Administración de sistemas II

Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Despliegue de sistemas distribuidos básicos de red

Francisco Javier Pizarro Martínez 821259

08/03/2023

Índice

Objetivos de la práctica

En esta práctica se desean 2 objetivos principales:

- Implementar los servicios distribuidos básicos necesarios en una red para poder desplegar aplicaciones más complejas en la misma.
- Automatización de tareas sobre sistemas heterogéneos.
- Ser capaces de acceder mediante un DNS público a la subred.

Para el primer objetivo se van a implementar 2 nuevas MVs así como a redefinir partes de las MVs anteriores. Concretamente se va a implementar un servidor NTP, un servidor principal DNS para la zona de la subred y un servidor UNBOUND que va a ser el encargado de gestionar las peticiones de todas las máquinas cacheando estas, concretamente si recibe una petición de una dirección interna de la zona, se pondrá en contacto con el SOA de la zona que es el DNS previamente mencionado y en caso de que la petición desee conocer una dirección que sea externa a nuestra zona la redirigirá al DNS de la zona maestra de esta.

Para el segundo objetivo se van a emplear varios scripts desarrollados en Ruby y en Shell los 3 principales van a tener estas funcionalidades:

- Encender remotamente todas las MVs
- Apagar remotamente todas las MVs
- Ejecutar comandos/ping sobre todas las MVs

Para el correcto funcionamiento de dichos scripts adicionalmente se debe configurar el salto de usar una clave publica para identificarse en el ssh.

Para el tercer objetivo solo se debe configurar de forma adecuada los servidores DNS y UNBOUND de la red de forma que estos funcionen de forma adecuada con los glue records.

Es esencial lograr el correcto funcionamiento de los servicios distribuidos básicos ya que si estos fallan pueden dar lugar a problemas en aplicaciones más complejas que dependen de ellos, por lo que es muy importante además de implementarlos, probar su correcto funcionamiento de forma adecuada.

Diseño a alto nivel

Para lograr alcanzar la red deseada así como las funcionalidades que se esperan de la misma se deben realizar los siguientes pasos:

Crear las nuevas máquinas virtuales 3 y 4, así como modificar la máquina virtual 2 para que esta tenga una IP estática asociada.

Crear y testear los scripts de automatización así como modificar lo necesario en las máquinas virtuales para que estos funcionen de forma adecuada.

Poner en funcionamiento el servidor NTP en la máquina virtual 2, una vez hecho esto, comprobar su correcto funcionamiento y configurar las demás maquinas virtuales para que tengan dicho servidor como servidor NTP de referencia.

Poner en funcionamiento el servicio nsd en la máquina virtual 3, realizar la configuración general de dicho servicio, comprobar que esta es correcta y aplicarla, posteriormente realizar la configuración de la zona directa, comprobar su correcta implementación y recargar la configuración de zona para que los cambios surtan efecto, realizar todo lo anterior para la zona inversa de la misma forma, una vez todo esto se haya realizado, ejecutar tests con el comando dig -6 para comprobar el correcto funcionamiento del DNS.

Poner en funcionamiento el servicio unbound en la máquina virtual 4, configurar dicho servicio de forma adecuada y recargar el mismo para aplicar los cambios, testear con el comando dig -6 que este funciona adecuadamente.

Configurar en todas las MVs de la red el servidor de la máquina 4 como servidor DNS principal(en la MV 4 también).

