Trabajo Final
Integración de Servicios 2024-2

Indicaciones generales

- La resolución es individual. Se pueden compartir configuraciones y lo que consideren necesario pero la interrogación será individual.
- Cada alumno deberá entregar las máquinas virtuales que considere necesarias para la resolución del problema.
- Las máquinas deben tener sistema operativo Debian (versión trabajada en clases).
- La implementación del Firewall se debe realizar usando IPTables (no se evaluarán otras versiones de Firewall).
- Los servidores deben tener interfaz de comando. ***NO*** deben tener interfaz gráfica
- Es responsabilidad de cada alumno el respaldo y cuidado de sus máquinas virtuales.

Fecha de entrega productos finales

- Miércoles 18 de diciembre de 2024 a las 17:10 horas.

Modalidad de evaluación

- Revisión archivos configuración
- Interrogación individual

Situación actual

Usted trabaja en una empresa en la que se ha detectado una vulnerabilidad en la seguridad de las conexiones remotas, lo que se evidencia en la venta de credenciales en la Deep web.

El departamento de TI (en el cual usted trabaja) ha tomado como primera medida de seguridad un reforzamiento de los procedimientos de conexión implementando una autenticación de dos pasos para todos los usuarios de la empresa.

La segunda medida a implementar consiste en la implementación de un sistema de monitoreo Web basado en NAGIOS que permita hacer una representación física de la red de datos.

Antecedentes

- La empresa sólo trabaja con conexiones a través de interfaz de comandos usando SSH. Eventualmente los usuarios utilizan la herramienta Putty para conexiones.
- Los usuarios poseen teléfonos inteligentes con sistema operativo Android (desde las versiones 10 en adelante).
- Los teléfonos inteligentes ***no*** se conectan a la red corporativa inalámbrica.
- Los usuarios tienen un alto nivel de alfabetización digital.
- La red de datos de la empresa tiene conexiones LAN y WLAN.

- La red WLAN está controlada por un router WLAN de propiedad de la empresa y entrega direcciones IP para la red 10.0.15.0 /24
- Un router que administra la red 192.168.23.0 /24
- El equipamiento computacional y de red está conformado por:
    - 4 servidores con sistema operativo Debian12 (1 de desarrollo con Apache con la IP 192.168.23.3 y MariaDB con la IP 192.168.23.4); 1 Web corporativo (con IP pública 200.27.0.23); 1 de producción con la IP 192.168.23.2 y 1 de hosting (con IP pública 200.27.0.24))
    - 15 pc con sistema operativo Windows (distintas versiones) según la siguiente distribución por áreas:
        - 2 (Administración) IP 192.168.23.15 - 16
        - 10 (Desarrollo) IP 192.168.23.20 - 30
        - 3 (Administración T.I.) IP 192.168.23.31 - 33
    - 3 impresoras LAN (1 Administración; 1 para Desarrollo y 1 para Administración T.I.) IP 192.168.23.40 - 42
    - 3 switch administrables IP 192.168.23.250 - 252
    - 1 AP WLAN IP 192.168.23.240
    - 1 Router empresarial (reemplaza al entregado por el ISP) con la IP 192.168.23.1
    - 1 servidor Debian 11 con 5 tarjetas de red para la implementación de un firewall por software con la IP 192.168.23.5

Requerimiento

La empresa le pide que implemente lo siguiente:

- Implementación de un sistema de autenticación en dos pasos para ***todos*** los servidores de la empresa.
- Implementación de un firewall que permita lo siguiente:
    - Sólo aceptar conexiones entrantes/salientes vía ssh desde las direcciones 192.168.23.0/24, 200.27.0.1/24, 146.83.1.0/24
    - Aceptar todas las conexiones web dirigidas al servidor de hosting y al servidor web corporativo provenientes del exterior/interior de la empresa
    - Aceptar las conexiones a los demás servidores sólo para la red interna de la empresa
    - Bloquear toda conexión no especificada acá
- Implementación de NAGIOS con el mapa de toda la red corporativa

Autenticación en dos pasos con Google Authenticator:

1.- Actualizar el sistema y preparar el entorno con los siguientes comandos:

apt update && apt upgrade -y

```
bgarri01 login: root
Password:
Linux bgarri01 5.10.0-32-amd64 #1 SMP Debian 5.10.223-1 (2024-08-10) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Nov 11 16:24:59 -03 2024 on tty1
root@bgarri01:~# apt update && apt upgrade -y
```

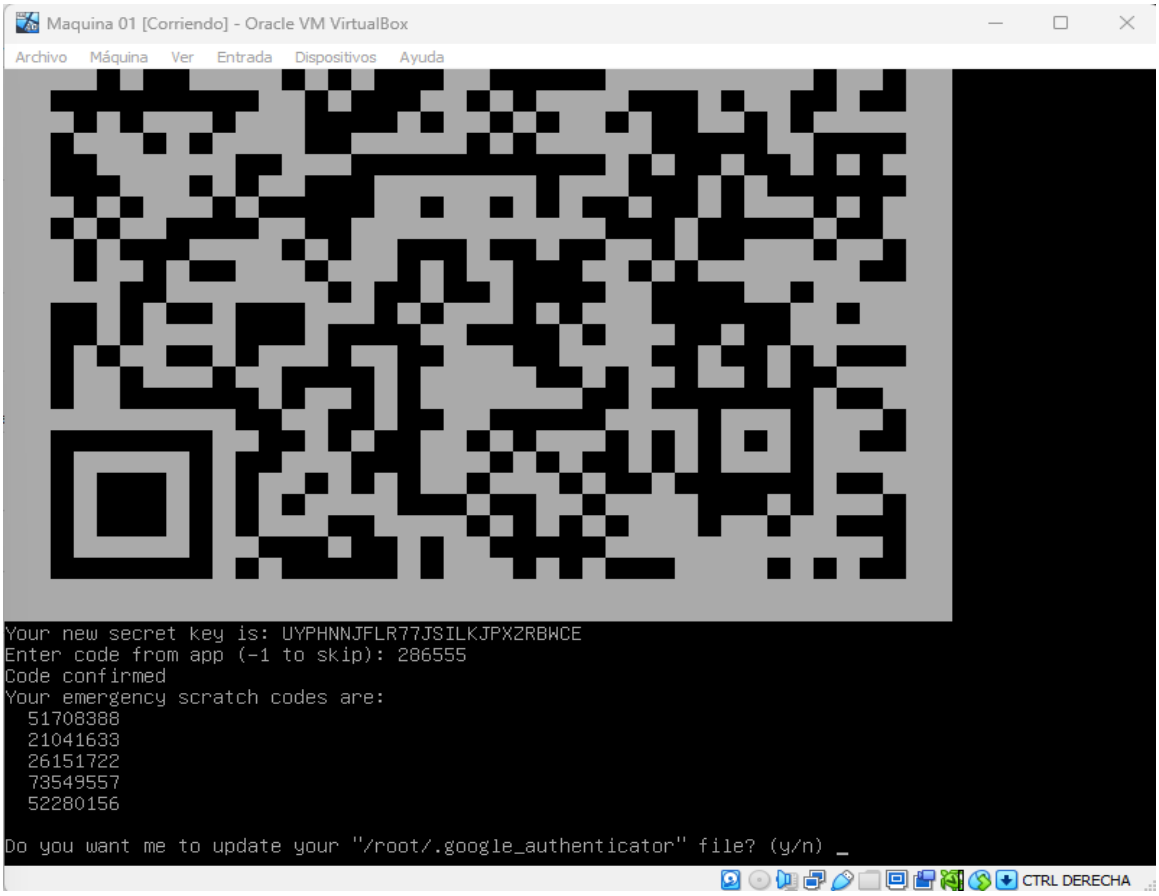
apt install libpam-google-authenticator -y

```
root@bgarri01:~# apt-get install libpam-google-authenticator -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-5.10.0-20-amd64
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  libqrencode4
The following NEW packages will be installed:
  libpam-google-authenticator libqrencode4
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 85.9 kB of archives.
After this operation, 229 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main amd64 libqrencode4 amd64 4.1.1-1 [40.4 kB]
Get:2 http://deb.debian.org/debian bullseye/main amd64 libpam-google-authenticator amd64 20191231-2
[45.5 kB]
Fetched 85.9 kB in 0s (844 kB/s)
Selecting previously unselected package libqrencode4:amd64.
(Reading database ... 40557 files and directories currently installed.)
Preparing to unpack .../libqrencode4_4.1.1-1_amd64.deb ...
Unpacking libqrencode4:amd64 (4.1.1-1) ...
Selecting previously unselected package libpam-google-authenticator.
Preparing to unpack .../libpam-google-authenticator_20191231-2_amd64.deb ...
Unpacking libpam-google-authenticator (20191231-2) ...
Setting up libqrencode4:amd64 (4.1.1-1) ...
Setting up libpam-google-authenticator (20191231-2) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for libc-bin (2.31-13+deb11u11) ...
root@bgarri01:~#
```

