

curso “Experto en Ciberseguridad de Redes y Sistemas”

TEMAS

1. Fingerprinting
2. Footprinting
3. IDSs - Snort
4. Nessus
- 5. Firewalls**
6. SGSI
7. Antiphishing
8. MITRE
9. Red Team y Blue Team
10. Principales procesos de Seguridad



Firewalls

5.1 Firewalls – Presentación

Primera charla de esta serie de videos dedicados totalmente a Firewalls, empleando inicialmente "**iptables**" y luego "**nftables**" que son las mejores herramientas DEL MUNDO para gestionar estos dispositivos. Emplearemos también la interfaz gráfica "**Firewall Builder**" para que puedas ampliar la capacidad de gestión de estos dispositivos y aplicarlos a gran escala. Como corresponde, TODO con Open Source.

Hoy presentaremos los conceptos iniciales, tomando como referencia nuestro libro "**Seguridad por Niveles**", en el punto 6.3 “Firewalls”, que recuerda, puedes descargarlo gratuitamente (y los otros cinco también) desde nuestra Web:

www.darFe.es



Un **firewall** es un sistema de seguridad que actúa como una barrera entre una red interna (como la de una empresa o un hogar) y redes externas, como Internet. Su función principal es **controlar el tráfico de datos** que entra y sale, permitiendo únicamente las comunicaciones autorizadas y bloqueando aquellas que puedan suponer un riesgo.

Su funcionamiento se basa en un conjunto de **reglas predefinidas** que analizan aspectos como la dirección IP de origen y destino, los puertos utilizados o el tipo de protocolo. Cuando un paquete de datos intenta atravesar el firewall, este lo examina y decide si lo deja pasar o lo rechaza según dichas reglas.

Existen distintos tipos de firewall, desde los más simples, que filtran paquetes, hasta los más avanzados, capaces de inspeccionar el contenido del tráfico en profundidad (*deep packet inspection*). Gracias a ello, son una herramienta fundamental para **prevenir accesos no autorizados, ataques informáticos y fugas de información**.

Vamos a ver el tema con más detalle en el video 121 (**Obligatorio**).

 **Firewalls - (Parte 1: Presentación) - Charla 121** - Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/alg5ffU4K7Y>



5.2 Firewalls – Funcionamiento

En este capítulo veremos el funcionamiento de un Firewall, sus políticas **permitivas y restrictivas**. Concepto de políticas y reglas, formato de las reglas de un firewall. Acciones de cada regla. Orígenes de "iptables", su historia "ipfwadm" e "ipchains". Descárgate el artículo "Teoria_y_practica_con_Firewalls.pdf" (es un lujo) desde: www.darFe.es

Historia de iptables, ipchains e ipfwadm (ipfadmin)

La evolución de los sistemas de filtrado de paquetes en Linux ha pasado por varias etapas hasta llegar a las soluciones actuales. Herramientas como **iptables**, **ipchains** y **ipfwadm** (a veces llamado erróneamente *ipfadmin*) han sido fundamentales en este desarrollo.

ipfwadm (ipfadmin)

ipfwadm fue una de las primeras herramientas de firewall en Linux, utilizada en los kernels 2.0. Su función era permitir la configuración de reglas básicas de filtrado de paquetes mediante listas simples. Aunque supuso un gran avance en su momento, tenía limitaciones importantes:

- No mantenía el estado de las conexiones
- Gestión poco flexible de reglas
- Sintaxis limitada

Aun así, sentó las bases del filtrado de red en sistemas Linux.

ipchains

Con la llegada del kernel 2.2 apareció **ipchains**, que mejoró significativamente a ipfwadm. Introdujo una estructura más organizada basada en cadenas (*chains*), lo que permitía una gestión más clara del tráfico.

Entre sus mejoras destacaban:

- Mayor control sobre paquetes
- Mejor organización de reglas
- Más flexibilidad en el filtrado

Sin embargo, todavía carecía de una gestión avanzada del estado de las conexiones, lo que limitaba su capacidad frente a ataques más complejos.

iptables

El siguiente gran paso llegó con **iptables**, introducido en el kernel 2.4 junto con el sistema **Netfilter**. Esta herramienta supuso una revolución en la gestión de firewalls en Linux.

Sus principales avances fueron:

- **Filtrado con estado (stateful)**: permite seguir conexiones completas, no solo paquetes individuales
- **Tablas y cadenas avanzadas** (filter, nat, mangle, etc.)
- **Mayor modularidad y extensibilidad**
- Mejor rendimiento y control detallado del tráfico

iptables se convirtió durante años en el estándar de facto para la configuración de firewalls en Linux, siendo ampliamente utilizado tanto en servidores como en entornos empresariales.

Evolución posterior

Con el tiempo, iptables ha ido siendo sustituido progresivamente por herramientas más modernas como **nftables**, que unifica y simplifica la gestión del filtrado de red en sistemas Linux actuales.

Resumen

La evolución puede entenderse como una mejora progresiva:

- **ipfwadm** → básico y limitado
- **ipchains** → más organizado y flexible
- **iptables** → potente, modular y con control de estado

Este desarrollo refleja cómo la seguridad en redes ha ido adaptándose a amenazas cada vez más complejas.

Vamos a ver el tema con más detalle en el video 122 (**Obligatorio**).

 **Firewalls - (Parte 2: Funcionamiento) - Charla 122** -
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/67AdHRIGByE>



5.3 Firewalls – Iptables y Netfilter

Ya nos conocéis, y sabéis bien, que la "didáctica y metodología" son algo fundamental para nosotros, por eso hoy comenzamos con el artículo "**Teoria_y_practica_con_Firewalls.pdf**" que hemos publicado en www.darFe.es y, de verdad es un lujo de artículo de 40 páginas, en el que avanzamos

"paso a paso" y por medio de ejercicios y prácticas, primero con "iptables", luego con "Firewall Builder" y finalmente con "nftables". La base de este artículo son los ejercicios prácticos y amigables para que puedas dominar al detalle estos firewalls Open Source. Descárgate el artículo "Teoria_y_practica_con_Firewalls.pdf" (es un lujo) desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Comenzaremos con la presentación del proyecto "Netfilter", que puedes ver todo su desarrollo en la siguiente URL:

<https://netfilter.org>

Vamos a ver el tema con más detalle en el video 123 (**Obligatorio**).

 **Firewalls - (Parte 3: Artículo de Firewalls) - Charla 123 -**
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/FYM5v-yv3DU>



5.4 Firewalls – iptables, primeros pasos

Llegó "la chicha" de la serie sobre Firewalls. Hoy comenzamos a mojarnos y a picar líneas de código, por supuesto con la interfaz de comandos de nuestro "Kali" (no podía ser de otra forma). Lanzamos los primeros comandos de "iptables", los más sencillos para que empieces a amigarte con esta maravilla de herramienta que nos ofrece Linux.

Tablas, cadenas y acciones en iptables

En **iptables**, la gestión del tráfico de red se organiza mediante tres conceptos fundamentales: **tablas**, **cadenas (chains)** y **acciones (targets)**. Comprender cómo se relacionan es clave para configurar un firewall eficaz en sistemas Linux.

Tablas

Las **tablas** son contenedores que agrupan reglas según su propósito. Las principales son:

- **filter**: es la tabla por defecto y se utiliza para permitir o bloquear tráfico.
- **nat**: se encarga de la traducción de direcciones (por ejemplo, NAT o redirecciones de puertos).
- **mangle**: permite modificar paquetes (como cambiar cabeceras o marcar tráfico).
- **raw**: se usa para configuraciones avanzadas, como excluir paquetes del seguimiento de conexiones.

Cada tabla contiene sus propias cadenas y reglas.

