

DESARROLLO DE UN FIREWALL CON UNA ARQUITECTURA DE BAJO
COSTO PARA SISTEMAS DE MONITOREO Y CONTROL EN REDES
INDUSTRIALES

NAIRNE OVALLES HERNÁNDEZ
HENRY JIMÉNEZ ROSERO

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS TECNOLOGÍA E INGENIERÍA.
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
SANTIAGO DE CALI.
2018

DESARROLLO DE UN FIREWALL CON UNA ARQUITECTURA DE BAJO
COSTO PARA SISTEMAS DE MONITOREO Y CONTROL EN REDES
INDUSTRIALES

NAIRNE OVALLES HERNÁNDEZ.
HENRY JIMÉNEZ ROSERO

Monografía

Ingeniero: John Freddy Quintero Tamayo
Director

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS TECNOLOGÍA E INGENIERÍA.
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
SANTIAGO DE CALI.
2018

Nota de Aceptación

Presidente del Jurado

Jurado

Jurado

DEDICATORIA

Este proyecto se lo dedicamos a nuestro Padre Dios por toda la sabiduría que nos da a diario y su fortaleza para seguir sin desvanecer.

A nuestra hija porque es el regalo más hermoso que Dios nos ha dado, convirtiéndose en el motor de nuestra vida. También a nuestros padres por su gran apoyo, comprensión y amor.

CONTENIDO

	Pág.
INTRODUCCIÓN	12
1. FORMULACIÓN DEL PROBLEMA.....	13
2. JUSTIFICACIÓN DEL PROBLEMA	15
3. OBJETIVOS.....	16
3.1 OBJETIVO GENERAL	16
3.2 OBJETIVOS ESPECÍFICOS.....	16
4. MARCO DE REFERENCIA.....	17
4.1. MARCO TEORICO	17
4.1.1. FIREWALLS.....	17
4.1.2. DISEÑO DE FIREWALLS.....	17
4.1.3. TOPOLOGÍAS LÓGICAS DE UN FIREWALL	19
4.1.3.1. FIREWALL DE FILTRADO DE PAQUETES.....	19
4.1.3.2. DUAL-HOMED HOST.....	19
4.1.3.3. SCREENED HOST.....	20
4.1.3.4. SCREENED SUBNET (DMZ):	20
4.1.4. FILTRADO DE PAQUETES.....	21
4.1.5. REDES INDUSTRIALES Y PROTOCOLOS	22
4.1.6. PROTOCOLO MODBUS	23
4.1.7. MODOS DE TRANSMISIÓN	24
4.1.7.1. MODO DE TRANSMISIÓN ASCII	24
4.1.7.2. MODO DE TRANSMISIÓN RTU	24
4.1.7.3. MODO DE TRANSMISION MODBUS TCP.....	25
4.1.8. FIREWAL INDUSTRIAL.....	25
4.2. MARCO CONCEPTUAL.....	28
4.3. MARCO LEGAL	31
5. DESARROLLO FASES DEL PROYECTO.....	36
6. DESARROLLO DEL FIREWALL.....	39
6.1 IPTABLES.....	39
6.2 PYPTABLES.....	47
6.3 PYMODBUS.....	52

6.4 PLATAFORMA Y SISTEMA OPERATIVO.....	58
6.5 IMPLEMENTACIÓN DEL FIREWALL INDUSTRIAL.....	66
VALIDACION DEL FIREWALL.....	88
BUILD ROOTS.....	99
ANALISIS DE RESULTADOS.....	101
CONCLUSIONES.....	102
RECOMENDACIONES.....	104
DIVULGACION.....	105
BIBLIOGRAFIA.....	106
ANEXOS.....	112
Anexo A. Resumen Analítico RAE.....	112

LISTA DE TABLAS

	Pág.
Tabla I. Sistemas operativos soportados por Rpi.....	57
Tabla II. Características de la Raspberry.....	58
Tabla III. Desempeño de procesadores de equipos de escritorio.....	60
Tabla IV. Paquetes requeridos para la ejecución de buildroots.....	87

LISTA DE FIGURAS

	Pág.
Figura 1. Viaje de un paquete al interior del kernel de Linux.	40
Figura 2. Política de iptables, Raspbian jessie lts	43
Figura 3. Servidor Modbus en PYMODBUS	52
Figura 4. Servidor Modbus asíncrono en ejecución.	55
Figura 5. Cliente Modbus	55
Figura 6. GPIO de la raspberry pi	61
Figura 7. Raspberry pi	62
Figura 8. Desempeño de las diferentes versiones de las Raspberry pi.	62
Figura 9. Topología 1. Firewall Industrial.	66
Figura 10. Topología 2. Firewall Industrial.	67
Figura 11. Diagrama de interconexiones del software al interior de la Rpi.	68
Figura 12. Comando lsusb, detección de la tarjeta wlan0.	69
Figura 13. Configuración del servidor DHCP.	71
Figura 14. Resultado de los comandos de modificación archivo	72
Figura 15. Resultado de los comandos de modificación archivo	73
Figura 16. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.	74
Figura 17. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.	75
Figura 18. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.	76
Figura 19. Resultado de los comandos de modificación del archivo sysctl.conf en la configuración del reenvío de paquetes de la red modbus hacia el router de internet.	77
Figura 20. Resultado de los comandos de activación de NAT y de Persistencia de las reglas iptables.	78
Figura 21. Resultado de los comandos de activación de servicios	79

Figura 22. Evidencia del funcionamiento del AP ModbusF	80
Figura 23. simply modbus client en conexión con servidor pymodbus Rpi	81
Figura 24. El cliente lee registros del servidor pymodbus en Rpi.	82
Figura 25. Escritura de un byte desde el cliente y hacia el servidor	83
Figura 26. Dato modificado por el cliente.	84
Figura 27. Código de implementación de firewall por filtrado de puerto y protocolo.	85
Figura 28. Operación de servidor con firewall que acepta todo el tráfico tcp Por el puerto 502.	86
Figura 29. Operación de servidor con firewall que impide el paso de Paquetes por el puerto 502 del protocolo modbus TCP.	85
Figura 30. Mensaje en el cliente cuando la conexión con el servidor Es bloqueada por el firewall.	88
Figura 31. Arquitectura de la Red con los Ips de las interfaces de red.	89
Figura 32. Inicio de servidor Nessus Fuente: Los autores.	90
Figura 33. Conjunto de análisis disponibles en Nessus.	91
Figura 34. Escaneo Básico.	92
Figura 35. Topología de red para el análisis 1, desde el exterior de la red	93
Figura 36. Resultado del Análisis 1 desde el exterior con el firewall Desactivado	94
Figura 37. Resultado del Análisis 1, desde el exterior con el firewall activo.	95
Figura 38. Topología de red para el análisis 2, desde el interior de la red.	96
Figura 39. Resultado para el análisis 2, desde el interior de la red, Con firewall abierto.	97
Figura 40. Resultado para el análisis 2, desde el interior de la red, Con firewall cerrado.	98
Figura 41. Creación imagen del sistema operativo	99

LISTA DE ANEXOS

Pág.

Anexo A. Resumen Analítico RAE **¡Error! Marcador no definido.**

RESUMEN

Los sistemas de control modernos son cada vez más complejos, digitales y conectados a la red. En el pasado estos sistemas de control industrial permanecieron aislados de otras redes, sin embargo, para los operadores actuales resulta muy importante que la información sea transferida entre el proceso de producción y el nivel de administración o entre plantas y el nivel de gerencia. De esta forma se ha creado un espacio potencial para que los piratas informáticos busquen acceder y alterar los sistemas de control en tiempo real y la infraestructura dependiente. Este desarrollo muestra los aspectos necesarios para construir y validar un firewall orientado a redes industriales. El desarrollo utiliza hardware y software de libre distribución y se valida sobre una microred en protocolo modbus.

INTRODUCCIÓN

Los sistemas Scada es un sistema que permite monitorear, tener control y obtener datos los cuales ayudan a la toma decisiones, toda esta información se puede acceder de manera remota. Por consiguiente, este sistema está conectado a la red corporativa y de ésta red a internet siendo un gran progreso en la automatización industrial debido a que permiten tener el control en tiempo real de los dispositivos, generar alarmas y mejorar procesos.

La seguridad de este tipo de sistemas es de gran importancia debido a que cualquier amenaza y ataque genera en la industria grandes pérdidas, tiempo de recuperación y capacidad para producir. Por todo lo anterior, se ha incrementado la preocupación por la seguridad de estos sistemas.

Por lo tanto, como solución en este proyecto se desarrollará un firewall industrial con hardware y software de libre distribución, el cual permitirá disminuir los riesgos, vulnerabilidades y amenazas que se puedan presentar.

1. FORMULACIÓN DEL PROBLEMA

La gran mayoría de redes industriales actuales se interconectan de forma directa a los sistemas administrativos empresariales empleando sistemas de seguridad convencionales no especializados, dejando expuesta información crítica del proceso involucrado la cual puede ser modificada o monitoreada desde el interior o exterior de la empresa por personal no autorizado, estos síntomas pueden ser debidos a múltiples causas, las cuales pueden abarcar desde políticas propias, pasando por implementaciones incorrectas y llegando en algunos casos hasta el desconocimiento del tema.

Las implementaciones incorrectas se relacionan con el desconocimiento de los diferentes propósitos de un sistema Scada y un sistema de información convencional. El primero pensado en la confiabilidad, respuesta en tiempo real, tolerancia a situaciones de emergencia, seguridad del personal, calidad del producto, seguridad física de la planta y el personal. El segundo enfocado con la provisión de conectividad interna y externa, el manejo de la productividad, manejo de bases de datos y con la implementación de extensos mecanismos de seguridad para la autenticación y autorización. (Kruz Ronald 2004. Cap1, pag 16).

Se puede llegar a predecir que los problemas de seguridad en redes industriales se deben a interconexiones incorrectas entre los dos sistemas generalmente ocasionados por la falta de uso de aplicaciones especializadas.

Dentro de las mejores prácticas para solucionar esta problemática se parte desde la implementación de políticas empresariales, registro de actividades, biometría para el acceso, sistemas de detección de intrusos, criptografía e implementación de firewalls, los cuales pueden proteger a la red Scada de una posible intrusión sobre la red convencional empresarial.

Entonces el presente anteproyecto pretende dar respuesta a esta problemática mediante el desarrollo de un filtrado de paquetes tipo Iptables en una arquitectura de bajo costo Raspberry PI para una red Industrial SCADA y da respuesta a las siguientes preguntas:

Cuáles son las principales consideraciones de un firewall embebido en una red industrial y sistema scada?

Cuál es la arquitectura de un firewall industrial, con respecto a paquetes de datos especializados que garantice el aislamiento de redes?

2. JUSTIFICACIÓN DEL PROBLEMA

El presente proyecto consiste en el desarrollo de un filtrado de paquetes que permita la detección de paquetes propios de una red industrial y que a su vez aisle el tráfico no caracterizado.

De la misma forma resulta de especial interés la implementación sobre una plataforma de hardware libre como la raspberry pi y software open source como Linux y su estructura de iptables. Lo cual garantiza la reducción en los costos de implementación y pruebas para de esta forma competir con los costosos productos existentes en el mercado.

Se espera que en el corto plazo el proyecto permita apropiar el conocimiento requerido para su desarrollo demostrando su funcionalidad y que en el mediano y largo plazo permita expandir la investigación en el área.

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Desarrollar un filtrado de paquetes empleando Iptables en una arquitectura de bajo costo Raspberry PI para sistemas SCADA en redes industriales

3.2 OBJETIVOS ESPECÍFICOS

- Estado del arte orientado a firewalls de redes industriales y Raspberry pi.
- Desarrollar una aplicación de software libre que permita configurar las características de Iptables para ser usado como un firewall en una red industrial.
- Realizar pruebas de seguridad con shodan antes y después de implementar el firewall.
- Realizar un evento de divulgación para dar a conocer los resultados de la monografía.

4. MARCO DE REFERENCIA

4.1.MARCO TEORICO

4.1.1. FIREWALLS

Un firewall es un sistema que se usa para aislar una maquina o subred, creando un perímetro de seguridad que pertenece a la empresa y permite protegerla de redes externas como servicios y protocolos que desde afuera puedan ser una amenaza para la seguridad de la organización.

La mejor política de seguridad es el aislamiento total, pero hoy en día las organizaciones requieren compartir información con personas externas por lo tanto esta opción no es viable. Por esta razón se hace necesario tener un firewall que permita la separación entre la zona de riesgo y el perímetro de seguridad.

4.1.2. DISEÑO DE FIREWALLS

El diseño o configuración de un firewall se puede realizar de varias formas, la primera está relacionada con la política de seguridad que tenga definida la organización la cual puede ser el bloqueo del tráfico externo que va hacia la red corporativa, excluyendo las consultas realizadas sobre la página web. La otra forma podría ser el bloqueo de todos los servicios de salida que van al exterior menos el correo electrónico.

Después de saber que política de seguridad se va a utilizar se debe definir cómo será la implementación es decir lo que será permitido y lo que no. Por lo tanto, se puede realizar de dos maneras denegando todo lo que explícitamente no se permita o permitiendo todo menos lo que se ha denegado. La más adecuada respecto a la seguridad es la primera opción.

Por otro lado, los gastos varían dependiendo de la red que se va a proteger, es decir un área en donde exista poca cantidad de equipos se puede proteger usando como firewall un pc con Linux. Pero si por el contrario es una red de gran tamaño se debería utilizar sistemas propietarios que por lo general son de alto costo. Igualmente, no solo se debe pensar en el gasto e implementación de un firewall sino también en el precio de su mantenimiento.¹

Otro punto que se debe tener en cuenta es la ubicación del firewall para que la cobertura de la red sea total. En caso de que el firewall quede en un punto que no proteja toda la red se debe adicionar un firewall interno dentro de la misma para tener así mayor seguridad.

Por otro lado, se debe tener uno o varios elementos físicos como bastión, teniendo en cuenta que entre menos servicios ofrezca, su mantenimiento será más fácil y de esta manera mayor su seguridad. Es muy importante que el bastión este asegurado para que el firewall opere correctamente, debido a que es el elemento más accesible de la red y debe soportar los ataques que le realicen a esta. En el caso de que la seguridad se vea afectada, estarían en amenaza todos los equipos que estén en el perímetro de seguridad. Una opción muy conveniente sería utilizar un servidor en el que este corriendo Unix debido a la alta seguridad del sistema operativo y también gran parte de las aplicaciones firewall han sido desarrolladas y probadas desde hace varios años sobre este sistema operativo.

¹ SEGURIDAD EN UNIX Y REDES, 2002. Disponible en: <https://www.rediris.es/cert/doc/unixsec/node23.html>

4.1.3. TOPOLOGÍAS LÓGICAS DE UN FIREWALL

4.1.3.1. FIREWALL DE FILTRADO DE PAQUETES

Los firewall simples emplean técnicas de filtrado de paquetes en principio son implementados aprovechando la arquitectura de algunos routers para realizar un enrutamiento optativo, de esta forma puede permitir el bloqueo o paso de paquetes, las cuales se apoyan en funciones programadas aprovechando características propias de las tramas de datos. En el filtrado de paquetes se hace a través de la direcciones origen y destino (TCP, UDP) al igual que el protocolo y puertos origen y destino. También se realiza por el tipo de mensaje (ICMP) e interfaces I/O de la trama del router.

En el firewall de filtrado la información que va hacia el exterior y no se encuentre bloqueada es directa, no se requiere proxies como los que usan un pc con dos tarjetas de red, esta arquitectura es la más simple y la más utilizada en las empresas que no tienen la necesidad de altos niveles de seguridad. Uno de sus inconvenientes es que no tienen sistema de monitoreo y el administrador no se puede dar cuenta de un ataque. Otro de sus inconvenientes es la difícil implementación de reglas de filtrado.²

4.1.3.2. DUAL-HOMED HOST

Otro tipo de firewall muy conocido es el que emplea servidores con tarjetas de red múltiples, de esta forma cada tarjeta hace referencia a una red y el núcleo iptables del Linux se utiliza para implementar no solo reglas de filtrado sino políticas completas.

² SEGURIDAD EN UNIX Y REDES, 2002. Disponible en: <https://www.rediris.es/cert/doc/unixsec/node23.html>

Se recomienda adicionar un filtrado de direcciones mediante proxy en cada tarjeta de red y desactivar el IP Forwarding, eliminando su capacidad de ruteo de paquetes. De esta manera el sistema externo ve el host por una tarjeta y los internos por la otra, en ambas todo el tráfico debe pasar por el firewall. Esta técnica tiene como desventaja que un usuario que acceda a la maquina e incremente su nivel de privilegios puede modificar toda la operación de la misma y por lo tanto eliminar la capacidad operativa del firewall.

4.1.3.3. SCREENED HOST

En este tipo de topología se caracteriza por emplear un enrutador con un servidor de vigilancia, entonces en el primero se ejecuta el filtrado de paquetes junto con su políticas establecidas mientras que en el último se implementan proxies, además de un filtrado de paquetes para la red interna, de esta forma todo paquete que entra al sistema debe pasar por el router principal y luego debe pasar por los proxies para luego ser enrutado, el enrutamiento al interior de la maquina debe sobrepasar las políticas de filtrado propias del sistema operativo residente para de esta forma no afectar a la red interna.

4.1.3.4. SCREENED SUBNET (DMZ):

La topología Screened Subnet, también referenciada como red perimétrica permite hacer un control robusto de la seguridad ya que agrega una zona desmilitarizada como camino obligatorio entre las redes externa e interna, la cual busca reducir de forma drástica los efectos alcanzados tras un ataque con éxito al host que soporta el cortafuegos. Como el servidor cortafuegos es un objetivo interesante para un ataque, la arquitectura DMZ intenta aislarlo en

una red perimétrica para que el atacante al ingresar no pueda tener acceso total sobre la subred protegida.³

La operación de este esquema de cortafuegos hace uso de dos enrutadores, el primero se ubica entre la red perimétrica y la red externa, el segundo se ubica entre la red interna y la red perimétrica. Al interior de esta red perimétrica se coloca el servidor cortafuegos, los proxies y otros servicios que requieran autenticación. Entonces cuando un paquete sale debe pasar por el enrutador dos luego deber ser validado en el proxy, luego ser filtrado por el servidor cortafuegos y por ultimo superar el enrutador 1.

4.1.4. FILTRADO DE PAQUETES

El filtrado de paquetes es uno de dos tipos de firewall existentes, el segundo es el firewall proxy. El primero solo va hasta el tercer nivel de la capa OSI que es donde se encuentra los paquetes, tomando decisiones de acuerdo a la información que existe en la cabecera del paquete IP. El segundo va hasta el nivel de aplicación debido a que sus decisiones se basan en los datos que hay en este nivel. La información que va por la red IP se encuentra dividida en varias partes llamadas paquetes, las cuales se envían de forma separada.

La función de un enrutador es definir la ruta que tendrá cada paquete que le llega de acuerdo a su destino final. Por lo tanto, un paquete solo lleva la IP destino para de esta forma facilitarle al router definir su ruta. El paquete dice el destino pero no el camino por donde debe ser enviado.

Los routers manejan entre ellos comunicación y lo hacen a través de protocolos de enrutamiento. Existen varios entre ellos (RIP) caracterizado por estar entre los

³ SEGURIDAD EN UNIX Y REDES, 2002. Disponible en: <https://www.rediris.es/cert/doc/unixsec/node23.html>

más sencillos. Otro es el protocolo (OSPF) que es para la construcción de tablas de ruta, con las cuales se define como enviar hasta su destino cada paquete. En el momento que le llega un paquete al router este se fija en la dirección destino para enrutarlo, pero también valida si se debe enviar o no el paquete de acuerdo a las políticas de seguridad que tenga configuradas. ⁴

4.1.5. REDES INDUSTRIALES Y PROTOCOLOS

Una red industrial está conformada por estaciones de transmisión y recepción de datos, Se resalta que los datos que en estas redes se transportan corresponden a variables de proceso o a órdenes de actuación sobre motores o controladores que generalmente evolucionan en texto plano. Además estas redes implementan las topologías de cualquier red de datos, algunas veces los componentes son referidos como nodos los cuales pueden pertenecer a un anillo serie (Al estilo de token ring) o dispositivos interconectados entre ellos como en una red de árbol (Como ethernet), La elección de la topología depende de la capa física seleccionada y del protocolo que sobre ellas se ejecuta. Estas redes industriales están compuestas por diferentes equipos:

PC's Industriales.

Controladores

Sistemas de Control Distribuido.

Transductores y Actuadores.

Módulos Inteligentes.

Interfaces de Operador

Las redes industriales usan protocolos que permiten intercambiar datos entre dispositivos, Uno de los protocolos más utilizado es modbus. ⁵

⁴ Diseño e implementación de un firewall, 2002. Disponible en:
<http://users.salleurl.edu/~is06200/proyectos/TFC-is06200.pdf>

4.1.6 PROTOCOLO MODBUS

Fue desarrollado por Modicon con el objetivo de que su aplicación sea estándar en el área industrial. Entre sus ventajas esta que existe una comunicación transparente entre los dispositivos que utilicen el protocolo.

Debido a que Modbus es una red digital usa del modelo Osi el nivel físico, de enlace de aplicación. Utiliza como topología Maestro Esclavo de manera lineal, por lo tanto solo hay un maestro, que es el encargado tener el control sobre el acceso y monitorea la operación de la red. El resto de dispositivos son esclavos los cuales se comportan de acuerdo a lo solicitado por el maestro.

Por otro lado, su comunicación es serial asíncrona y utiliza los estándares como RS-232 ó RS-485 y RS-422, los medios físicos usados son cable, radio frecuencia, fibra óptica y su velocidad de transmisión va de 75 a 19.200 baudios.

En un comienzo fue usado por la compañía Modicon para la comunicación entre PLCs. Posteriormente se siguió utilizando teniendo varias actualizaciones convirtiéndolo en estándar clave debido a que facilita a la industria en su automatización. Esto lo logra gracias a la estructura que maneja en los mensajes y su utilización de direcciones de memoria permitiéndole su adaptación a varios dispositivos.

Este protocolo contiene información de cómo se puede acceder a la información contenida en un dispositivo, la manera como éste debe responder y cuáles son los posibles mensajes de error. Todo esto con el fin de que haya un adecuado flujo de datos entre todos los dispositivos que hacen parte de la red.⁶

⁵ Software para aplicaciones industriales. Disponible en:
<ftp://ftp.unicauca.edu.co/Facultades/FIET/DEIC/Materias/SW%20para%20aplicaciones%20Industriales%20II/Sw%20II/Conferencias/Capitulo%205.pdf>

⁶ Implementación del protocolo en microcontrolador, 2006 Disponible en:
<http://repositorio.uis.edu.co/jspui/bitstream/123456789/3176/2/119444.pdf>

4.1.7 MODOS DE TRANSMISIÓN

El protocolo Modbus utiliza para el intercambio de los mensajes tres modos de transmisión: el ASCII, el RTU y el modo TCP.

4.1.7.1 MODO DE TRANSMISIÓN ASCII

En este modo todos los mensajes empiezan con el carácter dos puntos (:) y finaliza con CR/LF que es retorno de carro/ avance de línea. Por lo tanto, los dispositivos que están en la red monitorean el bus hasta que encuentren caracteres Ascii válidos y en el momento que encuentran el carácter dos puntos se dan cuenta que uno de los dispositivos ha comenzado a transmitir un mensaje y posteriormente analizan el campo siguiente que es la dirección destino del mensaje. En el caso que ésta sea la dirección de uno de los esclavos, éste empieza a recibir la trama y a realizar la acción indicada.

Entre sus ventajas está el tiempo que dura entre caracteres que va hasta un segundo y el dispositivo receptor no lo ve como error de comunicación.

4.1.7.2 MODO DE TRANSMISIÓN RTU

En este modo los mensajes empiezan con un silencio de 3.5 s el tiempo para enviar un carácter, luego de este silencio se envía la dirección del esclavo que tiene una longitud de 8 bits. También tiene 8 bits para el código de función, para los datos usa múltiplos enteros de 8 bits y 16 bits para el CRC. Por consiguiente, la trama finaliza con un silencio como mínimo 3.5 veces el tiempo para mandar un carácter.

4.1.7.3 MODO DE TRANSMISION MODBUS TCP

Este modo de operación fue diseñado para emplear las redes Ethernet existentes, sin la necesidad de implementar una capa física de hardware diferente. Básicamente el modo tcp de modbus, lo que realiza es manejar la misma información de modbus RTU pero al interior de tramas TCP. De esta forma al momento de la comunicación, el elemento cliente, que es el esclavo modbus, crea una sesión de comunicación con el esclavo, que es el servidor modbus. Luego se procede al envío de paquetes según la operación básica de tcp. Entonces no se garantiza el camino de envío de los paquetes pero si la llegada de los mismos.⁷

4.1.8 FIREWAL INDUSTRIAL

Es un dispositivo que puede ser hw o sw y su función es el monitoreo y control del tráfico que existe entre dos redes, detectando el tráfico no permitido y lo compara con reglas predefinidas de acuerdo al nivel en que este (paquete, trama, segmento)

Un firewall se categoriza como industrial por:

- El diseño que es para entornos ambientales y redes industriales.
- Su instalación y despliegue no es intrusiva ni invasiva.
- Es de fácil uso la configuración y módulos de gestión de reglas.
- Aumenta la seguridad de la red usando funcionalidades específicas.

De acuerdo a su funcionamiento existen varios tipos de firewalls:

⁷ Implementación del protocolo en microcontrolador, 2006 Disponible en:
<http://repositorio.uis.edu.co/jspui/bitstream/123456789/3176/2/119444.pdf>

Los firewall que operan a nivel de red como los Packet Filter Firewall, debido a que realizan reglas básicas sin tener en cuenta la relación entre los paquetes y Stateful Firewalls.

Otro firewall son los que operan en la capa de aplicación en el modelo Osi, llamándose Application Firewall o Proxy Firewalls, su nivel de análisis es a más alto nivel teniendo en cuenta los parámetros de cada aplicación. Algunos protocolos en los que tienen reglas de segmentación son HTTP, SMTP, Telnet, FTP.

El firewall DPI siendo el más avanzado debido a que puede filtrar por protocolos/tipos de archivos particulares como es SOAP o XML. Este tipo de firewall se usa a nivel físico en los sistemas Scada, HMI y también en los dispositivos PLCs, DCSs, RTUs.

Permite definir reglas de segmentación por cada protocolo industrial (Ethernet/IP, Modbus entre otros) bloqueando todo tráfico no permitido.⁸

ESTADO DEL ARTE DE FIREWALLS INDUSTRIALES

Una búsqueda en Scopus de investigaciones recientes relacionadas con las palabras: Industrial Firewall, muestra 295 resultados, con publicaciones iniciadas desde el año 1996, La evolución de estudios en esta área sufrió un pico importante en 2010 declinando hasta 2016 donde el tema volvió a mantener relevancia hasta 2018.

Las investigaciones abarcan temas diversos en seguridad de redes tanto como el modelamiento de esquemas de protección. (Cheminod, M., Durante, L., Seno, L., Valenzano, n.d.).

⁸ Firewall Industriales DPI. Disponible en: http://www.ciberseguridadlogitek.com/wp-content/uploads/Firewalls-industriales_Hirschmann_Tofino_DPI.pdf

También se observan investigaciones que añaden confidencialidad a los datos al interior de redes industriales modificando el protocolo IEC 60870-5-101, el cual es un protocolo de comunicaciones SCADA.(CHERIFI & HAMAMI, 2018).

Estudios muy detallados muestran metodologías de ataques semánticos a redes SCADA, en los cuales los monitores de red no pueden detectar anomalías, pero cuyo alcance puede afectar incluso las interfaces hombre máquina de un operario al interior de un proceso industrial.(Kleinmann et al., 2018).

Una aproximación a estos análisis permite crear al interior de un proceso industrial un esquema de red definida por software que segmenta los diferentes procesos existentes mediante el uso de firewalls, de esta manera para pasar de un segmento a otro es necesario pasar por las reglas de control.

