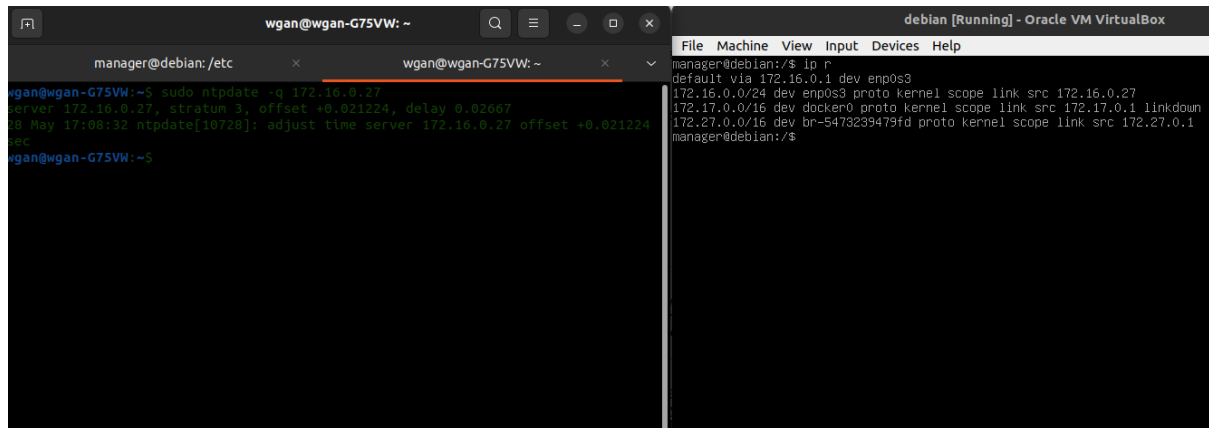


```
manager@debian: ~  
manager@debian:~$ date  
Tue 23 May 2023 04:27:24 PM -04  
manager@debian:~$ sudo date -s "12:00:00"  
Tue 23 May 2023 12:00:00 PM -04  
manager@debian:~$ sudo ntpdate -q 200.2.15.250  
server 200.2.15.250, stratum 2, offset +16055.297082, delay 0.02763  
23 May 12:00:16 ntpdate[898]: step time server 200.2.15.250 offset +16055.297082  
sec  
manager@debian:~$ sudo ntpdate 200.2.15.250  
23 May 16:28:03 ntpdate[900]: step time server 200.2.15.250 offset +16055.297127  
sec  
manager@debian:~$
```

- 5) Un servidor (NTP) dentro de la red de la UCAB.
- 6) ¿Cómo instalar, configurar y activar un servidor NTP en Debian?
  - a) Instalación: `sudo apt-get install ntp`
  - b) Configuración: abrir el archivo ubicado en la ruta `/etc/ntp.conf` con permisos de usuario y modificar a las necesidades. Se pueden restringir subredes enteras, direcciones específicas y también modificar cuáles y cuántos servidores de consulta se usarán.
  - c) Activación: correr el comando `sudo systemctl restart ntp`, el cual dará reinicia el servidor para cargar las modificaciones hechas en el paso anterior.

Para confirmar esto se realizó una instalación en una máquina virtual de Debian y se probó todo lo anterior, teniendo como resultado el siguiente:

Screenshot del servidor funcionando en la máquina virtual:



7) Secuencia de comandos para crear la estructura de nodos con los datos asignados:

```
WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[zk: zookeeper(CONNECTED) 0] create /app_01
Created /app_01
[zk: zookeeper(CONNECTED) 1] ls /
[app_01, zookeeper]
[zk: zookeeper(CONNECTED) 2] create app_01/sala_01
Path must start with / character
[zk: zookeeper(CONNECTED) 3] create app_01/sala_02
Path must start with / character
[zk: zookeeper(CONNECTED) 4] ls
ls [-s] [-w] [-R] path
[zk: zookeeper(CONNECTED) 5] ls /
[app_01, zookeeper]
[zk: zookeeper(CONNECTED) 6] ls /app_01
[]
[zk: zookeeper(CONNECTED) 7] ls /ap_01
Node does not exist: /ap_01
[zk: zookeeper(CONNECTED) 8] ls /app_01
[]
[zk: zookeeper(CONNECTED) 9] ls /app_01/
Path must not end with / character
[zk: zookeeper(CONNECTED) 10] ls /app_01
[]
[zk: zookeeper(CONNECTED) 11] create /app_01/sala_02
Created /app_01/sala_02
[zk: zookeeper(CONNECTED) 12] create /app_01/sala_01
Created /app_01/sala_01
[zk: zookeeper(CONNECTED) 13] ls /app_01
[sala_01, sala_02]
[zk: zookeeper(CONNECTED) 14] set data "Esto es un dato cualquiera 1" /app_01/sa
la_01
Path must start with / character
[zk: zookeeper(CONNECTED) 15] set /app_01/sala_01 "Esto es un dato cualquiera 1"
[zk: zookeeper(CONNECTED) 16] get /app_01/sala_01
Esto es un dato cualquiera 1
[zk: zookeeper(CONNECTED) 17] set /app_01/sala_02 "Esto es un dato cualquiera 2"
[zk: zookeeper(CONNECTED) 18]
```