2.- Ejecutar la configuración del Google Authenticator:

- Ejecutar el siguiente comando:
    - **google-authenticator (se me olvidó sacar captura cuando ejecuté el comando)**
- Responder las siguientes preguntas conforme a las políticas de seguridad:
    - Generar claves únicas **(y/n)** respondemos: y
    - Evitar múltiples usos del mismo código **(y/n)** respondemos: y

- Habilitar desfase de tiempo de 4 minutos **(y/n)** respondemos: y
- Activar límite de intentos fallidos (rate-limiting) **(y/n)** respondemos: y




```
Do you want me to update your "/root/.google_authenticator" file? (y/n) y

Do you want to disallow multiple uses of the same authentication
token? This restricts you to one login about every 30s, but it increases
your chances to notice or even prevent man-in-the-middle attacks (y/n) y

By default, a new token is generated every 30 seconds by the mobile app.
In order to compensate for possible time-skew between the client and the server,
we allow an extra token before and after the current time. This allows for a
time skew of up to 30 seconds between authentication server and client. If you
experience problems with poor time synchronization, you can increase the window
from its default size of 3 permitted codes (one previous code, the current
code, the next code) to 17 permitted codes (the 8 previous codes, the current
code, and the 8 next codes). This will permit for a time skew of up to 4 minutes
between client and server.
Do you want to do so? (y/n) n

If the computer that you are logging into isn't hardened against brute-force
login attempts, you can enable rate-limiting for the authentication module.
By default, this limits attackers to no more than 3 login attempts every 30s.
Do you want to enable rate-limiting? (y/n) y
root@bgarri01:~# _
```

3.- Configuración del módulo PAM:

- Editar el archivo con **nano /etc/pam.d/sshd** y añadir:

auth required pam_google_authenticator.so

```
root@bgarri01:~# nano /etc/pam.d/sshd_
```

```
  GNU nano 5.4                                /etc/pam.d/sshd *
# PAM configuration for the Secure Shell service

# Standard Un*x authentication.
@include common-auth
auth required pam_google_authenticator.so_
# Disallow non-root logins when /etc/nologin exists.
account    required     pam_nologin.so

# Uncomment and edit /etc/security/access.conf if you need to set complex
# access limits that are hard to express in sshd_config.
# account  required     pam_access.so

# Standard Un*x authorization.
@include common-account

# SELinux needs to be the first session rule.  This ensures that any
# lingering context has been cleared.  Without this it is possible that a
# module could execute code in the wrong domain.
session [success=ok ignore=ignore module_unknown=ignore default=bad]        pam_selinux.so close

# Set the loginuid process attribute.
session    required     pam_loginuid.so

# Create a new session keyring.
session    optional     pam_keyinit.so force revoke

# Standard Un*x session setup and teardown.
@include common-session

# Print the message of the day upon successful login.
# This includes a dynamically generated part from /run/motd.dynamic
# and a static (admin-editable) part from /etc/motd.
session    optional     pam_motd.so  motd=/run/motd.dynamic

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line M-E Redo
```

4.- Habilitar Challenge-Response Authentication en SSH:

- Editar con **nano /etc/ssh/sshd_config** y configurar la siguiente línea:

ChallengeResponseAuthentication yes

```
root@bgarri01:~# nano /etc/ssh/sshd_config_
```

```
  GNU nano 5.4                          /etc/ssh/sshd_config *
# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication yes

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no

# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute   ^C Location  M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify   ^_ Go To Line M-E Redo
```

5.- Reiniciar el servicio SSH:

Ejecutar el comando: **systemctl restart ssh**

```
root@bgarri01:~# systemctl restart ssh_
```

6.- Prueba de Autenticación en 2 Pasos

- **Iniciar sesión SSH desde una segunda máquina:** El usuario ejecuta el siguiente comando en la terminal:

  ssh bgarri@192.168.23.165

- **Proceso de autenticación:**
  - Password: Se introduce la contraseña del usuario configurada previamente.
  - Verification code: El sistema solicita el código de verificación generado por la aplicación Google Authenticator en el dispositivo móvil del usuario.
  - La combinación de la contraseña y el código de verificación implementa el doble factor de autenticación (2FA).

Implementación de Nagios

1. Instalación de Dependencias

Primero, se deben instalar las dependencias necesarias para compilar e instalar Nagios:

apt update && apt install wget apache2 php libapache2-mod-php build-essential libgd-dev unzip -y

```
root@bgarri01:~# apt install wget apache2 php libapache2-mod-php build-essential libgd-dev unzip
```

2. Descarga e Instalación de Nagios Core

- Descargar Nagios Core:

  wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.14.tar.gz

```
root@bgarri01:/# wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.14.tar.gz
--2024-12-05 16:24:32--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.14.tar.g
z
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11341108 (11M) [application/x-gzip]
Saving to: ‘nagios-4.4.14.tar.gz’

nagios-4.4.14.tar.gz     100%[===================================>]  10.82M  1.54MB/s    in 6.6s

2024-12-05 16:24:40 (1.63 MB/s) - ‘nagios-4.4.14.tar.gz’ saved [11341108/11341108]

root@bgarri01:/# _
```

- Extraer el archivo descargado:
  tar -zxvf nagios-4.4.14.tar.gz

```
root@bgarri01:/# tar -zxvf nagios-4.4.14.tar.gz_
```

- Entramos al directorio con el comando:

  cd nagios-4.4.14 y luego ejecutar ./configure

```
root@bgarri01:/nagios-4.4.14# ./configure_
```

```
*** Configuration summary for nagios 4.4.14 2023-08-01 ***:

General Options:
-------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagios
            Event Broker:   yes
       Install ${prefix}:   /usr/local/nagios
   Install ${includedir}:   /usr/local/nagios/include/nagios
                Lock file:  /run/nagios.lock
  Check result directory:   /usr/local/nagios/var/spool/checkresults
          Init directory:   /lib/systemd/system
 Apache conf.d directory:   /etc/apache2/sites-available
             Mail program:  /bin/mail
                  Host OS:  linux-gnu
          IOBroker Method:  epoll

Web Interface Options:
------------------------
                HTML URL:  http://localhost/nagios/
                 CGI URL:  http://localhost/nagios/cgi-bin/
Traceroute (used by WAP):  /usr/sbin/traceroute


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.
```

```
root@bgarri01:/nagios-4.4.14# ./configure --with-httpd-conf=/etc/apache2/sites-enable_
```

- Configurar, compilar e instalar Nagios:

  Ejecutamos **make all**

```
root@bgarri01:/nagios-4.4.14# make all_
```

```
 make install-exfoliation
    - This installs the Exfoliation theme for the Nagios
      web interface

 make install-classicui
    - This installs the classic theme for the Nagios
      web interface


*** Support Notes *******************************************

If you have questions about configuring or running Nagios,
please make sure that you:

    - Look at the sample config files
    - Read the documentation on the Nagios Library at:
          https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

    - What version of Nagios you are using
    - What version of the plugins you are using
    - Relevant snippets from your config files
    - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

      https://support.nagios.com

****************************************************************

Enjoy.
```

Ejecutamos **make install**

```
root@bgarri01:/nagios-4.4.14# make install_
```

```
/usr/bin/install -c -m 664 -o nagios -g nagios bootstrap-3.3.7/css/bootstrap-theme.min.css /usr/loca
l/nagios/share/bootstrap-3.3.7/css
/usr/bin/install -c -m 664 -o nagios -g nagios d3/d3.min.js /usr/local/nagios/share/d3
/usr/bin/install -c -m 664 -o nagios -g nagios spin/spin.min.js /usr/local/nagios/share/spin
make[1]: Leaving directory '/nagios-4.4.14/html'
make install-exfoliation
make[1]: Entering directory '/nagios-4.4.14'

*** Exfoliation theme installed ***
NOTE: Use 'make install-classicui' to revert to classic Nagios theme

make[1]: Leaving directory '/nagios-4.4.14'
make install-basic
make[1]: Entering directory '/nagios-4.4.14'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/libexec
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/archives
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/spool/checkresults
chmod g+s /usr/local/nagios/var/spool/checkresults

*** Main program, CGIs and HTML files installed ***

You can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

  make install-init
     - This installs the init script in /lib/systemd/system

  make install-commandmode
     - This installs and configures permissions on the
       directory for holding the external command file

  make install-config
     - This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/nagios-4.4.14'
root@bgarri01:/nagios-4.4.14#
```

Ejecutamos **make install-init**: Instala el script de inicio.

```
root@bgarri01:/nagios-4.4.14# make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.servic
e
root@bgarri01:/nagios-4.4.14# make install-commandmode
```

Ejecutamos **make install-commandmode**: Configura permisos para comandos externos.

```
root@bgarri01:/nagios-4.4.14# make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***
```

Ejecutamos **make install-config:** Instala ejemplos de configuración.

```
root@bgarri01:/nagios-4.4.14# make install-config
```

```
*** External command directory configured ***

root@bgarri01:/nagios-4.4.14# make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nag
ios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg /usr/local/nagios/etc/cgi.cf
g
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/r
esource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/l
ocal/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/lo
cal/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/lo
cal/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr
/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/l
ocal/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/loc
al/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/loc
al/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switch.cfg /usr/loca
l/nagios/etc/objects/switch.cfg

*** Config files installed ***

Remember, these are *SAMPLE* config files.  You'll need to read
the documentation for more information on how to actually define
services, hosts, etc. to fit your particular needs.
```

Ejecutamos **make install-webconf:** Configura Apache para Nagios.

(Acá se me olvidó tomar captura para demostrar este comando)

```
root@bgarri01:/nagios-4.4.14# make install-webconf
```

3. Configuración del Acceso Web a Nagios

- Crear un usuario para acceder a la interfaz web de Nagios:

  htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

```
root@bgarri01:/# htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
root@bgarri01:/#
```

  - Introduce una contraseña segura para el usuario **nagiosadmin**.
    - CONTRASEÑA SEGURA: **N@10$2O24**

- **Habilitar módulos necesarios en Apache:**
    - Ejecutamos**:**
        - a2enmod rewrite
        - a2enmod cgi
        - systemctl restart apache2

```
root@bgarri01:/# a2enmod rewrite
Module rewrite already enabled
root@bgarri01:/# a2enmod cgi
Module cgi already enabled
root@bgarri01:/# systemctl restart apache2
root@bgarri01:/#
```

4. Instalación de Plugins de Nagios

- Descargar Nagios Plugins:

  wget https://nagios-plugins.org/download/nagios-plugins-2.4.3.tar.gz

```
root@bgarri01:/# wget https://nagios-plugins.org/download/nagios-plugins-2.4.3.tar.gz
--2024-12-05 17:00:32--  https://nagios-plugins.org/download/nagios-plugins-2.4.3.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2748045 (2.6M) [application/x-gzip]
Saving to: ‘nagios-plugins-2.4.3.tar.gz’

nagios-plugins-2.4.3.tar 100%[=====================================>]   2.62M  1.27MB/s    in 2.1s

2024-12-05 17:00:35 (1.27 MB/s) - ‘nagios-plugins-2.4.3.tar.gz’ saved [2748045/2748045]
```

- Extraer el archivo y compilar los plugins:
  tar -zxvf nagios-plugins-2.4.3.tar.gz
  cd nagios-plugins-2.4.3

```
root@bgarri01:/# tar -xzf nagios-plugins-2.4.3.tar.gz
root@bgarri01:/# cd nagios-plugins-2.4.3/
root@bgarri01:/nagios-plugins-2.4.3# _
```

./configure

```
root@bgarri01:/nagios-plugins-2.4.3# ./configure
```

```
              --with-ping-command: /usr/bin/ping -n -U -w %d -c %d %s
                     --with-ipv6: yes
                    --with-mysql: no
                  --with-openssl: yes
                   --with-gnutls: no
              --enable-extra-opts: yes
                     --with-perl: /usr/bin/perl
            --enable-perl-modules: no
                    --with-cgiurl: /nagios/cgi-bin
              --with-trusted-path: /usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
                  --enable-libtap: no
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating gl/Makefile
config.status: creating nagios-plugins.spec
config.status: creating tools/build_perl_modules
config.status: creating Makefile
config.status: creating tap/Makefile
config.status: creating lib/Makefile
config.status: creating plugins/Makefile
config.status: creating lib/tests/Makefile
config.status: creating plugins-root/Makefile
config.status: creating plugins-scripts/Makefile
config.status: creating plugins-scripts/utils.pm
config.status: creating plugins-scripts/utils.sh
config.status: creating perlmods/Makefile
config.status: creating test.pl
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
root@bgarri01:/nagios-plugins-2.4.3# make_
```

make install

```
root@bgarri01:/nagios-plugins-2.4.3# make install
```

```
/plugins/utils.o  -L. ../lib/libnagiosplug.a ../gl/libgnu.a -lnsl -lresolv -lssl -lcrypto -lpthread
-ldl
make[2]: Entering directory '/nagios-plugins-2.4.3/plugins-root'
 /usr/bin/install -c  check_dhcp /usr/local/nagios/libexec/check_dhcp
 chown root /usr/local/nagios/libexec/check_dhcp
 chmod ug=rx,u+s /usr/local/nagios/libexec/check_dhcp
 /usr/bin/install -c  check_icmp /usr/local/nagios/libexec/check_icmp
 chown root /usr/local/nagios/libexec/check_icmp
 chmod ug=rx,u+s /usr/local/nagios/libexec/check_icmp
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/nagios-plugins-2.4.3/plugins-root'
make[1]: Leaving directory '/nagios-plugins-2.4.3/plugins-root'
Making install in po
make[1]: Entering directory '/nagios-plugins-2.4.3/po'
/usr/bin/mkdir -p /usr/local/nagios/share
installing fr.gmo as /usr/local/nagios/share/locale/fr/LC_MESSAGES/nagios-plugins.mo
installing de.gmo as /usr/local/nagios/share/locale/de/LC_MESSAGES/nagios-plugins.mo
if test "nagios-plugins" = "gettext-tools"; then \
  /usr/bin/mkdir -p /usr/local/nagios/share/gettext/po; \
  for file in Makefile.in.in remove-potcdate.sin    Makevars.template; do \
    /usr/bin/install -c  -m 644 ./$file \
                    /usr/local/nagios/share/gettext/po/$file; \
  done; \
  for file in Makevars; do \
    rm -f /usr/local/nagios/share/gettext/po/$file; \
  done; \
else \
  : ; \
fi
make[1]: Leaving directory '/nagios-plugins-2.4.3/po'
make[1]: Entering directory '/nagios-plugins-2.4.3'
make[2]: Entering directory '/nagios-plugins-2.4.3'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/nagios-plugins-2.4.3'
make[1]: Leaving directory '/nagios-plugins-2.4.3'
root@bgarri01:/nagios-plugins-2.4.3#
```

