

Nicolás David Rincón Ballesteros  
Ingeniería de Sistemas. Pontificia Universidad Javeriana  
Bogotá, Colombia  
ni.rincon @javeriana.edu.co

Juan Diego Palacios Toledo  
Ingeniería de Sistemas. Pontificia Universidad Javeriana  
Bogotá, Colombia  
ju.palacios@javeriana.edu.co

**Abstract**—The DNS was created for the protection of data when traveling over the Internet with the TCP/IP protocol. This article provides relevant information about the history, important functions, and an example of a DNS Server that will be presented using GNS3 and VirtualBox, with their respective configuration and tests.

**Keywords**—DNS Server, DNS Protocol, TCP/IP, GNS3, Oracle VM Virtualbox, RFC 1035.

## I. INTRODUCCIÓN

Todo dispositivo electrónico capaz de conectarse a cualquier sitio web usa el sistema DNS (Domain Name System) el cual se comunican entre sí por medio del uso de números los son conocidos como direcciones IP, con esto no es necesario que al querer ingresar a cualquier búsqueda se tenga que poner números si no que solo un dominio como “ejemplo.com”. Estos convertidos por los servidores DNS a IP como el 192.0.2.1 el cual se controla a que destino se dirige el usuario.

Esto gracias a los puertos, están diferenciados por números a lo cual el sistema DNS usa el puerto 53 que permite utilizar los servicios DNS a su vez el uso del protocolo TCP como el UDP para la comunicación hacia los servidores.[1]

## II. HISTORIA

DNS parte para optimizar ARPANET el cual nace en los 70s con el departamento de defensa de los EE.UU con el objetivo de eliminar las dependencias de un ordenador central distribuyendo varias computadores en algunas empresas, universidades y entidades de gobierno también se sabe que ARPANET no contaba con el protocolo TCP/IP si no que utilizaba el protocolo NCP (Network Control Protocol) con un total de 256 direcciones disponibles las cuales era normalmente llevadas a mano, llega 1978 se extiende ARPANET con conexiones en Noruega e Inglaterra nace a su vez el cambio del protocolo al TCP/IP.[2]

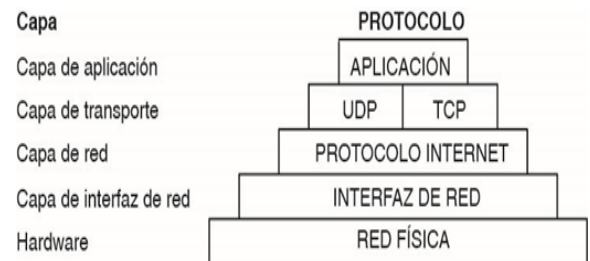


Figura 1. Conjunto de protocolo TCP/IP [3].

“TCP/IP define cuidadosamente cómo se mueve la información desde el remitente hasta el destinatario. En primer lugar, los programas de aplicación envían mensajes o corrientes de datos a uno de los protocolos de la capa de transporte de Internet, **UDP (User Datagram Protocol)** o **TCP (Transmission Control Protocol)**. Estos protocolos reciben los datos de la aplicación, los dividen en partes más pequeñas llamadas paquetes, añaden una dirección de destino y, a continuación, pasan los paquetes a la siguiente capa de protocolo, la capa de red de Internet.

La capa de red de Internet pone el paquete en un datagrama de IP (Internet Protocol), pone la cabecera y la cola de datagrama, decide dónde enviar el datagrama (directamente a un destino o a una pasarela) y pasa el datagrama a la capa de interfaz de red.

La capa de interfaz de red acepta los datagramas IP y los transmite como tramas a través de un hardware de red específico, por ejemplo, redes Ethernet o de Red en anillo.” [2], [3].

La idea era aumentar la cantidad de computadoras conectadas a la red a su vez dada la alta demanda de servidores se crea un único archivo “host.txt” el cual estaba siendo mantenido por la universidad de Stanford. Este archivo era mantenido a mano y las solicitudes para agregar un registro se hacían mediante el teléfono en horas de trabajo.

Llegado el 1983 el archivo Host se volvió muy lento el cual no podía seguir los cambios que surgían en la red. El ingeniero John Postel el cual era el encargado de del mantenimiento del archivo host con su colega Paul Boca Pettis estaban en busca de mejorar este sistema de Host con la idea de 5 diferentes



DNS en varias maneras. La siguiente figura es una de las configuraciones más típicas y simples que existen.

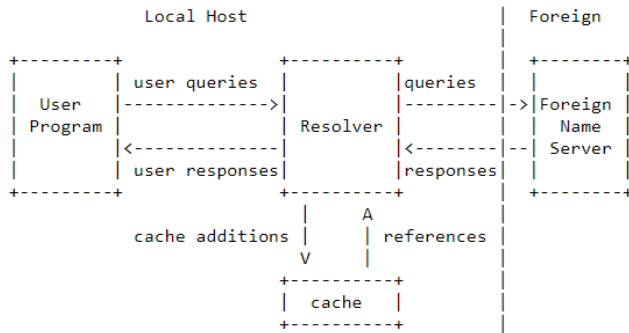


Figura 5. DNS con cache [5].

En esta configuración, los programas de usuario interactúan con el espacio de nombres de dominio a través de Resolvers. En esta configuración, las consultas suelen ser llamadas desde el sistema operativo y el Resolver con respectivo cache, que hacen parte del sistema operativo del host. En pocas palabras, los Resolvers responden a las consultas de los usuarios con información adquirida de otros **Foreign Name Server** y la de su **cache local**. [5]

Otra configuración es en el caso que servidores designados secundarios consigan zonas y busquen por actualizaciones desde el servidor primario usando el protocolo de transmisión de zona de DNS. La siguiente figura ilustra esa configuración.

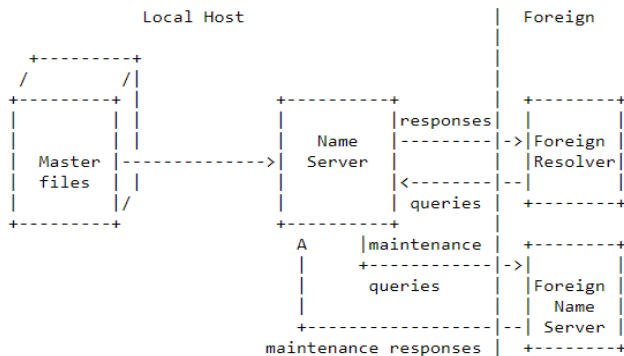


Figura 6. DNS con dos Foreign Name Server [5].

En esta configuración, se establece periódicamente un circuito virtual a un Foreign Name Server para adquirir una copia de la zona y revisar que esta no ha cambiado. Esta se basa en consultas y respuestas de mantenimiento.

### C. Formato y sintaxis nombres de dominio

Al momento de asignar nombres de dominio, estas ya sean de mail, se deben seguir ciertas normas especificadas en el **RFC-1035**. Esto para que cualquier nombre de dominio pueda ser expresado en notación DNS, para así ser compatible con nomenclaturas de hosts existentes. La siguiente figura muestra el formato:

```
<domain> ::= <subdomain> | " "
<subdomain> ::= <label> | <subdomain> "." <label>
<label> ::= <letter> [ [ <ldh-str> ] <let-dig> ]
<ldh-str> ::= <let-dig-hyp> | <let-dig-hyp> <ldh-str>
<let-dig-hyp> ::= <let-dig> | "-"
<let-dig> ::= <letter> | <digit>
<letter> ::= any one of the 52 alphabetic characters A through Z in
upper case and a through z in lower case
<digit> ::= any one of the ten digits 0 through 9
```

Figura 7. Formato nombres de dominio [5].

Las letras mayúsculas y minúsculas, aunque estén permitidas, no tienen importancia en los nombres de dominio. Esto quiere decir que dos nombres con las mismas letras, pero distintas letras mayúsculas y minúsculas, serán tratadas como iguales [5].

Se siguen las reglas de **ARPANET** para nombres de host. Estas reglas nos dicen que deben iniciar con una letra, terminar con una letra o dígito y que los caracteres interiores solo pueden ser letras, números y guiones.

