Análisis de Vulnerabilidades en Sistemas Operativos de Dispositivos IoT mediante Shodan

1st Sara Lorena Duque Ramírez, 2nd Valeria Ballestero Ortiz, 3ro Isaac Esteban Uribe Jaramillo

Ingeniería de Sistemas, Universidad de

AntioquiaMedellín, Colombia

[1]sara.duque1@udea.edu.co

[2]valeria.ortizp@udea.edu.co

[3]isaac.uribej@udea.edu.co

I. INTRODUCCIÓN

Este proyecto aborda el análisis de vulnerabilidades en sistemas operativos de dispositivos IoT utilizando la herramienta Shodan, en el contexto de la creciente preocupación por la seguridad en el Internet de las Cosas. La motivación principal radica en la necesidad de proteger estos dispositivos, cuya integración en entornos domésticos e industriales ha incrementado su exposición a riesgos de ciberseguridad.

Mediante la metodología de hardening, se busca mitigar vulnerabilidades al deshabilitar funciones innecesarias y aplicar configuraciones mínimas de seguridad, evaluando además el impacto en el rendimiento operativo. El enfoque experimental incluye la medición de variables clave como el tiempo de arranque y la eficiencia del sistema de archivos antes y después de implementar estas estrategias. Los resultados obtenidos, a través de análisis estadísticos, permitirán determinar la efectividad del hardening en la mejora de la seguridad sin comprometer el rendimiento, aportando un marco sólido para la optimización de sistemas operativos IoT ante amenazas contemporáneas.

II. MARCO TEORICO

Los dispositivos IoT han revolucionado el ámbito tecnológico, integrándose en entornos domésticos, industriales y de servicios, lo que los convierte en objetivos clave para ataques cibernéticos. Según investigaciones, muchas vulnerabilidades en estos dispositivos derivan de configuraciones predeterminadas débiles, firmware desactualizado y falta de medidas de seguridad robustas. Estrategias como el hardening son esenciales para mitigar riesgos. El hardening consiste en aplicar configuraciones que reduzcan la superficie de ataque, desactivando funciones innecesarias y limitando el acceso a componentes externos como dispositivos USB. Además, herramientas como Shodan, un motor de búsqueda para dispositivos conectados, permiten identificar dispositivos IoT vulnerables analizando información pública como IPs y servicios abiertos, facilitando el diseño de medidas correctivas específicas. En este contexto, es crucial comprender el impacto que estas estrategias tienen no solo en la seguridad, sino también en el rendimiento del sistema operativo.

III. METODOLOGÍA

El proyecto se desarrolló en varias fases, utilizando la herramienta Shodan para explorar dispositivos IoT y sus vulnerabilidades.

A. Exploración inicial

Identificar dispositivos IoT por ejemplo cámaras a través de sus direcciones IP, recopilando información sobre sistemas operativos y configuraciones mediante shodan.

B. Diagnóstico de vulnerabilidades

Análisis detallado de las brechas de seguridad presentes en los dispositivos identificados, enfocándose en aspectos como configuraciones por defecto y servicios innecesarios habilitados.

C. Aplicación de hardening

Implementación de medidas de seguridad en los dispositivos seleccionados, incluyendo la desactivación de funciones innecesarias y la restricción de acceso a dispositivos externos.

D. Pruebas de rendimiento

Evaluación del tiempo de arranque, la eficiencia del sistema de archivos y la estabilidad del dispositivo antes y después del hardening, utilizando análisis estadísticos como ANOVA para identificar cambios significativos.

IV. IMPLEMENTACIÓN

La implementación del proyecto se centró en la creación de un entorno virtual para analizar y reforzar la seguridad de sistemas operativos utilizados en dispositivos IoT. Para ello, se utilizaron herramientas como VirtualBox y distribuciones de Linux especializadas en seguridad informática, incluyendo Kali Linux, Parrot OS y Kodachi. A continuación, se detalla el proceso paso a paso:

A. Preparación del entorno virtual

Aquí se detalla el paso a paso que se siguió para preparar el ambiente virtual que seria utilizado para realizar las pruebas.

1) Descarga e instalación de VirtualBox: Descargue VirtualBox desde su sitio oficial, eligiendo la versión de acuerdo al sistema operativo que tenga (Windows, macOS, o Linux). Instale siguiendo las instrucciones predeterminadas del instalador.

B. Descarga de Distribuciones de Linux

Se procederá a descargar imágenes ISO de distribuciones de Linux, específicamente Kali Linux, Parrot OS y Kodachi.