Este esquema se ha validado mediante un prototipo empleando el protocolo de comunicaciones OPC UA y como conclusiones muestra que es altamente efectivo contra ataques que realizan una exploración de la red.(Tsuchiya, A.aEmail Author, Fraile, F.bEmail Author, Koshijima, I.aEmail Author, Órtiz, A.bEmail Author, Poler, 2018).

Otro enfoque pensado para las vulnerabilidades de las redes involucradas en el internet de las cosas, propone aplicar parches a los firewalls existentes, basados en un análisis previo de vulnerabilidades.(Kolisnyk, M.aEmail Author, Piskachova, I.b, Kharchenko, n.d.).

Se incluyen también esquemas de firewalls funcionando en pareja para mejorar el tráfico al interior de la red. (Jiang, Lin, Yin, & Xi, 2017).

Con el mismo objetivo se realizó una búsqueda de patentes en el tema, de esta forma en google patents con las mismas palabras claves lo cual arrojó 62057 resultados, de los cuales el top 1000 muestra como la empresa Rockwell

automation posee el 11.3% de las patentes, seguido por Bedrock Automation Platforms Inc. con el 1.8% y Schneider Automation Inc. con el 1.6% de las patentes.

El mayor dueño de patentes muestra invenciones y modelos de utilidad en diversas áreas como lo son el diseño completo del firewall industrial y sus métodos de ruteo y encapsulación en US 2006/0155865 A1, así como mecanismos para la detección de anomalías los datos en redes industriales US20060236374A1, firewalls con respuesta en tiempo real US20080320582A1.

Estas patentes muestran un crecimiento de productos comerciales por ejemplo la empresa Rockwell ofrece productos como stratix-5950-security-appliance cuyo precio está en función de las utilidades añadidas y puede alcanzar los US\$40000.

4.2.MARCO CONCEPTUAL

Sistema Scada: Constituyen redes especializadas de computadores y dispositivos que permiten monitorear y controlar procesos propios de maquinaria de producción en instalaciones industriales o militares. Dichos procesos involucran la medición de variables propias del mismo la cual es realizada con una amplia variedad de tipos de sensores (temperatura, presión, flujo entre otros). Estos procesos industriales también involucran actuar sobre las variables, para esto se emplean motores, solenoides, pistones entre otros. El SCADA es usado para tomar decisiones por ejemplo para abrir una válvula y liberar agua de un tanque cuando este se llena o por ejemplo iniciar una parada de emergencia en una planta nuclear. Estos sistemas son típicamente empleados en tres principales áreas: (Kruz Ronald 2004).

Manejo de procesos industriales. En estos se pueden incluir procesos de manufactura, producción, procesos químicos, generación de potencia, fabricación e industrias de refinado.

Manejo de infraestructura. Aquí se pueden incluir por ejemplo, procesos de tratamiento de agua, distribución, disposición de basuras, tuberías de combustible y gas, sistemas de transmisión y distribución y grandes sistemas de telecomunicaciones.

Manejo de instalaciones. En esta subcategoría se pueden encontrar oficinas, centros de datos aeropuertos, barcos, etc, para el manejo de aires acondicionados, sistemas de calentamiento, acceso físico y control de consumo de energía.

Entonces el SCADA es el sistema de información de una planta industrial, el cual se comunica con sensores y actuadores usando protocolos de comunicación adaptados a entornos industriales tales como: Device Net, Can, Modbus, DNP, ICCP. De la misma forma el sistema SCADA debe comunicarse con el sistema administrativo para generar reportes, y llevar bases de datos que le permiten a este último la toma de decisiones. La comunicación con el sistema administrativo puede ser realizada con:

- Interfaces Hombre-Máquina.
- Sistemas Supervisorios.
- Terminales remotos.
- Infraestructura de comunicación.

Los Scada son realmente importantes ya que generalmente se encuentran a cargo de infraestructura empresarial o nacional crítica (estaciones de generación y distribución de electricidad, Instalaciones de procesamiento de petróleo etc). En los cuales la pérdida de información, pérdida de acceso o el uso indebido puede llegar a ocasionar daños físicos severos y por lo tanto problemas financieros. De esta forma se puede concluir que la seguridad del sistema Scada debe ser una alta prioridad. (Galdámez P.2008)

Hoy en día los Scada se encuentran interconectados a redes administrativas las cuales a su vez se conectan con el mundo mediante internet, lo cual abre la posibilidad a los típicos problemas de seguridad de toda red de datos. Si a esto se adiciona que muchos protocolos de comunicación industrial no fueron diseñados para su uso en internet las posibilidades de accesos no autorizados o monitoreo no permitido aumentan, aun mas considerando que estos protocolos transmiten texto plano de un punto a otro de la planta. (Kruz Ronald 2004).

El presente proyecto pretende clasificar las topologías de firewall que puedan implementarse en una red industrial, para el filtrado de paquetes, que constituya el núcleo de un firewall embebido, el cual al final pueda ser implementado y probado.

Firewall: Dispositivo que realiza el monitoreo en una red y permite o bloquea el tráfico de acuerdo a las reglas de seguridad predefinidas

Protocolo: Es un estándar que permite la conexión y transmisión de información entre dos o más entidades o computadores.

Red: Son un conjunto de medios técnicos que se usan para enviar comunicación entre ellos. Entre los cuales se existe un transmisor el cual genera y prepara la información. Un canal el cual es la conexión física entre transmisor y receptor. El receptor que es el encargado de recibir la información.

Usuario: Es toda persona, entidad o proceso que obtenga, genere o transforme información a través de un papel o por medio de un medio digital o una red de datos.

Iptables: Es un software integrado al núcleo de sistemas operativos descendientes de UNIX, tales como Linux y FreeBSD, se encarga de la verificación de paquetes tanto entrantes como salientes.

Regla: Es una proposición lógica que permite controlar y tomar una decisión de lo que se debe hacer.

Topología de red: Es un mapa que puede ser físico o lógico y permite el intercambio de datos a través de nodos que van interconectados.

Raspberry pi: Arquitectura completamente abierta, los planos y el firmware pueden descargarse de internet, además esta soportada por una amplia variedad de desarrolladores alrededor del mundo

Servidor: Es un programa que se ejecuta y realiza conexiones con el cliente bidireccionales o unidireccionales y síncronas o asíncronas en donde le da una respuesta usando un lenguaje

4.3.MARCO LEGAL

El marco legal colombiano que involucra el uso, distribución y creación tanto de software libre como privado, hace uso de leyes que abarcan los aspectos de Derechos de autor, Comercio electrónico y firmas digitales, propiedad industrial, políticas públicas, protección de datos y el código penal colombiano ⁹ Este conjunto de normativas se enumeran a continuación con la intención de aproximarse a la pirámide de Kelsen¹⁰, en lo referente al orden de las leyes.

Leyes de Derechos de autor

Decisión 351 de la CAN

Ley 23 de 1982

⁹ Sensibilización seguridad de la información, Agosto de 2013. Disponible en : http://www.mineducacion.gov.co/1759/articles-327021_archivo_pdf_Dia1_6_Sensibilizacion.pdf

¹⁰ Pirámide De Kelsen, febrero de 2011. Disponible en: <https://iusuniversalis.blogia.com/2011/022402-piramide-de-kelsen.php>

Decreto 1360 de 1989
Ley 44 de 1993
Decreto 460 de 1995
Decreto 162 de 1996
Ley 545 de 1999
Ley 565 de 2000
Ley 603 de 2000
Ley 719 de 2001
Art. 67 Decreto Nacional 3942 de 2010

Leyes de Comercio electrónico y firmas digitales.

Decreto Nacional 333 de 2014
Resolución 26930 de 2000
Decreto 2364 de 2012
Circular externa 042 del 2012 (SFC)

Leyes de propiedad industrial

Ley 170 de 1994 - Organización Mundial de Comercio
Ley 463 de 1998 – Tratado de cooperación de patentes
Leyes de Políticas Públicas.

Ley 1341 de 2009
Decreto 32 del 2013
Circular 052

Leyes de Protección de datos

Ley 1266 de 2008

Ley 1581 de 2012

Código penal colombiano.

Ley 1273 de 2009

Decisión 351 de la CAN

Este documento hace referencia a un conjunto de definiciones y normas firmado por los países de la comunidad andina de naciones, en los cuales se definen las políticas comunes referentes a protección de los derechos de autor en los campos literarios, artísticos, científicos e industriales. En este documento tanto los programas de computador como las bases de datos se protegen siguiendo los mismos lineamientos que protegen a las obras literarias. Lo cual es ampliamente detallado en el capítulo VIII del mismo documento.¹¹

Ley 23 de 1982.

Esta ley fundamenta legalmente que son los derechos de autor en Colombia, a quienes cubre, cuáles son sus derechos morales, patrimoniales y su tiempo de aprovechamiento. Al ser el software protegido como una obra literaria esta ley define todo el conjunto de criterios que protege al escritor del mismo.¹²

Decreto 1360 de 1989

Este decreto reglamenta los pasos que deben seguir los desarrolladores de software para realizar su inscripción en el registro nacional de derechos de autor

¹¹ Decisión 351 de la comunidad andina de naciones. Disponible en:
<http://www.wipo.int/edocs/lexdocs/laws/es/can/can010es.pdf>

¹² Ley 23 de 1982, Sobre los derechos de autor. Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=3431>

con el objetivo de proteger su código fuente y códigos objeto, aquí referidos con generalidad en soporte lógico.¹³

Ley 44 de 1993

Se hace una actualización de la ley 23 de 1982 en lo referente a los derechos de servidores públicos sobre software desarrollado por los mismos, a la duración de los derechos de autor en personas naturales y jurídicas, al derecho de autor colectivo y a las sanciones hacia quienes no respeten estos derechos.¹⁴

Los decretos 162 de 1996 y 460 de 1995, hacen referencia a la formalización de procedimientos y cobros respectivos por los servicios prestados por el estado en lo referente al registro en el sistema nacional de derechos de autor (Decreto 460) y a la reglamentación nacional de la decisión de 365, enumerada once años antes.¹⁵

Es necesario aquí resaltar que esta ley fue derogada por el art. 67 del Decreto Nacional 3942 de 2010, en varios de sus aspectos, especialmente a la adjudicación colectiva de los derechos de autor.

En lo referente a derechos de autor las Ley 545 de 1999 y Ley 565 de 2000 adoptan y reglamentan en Colombia el tratado de la OMPI en lo referente a derechos de autor y su relación con las leyes anteriores, derogándolas o modificándolas. De la misma forma las leyes 603 de 2000 y 719 de 2001 se enfocan en los cobros por impuestos de derechos de autor que deben pagarse al estado por el uso de obras licenciadas.

¹³ DECRETO 1360 DE 1989, inscripción del soporte lógico. Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=10575>

¹⁴ Ley 44 de 1993, modificación a la ley 23 de 1982. Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=3429>

¹⁵ Decreto 162 de 1996. Disponible en:
<http://www.alcaldiabogota.gov.com/sisjur/normas/Norma1.jsp?i=10574>

Decreto Nacional 333 de 2014

Este decreto explica y reglamenta el uso de las firmas digitales y definiendo quienes son los organismos certificadores su funcionamiento y costos asociados, por lo tanto define un marco legal para el comercio electrónico en Colombia.¹⁶

Por su parte la definición de políticas públicas regulatorias al interior de las entidades estatales se reglamentan en las leyes 1341 de 2009 y el Decreto 32 del 2013, creando la agencia nacional del espectro, y las políticas públicas que definen: "la protección al usuario, así como lo concerniente a la cobertura, la calidad del servicio, la promoción de la inversión en el sector y el desarrollo de estas tecnologías, el uso eficiente de las redes y del espectro radioeléctrico, así como las potestades del Estado en relación con la planeación, la gestión, la administración adecuada y eficiente de los recursos, regulación, control y vigilancia del mismo y facilitando el libre acceso y sin discriminación de los habitantes del territorio nacional a la Sociedad de la Información".¹⁷

La Ley 1266 de 2008 y la Ley 1581 de 2012 son de vital importancia ya que definen la protección de los datos y el manejo que debe ser dado tanto por entidades públicas como privadas, de esta forma se protege la información financiera, industrial, científica y personal almacenada en bases de datos en Colombia. Ambas son reglamentadas en la ley 1377 DE 2013.¹⁸

LEY 1273 DE 2009

¹⁶ DECRETO 333 DE 2014, Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=56767#22>

¹⁷ Ley 1341 de 2019, Disponible en: http://www.mintic.gov.co/portal/604/articles-3707_documento.pdf

¹⁸ Ley estatutaria 1581 de 2012. Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=49981>

Define un nuevo bien Jurídico con peso legal, el cual debe ser protegido como cualquier otro bien, pero que rige en la protección de la información y de los datos, resalta además que se deben preservar integralmente los sistemas que utilicen las tecnologías de la información en el manejo de información, comercial, financiera, industrial etc. resalta las definiciones de los atentados contra la confidencialidad, la integridad y la disponibilidad de los datos y de los sistemas informáticos así como las sanciones que acarrear. ¹⁹

Este marco legal asociado al proyecto desarrollado, indica las protecciones legales que lo cobijan, así como su utilidad que el desarrollo forma parte de la cadena de protección de datos, consagrada en la ley 1581. De esta forma se comprende el entorno legal para su aplicación y explotación.

5. DESARROLLO FASES DEL PROYECTO

Fase Diseño

El presente texto se enfoca en el análisis descriptivo de tipo exploratorio. Un estudio descriptivo permite recolectar información medible mediante la observación de conceptos y variables aplicables al problema como tal. Estas últimas señalan patrones de la realidad, que pueden inferirse y cuyos valores pueden cambiar la dinámica del sistema estudiado.

Debe aclararse que la parte exploratoria define una aplicación especial, debido a que da pie a la formulación de hipótesis de grado uno y grado dos, las cuales pueden ser sobresalientes en un nivel más detallado del análisis que se lleva a cabo.

¹⁹ LEY 1273 DE 2009, Código penal, Disponible en:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34492>

Con bases en variables conocidas a priori se puede tener un marco de referencia del problema, lo que permite tener planteamientos más claros del mismo y los que además se constituyen en la base para formulación de experimentos.

Una vez definidos: el problema, el marco teórico, los objetivos y el método se establece como el siguiente paso: el diseño de la encuesta o del mecanismo de recolección de datos. Este mecanismo de medición debe almacenar datos observables sobre los conceptos o las variables. La robustez de este se da al cumplir dos objetivos básicos: Los datos deben ser confiables y deben ser validos dentro del experimento registrado.

Fase diseño del software

Para realizar el diseño y desarrollo de un sistema de información se referencia como:

Diseño en espiral debido a que se pueden retomar pasos anteriores sobre cualquier punto del proceso siguiendo los procesos lógicos de desarrollo y verificación en ingeniería que en muchas ocasiones dan lugar a correcciones que obligan a la generación de nuevas versiones.

La aplicación se desarrollara sobre plataformas de hardware y software de libre distribución en donde sus arquitecturas son completamente conocidas de allí que las variables más importantes como el sistema operativo, la capacidad de procesamiento y las interfaces de red se conocen a priori y dan respaldo al desarrollo mismo.

El programa como tal se ejecutará sobre el sistema operativo Linux (raspbian jessie) y se desarrollara sobre el intérprete de Python 2.7. Lo cual le da amplia portabilidad.

Fase recolección de datos

La técnica utilizada para la recolección de datos es la recopilación documental en donde se usará fuentes bibliográficas como libros, textos, tesis, páginas web, revistas y todos los documentos necesarios para recopilar toda la información relacionada con el proyecto.

Técnica de Análisis de Datos

Revisión documental Se consultará la información necesaria que respalde los conceptos sobre firewall industrial, protocolo modbus y sistema scada

6. DESARROLLO DEL FIREWALL

Se muestran de forma individual los componentes que forman parte de una topología de firewall basado en la detección de paquetes, probado sobre un protocolo de red industrial, en este caso modbus.

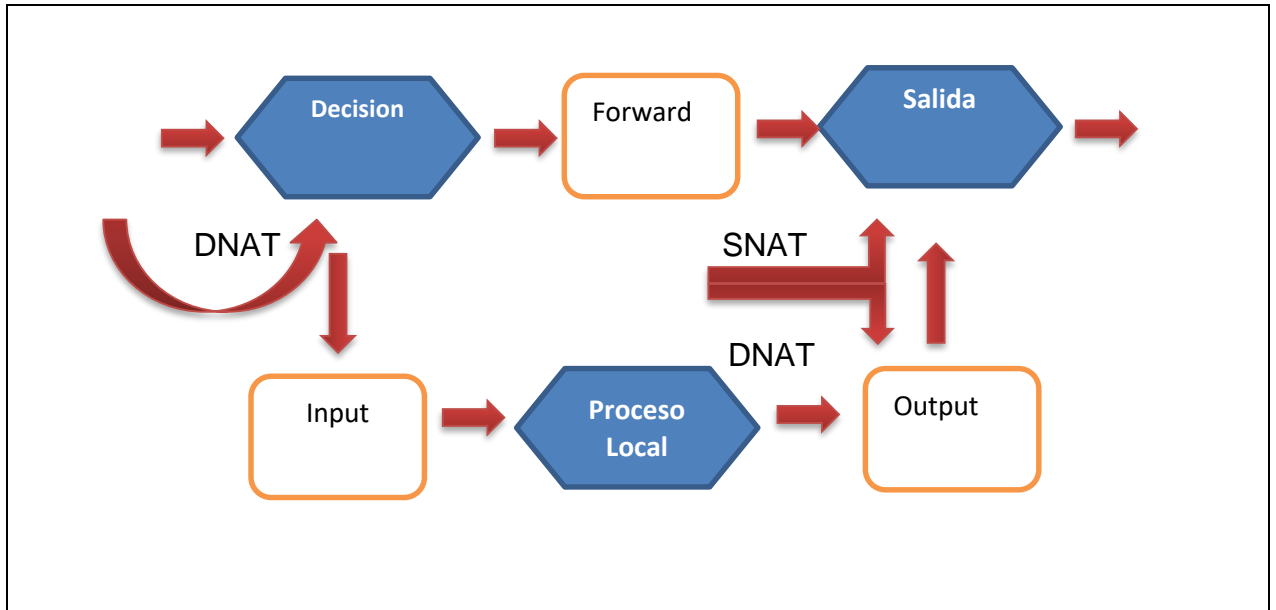
6.1 IPTABLES

Es un software integrado al núcleo de sistemas operativos descendientes de UNIX, tales como Linux y FreeBSD, se encarga de la verificación de paquetes tanto entrantes como salientes. Esta revisión se realiza a nivel de kernel, lo cual asegura una alta confiabilidad en la detección ya que se necesitaría de un acceso completo al sistema para tener privilegios de administrador y de esta manera modificar su operación. Al estar ligado al núcleo del sistema operativo, se garantiza su operación, solo es posible modificar sus características empleando comandos propios del sistema o a través de lenguajes de alto nivel con interfaces de programación de alto nivel especializadas.

La operación de iptables es simple: Cada paquete que llega al sistema o cada paquete que debe salir del mismo, es analizado por protocolo y puerto; en el análisis cada paquete es comparado con un conjunto de reglas predefinidas en el sistema y conforme a estas se permite o deniega su paso.

El núcleo de Linux soporta los protocolos estándar, TCP,IP,UDP,ARP,NAT DRP etc y todos los posibles puertos de sistema. Por lo tanto al tener tantas posibles combinaciones, antes de escribir las reglas del iptables es necesario definir una política que guíe la escritura de las reglas, de esta manera se puede controlar la escritura de las mismas.

Figura 1. Viaje de un paquete al interior del kernel de Linux.



Fuente El Autor

Como se ve en la Figura 1, el núcleo valida si el paquete tiene como destino su máquina u otra. En aquellos paquetes o datagramas que tienen destino la misma máquina se le aplica reglas INPUT y OUTPUT. En el caso que el destino es una máquina diferente se le aplica reglas FORWARD.

Entre las reglas de filtrado tenemos las INPUT, OUTPUT, FORWARD y NAT que se usa para re direccionar puertos y cambiar direcciones ip tanto origen como destino.

A continuación dos principales reglas en iptables:

- **NAT:** reglas PREROUTING, POSTROUTING.
- **FILTER:** reglas INPUT, OUTPUT, FORWARD.²⁰

Aunque el objetivo aquí no es hacer la configuración de iptables desde el shell de Linux, si es importante conocer su sintaxis y semántica. Dado que pueden tenerse múltiples reglas con configuraciones particulares lo que al final resulta en un script

²⁰ iptables Manual Práctico, 2014, internet: <https://mizonapc.wordpress.com/category/linux/page/2/>

de Linux demasiado grande y con bastante complejidad, lo que se convierte generalmente en el origen de los errores de configuración.

El comando IP tables soporta varias funcionalidades, así:

- -A –append → agrega una regla a una cadena.
- -D –delete → borra una regla de una cadena especificada.
- -R –replace → reemplaza una regla.
- -I –insert → inserta una regla en lugar de una cadena.
- -L –list → muestra las reglas que le pasamos como argumento.
- -F –flush → borra todas las reglas de una cadena.
- -Z –zero → pone a cero todos los contadores de una cadena.
- -N –new-chain → permite al usuario crear su propia cadena.
- -X –delete-chain → borra la cadena especificada.
- -P –policy → explica al kernel qué hacer con los paquetes que no coincidan con ninguna regla.
- -E –rename-chain → cambia el orden de una cadena.

Cuando se desean aplicar las reglas, es necesario definir algunas condiciones generales.

- -p –protocol → la regla se aplica a un protocolo.
- -s –src –source → la regla se aplica a una IP de origen.
- -d –dst –destination → la regla se aplica a una IP de destino.
- -i –in-interface → la regla se aplica a una interfaz de origen, como eth0.
- -o –out-interface → la regla se aplica a una interfaz de destino.²¹

Sintaxis de una regla:

```
$ iptables -A TipoDeRegla -i Interfaz -s OrigenDelPaquete -p protocolo -dport puertodestino -j Decision
```

²¹ P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

Por ejemplo una regla que acepta conexiones por el puerto 80, se define de la siguiente forma:

```
$ iptables -A INPUT -i eth0 -s 0.0.0.0/0 -p TCP -dport www -j ACCEPT
```

Dónde:

-i eth0: interfaz de red eth0.

-s 0.0.0.0/0: dirección de acceso (cualquiera en este caso).

-p TCP: protocolo.

-dport: puerto de destino.

-j ACCEPT: destino del paquete (es aceptado, podría ser DROP, LOG, REJECT,..).²²

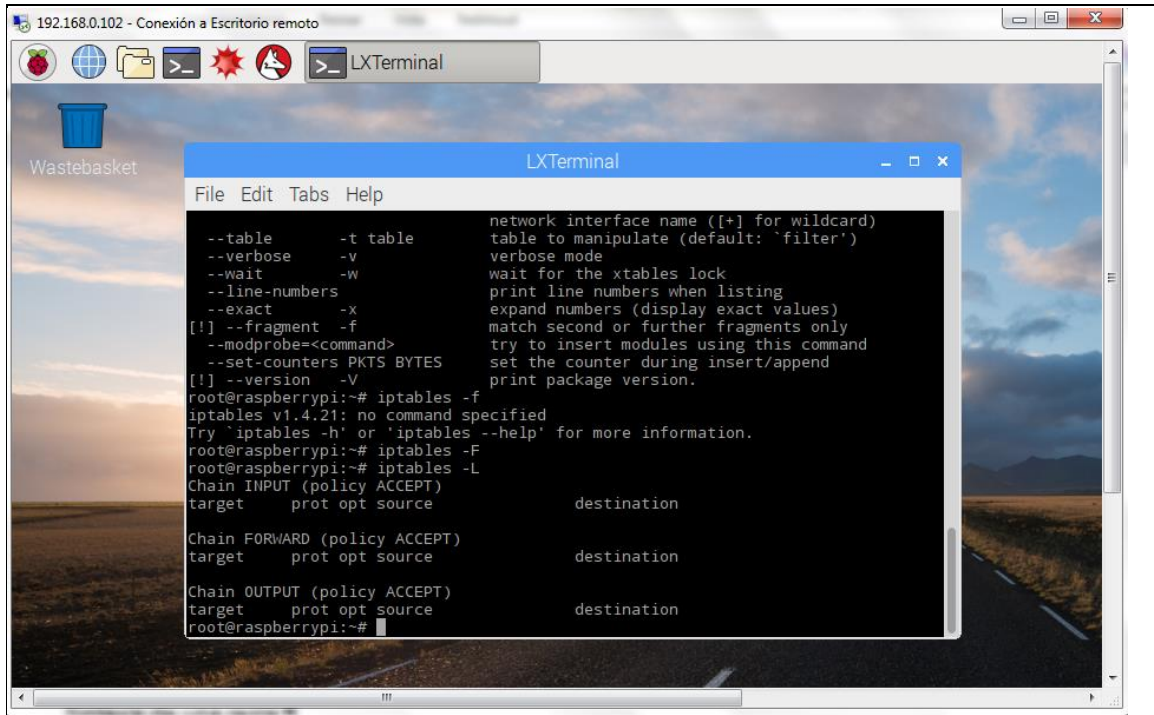
Otro punto importante se relaciona con la política por defecto con la cual el sistema opera, se trata de definir una característica del mismo para aceptar o denegar el paso de paquetes, de esta manera si la política es denegar todas las direcciones ip y puertos entonces las reglas solo necesitaran adicionar puertos y direcciones permitidas.

También puede operarse con una política en la cual se acepta todo, tanto para los paquetes de INPUT, OUTPUT COMO FORWARD, de esta manera todos los puertos y protocolos son aceptados, esto puede llegar a ser muy peligroso ya que cualquier programa puede iniciar el uso de un puerto específico sin previo aviso, e iniciar de esta manera comunicaciones no permitidas, esto obliga a un monitoreo constante de puertos para analizar si están permitidos o no.

En la figura 2, se muestra la política por defecto de Linux en su versión Raspbian Jessie, en conexión xdrp.

²² P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

Figura 2. Política de iptables, Raspbian jessie lts



Fuente El Autor

En este ejemplo la salida del comando `iptables -L` arroja como resultado una política de aceptar todo.

La configuración por defecto de un firewall debe ser: “bloquear todo excepto [reglas]”. Para configurar el firewall para que bloquee todas las conexiones debemos teclear:

```
iptables -P INPUT DROP
```

```
iptables -P FORWARD DROP
```

```
iptables -P OUTPUT DROP
```

Con esto el sistema pierde toda conectividad no opera ningún protocolo de internet ni de conexión por escritorio remoto xdrp, por lo que a continuación se deben agregar reglas permisivas para una conectividad mínima pero controlada.

Para aplicar una regla que filtre un determinado puerto, se debe ejecutar:

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

El siguiente ejemplo muestra un script más complejo en definición de reglas y políticas de uso:

```
iptables -F
```

```
iptables -X
```

```
iptables -Z
```

```
iptables -t nat -F
```

```
## Se establece una política por defecto
```

```
iptables -P INPUT ACCEPT
```

```
iptables -P OUTPUT ACCEPT
```

```
iptables -P FORWARD ACCEPT
```

```
iptables -t nat -P PREROUTING ACCEPT
```

```
iptables -t nat -P POSTROUTING ACCEPT
```

```
# El localhost se deja (por ejemplo conexiones locales a mysql)
```

```
iptables -A INPUT -i lo -j ACCEPT
```

```
# Se acepta todo de la ip relacionada
```

```
iptables -A INPUT -s 168.4.1 -j ACCEPT
```

```
# se permite el acceso de la ip relacionada al puerto 3306 de la DB
```

```
iptables -A INPUT -s 168.4.2 -p tcp --dport 3306 -j ACCEPT23
```

```
# se permite el acceso de la ip relacionada los puertos 20 y 21 del ftp
```

²³ P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

```
iptables -A INPUT -s 168.4.3 -p tcp -dport 20:21 -j ACCEPT
```

El puerto 80 de www debe estar abierto si se opera un servidor web.

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

se cierran el resto de los puertos

```
iptables -A INPUT -p tcp --dport 20:21 -j DROP
```

```
iptables -A INPUT -p tcp --dport 3306 -j DROP
```

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

```
iptables -A INPUT -p tcp --dport 10000 -j DROP
```

Las políticas de cerrar todo son más eficientes, ya que dan lugar a la apertura solo de los puertos necesarios, sin dejar posibilidades a programas y usuarios.