5. Configuración de Nagios

Editar el archivo principal de configuración de Nagios:

```
root@bgarri01:/# nano /usr/local/nagios/etc/nagios.cfg
```

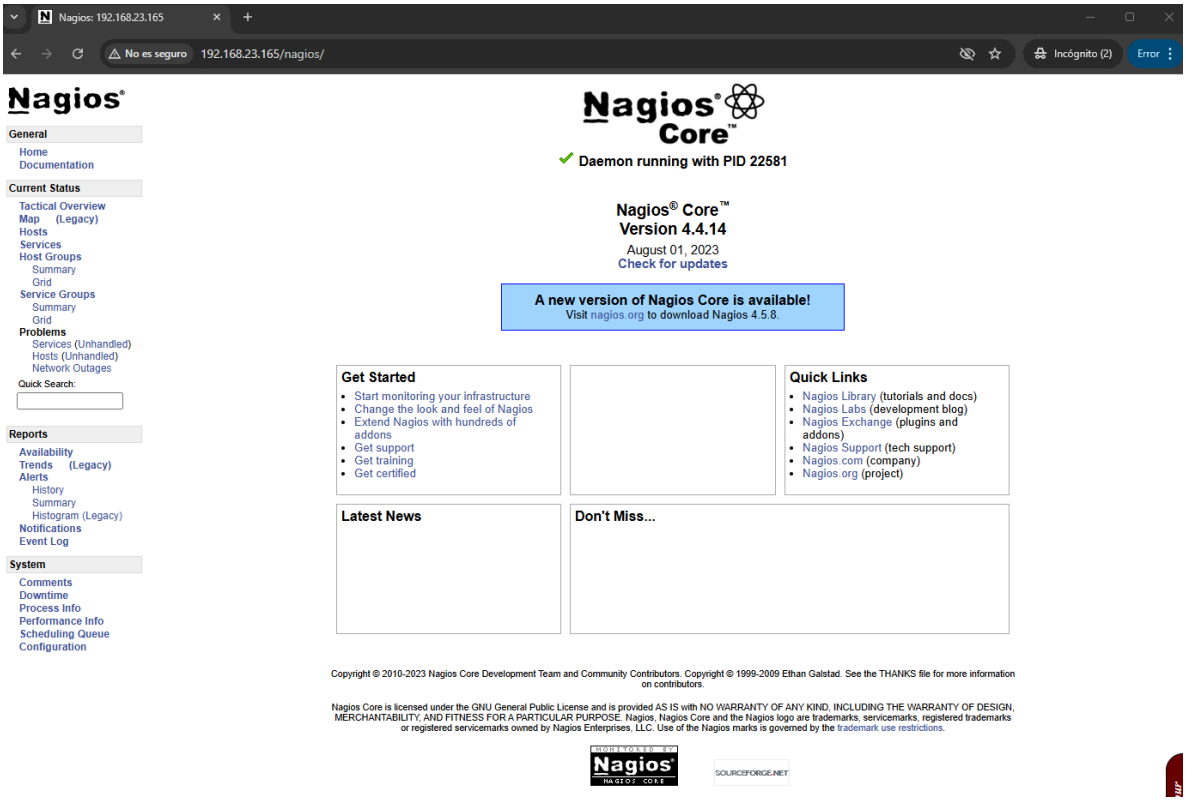
Asegurarse que la línea cfg_file=/usr/local/nagios/etc/objects/localhost.cfg no esté comentada

```
  GNU nano 5.4                          /usr/local/nagios/etc/nagios.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg
```

Guardar los cambios y reiniciar nagios

```
root@bgarri01:/# systemctl restart nagios
Warning: The unit file, source configuration file or drop-ins of nagios.service changed on disk. Run
 'systemctl daemon-reload' to reload units.
root@bgarri01:/# systemctl daemon-reload
root@bgarri01:/# systemctl restart nagios
```

Luego abrimos nagios en el navegador en este caso con mi IP **192.168.23.165/nagios/**



Definir dispositivos en Nagios:

Primeramente se accede a **usr/local/nagios/etc/objects/localhost.cfg**

```
root@bgarri01:~# nano /usr/local/nagios/etc/objects/localhost.cfg _
```

Dentro de este archivo se define cada dispositivo especificado, para realizar una prueba, dentro de esto configuramos directamente los parents para que quede de la mejor manera además de agregar las imágenes correspondientes a cada equipo.

Para asignar imágenes a los dispositivos, esto se puede hacer mediante la configuración de los iconos y las imágenes que se muestran en el mapa de red.

Estas imagenes se encuentran ya predefinidas para representar los dispositivos, se encuentran en la carpeta:

```
root@bgarri01:~# ls /usr/local/nagios/share/images/logos/_
```

```
aix.gif              freebsd40.gif        mandrake.jpg     router40.jpg    switch.gd2
aix.jpg              freebsd40.jpg        mandrake.png     router40.png    switch.gif
aix.png              freebsd40.png        monitor.png      router.gd2      thin-client.gd2
amiga.gd2            globe.png            nagios.gd2       router.gif      thin-client.gif
amiga.gif            graph.gif            nagios.gif       san.gd2         turbolinux.gd2
amiga.jpg            hp-printer40.gd2     nagiosvrml.png   san.gif         turbolinux.gif
amiga.png            hp-printer40.gif     netbsd.gif       satellite.png   turbolinux.jpg
apple.gd2            hp-printer40.jpg     netbsd.jpg       server.png      turbolinux.png
apple.gif            hp-printer40.png     netbsd.png       signal.png      ultrapenguin.gd2
apple.jpg            hpux.gd2             next.gd2         slackware.gd2   ultrapenguin.gif
apple.png            hpux.gif             next.gif         slackware.gif   ultrapenguin.jpg
beos.gd2             hpux.jpg             next.jpg         slackware.jpg   ultrapenguin.png
beos.gif             hpux.png             next.png         slackware.png   unicos.gd2
beos.jpg             hub.gd2              ng-switch40.gd2  stampede.gd2    unicos.gif
beos.png             hub.gif              ng-switch40.gif  stampede.gif    unicos.jpg
bluetooth.png        internet_device.png  ng-switch40.jpg  stampede.jpg    unicos.png
caldera.gd2          internet.gd2         ng-switch40.png  stampede.png    unknown.gd2
caldera.gif          internet.gif         notebook.gd2     station.gd2     unknown.gif
caldera.jpg          ip-pbx.gd2           notebook.gif     storm.gd2       webcamera.png
caldera.png          ip-pbx.gif           novell40.gd2     storm.gif       wifi.gd2
cat1900.gd2          irix.gd2             novell40.gif     storm.jpg       wifi.gif
cat2900.gd2          irix.gif             novell40.jpg     storm.png       wifi_modem.png
cat5000.gd2          irix.jpg             novell40.png     sun40.gd2       win40.gd2
database.gd2         irix.png             openbsd.gd2      sun40.gif       win40.gif
database.gif         linux40.gd2          openbsd.gif      sun40.jpg       win40.jpg
debian.gd2           linux40.gif          openbsd.jpg      sun40.png       win40.png
debian.gif           linux40.jpg          openbsd.png      sunlogo.gd2     workstation.gd2
debian.jpg           linux40.png          printer.gd2      sunlogo.gif     workstation.gif
debian.png           logo.gd2             printer.gif      sunlogo.jpg     workstation_locked.png
desktop-server.gd2   mac40.gd2           rack-server.gd2  sunlogo.png     workstation.png
desktop-server.gif   mac40.gif           rack-server.gif  suse.gif        yellowdog.gd2
ethernet_card.png    mac40.jpg            redhat.gd2       suse.jpg        yellowdog.gif
fax.gd2              mac40.png            redhat.gif       suse.png        yellowdog.jpg
fax.gif              mainframe.gd2        redhat.jpg       switch40.gd2    yellowdog.png
firewall.gd2         mainframe.gif        redhat.png       switch40.gif
firewall.gif         mandrake.gd2         router40.gd2     switch40.jpg
root@bgarri01:~# ~~~~~~~~
```