1) Obtención de las imágenes ISO: Descargue las distribuciones Linux necesarias:

- Kali Linux desde kali.org.
- Parrot OS desde parrotsec.org.
- Kodachi desde distrowatch.com.

C. Creación de máquinas virtuales

Se realizará la creación de máquinas virtuales para las distribuciones de Linux descargadas, utilizando VirtualBox. Este proceso incluye asignar un nombre a cada máquina, configurar la memoria RAM y crear un disco duro virtual con un tamaño adecuado.

- Abra VirtualBox y haga clic en **"Nueva"**.
- Asigne un nombre a la máquina virtual (por ejemplo, "Kali Linux") y seleccione el tipo de sistema operativo como **Linux** y la versión correspondiente.
- Configure la memoria RAM recomendada (2 GB para Parrot y Kali; 4 GB para Kodachi).
- Cree un disco duro virtual asignándole un tamaño de almacenamiento adecuado (por ejemplo, 20 GB).

D. Configuración de la máquina virtual

Se configurará la máquina virtual accediendo a la configuración avanzada en VirtualBox. Esto incluye habilitar el soporte para virtualización por hardware en la pestaña "Sistema", seleccionar el adaptador de red adecuado (NAT o Bridge) y asociar la imagen ISO de la distribución como medio de arranque en la sección "Almacenamiento".

- Antes de iniciar, acceda a la configuración avanzada de la máquina virtual.
- En la pestaña "Sistema", habilite el soporte para virtualización por hardware si está disponible.
- En "Red", seleccione el adaptador NAT o Bridge según las necesidades del análisis de red.
- En "Almacenamiento", asocie la imagen ISO de la distribución seleccionada como medio de arranque.

E. Instalación del sistema operativo

Se inicia la máquina virtual y se siguen los pasos del asistente de instalación para configurar la distribución de Linux.

- Inicie la máquina virtual e instale la distribución siguiendo los pasos del asistente de instalación.
- Configure un nombre de usuario y contraseña seguros, y limite los servicios habilitados por defecto durante la instalación para optimizar el entorno de pruebas.
- Una vez instalado, actualice el sistema operativo ejecutando los comandos correspondientes, como: sudo apt update && sudo apt upgrade -y

F. Configuración de las herramientas de seguridad

Para implementar las herramientas de seguridad en el proyecto, se siguieron los pasos detallados a continuación. Esto permitió integrar Shodan para el análisis de vulnerabilidades, realizar pruebas de rendimiento, y reforzar la configuración del sistema operativo para mitigar riesgos.

1) Actualización del Sistema: Se ejecuta el comando sudo apt update, esto garantiza que los repositorios del sistema estén actualizados y preparados para la instalación de paquetes.

```
┌─[✗]─[saraduque@parrot]─[~]
└──╼ $sudo apt update
Obj:1 https://deb.parrot.sh/parrot lory InRelease
Obj:2 https://deb.parrot.sh/direct/parrot lory-security InRelease
Obj:3 https://deb.parrot.sh/parrot lory-backports InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se pueden actualizar 157 paquetes. Ejecute «apt list --upgradable» para verlos.
```

2) Configuración de shodan: Se crea una cuenta en shodan para obtener una clave API, se comprueba si shodan esta instalado shodan version, si no esta instalado utilice pip install shodan, se hace la inicialización de shodan ingresando la API KEY para autenticar la conexión con el servidor de shodan con el comando shodan init ‘mi_api_key’.

```
┌─[✗]─[saraduque@parrot]─[~]
└──╼ $shodan host 8.8.8.8
8.8.8.8
Hostnames:              dns.google
City:                   Mountain View
Country:                United States
Organization:           Google LLC
Updated:                2024-12-05T11:00:18.619519
Number of open ports:   2

Ports:
     53/tcp
     53/udp
    443/tcp
        |-- SSL Versions: -SSLv2, -SSLv3, -TLSv1, -TLSv1.1, TLSv1.2, TLSv1.3
```

3) Creación y ejecución de scripts de análisis: Se crea el archivo touch shodan_analysis.py y se edita nano shodan_analysis.py, inserte el código que realiza consultas sobre dispositivos con un servidor apache y ejecuta búsquedas avanzadas con shodan y ejecuta con el comando python3 shodan_analysis.py.

```python
import shodan

SHODAN_API_KEY = 'EGYetoe97AyEiexORNMq73zxzJBw9C4p'

api = shodan.Shodan(SHODAN_API_KEY)

query = 'apache'
page_size = 100
max_results = 10

total_results = api.count(query)['total']
print(f"Número total de resultados estimados: {total_results}")

results_shown = 0

for page in range(1, (total_results // page_size) + 2):
    try:
        results = api.search(query, page=page)

        for result in results['matches']:
            if results_shown < max_results:
                print(f"IP: {result['ip_str']}")
                print(f"Puerto: {result['port']}")
                print(f"Data: {result['data']}")
                results_shown += 1
            else:
                break

        if results_shown >= max_results:
            break

    except shodan.APIError as e:
        print(f"Error en la página {page}: {e}")

print("Fin de la búsqueda")

```

Modificamos el Script para buscar dispositivos webcampXP, ajustando los parámetros de búsqueda. Ejecute el archivo nuevamente tras guardar los cambios.