En otro ejemplo, se configura un equipo expuesto a internet.

```
iptables -F
```

```
iptables -X
```

```
iptables -Z
```

```
iptables -t nat -F
```

```
## Establecemos política
```

```
iptables -P INPUT ACCEPT
```

```
iptables -P OUTPUT ACCEPT
```

```
iptables -P FORWARD ACCEPT
```

```
iptables -t nat -P PREROUTING ACCEPT24
```

```
## Nota: eth0 es el interfaz conectado al router y eth1 a la LAN
```

```
# El localhost se deja
```

²⁴ P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

```
iptables -A INPUT -i lo -j ACCEPT
```

Se permite el acceso a un ip específico en una interfaz de red específica

```
iptables -A INPUT -s 192.168.4.0/30 -i eth1 -j ACCEPT
```

Ahora con regla FORWARD se filtra el acceso de la red local

los paquetes que no van dirigidos a la propia maquina se les aplican reglas de FORWARD

se permite que vayan a puertos 80

```
iptables -A FORWARD -s 192.168.4.0/30 -i eth1 -p tcp --dport 80 -j ACCEPT
```

#Se permiten los puertos https

```
iptables -A FORWARD -s 192.168.4.0/30 -i eth1 -p tcp --dport 443 -j ACCEPT
```

Se permite la consulta DNS

```
iptables -A FORWARD -s 192.168.4.0/30 -i eth1 -p tcp --dport 53 -j ACCEPT
```

```
iptables -A FORWARD -s 192.168.4.0/30 -i eth1 -p udp --dport 53 -j ACCEPT
```

Se deniega el resto.

```
iptables -A FORWARD -s 192.168.4.0/30 -i eth1 -j DROP
```

25

Ahora se permite el progreso de paquetes con forward

Que otras máquinas puedan salir a través de la maquina actual.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

²⁵ P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

```
## Y se cierran los accesos no deseados:  
# Nota: 0.0.0.0/0 significa: cualquier red  
# Se deniega un rango de puertos  
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 1:1024 -j DROP  
iptables -A INPUT -s 0.0.0.0/0 -p udp -dport 1:1024 -j DROP  
# Se cierra un puerto de administracion: webmin  
iptables -A INPUT -s 0.0.0.0/0 -p tcp -dport 10000 -j DROP26
```

Como se observa las configuraciones son complejas y la probabilidad de errores es muy alta.

6.2 PYPTABLES

Surge como una alternativa de configuración a las Iptables del sistema operativo, empleando un lenguaje de alto nivel como el python. Con esta herramienta se hace más sencillo el trabajo con las reglas, es posible automatizar los procesos de creación y borrado, además pueden realizarse verificaciones previas sobre los puertos y el estado en sí de la red. Solo conllevan a resultados cuando estos programas son ejecutados desde la interfaz de administrador y sobre la misma máquina.

Pytables es un conjunto de objetos escritos en python que permiten la manipulación de las reglas de una forma más amigable. La clase principal es la clase Tables, la cual representa una colección de iptables (filter, mangle, nat). La mayoría de las veces se utiliza la estructura default_tables(), la cual crea la estructura básica de reglas y cadenas viables en el kernel de linux. Devuelve un diccionario, indexable por nombre de tabla mediante el operador.

²⁶ P. Xabier Altadill, IPTABLES, Disponible en: <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

```
tables = default_tables()
```

```
table = tables['filter ']
```

Las tablas también pueden indexarse por la cadena ya sea INPUT, OUTPUT O FORWARD.

```
Cadena = tables ['filter']['INPUT'].
```

Las cadenas mantienen un conjunto ordenado de reglas, las cuales pueden estar clasificadas en preinstaladas en el sistema o construidas por el usuario. Pyptables posee su propia clase para representarlas en BuiltChain, UserChain. Y la primera contiene la política por defecto ejecutada por el sistema.

De esta manera los diccionarios de tablas pueden ser complementados con las cadenas empleando los operadores de los objetos.

- append (regla).
- remove (regla).
- insert(regla).
- insert (regla , posicion) ²⁷

En las sintaxis de pyptables se implementan de la forma:

```
tables ['filter']['INPUT'].append (Rule(..))
```

```
tables ['filter']['INPUT'].insert (Rule (..), 0)
```

Un ejemplo en python que devuelve las políticas programadas en el sistema operativo e:

```
def default_tables():  
    """Devuelve todas las tablas y cadenas pre instaladas"""  
  
    return Tables(Table('filter',
```

²⁷ J. Cockburn, Python PyPTables 1.0.1, Disponible en:
<https://pypi.python.org/pypi/PyPTables/1.0.1>


```

        BuiltinChain('INPUT', 'ACCEPT'),
        BuiltinChain('FORWARD', 'ACCEPT'),
        BuiltinChain('OUTPUT', 'ACCEPT'),
    ),
    Table('nat',
        BuiltinChain('PREROUTING', 'ACCEPT'),
        BuiltinChain('OUTPUT', 'ACCEPT'),
        BuiltinChain('POSTROUTING', 'ACCEPT'),
    ),
    Table('mangle',
        BuiltinChain('PREROUTING', 'ACCEPT'),
        BuiltinChain('INPUT', 'ACCEPT'),
        BuiltinChain('FORWARD', 'ACCEPT'),
        BuiltinChain('OUTPUT', 'ACCEPT'),
        BuiltinChain('POSTROUTING', 'ACCEPT'),
    ),
)28

```

La clase Rule representa una regla actual de iptables, estas son creadas usando la sintaxis simple de python y pueden ser adicionadas a una cadena. Por ejemplo el siguiente llamado puede producir una regla la cual atrapa el tráfico destinado al puerto 502 y lo rechaza.

```
reject_mod = Rule(proto='tcp', dport='502', jump='REJECT')
```

Se puede por ejemplo adicionar una regla para prevenir el acceso ssh a la maquina local:

```
tables['filter']['INPUT'].append(reject_ssh)
```

²⁸ J. Cockburn, Python PyPTables 1.0.1, Disponible en: <https://pypi.python.org/pypi/PyPTables/1.0.1>

Esto al momento de la ejecución produce los comandos de iptables vistos en la sección anterior:

```
- A INPUT -p tcp -j REJECT --dport 22
```

En pytables existen varios tipos de reglas predefinidas que proveen acciones por defecto para varios tipos de parámetros comunes, por ejemplo los objetivos comunes de salto ACCEPT,DROP,REJECT etc. Se escriben de la siguiente forma:

```
from pytables.rules import Reject
reject_ssh = Reject(proto='tcp', dport='22')
```

Así la regla completa puede asignarse a una variable para ser usada después. Y se pueden crear nuevos tipos de reglas, por ejemplo un tipo de regla para atrapar paquetes ssh pero aplicado de diferentes formas.

```
SSH = Rule(proto='tcp', dport='22')
tables['filter']['INPUT'].append(SSH(jump='ACCEPT', source='1.1.1.1',
comment='Permitir SSH desde una estación de trabajo'))
```

```
tables['filter']['INPUT'].append(SSH(jump='REJECT', comment='Evita accesos
SSH'))
```

```
tables['filter']['FORWARD'].append(SSH(jump='REJECT', comment='Evita el
envoi de trafico ssh '))
```

El código anterior de PyTables genera el siguiente código en script de Linux:

```
#####
# filter table (/blocker/share/python/iptables/__init__.py:14 default_tables) #
#####
*filter
:INPUT ACCEPT [0:0]
```

```
:FORWARD ACCEPT [0:0]
```

```
:OUTPUT ACCEPT [0:0]
```

```
# Builtin Chain "INPUT" (/blocker/share/python/iptables/__init__.py:12
default_tables)"
```

```
# Rule: Allow access to local SSH from my workstation (<stdin>:1 <module>)
```

```
-A INPUT -p tcp -s 1.1.1.1 -j ACCEPT --dport 22 -m comment --comment
"Permitir SSH desde una estación de trabajo"
```

```
# Rule: Prevent any other access to local SSH (<stdin>:1 <module>)
```

```
-A INPUT -p tcp -j REJECT --dport 22 -m comment --comment " Evita accesos
SSH"29
```

```
# Builtin Chain "FORWARD" (/blocker/share/python/iptables/__init__.py:13
default_tables)"
```

```
# Rule: Prevent any SSH traffic being routed through this box (<stdin>:1
<module>)
```

```
-A FORWARD -p tcp -j REJECT --dport 22 -m comment --comment " Evita el
envoi de trafico ssh "
```

```
# Builtin Chain "OUTPUT" (/blocker/share/python/iptables/__init__.py:14
default_tables)"
```

```
# No rules
```

Las reglas creadas en pyptables son almacenadas en la iptables con la función restore, teniendo como argumento a las tablas creadas al interior.

La interfaz Pyptables permite configurar cualquier regla de filtrado de iptables desde un lenguaje intuitivo, de alto nivel con la posibilidad de automatizar la

²⁹ J. Cockburn, Python PyPTables 1.0.1, Disponible en:
<https://pypi.python.org/pypi/PyPTables/1.0.1>

creación de las mismas, reduciendo de esta forma la probabilidad de errores y la disminución de los tiempos de depuración de las mismas.

6.3 PYMODBUS

PYMODBUS es una implementación de software libre en python, bajo licencia gnu en la cual es posible implementar servidores modbus y cliente modbus. Recuérdese que modbus es un protocolo de comunicaciones de alto uso en redes industriales. Es en este punto necesario recordar que un cliente modbus se asocia con un dispositivo de planta asociado a alguna máquina, sensor o actuador y que puede recibir o enviar comandos en función de los requerimientos de un servidor. Los servidores toman el control de clientes pero también están asociados a una máquina en un punto de vista más amplio. Entonces en el contexto de una red industrial todos los equipos pueden considerarse como pequeños computadores embebidos con la obligación de devolver datos de operación y de actuar en tiempo real. El tiempo real es una característica necesaria ya que el control físico de los procesos controlados por los mismos así lo requiere.

Se pasa entonces a la descripción de las clases interpretadas que permiten el funcionamiento como clientes y como servidor.

Figura 3. Servidor Modbus en PYMODBUS

```
#!/usr/bin/env python
#-----#
# Implementación general del servidor
# Tomado de:
# https://pymodbus.readthedocs.io/en/latest/examples/asynchronous-
# server.html
# Modificado Por: Henry Jimenez Rosero.
#-----#
```

```

from pymodbus.server.async import StartTcpServer
from pymodbus.server.async import StartUdpServer
from pymodbus.server.async import StartSerialServer

from pymodbus.datastore import ModbusSequentialDataBlock
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext

#-----#
# Configura el loggeo de información, con propósitos de depuración
#-----#
import logging
logging.basicConfig()
log = logging.getLogger()
log.setLevel(logging.DEBUG)

#-----#
# Inicializa los almacenes de datos, que referencian bobinas, entradas o # salidas
# del mismo
#-----#
store = ModbusSlaveContext(
# Separa espacio para las entradas discretas del dispositivo
    di = ModbusSequentialDataBlock(0, [17]*100),
# Separa espacio para el manejo de las bobinas de salida
    co = ModbusSequentialDataBlock(0, [17]*100),
# Define el espacio la los registros de retención
    hr = ModbusSequentialDataBlock(0, [17]*100),
# Establece el espacio para los registros de entrada.
    ir = ModbusSequentialDataBlock(0, [17]*100))
# Crea el context de operación, que no es más que el espacio en memoria

```

```
# de todo el objeto modbus.
context = ModbusServerContext(slaves=store, single=True)

#-----#
# Inicia la ejecución del servidor en el contexto definido previamente
#-----#
StartTcpServer(context)
#StartUdpServer(context) #Puede ser UDP
#StartSerialServer(context, port='/tmp/tty1') #o via serial USB o rs-232
```

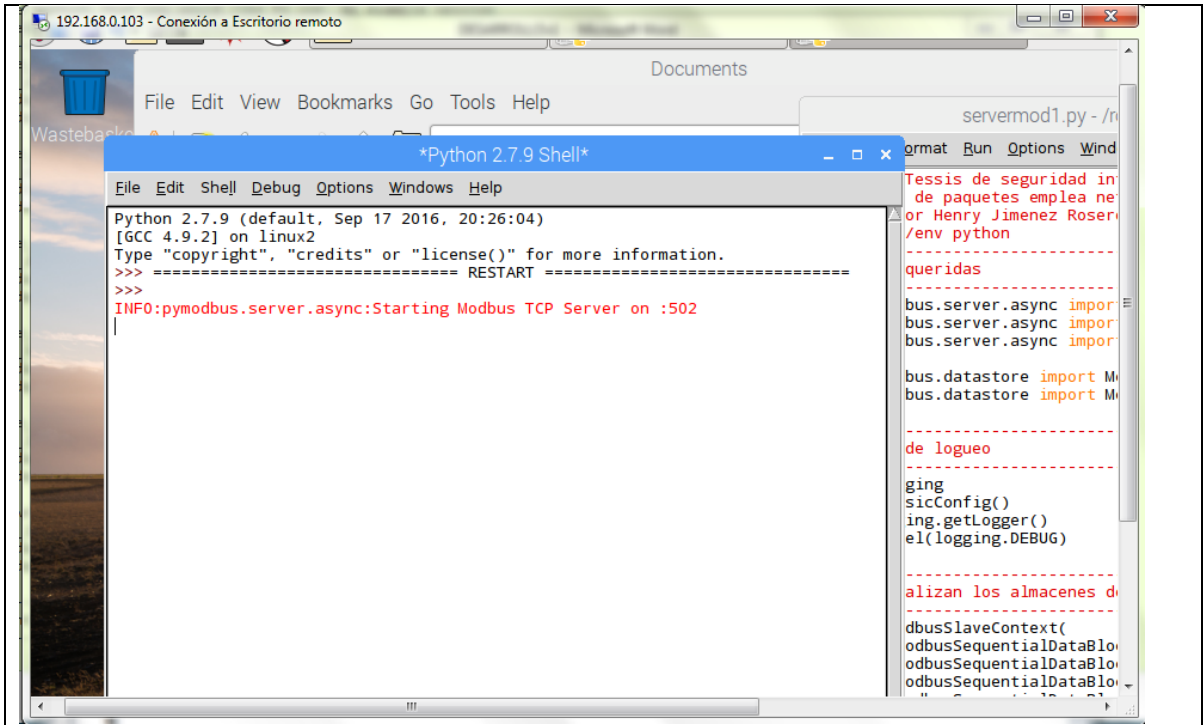
Fuente El Autor

Los almacenes de datos responden únicamente a las direcciones que obtuvieron en la inicialización del servidor, si por ejemplo se define de 0x00 a 0xFF un requerimiento de 0x100 es respondido con una excepción de dirección invalida.

En el programa servidor se crean cuatro almacenes de datos secuenciales, el primero inicia en la dirección cero y es de tamaño [17]*100, el segundo inicia en la posición [17]*100 + 1 y así sucesivamente, se asignan nombres para hacer referencia a data input di, Coil out co, ir input registers ir, half register hf. Propios de la configuración de un módulo modbus.

Finalmente se inicializa el servidor y se lanza en modo TCP, aunque la clase soporta varios protocolos, udp, serial etc. La figura 4 muestra al servidor en ejecución, en el entorno raspberry pi.

Figura 4. Servidor Modbus asíncrono en ejecución.



Fuente El Autor

Figura 5. Cliente Modbus

```
#!/usr/bin/env python
'''
Pymodbus Asynchronous Client
.
'''
#-----
#-----#
from pymodbus.client.async import ModbusTcpClient as ModbusClient
#from pymodbus.client.async import ModbusUdpClient as ModbusClient
#from pymodbus.client.async import ModbusSerialClient as ModbusClient
#-----#
```

```

# Configura el loggeo de información con propósitos de depuración
#-----#
import logging
logging.basicConfig()
log = logging.getLogger()
log.setLevel(logging.DEBUG)

#-----#

client = ModbusClient('127.0.0.1') # la dirección ip hace referencia a la
#ubicación del servidor, en este ejemplo se supone un servidor local

#-----#
# probador de las llamadas
#-----#
def dassert(deferred, callback):
    def _tester():
        assert(callback())
    deferred.callback(_tester)
    deferred.errback(lambda _: assert(False))

#-----#
# requerimientos
#-----#
# simplemente se llaman a los métodos que desean usarse.
#
#-----#
rq = client.write_coil(1, True) # Escribe 1 en la bobina 1
rr = client.read_coils(1,1) # Lee un dato de la bobina 1, permite verificar el estado

```



```

# de la misma
dassert(rq, lambda r: r.function_code < 0x80)
dassert(rr, lambda r: r.bits[0] == True)
rq = client.write_coils(1, [True]*8)
rr = client.read_coils(1,8)
dassert(rq, lambda r: r.function_code < 0x80)
# Las funciones dassert imlementan un metodo de verificacion de datos atrasados
# o dicho de otra forma de datos que aun no han sido leidos por el cliente.
dassert(rr, lambda r: r.bits == [True]*8)      #
rq = client.write_coils(1, [False]*8)
rr = client.read_discrete_inputs(1,8)
dassert(rq, lambda r: r.function_code < 0x80)
# dassert(rr, lambda r: r.bits == [False]*8)
  rq = client.write_register(1, 10)
rr = client.read_holding_registers(1,1)
dassert(rq, lambda r: r.function_code < 0x80)
dassert(rr, lambda r: r.registers[0] == 10)
  rq = client.write_registers(1, [10]*8)
rr = client.read_input_registers(1,8)
dassert(rq, lambda r: r.function_code < 0x80)  #
dassert(rr, lambda r: r.registers == [10]*8)  #
rq = client.readwrite_registers(1, [20]*8)
rr = client.read_input_registers(1,8)
dassert(rq, lambda r: r.function_code < 0x80)  #
dassert(rr, lambda r: r.registers == [20]*8)  #
#-----
#-----
client.close()

```

Fuente El Autor

La operación del cliente es sencilla, debido a la configuración maestro esclavo del bus, selecciona el esclavo mediante su dirección ip y posteriormente lee o escribe sobre las diferentes áreas de datos.

6.4 PLATAFORMA Y SISTEMA OPERATIVO

Se dimensiona una plataforma hardware que pueda ser convertida en un firewall embebido. Para esto la plataforma debe cumplir con las siguientes características.

- Plataforma hardware abierta.
- Sistema operativo de dominio público.
- Con la capacidad de soportar por lo menos dos tarjetas de red, cableadas y/o inalámbricas.
- Con capacidad para ejecutar compiladores e intérpretes de alto nivel como C/C++ y python.
- Con bajo consumo de potencia.
- De bajo precio.
- Con una capacidad de procesamiento que se enmarque dentro de la capacidad de ejecución de servidores, web y de bases de datos pequeños.

Existen en el mercado plataformas hardware que reúnen algunas de las características anteriores, por ejemplo los Intel Edison e Intel galileo, que se apoyan en la arquitectura x86, pero con un soporte de software reducido y costos elevados.

También se encuentra en el mercado la plataforma raspberry pi, la cual es una arquitectura completamente abierta, los planos y el firmware pueden descargarse de internet además esta soportada por una amplia variedad de desarrolladores alrededor del mundo; actualmente está en capacidad de ejecutar los siguientes sistemas operativos:

Tabla 1. Sistemas operativos soportados por Rpi.

- AROS
- Linux
 - Android
 - Arch Linux ARM.
 - Raspbian Jessie.
 - Debian Wheezy Soft-Float, versión de Debian sin soporte para coma flotante por hardware
 - DietPi, distribución ligera basada en Raspbian y de sencilla configuración mediante menús
 - Firefox OS
 - Gentoo Linux
 - Google Chromium OS
 - Kali Linux
 - Open webOS.
 - PiBang Linux, distribución Linux derivada de Raspbian con diferente escritorio y aplicaciones.
 - Pidora, versión Fedora Remix optimizada.
 - QtonPi, distribución linux con un framework de aplicaciones multiplataforma basado en Qt framework
 - Raspbian,¹⁰⁴ versión de Debian Wheezy para ARMv6 con soporte para coma flotante por hardware
 - Slackware ARM, también conocida como ARMedslack
 - Ubuntu MATE
- Plan 9 from Bell Labs^{105 106}
- RISC OS 5²
- Unix
 - FreeBSD¹⁰⁷
 - NetBSD^{108 109}
- Windows 10
 - Windows CE

Fuente: https://es.wikipedia.org/wiki/Raspberry_Pi

Los sistemas apoyados en Linux pueden ejecutar compiladores e intérpretes de alto nivel como python el cual a su vez soporta todas las librerías estándar del mismo.

En cuanto al hardware la tabla 3 muestra las especificaciones de sus versiones.

Tabla 2 características de la Raspberry pi.

	RPI Model A	RPI Model A+	RPI Model B	RPI Model B+	RPI 2 Model B
SoC	BROADCOM BCM2835	BROADCOM BCM2835	BROADCOM BCM2835	BROADCOM BCM2835	BROADCOM BCM2836
CPU	ARM11 ARMV6 700 MHZ.	ARM11 ARMV6 700 MHZ.	ARM11 ARMV6 700 MHZ.	ARM11 ARMV6 700 MHZ.	ARM11 ARMV7 ARM CORTEX- A7 4 NÚCLEOS 900 MHZ.
GPU	BROADCOM VIDEOCORE IV 250 MHZ. OPENGL ES 2.0	BROADCOM VIDEOCORE IV 250 MHZ. OPENGL ES 2.0	BROADCOM VIDEOCORE IV 250 MHZ. OPENGL ES 2.0	BROADCOM VIDEOCORE IV 250 MHZ. OPENGL ES 2.0	BROADCOM VIDEOCORE IV 250 MHZ. OPENGL ES 2.0
Memoria RAM	256 MB LPDDR SDRAM 400 MHZ.	256 MB LPDDR SDRAM 400 MHZ.	512 MB LPDDR SDRAM 400 MHZ.	512 MB LPDDR SDRAM 400 MHZ.	1 GB LPDDR2 SDRAM 450 MHZ.
Puertos USB	1.0	1.0	1.0	2.0	2.0
GPIO	26 PINES	40 PINES	26 PINES	40 PINES	40 PINES
Vídeo	HDMI 1.4 1920X1200	HDMI 1.4 1920X1200	HDMI 1.4 1920X1200	HDMI 1.4 1920X1200	HDMI 1.4 1920X1200

Almacenamiento	SD	SD	SD	SD	SD
Ethernet 10/100MBPS	NO	NO	SI	SI	SI
Tamaño	85,60X56,5 MM	65X56,5 MM.	85,60X56,5 MM	85,60X56,5 MM	85,60X56,5 MM
Peso en g.	45	23	45	45	45
Precio	29,95€	29,95€	34,95€	34,95€	39,95€

Fuente: https://es.wikipedia.org/wiki/Raspberry_Pi

Además las plataformas raspberry pi, cuentan con un puerto GPIO de propósito general, con el cual es posible leer y escribir datos del exterior, enfocado en el internet de las cosas, para conectar sensores o actuadores tales como motores dc, servomotores etc.

La figura 6. Muestra el listado de los pines de io soportados.

Figura 6. GPIO de la raspberry pi

GPIO Numbers		
Raspberry Pi Model B Rev 1 P1 GPIO Header Pin No. 3.3V 1 2 5V GPIO0 3 4 5V GPIO1 5 6 GND GPIO4 7 8 GPIO14 GND 9 10 GPIO15 GPIO17 11 12 GPIO18 GPIO21 13 14 GND GPIO22 15 16 GPIO23 3.3V 17 18 GPIO24 GPIO10 19 20 GND GPIO9 21 22 GPIO25 GPIO11 23 24 GPIO6 GND 25 26 GPIO7	Raspberry Pi Model A/B Rev 2 P1 GPIO Header Pin No. 3.3V 1 2 5V GPIO2 3 4 5V GPIO3 5 6 GND GPIO4 7 8 GPIO14 GND 9 10 GPIO15 GPIO17 11 12 GPIO18 GPIO27 13 14 GND GPIO22 15 16 GPIO23 3.3V 17 18 GPIO24 GPIO10 19 20 GND GPIO9 21 22 GPIO25 GPIO11 23 24 GPIO6 GND 25 26 GPIO7	Raspberry Pi Model A+/B+/Pi 2 B+ J8 GPIO Header Pin No. 3.3V 1 2 5V GPIO2 3 4 5V GPIO3 5 6 GND GPIO4 7 8 GPIO14 GND 9 10 GPIO15 GPIO17 11 12 GPIO18 GPIO27 13 14 GND GPIO22 15 16 GPIO23 3.3V 17 18 GPIO24 GPIO10 19 20 GND GPIO9 21 22 GPIO25 GPIO11 23 24 GPIO6 GND 25 26 GPIO7 DNC 27 28 DNC GPIO5 29 30 GND GPIO6 31 32 GPIO12 GPIO13 33 34 GND GPIO19 35 36 GPIO15 GPIO26 37 38 GPIO20 GND 39 40 GPIO21
Key Power + UART GND SPI I ² C GPIO		

Fuente: https://es.wikipedia.org/wiki/Raspberry_Pi

Figura 7. Raspberry Pi

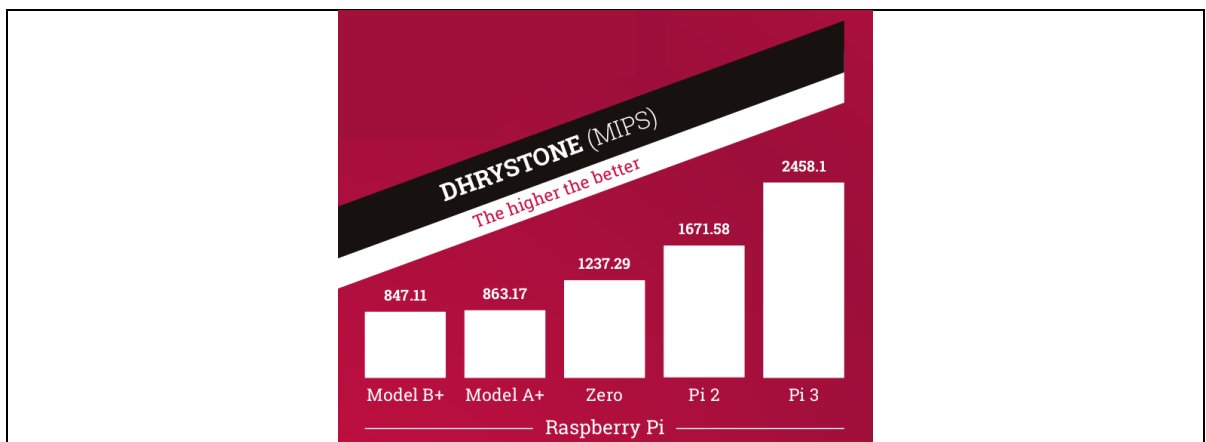


Fuente El Autor

Otro punto muy importante es su capacidad para ejecutar aplicaciones con un gran consumo de procesador. Las raspberry pi cuenta con alrededor de 2400 MIPS, capacidad que le permite ejecutar aplicaciones tipo servidor y de diferentes enfoques.

La figura 8, muestra la medida del benchmark Dhrystone para las raspberry pi. Y la tabla 3 muestra los desempeños de otros procesadores empleados en computadores de escritorio.

Figura 8. Desempeño de las diferentes versiones de las Raspberry pi.



Fuente: https://es.wikipedia.org/wiki/Raspberry_Pi

Tabla 3. Desempeño de procesadores de equipos de escritorio.