Cadenas (chains)

Las **cadenas** son listas ordenadas de reglas que se aplican en distintos puntos del flujo de red. Las más importantes son:

- **INPUT**: procesa los paquetes que llegan al propio sistema.

- **OUTPUT:** gestiona los paquetes que salen desde el sistema.
- **FORWARD:** controla los paquetes que pasan a través del sistema (por ejemplo, si actúa como router).

En otras tablas también existen cadenas específicas, como:

- **PREROUTING:** antes de que el paquete sea enroutado
- **POSTROUTING:** después de decidir su destino

Cada paquete que atraviesa el sistema pasa por una o varias de estas cadenas.

Acciones (targets)

Las **acciones** indican qué hacer cuando un paquete coincide con una regla. Algunas de las más comunes son:

- **ACCEPT:** permite el paso del paquete
- **DROP:** bloquea el paquete sin notificar al origen
- **REJECT:** bloquea el paquete y envía una respuesta de rechazo
- **LOG:** registra información del paquete
- **DNAT / SNAT:** modifican direcciones en la tabla NAT

Cómo trabajan juntos

Cuando un paquete llega al sistema:

- 1) Entra en una **tabla** según el tipo de procesamiento necesario
- 2) Recorre una **cadena** específica (como INPUT o FORWARD)
- 3) Se compara con las **reglas** en orden
- 4) Al coincidir, se aplica una **acción**

Si no coincide con ninguna regla, se aplica la política por defecto de la cadena (por ejemplo, permitir o denegar).

Resumen

- **Tablas** → agrupan reglas por función
- **Cadenas** → definen el recorrido del paquete
- **Acciones** → determinan qué hacer con el paquete

Este modelo modular hace que iptables sea una herramienta potente y flexible para el control del tráfico de red.

Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 124 **(Obligatorio)**.

🌐 **Firewalls - (Parte 4: Comienzan los ejercicios) - Charla 124 -**
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/2C1eaq5CLis>



5.5 Firewalls – iptables, ajuste de reglas

Sigue lo bueno con "iptables", en este capítulo creamos reglas mucho más ajustadas y verificaremos su funcionamiento con Wireshark. Con este tema ya estarás en capacidad de dominar bastante más las reglas de filtrado de este firewall.

Vamos a ver el tema con más detalle en el video 125 **(Obligatorio)**.

🌐 **Firewalls - (Parte 5: mejorando reglas) - Charla 125 -**
Aprendiendo Ciberseguridad paso a paso

https://youtu.be/PQs_Y_CIOFg



5.6 Firewalls – reglas HTTP

Importancia de controlar reglas HTTP y HTTPS en iptables

El control del tráfico **HTTP** (puerto 80) y **HTTPS** (puerto 443) en un firewall como **iptables** es fundamental para garantizar la seguridad de cualquier sistema conectado a una red. Estos protocolos son los más utilizados para la navegación web y, precisamente por ello, también son uno de los principales vectores de ataque.

Protección frente a accesos no autorizados

Permitir o bloquear correctamente el tráfico HTTP/HTTPS evita que servicios web queden expuestos innecesariamente. Una mala configuración puede permitir que atacantes accedan a aplicaciones internas o paneles de administración que deberían estar restringidos.

Prevención de ataques comunes

El control de estas reglas ayuda a mitigar ataques como:

- **Fuerza bruta** sobre formularios web
- **Escaneo de vulnerabilidades**
- **Explotación de fallos en aplicaciones web**
- **Denegación de servicio (DoS)**

Limitando el acceso o aplicando filtros, se reduce significativamente la superficie de ataque.

Control del tráfico saliente

No solo es importante el tráfico entrante. Restringir conexiones HTTP/HTTPS salientes permite:

- Evitar que malware se comunique con servidores externos
- Controlar el acceso de usuarios a sitios no autorizados
- Detectar comportamientos sospechosos en la red

Segmentación y acceso controlado

Mediante reglas específicas, se puede:

- Permitir acceso web solo desde ciertas IPs
- Restringir servicios a redes internas
- Redirigir tráfico a proxies o sistemas de inspección

Esto es clave en entornos empresariales donde se requiere un control granular.

Diferencia clave entre HTTP y HTTPS

Mientras HTTP transmite datos sin cifrar, HTTPS añade cifrado, lo que protege la información en tránsito. Sin embargo, ambos deben ser controlados:

- HTTP puede ser utilizado para ataques o redirecciones maliciosas
- HTTPS, aunque seguro en cifrado, puede ocultar tráfico malicioso si no se inspecciona adecuadamente

Conclusión

Controlar correctamente las reglas de HTTP y HTTPS en iptables es esencial para mantener la seguridad, reducir riesgos y asegurar que solo el tráfico legítimo accede o sale del sistema. Una configuración adecuada no solo protege servicios, sino que también ayuda a detectar y prevenir amenazas en tiempo real.

Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 126 **(Obligatorio)**.

 **Firewalls - (Parte 6: iptables - Web) - Charla 126** -
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/fIJTZI8hqEI>



5.7 Firewalls – SSH

En este ejercicio, trabajaremos con dos portátiles y una máquina virtual para simular el empleo de una "Máquina de salto" y para fortalecer con "iptables" nuestras redes de gestión, evitando lo que se conoce como "**salto lateral**". Para ello veremos como aplicar filtros de iptables en los hosts para que únicamente permita la conexión desde la dirección IP de la máquina de salto.

Salto lateral en ciberseguridad

El **salto lateral** (en inglés, *lateral movement*) es una técnica utilizada por atacantes una vez que han conseguido acceso inicial a un sistema dentro de una red. Consiste en **moverse de un equipo a otro dentro de la misma red** con el objetivo de ampliar el acceso, escalar privilegios y alcanzar sistemas más valiosos.

¿En qué consiste?

En lugar de atacar directamente el objetivo principal desde el exterior, el atacante entra por un punto más débil (por ejemplo, un equipo de usuario) y desde ahí se desplaza internamente aprovechando:

- Credenciales robadas
- Vulnerabilidades en otros equipos
- Configuraciones inseguras

Este movimiento suele ser discreto para evitar ser detectado.

Objetivos del salto lateral

- Acceder a servidores críticos (bases de datos, controladores de dominio)
- Obtener privilegios de administrador
- Robar información sensible
- Mantener persistencia dentro de la red

Ejemplos de salto lateral

1) Uso de credenciales robadas

Un atacante compromete el ordenador de un empleado y obtiene sus credenciales. Después, utiliza esas credenciales para acceder a otros equipos de la red mediante protocolos como RDP o SMB.

2) Ataque a un servidor interno

Tras acceder a una máquina, el atacante detecta un servidor mal configurado en la red interna y lo explota para obtener mayor control.

3) Pass-the-Hash

El atacante no necesita la contraseña en texto plano, sino el hash almacenado en memoria. Con él puede autenticarse en otros sistemas sin conocer la contraseña real.

4) Movimiento mediante herramientas legítimas

El atacante utiliza herramientas del propio sistema (como PowerShell o PsExec) para moverse sin levantar sospechas, simulando actividad normal.

Importancia de prevenirlo

El salto lateral es especialmente peligroso porque:

- Permite a los atacantes expandir el alcance del ataque
- Dificulta la detección al usar recursos internos
- Puede comprometer toda la infraestructura si no se controla

Medidas de mitigación

- Segmentación de red (por medio de los comandos que veremos en este tema)
- Uso de autenticación multifactor
- Monitorización de accesos internos
- Principio de mínimo privilegio
- Detección de comportamientos anómalos

Conclusión

El salto lateral es una fase clave en muchos ataques avanzados. Detectarlo y prevenirlo es esencial para evitar que una intrusión inicial se convierta en una brecha de seguridad grave.

Vamos a ver el tema con más detalle en el video 127 **(Obligatorio)**.