### D. Modo de almacenar la información en DNS

Los servidores de nombres de dominio almacenan la información en unos registros llamados *Resource Record* (RR), estos registros tienen su formato especial y son representados como una secuencia de etiquetas. Estas están compuestas por un byte que indica la longitud de bytes de esta etiqueta y tienen las siguientes limitaciones:

- El nombre de dominio debe terminar por un byte de longitud de valor 0.
- Los dos bits de mayor valor de posición de cada uno de los octetos (byte), debe ser 0, esto limitando el valor de cada etiqueta a 63 o menos por octeto.
- La longitud total de un nombre de dominio está restringida a 255 octetos o menos.

Como fue mencionado anteriormente, se debe obviar la diferencia entre mayúsculas y minúsculas en los *Name Servers* y *Resolvers*. Esto quiere decir que trabajaran de una forma case-insensitive (obviando las mayúsculas y minúsculas). Ejemplo: B=b.

La siguiente figura ilustra el formato RR:

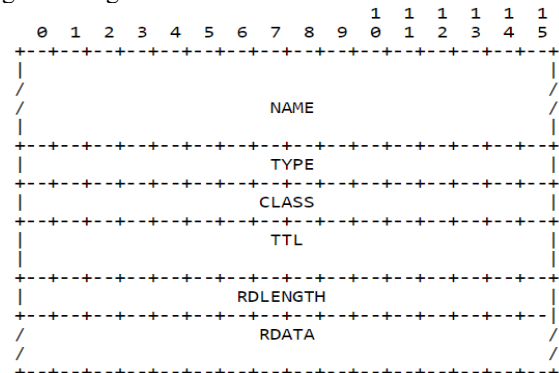


Figura 8. Formato RR. [5]

**NAME:** Nombre del nodo del cual este registro pertenece [5].

**TYPE:** Dos (2) octetos conteniendo uno de los códigos TYPE (tipo) de RR [4].

**CLASS:** Dos (2) octetos conteniendo uno de los códigos CLASS (clase) de RR [5].

**TTL:** Especifica el intervalo de tiempo que el registro de recursos puede almacenar en cache antes de que se convierta en 0 [5].

**RDLLENGTH:** Entero de 16 bits no asignado que especifica la longitud en octetos del campo RDATA [5].

**RDATA:** Longitud variable de strings de octetos que describe el recurso. El formato de esta depende del TYPE y CLASS de este RR [5].

#### E. Elementos importantes del DNS

**Domain Name Space (Espacio de Nombres de Dominio):** El encargado de guardar los nombres de dominio. Tienen una particularidad, estos se expresan en términos de una secuencia de etiquetas. Cada una con un octeto de longitud, seguido por el número de octetos [5].

**Name Servers (Servidores de Nombres):** Manejan dos tipos de datos. La primera son las llamadas zonas mencionadas anteriormente, cada una de estas zonas es una base de datos particular a cada una de las ramas del Domain Space. Este regularmente revisa si sus zonas están actualizadas, y si este no es el caso, consigue una nueva copia de las zonas actualizadas [5].

**Resolvers:** Son quienes se aprenden los contenidos de otros Name Servers y son los responsables de tratar con la distribución del Domain Space y los efectos de una posible falla del Name Server, consultando constantemente bases de datos redundantes en otros servidores [5].

**Master Files:** Estos son archivos de texto con registros de datos (RRs) en formato de texto. Sera explicado más a detalle más adelante [5].

#### F. Formato Mensajes DNS

DNS funciona gracias a peticiones y respuestas, la siguiente figura ilustra el formato de estos mensajes.

Header	
Question	the question for the name server
Answer	RRs answering the question
Authority	RRs pointing toward an authority
Additional	RRs holding additional information

Figura 9. Formato mensajes DNS [5].

Como se ve en la figura, este está dividido en cinco (5) secciones, que en algunos casos pueden estar vacías.

- **Header:** Incluye campos que especifican cuales de las secciones están presentes, si es un mensaje de consulta o respuesta, si es una consulta estándar, entre otras cosas [5].
- **Question:** Contiene campos que describen el tipo de pregunta al servidor de nombres (Name Server), estas pueden ser QTYPE (consulta de tipo), QCLASS (consulta de clase) y QNAME (consulta de nombre de dominio) [5].
- **Answer:** Posible lista de RRs concatenados como respuesta de la pregunta [5].
- **Authority:** RRs que apuntan hacia el Servidor de Nombres autoritativo [5].
- **Additional:** RRs que se relacionan a la consulta, pero no son respuestas de la pregunta [5].

La siguiente figura es el formato de un **Header**.

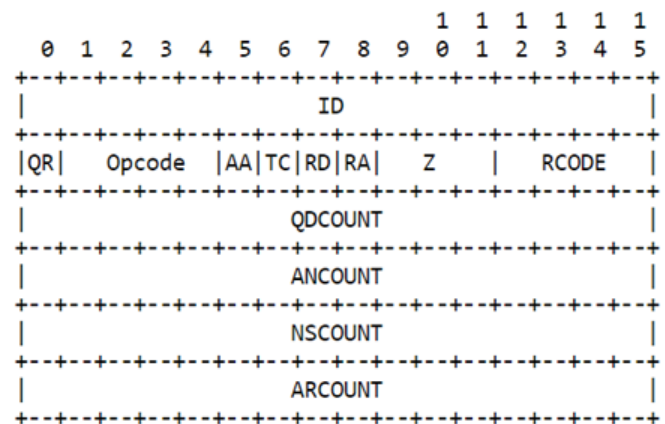


Figura 10. Campos de un Header [5].

La siguiente figura es el formato de **Pregunta (Question)**.

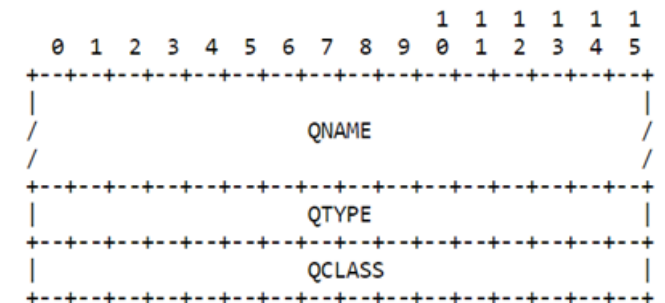


Figura 11. Campos de una Pregunta [5].

**QNAME** es el dominio que se quiere buscar al momento de hacer la consulta representado como una secuencia de etiquetas, **QTYPE** es el tipo consulta que se está haciendo y **QCLASS** es un código de dos (2) octetos que especifica la clase de la consulta, por ejemplo, **IN** se refiere a Internet [5].

#### G. Master Files

Estos son archivos de texto que contienen **RRs** en forma de texto. El formato de estos son una secuencia de entradas, orientadas en líneas. La combinación de **espacios** o **TABS** se usa para separar ítems que componen una entrada. Al dentro

de estos archivos se tienen varios **RRs**, una función importante que cumplen estos es la definición de zonas como la expresión de esta lista de **RRs** [5].

La siguiente figura muestra los campos de posibles entradas:

```
$ORIGIN <domain-name> [<comment>]
$INCLUDE <file-name> [<domain-name>] [<comment>]
<domain-name><rr> [<comment>]
<blank><rr> [<comment>]
```

Figura 12. Posibles entradas a un Master File [5].

Hay dos entidades de control **\$ORIGIN** y **\$INCLUDE**. **\$ORIGIN** reinicia el origen actual al nombre de dominio especificado. **\$INCLUDE** inserta el nombre de archivo especificado al archivo actual del Master File, también se le puede adicionar un comentario.

**<rr>** puede tener una de estas dos formas:

```
[<TTL>] [<class>] <type> <RDATA>
[<class>] [<TTL>] <type> <RDATA>
```

Figura 13. Posibles formas **<rr>** [5].

Los nombres de dominio que acaben en “.” son llamados absolutos y se consideran como completos, aquellos que no tengan el punto son considerados como relativos.

#### A. Manejo Servidor de Nombres

Un servidor de nombres debe ejecutar múltiples actividades de forma concurrente. Simplemente no es aceptable que un servidor de nombres bloquee el servicio de solicitudes de **UDP** mientras espera a datos **TCP** para refrescarse o realizar actividades de consulta.

Aunque se puede utilizar cualquier estructura para guardar los datos, el **RFC 1035** sugiere la siguiente base de datos:

- Catálogo con una lista de zonas disponibles en el servidor con su respectivo puntero.
- Estructuras de datos por cada zona del servidor de nombres.
- Estructura de datos para la información de cache.