CPU	Dhry1		Dhry2		MIPS
	Opt	NoOpt	Opt	NoOpt	
	VAX	VAX	VAX	VAX	
	MHz	MIPS	MIPS	MIPS	
AMD 80386	40	17.5	4.32	13.7	4.53
IBM 486D2	50	26.6	7.89	22.4	7.89
80486 DX2	66	45.1	12.0	35.3	12.4
IBM 486BL	100	53.9	12.0	40.9	11.8
AMD 5X86	133	84.5	9.37	84.5	9.42
Pentium	75	112	19.3	87.1	18.9
Cyrix P150	120	175	27.9	160	28.3
Pentium	100	169	31.8	122	32.2
Cyrix PP166	133	219	38.4	180	39.8
IBM 6x86	150	234	44.1	188	43.9
Pentium	133	239	38.3	181	39.0
Pentium	166	270	43.6	189	43.9
Cyrix PR233	188	286	46.4	232	45.8
Pentium	200	353	47.4	269	48.1
Pentium MMX	200	352	51.4	276	51.0
AMD K6	200	349	43.1	289	43.3
Pentium Pro	200	373	92.4	312	91.9
Celeron A	300	553	133	484	136
Pentium II	300	544	132	477	136
AMD K62	500	778	77.8	606	76.8
AMD K63	450	804	76.3	645	77.4
Pentium II	450	813	199	713	204
Celeron A	450	828	198	720	202

Pentium III	450	846	197	722	203
Pentium III	600	1105	263	959	270
Athlon	600	1316	321	942	316
Duron	600	1382	350	999	349
Pentium III	1000	1858	461	1595	465
PIII Tualatin	1200	2205	546	1907	571
Pentium 4	1700	2262	239	1843	242
Athlon Tbird	1000	2282	634	1659	602
Duron	1000	2288	576	1674	587
Celeron M	1295	2440	640	2273	645
Atom	1600	2462	717	1828	728
Atom	1666	2600	772	1948	780
P4 Xeon	2200	3028	300	2265	309
Atom Z8300	1840	3203	904	2686	927
Athlon 4	1600	3707	956	2830	1004
Pentium M	1862	4082	954	3933	975
Ath4 Barton	1800	4181	1061	3172	1099
Pentium 4E	3000	4379	566	3553	566
Athlon XP	2080	4826	1228	3700	1312
Turion 64 M	1900	4972	1186	3742	1150
Pentium 4	3066	5052	432	4012	434
Opteron	1991	5077	1268	3985	1223
Core 2 Duo M	1830	5379	892	4952	966
Athlon XP	2338	5433	1400	4160	1482
Athlon 64	2150	5658	1312	4288	1355
Pentium 4	3678	5787	511	4227	480
Athlon 64	2211	5798	1348	4462	1312
Celeron C2 M	2000	5804	932	5275	1050
Core 2 Duo 1 CP 2400	7145	1198	6446	1251	

Core i5 2467M	@@@@	8338	1183	4752	1148
Phenom II 1 CP	3000	9462	2250	7615	2253
Core i7 930	****	9826	1662	8684	1661
Core i7 860	####	10094	1789	9978	1847
Core i7 3930K	&&&&	13871	1960	11197	1972
Core i7 4820K	\$\$\$1	14136	1958	11867	1981
Core i7 4820K	\$\$\$2	14776	2006	11978	2014
Core i7 3930K	OC	17269	2444	13877	2432

Fuente: https://es.wikipedia.org/wiki/Raspberry_Pi

Lo cual demuestra que las raspberry en aplicaciones de 32 bits tienen un desempeño muy similar a un procesador Pentium 4 o Atom.

El último sistema operativo desarrollado y estabilizado para la Rpi es Raspbian Jessie de 32 o de 64 bits. Este sistema operativo está basado en debían y se encuentra optimizado para el conjunto de instrucciones de los procesadores ARM. Este es más que un sistema operativo puro ya que cuenta con más de 35 000 paquetes, software pre compilado y un formato de instalación fácil para la rpi.

El objetivo de esta plataforma hardware fue el de llegar a cada alumno de educación básica en el mundo con el proyecto de computadores de US \$ 100. Para permitir el acercamiento de los niños a la programación y ejecutar código en lenguajes como Scratch o Kids Ruby y otros de aprendizaje más sencillo, pero ya más profesionales como Python. Además, incluye aplicaciones matemáticas para trabajar con álgebra y otras tareas como Matemática.

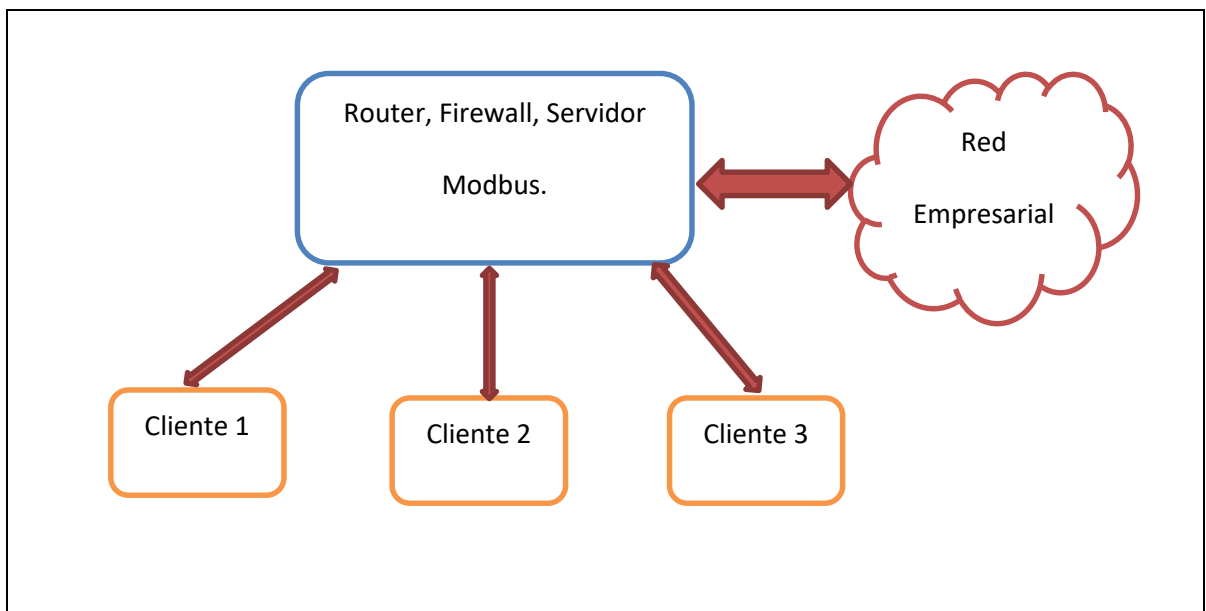
La Raspberry Pi es, en resumen, un dispositivo que permite aprovechar todas las ventajas de un computador de escritorio por un muy bajo costo y un muy reducido tamaño.

Esto lo convierte en la solución ideal para algunos proyectos de ingeniería. Pero no solo es aplicable a proyectos tecnológicamente avanzados.

6.5 IMPLEMENTACIÓN DEL FIREWALL INDUSTRIAL

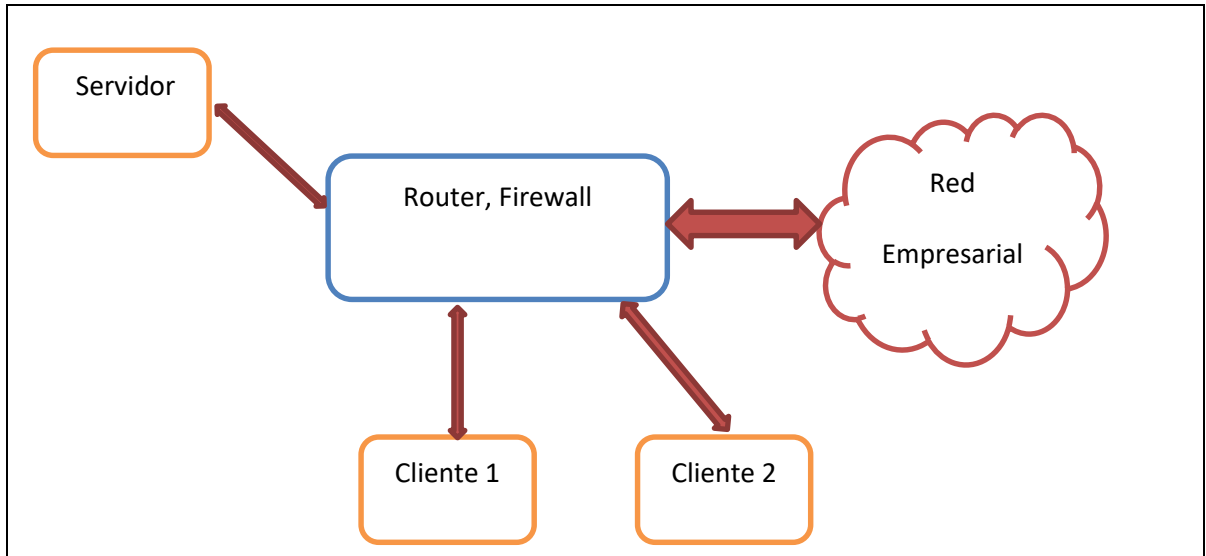
El objetivo es explicar la metodología seguida para el desarrollo de un prototipo mínimo viable de un firewall industrial, empleando el protocolo modbus, para esto se han diseñado dos topologías.

Figura 9. Topología 1. Firewall Industrial.



Fuente El Autor

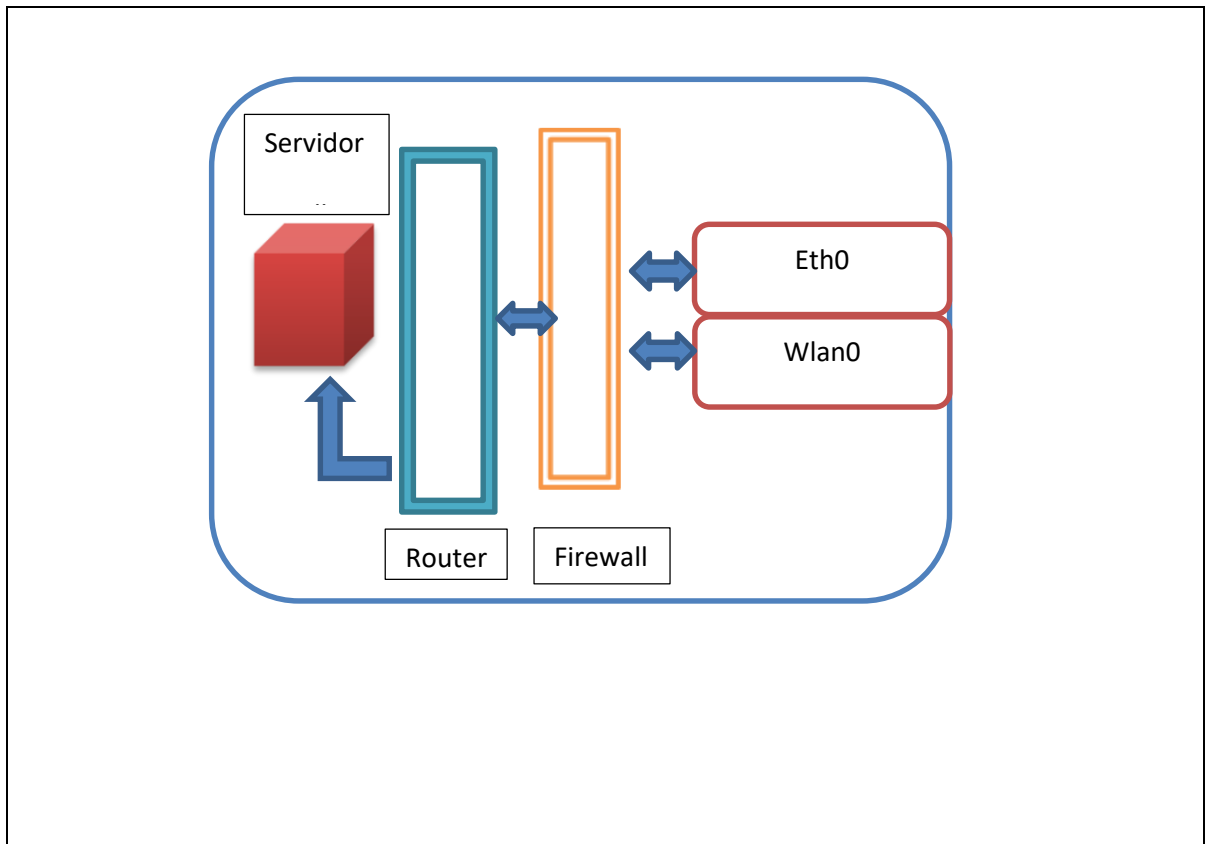
Figura 10. Topología 2. Firewall Industrial.



Fuente El Autor

Para esto, en las dos topologías la raspberry debe soportar un comportamiento como router, la configuración del firewall y el servidor modbus.

Figura 11. Diagrama de interconexiones del software al interior de la Rpi.



Fuente El Autor

Se debe resaltar que las configuraciones a realizar se apoyan en capas de software diferentes, el primer paso es la configuración de la Rpi como un router para esto se adiciona una tarjeta usb nano encore.

Configuración de Rpi como router TCP:

Sobre una instalación de Raspbian jessie se procede a la actualización del conjunto de paquetes con:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

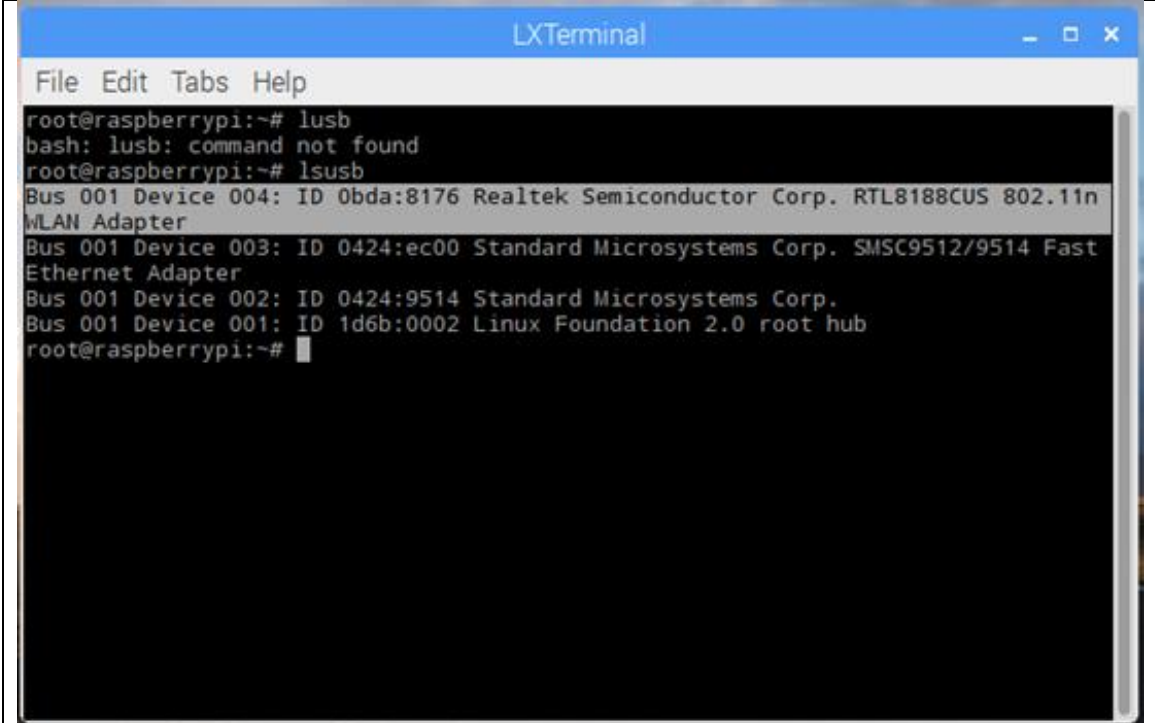
```
sudo apt-get dist-upgrade
```

En la topología planteada el Puerto Ethernet cableado va a realizar la conexión hacia la red externa y una tarjeta usb wifi se asocia con la red de clientes usb.

Entonces se instala la tarjeta y se verifica su detección en el sistema operativo con

```
lsusb
```

Figura 12. Comando lsusb, detección de la tarjeta wlan0.



```
LXTerminal
File Edit Tabs Help
root@raspberrypi:~# lsusb
bash: lsusb: command not found
root@raspberrypi:~# lsusb
Bus 001 Device 004: ID 0bda:8176 Realtek Semiconductor Corp. RTL8188CUS 802.11n WLAN Adapter
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
root@raspberrypi:~# █
```

Fuente El Autor

Y se corrobora que la tarjeta soporta el modo Access point con:

```
iw list.
```

Ambos comandos deben detectar a la tarjeta wlan0.

La mayor parte del software necesario viene por defecto instalado en Raspbian,

El servicio de creación de puntos de acceso y de creación del servidor dhcp se instalan de forma manual.

```
sudo apt-get install isc-dhcp-server hostapd
```

Una vez se llega a este punto se puede reiniciar el Raspberry para empezar con la configuración.

Posteriormente se configura el servidor dhcp, editando

```
sudo nano /etc/dhcp/dhcpd.conf
```

Las siguientes líneas están por defecto sin comentar, se comentan con el símbolo # delante de manera que dejen de estar habilitadas quedando de la siguiente manera:

```
#option domain-name "example.org";  
#option domain-name-servers ns1.example.org, ns2.example.org
```

Se descomenta la línea authoritative del mismo archivo.

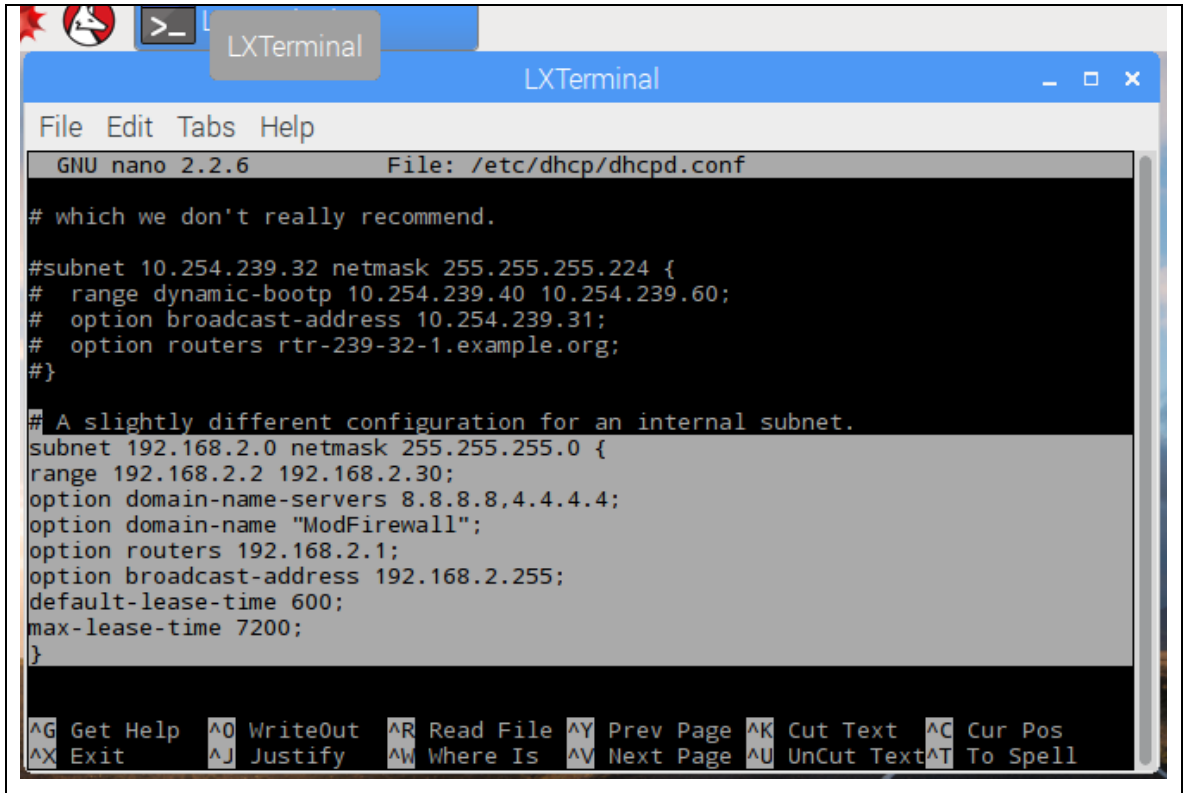
Dentro del mismo archivo se configura la red que creara el router:

```
subnet 192.168.2.0 netmask 255.255.255.0 {  
range 192.168.2.2 192.168.2.30;  
option broadcast-address 192.168.2.255;  
option routers 192.168.2.1  
default-lease-time 600;  
max-lease-time 7200;  
option domain-name "local  
option domain-name-servers 8.8.8.8, 8.8.4.4;30
```

La figura 13 muestra dicha configuración.

³⁰ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

Figura 13. Configuración del servidor DHCP.



```
GNU nano 2.2.6 File: /etc/dhcp/dhcpd.conf

# which we don't really recommend.

#subnet 10.254.239.32 netmask 255.255.255.224 {
# range dynamic-bootp 10.254.239.40 10.254.239.60;
# option broadcast-address 10.254.239.31;
# option routers rtr-239-32-1.example.org;
#}

## A slightly different configuration for an internal subnet.
subnet 192.168.2.0 netmask 255.255.255.0 {
range 192.168.2.2 192.168.2.30;
option domain-name-servers 8.8.8.8,4.4.4.4;
option domain-name "ModFirewall";
option routers 192.168.2.1;
option broadcast-address 192.168.2.255;
default-lease-time 600;
max-lease-time 7200;
}

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Fuente El Autor

Se edita posteriormente el nano `/etc/default/isc-dhcp-server` en el cual se adiciona o descomenta la línea

```
INTERFACES="wlan0"
```

Lo primero que se hace será desconectar la tarjeta WI-FI. Para esto se escribe:

```
% sudo ifdown wlan0
```

A continuación se abre el fichero "interfaces":

```
% sudo cp /etc/network/interfaces /etc/network/interfaces.orig
```

```
% sudo nano /etc/network/interfaces
```

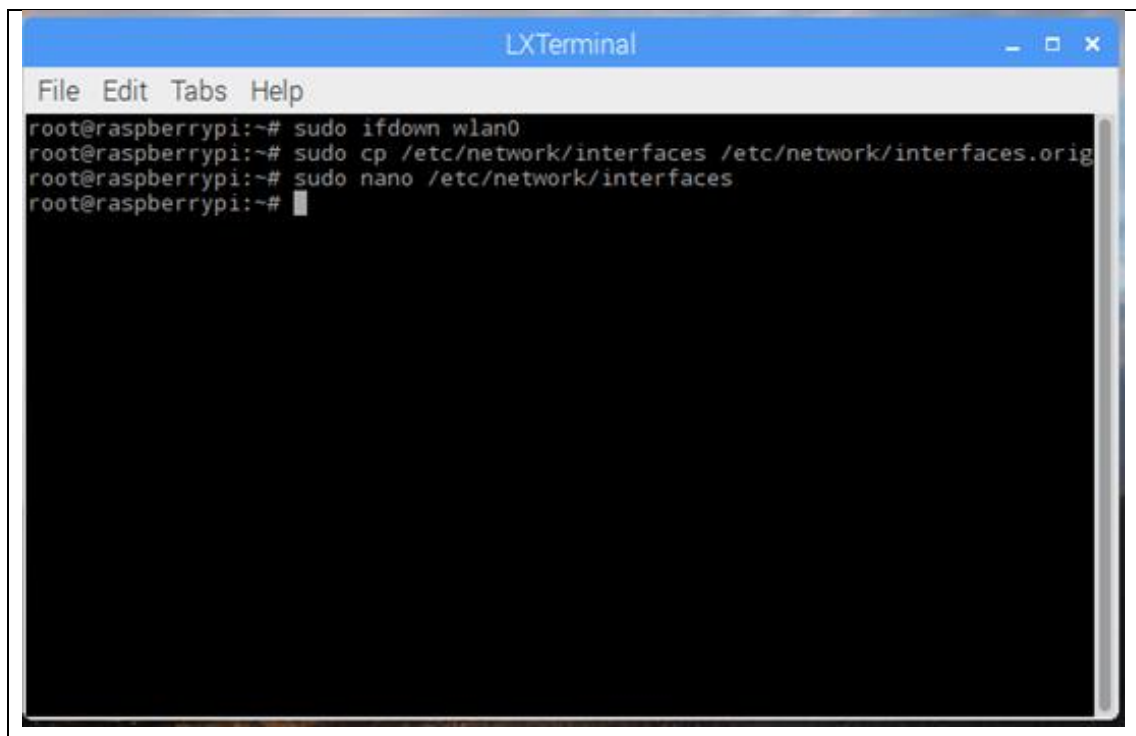
Y se configura de la siguiente manera:

```
auto lo
iface lo inet loopback
iface eth0 inet dhcp
allow hotplug wlan0
iface wlan0 inet static
address 192.168.2.1
netmask 255.255.255.0
```

Se comenta o se borra las demás líneas. Se guardan los cambios y se cierra el documento. El archivo interfaces se puede observar en la figura 14. Para aplicar los cambios al momento debemos teclear:

```
% sudo ifconfig wlan0 192.168.2.131
```

Figura 14. Resultado de los comandos de modificación del archivo interfaces.

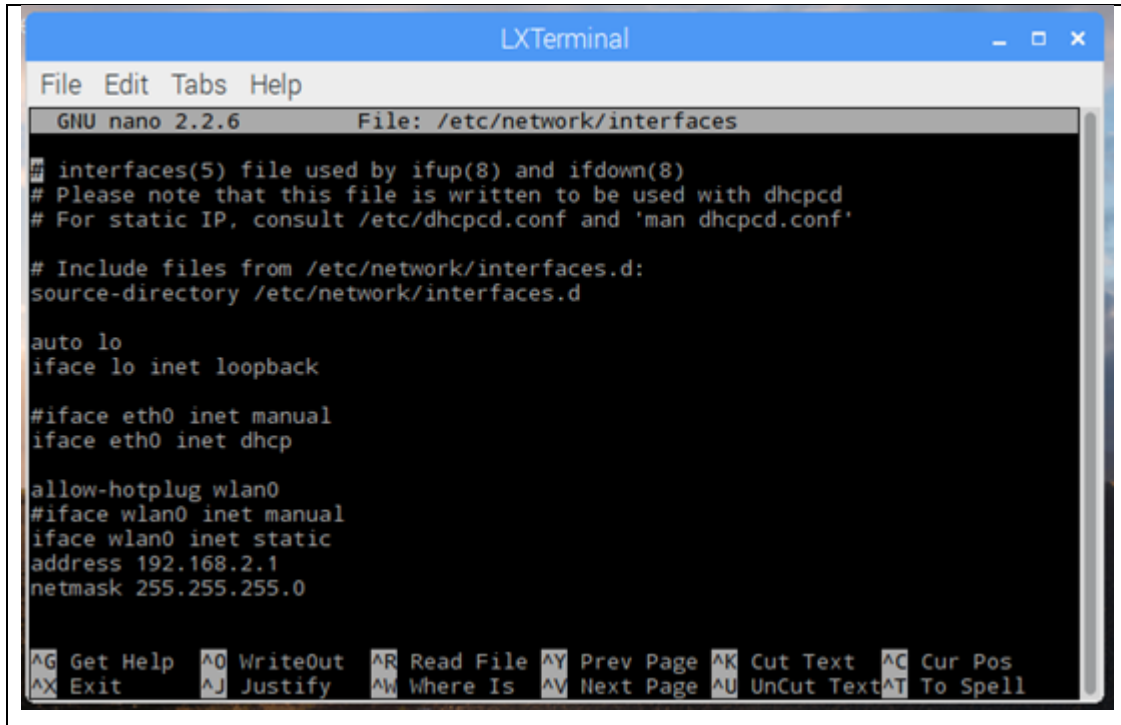


```
LXTerminal
File Edit Tabs Help
root@raspberrypi:~# sudo ifdown wlan0
root@raspberrypi:~# sudo cp /etc/network/interfaces /etc/network/interfaces.orig
root@raspberrypi:~# sudo nano /etc/network/interfaces
root@raspberrypi:~# █
```

Fuente El Autor

³¹ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

Figura 15. Resultado de los comandos de modificación del archivo interfaces.