 **Firewalls - (Parte 7: SSH - máquina de salto) - Charla 127**
- Aprendiendo Ciberseguridad paso a paso

https://youtu.be/uCU_nVk5agw



5.8 Firewalls – Defensa en profundidad

En esta charla desarrollamos la seguridad perimetral de una red que emplea "defensa en profundidad" y ejercitaremos reglas sencillas de iptables para regular las diferentes entradas y salidas por los puertos TCP (que también aplica a UDP), considerando ya dos interfaces de red (input y output). Aprenderemos también a crear interfaces virtuales de red en una máquina virtual Kali.

La **defensa en profundidad** (*defense in depth*) es un principio fundamental de la ciberseguridad que consiste en **proteger un sistema mediante múltiples capas de seguridad**, en lugar de depender de una única barrera. La idea clave es que, si un mecanismo falla, otros puedan seguir protegiendo el sistema.

Concepto clave

En lugar de confiar solo en un firewall o un antivirus, este enfoque propone desplegar **varios controles de seguridad distribuidos en distintos niveles** (red, sistemas, aplicaciones, usuarios, etc.). Así, un atacante tendría que superar múltiples obstáculos antes de comprometer completamente el entorno.

Capas típicas de defensa

La defensa en profundidad se suele estructurar en diferentes niveles:

- **Perímetro:** firewalls, sistemas de detección de intrusos (IDS/IPS)
- **Red interna:** segmentación de red, VLANs, control de tráfico
- **Host o equipo:** antivirus, control de acceso, endurecimiento del sistema
- **Aplicaciones:** validación de entradas, parches, autenticación segura
- **Datos:** cifrado, copias de seguridad, control de acceso

- **Usuario:** formación en seguridad, políticas de contraseñas

Cada capa protege frente a distintos tipos de amenazas.

Cómo funciona en la práctica

Imagina un ataque típico:

1. Un atacante intenta acceder desde Internet → el firewall filtra tráfico
2. Si logra entrar, encuentra una red segmentada → limita su movimiento
3. En un equipo, necesita credenciales → controles de acceso lo dificultan
4. Para acceder a datos → se enfrenta a cifrado y permisos

Este encadenamiento de barreras reduce enormemente las probabilidades de éxito.

Ventajas principales

- **Reducción del riesgo global**
- **Mayor capacidad de detección**
- **Resiliencia ante fallos individuales**
- **Dificulta el movimiento lateral del atacante**

Limitaciones

- No elimina el riesgo completamente
- Puede aumentar la complejidad de gestión
- Requiere una buena configuración y mantenimiento continuo

Conclusión

La defensa en profundidad no se basa en una única solución milagrosa, sino en la **combinación estratégica de múltiples medidas de seguridad**. Es uno de los enfoques más eficaces para proteger sistemas modernos frente a amenazas cada vez más sofisticadas.

Si queréis investigar el tema con más detalle, se desarrolla en el punto 2.2 Planificación de la Seguridad, de nuestro libro "**Seguridad en Redes**", que podéis descargar gratuitamente desde nuestra web:

<https://darfe.es>



Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 128 (**Obligatorio**).

 **Firewalls - (Parte 8: Proteger salidas y entradas) - Charla 128** - Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/UoxKOszG6fl>



5.9 Firewalls – Seguimiento de sesiones

¿Cómo hacer el seguimiento de las sesiones? NO TE LO PIERDAS. Como ya hemos desarrollado en las charlas del nivel de transporte (Charlas 63 a 76 de nuestro ciclo), el establecimiento de sesiones TCP es una de las tareas más importante que tiene este protocolo. Una inadecuada gestión, supervisión y monitorización de las mismas, es foco de varios tipos de ataques. En la charla de hoy, veremos como "iptables" nos ofrece opciones muy potentes para poder realizar todo el seguimiento de las mismas. En la jerga de FWs, esto se define como "Firewalls con control de estados".

Iptables implementa un modelo de **filtrado con estado** (*stateful firewall*), lo que significa que no solo analiza paquetes de forma aislada, sino que es capaz de **seguir el estado de las conexiones o sesiones de red**.

Esto se logra gracias a un subsistema del kernel de Linux llamado **Netfilter**, concretamente mediante el módulo de *connection tracking* (conntrack).

¿Qué es el seguimiento de sesiones?

El **seguimiento de sesiones** consiste en que el firewall mantiene una tabla interna donde registra las conexiones activas (por ejemplo, una conexión TCP entre cliente y servidor). Así puede identificar si un paquete:

- Pertenece a una conexión ya establecida
- Es parte de una nueva conexión
- Es inválido o sospechoso

Esto permite tomar decisiones mucho más inteligentes que el filtrado tradicional.

Estados de conexión en iptables

iptables clasifica los paquetes en varios estados:

- **NEW** → paquete que inicia una nueva conexión
- **ESTABLISHED** → pertenece a una conexión ya activa
- **RELATED** → conexión asociada a otra (por ejemplo, FTP)
- **INVALID** → paquete corrupto o no reconocido

Estos estados se utilizan en reglas mediante el módulo `-m state` o `-m conntrack`.

Ejemplo típico de uso

Una configuración muy común en firewalls es:

- Permitir conexiones ya establecidas

- Permitir nuevas conexiones solo en ciertos puertos
- Bloquear el resto

Esto se traduce conceptualmente en:

- Aceptar tráfico **ESTABLISHED, RELATED**
- Permitir **NEW** solo en servicios específicos (como SSH o HTTP)
- Denegar lo demás

Ventajas del filtrado con estado

El seguimiento de sesiones aporta importantes beneficios:

- **Mayor seguridad:** evita respuestas a conexiones no iniciadas
- **Menos reglas necesarias:** simplifica la configuración
- **Mejor control del tráfico:** entiende el contexto de la comunicación
- **Protección frente a ataques:** como spoofing o paquetes maliciosos

Ejemplo práctico

Cuando visitas una web:

- 1) Tu equipo inicia una conexión (estado NEW)
- 2) El servidor responde (estado ESTABLISHED)
- 3) El firewall permite automáticamente las respuestas sin abrir todos los puertos

Sin seguimiento de sesiones, habría que permitir manualmente mucho más tráfico, aumentando el riesgo.

Conclusión

El seguimiento de sesiones convierte a iptables en un firewall **inteligente y eficiente**, capaz de entender el contexto de las comunicaciones. Es una pieza clave para construir configuraciones seguras sin necesidad de reglas excesivamente complejas.

Vamos a ver el tema con más detalle en el video 129 **(Obligatorio)**.

 **Firewalls - (Parte 9: Control de estados) - Charla 129 -**
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/y7CNpzbeOos>



5.10 Firewalls – Gestión de Logs

Si no gestionas bien tus Logs... mal futuro...

En la charla "Syslog (ejercicio, teoría y práctica)"

<https://www.youtube.com/watch?v=sXC0VSLhWXk>



Se presentó como funciona el sistema de "**Syslog**" de Unix/Linux. La adecuada gestión de Logs, será el pilar fundamental para la gestión de incidentes, los que ya sabes, que "sí o sí" sucederán en tus infraestructuras de Ciberseguridad. En este capítulo, veremos cómo se deben gestionar los mismos desde "iptables" para que puedas enviar los **Logs** de las reglas que consideres más importantes. También desarrollaremos cómo implementar "rsyslog" en Kali Linux.

La **gestión de logs** en un firewall como **iptables** es esencial para **monitorizar, auditar y detectar incidentes de seguridad**. Los registros permiten saber qué tráfico está siendo permitido, bloqueado o considerado sospechoso, ofreciendo visibilidad sobre lo que ocurre en la red.

¿Cómo funciona el registro en iptables?

Iptables no almacena logs por sí mismo, sino que utiliza el sistema de registro del kernel de Linux (como **syslog** o **journald**). Para generar eventos, se emplea la acción:

- **LOG** → registra información del paquete que coincide con una regla

Cuando una regla con LOG se activa, el evento se envía al sistema de logs, donde puede ser consultado o procesado.

