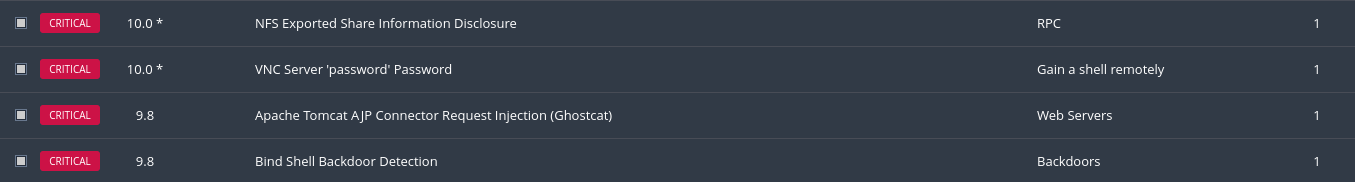
**Task**: effettuare scansione completa sulla macchina target Metasploitable e scegliere da un minimo di 2 ad un massimo di 4 vulnerabilità di tipo **critical/high** implementando delle azioni di rimedio. Infine rifare uno scan di verifica per dimostrarne l’efficacia.

Di seguito le vulnerabilità scelte:

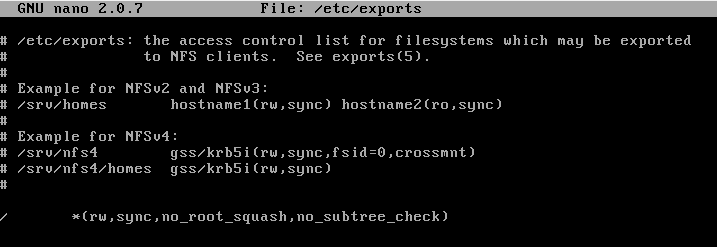


1. **NFS Exported Share Information Disclosure**

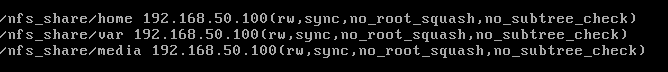
NFS (Network File System) è un protocollo di rete che permette la condivisione di file e risorse attraverso una rete. Questa vulnerabilità indica che almeno una delle condivisioni NFS esportate dal server remoto può essere montata dall'host di scansione. Ciò significa che un potenziale attaccante potrebbe sfruttare questa vulnerabilità per leggere (e possibilmente scrivere) file sul server remoto.

Le possibili soluzioni da adottare sono due:

* Possiamo modificare le impostazioni di configurazione NFS nel file /etc/exports restringendo le cartelle condivisibili e limitando allo stesso tempo le macchine che possono accedere.



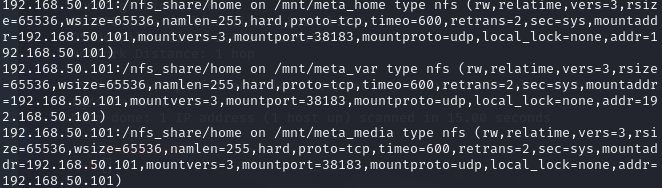
Come possiamo vedere, il file riporta che tutte le directory sono condivisibili e che tutti possono accedere a tutte le share. Modifichiamo il file specificando gli host autorizzati (ad esempio la nostra Kali Linux) e le cartelle che vogliamo siano condivise (ne ho create tre tra quelle comuni che spesso vengono condivise tramite NFS). Potremmo inoltre specificare anche le opzioni di montaggio per controllare le autorizzazioni di accesso.



Se vogliamo poi montare le cartelle condivisibili sul nostro client Kali, possiamo usare il comando:

**sudo mount -t nfs <indirizzo\_server>:<percorso\_condivisione> /mnt/nfs\_share,**

specificando Ip server, percorso di condivisione NFS e directory di mount creata. Tramite il comando **mount** senza argomenti, possiamo verificarne il corretto montaggio.

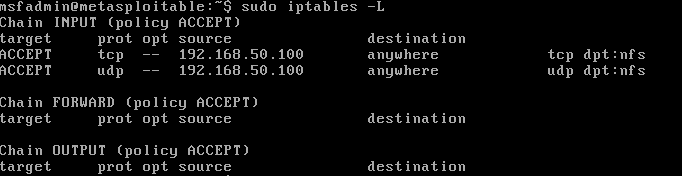


* Possiamo creare delle FW rules addizionali per gestire il traffico e le connessioni sulla porta 2049 del protocollo NFS.

Tramite il comando:

**sudo iptables –A INPUT –p tcp –dport 2049 -s <IP client> –j ACCEPT** consentiremo l’accesso solamente al nostro client, specificando protocollo, porta, ip sorgente ed azione.

Verifichiamo il corretto inserimento delle rules tramite il comando **sudo iptables -L**

****

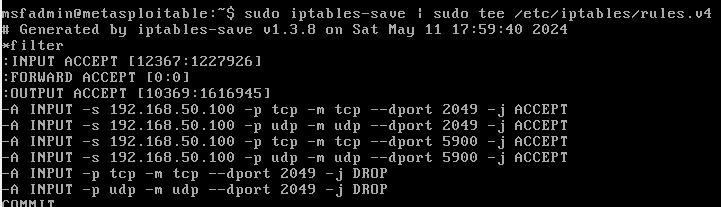
Potremmo inoltre aggiungere anche un’altra regola di input di default per bloccare tutto il traffico non autorizzato tramite il comando:

**sudo iptables -A INPUT -p tcp –dport2049 -j DROP**

Assicuriamoci infine di salvarle in un file (ad esempio /etc/iptables/rules.v4) così che rimangano persistenti anche dopo un riavvio della macchina, possiamo farlo tramite il comando