Mapa de red

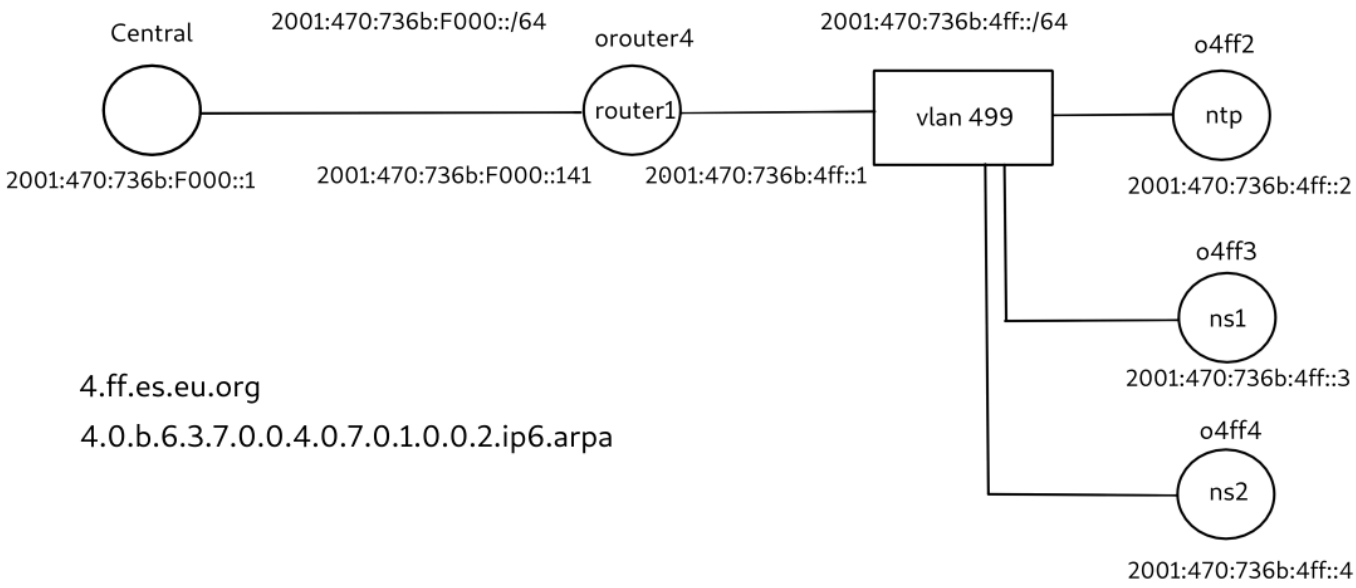

Tabla IPAM de servicios básicos de red

Nombre máquina	Nombre DNS	IPv6
Central	central.cps.unizar.es	2001:470:736b:f000::1
orouter4	router1.4.ff.es.eu.org	2001:470:736b:4ff::1
o4ff2	ntp.4.ff.es.eu.org	2001:470:736b:4ff::2
o4ff3	ns1.4.ff.es.eu.org	2001:470:736b:4ff::3
o4ff4	ns2.4.ff.es.eu.org	2001:470:736b:4ff::4

Scripts de automatización

Encendido remoto

Para este script no existe ninguna complicación ya que recurre al comando virsh empleando qemu+ssh.

```
#!/usr/bin/bash
for mv in {orouter4,o4ff{2..4}}
do
    echo "Encendiendo la máquina $mv"
    virsh -c qemu+ssh://a821259@155.210.154.207/system start $mv
done
```

Apagado remoto

Este script aparte de la interacción con virsh tiene que tener en cuenta un aspecto adicional y es que dado que en OpenBSD la consistencia del sistema de ficheros frente a apagados inesperados es débil, antes de forzar el apagado mediante virsh, debemos apagar internamente todas las MVs así como esperar unos segundos para que se estabilicen los sistemas de ficheros evitando así corromper las MVs, otro aspecto que debe tener en cuenta es que la máquina orouter4 debe ser la última en apagarse.

```
#!/usr/bin/bash
for mv in {4..1}
do
ssh -6 -n a821259@2001:470:736b:4ff::${mv} "doas shutdown -h now"
    echo "Apagando internamente la máquina $mv"
done
sleep 10
for mv in {o4ff{2..4},orouter4}
do
    virsh -c qemu+ssh://a821259@155.210.154.207/system destroy $mv --graceful
    echo "Apagando forzadamente la máquina $mv"
done
```

Cuestiones de agilizado general

Para aumentar la productividad y la velocidad de desarrollo de la práctica se han empleado técnicas adicionales tales como:

- Definir como variables de entorno las IPv6 estáticas asociadas a cada máquina en central(esto se hizo cuando aún no había dns).
- Emplear las terminales de terminator con difusión general para ejecutar comandos sobre todas las MVs,
- Emplear terminales remotas y no la interfaz gráfica proporcionada por virt-manager.
- Para mejorar el workflow tomar las anotaciones de las configuraciones realizadas sobre markdown dado que además de ser rápido y eficiente permite copiar fragmentos de configuración o de código lo cual resulta muy útil.

Variables de entorno empleadas en central para agilizar los saltos por ssh sin depender del DNS:

> export as2Fol="/misc/alumnos/as2/as22022/a821259"
> export subRedAS2="2001:470:736b:4ff::"
> export router1="$subRedAS21" "$exportntp = "{subRedAS2}2"
> export ns1="$subRedAS23" "$exportns2 = "{subRedAS2}4"

Monitorización de máquinas y ejecución remota de comandos

Este script es algo más complejo ya que tiene varios aspectos más refinados que los anteriores.

