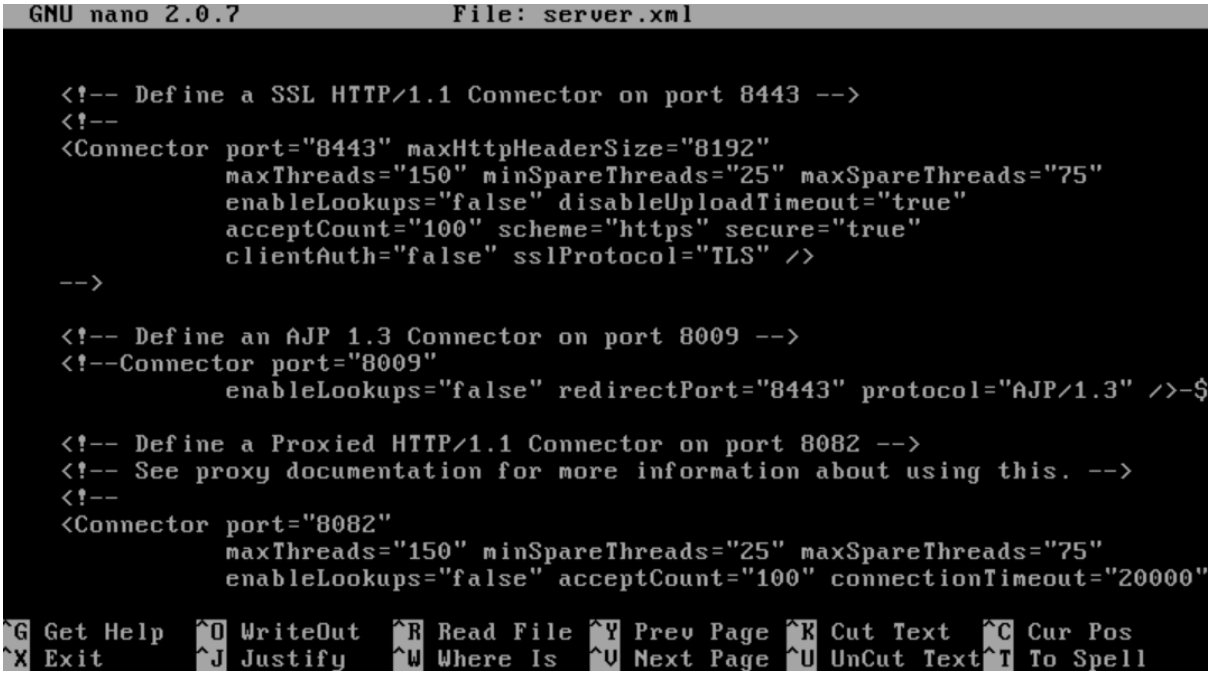
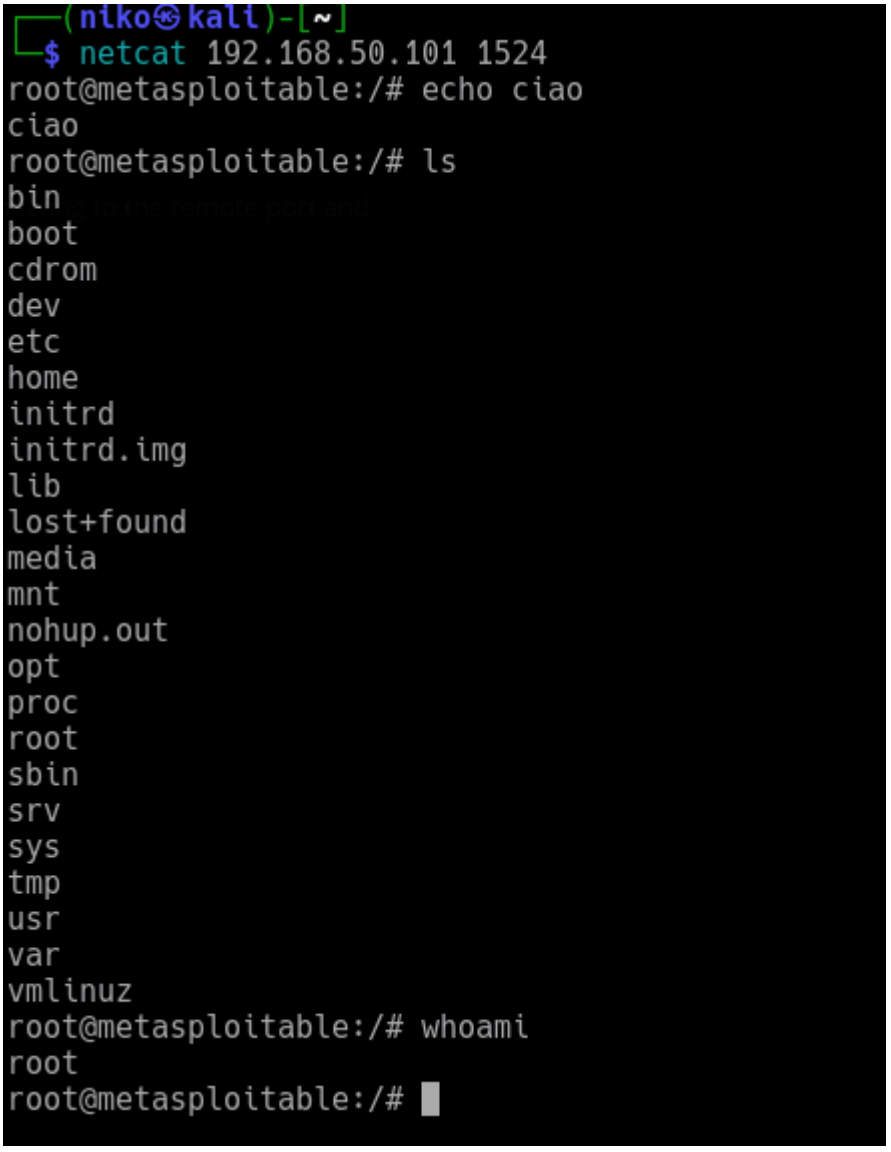
Risoluzione di alcune criticità trovate con Nessus