```
GNU nano 2.2.6 File: /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
# Please note that this file is written to be used with dhcpd
# For static IP, consult /etc/dhcpd.conf and 'man dhcpd.conf'

# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

#iface eth0 inet manual
iface eth0 inet dhcp

allow-hotplug wlan0
#iface wlan0 inet manual
iface wlan0 inet static
address 192.168.2.1
netmask 255.255.255.0

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Fuente El Autor

Para configurar el punto de acceso se edita otro fichero tecleando:

```
% sudo cp /etc/hostapd/hostapd.conf /etc/hostapd/hostapd.conf.orig
% sudo nano /etc/hostapd/hostapd.conf
```

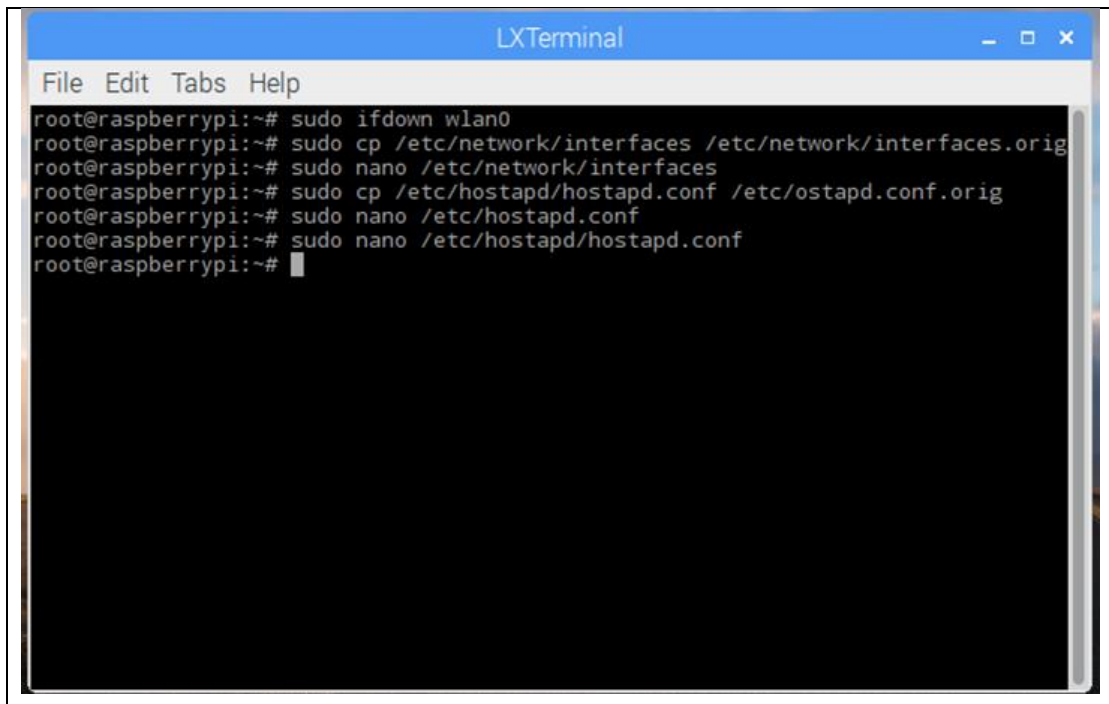
Y en este archivo se pega:

```
interface=wlan0
ssid=RaspiAP
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
```

```
wpa_passphrase=password  
wpa_key_mgmt=WPA-PSK  
wpa_pairwise=TKIP  
rsn_pairwise=CCMP32
```

La figura 16, muestra el archivo hostapd que soporta el punto de acceso y en el cual se evidencian los nombre del punto de acceso, el tipo de seguridad y el password.

Figura 16. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.

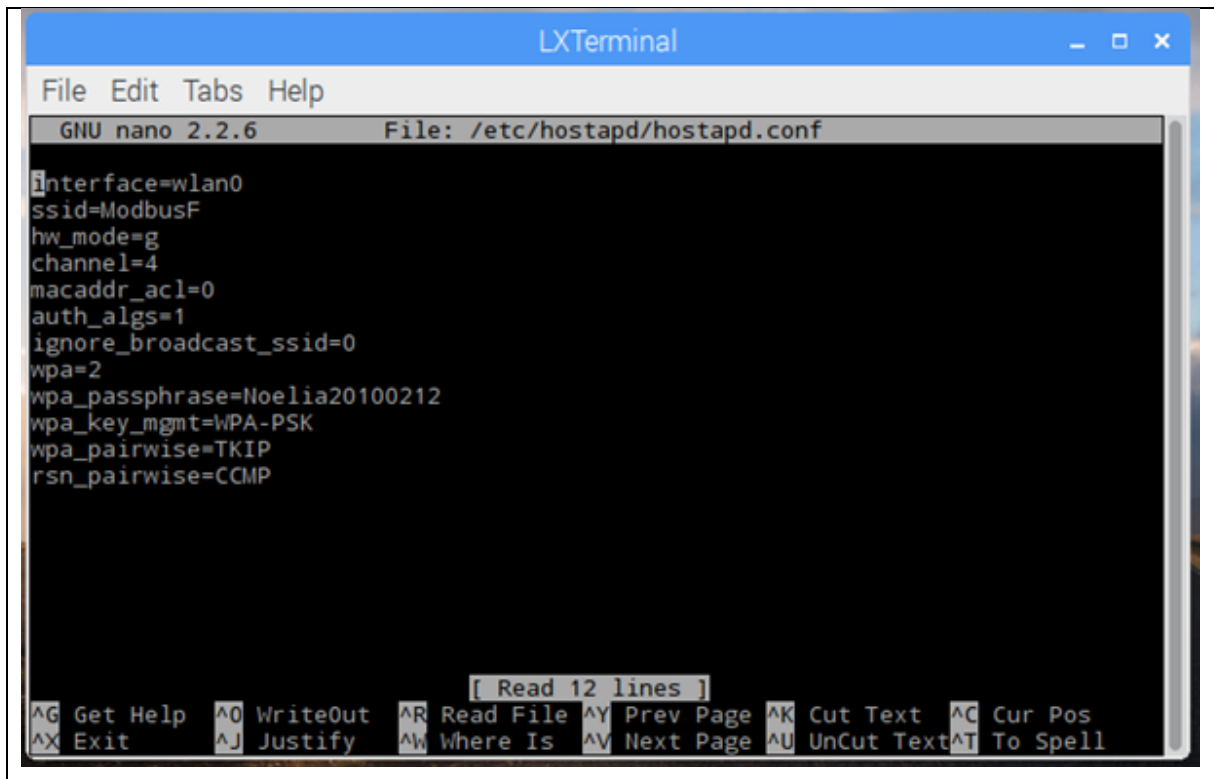


```
LXTerminal  
File Edit Tabs Help  
root@raspberrypi:~# sudo ifdown wlan0  
root@raspberrypi:~# sudo cp /etc/network/interfaces /etc/network/interfaces.orig  
root@raspberrypi:~# sudo nano /etc/network/interfaces  
root@raspberrypi:~# sudo cp /etc/hostapd/hostapd.conf /etc/ostapd.conf.orig  
root@raspberrypi:~# sudo nano /etc/hostapd.conf  
root@raspberrypi:~# sudo nano /etc/hostapd/hostapd.conf  
root@raspberrypi:~# █
```

Fuente El Autor

³² Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

Figura 17. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.



```
LXTerminal
File Edit Tabs Help
GNU nano 2.2.6 File: /etc/hostapd/hostapd.conf
interface=wlan0
ssid=ModbusF
hw_mode=g
channel=4
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=Noelia20100212
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
[ Read 12 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Fuente El Autor

Se puede cambiar el SSID por el nombre que se quiera dar a la red como el canal en channel y la wpa_passphrase con la contraseña, en texto plano, que se quiera utilizar para conectarse.

Para finalizar con la configuración se abre un nuevo archivo de configuración tecleando:

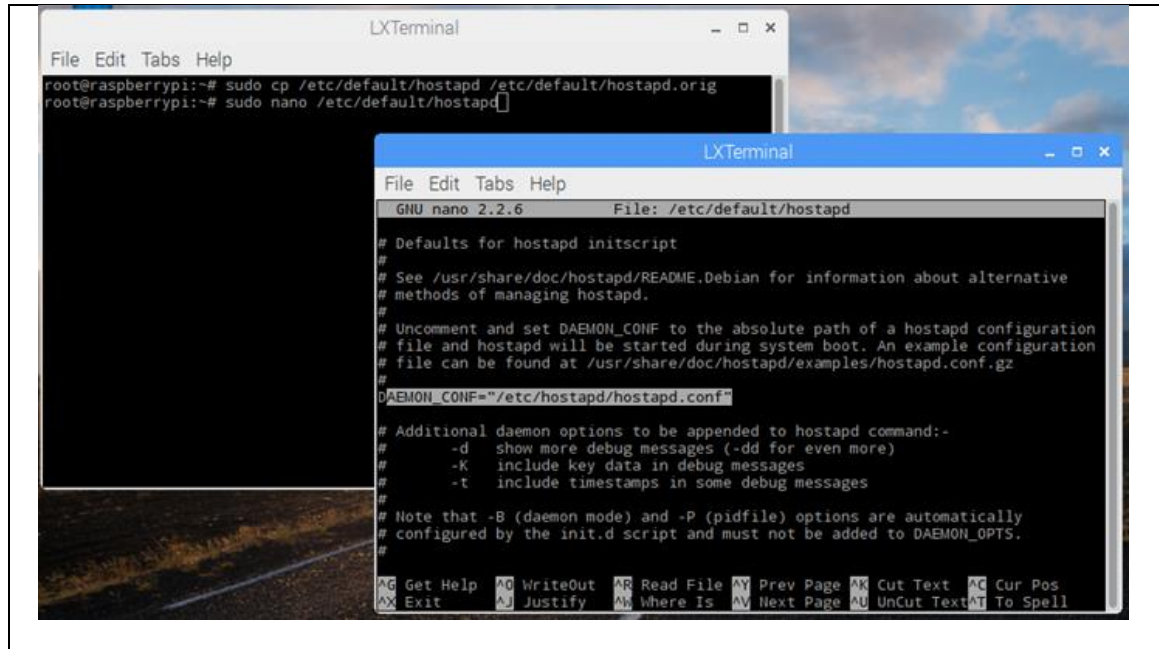
```
% sudo cp /etc/default/hostapd /etc/default/hostapd.orig
```

```
% sudo nano /etc/default/hostapd
```

Se descomenta y se cambia la línea `#DAEMON_CONF=""` por:

DAEMON_CONF="/etc/hostapd/hostapd.conf"³³

Figura 18. Resultado de los comandos de modificación del archivo hostapd en la configuración del punto de acceso de la red modbus.



Fuente El Autor

Se guarda el archivo y se configura el SO para que reinicie todos los servicios de forma automática, para esto se debe:

Configurar el reenvío de paquetes para que Raspberry Pi transmita las tramas desde su núcleo al enrutador. Por lo tanto, se realiza la siguiente configuración:

Se edita el archivo sysctl:

```
% sudo cp /etc/sysctl.conf /etc/sysctl.conf.orig
```

```
% sudo nano /etc/sysctl.conf
```

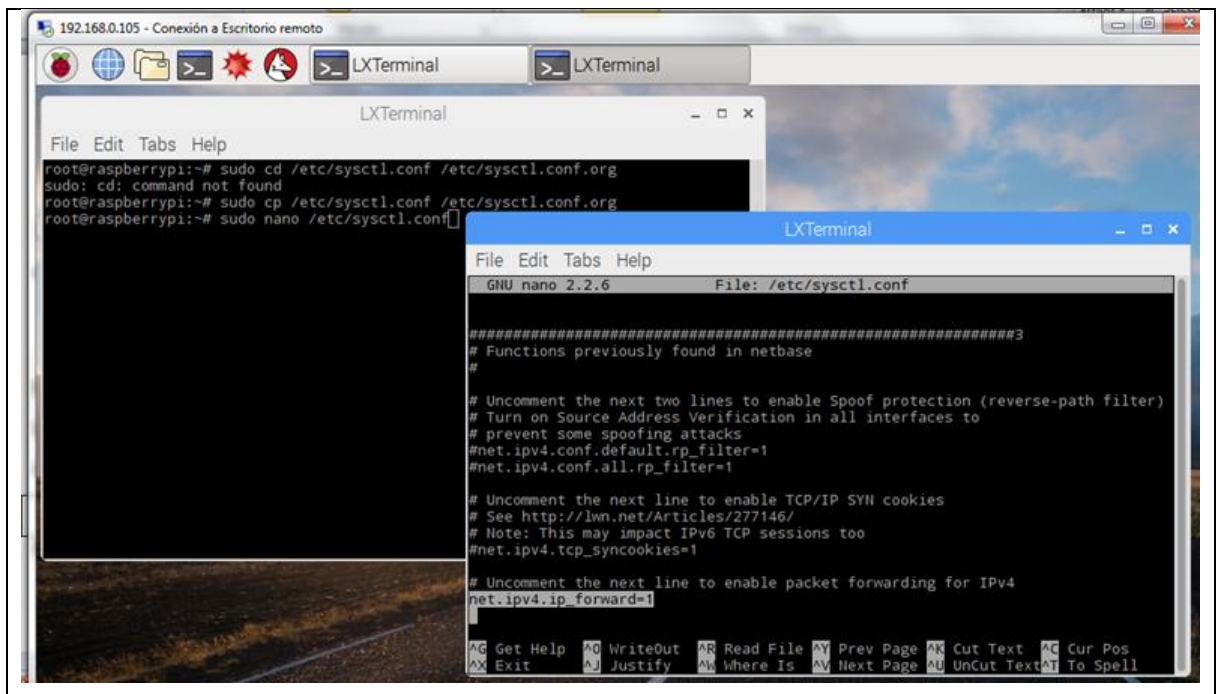
Se busca la línea "# net.ipv4.ip_forward=1" y se descomenta, quedando:

³³ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

```
net.ipv4.ip_forward=134
```

La figura 19 evidencia la configuración para el reenvío de paquetes.

Figura 19. Resultado de los comandos de modificación del archivo sysctl.conf en la configuración del reenvío de paquetes de la red modbuF hacia el router de internet.



Fuente El Autor

Se guarda y se cierra el documento. Para que se tomen los cambios se digita:

```
% sudo sysctl -p /etc/sysctl.conf
```

Se habilita la NAT digitando:

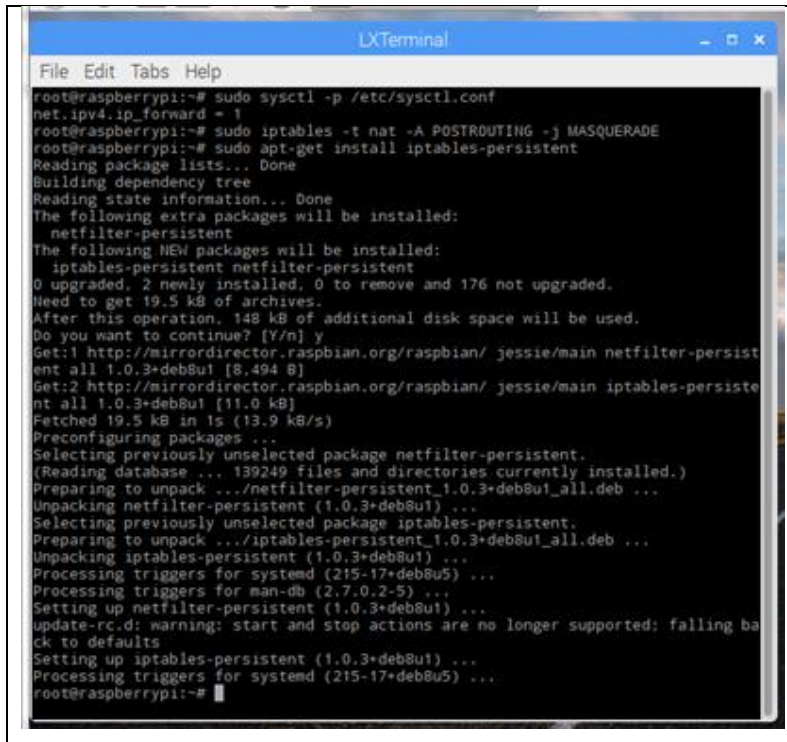
```
% sudo iptables -t nat -A POSTROUTING -j MASQUERADE35
```

³⁴ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

³⁵ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberry-pi/manual-para-configurar-raspberry-pi-como-un-router-wi-fi/>

posteriormente se guardan las reglas, para garantizar su persistencia instalando
% sudo apt-get install iptables-persistent.

Figura 20. Resultado de los comandos de activación de NAT y de persistencia de las reglas iptables.



```
LXTerminal
File Edit Tabs Help
root@raspberrypi:~# sudo sysctl -p /etc/sysctl.conf
net.ipv4.ip_forward = 1
root@raspberrypi:~# sudo iptables -t nat -A POSTROUTING -j MASQUERADE
root@raspberrypi:~# sudo apt-get install iptables-persistent
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  netfilter-persistent
The following NEW packages will be installed:
  iptables-persistent netfilter-persistent
0 upgraded, 2 newly installed, 0 to remove and 176 not upgraded.
Need to get 19.5 kB of archives.
After this operation, 148 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://mirrordirector.raspbian.org/raspbian/ jessie/main netfilter-persistent all 1.0.3+deb8u1 [8.494 B]
Get:2 http://mirrordirector.raspbian.org/raspbian/ jessie/main iptables-persistent all 1.0.3+deb8u1 [11.0 kB]
Fetched 19.5 kB in 1s (13.9 kB/s)
Preconfiguring packages ...
Selecting previously unselected package netfilter-persistent.
(Reading database ... 139249 files and directories currently installed.)
Preparing to unpack .../netfilter-persistent_1.0.3+deb8u1_all.deb ...
Unpacking netfilter-persistent (1.0.3+deb8u1) ...
Selecting previously unselected package iptables-persistent.
Preparing to unpack .../iptables-persistent_1.0.3+deb8u1_all.deb ...
Unpacking iptables-persistent (1.0.3+deb8u1) ...
Processing triggers for systemd (215-17+deb8u5) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up netfilter-persistent (1.0.3+deb8u1) ...
update-rc.d: warning: start and stop actions are no longer supported; falling back to defaults
Setting up iptables-persistent (1.0.3+deb8u1) ...
Processing triggers for systemd (215-17+deb8u5) ...
root@raspberrypi:~#
```

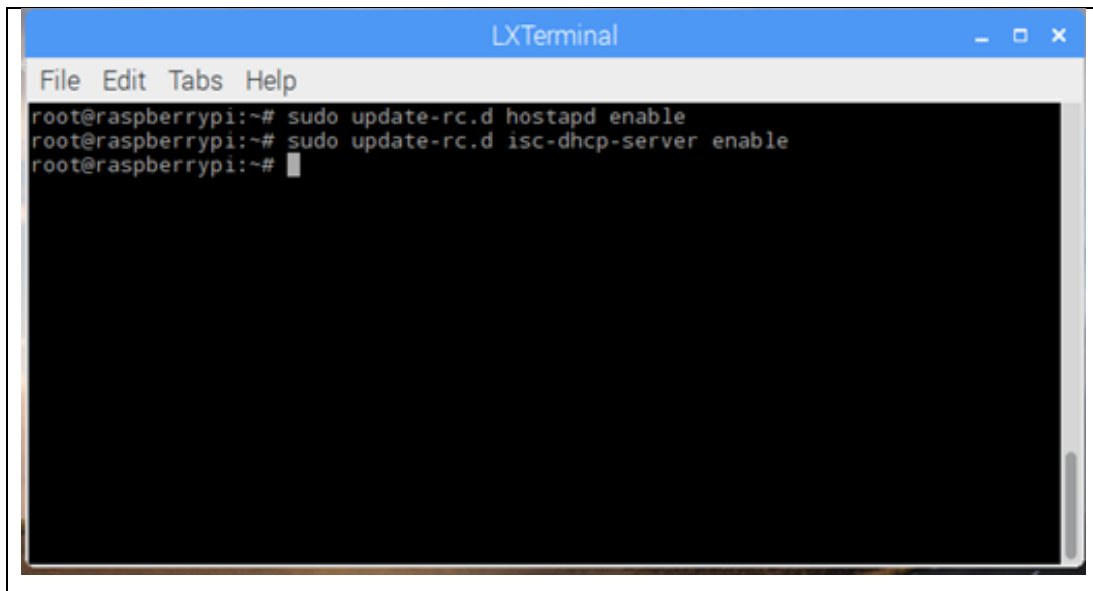
Fuente El Autor

Luego se configuran los servicios para que se inicien al iniciar el SO.

%sudo update-rc.d hostapd enable

%sudo update-rc.d isc-dhcp-server enable

Figura 21. Resultado de los comandos de activación de servicios durante el arranque del sistema



```
LXTerminal
File Edit Tabs Help
root@raspberrypi:~# sudo update-rc.d hostapd enable
root@raspberrypi:~# sudo update-rc.d isc-dhcp-server enable
root@raspberrypi:~#
```

Fuente El Autor

Antes de finalizar, hay un fichero de WPA supplicant se mueve a la ruta:

```
sudo mv /usr/share/dbus-1/system-
services/fi.epitest.hostap.WPA supplicant.service /home/pi36
```

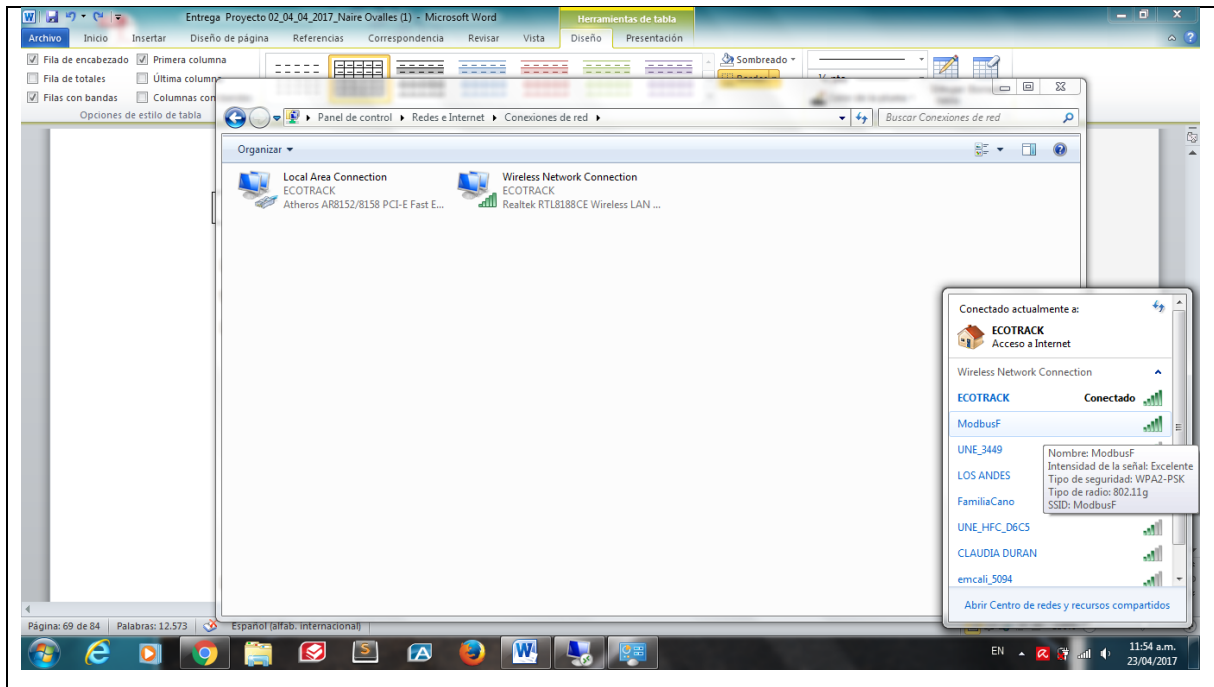
Resaltando, el nombre del router, los rangos de direcciones, las máscaras de subred, los DNS.

De esta forma se tiene en ejecución un AP con el nombre modbus el cual puede rutear paquetes desde la subred y hacia internet o viceversa.

La figura 20 evidencia el funcionamiento del punto de acceso ModbusF, se resaltan las características de la configuración realizadas en los pasos previos.

³⁶ Manual para configurar Raspberry PI Disponible en: <https://www.redeszone.net/raspberrypi/manual-para-configurar-raspberrypi-como-un-router-wi-fi/>

Figura 22. Evidencia del funcionamiento del AP ModbusF.



Fuente El Autor

La instalación de PYMODBUS y PYPTABLES posee las siguientes dependencias tanto del sistema operativo como del entorno python 2.7:

La instalación requiere conexión a internet y privilegios de súper usuario.

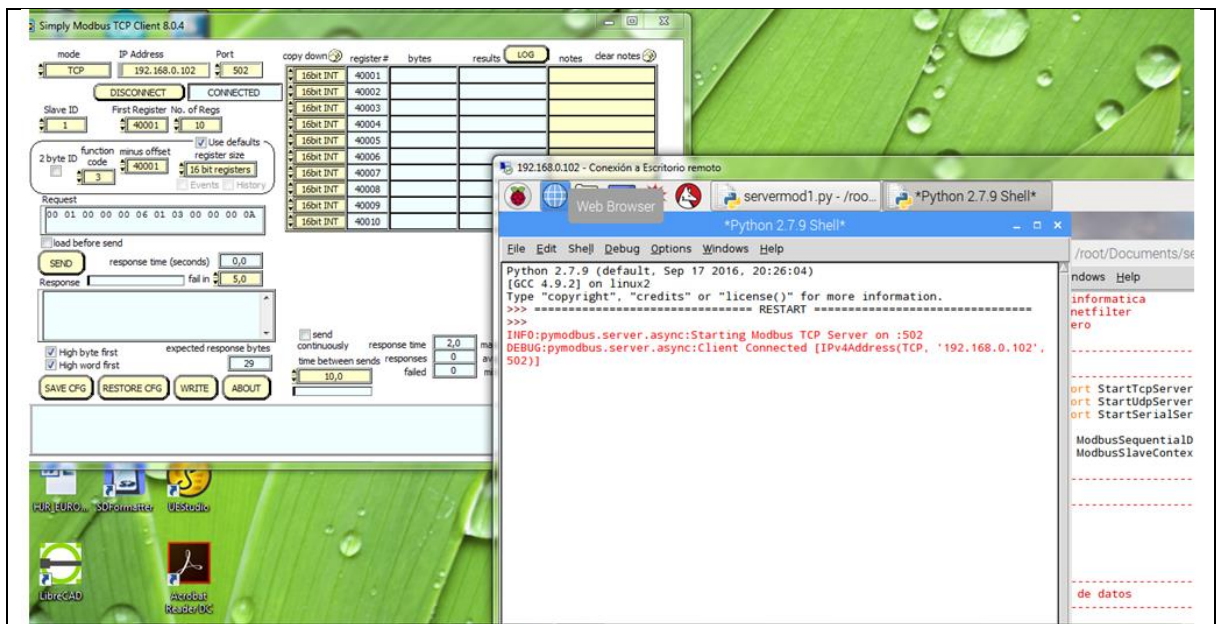
- apt-get install build-essential python-dev.
- pip install pymodbus.
- pip install netfilter.
- apt-get install libssl-dev.
- apt-get install libffi-dev.
- pip install pyptables.

La ejecución del servidor puede realizarse vía comandos con: Python pymodbus. De la misma forma los diferentes módulos de prueba de pyptables y el firewall industrial.

Pruebas del Servidor Modbus para la topología 1.

Se inicia el servidor modbus, siguiendo el orden dado por la topología 1, el objetivo específico aquí es validar la operación del servidor en conexión con un cliente local ejecutado desde el pc, el cliente local usado es simply modbus client, para Windows, el cual es una herramienta gratuita, sin limitaciones en funcionalidad, pero si con limitaciones en cuanto al número de llamados que puede realizar por sesión. La figura 23, muestra la conexión de este cliente con el servidor.

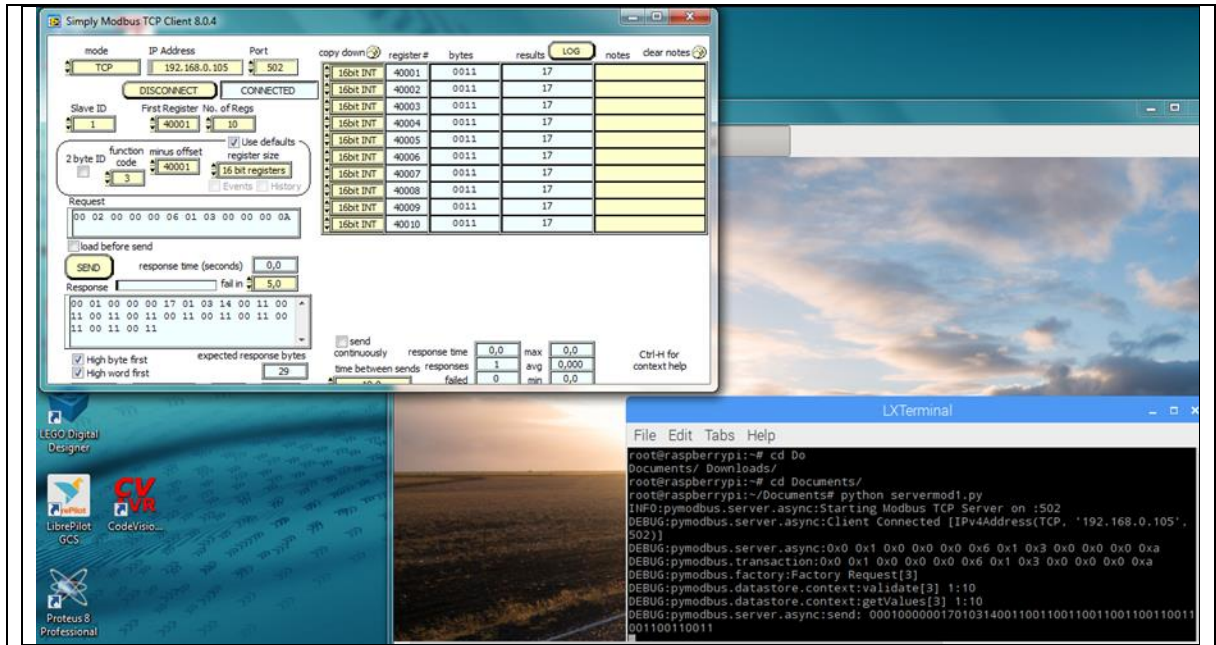
Figura 23. simply modbus client en conexión con el servidor pymodbus de la Rpi.



Fuente El Autor

La Rpi se encuentra en la dirección dhcp 192.168.0.102 y es accedida desde el pc vía el protocolo drp de Windows y mediante la herramienta xdrp para Linux. Una vez el servidor está en ejecución muestra su estado y el puerto que consume, en este caso el puerto 502. En la línea DEBUG se observa el lanzamiento de la clase Client la cual indica que se ha conectado un cliente con ip4 y con protocolo TCP. Cuando se realizan lecturas de los registros al interior del Servidor, este lo reporta, ver la figura 24.

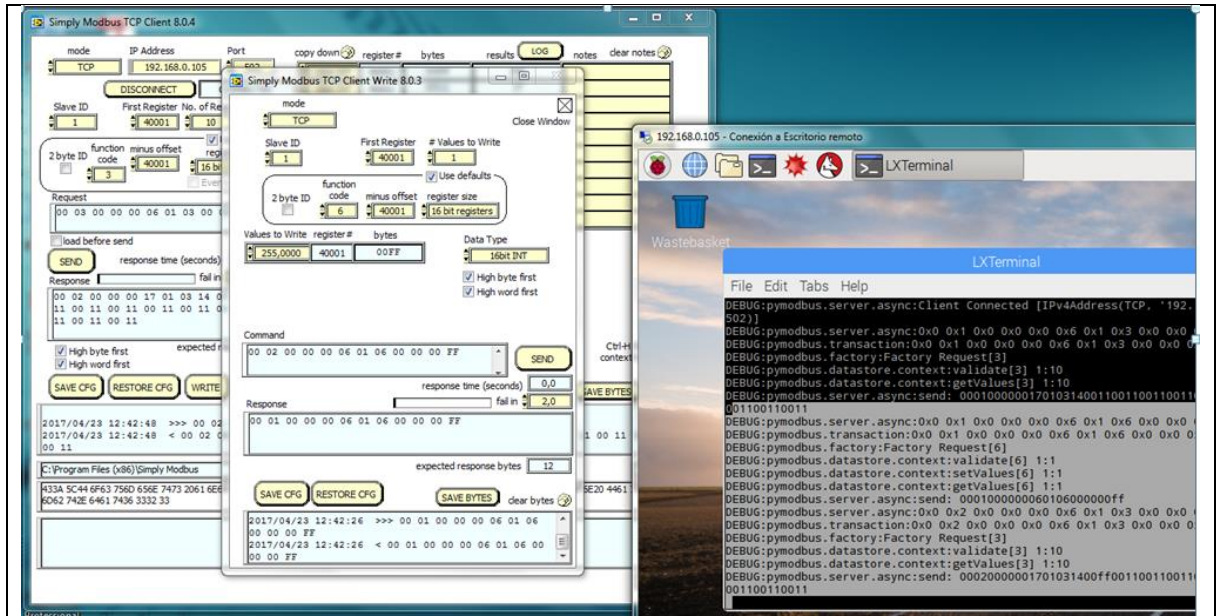
Figura 24. El cliente lee registros del servidor pymodbus en Rpi.



Fuente El Autor

En la lectura el servidor reporta con el log DEBUG, los bytes enviados, de la misma forma el cliente muestra los mismos bytes, así como la fecha y la hora. El cliente puede hacer modificaciones en los registros del servidor, según la funcionalidad de los registros. En entornos reales estos registros se asocian con entradas o salidas de dispositivos reales, como plc's o máquinas de planta, en las entradas se conectan generalmente sensores ya sea de temperatura, humedad, presión, voltaje o corriente. Mientras que en las salidas se conectan actuadores tales como motores, bombas eléctricas, maquinas lineales. Como se observa en la figura 25, en donde el cliente escribe en el registro 1 la palabra 255 (0x0FF).

Figura 25. Escritura de un byte desde el cliente y hacia el servidor.

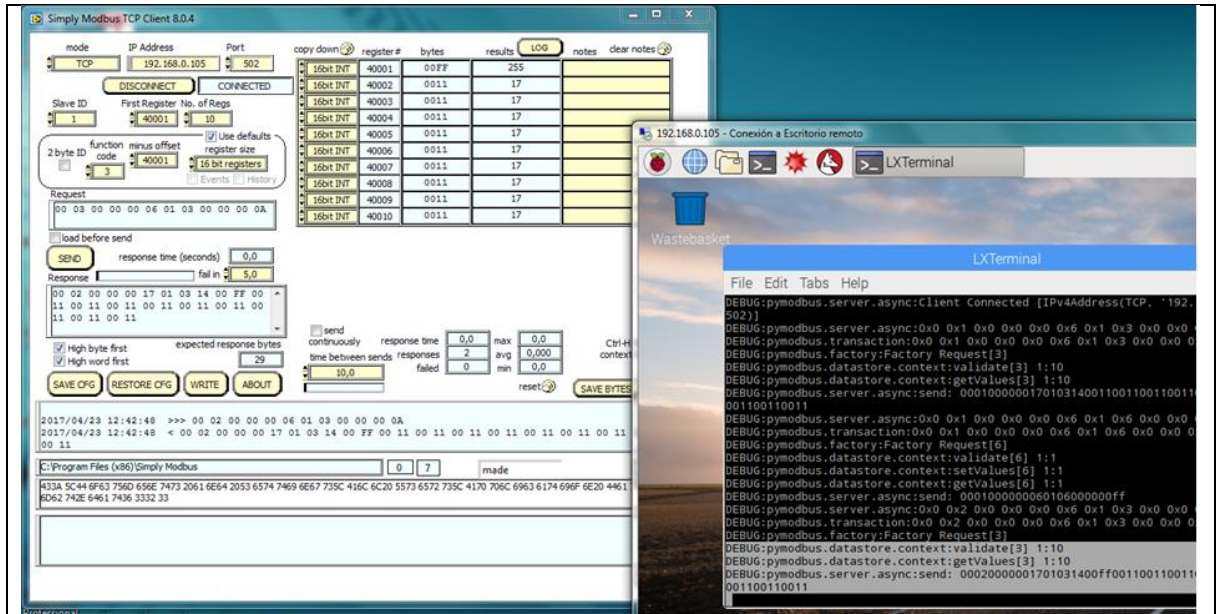


Fuente El Autor

Cuando se despliega la ventana de escritura en el cliente se da espacio para el ingreso del dato, luego para enviarlo se presiona el botón send. La ventana de la derecha muestra el histórico generado por el servidor ya en la Rpi. Allí se ve como el último dato es 0xFF que es el dato enviado por el cliente.

La validación final de la comunicación cliente servidor se observa en la figura 26, en donde el cliente lee del servidor y se observa el registro 1, que el mismo cliente modificó.

Figura 26. Dato modificado por el cliente.



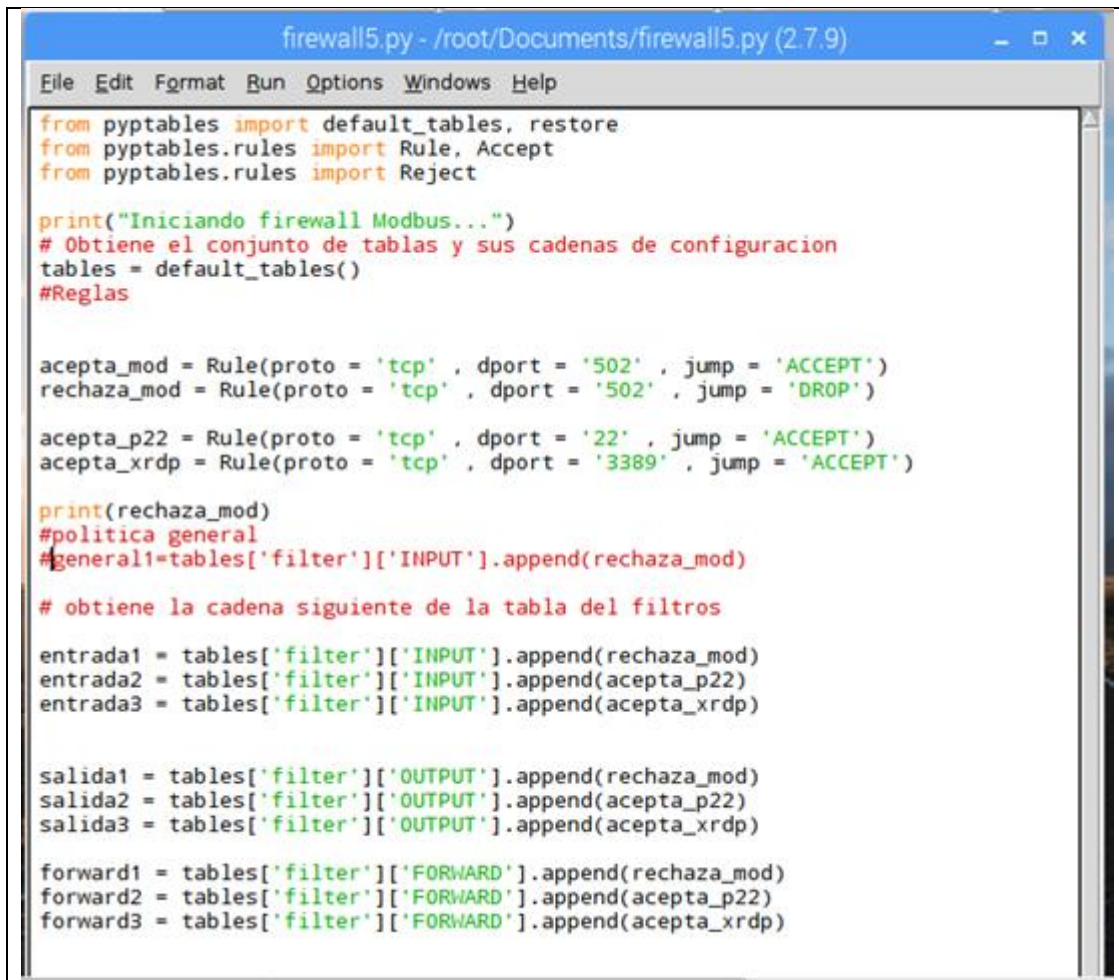
Fuente El Autor

De la misma forma se observa el histórico del servidor.

Entonces conociendo la filosofía de la comunicación, se puede comprender la relevancia que alcanza el firewall industrial, ya que en planta pueden ocurrir situaciones no permitidas, por ejemplo la activación en el servidor de un motor por parte de un cliente que no está autorizado. Dicha situación es más común de lo que parece ya que generalmente los clientes son configurados por personal de ingeniería diverso, contratistas y proveedores y en tiempo de depuración pueden ocasionar efectos secundarios en otros equipos. Allí es importante que la planta cuente con una política de seguridad, que añada un nivel de revisión más y que evite la ocurrencia de estas situaciones. También pueden ocurrir situaciones en las cuales se desea perpetrar un daño a la planta, si el protocolo es conocido, se pueden identificar las direcciones de servidores y clientes para de esta forma modificar la información de proceso y cambiar las condiciones de operación con consecuencias impredecibles, Por lo tanto la existencia de una política de seguridad al interior de la red industrial se hace relevante.

A continuación se muestran las diferentes pruebas de comunicación realizadas para configurar el firewall y el funcionamiento de la topología 1.

Figura 27. Código de implementación de firewall por filtrado de puerto y protocolo.



```
firewall5.py - /root/Documents/firewall5.py (2.7.9)
File Edit Format Run Options Windows Help

from pytables import default_tables, restore
from pytables.rules import Rule, Accept
from pytables.rules import Reject

print("Iniciando firewall Modbus...")
# Obtiene el conjunto de tablas y sus cadenas de configuracion
tables = default_tables()
#Reglas

acepta_mod = Rule(proto = 'tcp' , dport = '502' , jump = 'ACCEPT')
rechaza_mod = Rule(proto = 'tcp' , dport = '502' , jump = 'DROP')

acepta_p22 = Rule(proto = 'tcp' , dport = '22' , jump = 'ACCEPT')
acepta_xrdp = Rule(proto = 'tcp' , dport = '3389' , jump = 'ACCEPT')

print(rechaza_mod)
#politica general
#general1=tables['filter']['INPUT'].append(rechaza_mod)

# obtiene la cadena siguiente de la tabla del filtros

entrada1 = tables['filter']['INPUT'].append(rechaza_mod)
entrada2 = tables['filter']['INPUT'].append(acepta_p22)
entrada3 = tables['filter']['INPUT'].append(acepta_xrdp)

salida1 = tables['filter']['OUTPUT'].append(rechaza_mod)
salida2 = tables['filter']['OUTPUT'].append(acepta_p22)
salida3 = tables['filter']['OUTPUT'].append(acepta_xrdp)