El primero de ellos es que contiene instrucciones de ejecución y flags de ayuda,
Además en cuenta de emplear una expansión hardcodeada para conocer todo el conjunto de MVs recurre a un fichero de configuración definido en **~/.u/host** ,
El programa en su versión actual ofrece dos opciones: La primera de ellas es ejecutar un ping sobre todas las MVs, la segunda opción es ejecutar un comando via ssh en todas las MVs mostrando para cada una la salida estándar y la de error.

Ejemplos de uso:

```
u p
u s "ifconfig"
u --help
```

Para poder ejecutar el script en las MVs de OpenBSD se deben ejecutar los siguientes comandos en la máquina donde quiera ser ejecutado estos descargan ruby y las gemas necesarias:

```
pkg_add ruby
gem install net-ping --user-install
gem install net-ssh --version 3.0.1 --user-install
```

La implementación concreta del script se encuentra en el Anexo 2 de este archivo.

NTP

Configuración servidor

Añadimos en el fichero /etc/rc.conf.local el contenido:

> ntpd_flags=-s

Comentamos los servidores del fichero /etc/ntpd.conf y escribimos lo siguiente:

> server 2001:470:0:50::2
> server 2001:470:0:2c8::2
> listen on 2001:470:736b:4ff::2

Para habilitar el servicio:

```
ntpd
```

En caso de haber habilitado ya el servicio:

```
rcctl restart ntpd
```

Para comprobar que el servicio ntpd esta bien configurado:

```
ntpctl -s all
```

Para comprobar que funciona correctamente, ejecutar en central los siguientes comandos:

```
ntpdate -q ntp.unizar.es
ntpdate -q 2001:470:736b:4ff::2
```

Configurar clientes

Comentamos los servidores del fichero /etc/ntpd.conf y escribimos lo siguiente:

> server 2001:470:736b:4ff::2

Ejecutamos el comando:

```
ntpd
```

Añadimos en el fichero /etc/rc.conf.local el siguiente contenido:

> ntpd_flags=-s

DNS

Configurar servidor maestro

Iniciar servicio y configuraciones generales

Ejecutamos los siguientes comandos para el setup inicial del dns:

```
rcctl enable nsd
nsd-control-setup
```

Creamos la carpeta que va a contener la información de las zonefiles:

```
cd /var/nsd/zones/
touch 4.ff.es.eu.org.inverso
touch 4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa.
```

Si en algún momento aplicando las siguientes intrucciones llegamos al error "/var/run/nsd.sock no existe" -> Realizar los siguientes comandos para arreglarlo:

```
touch /var/run/nsd.sock
chown _nsd:wheel /var/run/nsd.sock
```

Modificamos el fichero /var/nsd/etc/nsd.conf y escribimos lo siguiente:

```
server:
        hide-version: yes
        verbosity: 1
        database: "/var/nsd/db/nsd.db"
        username: _nsd
        logfile: "/var/log/nsd.log"
        pidfile: "/var/nsd/run/nsd.pid"
        port: 53
        server-count: 1
        ip6-only: yes
        zonesdir: "/var/nsd/zones"

remote-control:
        control-enable: yes
        control-interface: /var/run/nsd.sock

zone:
        name: "4.ff.es.eu.org"
        zonefile: "4.ff.es.eu.org.directo"

zone:
        name: "4.0.b.6.3.7.0.7.4.0.1.0.0.2.ip6.arpa."
        zonefile: "4.0.b.6.3.7.0.7.4.0.1.0.0.2.ip6.arpa."
```

Comprobamos que la configuración es correcta con el comando:

```
nsd-checkconf /var/nsd/etc/nsd.conf
```

Configurar zona directa

Modificamos el fichero /var/nsd/zones/4.ff.es.eu.org.directo y escribimos lo siguiente:

```
$ORIGIN 4.ff.es.eu.org.
$TTL 86400
@       IN      SOA     ns1.4.ff.es.eu.org. root.4.ff.es.eu.org (
        0000000
        28800
        7200
        864000
        86400
        )
        NS      ns1.4.ff.es.eu.org.

ns1     IN      AAAA    2001:470:736b:4ff::3
ns2     IN      AAAA    2001:470:736b:4ff::4
router1 IN      AAAA    2001:470:736b:4ff::1
ntp     IN      AAAA    2001:470:736b:4ff::2
```