Qué información se registra

Los logs generados pueden incluir:

- Dirección IP de origen y destino
- Puertos utilizados
- Protocolo (TCP, UDP, ICMP)
- Interfaz de red
- Estado de la conexión

Además, se puede añadir un **prefijo personalizado** para identificar fácilmente el tipo de evento.

Ejemplo conceptual

Una regla de logging suele colocarse antes de una acción como DROP:

- Registrar el paquete sospechoso
- Bloquearlo posteriormente

Esto permite saber **qué se está bloqueando y por qué**, algo clave para análisis forense o ajustes de reglas.

Buenas prácticas

Para una gestión eficaz de logs:

- **No registrar todo el tráfico** → evitar saturación de logs
- **Usar prefijos claros** → facilita el análisis posterior
- **Registrar solo eventos relevantes** → intentos fallidos, accesos no autorizados
- **Rotar logs** → evitar que ocupen demasiado espacio
- **Centralizar registros** → mediante servidores de logs (SIEM)

Riesgos de una mala gestión

- Exceso de logs → dificulta encontrar eventos importantes
- Falta de logs → imposibilidad de investigar incidentes
- Logs sin analizar → pérdida de información crítica

Importancia en ciberseguridad

La gestión de logs en iptables permite:

- Detectar intentos de intrusión
- Identificar patrones de ataque
- Auditar accesos y políticas de red
- Responder a incidentes con evidencias

Conclusión

Los logs convierten al firewall en una herramienta no solo de protección, sino también de **visibilidad y análisis**. Una configuración adecuada de registro en iptables es clave para mantener el control y la seguridad de cualquier infraestructura.

Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 131 (**Obligatorio**).

 **Firewalls - (Parte 10: Gestión de Logs) - Charla 131 -**
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/tH7RdLXVRz8>



5.11 Firewalls – Guardar y recuperar reglas de iptables

Este breve capítulo es IMPORTANTÍSIMO, para que puedas ir almacenando los históricos de cada cambio que realices en tu firewall y de esta forma, luego restaurarlo en el momento que desees. También es una buena mecánica para implementar plantillas de reglas estándar que puedes aplicar en el proceso de "entrada en producción" de firewalls, pues sencillamente debes restaurar esta plantilla estándar en cada uno de los firewalls que vayas incorporando a tu organización y sobre la base de la misma, luego ampliar las reglas específicas de cada uno de ellos.

En **iptables**, las reglas configuradas en memoria **no son persistentes por defecto**. Esto significa que, si el sistema se reinicia, las reglas se pierden a menos que se guarden previamente. Por ello, es fundamental conocer cómo **guardar y restaurar** la configuración del firewall.

Guardar reglas

Para almacenar las reglas actuales, se utiliza el comando:

- **iptables-save**

Este comando genera una representación completa de la configuración (tablas, cadenas y reglas), que normalmente se redirige a un fichero:

- Ejemplo conceptual: guardar reglas en un archivo
→ permite hacer copias de seguridad o mantener configuraciones estables

Este fichero puede revisarse o editarse manualmente si es necesario.

Restaurar reglas

Para recuperar una configuración previamente guardada, se usa:

- **iptables-restore**

Este comando carga todas las reglas desde un fichero y las aplica directamente al sistema. Es rápido y eficiente, ya que reconstruye toda la configuración de una vez.

Persistencia automática

Dependiendo de la distribución de Linux, existen mecanismos para hacer persistentes las reglas:

- Servicios como iptables-persistent (en Debian/Ubuntu)
- Scripts de arranque que ejecutan iptables-restore al iniciar
- Integración con herramientas de gestión de red

Esto permite que las reglas se carguen automáticamente tras reiniciar el sistema.

Ubicación de los ficheros

Los archivos de reglas suelen almacenarse en rutas como:

- /etc/iptables/rules.v4
- /etc/sysconfig/iptables

Estas ubicaciones pueden variar según la distribución.

Buenas prácticas

- **Hacer copias de seguridad** antes de modificar reglas
- **Documentar cambios** en los ficheros
- **Probar configuraciones** antes de aplicarlas en producción
- **Evitar errores** que puedan bloquear el acceso remoto

Precaución importante

Al restaurar reglas, si hay errores en la configuración, se puede perder el acceso al sistema (por ejemplo, por SSH). Por eso, es recomendable:

- Tener acceso local o consola
- Aplicar cambios con cuidado

Conclusión

Guardar y recuperar reglas en iptables es esencial para mantener la seguridad y estabilidad del sistema. Permite **persistir configuraciones, restaurar estados conocidos y gestionar el firewall de forma profesional**.

Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 132 **(Obligatorio)**.

 **Firewalls - (Parte 11: iptables save y restore) - Charla 132 - Aprendiendo Ciberseguridad paso a paso**

<https://youtu.be/fg16xHu32nA>



5.12 Firewalls – iptables-persistent

¿Cómo podemos hacer persistentes nuestras reglas de iptables? En este capítulo, instalaremos y configuraremos el software "**iptables-persistent**" que nos permitirá ir almacenando nuestro histórico de políticas y reglas. Para que también podamos tener un valor agregado y redundancia en estas medidas, desarrollaremos también un sencillo "**script**" en "**bash**" y dejarlo en el demonio de arranque de la red (network) para que siempre que arranque nuestro sistema operativo, cargue las últimas reglas que hemos creado.

iptables-persistent es una herramienta utilizada en sistemas Linux (especialmente en distribuciones basadas en Debian/Ubuntu) que permite **guardar y restaurar automáticamente las reglas de iptables** al iniciar el sistema.

¿Para qué sirve?

Por defecto, las reglas de iptables se almacenan en memoria y se pierden al reiniciar el equipo. **iptables-persistent soluciona este problema**, haciendo que la configuración del firewall sea **permanente**.

Cómo funciona

Esta herramienta se basa en dos ideas simples:

1. **Guardar reglas actuales** en archivos de configuración
2. **Restaurarlas automáticamente** durante el arranque del sistema

Utiliza internamente los comandos:

- iptables-save → para guardar reglas
- iptables-restore → para cargarlas

Archivos de configuración

iptables-persistent guarda las reglas en ficheros como:

- /etc/iptables/rules.v4 → para IPv4

- /etc/iptables/rules.v6 → para IPv6

Estos archivos contienen toda la configuración del firewall y pueden editarse manualmente si es necesario.

Proceso de arranque

Cuando el sistema se inicia:

- 1) El servicio de iptables-persistent se ejecuta
- 2) Lee los archivos de reglas
- 3) Aplica automáticamente la configuración

Esto garantiza que el firewall esté activo desde el inicio, sin intervención manual.

Uso básico

Las acciones típicas incluyen:

- Guardar reglas actuales tras configurarlas
- Modificar archivos para ajustes avanzados
- Recargar reglas sin reiniciar el sistema

Ventajas

- **Persistencia automática** del firewall
- **Simplicidad de uso**
- **Integración con el sistema**
- **Evita configuraciones manuales tras reinicio**

Consideraciones

- Un error en los archivos puede bloquear accesos (por ejemplo, SSH)
- Es importante probar cambios antes de guardarlos
- Conviene mantener copias de seguridad

Conclusión

iptables-persistent es una herramienta clave para administrar firewalls en Linux de forma estable y profesional, asegurando que las reglas de iptables se mantengan activas incluso tras reinicios del sistema.

Seguimos con el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Vamos a ver el tema con más detalle en el video 133 (**Obligatorio**).