**sudo iptables-save > /etc/iptables/rules.v4** oppure

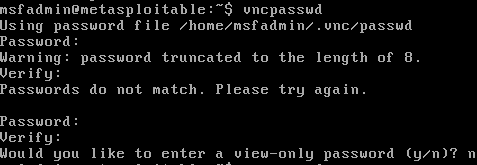
**sudo iptables-save | sudo tee /etc/iptables/rules.v4**

****

(Le rules per la porta 5900 le vedremo nella prossima vulnerabilità)

1. **VNC Server “Password” Password**  
   Questa vulnerabilità indica che il server VNC (Virtual Network Computing) remoto è configurato con una password debole("password"). Questo rende il sistema vulnerabile a potenziali attacchi da parte di utenti non autorizzati che potrebbero sfruttare la debolezza per ottenere il controllo del sistema.

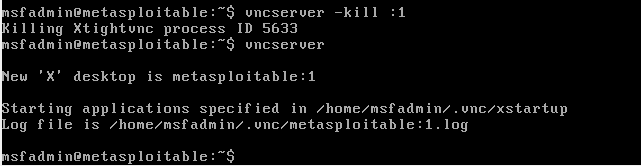
Per risolvere questa vulnerabilità, configuriamo il servizio VNC con una password più forte tramite il comando **vncpasswd**, assicurandoci di scegliere una password sicura e complessa.



Sarà poi necessario riavviare il servizio VNC manualmente per applicare le modifiche, causa versione datata della nostra Ubuntu (8.04). Per fare ciò dovremo terminare ogni sessione VNC attiva tramite i numeri di display associati. Per prima cosa troviamo i numeri di display tramite il comando **ps aux | grep Xtightvnc** (in questo caso 0 in quanto le avevo già killate precedentemente)

****

per poi arrestare le sessioni con **vncserver -kill :<display\_number>**.



Infine riavviamo il servizio usando **vncserver**.

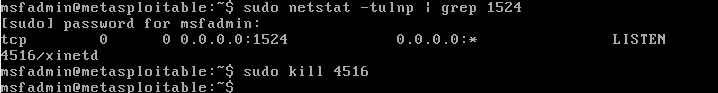
Ho immesso poi un FW rule aggiuntiva per gestire il traffico sulla porta di questo servizio (5900) (figura alla vulnerabilità 1)

1. **Bind Shell Backdoor Detection**

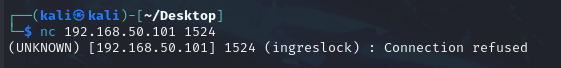
Questa vulnerabilità indica la presenza di una backdoor su una porta TCP specifica (1524). Una bind shell è un tipo di backdoor che apre una porta sulla macchina remota e ascolta le connessioni in arrivo, consentendo a un attaccante di ottenere l'accesso a un prompt di comando remoto senza autenticazione.

Per rimuovere la backdoor possiamo terminare il processo associato a quella porta.

Utilizziamo il comando **netstat -tulnp** per trovare il PID del processo in ascolto sulla porta 1524 per poi terminarlo tramite il comando **-kill**



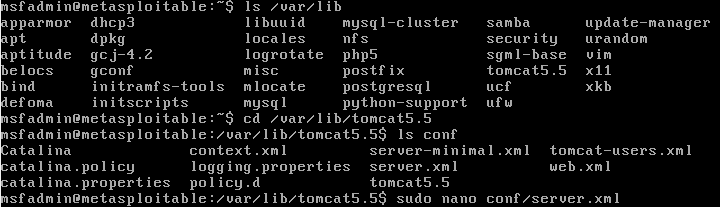
Possiamo inoltre verificare anche dal terminale di Kali come non sia più possibile connettersi a quella porta tramite il comando **nc**

****

1. **Apache Tomcat AJP Connector Request Injection (Ghostcat)**

Questa vulnerabilità riguarda il server Apache Tomcat e può consentire a un attaccante di eseguire codice arbitrario in modo remoto o causare un DoS, sfruttando una debolezza nell'implementazione del protocollo AJP (Apache JServ Protocol).

La soluzione sta nell’aggiornare la configurazione AJP per richiedere l'autorizzazione, modificando il file di configurazione di Apache Tomcat, **server.xml**, aggiungendo l’autorizzazione richiesta.



Con **ls /var/lib** trovo la versione di tomcat installata **(tomcat5.5)**

Spostandomi su quella directory, trovo il file server.xml tramite **ls conf** (lista)

Infine, accedo al file con l’editor nano.

La configurazione che ci interessa avrà una struttura simile a questa

**<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />**

Una volta trovata, al suo interno aggiungiamo la configurazione

**secretRequired="required"** o **secretRequired=”true”**

per richiedere un accesso segreto all’AJP



Una volta apportate le modifiche, salviamo e chiudiamo il file.

Così facendo la configurazione di Apache Tomcat richiederà l'autorizzazione per l'accesso al servizio AJP, contribuendo a mitigare il rischio associato alla vulnerabilità.

Per dimostrare l’efficacia delle nostre azioni di rimedio, andiamo ad effettuare una nuova scansione tramite Nessus della nostra macchina Metasploitable.

Immagine che contiene testo, schermata, software, Software multimediale

Descrizione generata automaticamente

La vulnerabilità di Apache Tomcat persiste nonostante la nostra implementazione, per risolverla completamente dovrebbe essere necessario scaricare la versione più recente (versioni 7.0.100, 8.5.51, 9.0.31 o successive) dal sito ufficiale.