Cada imagen para cada dispositivo se utilizaron de la siguiente manera:

DISPOSITIVO	IMAGEN
Router Empresarial	router40.png
Servidores	server.png
Punto de acceso WLAN	wifi.gif
Switches	switch.gif
Impresoras	printer.gif
PC’s	win40.png

- 1 Router empresarial (reemplaza al entregado por el ISP) con la IP 192.168.23.1

```
  GNU nano 5.4                  /usr/local/nagios/etc/objects/localhost.cfg
############R ROUTER EMPRESARIAL ################################
define host{
        use             linux-server
        host_name       router-principal
        alias           Router Empresarial
        address         192.168.23.1
        icon_image      router40.png
}
```

- 4 servidores con sistema operativo Debian12 (1 de desarrollo con Apache con la IP 192.168.23.3 y MariaDB con la IP 192.168.23.4); 1 Web corporativo (con IP pública 200.27.0.23); 1 de producción con la IP 192.168.23.2 y 1 de hosting (con IP pública 200.27.0.24))

```
  GNU nano 5.4                 /usr/local/nagios/etc/objects/localhost.cfg
############## SERVIDORES ##############################
define host{
        use             linux-server
        host_name       produccion
        alias           Servidor de produccion
        address         192.168.23.2
        parents         switch-3
        icon_image      server.png
}

define host{
        use             linux-server
        host_name       desarrollo-apache
        alias           Servidor de Desarrollo (Apache)
        address         192.168.23.3
        parents         switch-2
        icon_image      server.png
}

define host{
        use             linux-server
        host_name       desarrollo-mariadb
        alias           Servidor de Desarrollo (MariaDB)
        address         192.168.23.4
        parents         switch-2
        icon_image      server.png
}
```

```
  GNU nano 5.4                 /usr/local/nagios/etc/objects/localhost.cfg

define host{
        use             linux-server
        host_name       web-corporativo
        alias           Servidor Web Corporativo
        address         200.27.0.23
        parents         switch-3
        icon_image      server.png
}

define host{
        use             linux-server
        host_name       hosting
        alias           Servidor de Hosting
        address         200.27.0.24
        parents         switch-1
        icon_image      server.png
}
```

- 1 AP WLAN IP 192.168.23.240

```
  GNU nano 5.4                 /usr/local/nagios/etc/objects/localhost.cfg
################# AP WLAN ########################
define host{
        use             linux-server
        host_name       ap-wlan
        alias           Punto de Acceso WLAN
        address         192.168.23.240
        parents         firewall-server
        icon_image      wifi.gif
}
define host{
        use             linux-server
        host_name       dispositivo-wlan
        alias           Dispositivo conectado al WLAN
        address         10.0.15.10
        parents         ap-wlan
}
```

- 3 switch administrables IP 192.168.23.250 - 252

```
  GNU nano 5.4                 /usr/local/nagios/etc/objects/localhost.cfg
################## SWITCH ########################

define host{
        use             generic-switch
        host_name       switch-1
        alias           Switch Administración
        address         192.168.23.250
        parents         firewall-server
        icon_image      switch.gif
}


define host{
        use             generic-switch
        host_name       switch-2
        alias           Switch Desarrollo
        address         192.168.23.251
        parents         firewall-server
        icon_image      switch.gif
}


define host{
        use             generic-switch
        host_name       switch-3
        alias           Switch Administración T.I
        address         192.168.23.252
        parents         firewall-server
        icon_image      switch.gif
}
```

- 3 impresoras LAN (1 Administración; 1 para Desarrollo y 1 para Administración T.I.) IP 192.168.23.40 - 42

```
  GNU nano 5.4                 /usr/local/nagios/etc/objects/localhost.cfg
################### IMPRESORAS ###########################
define host{
        use             generic-printer
        host_name       impresora-administracion
        alias           Impresora Administración
        address         192.168.23.40
        parents         switch-1
        icon_image      printer.gif
}

define host{
        use             generic-printer
        host_name       impresora-desarrollo
        alias           Impresora Desarrollo
        address         192.168.23.41
        parents         switch-2
        icon_image      printer.gif
}

define host{
        use             generic-printer
        host_name       impresora-ti
        alias           Impresora Administración T.I.
        address         192.168.23.42
        parents         switch-3
        icon_image      printer.gif
}
```

- 15 pc con sistema operativo Windows (distintas versiones) según la siguiente distribución por áreas:
    - 2 (Administración) IP 192.168.23.15 - 16

```
  GNU nano 5.4                    /usr/local/nagios/etc/objects/localhost.cfg
######################## PC_ADMINISTRACION ######################################
define host{
        use             windows-server
        host_name       pc-admin-1
        alias           PC Administracion 1
        address         192.168.23.15
        parents         switch-1
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-admin-2
        alias           PC Administracion 2
        address         192.168.23.16
        parents         switch-1
        icon_image      win40.png
}
```

    - 10 (Desarrollo) IP 192.168.23.20 - 29

```
  GNU nano 5.4                    /usr/local/nagios/etc/objects/localhost.cfg
######################## PC_DESARROLLO ##########################################
define host{
        use             windows-server
        host_name       pc-dev-1
        alias           PC Desarrollo 1
        address         192.168.23.20
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-2
        alias           PC Desarrollo 2
        address         192.168.23.21
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-3
        alias           PC Desarrollo 3
        address         192.168.23.22
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-4
        alias           PC Desarrollo 4
        address         192.168.23.23
        parents         switch-2
        icon_image      win40.png
}
```

```
  GNU nano 5.4                    /usr/local/nagios/etc/objects/localhost.cfg
define host{
        use             windows-server
        host_name       pc-dev-5
        alias           PC Desarrollo 5
        address         192.168.23.24
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-6
        alias           PC Desarrollo 6
        address         192.168.23.25
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-7
        alias           PC Desarrollo 7
        address         192.168.23.26
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-8
        alias           PC Desarrollo 8
        address         192.168.23.27
        parents         switch-2
        icon_image      win40.png
}
```

```
  GNU nano 5.4                    /usr/local/nagios/etc/objects/localhost.cfg
define host{
        use             windows-server
        host_name       pc-dev-9
        alias           PC Desarrollo 9
        address         192.168.23.28
        parents         switch-2
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-dev-10
        alias           PC Desarrollo 10
        address         192.168.23.29
        parents         switch-2
        icon_image      win40.png
}
```

- 3 (Administración T.I.) IP 192.168.23.31 - 33

```
  GNU nano 5.4                /usr/local/nagios/etc/objects/localhost.cfg
################### PC_ADMINISTRACION_T.I.#################
define host{
        use             windows-server
        host_name       pc-ti-1
        alias           PC Administracion T.I. 1
        address         192.168.23.31
        parents         switch-3
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-ti-2
        alias           PC Administracion T.I. 2
        address         192.168.23.32
        parents         switch-3
        icon_image      win40.png
}
define host{
        use             windows-server
        host_name       pc-ti-3
        alias           PC Administracion T.I. 3
        address         192.168.23.33
        parents         switch-3
        icon_image      win40.png
}
```