 **Firewalls - (Parte 12: netfilter-persistent) - Charla 133 -**
Aprendiendo Ciberseguridad paso a paso



<https://youtu.be/N1GFXGm5juo>

5.13 Firewalls – Ejercicios varios

Este capítulo, tal cual su nombre lo indica, se trata de una serie de ejercicios que también están planteados en el artículo "Teoria_y_practica_con_Firewalls.pdf", que te recordamos puedes descargar desde: www.darFe.es en el siguiente enlace:

<https://darfe.es/download/teoria-y-practica-con-firewalls-05/?wpdmdl=38017&refresh=6a6de7226a8951785587490>

Por lo que vamos directamente a verlo en el video 134 **(Obligatorio)**.

 **Firewalls - (Parte 13: Desafíos) - Charla 134 - Aprendiendo Ciberseguridad paso a paso**

<https://youtu.be/FoLKFXH6Png>



5.14 Firewalls – Gestión gráfica de reglas

¡¡Comenzamos con la interfaz gráfica!!

Cuando tenemos miles de reglas y/o varios firewalls, gestionarlos por línea de comandos, puede ser complicado y hasta peligroso. Las herramientas gráficas, nos ofrecen una serie de ventajas. En el video de este capítulo, comenzamos con algunos conceptos que figuran en nuestro libro "**Seguridad en Redes**" para luego pasar a la instalación de la herramienta gráfica que emplearemos, cuyo nombre es "**FirewalBuilder**".



Este tema se desarrolla en detalle, se desarrolla en el punto 6.11 Herramientas de Gestión de Firewalls, de nuestro libro "**Seguridad en Redes**", que podéis descargar gratuitamente desde nuestra web:

<https://darfe.es>



Contar con herramientas gráficas para la gestión de firewalls es importante por varias razones prácticas que van más allá de la simple comodidad. Aunque soluciones como *iptables* son muy potentes desde línea de comandos, su uso puede volverse complejo rápidamente, especialmente en entornos reales.

Una de las principales ventajas es la **reducción de errores humanos**. Cuando se trabaja con reglas manuales, es fácil equivocarse en el orden, la sintaxis o los parámetros, lo que puede dejar servicios expuestos o bloquear tráfico legítimo. Una interfaz gráfica suele validar entradas, mostrar advertencias y ayudar a evitar configuraciones incoherentes.

También mejora mucho la **visibilidad y comprensión de las reglas**. En iptables, las reglas se organizan en listas que pueden ser difíciles de interpretar cuando crecen. Las herramientas gráficas suelen representar las políticas de forma más clara (tablas, diagramas o flujos), lo que permite entender rápidamente qué tráfico está permitido o bloqueado.

Otro aspecto clave es la **gestión y mantenimiento a largo plazo**. En entornos empresariales, donde múltiples administradores trabajan sobre el mismo sistema, una interfaz gráfica facilita:

- Documentar reglas
- Revisar cambios
- Auditar configuraciones

Esto es fundamental para cumplir normativas de seguridad.

Además, estas herramientas suelen incluir funcionalidades adicionales como:

- **Monitorización en tiempo real del tráfico**
- **Visualización de logs**
- **Asistentes de configuración** para reglas comunes (HTTP, HTTPS, SSH, etc.)

Por último, son especialmente útiles para perfiles menos especializados o equipos mixtos, ya que reducen la barrera de entrada. No todo el mundo necesita dominar la sintaxis de iptables para gestionar un firewall de forma segura.

En resumen, las herramientas gráficas no sustituyen la potencia de iptables, pero sí aportan **claridad, seguridad y eficiencia**, lo que resulta crucial en la gestión diaria de la seguridad de red.

Vamos a ver el tema con más detalle en el video 135 (**Obligatorio**).

🌐 **Firewalls - (Parte 14: Interfaz gráfica, introducción) - Charla 135**
- Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/-JBP5CYNZCU>



5.15 Firewalls – configuración FWBuilder

Comenzamos a crear nuestro primer firewall desde la interfaz Gráfica empleando la herramienta "FirewallBuilder". Iremos viendo, siempre paso a paso, todas las facilidades que nos ofrece este software para preconfigurar la política de seguridad de cualquier tipo de firewall. Analizaremos rápidamente las primeras reglas que trae preconfiguradas.

Configurar tu primer firewall con **Firewall Builder** es bastante más sencillo de lo que parece si sigues una secuencia lógica. La herramienta está pensada para guiarte desde la definición de la red hasta el despliegue final. Aquí tienes los pasos básicos bien estructurados:

1) Crear un nuevo proyecto

Al abrir Firewall Builder:

- Crea un **nuevo archivo de trabajo** (File → New)

- Se lanzará un asistente (*wizard*) para crear un firewall
- Elige el tipo de firewall (por ejemplo, *iptables* si trabajas en Linux)

Aquí defines la base sobre la que se generarán todas las reglas.

2) Definir el firewall y sus interfaces

En este paso describes el propio dispositivo:

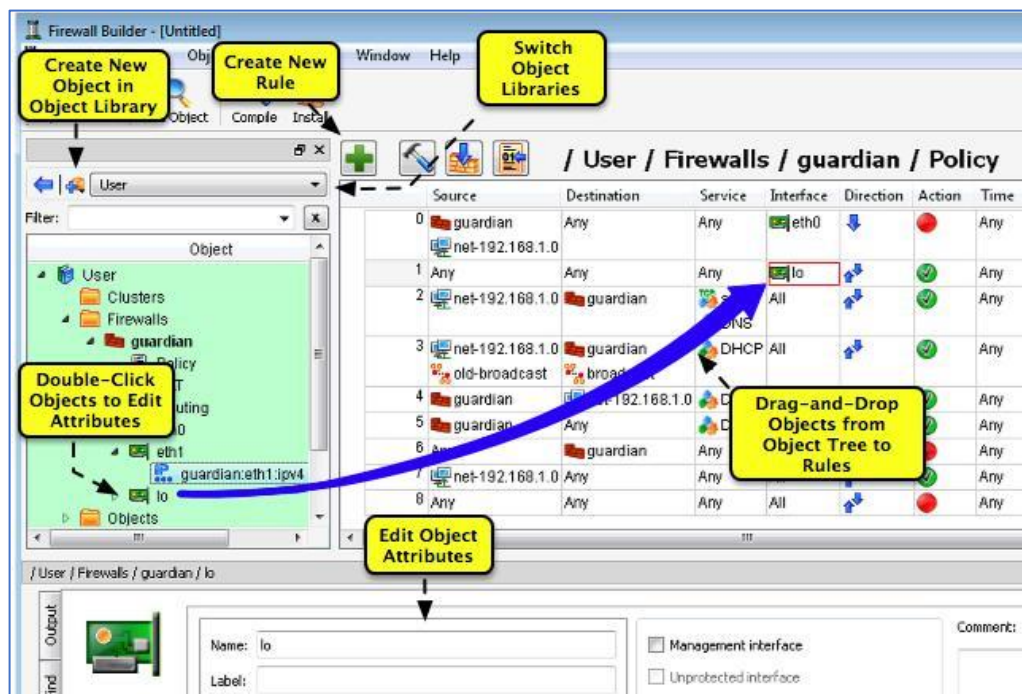
- Nombre del firewall
- Sistema operativo (Linux, BSD, etc.)
- Interfaces de red (por ejemplo):
 - eth0 → Internet (WAN)
 - eth1 → Red interna (LAN)

También puedes asignar:

- Direcciones IP
- Máscaras de red

Esto es clave porque las reglas dependerán de cómo está conectada la red.

3) Crear objetos de red



Antes de crear reglas, debes definir los “bloques”:

- **Redes** (ej: 192.168.1.0/24)
- **Hosts** (ej: servidor web)
- **Servicios:**

- HTTP (puerto 80)
- HTTPS (puerto 443)
- SSH (puerto 22)

Esto permite reutilizar elementos y mantener todo organizado.