La información **TTL** de los datos para los **RRs** y el cronometraje de los datos para refrescar actividades depende de temporizadores de 32 bits en segundos.

## IV. ESCENARIO PARA PRUEBAS

### A. Primer vistazo

La siguiente figura es el primer bosquejo de la topología a usar para el desarrollo del proyecto elaborada en el programa **Cisco Packet Tracer**.

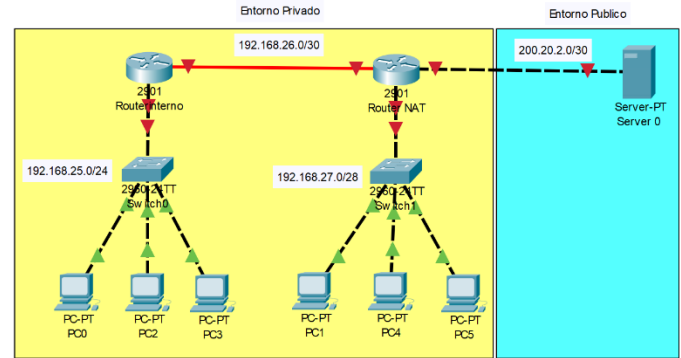


Figura 14. Primer Bosquejo Topología. Fuente: Elaboración Propia

Tras haber realizado el bosquejo, del escenario de pruebas de este servidor DNS, se decidió utilizar el programa **GNS3** para la creación de la topología, en conjunto con el programa **Oracle VM VirtualBox** para la utilización de máquinas virtuales.

Dentro de este escenario de pruebas se utilizarán 4 redes distintas y los principales dispositivos que se manejarán en son 4 tipos: **Routers**, **Switches**, **Máquinas Virtuales** y **Servidores**.

Para los routers, se decidió utilizar el **router cisco c3725**. De estos, se necesitan dos y respectivamente en la topología se llaman: Router Interno y NAT. Los switches utilizados, son los switches por defecto incluidos en **GNS3**, de estos se necesitan dos (2) y son denominados **Ethernet Switch**.

Los switches también son parte fundamental de este escenario ya que se utilizan principalmente para simular que tenemos conectados varios hosts por red, este número de hosts representado por la máscara (prefijo) de cada una de estas redes. Sin embargo, debido a un gran consumo de recursos de nuestro dispositivo físico al usar máquinas virtuales, se decidió solo utilizar una máquina virtual por red.

Hablando de las máquinas virtuales, se utilizarán dos (2) de estas, una siendo un Windows 7 y otra siendo un Linux Mint. Cada una de estas está conectada al switch por la interfaz ethernet de cada dispositivo. Estas máquinas virtuales fueron instaladas en el programa **Oracle VM VirtualBox** y fueron enlazadas mediante una plantilla dentro del programa **GNS3** dentro de la parte de “*VirtualBox VMs Templates*”. Esto será explicado con más detalle en la siguiente sección.

El último tipo de dispositivo que se utilizara en la topología es un servidor Ubuntu en el cual se integrara un Servidor Apache para su codificación DNS.

Todas estas máquinas virtuales mencionadas e imágenes de routers fueron descargadas de internet y funcionan como lo haría un dispositivo físico de la vida real.



Hablando principalmente de las conexiones, todas son mediante **FastEthernet** en cada uno de los dispositivos. La siguiente figura es la topología para las pruebas en **GNS3**.

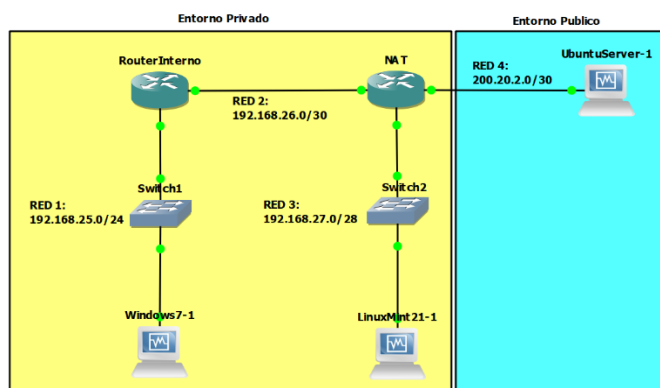


Figura 15. Escenario de Pruebas en GNS3. Fuente: Elaboración propia

Como se puede ver, este escenario cuenta de dos entornos, uno privado y otro público. Asimismo, consta de 4 redes distintas. Las tres (3) primeras se encuentran en la parte privada y la cuarta (4) en el entorno público en el cual está el servidor DNS.

Las redes privadas se basan en el **RFC 1918**, que se refiere a las direcciones IP privadas, que en este caso se decidió poner de clase C. El rango de estas consta desde la dirección **192.168.0.0** hasta la dirección **192.168.255.255**. La siguiente es el número de hosts que puede albergar cada una de estas redes privadas. [8]

1. **Red 1: 192.168.25.0/24:** Esta al ser prefijo 24, tiene un número total de direcciones IP de 256, sin embargo, las realmente utilizables son 254. Esto debido a que se le resta la dirección de subred y broadcast.
2. **Red 2: 192.168.26.0/30:** Esta al ser prefijo 30, tiene un número total de direcciones IP de 4, sin embargo, las realmente utilizables son 2. Esto debido a que se le resta la dirección de subred y broadcast. Esta es de prefijo 30 porque es una conexión punto a punto entre dos routers.
3. **Red 3: 192.168.27.0/28:** Esta al ser prefijo 28, tiene un número total de direcciones IP de 16, sin embargo, las realmente utilizables son 14. Esto debido a que se le resta la dirección de subred y broadcast.

La única red pública que se utiliza es la que conecta directamente el router NAT y el Ubuntu Server, esta es la **Red 4: 200.20.2.0/30** clase C. Esta también es de prefijo 30 y consta de un total de 4 direcciones IP, dentro de las cuales 2 son utilizables. Ergo, es una conexión punto a punto entre el router NAT y el Ubuntu Server.

Cabe recalcar que en este escenario de pruebas también consideraremos el protocolo cliente/servidor DHCP para una asignación automática de direcciones IP para los hosts dentro del entorno privado, estos hosts siendo la máquina virtual

Windows 7 en la red 1 y la máquina virtual Linux Min en la red 3. [9]

Ya que estamos hablando de un entorno privado y un entorno público, es necesario un NAT o por sus siglas en ingles **Network Address Translation**. Este es un mecanismo en el cual se traducen direcciones privadas a direcciones públicas para poder trabajar en internet. El dispositivo NAT, que en nuestro caso es el router "NAT", opera en el borde de una red tipo **STUB**, este con una sola conexión a la red del servidor. Existen tres (3) tipos de NAT, el **NAT estático**, **NAT dinámico** y **PAT**. En nuestro escenario de pruebas se decidió utilizar el **PAT** o por sus siglas en ingles **Port Address Translation**, esto se especificará más a detalle en la siguiente sección.

En la segmentación de nuestro entorno se considerará el protocolo de enrutamiento interior por vector-distancia, **RIP**, este en **versión 2**, que se configurará en cada uno de los routers del entorno privado. Esto para determinar la ruta de los paquetes que se usaran en las pruebas. Dentro de esto, se redistribuirá una ruta estática por defecto hacia el server por el router "NAT" que actúa de borde entre ambos entornos.

#### B. Explicación de su configuración

Antes de empezar a la configuración de los routers, es importante entender que en el programa de **GNS3** previamente se debió instalar la imagen del router **c3725** proporcionado por el profesor y las plantillas de las máquinas virtuales (Windows 7, Linux Mint y Ubuntu Server) enlazadas que también previamente fueron instaladas en **Oracle VM Virtualbox**.

Las imágenes de los routers se agregan al programa GNS3 en la parte de editar, preferencias, IOS routers, nuevo y ahí se selecciona nueva imagen con la ruta del archivo .iso que se descargó. Sobre las máquinas virtuales, para enlazarlas con GNS3, es el mismo proceso, pero en la parte de preferencias se debe entrar a VirtualBox VMs. Cabe recalcar que estas máquinas virtuales previamente ya deben estar instaladas en el programa de Oracle para que sea posible enlazarlas como se ha mencionado.

Entrando a como fue la configuración de la topología, principalmente se decidió crear un .txt en el cual se escribiría la configuración de cada uno de los routers. Cada uno de estos routers cumplen ciertas funciones importantes para el buen funcionamiento de la topología.