forward1 = tables['filter']['FORWARD'].append(rechaza_mod)
forward2 = tables['filter']['FORWARD'].append(acepta_p22)
forward3 = tables['filter']['FORWARD'].append(acepta_xrdp)
```

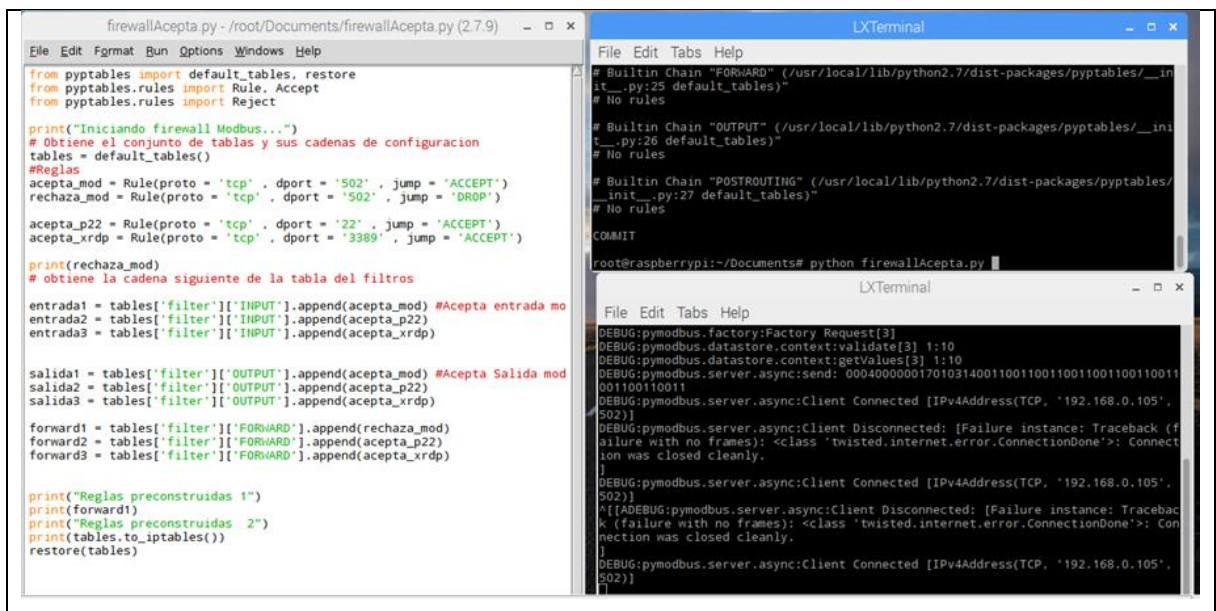
Fuente El Autor

La metodología para crear el firewall, se realiza en tres pasos: el primero implica crear la estructura base para las reglas a través del método `default_tables()`. El segundo paso radica en crear las reglas que definen el comportamiento del firewall, mediante el método `Rule`. En este se definen de forma muy amigable el protocolo, el puerto afectado y la opción de aceptar, rechazar o dejar pasar el paquete analizado.

Operación del sistema:

Con dos consolas abiertas sobre Linux se valida la operación, en la primer consola se lanza el servidor modbus, y en la segunda consola se lanza el firewall. En la primera ejecución se lanza un firewall que acepta paquetes de entrada y salida del puerto 502 y tcp. De esta forma se puede establecer la comunicación entre el cliente que se ejecuta en Windows y el servidor al interior de la raspberry pi. Esto se evidencia en la figura 28.

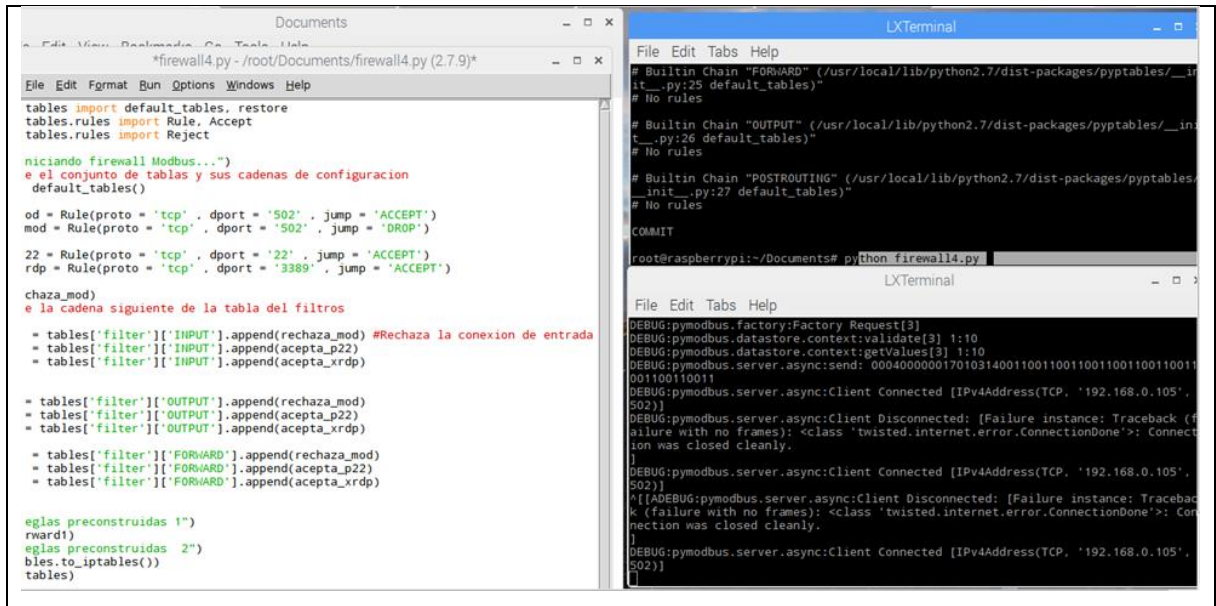
Figura 28. Operación de servidor con firewall que acepta todo el tráfico tcp por el puerto 502.



Fuente El Autor

En una ejecución posterior se activa la seguridad del sistema, modificando el Script del firewall, en este caso de deniega el acceso al puerto 502. De esta forma la conexión del cliente al servidor es rechazada, por el firewall programado. La figura 29 muestra esta ejecución.

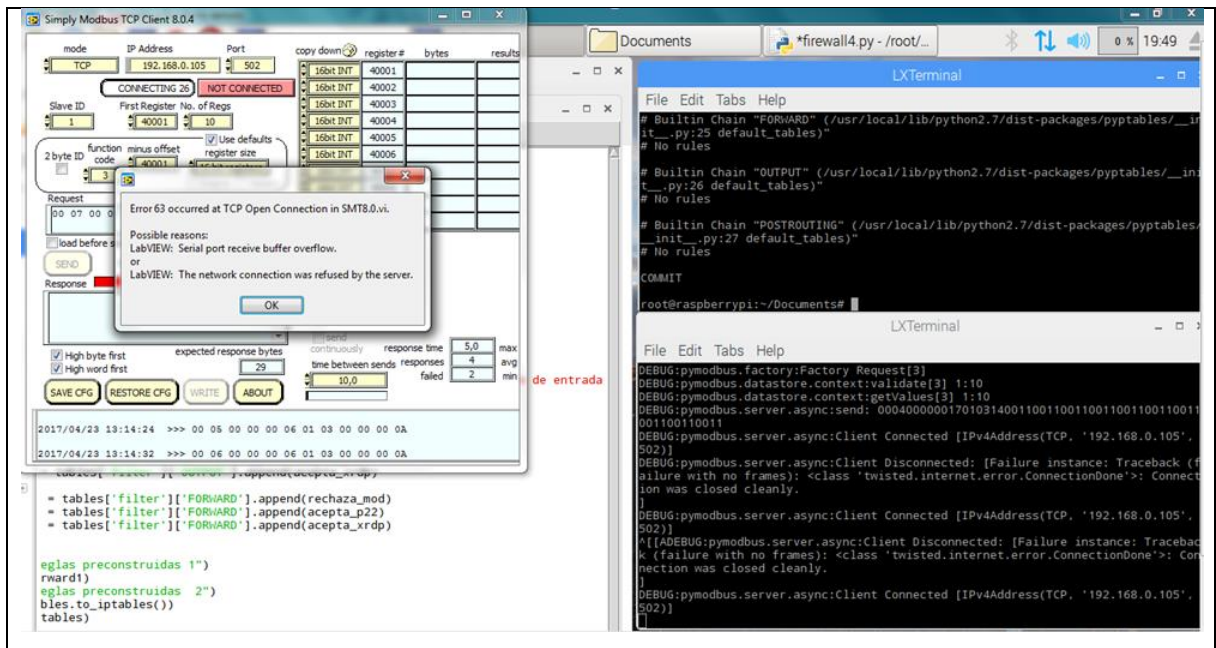
Figura 29. Operación de servidor con firewall que impide el paso de paquetes por el puerto 502 del protocolo modbus TCP.



Fuente El Autor

La figura 30, muestra el mensaje en el cliente, cuando es bloqueado por el firewall, se demuestra de esta forma la funcionalidad del mismo dada la topología planteada. Obsérvese que el cliente permanece no conectado.

Figura 30. Mensaje en el cliente cuando la conexión con el servidor es bloqueada por el firewall.



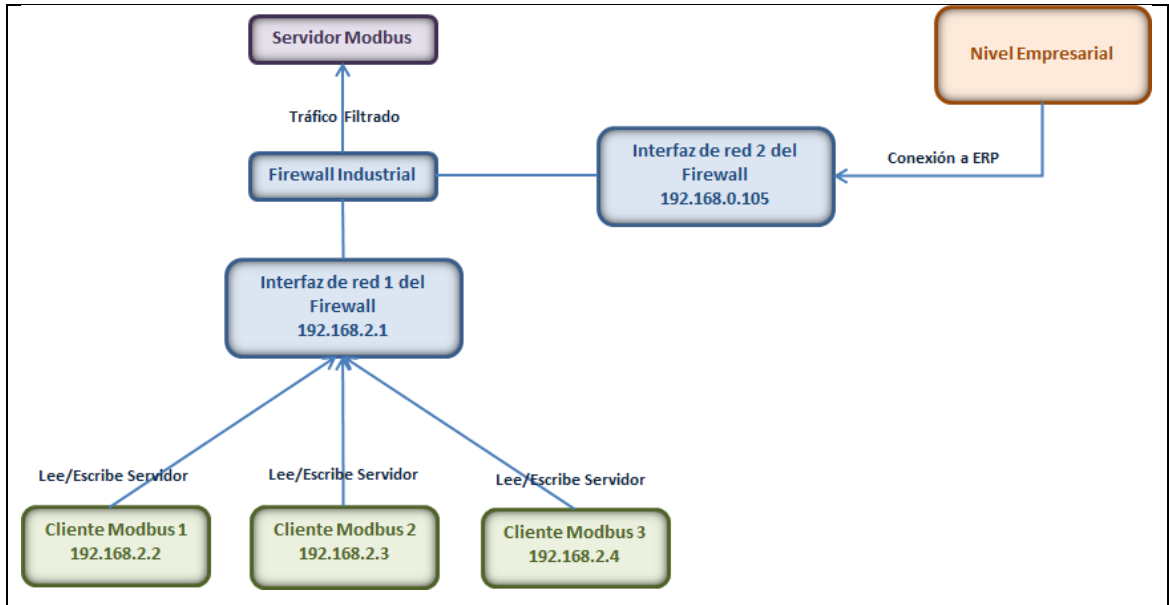
Fuente El Autor

VALIDACION DEL FIREWALL

El proceso de verificación de la funcionalidad del firewall considera aspectos importantes como: La topología de red, la ubicación del ataque, la activación del firewall, y el tipo de funcionalidad.

La figura 31 resalta la topología de red implementada, se destaca el firewall con dos interfaces de red, el servidor modbus está programado dentro de la raspberry pi.

Figura 31. Arquitectura de la Red con los Ips de las interfaces de red.



Fuente El Autor

Paso 1:

Para la validación se utiliza la herramienta Nessus en su licencia de siete días. Nessus es un escáner de vulnerabilidades patentado desarrollado por Tenable Network Security. Es gratuito para uso personal en un entorno no empresarial³⁷.

Nessus permite escaneos para los siguientes tipos de vulnerabilidades:

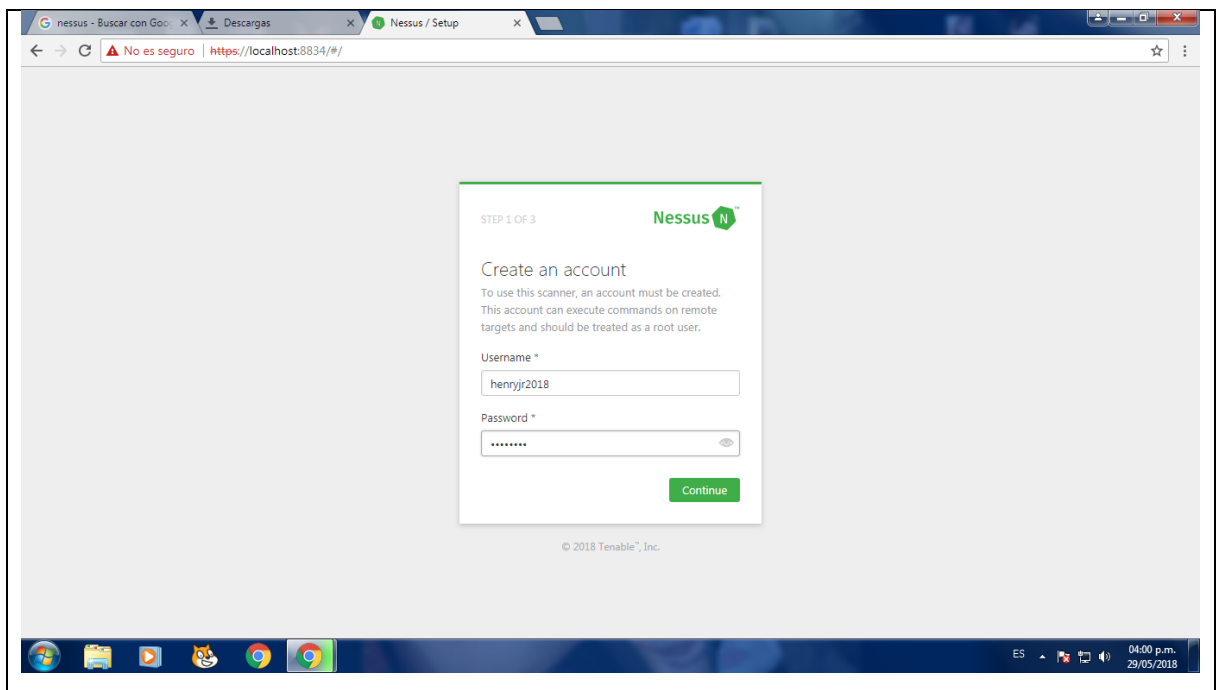
- Vulnerabilidades que permiten a un hacker remoto controlar o acceder a datos confidenciales en un sistema.
- Configuración incorrecta (por ejemplo, retransmisión de correo abierto, parches faltantes, etc.).
- Contraseñas predeterminadas, algunas contraseñas comunes y contraseñas en blanco / ausentes en algunas cuentas del sistema.

³⁷ "Nessus Release Notes". Tenable Network Security. Retrieved 2017-04-13. Copyright © 2015. Tenable Network Security, Inc. All rights reserved. Tenable Network Security and Nessus are registered trademarks of Tenable Network Security, Inc

- Nessus también puede llamar a Hydra (una herramienta externa) para lanzar un ataque de diccionario.
- Denegaciones de servicio contra la pila TCP / IP mediante el uso de paquetes mal formados
- Preparación para las auditorías PCI DSS

El programa crea un entorno de servidor local descargando las librerías para su operación, la figura 32 muestra el entorno de inicio.

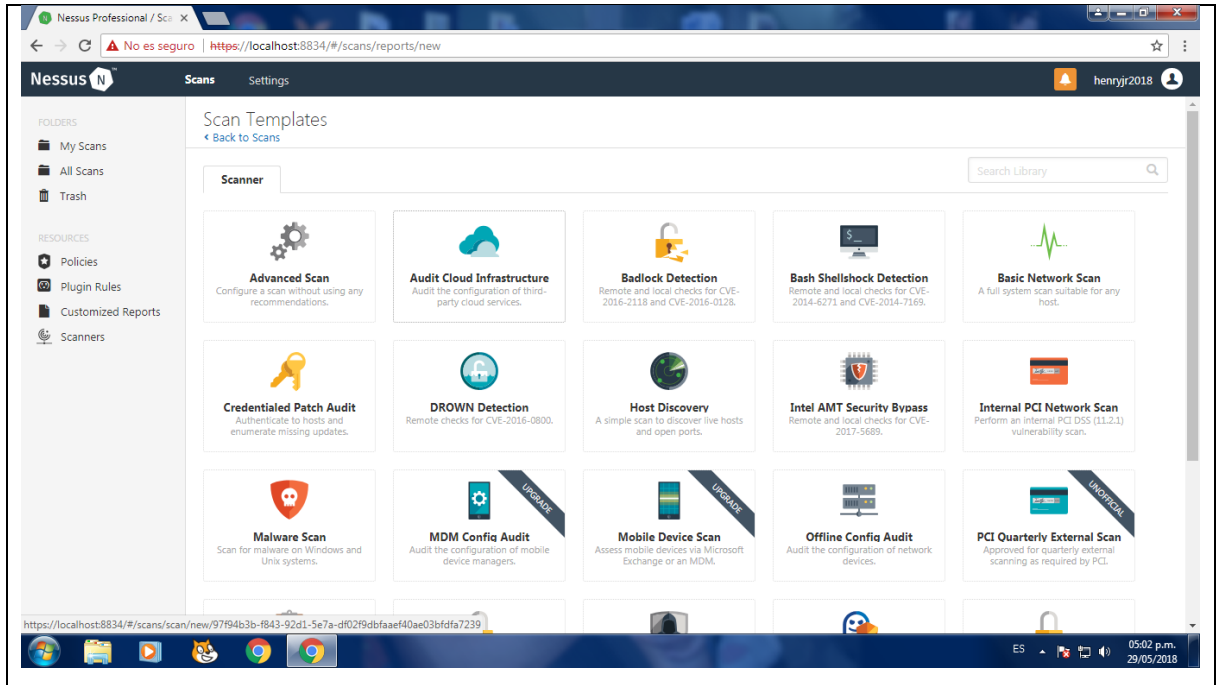
Figura 32. Inicio de servidor Nessus Fuente: Los autores.



Fuente El Autor

Al iniciar sesión el programa permite elegir entre diferentes tipos de escaneos, desde el análisis básico, detección de badlocks, búsqueda de hosts, Bypass de seguridad ATM, escaneo interno de red PCI, escaneo de malware etc. La figura 33 muestra la ventana con la relación a los tipos de escáner.

Figura 33. Conjunto de análisis disponibles en Nessus



Fuente El Autor

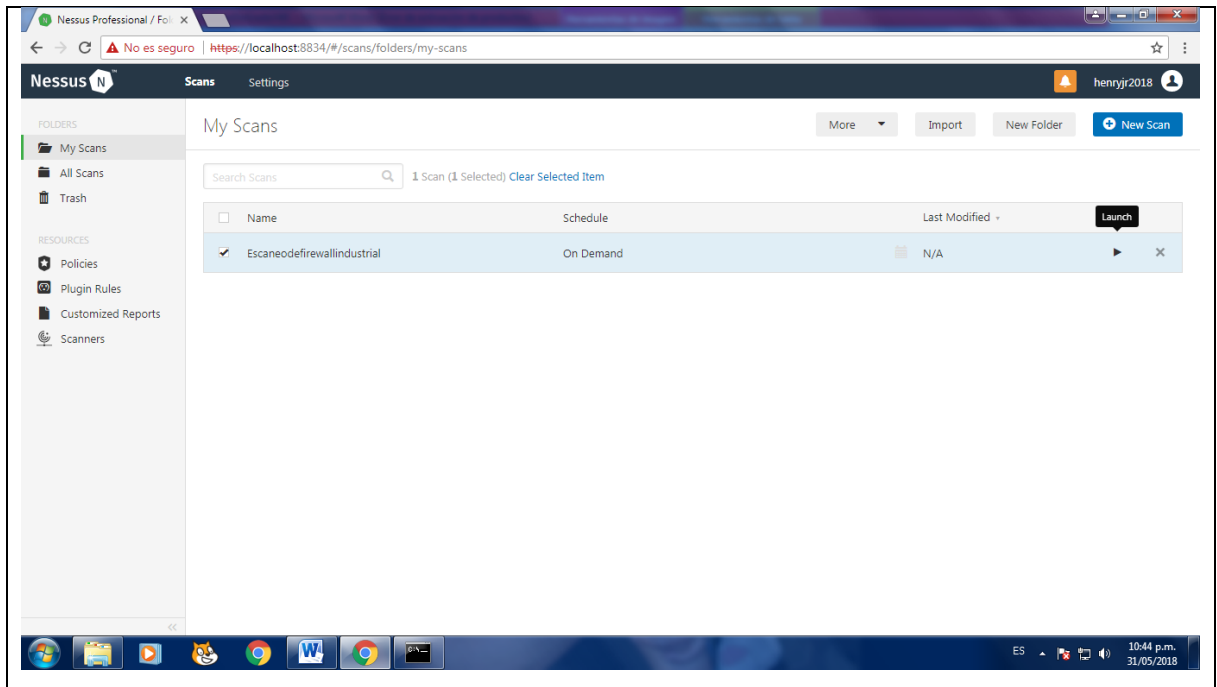
Los análisis realizados en la presente validación emplean la plantilla de escaneo de red básico ya que realiza un análisis completo del sistema que es adecuado para cualquier host. Por ejemplo, se puede usar esta plantilla para realizar un análisis interno de vulnerabilidades en los sistemas informáticos de la red interna.³⁸

Paso 2:

El inicio del análisis es sencillo, se establece el tipo de escaneo, se elige configuración personalizada y se asigna el rango o la ip que se desea analizar. Luego se inicia el proceso dando ejecutar en la lista de escaneos configurados. Figura 34.

³⁸ Nessus 6.4 User Guide, April 5, 2016 (Revision 3), Copyright © 2015. Tenable Network Security, Inc. All rights reserved. Tenable Network Security and Nessus are registered trademarks of Tenable Network Security, Inc.

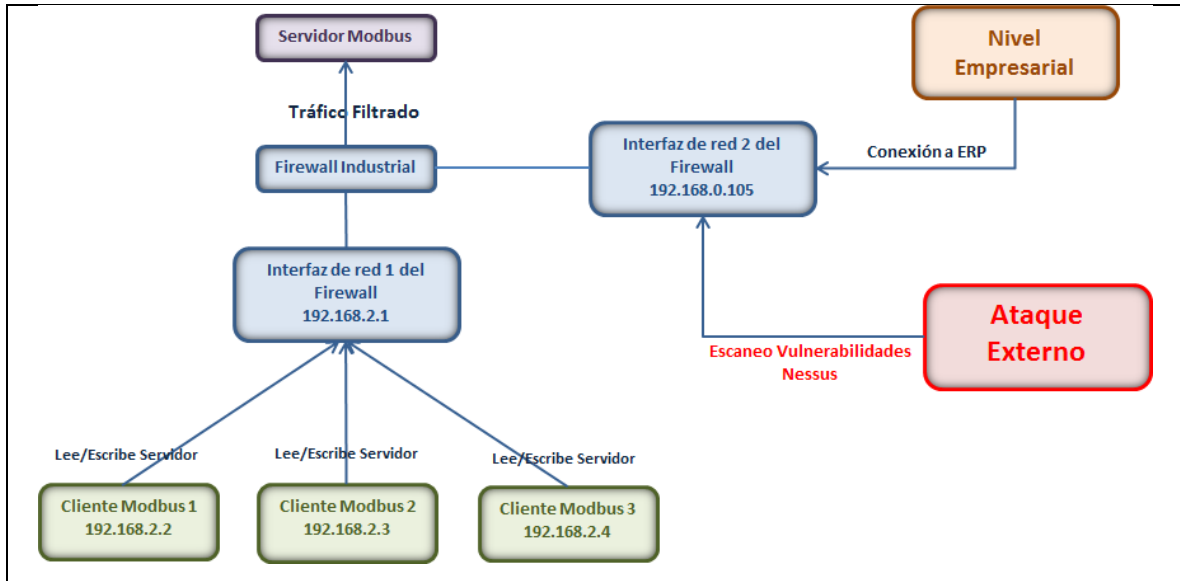
Figura 34. Escaneo Básico.



Fuente El Autor

El primer análisis considera un escaneo de vulnerabilidades desde la red externa, empleando la interfaz de red 2 del firewall. Se simula una situación de un posible ataque desde el nivel administrativo o externo a la planta. La figura 35 muestra la ubicación del “ataque”.

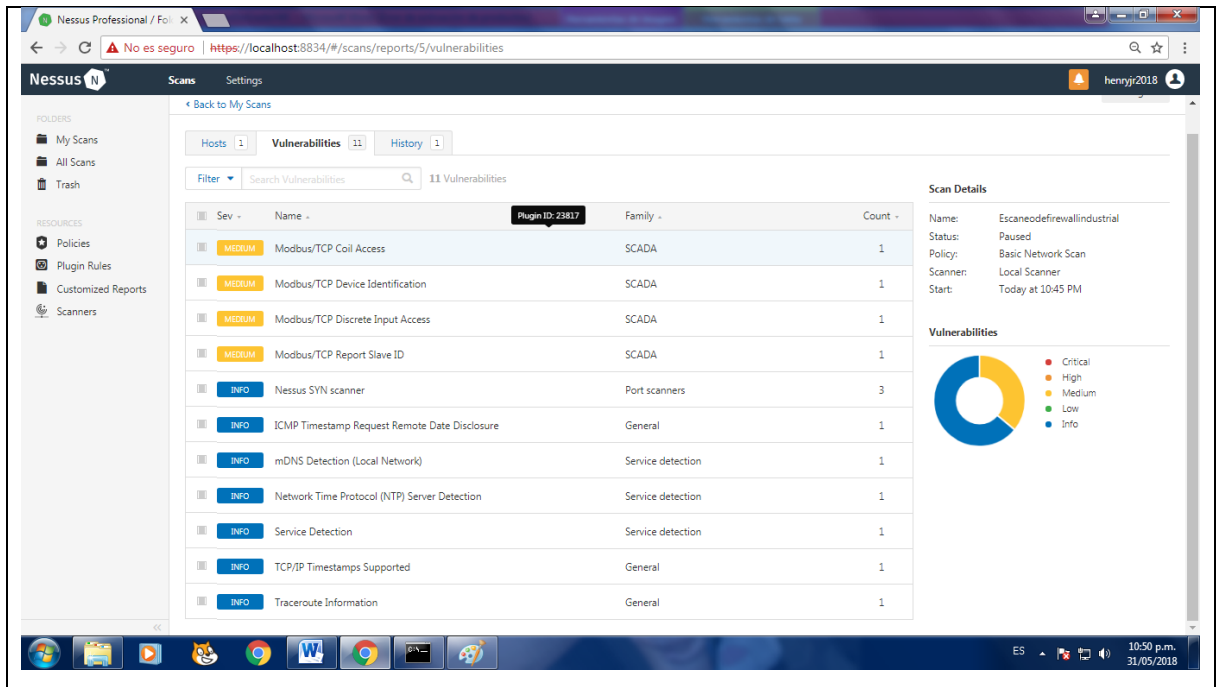
Figura 35. Topología de red para el análisis 1, desde el exterior de la red



Fuente El Autor

El proceso toma aproximadamente diez minutos debido a que la red es muy pequeña, ofrece como resultado un conjunto de reportes que señalan la cantidad de vulnerabilidades y su tipo. AL mismo tiempo explica la vulnerabilidad, el origen y un conjunto de contramedidas para reducir su riesgo. Las vulnerabilidades se clasifican en informativas, de nivel medio o alto y críticas. La figura 36 muestra el resultado con las reglas del firewall desactivadas.

Figura 36. Resultado del Análisis 1, desde el exterior con el firewall desactivado.

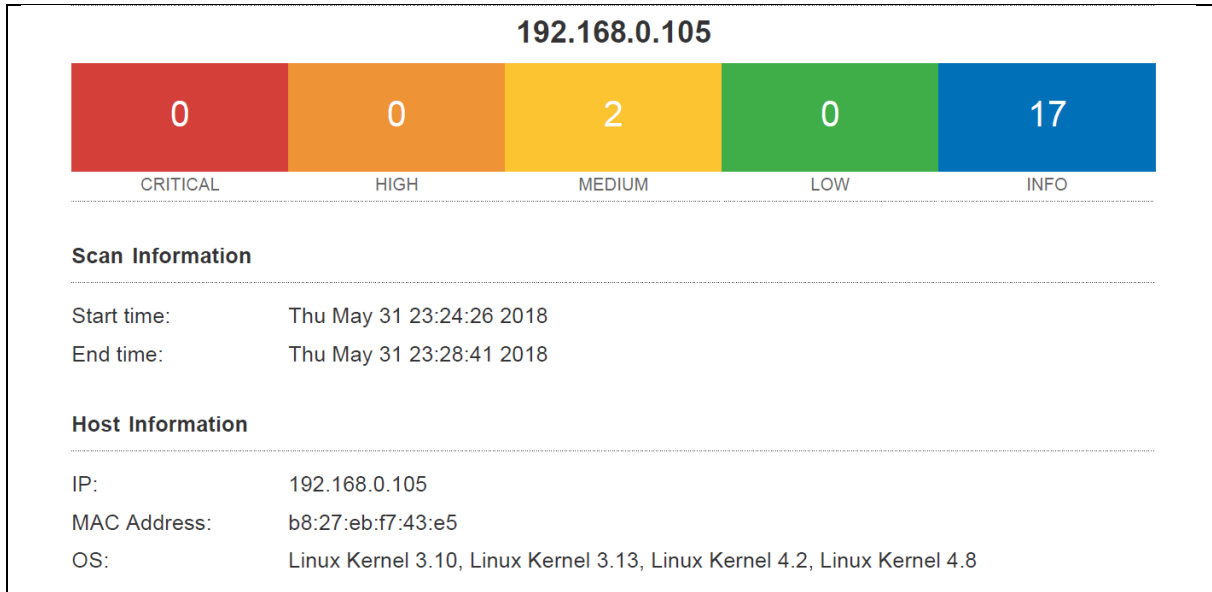


Fuente El Autor

Paso 3:

Bajo la misma topología se activa el firewall y se inicia otro escaneo, el resultado se puede observar en la figura 37.

Figura 37. Resultado del Análisis 1, desde el exterior con el firewall activo



Fuente El Autor

El resultado de la figura 37 es tomado del reporte completo generado por el sistema. Para este análisis desde el exterior, se tienen conclusiones varias, con el firewall abierto la herramienta puede ingresar al servidor modbus, el cual contiene la información crítica, mostrando las entradas, las salidas, la identificación del dispositivo y la identificación del esclavo que en ese momento se encuentra conectado, además de advertencias no relacionadas al protocolo industrial como en seguimiento de ip y apertura del puerto 22. Aunque la vulnerabilidad es de nivel medio esta información es suficiente para acceder al servidor modificando su información. Se puede recordar en este punto el virus Stuxnet el cual ingreso a una red industrial.

Al activar el firewall, se cierran las conexiones externas, el resultado muestra que las vulnerabilidades se reducen a dos. Pero estas no están asociadas al protocolo modbus y a la red industrial como tal.

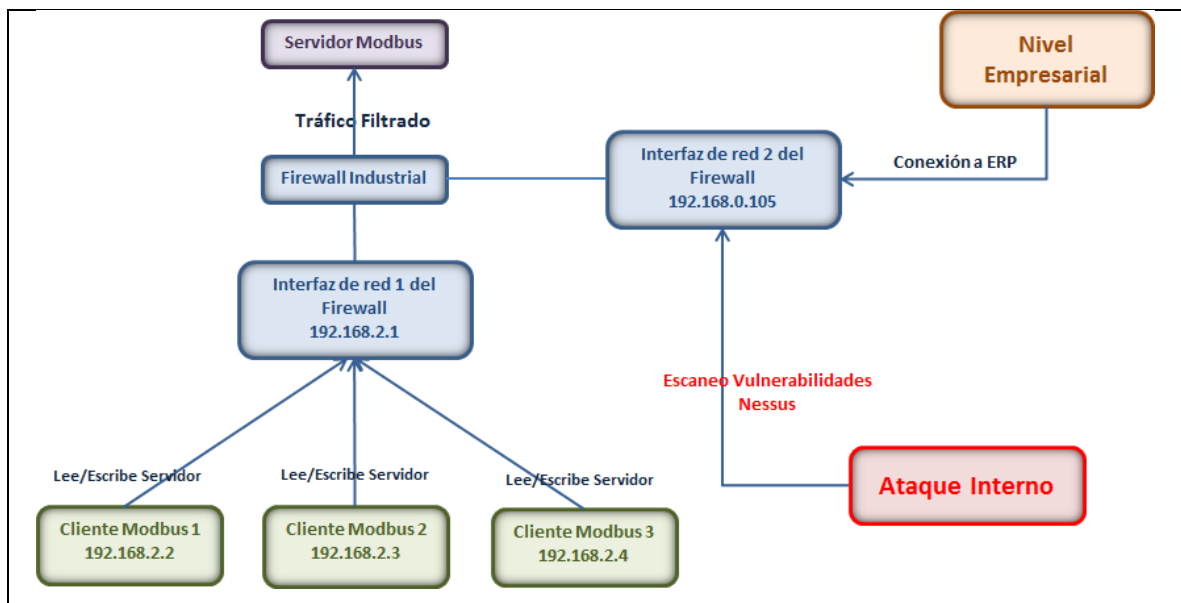
La primera muestra que se tiene activado el puerto 22, por medio del cual se hacen conexiones ssh para configuración de la raspberry. La segunda muestra que la máquina se encuentra haciendo ruteo de ip. Esto es debido a que la

raspberry está configurada como punto de acceso, para hacer el ruteo de paquetes en la red modbus.

Paso 4:

El segundo análisis se realiza desde la red interna, ver la figura 38. Aquí la herramienta de escaneo se conecta a la interfaz 1 del firewall, simula una situación en donde un cliente modbus no autorizado pretende acceder al servidor tanto con el firewall abierto como con el firewall cerrado.

Figura 38. Topología de red para el análisis 2, desde el interior de la red.

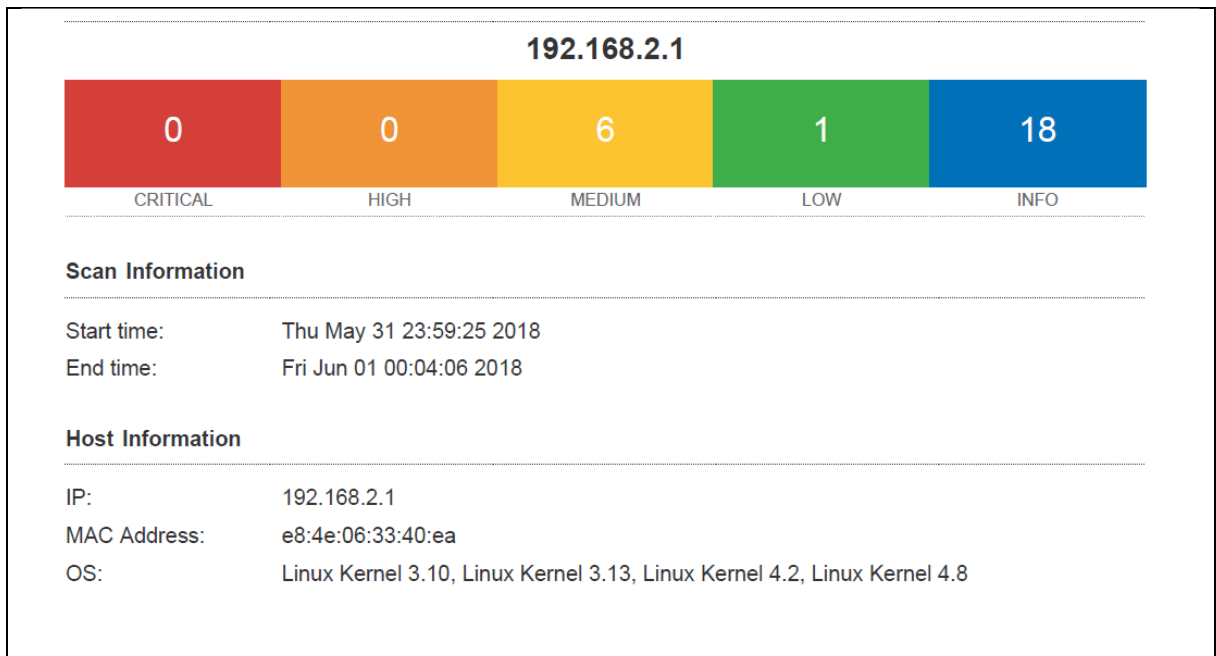


Fuente El Autor

El primer resultado para el análisis desde la red interna con firewall abierto muestra las mismas vulnerabilidades: las entradas del servidor, las salidas del servidor, la identificación del dispositivo y la identificación del esclavo que en ese momento se encuentra conectado, además de advertencias no relacionadas al protocolo industrial como en seguimiento de ip y apertura del puerto 22.

Este resultado se puede ver en la figura 39.

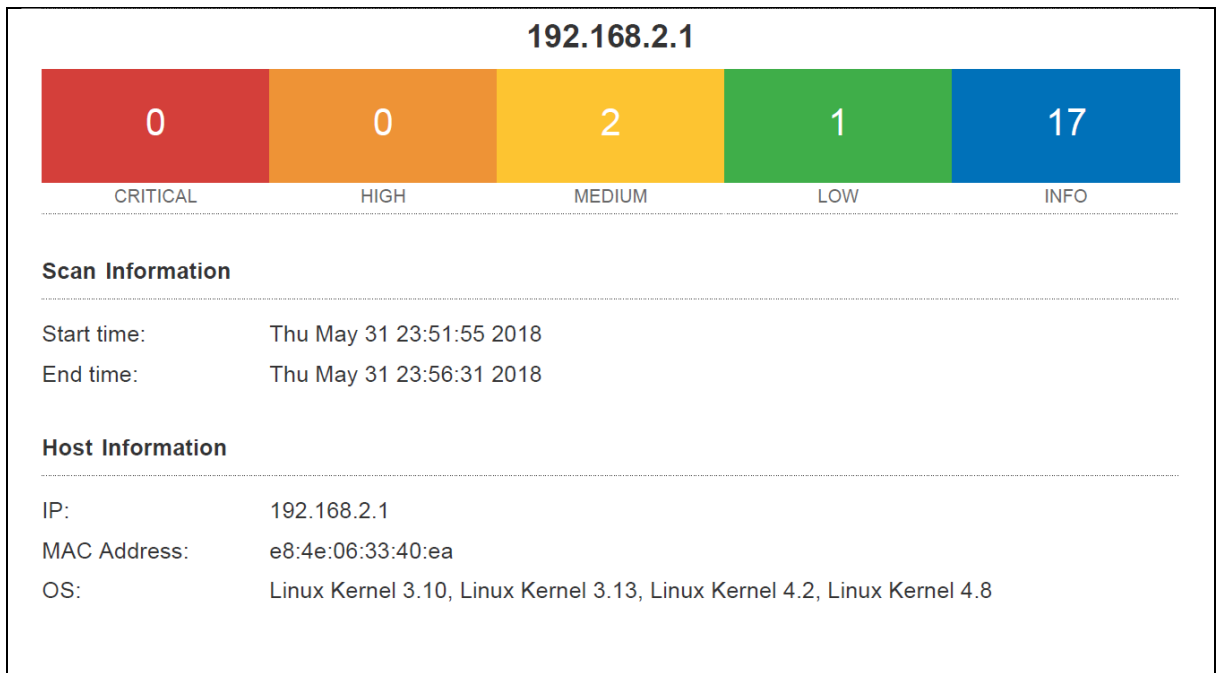
Figura 39. Resultado para el análisis 2, desde el interior de la red, con firewall abierto.



Fuente El Autor

La figura 40 muestra las vulnerabilidades para el análisis de la red interna, con firewall cerrado. Se limita de la misma forma a la información de nivel industrial, pero se continúa con las vulnerabilidades a nivel de acceso ssh y de seguimiento de ip ya vistas anteriormente.

Figura 40. Resultado para el análisis 2, desde el interior de la red, con firewall cerrado.



Fuente El Autor

Con este conjunto de comparaciones se valida la operación del firewall bajo dos escenarios se concluye que es completamente operativo y que el conjunto de reglas puede ampliarse para proteger más puertos o protocolos instalados al interior de una red industrial. Los anexos incluyen los 4 reportes generados por la herramienta para los casos de análisis.

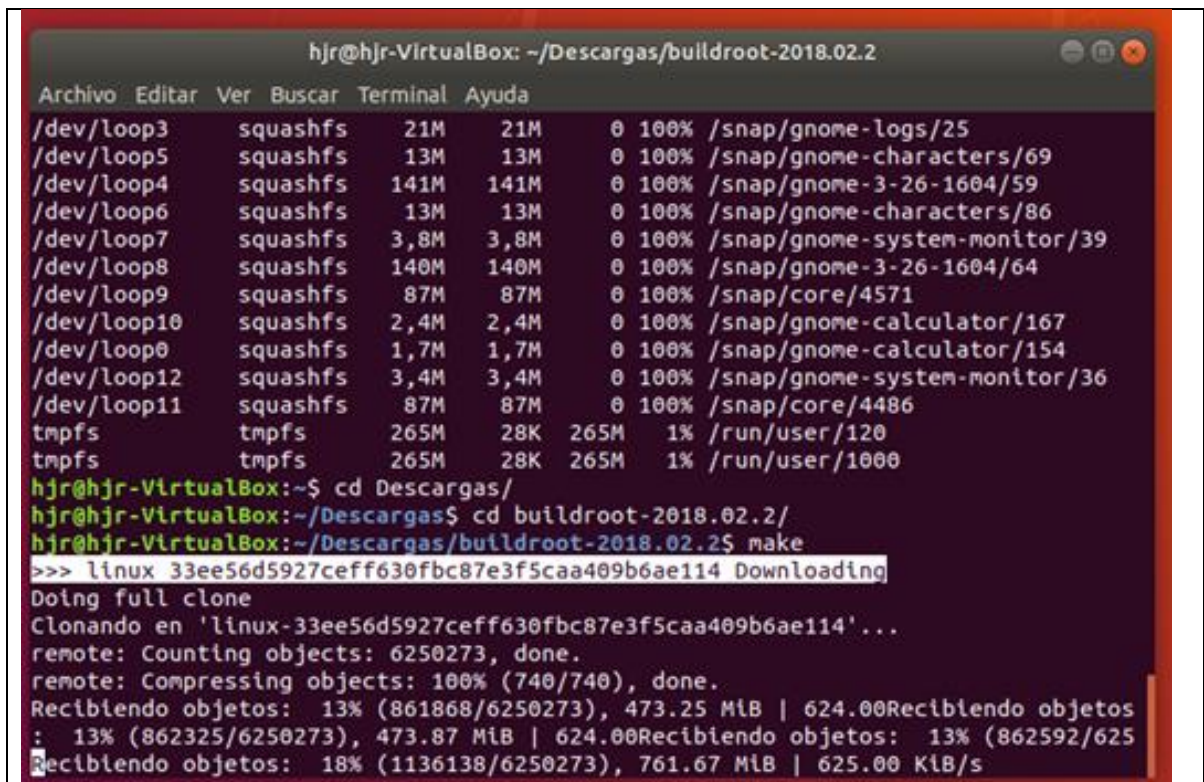
BUILD ROOTS

En la configuración realizada se añadió la opción de ejecución para los scripts de Python.

En este paso se ejecuta el make para la creación de la imagen del sistema operativo que debe ser luego grabada en sd o usb para el arranque de la raspberry. La orden de ejecución es simple: make.

La figura 41 muestra su ejecución, la cual dependiendo del computador host puede requerir entre una y dos horas.

Figura 41. Creación imagen del sistema operativo