4) Comandos adicionales con shodan: Explore diversas funcionalidades de shodan utilizando comandos.

```
shodan host 8.8.8.8
shodan count apache
shodan info
shodan myip
shodan radar
shodan scan submit 200.122.209.14
shodan stats apache
shodan host 200.122.209.14
```

5) Instalación y pruebas de rendimiento con sysbench: Instale la herramienta sysbench sudo apt install sysbench, realice pruebas de rendimiento de la CPU sysbench cpu –cpu-max-prime=20000 run.

6) Creación de un script de tiempos de consulta: Se crea un archivo llamado consulta_shodan.py con touch consulta_shodan.py y edita el archivo con nano consulta_shodan.py, se inserta el código para medir el tiempo de respuesta de las consultas a shodan, se ejecuta el script con python3 consulta_shodan.py.

```python
import shodan
import time

SHODAN_API_KEY = 'xcWxcPC2dEz81ZKn5ZpRVKpy7uBCfcUc'
api = shodan.Shodan(SHODAN_API_KEY)

def medir_tiempo_consulta(query):
    inicio = time.time()
    try:
        resultados = api.search(query)
    except shodan.APIError as e:
        print(f"Error en la consulta: {e}")
    fin = time.time()
    return fin - inicio

query = 'apache'
tiempo_respuesta = medir_tiempo_consulta(query)
print(f"Tiempo de respuesta: {tiempo_respuesta:.2f} segundos")
```

7) Configuración de SSH: Se edita el archivo de configuración de SSH para deshabilitar el acceso root con el comando sudo nano /etc/ssh/sshd_config y se cambia PermitRootLogin no, se reinicia el servicio SSH sudo systemctl restart ssh.

8) Configuración de Firewall con UFW: Se instala y habilita UFW con los comandos sudo apt-get install ufw, sudo ufw enable, se configura reglas de acceso y restricciones con los comandos sudo ufw allow ssh, sudo ufw deny 80, sudo ufw default deny incoming, sudo ufw default deny outgoing.

```
┌──(mi_entorno)─(saraduque㉿kali-linux)-[~]
└─$ sudo ufw allow ssh
Rules updated
Rules updated (v6)

┌──(mi_entorno)─(saraduque㉿kali-linux)-[~]
└─$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)

┌──(mi_entorno)─(saraduque㉿kali-linux)-[~]
└─$ sudo ufw default deny outgoing
Default outgoing policy changed to 'deny'
(be sure to update your rules accordingly)

┌──(mi_entorno)─(saraduque㉿kali-linux)-[~]
└─$ sudo systemctl disable avahi-daemon
```

9) *Deshabilitación de servicios innecesarios:* Servicios no esenciales para reducir la superficie de ataque con los comandos sudo systemctl disable avahi-daemon, sudo systemctl disable cups.

V. DISEÑO DE EXPERIMENTOS

El protocolo de experimentación se diseñó para evaluar de manera segura y eficiente los efectos de diferentes distribuciones de Linux y la implementación de hardening sobre los tiempos de respuesta en un entorno controlado. Este proceso incluyó estrategias específicas para garantizar la validez de los datos y la reproducibilidad de los resultados, utilizando herramientas confiables para recopilación, procesamiento y análisis.

A. Análisis de varianza (ANOVA)

Se aplicó para evaluar si las diferencias en los tiempos de respuesta entre las distribuciones y las configuraciones (con y sin hardening) eran estadísticamente significativas. ANOVA se seleccionó por su capacidad de comparar múltiples grupos de datos.

B. Validación de datos y control de variables independientes

Se mantuvieron constantes factores como la configuración de hardware virtual, el número de pruebas por distribución y los parámetros de consulta a Shodan.

VI. RESULTADO

PR(>F) (Valor p):

Es el valor de probabilidad asociado con la estadística F. Indica la probabilidad de obtener un valor F tan grande como el observado si la hipótesis nula es cierta (es decir, si no hay diferencia entre los grupos).