- 1 servidor Debian 11 con 5 tarjetas de red para la implementación de un firewall por software con la IP 192.168.23.5

```
  GNU nano 5.4                  /usr/local/nagios/etc/objects/localhost.cfg
############### SERVIDOR DE FIREWALL ##############################
define host{
        use             linux-server
        host_name       firewall-server
        alias           Servidor Firewall
        address         192.168.23.5
        parents         router-principal
        icon_image      firewall.gif
}
```

Se hace uso del comando “/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg” con el fin de verificar que todo esté correcto dentro del archivo

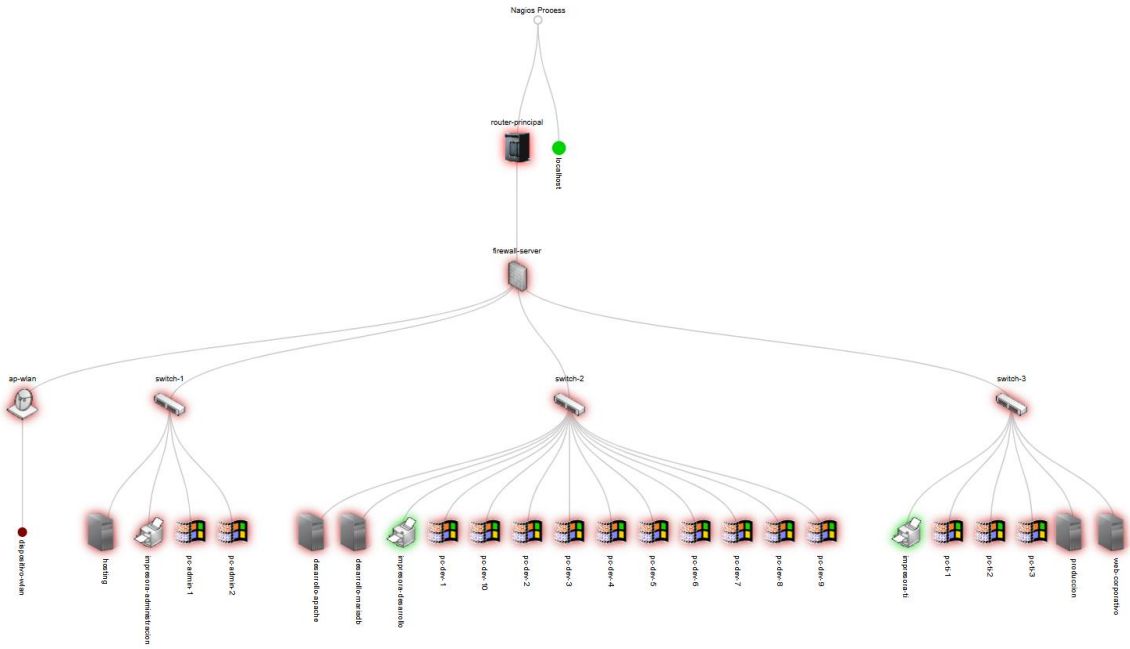
```
root@bgarri01:~# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

```
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2023-08-01
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 8 services.
        Checked 31 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 31 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
root@bgarri01:~#
```

Se reinicia nagios con el comando “**systemctl restart nagios**” para guardar los cambios y se valida que los cambios hayan sido guardados accediendo nuevamente a 192.168.23.165/nagios

```
root@bgarri01:~# systemctl restart nagios
```

Y así quedaría la estructura presentada con las imágenes de cada dispositivo:



La estructura se diseñó en base a principios básicos de arquitectura y las necesidades específicas que se describieron en los requerimientos:

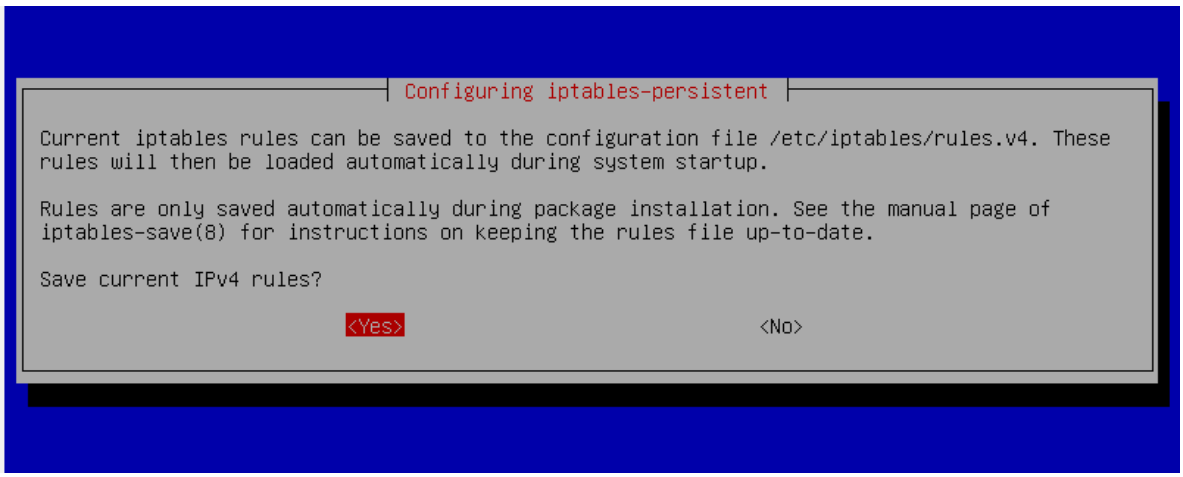
1. Basada en la Red Física y Lógica: Respeta el diseño real de la red descrito en los antecedentes (router, firewall, switches, dispositivos finales).
2. Cumple con los requerimientos del Firewall: Coloca al firewall-server como punto central para proteger todo el tráfico interno y externo, asegurando conexiones específicas y bloqueando no permitidas.
3. Segmentación Funcional: Los switches organizan la red por áreas (Administración, Desarrollo, T.I.), lo que facilita la administración y la seguridad.
4. Flujo Lógico del Tráfico:
    a. Tráfico externo: Router → Firewall → Dispositivos internos.
    b. Tráfico interno: Dispositivos → Switches → Firewall → Router.
5. Escalable y Modular: Permite agregar dispositivos o segmentos sin alterar la estructura general.
6. Optimización del Monitoreo: Refleja las dependencias físicas, facilitando la detección de fallos en dispositivos o segmentos.

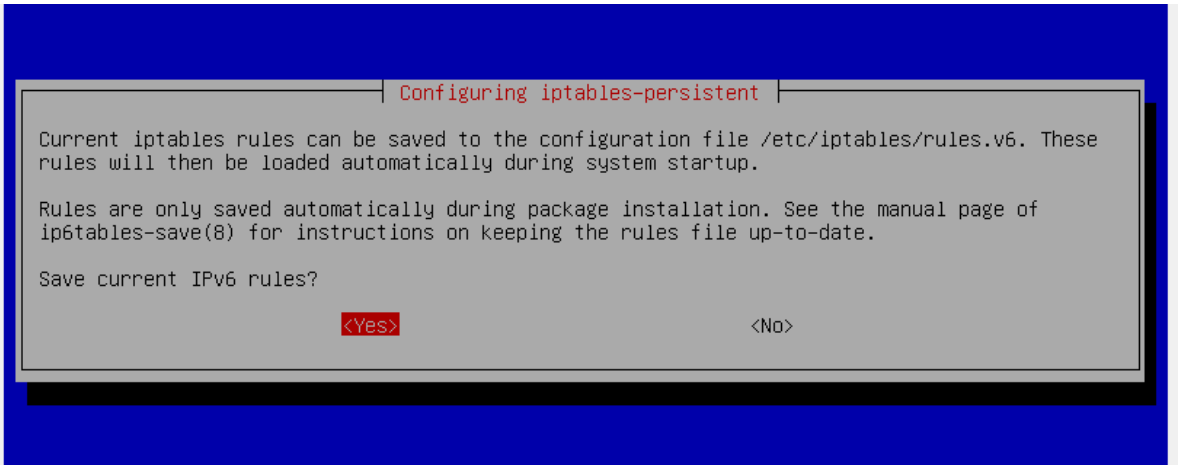
Esta estructura garantiza seguridad, facilidad de monitoreo y coherencia con los requerimientos de la empresa.

Firewall & proceso de Configuración de IPTables

Instalación de herramientas de Firewall