Apache Tomcat AJP Connector Request Injection (Ghostcat): È stata rilevata una vulnerabilità di lettura/inclusione di file nel connettore AJP. Un utente malintenzionato remoto e non autenticato potrebbe sfruttare questa vulnerabilità per leggere i file delle applicazioni Web da un server vulnerabile. Nei casi in cui il server vulnerabile consenta il caricamento di file, un utente malintenzionato potrebbe caricare codice JavaServer Pages (JSP) dannoso all'interno di una varietà di tipi di file e ottenere l'esecuzione di codice in modalità remota (RCE).

Andiamo a commentare questo pezzo di codice nel file server.xml cosi disabiliteremo il servizio AJP connector, poi effettuiamo un restart di Tomcat. Un'altra soluzione sarebbe quella di aggiungere delle credenziali di autenticazione nel campo "requiredSecret attribute"



Bind Shell Backdoor Detection: Una shell è in ascolto sulla porta remota senza che sia richiesta alcuna autenticazione. Un utente malintenzionato può utilizzarlo collegandosi alla porta remota e inviando direttamente i comandi. Collegandoci in ascolto tramite il comando “netcat” possiamo verificare che riusciamo ad entrare nella macchina senza autenticarci sulla porta 1524.



Da qui killiamo il processo relativo alla porta 1524

```
┌──(niko㉿kali)-[~]
└─$ netcat 192.168.50.101 1524
root@metasploitable:/# fuser -k -n tcp 1524
1524/tcp:             4312  6129
```

Facciamo una scansione per verificare che non compaia la porta ed effettivamente non compare più.

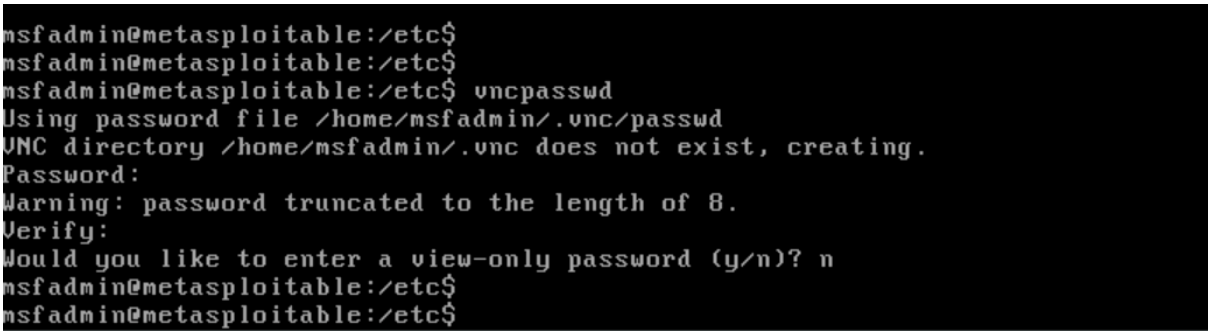
```
┌──(niko㉿kali)-[~]
└─$ nmap 192.168.50.101
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-28 10:22 CEST
Nmap scan report for 192.168.50.101
Host is up (0.00094s latency).
Not shown: 983 closed tcp ports (conn-refused)
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1099/tcp open  rmiregistry
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
```

C’è un piccolo problema però, in questo modo quando andremo a riavviare la macchina il servizio ripartirà, quindi è solo una soluzione temporanea, la soluzione migliore sarebbe bloccare il traffico su quella porta attraverso il firewall (io l’ho già utilizzato successivamente)

VNC Server 'password' Password: Il server VNC in esecuzione sull'host remoto è protetto con una password debole. Nessus è

riuscito ad accedere utilizzando l'autenticazione VNC e una password "password". Un utente malintenzionato remoto e non autenticato potrebbe sfruttare questa situazione per assumere il controllo del sistema.