8) Con el servidor de Zookeeper corriendo, se corrió un script de python. El script usa la librería kazoo para crear el objeto cliente con la dirección IP del servidor y el puerto que normalmente es 2181, luego llama al método start() para iniciar la conexión, luego llama dos veces al método ensure\_path(rutas), el cual recibe por parámetro las rutas de interés y asegura la existencia de estas y en caso de que no estén creadas, las crea. Luego está el decorador @zk.DataWatch(ruta), el cual se encarga de observar los cambios que suceden en esa ruta dada y los imprime. Por último hay un time.sleep(1), manda al proceso a dormir por un segundo. Todo esto

en conjunto tiene el propósito de analizar a través de una terminal los cambios que están sucediendo en los nodos de interés en el servidor de Zookeeper como se puede observar en los siguientes screenshots:

```
server zookeeper1/172.19.0.2:2181, session id = 0x1000063afb00001, negotiated tim
eout = 30000

WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[zk: zookeeper1(CONNECTED) 0] ls /
[app_01, zookeeper]
[zk: zookeeper1(CONNECTED) 1] ls /app_01
[]
[zk: zookeeper1(CONNECTED) 2] create /app_01/sala_01
Created /app_01/sala_01
[zk: zookeeper1(CONNECTED) 3] create /app_01/sala_02
Created /app_01/sala_02
[zk: zookeeper1(CONNECTED) 4] set data "Esto es un dato cualquiera 1" /app_01/sa
la_01
Path must start with / character
[zk: zookeeper1(CONNECTED) 5] set /app_01/sala_01 "Esto es un dato cualquiera 1"
[zk: zookeeper1(CONNECTED) 6] set /app_01/sala_02 "Esto es un dato cualquiera 2"
[zk: zookeeper1(CONNECTED) 7] set /app_01/sala_01 "Esto es otro dato de prueba e
n la sala 1"
[zk: zookeeper1(CONNECTED) 8]
```

```
wgan@wgan-G75VW:~/Desktop/distribuidos$ nano pruebaZookeeper.py
wgan@wgan-G75VW:~/Desktop/distribuidos$ python3 pruebaZookeeper.py
Version: 1, data: Esto es un dato cualquiera 1
Version: 1, data: Esto es un dato cualquiera 2

Version: 2, data: Esto es otro dato de prueba en la sala 1
Version: 2, data: Esto es otro dato de prueba en la sala 2
```

```
wgan@wgan-G75VW:~/Desktop/distribuidos$ python3 pruebaZookeeper.py
Version: 2, data: Esto es otro dato de prueba en la sala 1
Version: 1, data: Esto es un dato cualquiera 2
Version: 2, data: Esto es otro dato de prueba en la sala 2
```

- 9) El script de Python usa la librería Kazoo para crear un objeto del cliente. Se conecta llamando el método `start()`, luego asegura la ruta con el método `ensure_path(ruta)`, crea un nodo con los valores 1) el parametro que se pasó al ejecutar el script 2) `ephemeral = True`, lo cual hace que el nodo se elimine al momento en el que Zookeeper detecte que la sesión asociada al nodo ha expirado, 3) `sequence= True`,

lo cual hace que los nodos que se creen tengan un valor secuencial de 10 dígitos y cada uno con un incremento de 1 con respecto al anterior. Luego está el decorador ChildrenWatch(ruta), el cual se encarga de observar los cambios en la ruta dada, en este caso imprime los hijos del nodo principal. El proposito de este script es observar los nodos que se crean o eliminan e imprimir los nodos existentes así como se puede observar en los screenshots, claramente el primer nodo muestra toda la actividad y el último no porque no existía cuando se crearon los demás.

```
Windows PowerShell
PS D:\> python whatever2.py nodo01
#####
nodo01
#####
#####
nodo02
nodo01
#####
#####
nodo02
nodo03
nodo01
#####

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> cd d:
PS D:\> python whatever2.py nodo02
#####
nodo02
nodo01
#####
#####
nodo02
nodo03

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> cd d:
PS D:\> python whatever2.py nodo03
#####
nodo02
nodo03
nodo01
#####
```