Comprobamos que la configuración es correcta con el comando:

```
nsd-checkzone 4.ff.es.eu.org /var/nsd/zones/4.ff.es.eu.org.directo
```

Aplicamos la nueva configuración con el comando:

```
nsd-control reconfig
nsd-control reload 4.ff.es.eu.org.directo
```

Comprobamos que todo funciona como debería con los siguientes comandos:

```
nsd-control zonestatus 4.ff.es.eu.org
dig -6 @2001:470:736b:4ff::3 ns2.4.ff.es.eu.org
```

Configurar zona indirecta

Modificamos el fichero /var/nsd/zones/4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa. y escribimos lo siguiente:

```
$ORIGIN 4.0.b.6.3.7.0.7.4.0.1.0.0.2.ip6.arpa.
$TTL 86400
@       IN      SOA     ns1.4.ff.es.eu.org. root.4.ff.es.eu.org. (
                0000001
                28800
                7200
                864000
                86400
                )
                NS      ns1.4.ff.es.eu.org.

1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.f.f       IN      PTR     router1.4.ff.es.eu.org.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.f.f       IN      PTR     ntp.4.ff.es.eu.org.
3.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.f.f        IN      PTR     ns1.4.ff.es.eu.org.
4.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.f.f        IN      PTR     ns2.4.ff.es.eu.org.
```

Comprobamos que la configuración es correcta con el comando:

```
nsd-checkzone 4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa. /var/nsd/zones/4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa.
```

Aplicamos la nueva configuración con los comandos:

```
nsd-control reconfig
nsd-control reload 4.0.b.6.3.7.0.0.0.7.4.0.1.0.0.2.ip6.arpa.
```

Comprobamos que todo funciona como debería con los siguientes comandos:

```
nsd-control zonestatus 4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa.
dig -6 @2001:470:736b:4ff::3 -x 2001:470:736b:4ff::1
```

Configurar servidor unbound

Iniciamos el servicio y realizamos su configuración inicial con los siguientes comados:

```
rcctl enable unbound
unbound-control-setup
```

Modificamos el fichero /var/unbound/etc/unbound.conf y escribimos lo siguiente:

```
server:
        interface: 0.0.0.0
        interface: ::0
        interface: ::1
        do-ip6: yes
        access-control: 0.0.0.0/0 refuse
        access-control: 2001:470:736b::/48 allow
        access-control: 127.0.0.0/8 allow
        access-control: ::0/0 refuse
        access-control: ::1 allow
        hide-identity: yes
        hide-version: yes
        val-log-level: 2
        aggressive-nsec: yes

remote-control:
        control-enable: yes
        control-interface: /var/run/unbound.sock

stub-zone:
        name:"4.ff.es.eu.org"
        stub-addr: 2001:470:736b:4ff::3
        stub-first:yes

stub-zone:
        name:"4.0.b.6.3.7.0.0.4.0.7.0.1.0.0.2.ip6.arpa."
        stub-addr: 2001:470:736b:4ff::3
        stub-first:yes
forward-zone:
        name: "."                       # use for ALL queries
        forward-addr: 2001:470:20::2    # example address only
        forward-first: yes              # try direct if forwarder fails
```

Comprobamos que la configuración es correcta y reiniciamos el servicio con la nueva configuración ejecutando los siguientes comandos:

```
unbound-checkconf
unbound-control reload
```

En caso de tener el error "unbound.sock doesnt exist" ejecutar los siguientes comandos:

```
touch /var/run/unbound.sock
chown _nsd:wheel /var/run/unbound.sock
```

Configurar clientes

Realizar esto en todas las máquinas
Editar el fichero /etc/resolv.conf y escribir lo siguiente:

> nameserver 2001:470:736b:4ff::4

Para aplicar los cambios ejecutar el comando:

```
sh /etc/netstart
```

Para comprobar que todo funciona correctamente ejecutar los comandos:

```
ping6 router1.4.ff.es.eu.org
ping6 ipv6.google.com
```