C(Distribucion): p = 0.307469, lo que es mayor que 0.05. Esto significa que no hay suficiente evidencia para rechazar la hipótesis nula de que las distribuciones no afectan significativamente los tiempos de respuesta. En otras palabras, la distribución de Linux no tiene un efecto significativo sobre los tiempos de respuesta.

C(Hardening): p = 0.059533, lo que está cerca de 0.05. Esto sugiere que el hardening podría tener un efecto significativo sobre los tiempos de respuesta, pero debido a que el valor p es ligeramente mayor que 0.05, la evidencia no es completamente concluyente. Sin embargo, podemos decir que el hardening tiende a afectar significativamente los tiempos de respuesta.

```
Datos del Experimento:
  Distribucion      Hardening  Tiempo
0         Kali  Sin Hardening    0.55
1         Kali  Con Hardening    1.38
2      Kodashi  Sin Hardening    1.20
3      Kodashi  Con Hardening    2.21
4       Parrot  Sin Hardening    0.71
5       Parrot  Con Hardening    2.56

Resultado del ANOVA:
                   sum_sq   df          F    PR(>F)
C(Distribucion)  0.66760  2.0   2.252362  0.307469
C(Hardening)     2.26935  1.0  15.312753  0.059533
Residual         0.29640  2.0        NaN       NaN
```

Fig 1. Tabla ANOVA

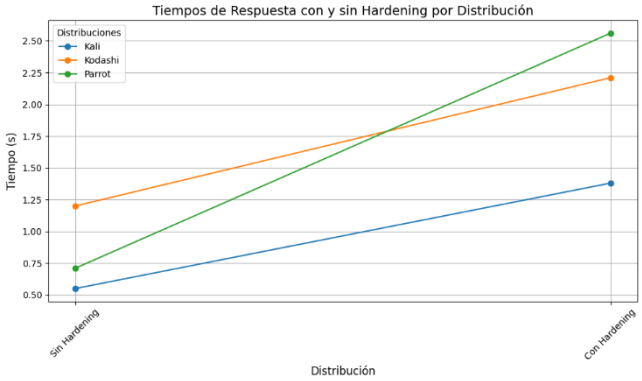

Fig 2. Gráfica comparativa entre el tiempo que gasta cada distribución.

VII. CONCLUSIONES

No hay evidencia suficiente para decir que las distribuciones (Kali, Kodashi, Parrot) afectan significativamente los tiempos de respuesta. El valor p de 0.307469 es mayor que 0.05, lo que indica que las diferencias en los tiempos de respuesta entre las distribuciones no son estadísticamente significativas.

El hardening parece tener un efecto significativo sobre los tiempos de respuesta, ya que el valor p es cercano a 0.05 (p

= 0.059533). Aunque no es completamente concluyente, indica que aplicar hardening podría aumentar el tiempo de respuesta en las aplicaciones.

Dado que no se muestra un valor p para la interacción entre la distribución y el hardening, se puede suponer que no se exploró explícitamente en el modelo, pero los resultados de las variables individuales ya sugieren que el hardening es el principal factor que afecta el rendimiento.

VIII. BIBLIOGRAFÍA

[1] "Documentation – Oracle VirtualBox". Oracle VirtualBox. Accedido el 6 de diciembre de 2024. [En línea].
Disponible: https://www.virtualbox.org/wiki/Documentation

[2] "Kali Docs | Kali Linux Documentation". Kali Linux. Accedido el 6 de diciembre de 2024. [En línea]. Disponible: https://www.kali.org/docs/

[3] "ParrotOS Documentation | ParrotOS Documentation". Parrot Security. Accedido el 6 de diciembre de 2024. [En línea]. Disponible: https://parrotsec.org/docs/

[4] "GitHub - WMAL/Linux-Kodachi: Linux Kodachi is a security-focused operating system designed for users who value privacy, anonymity, and a secure computing experience. Developed by Warith Al Maawali, Kodachi provides all the tools necessary for anonymous online activities while maintaining ease of use." GitHub. Accedido el 6 de diciembre de 2024. [En línea]. Disponible: https://github.com/WMAL/Linux-Kodachi

[5] "Shodan Developer". Shodan Developer. Accedido el 6 de diciembre de 2024. [En línea]. Disponible: https://developer.shodan.io/api

[6] "SPSS Statistics Subscription - Classic". IBM - United States. Accedido el 6 de diciembre de 2024. [En línea]. Disponible: https://www.ibm.com/docs/es/spss-statistics/saas?topic=sales-anova-table