Proyecto de Propuesta de Prevención en Pentesting

Índice

1. Introducción

Objetivo: Este informe tiene como finalidad identificar vulnerabilidades en una máquina virtual (Metasploitable y BeeBox) y el sitio web bWAPP para fortalecer su seguridad.

Alcance: La evaluación abarca análisis de puertos, servicios y configuraciones en sistemas vulnerables, así como el análisis de aplicaciones web. Se trabajó con máquinas virtuales en entornos controlados sin comprometer servicios críticos.

2. Enfoque y estrategia

Metodología General: Se aplicó un enfoque de caja negra utilizando herramientas como **Nmap** para identificar puertos y servicios activos, **Nikto** para analizar vulnerabilidades web y **Gobuster/Dirb** para descubrir directorios expuestos.

Enfoque diferenciado:

- **Infraestructura de Red y Servidores:** Identificación de servicios con versiones desactualizadas y configuraciones inseguras.
- **Aplicación Web:** Búsqueda de vulnerabilidades como XSS, archivos sensibles accesibles y configuraciones débiles.

3. Fases del Pentesting

1. Escaneo y Enumeración de Red:

- **Herramientas Usadas:** Nmap, whois, nslookup

- **Objetivo:** Descubrir servicios activos y su versión. Ejemplo: FTP (ProFTPD 1.3.1), Apache 2.2.8 y PostgreSQL desactualizados.

2. Evaluación del sitio web:

- **Herramientas Usadas:** Nikto, Gobuster, Dirb
- **Objetivo:** Detectar archivos expuestos (README, INSTALL.txt) y configuraciones inseguras como directorio de **phpMyAdmin** accesible y métodos HTTP inseguros.

4. Vulnerabilidades detectadas

1. **Puertos con Servicios Inseguros:**
   - FTP (ProFTPD 1.3.1) y Telnet transmiten datos sin cifrar.
   - SSH (OpenSSH 4.7) susceptible a fuerza bruta y vulnerabilidades conocidas.
2. **Software desactualizado:**
   - Apache HTTP 2.2.8 y PHP 5.2.4 alcanzaron su fin de soporte.
3. **Configuraciones web débiles:**
   - Falta de encabezados X-Frame-Options y X-Content-Type-Options.
   - Módulo TRACE habilitado (XST).
4. **Archivos y Directorios Sensibles Exposiciones:**
   - Archivos **README** , **INSTALL.txt** y phpMyAdmin accesibles.

5. Propuesta de prevención

1. **Actualización de Sistemas y Servicios:**
   - Migrar a versiones recientes de Apache, PHP y OpenSSH.
2. **Mejoras en Configuraciones Web:**
   - Configurar encabezados X-Frame-Options y X-Content-Type-Options.
   - Deshabilitar el método TRACE.
3. **Acceso Seguro a Servicios:**
   - Implementar restricciones de IP para phpMyAdmin y otros servicios críticos.
   - Usar firewalls (iptables/UFW) para controlar accesos no autorizados.
4. **Desarrollo Seguro:**
   - Capacitar al equipo de desarrollo en prácticas seguras.

6. Propuesta de Mitigación

1. **Aplicación de Parches de Seguridad:**
   - Actualizar servicios como FTP, SSH y Apache a versiones seguras.
2. **Configuraciones Seguras:**
   - Deshabilitar servicios inseguros como Telnet.

    - Asegurar phpMyAdmin con autenticación y acceso limitado.
3. **Cifrado de Comunicaciones:**
    - Implementar SSL/TLS para proteger servicios como HTTP, SMTP y FTP.
4. **Reducción de la superficie de ataque:**
    - Desactivar servicios no utilizados.

7. Análisis de mitigación

- **Actualización de software:** Elimina la posibilidad de explotación de vulnerabilidades conocidas.
- **Encabezados de Seguridad:** Protegen contra ataques de secuestro de sesión (Clickjacking) y XSS.
- **Acceso Controlado:** La restricción de IP y autenticación adicional reducen los riesgos de acceso no autorizado.
- **Cifrado SSL/TLS:** Mejora la confidencialidad y seguridad de las comunicaciones.

8. Impacto potencial

La aplicación de estas medidas tendrá un impacto significativo en la seguridad del sistema:

1. **Reducción de Riesgos:** Evita que los atacantes exploten vulnerabilidades comunes en servicios obsoletos.
2. **Protección de Datos Sensibles:** Configuraciones seguras previenen el acceso a archivos críticos.
3. **Disminución de la Superficie de Ataque:** Servicios innecesarios desactivados reducen vectores de ataque.

9. Conclusión

El análisis mostró que la combinación de servicios desactualizados, configuraciones débiles y archivos expuestos presentan riesgos significativos. Implementar actualizaciones periódicas, configuraciones seguras y restricciones de acceso permitirá fortalecer la seguridad del sistema. Además, la educación continua del equipo de desarrollo y monitoreo constante son clave para mantener un entorno protegido a largo plazo.