Cabe aclarar que la primera dirección de cada una de las redes estará asignada a la puerta de enlace.

#### Configuración Router Interno:

Este está configurado con **DHCP** para una asignación automática de direcciones IP para los hosts, excluyendo la primera dirección IP utilizable porque esta será utilizada para la puerta de enlace. También se le aplicara el protocolo de enrutamiento por vector distancia, **RIP en versión 2**, teniendo en cuenta las dos redes que tiene conectadas directamente. Los siguientes, fueron los pasos para su configuración.

1. Se configuran las interfaces del router asignándole direcciones IP y sus mascarar respectivas

```

R1#ena
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#hostname RouterInterno
RouterInterno(config)#interface fastethernet 0/1
RouterInterno(config-if)#ip address 192.168.25.1 255.255.255.0
RouterInterno(config-if)#no shut
RouterInterno(config-if)#exit
RouterInterno(config)#interface fastethernet 0/0
RouterInterno(config-if)#ip address 192.168.26.1 255.255.255.252
RouterInterno(config-if)#no shut
RouterInterno(config-if)#exit

```

Figura 16. Interfaces Router Interno. Fuente: Elaboración propia.

2. Se configura el router RIP

```

RouterInterno(config)#router rip
RouterInterno(config-router)#ver 2
RouterInterno(config-router)#no auto
RouterInterno(config-router)#network 192.168.25.0
RouterInterno(config-router)#network 192.168.26.0
RouterInterno(config-router)#exit

```

Figura 17. RIP router interno. Fuente: Elaboración propia.

3. Se configura el DHCP

```

R2#ena
R2#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#hostname NAT
NAT(config)#ip dhcp excluded-address 192.168.27.1 192.168.27.5
NAT(config)#ip dhcp pool red3
NAT(dhcp-config)#network 192.168.27.0 255.255.255.240
NAT(dhcp-config)#default-router 192.168.27.1
NAT(dhcp-config)#dns-server 200.20.2.2
NAT(dhcp-config)#domain-name puj.edu.co
NAT(dhcp-config)#exit

```

Figura 18. DHCP router interno. Fuente: Elaboración propia.

### Configuración Router NAT:

Este también está configurado con **DHCP** para la asignación automática de direcciones IP, se le aplicara el mismo protocolo de enrutamiento por vector distancia, **RIPv2**, sin embargo, este al estar en el borde de una red tipo **STUB**, tiene una configuración distinta. Además, se le debe agregar la parte **NAT (PAT)** porque este es el router designado para traducir las direcciones privadas a públicas.

- [1] Se configuran las interfaces del router asignándole direcciones IP y sus mascarar respectivas

```

NAT(config)#interface fastethernet 0/1
NAT(config-if)#ip address 192.168.27.1 255.255.255.240
NAT(config-if)#ip nat inside
NAT(config-if)#exit
NAT(config)#interface fastethernet 0/0
NAT(config-if)#ip address 192.168.26.2 255.255.255.252
NAT(config-if)#ip nat inside
NAT(config-if)#no shut
NAT(config-if)#exit
NAT(config)#interface fastethernet 1/0
NAT(config-if)#ip address 200.20.2.1 255.255.255.252
NAT(config-if)#ip nat outside
NAT(config-if)#no shut
NAT(config-if)#exit

```

Figura 19. Interfaces Router NAT. Fuente: Elaboración propia.

Dentro de cada una de las interfaces se debe escribir: **nat inside** o **nat outside**, esto porque este router es quien es el encargado de tener el protocolo **NAT** para cambiar direcciones IP privadas a direcciones **IP** públicas.

- [2] Se configura el router RIP

```

NAT(config)#router rip
NAT(config-router)#ver 2
NAT(config-router)#no auto
NAT(config-router)#network 192.168.25.0
NAT(config-router)#network 192.168.26.0
NAT(config-router)#network 192.168.27.0
NAT(config-router)#default-information originate
NAT(config-router)#exit

```

Figura 20. RIP Router NAT. Fuente: Elaboración propia.

En este se pone la línea de comando **default-information originate** porque este router es quien redistribuye una ruta estática por defecto hacia el servidor.

- [3] Se configura el DHCP

```

R2#ena
R2#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#hostname NAT
NAT(config)#ip dhcp excluded-address 192.168.27.1 192.168.27.5
NAT(config)#ip dhcp pool red3
NAT(dhcp-config)#network 192.168.27.0 255.255.255.240
NAT(dhcp-config)#default-router 192.168.27.1
NAT(dhcp-config)#dns-server 200.20.2.2
NAT(dhcp-config)#domain-name puj.edu.co
NAT(dhcp-config)#exit

```

Figura 21. DHCP Router NAT. Fuente: Elaboración propia.

- [4] Se configura el NAT PAT

```

NAT(config)#access-list 1 permit 192.168.24.0 0.0.3.255
NAT(config)#access-list 1 deny any
NAT(config)#ip nat inside source list 1 interface fastethernet 0/1 overload
NAT(config)#do wrf
Building configuration...
[OK]
NAT(config)#

```

Figura 22. NAT Router NAT. Fuente: Elaboración propia.

Cabe recalcar que dentro de cada una de las interfaces se puso **ip nat inside** o **ip nat outside** esto mostrando cuales son las redes internas privadas y cuáles la red externa pública.

### A. Pruebas y Resultados

Se corrió el comando para ver la dirección IP asignada a cada una de las máquinas virtuales para comprobar que el servicio **DHCP** para asignarlas de manera dinámica está funcionando correctamente.

### Windows:

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Nicolas>ipconfig /all

Windows IP Configuration

Host Name . . . . . : Nicolas-PC
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : puj.edu.co

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . : puj.edu.co
Description . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Physical Address. . . . . : 00-00-27-4A-24-5B
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::ddia:f8a:f41e:61c9%11(Preferred)
IPv4 Address. . . . . : 192.168.25.11(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : Monday, October 10, 2022 6:57:12 AM
Lease Expires . . . . . : Tuesday, October 11, 2022 6:57:12 AM
Default Gateway . . . . . : 192.168.25.1
DHCP Server . . . . . : 192.168.25.1
DHCPv6 Iaid . . . . . : 235405351
DHCPv6 Client DUID. . . . . : 00-01-00-01-2A-CA-8A-1B-08-00-27-4A-24-5B

DNS Servers . . . . . : 200.20.2.2
NetBIOS over Tcpip. . . . . : Enabled

Tunnel adapter isatap.puj.edu.co:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . : puj.edu.co
Description . . . . . : Microsoft ISATAP Adapter
Physical Address. . . . . : 00-00-00-00-00-00-E0
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . : Yes

C:\Users\Nicolas>

```

Figura 23. Comando **ipconfig /all** para comprobar dirección IP Windows 7. Fuente: Elaboración propia.

Como se puede ver, el **DHCP** esta activado y la dirección IPv4 es la **192.168.25.11**, esto quiere decir que está bien porque en la configuración excluimos las primeras 10 direcciones IP para que no fueran asignables.

### Linux Mint:

```

nicolas@nicolas-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

nicolas@nicolas-VirtualBox:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:1a:14:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.27.6/24 brd 192.168.27.15 scope global dynamic noprefixroute enp0s3
        valid lft 85896sec preferred_lft 85896sec
    inet6 fe80::952b:e82a:3ae8:fa79/64 scope link noprefixroute
        valid lft forever preferred_lft forever
nicolas@nicolas-VirtualBox:~$

```

Figura 24. Comando **ip a** para comprobar dirección IP Linux Mint. Fuente: Elaboración propia.

Como se puede ver, la dirección asignada por **DHCP** es la **192.168.27.6**, ergo la configuración está bien porque en este caso se excluyeron las primeras 5 direcciones IP para que no fueran asignables.

### Ubuntu Server:

Para cambiar la dirección IP en el Ubuntu Server, se configuro el archivo **yaml** para la configuración de la **red (netplan)** para asignarle una dirección IP estática.

El siguiente es el comando a ejecutar:

```
nicolas@server:~$ sudo nano /etc/netplan/00-installer-config.yaml
```

Figura 25. Comando para editar archivo **yaml**. Fuente: Elaboración propia.

Tras haber abierto este archivo de configuración netplan, se decidió asignarle una dirección IP estática, ya que por defecto el Ubuntu Server la consigue por **DHCP**. Asimismo, se le agrego el **servidor con dirección 200.20.2.2** que, en este caso, es el mismo.

El archivo fue configurado de la siguiente manera:

```

GNU nano 6.2 /etc/netplan/00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernet:
    enp0s3:
      addresses: [200.20.2.2/30]
      gateway4: 200.20.2.1
      nameservers:
        addresses: [200.20.2.2]

```

Figura 26. Configuración archivo **yaml**. Fuente: Elaboración propia.

Como se puede ver, se agregó la dirección IP del servidor y la dirección del servidor **DNS** que en este caso es la misma, esto con su respectivo **Gateway**.

Para revisar si la configuración se guardó correctamente, se introduce el siguiente comando:

```
nicolas@server:~$ sudo netplan try
```

Figura 27. Comando para revisar cambios hechos. Fuente: Elaboración propia.

El comando, nos arrojó los siguientes resultados:

```

nicolas@server:~$ sudo netplan try
** (generate:939): WARNING **: 23:05:01.189: 'gateway4' has been deprecated, use default routes instead.
See the 'Default routes' section of the documentation for more details.
** (process:937): WARNING **: 23:05:01.923: 'gateway4' has been deprecated, use default routes instead.
See the 'Default routes' section of the documentation for more details.
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 70 seconds
Configuration accepted.

```

Figura 28. Aceptación configuración de red. Fuente: Elaboración propia.

Como se puede ver al final de la imagen, esta configuración fue aceptada por la máquina.

Para cambiar esta configuración de manera permanente, se ejecutó el siguiente comando: **Sudo netplan apply**. Tras esto, es necesario un reinicio de la máquina que se efectuó con el siguiente comando: **sudo reboot**.

Una vez la maquina se ha reiniciado, se utiliza el comando **IP a** para comprobar que la dirección **IP** fue asignada correctamente.



```

nicolas@server:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:a7:4c:3d brd ff:ff:ff:ff:ff:ff
    inet 200.20.2.2/30 brd 200.20.2.3 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7a:4c3d/64 scope link
        valid_lft forever preferred_lft forever

```

Figura 29. Dirección IP asignada. Fuente: Elaboración propia.

Se reviso que el Gateway se haya configurado correctamente con el comando: **ip route show**, la siguiente figura muestra los resultados.

```

nicolas@server:~$ ip route show
default via 200.20.2.1 dev enp0s3 proto static
200.20.2.0/30 dev enp0s3 proto kernel scope link src 200.20.2.2

```

Figura 30. Comando **ip route show**. Fuente: Elaboración propia.

Estas dos figuras anteriores nos comprueban que el archivo de configuración fue correctamente aplicado. Tras haber confirmado las direcciones IP de cada uno de los dispositivos en la red, se siguió con la comprobación del buen funcionamiento de la topología. Para esto, es necesario hacer ping entre máquinas y primero empezaremos haciendo ping entre ambas máquinas virtuales que están dentro del entorno privado, la maquina Windows 7 y Linux Mint.

El siguiente es el ping desde la maquina Windows 7 hacia la maquina Linux Mint:

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Nicolas>ping 192.168.27.6

Pinging 192.168.27.6 with 32 bytes of data:
Reply from 192.168.27.6: bytes=32 time=41ms TTL=62
Reply from 192.168.27.6: bytes=32 time=33ms TTL=62
Reply from 192.168.27.6: bytes=32 time=38ms TTL=62
Reply from 192.168.27.6: bytes=32 time=33ms TTL=62

Ping statistics for 192.168.27.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 33ms, Maximum = 41ms, Average = 36ms

C:\Users\Nicolas>_

```

Figura 31. Resultado comando **ping 192.168.27.6**. Fuente: Elaboración propia.

Como se puede ver en la figura, las estadísticas fueron que se enviaron un total de 4 paquetes, se recibieron 4 y se perdieron 0, por lo cual, hubo un 0% de perdida de paquetes, mostrándonos como existe conectividad entre las maquinas del entorno privado.

Ahora, es importante probar un ping hacia el Servidor Ubuntu en el entorno público con una dirección IP asignada de 200.20.2.2.

Ping desde la maquina Windows 7:

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Nicolas>ping 200.20.2.2

Pinging 200.20.2.2 with 32 bytes of data:
Reply from 200.20.2.2: bytes=32 time=39ms TTL=62
Reply from 200.20.2.2: bytes=32 time=34ms TTL=62
Reply from 200.20.2.2: bytes=32 time=38ms TTL=62
Reply from 200.20.2.2: bytes=32 time=38ms TTL=62

Ping statistics for 200.20.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 38ms, Maximum = 39ms, Average = 35ms

C:\Users\Nicolas>

```

Figura 32. Resultado ping hacia Server Linux desde Windows 7.

Fuente: Elaboración propia.

Ping desde la maquina Linux Mint:

```

nicolas@nicolas-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

nicolas@nicolas-VirtualBox:~$ ping 200.20.2.2
PING 200.20.2.2 (200.20.2.2) 56(84) bytes of data.
64 bytes from 200.20.2.2: icmp_seq=2 ttl=63 time=14.6 ms
64 bytes from 200.20.2.2: icmp_seq=3 ttl=63 time=15.1 ms
64 bytes from 200.20.2.2: icmp_seq=4 ttl=63 time=16.2 ms
64 bytes from 200.20.2.2: icmp_seq=5 ttl=63 time=19.3 ms
64 bytes from 200.20.2.2: icmp_seq=6 ttl=63 time=12.9 ms
64 bytes from 200.20.2.2: icmp_seq=7 ttl=63 time=14.3 ms
^C
--- 200.20.2.2 ping statistics ---
7 packets transmitted, 6 received, 14.2857% packet loss, time 6030ms
rtt min/avg/max/mdev = 12.853/15.395/19.343/2.030 ms
nicolas@nicolas-VirtualBox:~$

```

Figura 33. Resultado ping hacia Server Linux desde Linux Mint.

Fuente: Elaboración propia.

La figura 32 y 33 nos muestran claramente como existe conectividad entre el entorno privado y entorno público al hacer la prueba ping con resultado satisfactorio hacia el Ubuntu Server con dirección IP 200.20.2.2.

## V. Actualización de topología

Tras la primera entrega del proyecto, se decidió actualizar la topología y volverla a crear en una de las máquinas virtuales asignadas por el profesor, la máquina en la cual se instaló el GNS3 tiene la dirección IP de 10.43.101.41.

Al intentar volver a instalar **Virtualbox**, nos dimos cuenta de que la máquina virtual suministrada no tiene activada la configuración para hacer posible la virtualización, soltando así un error: **ERROR: HAXM acceleration support is not installed on this host**. Para poder arreglar este error, es necesario ingresar a la **BIOS** de la máquina para activar esta función, sin embargo, no fue posible porque es una maquina virtualizada.

Para solucionar este error, la máquina virtual de **Windows 7** y **Linux Mint** fueron cambiadas por **VPCs**.

Asimismo, se tomó la decisión de instalar una nueva máquina sin interfaz gráfica, solo tiene consola y se maneja por línea de

comandos, esta es un **Alpine Linux** y se instaló por medio del archivo **alpine.qcow2** en la parte de **GNS3** de preferencias, **QEMU** y **QEMU VMs**.

El inicio de sesión de esta máquina Alpine es por medio de usuario y contraseña. El usuario es **root** y la contraseña es **Javeriana**.

Por otro lado, un cambio importante es la implementación de otro router donde estaba la máquina virtual **Ubuntu Server**. Este será el **R3** (Router NAT) y ahora tendrá la configuración del NAT PAT, que tenía el Router 2.

Para finalizar, se colocó una nube NAT para la conexión a la otra máquina virtual que tendrá el código del Servidor DNS, esta está directamente conectada al **R3** y cabe mencionar que esta interfaz del router recibe la dirección IP por medio de DHCP. La dirección IP de la otra máquina virtual es la **10.43.101.27**.

#### A. Configuración Nueva topología

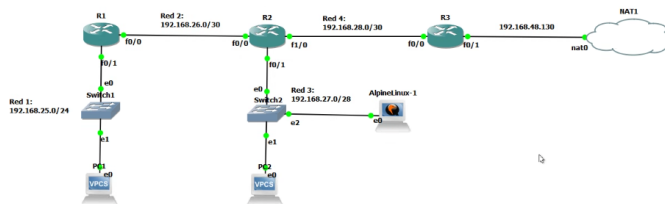


Figura 34. Nueva topología. Fuente: Elaboración propia.

Esta es la convención para las siguientes imágenes de configuración (Gran parte de las configuraciones siguen igual a la primera entrega)

**R1:** RouterInterno  
**R2:** RouterInterno2  
**R3:** Router NAT

Configuración interfaces del **R1** asignándole sus direcciones IP y máscaras respectivas.