4) Crear reglas de firewall (políticas)

Aquí defines qué tráfico permites o bloqueas.

Ejemplo típico de reglas básicas:

1. Permitir tráfico saliente desde la LAN a Internet
2. Permitir SSH desde tu IP de administración
3. Permitir HTTP/HTTPS hacia un servidor web
4. Bloquear todo lo demás (política por defecto)

Cada regla incluye:

- **Origen**
- **Destino**
- **Servicio**
- **Acción** (Accept, Deny, Reject)

👉 El orden es importante: se evalúan de arriba hacia abajo.

5) Configurar NAT (si es necesario)

Si tu red interna usa direcciones privadas:

- Crea reglas de **NAT** (masquerading)
- Ejemplo: permitir que la LAN salga a Internet usando la IP pública

Sin esto, los equipos internos no podrán navegar.

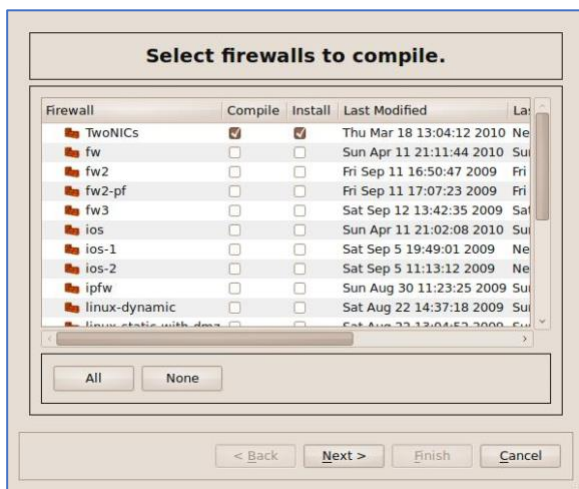
6) Compilar la configuración

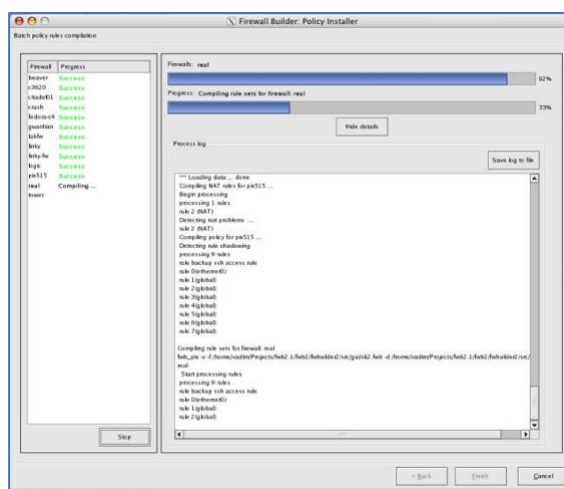
Una vez definidas las reglas:

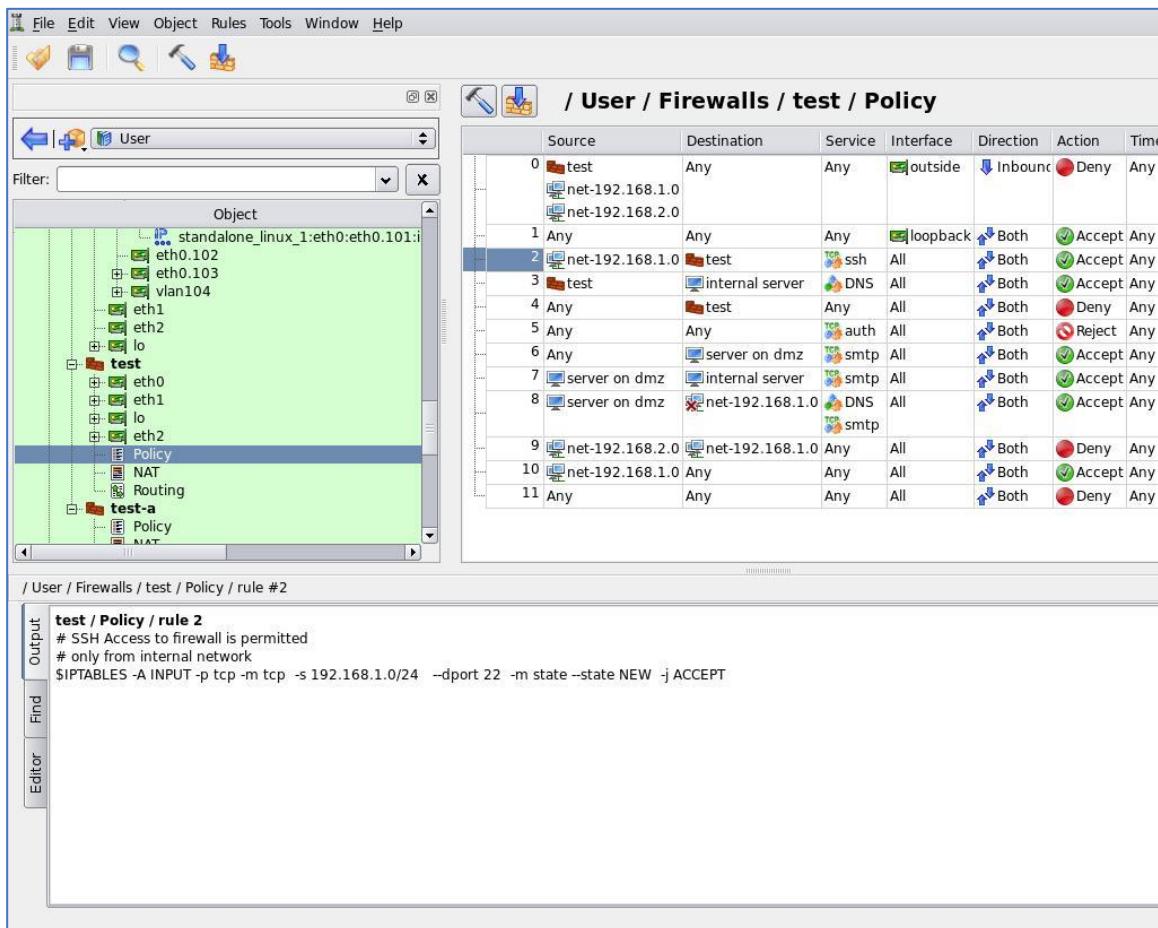
- Pulsa **Compile**
- Firewall Builder generará automáticamente el script (por ejemplo, reglas iptables)

Aquí puedes revisar si hay errores antes de aplicar nada.

7) Instalar las reglas en el firewall







	Source	Destination	Service	Interface	Direction	Action	Time
0	test	Any	Any	outside	Inbound	Deny	Any
1	Any	Any	Any	loopback	Both	Accept	Any
2	net-192.168.1.0	test	ssh	All	Both	Accept	Any
3	test	internal server	DNS	All	Both	Accept	Any
4	Any	test	Any	All	Both	Deny	Any
5	Any	Any	auth	All	Both	Reject	Any
6	Any	server on dmz	smtp	All	Both	Accept	Any
7	server on dmz	internal server	smtp	All	Both	Accept	Any
8	server on dmz	net-192.168.1.0	DNS	All	Both	Accept	Any
9	net-192.168.2.0	net-192.168.1.0	Any	All	Both	Deny	Any
10	net-192.168.1.0	Any	Any	All	Both	Accept	Any
11	Any	Any	Any	All	Both	Deny	Any

/ User / Firewalls / test / Policy / rule #2

test / Policy / rule 2

SSH Access to firewall is permitted
only from internal network

```
$IPTABLES -A INPUT -p tcp -s 192.168.1.0/24 --dport 22 -m state --state NEW -j ACCEPT
```

Finalmente:

- Usa la opción **Install**
- Conecta al firewall (normalmente por SSH)
- Sube y aplica las reglas automáticamente

También puedes exportar el script y ejecutarlo manualmente.