```
root@bgarri01:~# apt install iptables iptables-persistent -y
```





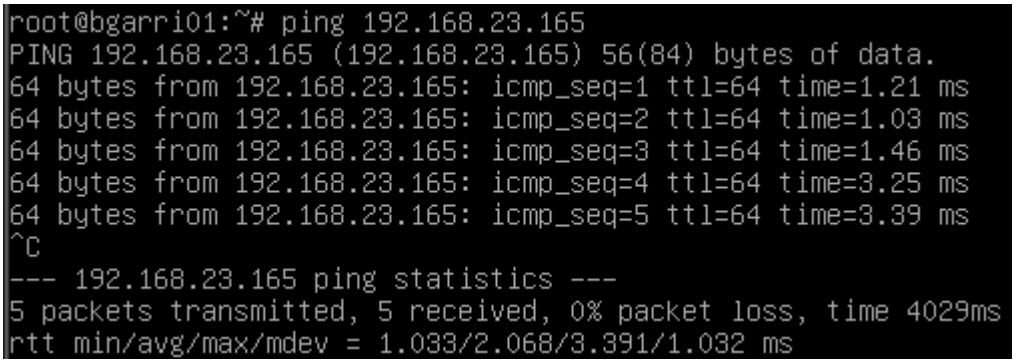

```
Selecting previously unselected package netfilter-persistent.
(Reading database ... 49226 files and directories currently installed.)
Preparing to unpack .../netfilter-persistent_1.0.15_all.deb ...
Unpacking netfilter-persistent (1.0.15) ...
Selecting previously unselected package iptables-persistent.
Preparing to unpack .../iptables-persistent_1.0.15_all.deb ...
Unpacking iptables-persistent (1.0.15) ...
Setting up netfilter-persistent (1.0.15) ...
Created symlink /etc/systemd/system/multi-user.target.wants/netfilter-persistent.service → /lib/syst
emd/system/netfilter-persistent.service.
Setting up iptables-persistent (1.0.15) ...
update-alternatives: using /lib/systemd/system/netfilter-persistent.service to provide /lib/systemd/
system/iptables.service (iptables.service) in auto mode
Processing triggers for man-db (2.9.4-2) ...
root@bgarri01:~# _
```

- Activamos el servicio **iptables-persistent** para asegurar que las reglas de firewall se guarden automáticamente después de cada reinicio.

Una vez instalados **iptables** e **iptables-persistent**, se accedió a la máquina 2 con el objetivo de realizar una serie de pruebas antes de configurar el firewall en la máquina 1.

Las pruebas consistieron en lo siguiente:

- **Ping** a la dirección IP 192.168.23.165 para verificar conectividad.



- **ssh** con el usuario bgarri a la máquina 192.168.23.165.

- **HTTP** mediante el comando curl http://192.168.23.153 para validar el acceso a servicios web. Para el comando curl, fue necesario instalar la herramienta en la máquina 2 ya que no estaba disponible por defecto.

```
root@bgarri01:~# apt install curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  linux-image-5.10.0-20-amd64
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  libcurl4
The following NEW packages will be installed:
  curl libcurl4
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 619 kB of archives.
After this operation, 1,200 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://security.debian.org/debian-security bullseye-security/main amd64 libcurl4 amd64 7.74.0-
1.3+deb11u14 [348 kB]
Get:2 http://security.debian.org/debian-security bullseye-security/main amd64 curl amd64 7.74.0-1.3+
deb11u14 [272 kB]
Fetched 619 kB in 0s (1,273 kB/s)
Selecting previously unselected package libcurl4:amd64.
(Reading database ... 37980 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.74.0-1.3+deb11u14_amd64.deb ...
Unpacking libcurl4:amd64 (7.74.0-1.3+deb11u14) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.74.0-1.3+deb11u14_amd64.deb ...
Unpacking curl (7.74.0-1.3+deb11u14) ...
Setting up libcurl4:amd64 (7.74.0-1.3+deb11u14) ...
Setting up curl (7.74.0-1.3+deb11u14) ...
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for libc-bin (2.31-13+deb11u11) ...
root@bgarri01:~# _
```

```
root@bgarri01:~# curl http://192.168.23.165
<html>
<body>
        <h1>Hola Mundo. Este es el servidor de Bayron_Garri_Mora </h1>
</body>
</html>

root@bgarri01:~#
```

Después de concluir las pruebas iniciales, se accedió a la máquina 1 para configurar **iptables**.

Luego, se configuró una política general para bloquear todo tráfico no autorizado. En la máquina 2, se verificó inicialmente enviando un **ping** a la máquina 1. Luego, en la máquina 1, se implementó la regla para bloquear tráfico de entrada por defecto:

```
root@bgarri01:~# iptables -P INPUT DROP
```

Y se volvió a probar el hacer ping desde la maquina02

```
root@bgarri01:~# ping 192.168.23.165
PING 192.168.23.165 (192.168.23.165) 56(84) bytes of data.
^C
--- 192.168.23.165 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5121ms

root@bgarri01:~#
```

A continuación, se repitieron las pruebas de **ping**, **SSH** y **curl** desde la máquina02, confirmando que las conexiones fueron bloqueadas correctamente.

```
root@bgarri01:~# curl http://192.168.23.165
^C
root@bgarri01:~# ssh bgarri@192.168.23.165
^C
root@bgarri01:~#
```

Tras comprobar que **iptables** estaba funcionando según lo esperado, se procedió a configurar resto de políticas.

```
root@bgarri01:~# iptables -P INPUT DROP
root@bgarri01:~# iptables -P FORWARD DROP
root@bgarri01:~# iptables -P OUTPUT DROP
```

Seguidamente se procedió a configurar las reglas para permitir tráfico autorizado:

- **Permitir conexiones SSH desde redes autorizadas:**
  - Se concedió acceso al puerto SSH únicamente a las redes **192.168.23.0/24**, **200.27.0.1/24**, y **146.83.1.0/24** con el siguiente comando:

```
root@bgarri01:~# iptables -A INPUT -p tcp -s 192.168.23.0/24 --dport 22 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -s 200.27.0.0/24 --dport 22 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -s 146.83.1.0/24 --dport 22 -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p tcp -d 192.168.23.0/24 --sport 22 -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p tcp -d 200.27.0.0/24 --sport 22 -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p tcp -d 146.83.1.0/24 --sport 22 -j ACCEPT
```

(ESCRIBIR AQUÍ QUE SIGNIFICA CADA UNA)

- **Habilitar tráfico HTTP/HTTPS a servidores web:**
  - Se permitió el acceso al tráfico HTTP y HTTPS para los servidores Web Corporativo (200.27.0.23) y Hosting (200.27.0.24):