```
RouterInterno(config)#interface fastethernet 0/1
RouterInterno(config-if)#ip address 192.168.25.1 255.255.255.0
RouterInterno(config-if)#no shut
RouterInterno(config-if)#exit
RouterInterno(config)#interface fastethernet 0/0
RouterInterno(config-if)#ip address 192.168.26.1 255.255.255.252
RouterInterno(config-if)#no shut
RouterInterno(config-if)#exit
```

Figura 35. Configuración RouterInterno. Fuente: Elaboración propia.

Configuración de router RIP del **R1**

```
RouterInterno(config)#router rip
RouterInterno(config-router)#ver 2
RouterInterno(config-router)#no auto
RouterInterno(config-router)#network 192.168.25.0
RouterInterno(config-router)#network 192.168.26.0
RouterInterno(config-router)#exit
RouterInterno(config)#do wri
Building configuration...
[OK]
```

Figura 36. RIP router interno. Fuente: Elaboración propia.

Configuración DHCP del **R1**

```
RouterInterno(config)#ip dhcp excluded-address 192.168.25.1 192.168.25.10
RouterInterno(config)#ip dhcp pool 20cinco
RouterInterno(dhcp-config)#network 192.168.25.0 255.255.255.0
RouterInterno(dhcp-config)#default-router 192.168.25.1
RouterInterno(dhcp-config)#exit
```

Figura 37. DHCP router interno. Fuente: Elaboración propia.

Configuración Interfaces del **R2** asignándole sus direcciones IP y máscaras respectivas

```
RouterInterno2#ena
RouterInterno2#conf t
Enter configuration commands, one per line. End with CNTL/Z.
RouterInterno2(config)#hostname RouterInterno2
RouterInterno2(config)#interface fastethernet 0/1
RouterInterno2(config-if)#ip address 192.168.27.1 255.255.255.240
RouterInterno2(config-if)#no shut
RouterInterno2(config-if)#exit
RouterInterno2(config)#interface fastethernet 0/0
RouterInterno2(config-if)#ip address 192.168.26.2 255.255.255.252
RouterInterno2(config-if)#no shut
RouterInterno2(config-if)#exit
RouterInterno2(config)#interface fastethernet 1/0
RouterInterno2(config-if)#ip address 192.168.28.1 255.255.255.252
RouterInterno2(config-if)#no shut
RouterInterno2(config-if)#exit
```

Figura 38. Configuración RouterInterno2. Fuente: Elaboración propia.

Configuración de router RIP del **R2**

```
RouterInterno2(config)#router rip
RouterInterno2(config-router)#ver 2
RouterInterno2(config-router)#no auto
RouterInterno2(config-router)#network 192.168.26.0
RouterInterno2(config-router)#network 192.168.27.0
RouterInterno2(config-router)#network 192.168.28.0
RouterInterno2(config-router)#exit
RouterInterno2(config)#do wri
Building configuration...
[OK]
```

Figura 39. RIP RouterInterno2. Fuente: Elaboración propia.

Configuración DHCP del **R2**

```
RouterInterno2(config)#ip dhcp excluded-address 192.168.27.1 192.168.27.5
RouterInterno2(config)#ip dhcp pool red3
RouterInterno2(dhcp-config)#network 192.168.27.0 255.255.255.240
RouterInterno2(dhcp-config)#default-router 192.168.27.1
RouterInterno2(dhcp-config)#exit
```

Figura 40. DHCP RouterInterno2. Fuente: Elaboración propia.

Configuración interfaces del **R3** asignándole sus direcciones IP y máscaras respectivas

```

RouterNAT#ena
RouterNAT#conf t
Enter configuration commands, one per line. End with CNTL/Z.
RouterNAT(config)#hostname RouterNAT
RouterNAT(config)#interface fastethernet 0/0
RouterNAT(config-if)#ip address 192.168.28.2 255.255.255.252
RouterNAT(config-if)#ip nat inside
RouterNAT(config-if)#no shut
RouterNAT(config-if)#exit
RouterNAT(config)#interface fastethernet 0/1
RouterNAT(config-if)#ip address dhcp
RouterNAT(config-if)#ip nat outside
RouterNAT(config-if)#no shut
RouterNAT(config-if)#exit

```

Figura 41. Configuración NAT. Fuente: Elaboración propia.  
Cabe mencionar que para este router NAT, la interfaz que da hacia la nube NAT, se le asignó dirección IP por medio de DHCP con el siguiente comando: **ip address dhcp**.

#### Configuración del router RIP del R3

```

RouterNAT(config)#router rip
RouterNAT(config-router)#ver 2
RouterNAT(config-router)#redistribute static
RouterNAT(config-router)#network 192.168.28.0
RouterNAT(config-router)#exit

```

Figura 42. RIP routerNAT. Fuente: Elaboración propia.

#### Configuración Access-list NAT del R3

```

RouterNAT(config)#access-list 1 permit 192.168.24.0 0.0.3.255
RouterNAT(config)#access-list 1 deny any
RouterNAT(config)#do source list 1 interface FastEthernet 0/1 overload
RouterNAT(config)#do wri

```

Figura 43. Access-list routerNAT. Fuente: Elaboración Propia.

Después de haber hecho la configuración de cada uno de los routers de la topología, se procedió a asignarle direcciones IP a cada uno de los hosts.

Primero, se le otorga una dirección IP a los VPCS con el comando: **ip dhcp** en la línea de comandos, si este da el comando DORA fue exitoso.

```

PC1> ip dhcp
DORA IP 192.168.25.11/24 GW 192.168.25.1
PC1> show ip all

```

NAME	IP/MASK	GATEWAY	MAC	DNS
PC1	192.168.25.11/24	192.168.25.1	00:50:79:66:68:00	

Figura 44. Asignación de IP al PC1. Fuente: Elaboración propia.

```

PC2> ip dhcp
DORA IP 192.168.27.6/28 GW 192.168.27.1
PC2> show ip all

```

NAME	IP/MASK	GATEWAY	MAC	DNS
PC2	192.168.27.6/28	192.168.27.1	00:50:79:66:68:02	

Figura 45. Asignación de IP al PC2. Fuente: Elaboración propia.

Revisar que el **Alpine** tenga dirección IP asignada, esto mediante el comando **ip a**.

```

alp:~# ip a
1: lo: <LOOPBACK,UP,LOWER UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 qdisc pfifo_fast
    link/ether 0c:fa:42:88:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.27.7/28 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::efa:42ff:fe88:0/64 scope link
        valid_lft forever preferred_lft forever

```

Figura 46. IP del Alpine. Fuente: Elaboración propia.

Como se puede ver, la dirección IP asignada para la máquina Alpine es la **192.168.27.7**, lo cual tiene sentido porque la anterior dirección IP utilizable fue asignada al PC2: **192.168.27.6**. Esta dirección IP fue asignada por medio de **DHCP** configurado en el R2.

#### Nodo NAT

En esta nueva tipología para garantizar NAT, este debe de conectarse a una red externa. GNS3 nos otorga el nodo NAT, el cual nos permite enviar paquetes fuera de la red hacia la otra máquina virtual que termina en **.27**. Esta funciona gracias al VMware para hacerse pasar por una tarjeta virtual de la máquina.

Cuando se quiera enviar un paquete hacia la red externa pasara por el **R3** y tomara dirección por el **puerto f0/1** hacia la nube NAT el cual hace un puente hacia la máquina virtual con dirección IP **10.43.101.27**.

Gracias a que el nodo NAT de GNS3 actúa como un nodo de DHCP no configurable se le asignara automáticamente una dirección IP del tipo 192.168.x.x.

El comando **show ip interface brief** nos mostrara la dirección IP asignada a esta interfaz.