8) Verificación y pruebas

Después de aplicar las reglas:

- Comprueba conectividad (ping, navegación, SSH...)
- Revisa logs si algo no funciona
- Ajusta reglas si es necesario

Resumen rápido

El flujo básico sería:

- 1) Crear firewall
- 2) Definir interfaces
- 3) Crear objetos
- 4) Definir reglas
- 5) Configurar NAT
- 6) Compilar
- 7) Instalar
- 8) Probar

Vamos a ver el tema con más detalle en el video 136 (**Obligatorio**).

🌐 **Firewalls - (Parte 15: Interfaz gráfica, primeros pasos) - Charla 136 - Aprendiendo Ciberseguridad paso a paso**

<https://youtu.be/f3oHKzg2OKE>



5.16 Firewalls – FWBuilder (Objetos, redes, reglas y compilación)

En **Firewall Builder**, los conceptos de **objetos**, **redes** y **compilación** son fundamentales porque definen cómo se construyen y aplican las reglas del firewall de forma organizada y reutilizable. Entenderlos bien es clave para trabajar con soltura.

Objetos en Firewall Builder

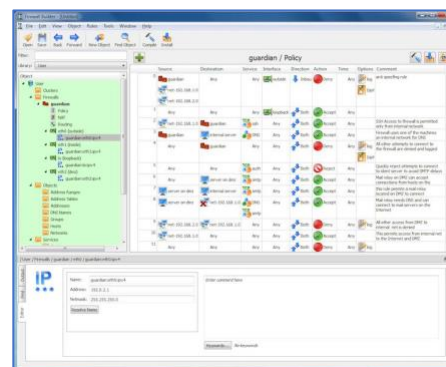
Los **objetos** son la base de toda la configuración. Representan elementos reales de la red y permiten abstraer la complejidad.

Tipos de objetos más comunes:

- **Hosts** → Un equipo concreto (ej: servidor web)
- **Networks (redes)** → Subredes completas (ej: 192.168.1.0/24)
- **Servicios** → Protocolos y puertos (HTTP, HTTPS, SSH...)
- **Grupos** → Conjuntos de objetos (ej: “servidores internos”)

Ventajas:

- Reutilización (defines una vez, usas muchas)
- Mayor claridad en las reglas

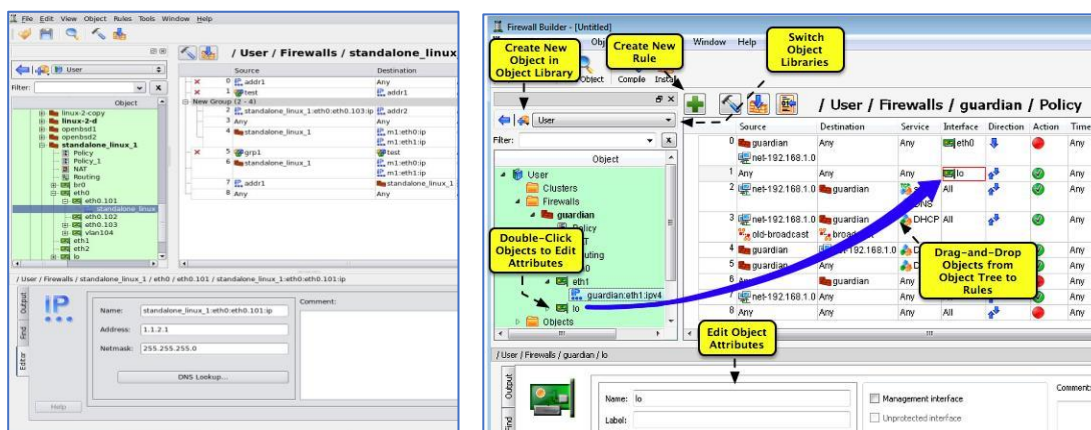


- Facilitan cambios globales (modificas el objeto → afecta a todas las reglas)

Ejemplo: si cambias la IP de un servidor, solo editas el objeto, no todas las reglas.

Redes en Firewall Builder

Las **redes** son un tipo específico de objeto, pero tienen un papel especialmente importante porque estructuran el tráfico.



Qué representan:

- Subredes completas (ej: LAN, DMZ, WAN)
- Rangos de direcciones IP

Uso en reglas:

Se utilizan para definir:

- Origen del tráfico (ej: red interna)
- Destino (ej: Internet o DMZ)

Ejemplo típico:

- LAN → 192.168.1.0/24
- WAN → Internet
- DMZ → 192.168.2.0/24

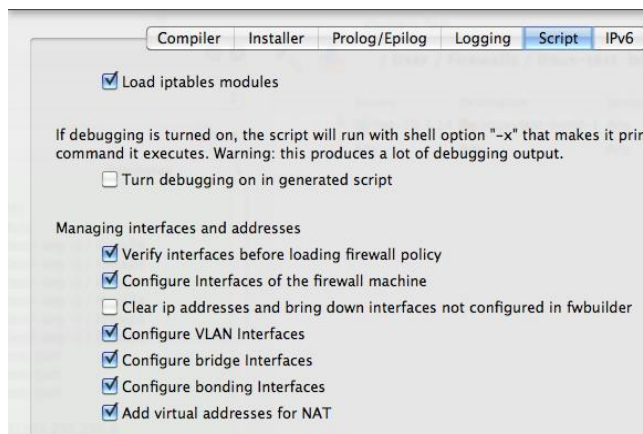
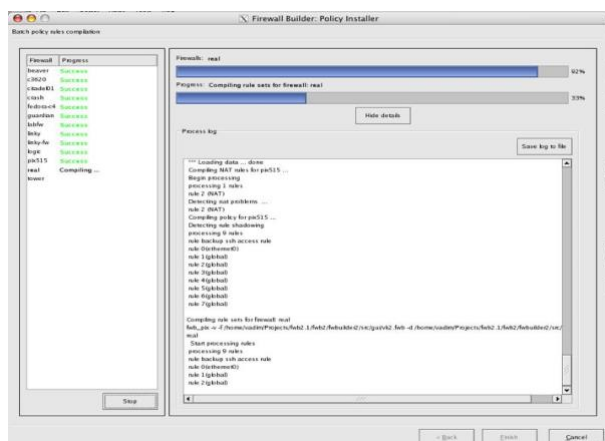
Esto permite crear reglas como: “Permitir tráfico HTTP desde LAN hacia Internet”

Compilación en Firewall Builder

La **compilación** es el proceso que transforma la configuración visual en reglas reales del firewall.

Qué hace exactamente:

- Convierte objetos y reglas en comandos específicos (ej: *iptables*)
- Optimiza el orden de las reglas
- Verifica errores o inconsistencias



Resultado:

Genera un **script listo para ejecutar**, por ejemplo:

`iptables -A INPUT -p tcp --dport 22 -j ACCEPT`

Ventajas:

- Evita escribir reglas manualmente
- Reduce errores de sintaxis
- Permite validar antes de aplicar

Fases:

- 1) Validación de objetos
- 2) Traducción de reglas
- 3) Generación del script final

Relación entre los tres conceptos

Estos tres elementos trabajan juntos:

- **Objetos** → Definen los elementos (IP, redes, servicios)
- **Redes** → Organizan el tráfico dentro de esos objetos
- **Compilación** → Convierte todo en reglas reales ejecutables

Flujo típico:

- 1) Creas objetos (red interna, servidor, servicio HTTP)
- 2) Usas esos objetos en reglas
- 3) Compilas → obtienes configuración real del firewall

Conclusión

Firewall Builder simplifica enormemente la gestión de firewalls gracias a este enfoque:

- Los **objetos** aportan modularidad y reutilización
- Las **redes** estructuran la lógica del tráfico
- La **compilación** convierte todo en reglas reales sin errores

Este modelo permite trabajar de forma mucho más clara, segura y escalable que editando directamente reglas de bajo nivel como en *iptables*.