```
hjr@hjr-VirtualBox: ~/Descargas/buildroot-2018.02.2
Archivo Editar Ver Buscar Terminal Ayuda
/dev/loop3 squashfs 21M 21M 0 100% /snap/gnome-logs/25
/dev/loop5 squashfs 13M 13M 0 100% /snap/gnome-characters/69
/dev/loop4 squashfs 141M 141M 0 100% /snap/gnome-3-26-1604/59
/dev/loop6 squashfs 13M 13M 0 100% /snap/gnome-characters/86
/dev/loop7 squashfs 3,8M 3,8M 0 100% /snap/gnome-system-monitor/39
/dev/loop8 squashfs 140M 140M 0 100% /snap/gnome-3-26-1604/64
/dev/loop9 squashfs 87M 87M 0 100% /snap/core/4571
/dev/loop10 squashfs 2,4M 2,4M 0 100% /snap/gnome-calculator/167
/dev/loop0 squashfs 1,7M 1,7M 0 100% /snap/gnome-calculator/154
/dev/loop12 squashfs 3,4M 3,4M 0 100% /snap/gnome-system-monitor/36
/dev/loop11 squashfs 87M 87M 0 100% /snap/core/4486
tmpfs tmpfs 265M 28K 265M 1% /run/user/120
tmpfs tmpfs 265M 28K 265M 1% /run/user/1000
hjr@hjr-VirtualBox:~$ cd Descargas/
hjr@hjr-VirtualBox:~/Descargas$ cd buildroot-2018.02.2/
hjr@hjr-VirtualBox:~/Descargas/buildroot-2018.02.2$ make
>>> linux 33ee56d5927ceff630fbc87e3f5caa409b6ae114 Downloading
Doing full clone
Clonando en 'linux-33ee56d5927ceff630fbc87e3f5caa409b6ae114'...
remote: Counting objects: 6250273, done.
remote: Compressing objects: 100% (740/740), done.
Recibiendo objetos: 13% (861868/6250273), 473.25 MiB | 624.00K/s
Recibiendo objetos: 13% (862325/6250273), 473.87 MiB | 624.00K/s
Recibiendo objetos: 13% (862592/6250273), 474.49 MiB | 624.00K/s
Recibiendo objetos: 18% (1136138/6250273), 761.67 MiB | 625.00K/s
```

Fuente El Autor

El último paso es crear la imagen en usb, puede hacerse con el comando fdisk o copiando la imagen en Windows, para su creación con win32manager, etcher o rufus.

Debe mencionarse que el sistema creado emplea: el kernel de Linux, el intérprete de Python y las interfaces de comunicaciones, con esto la imagen creada paso de 4.5 Gb a 500Mbytes.

El proceso que se siguió empleó la versión 18 de Ubuntu y los siguientes paquetes son requeridos para la ejecución de buildroots:

Tabla 4. Paquetes requeridos para la ejecución de buildroots

-which -sed -make -binutils -build-essential -gcc -g++ -bash -patch -gzip -bzip2	-perl -tar -cpio -python -unzip -rsync -file -bc
--	---

Fuente: <https://buildroot.org/downloads/manual/manual.pdf>

ANALISIS DE RESULTADOS

El desarrollo de software propuesto realizado en software de alto nivel, demostró estar en la capacidad de bloquear puertos, paquetes y protocolos sobre una plataforma de hardware reducido y con una versión de Linux de propósito general.

En relación con los objetivos propuestos, el lector puede detallar como se ha seguido uno a uno dentro de la construcción lógica de la investigación aplicada propuesta, detallando las topologías de firewalls en el cap, explicando los componentes de una red industrial en la sección 4.1.3, resumiendo el filtrado de paquetes en el ítem 4.1.4, realizando el desarrollo de software requerido que si bien es sencillo al ser implementado en alto nivel, alcanza altos niveles de portabilidad.

Si bien el estudio puede ser ampliado en relación con pruebas sobre otros tipos de protocolos de red industrial. La arquitectura de Iptables garantiza que los paquetes serán filtrados sin importar su orden interno siempre que se transporten sobre TCP o UDP.

Por el lado del aporte a trabajos anteriores, se ha dado un paso más al realizar el desarrollo en alto nivel, manteniendo la dependencia de Iptables y adicionando sencillez a la configuración y modificación de reglas y políticas de filtrado.

Por el lado del trabajo pendiente o futuro y por fuera del alcance de los objetivos aquí planteados y pensando en un prototipo mínimo viable, se puede desarrollar la interfaz gráfica de configuración y operación, así como la adición de un módulo de estadísticas y monitoreo de los paquetes filtrados; resultando de especial interés un conjunto de pruebas sobre otros equipos y protocolos de red industrial.

CONCLUSIONES

- Si bien se encuentra una amplia bibliografía en el desarrollo de firewalls abarcando diferentes metodologías y funcionalidades se encuentra muy poca información respecto a la implementación de estos en plataformas abiertas y de bajo costo.
- Los resultados demostraron la funcionalidad de la aplicación desde un lenguaje de alto nivel y de distribución libre con características básicas pero funcionales para configurar el núcleo IPTABLES de Linux en su función para el filtrado de paquetes usando el protocolo modbus.
- Debido a las restricciones funcionales de Shodan en lo referente al acceso remoto se empleó la herramienta Nessus la cual permitió la realización del análisis de vulnerabilidades, que demostró la funcionalidad del firewall.
- El desarrollo se divulga a la comunidad mediante una página blog de internet, planteando de esta forma una discusión abierta en el tema.
- El protocolo Modbus con características TCP, encaja en la topología de red en nivel 3 para el monitoreo de paquetes a través del núcleo Linux empleando iptables.
- Linux en su robustez de diseño, permite el análisis de todo tipo de paquetes siempre que las reglas de monitoreo se encuentren correctamente escritas.
- Se demuestra que la construcción de un firewall, apoyado sobre tecnologías open source es posible y sirve como prototipo funcional con fines comerciales al ser implementado en plataformas de hardware reducido como en la raspberry pi.

- Se demuestra la viabilidad de un firewall embebido para aplicaciones industriales que puede ser robustecido al agregarle otros protocolos industriales tales como CAN o Fieldbus, ampliamente usados en la industria.
- El análisis de protocolos se convierte en un objetivo de estudio en seguridad informática para la protección de infraestructura industrial y de producción, la cual en nuestro medio se encuentra todavía expuesta.
- Este producto obtenido puede convertirse en una alternativa a los firewalls embebidos disponibles

RECOMENDACIONES

Para llevar a cabo este proyecto y realizar la implementación, se debe tener conocimiento sobre firewall y topologías que soporta. Al igual sobre el funcionamiento de los protocolos de red industrial, el modo de transmisión y definir cuál de ellos se va utilizar.

Debido a que el núcleo de Linux soporta varios protocolos y puertos, logrando entre ambos varias combinaciones se debe definir una política que guie la escritura de las reglas y de esta manera lograr tener control.

Este proyecto se ha redactado de tal manera que le permite al lector entender todos los pasos que se llevaron a cabo para la implementación del firewall industrial.

DIVULGACION

OBJETIVOS	ACTIVIDADES DE DIFUSION	TAREAS	INSUMOS/RECURSOS
Dar a conocer el proyecto desarrollo de un firewall con una arquitectura de bajo costo para sistemas de monitoreo y control en redes industriales	Publicación de Blog https://firewall-industrial.blogspot.com/	Publicar Blog	*Ing. Henry Jiménez Rosero *Ing. Nairne Ovalles Hernández

BIBLIOGRAFIA

LOPEZ SANDOVAL. Jonny. Diseño de un prototipo que permita evaluar la viabilidad de un firewall en redes Scada, Fundación Universitaria Konrad Lorenz, 2010. Internet:
http://www.konradlorenz.edu.co/images/stories/articulos/PROTOTITPO_FIREWALL_REDES_SCADA.pdf

KRUTZ Ronald. Securing SCADA Systems. Wiley Publishing, Inc.Cap 4.Pag 76. 2004. Internet:
https://www.fer.unizg.hr/download/repository/Securing_SCADA_Systems.pdf

HSINCHUN C, EDNA, JOSHUA S, ANDREW S, BOAZ G .Terrorism Informatics: Knowledge Management and Data Mining for Homeland, 2005. Internet:
<http://www.springer.com/us/book/9780387716121>

GALDÁMEZ Pablo. Seguridad Informatica.2008. Internet:
<http://web.iti.upv.es/actualidadtic/2003/07/2003-07-seguridad.pdf>

WEISS Joe. crusader for critical infrastructure security(Q&A), 2010. Internet:
<http://www.cnet.com/news/joe-weiss-crusader-for-critical-infrastructure-security-q-a/>

WEISS Joe. DHS Blasts Reports of Illinois Water Station, 2010. Internet:
<http://krebsonsecurity.com/tag/joe-weiss/>

YOON Mike, WARREN Bruce. SCADA Systems. Internet:
<http://www.eolss.net/sample-chapters/c08/e6-187-06-00.pdf>

The President's Critical Infrastructure Protection (2010). Internet:

<https://www.federalregister.gov/agencies/president-s-critical-infrastructure-protection-board>

Board 21 Steps to Improve Cyber Security of SCADA Networks. 2011. Internet:
<http://www.oe.netl.doe.gov/docs/prepare/21stepsbooklet.pdf>

MORALES TEJEDA Gustavo. Diseño e implementación de un firewall,2002.
Internet:
<http://users.salleurl.edu/~is06200/proyectos/TFC-is06200.pdf>

RANUM Marcus. Thinking about Firewalls, 1998. Internet:
<http://csrc.nist.gov/publications/secpubs/fwalls.pdf>

RANUM Marcus. Firewalls Frequently Asked Questions. (1995). Internet:
<http://web.teipir.gr/WWWjgkad/security/firewall/faq.htm>

CHAPMAN Brend Building Internet Firewalls. O'Reilly & Associates, 1st edition,
Noviembre 1995. Internet:
http://dbmanagement.info/Books/Others/O'Reilly_Building_Internet_Firewalls_2nd_Edition.pdf

NATHAN Smith. Stack Smashing Vulnerabilities in the Unix Operating System,
1997. Internet:
http://web.eecs.umich.edu/~aparakash/security/handouts/Stack_Smashing_Vulnerabilities_in_the_UNIX_Operating_System.pdf

WACK Peter. Keeping your site comfortably secure: an introduction to Internet
Firewalls. Technical report, National Institute of Standards and Technology (NIST),
Diciembre 1994. Special Publication 800-10.

VILLALON Andres, Seguridad en Unix y Redes, 2002. Internet:

<https://www.rediris.es/cert/doc/unixsec/node23.html>

TORRES Roger, Implementación del protocolo en microcontrolador, 2006.

Internet:

<http://repositorio.uis.edu.co/jspui/bitstream/123456789/3176/2/119444.pdf>

Software para aplicaciones industriales. Internet:

<ftp://ftp.unicauca.edu.co/Facultades/FIET/DEIC/Materias/SW%20para%20aplicaciones%20Industriales%20II/Sw%20II/Conferencias/Capitulo%205.pdf>

Industrial Cybersecurity. Firewall Industriales DPI. Internet:

http://www.ciberseguridadlogitek.com/wp-content/uploads/Firewalls-industriales_Hirschmann_Tofino_DPI.pdf

Raspberry Pi. Internet:

https://es.wikipedia.org/wiki/Raspberry_Pi

PELLO Xabier, IPTABLES Internet:

<http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>

COCKBURN Jamie, Python PyPTables 1.0.1, Internet:

<https://pypi.python.org/pypi/PyPTables/1.0.1>

Sensibilización seguridad de la información, Agosto de 2013. Internet:

http://www.mineduacion.gov.co/1759/articles27021_archivo_pdf_Dia1_6_Sensibilizacion.pdf

Pirámide De Kelsen, febrero de 2011. Internet:

<https://iusuniversalis.blogia.com/2011/022402-piramide-de-kelsen.php>

LIMA PERU. Decisión 351 de la comunidad andina de naciones. Internet:
<http://www.wipo.int/edocs/lexdocs/laws/es/can/can010es.pdf>

COLOMBIA CONGRESO DE LA REPUBLICA. Ley 23 de 1982, Sobre los derechos de autor. Internet:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=3431>

COLOMBIA CONGRESO DE LA REPUBLICA. Decreto 1360 DE 1989, inscripción del soporte lógico. Internet:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=10575>

COLOMBIA CONGRESO DE LA REPUBLICA. Ley 44 de 1993, modificación a la ley 23 de 1982. Internet:
<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=3429>

COLOMBIA CONGRESO DE LA REPUBLICA. Decreto 162 de 1996 Internet:
[http:// www.alcaldiabogota.gov.com /sisjur/normas/Norma1.jsp?i=10574](http://www.alcaldiabogota.gov.com/sisjur/normas/Norma1.jsp?i=10574)

COLOMBIA CONGRESO DE LA REPUBLICA. DECRETO 333 DE 2014, Internet:
[http:// www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=56767#22](http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=56767#22)

COLOMBIA CONGRESO DE LA REPUBLICA. Ley 1341 de 2019, Internet:
http://www.mintic.gov.co/portal/604/articles-3707_documento.pdf

COLOMBIA CONGRESO DE LA REPUBLICA. Ley estatutaria 1581 de 2012. Internet:
[http:// www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=49981](http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=49981)

COLOMBIA CONGRESO DE LA REPUBLICA. LEY 1273 DE 2009, Código penal, Internet:

<http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34492>

Iptables Manual Práctico, 2014, internet:

<https://mizonapc.wordpress.com/category/linux/page/2/>

Buildroot, Wikipedia, internet:

https://en.wikipedia.org/wiki/Buildroot_2018.

"Nessus Release Notes". Tenable Network Security. Retrieved 2017-04-13. Copyright © 2015. Tenable Network Security, Inc. All rights reserved. Tenable Network Security and Nessus are registered trademarks of Tenable Network Security, Inc.

Nessus 6.4 User Guide, April 5, 2016 (Revision 3), Copyright © 2015. Tenable Network Security, Inc. All rights reserved. Tenable Network Security and Nessus are registered trademarks of Tenable Network Security, Inc.

Cheminod, M., Durante, L., Seno, L., Valenzano, A. (n.d.). Performance Evaluation and Modeling of an Industrial Application-Layer Firewall. IEEE Transactions on Industrial Informatics, 14(5), pp. 2159–2170.

CHERIFI, T., & HAMAMI, L. (2018). A practical implementation of unconditional security for the IEC 60780-5-101 SCADA protocol. International Journal of Critical Infrastructure Protection, 20, 68–84. , Internet:

<https://doi.org/10.1016/j.ijcip.2017.12.001>

Jiang, N., Lin, H., Yin, Z., & Xi, C. (2017). Research of paired industrial firewalls in defense-in-depth architecture of integrated manufacturing or production system. In 2017 IEEE International Conference on Information and Automation, ICIA 2017 (pp. 523–526). <https://doi.org/10.1109/ICInfA.2017.8078963>

Kleinmann, A., Amichay, O., Wool, A., Tenenbaum, D., Bar, O., & Lev, L. (2018). Stealthy deception attacks against SCADA systems. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10683 LNCS, 93–109. https://doi.org/10.1007/978-3-319-72817-9_7

Kolisnyk, M.aEmail Author, Piskachova, I.b, Kharchenko, V. . (n.d.). Patching the firewall software to improve the availability and security: Markov models for internet of things based smart business center. *CEUR Workshop Proceedings*, Volume 210, Pages 517-529.

Tsuchiya, A.aEmail Author, Fraile, F.bEmail Author, Koshijima, I.aEmail Author, Órtiz, A.bEmail Author, Poler, R. . (2018). Software defined networking firewall for industry 4.0 manufacturing systems. *Journal of Industrial Engineering and Management*.

ANEXOS

Anexo A. Resumen Analítico RAE

Título de Documento.	DESARROLLO DE UN FIREWALL CON UNA ARQUITECTURA DE BAJO COSTO PARA SISTEMAS DE MONITOREO Y CONTROL EN REDES INDUSTRIALES
Autor	Henry Jiménez Rosero Nairne Ovalles Hernández
Palabras Claves	Sistema Scada, firewall, protocolo, Red, Usuario, Iptables, servidor, raspberry pi
Descripción	<p>En este estudio se examina la importancia de un firewall, como concepto transversal en la seguridad de una red industrial. Considerando que las redes industriales han sido en principio diseñadas para proveer eficiencia en lo relacionado al tiempo de respuesta durante el transporte de datos, estos viajan como texto plano en las tramas de comunicaciones, dejando abierta la posibilidad de que cualquiera con un acceso mínimo a la red pueda escucharlos y o modificarlos.</p> <p>El texto muestra los conceptos fundamentales de redes industriales y de topologías lógicas de firewalls como preámbulo al desarrollo de un firewall en un lenguaje de alto nivel. El desarrollo explica el filtrado de paquetes propio de los núcleos de Linux, para permitir una implementación desde el lenguaje Python, garantizando su portabilidad. Las pruebas se realizan sobre hardware dedicado, empleando una plataforma raspberry pi 2.</p> <p>Los resultados alcanzados evidencian su funcionalidad y permiten prospectar un producto mínimo viable en el corto plazo.</p>
Fuentes Bibliográficas	<p>LOPEZ SANDOVAL. Jonny. Diseño de un prototipo que permita evaluar la viabilidad de un firewall en redes Scada, Fundación Universitaria Konrad Lorenz, 2010</p> <p>KRUTZ Ronald. Securing SCADA Systems. Wiley Publishing, Inc. Cap 4. Pag 76. 2004.</p>

	<p>GALDÁMEZ Pablo. Seguridad Informatica.2008.</p> <p>YOON Mike, WARREN Bruce. SCADA Systems</p> <p>MORALES TEJEDA Gustavo. Diseño e implementación de un firewall, 2002.</p> <p>RANUM Marcus. Firewalls Frequently Asked Questions, 1995.</p>
--	--

Contenido:

a) Descripción del problema:

La gran mayoría de redes industriales actuales se interconectan de forma directa a los sistemas administrativos empresariales empleando sistemas de seguridad convencionales no especializados, dejando expuesta información crítica del proceso involucrado la cual puede ser modificada o monitoreada desde el interior o exterior de la empresa por personal no autorizado, estos síntomas pueden ser debidos a múltiples causas, las cuales pueden abarcar desde políticas propias, pasando por implementaciones incorrectas y llegando en algunos casos hasta el desconocimiento del tema.

Las implementaciones incorrectas se relacionan con el desconocimiento de los diferentes propósitos de un sistema Scada y un sistema de información convencional. El primero pensado en la confiabilidad, respuesta en tiempo real, tolerancia a situaciones de emergencia, seguridad del personal, calidad del producto, seguridad física de la planta y el personal. El segundo enfocado con la provisión de conectividad interna y externa, el manejo de la productividad, manejo de bases de datos y con la implementación de extensos mecanismos de seguridad para la autenticación y autorización. (Krutz Ronald 2004. Cap1, pag 16).

Se puede llegar a predecir que los problemas de seguridad en redes industriales se deben a interconexiones incorrectas entre los dos sistemas generalmente ocasionados por la falta de uso de aplicaciones especializadas.

Dentro de las mejores prácticas para solucionar esta problemática se parte desde la implementación de políticas empresariales, registro de actividades, biometría para el acceso, sistemas de detección de intrusos, criptografía e implementación de firewalls, los cuales pueden proteger a la red Scada de una posible intrusión sobre la red convencional empresarial.

Entonces el presente anteproyecto pretende dar respuesta a esta problemática mediante la implementación de un firewall especializado que brinde una respuesta a las preguntas:

Cuáles son las principales consideraciones de un firewall embebido en una red industrial y sistema scada?

Cuál es la arquitectura de un firewall industrial, con respecto a paquetes de datos especializados que garantice el aislamiento de redes?

b) Objetivo General.

Desarrollar un filtrado de paquetes empleando Iptables en una arquitectura de bajo costo Raspberry PI para sistemas SCADA en redes industriales

c) Objetivos Específicos.

- Estado del arte orientado a firewalls de redes industriales y Raspberry pi.
- Desarrollar una aplicación de software libre que permita configurar las características de Iptables para ser usado como un firewall en una red industrial.
- Realizar pruebas de seguridad con shodan antes y después de implementar el firewall.
- Realizar un evento de divulgación para dar a conocer los resultados de la monografía.

DESARROLLO FASES DEL PROYECTO

Fase Diseño

El presente texto se enfoca en el análisis descriptivo de tipo exploratorio. Un estudio descriptivo permite recolectar información medible mediante la observación de conceptos y variables aplicables al problema como tal. Estas últimas señalan patrones de la realidad, que pueden inferirse y cuyos valores pueden cambiar la dinámica del sistema estudiado.

Debe aclararse que la parte exploratoria define una aplicación especial, debido a que da pie a la formulación de hipótesis de grado uno y grado dos, las cuales pueden ser sobresalientes en un nivel más detallado del análisis que se lleva a cabo.

Con bases en variables conocidas a priori se puede tener un marco de referencia del problema, lo que permite tener planteamientos más claros del mismo y los que además se constituyen en la base para formulación de experimentos.

Una vez definidos: el problema, el marco teórico, los objetivos y el método se establece como el siguiente paso: el diseño de la encuesta o del mecanismo de recolección de datos. Este mecanismo de medición debe almacenar datos observables sobre los conceptos o las variables. La robustez de este se da al cumplir dos objetivos básicos: Los datos deben ser confiables y deben ser válidos dentro del experimento registrado.

Fase diseño del software

Para realizar el diseño y desarrollo de un sistema de información se referencia como:

Diseño en espiral debido a que se pueden retomar pasos anteriores sobre cualquier punto del proceso siguiendo los procesos lógicos de desarrollo y verificación en ingeniería que en muchas ocasiones dan lugar a correcciones que obligan a la generación de nuevas versiones.

La aplicación se desarrollara sobre plataformas de hardware y software de libre distribución en donde sus arquitecturas son completamente conocidas de allí que las variables más importantes como el sistema operativo, la capacidad de procesamiento y las interfaces de red se conocen a priori y dan respaldo al desarrollo mismo.

El programa como tal se ejecutará sobre el sistema operativo Linux (raspbian jessie) y se desarrollara sobre el intérprete de Python 2.7. Lo cual le da amplia portabilidad.

Fase recolección de datos

La técnica utilizada para la recolección de datos es la recopilación documental en donde se usará fuentes bibliográficas como libros, textos, tesis, páginas web, revistas y todos los documentos necesarios para recopilar toda la información relacionada con el proyecto.

Técnica de Análisis de Datos

Revisión documental Se consultará la información necesaria que respalde los conceptos sobre firewall industrial, protocolo modbus y sistema scada

Conclusiones

- Si bien se encuentra una amplia bibliografía en el desarrollo de firewalls abarcando diferentes metodologías y funcionalidades se encuentra muy poca información respecto a la implementación de estos en plataformas abiertas y de bajo costo.
- Los resultados demostraron la funcionalidad de la aplicación desde un lenguaje de alto nivel y de distribución libre con características básicas pero funcionales para configurar el núcleo IPTABLES de Linux en su función para el filtrado de paquetes usando el protocolo modbus.
- Debido a las restricciones funcionales de Shodan en lo referente al acceso remoto se empleó la herramienta Nessus la cual permitió la realización del análisis de vulnerabilidades, que demostró la funcionalidad del firewall.
- El desarrollo se divulga a la comunidad mediante una página blog de internet, planteando de esta forma una discusión abierta en el tema.
- El protocolo Modbus con características TCP, encaja en la topología de red en nivel 3 para el monitoreo de paquetes a través del núcleo Linux empleando iptables.
- Linux en su robustez de diseño, permite el análisis de todo tipo de paquetes siempre que las reglas de monitoreo se encuentren correctamente escritas.
- Se demuestra que la construcción de un firewall, apoyado sobre tecnologías open source es posible y sirve como prototipo funcional con fines comerciales al ser implementado en plataformas de hardware reducido como en la raspberry pi.
- Se demuestra la viabilidad de un firewall embebido para aplicaciones industriales que puede ser robustecido al agregarle otros protocolos industriales tales como CAN o Fieldbus, ampliamente usados en la industria.
- El análisis de protocolos se convierte en un objetivo de estudio en seguridad informática para la protección de infraestructura industrial y de producción, la cual en nuestro medio se encuentra todavía expuesta.
- Este producto obtenido puede convertirse en una alternativa a los firewalls embebidos disponibles

Recomendaciones.

Para llevar a cabo este proyecto y realizar la implementación, se debe tener conocimiento sobre firewall y topologías que soporta. Al igual sobre el funcionamiento de los protocolos de red industrial, el modo de transmisión y definir cuál de ellos se va utilizar.

Debido a que el núcleo de Linux soporta varios protocolos y puertos, logrando entre ambos varias combinaciones se debe definir una política que guie la escritura de las reglas y de esta manera lograr tener control.

Este proyecto se ha redactado de tal manera que le permite al lector entender todos los pasos que se llevaron a cabo para la implementación del firewall industrial.

Anexo B. Reporte análisis desde el exterior de la red, con firewall abierto.

Anexo C. Reporte análisis desde el exterior de la red, con firewall cerrado

Anexo D. Reporte análisis desde el interior de la red, con firewall abierto

Anexo E. Reporte análisis desde el interior de la red, con firewall cerrado