```

RouterNAT#
RouterNAT#show ip interface brief

```

Interface	IP-Address	OK?	Method	Status	Protocol
FastEthernet0/0	192.168.28.2	YES	NVRAM	up	up
FastEthernet0/1	192.168.48.130	YES	DHCP	up	up
NVI0	192.168.28.2	YES	unset	up	up

Figura 47. Dirección IP asignada a F0/1. Fuente: Elaboración propia.

Como se puede ver, el **puerto f0/1 (FastEthernet0/1)** se le asignó la dirección **192.168.48.130**. Se procede a escribir esta dirección en la topología.

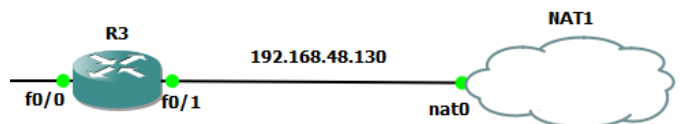


Figura 48. Dirección IP F0/1. Fuente: Elaboración propia.

#### B. Pruebas de la nueva tipología

Se realiza una prueba desde el **PC1** con dirección al **PC2**



```
PC1> ping 192.168.27.6
84 bytes from 192.168.27.6 icmp_seq=1 ttl=62 time=36.568 ms
84 bytes from 192.168.27.6 icmp_seq=2 ttl=62 time=22.843 ms
84 bytes from 192.168.27.6 icmp_seq=3 ttl=62 time=23.405 ms
84 bytes from 192.168.27.6 icmp_seq=4 ttl=62 time=38.605 ms
84 bytes from 192.168.27.6 icmp_seq=5 ttl=62 time=34.253 ms
```

Figura 49. Ping PC1-PC2: Elaboración propia.

Este envía 5 paquetes de 84 bytes con dirección a 192.168.27.6 confirmando la conectividad. Fuente:

Se realiza otra prueba desde el **PC1** con dirección al **Alpine**

```
PC1> ping 192.168.27.7
84 bytes from 192.168.27.7 icmp_seq=1 ttl=62 time=43.294 ms
84 bytes from 192.168.27.7 icmp_seq=2 ttl=62 time=48.136 ms
84 bytes from 192.168.27.7 icmp_seq=3 ttl=62 time=50.336 ms
84 bytes from 192.168.27.7 icmp_seq=4 ttl=62 time=52.115 ms
84 bytes from 192.168.27.7 icmp_seq=5 ttl=62 time=53.124 ms
```

Figura 50. Ping PC1-Alpine: Elaboración propia

Este volverá a enviar 5 paquetes de 84 bytes con dirección 192.168.27.7.

Se realiza una prueba inversa donde se manda desde el **Alpine** con dirección a **PC1**

```
alp:~# ping 192.168.25.11
PING 192.168.25.11 (192.168.25.11): 56 data bytes
64 bytes from 192.168.25.11: seq=0 ttl=62 time=3058.529 ms
64 bytes from 192.168.25.11: seq=1 ttl=62 time=2058.620 ms
64 bytes from 192.168.25.11: seq=2 ttl=62 time=1051.596 ms
64 bytes from 192.168.25.11: seq=3 ttl=62 time=63.971 ms
64 bytes from 192.168.25.11: seq=4 ttl=62 time=76.505 ms
64 bytes from 192.168.25.11: seq=5 ttl=62 time=55.932 ms
64 bytes from 192.168.25.11: seq=6 ttl=62 time=35.479 ms
64 bytes from 192.168.25.11: seq=7 ttl=62 time=57.851 ms
64 bytes from 192.168.25.11: seq=8 ttl=62 time=84.675 ms
64 bytes from 192.168.25.11: seq=9 ttl=62 time=40.343 ms
64 bytes from 192.168.25.11: seq=10 ttl=62 time=34.397 ms
^C
--- 192.168.25.11 ping statistics ---
11 packets transmitted, 11 packets received, 0% packet loss
round-trip min/avg/max = 34.397/601.627/3058.529 ms
```

Figura 51. Ping Alpine - PC1. Fuente: Elaboración Propia.

En esta prueba el PC Alpine envió alrededor de 11 paquetes de 64 bytes con dirección a 192.168.25.11 (para pararlo envió de paquetes se oprime “ctrl + c”) este obteniendo una transmisión del 100% y pérdida de 0% contando así con una conexión satisfactoria.

### C. Prueba de routers

Se realiza una prueba desde el **R1** con dirección a **R3**

```
RouterInterno#ping 192.168.28.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.28.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 100/116/140 ms
```

Figura 52. Ping R1-R3:Elaboración propia

Este enviara un total de 5 paquetes y se obtiene una respuesta del 100% 5/5 sin errores con dirección 192.168.28.2. Fuente: Elaboración propia.

### D. Prueba hacia la otra máquina virtual

Es importante revisar si existe conectividad entre ambas máquinas virtuales para proseguir con la elaboración del código en el lenguaje de programación **JAVA** dentro de la máquina virtual **10.43.101.27**. Es necesario hacer un ping desde la máquina que contiene la topología que es la que termina en **.41** hacia la que termina en **.27**. La siguiente figura es un ping desde la maquina **Alpine** hacia la **máquina virtual .27**.

```
alp:~# ping 10.43.101.27
PING 10.43.101.27 (10.43.101.27): 56 data bytes
64 bytes from 10.43.101.27: seq=1 ttl=126 time=83.692 ms
64 bytes from 10.43.101.27: seq=2 ttl=126 time=47.828 ms
64 bytes from 10.43.101.27: seq=3 ttl=126 time=44.017 ms
64 bytes from 10.43.101.27: seq=4 ttl=126 time=32.519 ms
64 bytes from 10.43.101.27: seq=5 ttl=126 time=34.591 ms
64 bytes from 10.43.101.27: seq=6 ttl=126 time=47.872 ms
^C
```

Figura 53. Ping desde Alpine a máquina virtual 10.43.101.27. Fuente: Elaboración propia.

Tras revisar que existe esta conectividad, se procede a configurar los servidores DNS en cada una de las maquinas dentro de la topología. Este se configurará con la dirección **10.43.101.27** porque allí es donde se encontrará el código **Java del servidor**.

Configuración DNS del PC1

```
PC1> ip dns 10.43.101.27

PC1> show ip
NAME       : PC1[1]
IP/MASK    : 192.168.25.11/24
GATEWAY    : 192.168.25.1
DNS        : 10.43.101.27
DHCP SERVER : 192.168.25.1
DHCP LEASE  : 86380, 86400/43200/75600
MAC        : 00:50:79:66:68:00
LPORT      : 10030
RHOST:PORT  : 127.0.0.1:10031
MTU        : 1500
```

Figura 54. Configuración DNS PC1. Fuente: Elaboración propia.

Configuración DNS del PC2

```
PC2> ip dns 10.43.101.27

PC2> show ip
NAME       : PC2[1]
IP/MASK    : 192.168.27.7/28
GATEWAY    : 192.168.27.1
DNS        : 10.43.101.27
DHCP SERVER : 192.168.27.1
DHCP LEASE  : 86392, 86400/43200/75600
MAC        : 00:50:79:66:68:01
LPORT      : 10032
RHOST:PORT  : 127.0.0.1:10033
MTU        : 1500
```

Figura 55. Configuración DNS PC2. Fuente: Elaboración propia.

### Configuración DNS del Alpine

Para este, es importante entender que la configuración es distinta, es necesario ejecutar la siguiente línea de comando “**vi /etc/resolv.conf**” en la cual es posible cambiar el **nameserver** a la dirección de la máquina que termina en **.27**.

La siguiente figura ilustra los cambios efectuados.