Vamos a ver el tema con más detalle en el video 137 (**Obligatorio**).

 **Firewalls - (Parte 16: Objetos, redes, reglas) - Charla 137** - Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/L9Go0shInd4>



5.17 Firewalls – Instalación de un FW, con FWBuilder

El proceso de **instalar (o “inyectar”) la configuración en un firewall remoto** con Firewall Builder es uno de sus puntos fuertes, porque automatiza lo que normalmente harías a mano con scripts. Básicamente, pasa de tu diseño visual a un sistema real en producción de forma controlada.

Te explico el flujo completo de forma clara:

1) Preparación del firewall remoto

Antes de instalar nada, el equipo destino debe estar listo:

- Tener acceso por **SSH**
- Tener instalado el sistema de firewall correspondiente (ej: *iptables*)
- Tener permisos de administrador (root o sudo)

Firewall Builder se conectará remotamente a este equipo.

2) Configurar parámetros de instalación en Firewall Builder

Dentro del propio objeto del firewall (en la GUI):

- Defines:
 - **Dirección IP o nombre del host**
 - **Usuario SSH**
 - Método de autenticación (contraseña o clave pública)
 - Ruta donde se copiará el script

También puedes configurar:

- Si quieres hacer copia de seguridad de reglas anteriores
- Cómo se aplicarán (reinicio completo o incremental)

3) Compilar la configuración

Antes de instalar:

- Pulsas **Compile**
- Firewall Builder genera el script final (por ejemplo, reglas *iptables*)

Este script es el que realmente se ejecutará en el firewall remoto.

4) Proceso de instalación (Install)

Cuando pulsas **Install**, ocurre lo siguiente automáticamente:

Paso a paso interno:

- a **Conexión SSH** al firewall remoto
- b **Transferencia del script** generado
- c (Opcional) Backup de configuración previa
- d **Ejecución del script** en el sistema
- e Aplicación de las nuevas reglas

Todo esto sin que tengas que conectarte manualmente.

5) Mecanismo de seguridad (rollback)

Firewall Builder incluye una protección muy importante:

- Puede activar un **temporizador de rollback**
- Si pierdes la conexión (por ejemplo, te bloqueas a ti mismo), el sistema:
 - Revierte automáticamente a la configuración anterior

Esto evita dejar el firewall inaccesible por error.

6) Verificación tras la instalación

Después de aplicar las reglas:

- Comprueba acceso SSH
- Verifica servicios (web, ping, etc.)
- Revisa logs si algo falla

Es recomendable probar siempre antes de dar por válida la configuración.

Alternativa: instalación manual

Si no quieres usar despliegue automático:

- a Compilas la configuración
- b Exportas el script
- c Lo copias manualmente (scp, USB, etc.)
- d Lo ejecutas en el firewall

Útil en entornos restringidos o sin acceso SSH directo.

Resumen del flujo completo

- 1) Diseñas reglas en GUI
- 2) Compilas → generas script
- 3) Configuras acceso remoto

- 4) Install → conexión SSH
- 5) Se copia y ejecuta automáticamente
- 6) Verificas funcionamiento

Idea clave

Firewall Builder actúa como un **orquestador**:

- Tú defines *qué quieres* (reglas)
- Él se encarga de *cómo aplicarlo* (script + despliegue)

Vamos a ver el tema con más detalle en el video 138 **(Obligatorio)**.

 **Firewalls - (Parte 17: Instalación de un FW, con FWBuilder)**
- **Charla 138** - Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/tB2m-wnxJZI>



5.18 Firewalls – nftables

¡¡Cerramos Firewalls!! con un capítulo de lujo "**nftables**" que es la evolución natural de "iptables". Desarrollamos la última parte de nuestro artículo: "**Teoria_y_practica_con_Firewalls.pdf**". Presentaremos la instalación de nftables, su configuración y algunos comandos básicos para la gestión de reglas. Como veremos que es muy similar a iptables, solo haremos un par de ejercicios, dejándote todos los enlaces y guías para que puedas seguir avanzando todo lo que desees.

¿Qué es "nftables"

nftables es el sistema moderno de filtrado de paquetes en Linux que sustituye progresivamente a herramientas clásicas como *iptables*, *ip6tables*, *arptables* y *ebtables*. Introducido en el kernel a partir de Linux 3.13, unifica todas esas funcionalidades en una única infraestructura más eficiente, flexible y coherente.

Se compone de dos partes principales:

- **El subsistema del kernel** (encargado de procesar paquetes)
- **La herramienta de usuario (nft)**, con la que defines las reglas

Características principales

1) Modelo unificado

A diferencia de *iptables*, que usa varias herramientas distintas, *nftables* gestiona:

- IPv4
- IPv6
- ARP
- Bridge

Todo desde una sola sintaxis.

2) Sintaxis más clara y moderna

Ejemplo sencillo:

```
nft add rule inet filtro input tcp dport 22 accept
```

Más legible y consistente que en *iptables*, y sin duplicar reglas para IPv4/IPv6.

3) Estructura jerárquica

nftables también se basa en:

- **Tablas**
- **Cadenas (chains)**
- **Reglas**

Pero con mayor flexibilidad:

- Puedes crear tus propias cadenas más fácilmente
- Permite estructuras más complejas sin “trucos”

4) Uso de conjuntos (sets) y mapas

Una de sus grandes mejoras:

```
nft add set inet filtro ips_permitidas { type ipv4_addr\; }
```

Permite agrupar múltiples IPs o puertos en una sola regla:

Más eficiente y más fácil de mantener.

5) Mayor rendimiento

- Reduce duplicación de reglas
- Menos llamadas al kernel
- Evaluación más eficiente del tráfico

Ideal para sistemas con muchas reglas.

6) Actualizaciones atómicas

Cuando aplicas cambios:

- Se cargan de forma **atómica** (todo o nada)
- Evita estados intermedios peligrosos

Muy importante en entornos críticos.

Ventajas frente a iptables

- ✓ Sintaxis más limpia
- ✓ Menos redundancia (IPv4/IPv6 juntos)
- ✓ Mejor rendimiento
- ✓ Configuraciones más escalables
- ✓ Gestión avanzada con sets y mapas

Inconvenientes o retos

- Curva de aprendizaje inicial (cambia bastante respecto a iptables)
- Algunas herramientas antiguas aún dependen de iptables
- Migración puede requerir adaptación

Relación con iptables

Hoy en día existen dos escenarios:

- Sistemas antiguos → usan *iptables*
- Sistemas modernos → usan *nftables* directamente o mediante compatibilidad (*iptables-nft*)

En muchas distribuciones actuales, *iptables* es solo una capa sobre *nftables*.

Ejemplo básico completo

```
nft add table inet filtro
```

```
nft add chain inet filtro input { type filter hook input priority 0 \; policy drop \; }
```

```
nft add rule inet filtro input ct state established,related accept
```

```
nft add rule inet filtro input tcp dport 22 accept
```

Esto:

- Crea una tabla
- Define una política restrictiva
- Permite conexiones ya establecidas y SSH

Conclusión

nftables representa la evolución natural del filtrado de paquetes en Linux:

- Más potente
- Más flexible
- Más eficiente

Es la opción recomendada hoy en día para nuevos despliegues, especialmente en entornos donde la seguridad, el rendimiento y la mantenibilidad son críticos.

Vamos a ver el tema con más detalle en el video 139 **(Obligatorio)**.

 **Firewalls (Fin) - (Parte 18:"nftables") - Charla 139 -**
Aprendiendo Ciberseguridad paso a paso

<https://youtu.be/NyjoEeQIoUQ>