```
root@bgarri01:~# iptables -A INPUT -p tcp -d 200.27.0.23 --dport 80 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -d 200.27.0.23 --dport 443 -j ACCEPT
root@bgarri01:~#
root@bgarri01:~# iptables -A OUTPUT -p tcp -s 200.27.0.23 -j ACCEPT
root@bgarri01:~#
root@bgarri01:~# iptables -A INPUT -p tcp -d 200.27.0.24 --dport 80 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -d 200.27.0.24 --dport 443 -j ACCEPT
root@bgarri01:~#
root@bgarri01:~# iptables -A OUTPUT -p tcp -s 200.27.0.24 -j ACCEPT
```

- Permitir conexiones a otros servidores solo desde la red interna:
  - Se permitió el acceso a los servidores de Desarrollo, Producción y MariaDB únicamente desde la red interna de la empresa (192.168.23.0/24):

```
root@bgarri01:~# iptables -A INPUT -p tcp -s 192.168.23.0/24 -d 192.168.23.3 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -s 192.168.23.0/24 -d 192.168.23.2 -j ACCEPT
root@bgarri01:~# iptables -A INPUT -p tcp -s 192.168.23.0/24 -d 192.168.23.4 -j ACCEPT
root@bgarri01:~#
root@bgarri01:~# iptables -A OUTPUT -p tcp -s 192.168.23.3 -d 192.168.23.0/24 -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p tcp -s 192.168.23.2 -d 192.168.23.0/24 -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p tcp -s 192.168.23.4 -d 192.168.23.0/24 -j ACCEPT
```

- Permitir tráfico ICMP para monitoreo:
    - Finalmente, se habilitó el tráfico ICMP para permitir el uso de ping con fines de monitoreo:

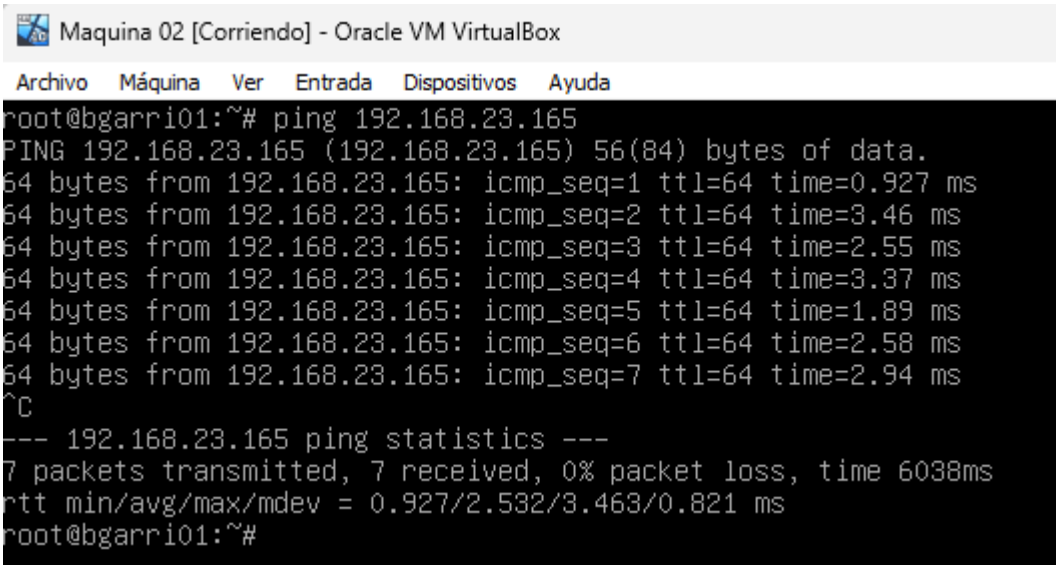
```
root@bgarri01:~# iptables -A INPUT -p icmp -j ACCEPT
root@bgarri01:~# iptables -A OUTPUT -p icmp -j ACCEPT
```

Para garantizar que las reglas configuradas persistan tras reiniciar la máquina, se utilizaron los comandos de iptables-persistent para guardar las reglas actuales en el archivo **rules.v4**:

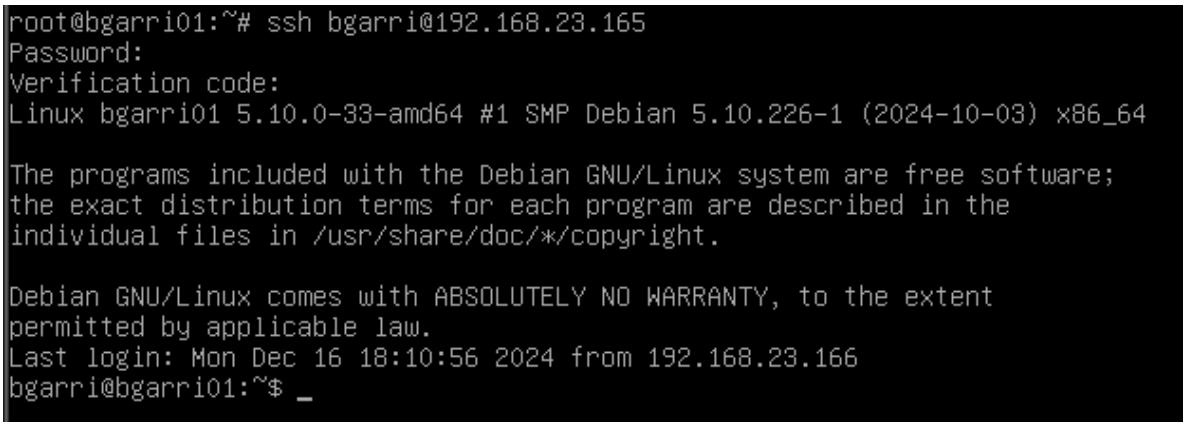
```
root@bgarri01:~# iptables-save | tee /etc/iptables/rules.v4
# Generated by iptables-save v1.8.7 on Mon Dec 16 18:37:19 2024
*filter
:INPUT DROP [2790:1024875]
:FORWARD DROP [0:0]
:OUTPUT DROP [3397:284357]
-A INPUT -s 192.168.23.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 200.27.0.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -s 146.83.1.0/24 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -d 200.27.0.23/32 -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -d 200.27.0.23/32 -p tcp -m tcp --dport 443 -j ACCEPT
-A INPUT -d 200.27.0.24/32 -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -d 200.27.0.24/32 -p tcp -m tcp --dport 443 -j ACCEPT
-A INPUT -s 192.168.23.0/24 -d 192.168.23.3/32 -p tcp -j ACCEPT
-A INPUT -s 192.168.23.0/24 -d 192.168.23.2/32 -p tcp -j ACCEPT
-A INPUT -s 192.168.23.0/24 -d 192.168.23.4/32 -p tcp -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A OUTPUT -d 192.168.23.0/24 -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -d 200.27.0.0/24 -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -d 146.83.1.0/24 -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -s 200.27.0.23/32 -p tcp -j ACCEPT
-A OUTPUT -s 200.27.0.24/32 -p tcp -j ACCEPT
-A OUTPUT -s 192.168.23.3/32 -d 192.168.23.0/24 -p tcp -j ACCEPT
-A OUTPUT -s 192.168.23.2/32 -d 192.168.23.0/24 -p tcp -j ACCEPT
-A OUTPUT -s 192.168.23.4/32 -d 192.168.23.0/24 -p tcp -j ACCEPT
-A OUTPUT -p icmp -j ACCEPT
COMMIT
# Completed on Mon Dec 16 18:37:19 2024
root@bgarri01:~#
```

Una vez completada la configuración del firewall, se realizaron nuevamente las pruebas en la máquina 2 para comprobar el funcionamiento de las reglas:

- **Ping** a la máquina 1 con IP 192.168.23.165.



- **ssh** con el usuario autorizado desde la maquina 2 a la máquina 1 con IP 192.168.23.165.



- **Acceso HTTP/HTTPS** utilizando curl.

Los resultados confirmaron que las reglas implementadas en **iptables** funcionaron de acuerdo con los requerimientos, permitiendo únicamente las conexiones autorizadas y bloqueando todo el tráfico no especificado.