```
alp:~# cat /etc/resolv.conf
nameserver 10.43.101.27
```

Figura 56. Configuración DNS Alpine. Fuente: Elaboración propia.

## VI. Diseño del servidor DNS

Cabe recalcar que se utilizó el **IDE Netbeans** para la creación y compilación del código en Java, contando con una clase **main** denominada **ServidorDNS**, este se encuentra en la máquina **10.43.101.27**.

*En este programa solo se manejarán dominios QTYPE A y QCLASS IN.*

### A. Pasos lógicos del programa

El código principalmente se basa en:

1. Se crea un arreglo de bytes de tipo **byte** con longitud de **512**.
2. Se instancia un **socket** por medio de la librería **DatagramSocket** de Java. Este se crea en el **puerto 53** con la respectiva dirección IP de la máquina para recibir el paquete de petición DNS.
3. Se recibe el paquete de la otra máquina virtual por medio del **socket** previamente creado y se guarda en un paquete tipo **request** que se instancia por medio de la librería **DatagramPacket** de Java.
4. Se entra en un ciclo **while** el cual siempre estará activo mientras el programa este corriendo, esto quiere decir que estará en *escucha* de otras peticiones.
5. Dentro de este ciclo **while**, se encuentra el desencapsulamiento del paquete que se acaba de recibir. Al desencapsular se guarda especialmente el nombre de dominio que acaba de recibir.
6. Este nombre de dominio se busca en el archivo de texto **MasterFiles.txt** el cual alberga nombres de dominio con el siguiente formato: **www.NombreDeDominio.com IN A x.x.x.x**
7. Si no se encuentra el nombre de dominio dentro de los **MasterFiles**, se recurrirá a un **Foreign Resolver** el cual es el DNS de la Javeriana.
8. Esta búsqueda en el DNS de la Javeriana es posible gracias a la librería **java.net.InetAddress**.
9. Tras encontrar esta dirección IP en el servidor de la Javeriana, se procede a escribirlo en el **MasterFiles** con su respectivo nombre de dominio.

10. El código al ya tener el nombre de dominio y su respectiva dirección IP, procede a imprimirla en consola para demostrar su traducción.
11. Se vuelve a encapsular, pero ahora con la dirección IP encontrada para su envío devuelta al host y sus cambios pertinentes como lo pueden ser cambiar el **QR a 1** porque esta pasa a ser una respuesta.
12. Tras su encapsulamiento, se instancia un nuevo paquete con la librería **DatagramPacket** mencionada anteriormente y se envía mediante la función **socket.send(x)**.

### B. Pruebas de funcionamiento

Tras la explicación del código, se realiza un ping desde el Alpine hacia [www.javeriana.edu.co](http://www.javeriana.edu.co).

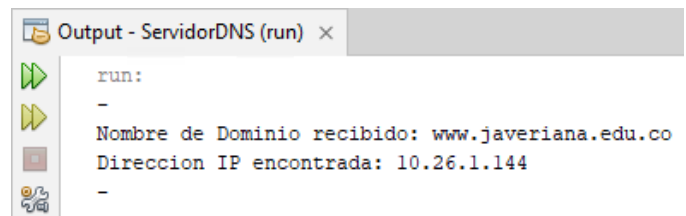
En pocas palabras, el **Alpine**, que se encuentra dentro de la topología **GNS3** en la máquina **10.43.101.41**, enviará un *request* hacia el **servidor DNS** que se alberga en la máquina **10.43.101.27** el cual tendrá el código **Java** corriendo, a través de la **nube NAT**. Este código resolverá el nombre de dominio y devolverá la dirección IP.

Resultado en el Alpine, en la máquina **10.43.101.41**

```
alp:~# ping www.javeriana.edu.co
PING www.javeriana.edu.co (10.26.1.144): 56 data bytes
64 bytes from 10.26.1.144: seq=1 ttl=126 time=69.553 ms
64 bytes from 10.26.1.144: seq=2 ttl=126 time=59.248 ms
64 bytes from 10.26.1.144: seq=3 ttl=126 time=47.348 ms
64 bytes from 10.26.1.144: seq=4 ttl=126 time=43.828 ms
64 bytes from 10.26.1.144: seq=5 ttl=126 time=50.557 ms
64 bytes from 10.26.1.144: seq=6 ttl=126 time=64.304 ms
^C
```

Figura 57. Resultado consola Alpine. Fuente: Elaboración propia.

Resultado en el IDE Netbeans, en la máquina **10.43.101.27**



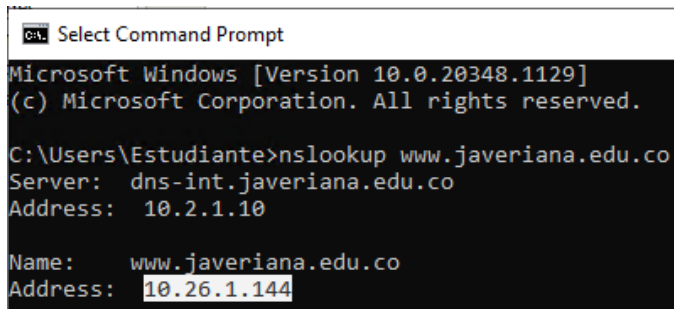
```
Output - ServidorDNS (run) x
run:
-
Nombre de Dominio recibido: www.javeriana.edu.co
Direccion IP encontrada: 10.26.1.144
-
```

Figura 58. Resultado consola Netbeans. Fuente: Elaboración propia.

Como se puede ver, el resultado es exitoso, se traduce el nombre de dominio a la dirección IP y se devuelve hacia el host quien pidió esta traducción.

Sin embargo, es necesario confirmar si la traducción es correcta, esto se puede hacer mediante el comando **nslookup** [www.javeriana.edu.co](http://www.javeriana.edu.co) desde el CMD (Command Prompt) de Windows y si devuelve la misma dirección IP, el código es correcto.





```

Select Command Prompt

Microsoft Windows [Version 10.0.20348.1129]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Estudiante>nslookup www.javeriana.edu.co
Server:  dns-int.javeriana.edu.co
Address:  10.2.1.10

Name:     www.javeriana.edu.co
Address:  10.26.1.144

```

Figura 59. Comprobacion resultados. Fuente: Elaboración propia.

En la figura anterior, se comprueba que la traducción del nombre de dominio a dirección IP es exitosa, se considera que el programa funciona correctamente.

## VII. CONCLUSIONES

A lo largo de este documento nos podemos dar cuenta de la complejidad de un servidor DNS pues estos son los encargados de una tarea esencial para el internet el cual es cambiar nombres de dominio a direcciones IP para tener conectividad. Gracias a esto, el usuario no necesita el uso de números para su búsqueda y así en milisegundos tener un grato viaje por el basto internet. Cabe recalcar que gracias a las clases de Comunicaciones y Redes podemos llevar a cabo este documento y su desarrollo de este para cumplir satisfactoriamente las entregas.

Al desarrollar un ejemplo con las herramientas dadas como lo fueron la aplicación GNS3, VirtualBox y una VPN por parte de la Universidad para esta implementación planteando así una estructura y componentes usados, a lo largo se explica el proceso de configuración de cada uno de estos, a su vez se demostró que se pudo hacer un ping entre el entorno privado y el entorno público, mostrando como quedaron bien configuradas estas máquinas y como la implantación de un servidor DNS codificado en el lenguaje Java como evolución de este proyecto para una funcionalidad completa del mismo.

## VIII. REFERENCIAS

- [1] CANDELAS-HERÍAS, Francisco A.; POMARES, Jorge. Práctica 3. Protocolos de transporte TCP y UDP. *Redes*, 2009.
- [2] DEFECTO DIGITAL, "DNS Explicado: Su historia, Que es? Y como funciona.", 5 de mar de 2018. [Video en línea]. Disponible en <https://www.youtube.com/watch?v=LqSqrtrW7w&t=14s>
- [3] IBM, «TCP/IP protocols - IBM,» IBM, 27 Septiembre 2022. [En línea]. Available: <https://www.ibm.com/docs/es/aix/7.3?topic=protocol-tcpip-protocols>. [Último acceso: 2 Octubre 2022].
- [4] AWS, «Que es DNS,» Amazon, 1 Octubre 2022. [En línea]. Available: <https://aws.amazon.com/es/route53/what-is-dns/>. [Último acceso: 5 octubre 2022].
- [5] MOCKAPETRIS, Paul V. Rfc1035: Domain names-implementation and specification. 1987..
- [6] I. Cloudflare, «¿Qué es una zona DNS?,» Cloudflare, 13 Junio 2022. [En línea]. Available: <https://www.cloudflare.com/es-es/learning/dns/glossary/dns-zone/>. [Último acceso: 1 octubre 2022].

- [7] SÁNCHEZ, Ernesto. *Un estudio comparativo en Extensiones de Seguridad para el Sistema de Nombres de Dominio (DNS)*. 2017. Tesis Doctoral. Universidad Nacional de La Plata.
- [8] REKHTER, Yakov, et al. *Address allocation for private internets*. 1996.
- [9] JasonGered, dknappettmsft, khdownie, cross-msft, iangpgh y jamesmci, «Dynamic Host Configuration Protocol (DHCP),» Microsoft, 29 julio 2021. [En línea]. Available: <https://learn.microsoft.com/en-us/windows-server/networking/technologies/dhcp/dhcp-top>. [Último acceso: 1 Octubre 2022].