Pruebas realizadas

Para realizar las pruebas se deben ejecutar los siguientes comandos en central, el objetivo de las pruebas es comprobar el correcto funcionamiento desde la propia de red de central tanto de resolución directa como de resolución inversa, adicionalmente comprobamos el correcto funcionamiento desde una red externa mediante el DNS de google comprobando así la correción de la configuración de los glue-records

```
nslookup -type=AAAA router1.4.ff.es.eu.org 2001:470:736bb:4ff::4
dig -6 @2001:470:736b:4ff::4 router1.4.ff.es.eu.org
dig -6 @2001:470:736b:4ff::4 -x 2001:470:736b:4ff::1
host -t PTR 2001:470:736b:4ff::1 2001:470:736b:4ff::4
# ahora empleando los dns de google para comprobar que la red es localizable desde el exterior
dig -6 @2001:4860:4860::8888 AAAA router1.4.ff.es.eu.org
dig -6 @2001:4860:4860::8888 -x 2001:470:736b:4ff::1
```

ANEXO 1

Creación y setup de las máquinas 3 y 4

```
qemu-img create -f qcow2 -o backing_file=o4.qcow2 o4ff3.qcow2
qemu-img create -f qcow2 -o backing_file=o4.qcow2 o4ff4.qcow2
chmod g+w o4ff3.qcow2 o4ff4.qcow2
cp o4ff2.xml o4ff3.xml
cp o4ff2.xml o4ff4.xml
```

Editamos los xml modificando los siguientes flag:

- uuid
- nombre
- source
- MAC

Dentro de ambas MV realizamos los siguientes pasos
Z es la de cada máquina

Definimos las MV

```
virsh -c qemu:///system define o4ff3.xml
virsh -c qemu:///system define o4ff4.xml
```

Editamos el contenido del fichero /etc/hostname.vio0 y escribimos el siguiente:

> up
> -inet6

Creamos el fiichero /etc/hostname.vlan499 y su contenido es el siguiente:

> vlan 499 vlandev vio0 up
> inet6 2001:470:736b:4ff:Z
> inet6 -temporary
> inet6 -soii

En el fichero /etc/myname escribimos lo siguiente:

> o4ffZ

Añadimos en /etc/mygate

> 2001:470:736b:4ff::1

Ejecutamos el siguiente comando para reiniciar y aplicar todos los cambios:

```
sh /etc/netstart
```

Comprobamos que las conexiones funcionan mediante el uso de ping6.

Cambios máquina 2

Queremos que la máquina ntp tenga una ip6 estática así que editamos el fichero /etc/hostname.vlan499 y cambiamos la linea:

"inet6 autoconf" -> "inet6 2001:470:736b:4ff:Z"

Anexo 2

Código fuente completo de la herramienta de monitorización y ejecución remota de comandos

El funcionamiento de la herramienta se ha explicado previamente, su implementación consta de fases diferencias:

1. Cargar gemas.
2. Comprobar flags de ejecución.
3. Cargar IPs del fichero ~/.u/hosts.
4. Crear función interna dependiendo del comando introducido.
5. Crear función que se va a aplicar sobre todas las IPs.
6. Aplicar la función sobre cada una de nuestras IPs.

```
#!/usr/bin/ruby -w
require 'net/ping/tcp'
require "net/ssh"
require 'optparse'
OptionParser.new do |opts|
  opts.banner = "Uso: u [p | s 'comando en shell']"
  opts.on("-h", "--h", "-help", "--help") do |h|
    puts "Configurar previamente en ~/.u/hosts un fichero con una IP por linea
Uso: u [p | s 'comando en shell']
Este programa con la opción p realiza un ping a dichas IPs
Con la opción s ejecuta via ssh el comando introducido en dichas IPs"
  end
end.parse!

f = File.open(ENV["HOME"] + "/.u/hosts")
ips = f.readlines.map(&:chomp)
f.close
case ARGV[0]
when "p"
    puts "Ejecutando ping sobre las máquinas definidas:"
        def accionInterna(ip,t)
                puts "#{t.host}:Funcionando con respuesta en #{t.duration} segundos"
    end
when "s"
    puts "Ejecutando el comando #{ARGV[1]} sobre las máquinas definidas via ssh"
    def accionInterna(ip,t)
        puts "Ejecutando en: #{t.host}"
                ssh = Net::SSH.start(ip, "a821259",:password => "AquíVaTuContraseña")
                output = ssh.exec!(ARGV[1])
                puts output
    end
else
    puts "Uso: u [p | s 'comando en shell']"
end

def accion(ip)
        t = Net::Ping::TCP.new("#{ip}",22,0.1) # puerto 22, timeout 0.02 s
        if t.ping?
                accionInterna(ip,t)
        else
                puts "#{t.host}:Falla"
        end
end

for ip in ips do
    accion(ip)
end
```