Con il comando “vncpsswd” ci verrà chiesto di inserire una password ma questo non risolverà il problema perchè connettendoci a vnc tramite la porta 5900 ci darà automaticamente permessi di root, quindi per cambiare effettivamente la password ho utilizzato il comando “sudo” e modificato la password contenuta all’interno del file /.vnc/passwd



NFS Exported Share Information Disclosure: Almeno una delle share NFS esportate dal server remoto potrebbe essere montata dall'host di scansione. Un utente malintenzionato potrebbe essere in grado di sfruttare questo per leggere (ed eventualmente scrivere) file sull'host remoto.
IMPORTANTE: tenere presente che i privilegi di root non erano necessari per montare le condivisioni remote poiché la porta di origine per montare le condivisioni era superiore a 1024.

Facciamo un check dei servizi network che sono avviati con il comando “rpcinfo”

```
msfadmin@metasploitable:~$ rpcinfo -p 192.168.50.101
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp  45626  status
    100024    1   tcp  54657  status
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100003    4   udp   2049  nfs
    100021    1   udp  40492  nlockmgr
    100021    3   udp  40492  nlockmgr
    100021    4   udp  40492  nlockmgr
    100003    2   tcp   2049  nfs
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100021    1   tcp  36996  nlockmgr
    100021    3   tcp  36996  nlockmgr
    100021    4   tcp  36996  nlockmgr
    100005    1   udp  44613  mountd
    100005    1   tcp  37123  mountd
    100005    2   udp  44613  mountd
    100005    2   tcp  37123  mountd
    100005    3   udp  44613  mountd
    100005    3   tcp  37123  mountd
msfadmin@metasploitable:~$
```

Utilizzando il comando “showmount” vediamo che tutto il fylesystem è montabile/scrivibile (/*)

```
msfadmin@metasploitable:~$ showmount -e 192.168.50.101
Export list for 192.168.50.101:
/ *
msfadmin@metasploitable:~$
```

Se facessi “service rpcbind start” potrei montare il network fylesystem senza credenziali.
Per evitare ciò dobbiamo modificare i permessi di chi può accedere alle share condivise.

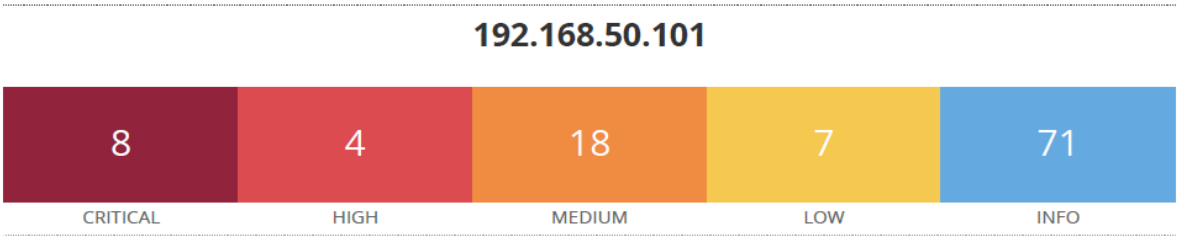
Ci sono tre file che posso modificare:

- /etc/hosts.allow: contiene gli host che hanno accesso alle cartelle condivise
- /etc/hosts.deny: contiene gli host che non hanno accesso alle cartelle condivise
- /etc/exports: contiene la lista delle cartelle condivisibili

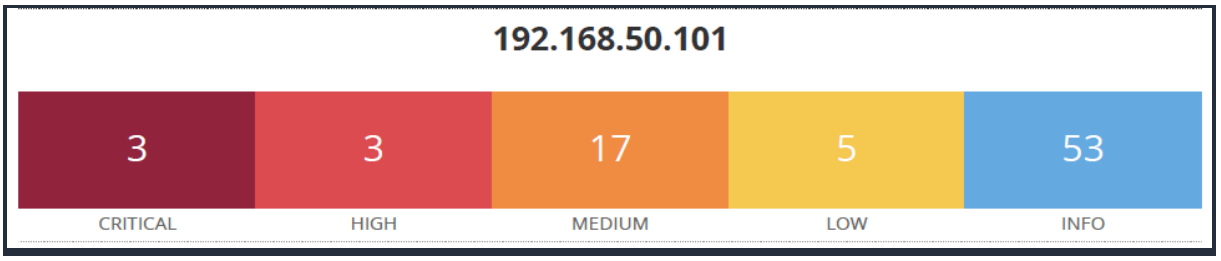
In questo caso però, io ho aggiunto una regola sul firewall per bloccare i pacchetti tcp/udp sulla porta 2049 ed anche sulla porta 111 (rpcbind) della metasploitable.

```
┌──(root㉿kali)-[/home/niko]
└─# iptables -vL --line-numbers
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 10796 packets, 2154K bytes)
num   pkts bytes target     prot opt in     out     source               destination
1       32  5024 DROP       udp  --  any    eth0    anywhere             192.168.50.101       udp dpt:nfs
2      108  6000 DROP       tcp  --  any    eth0    anywhere             192.168.50.101       tcp dpt:nfs
3       25  1140 DROP       tcp  --  any    eth0    anywhere             192.168.50.101       tcp dpt:sunrpc
4        4   336 DROP       udp  --  any    eth0    anywhere             192.168.50.101       udp dpt:sunrpc
```

REPORT INIZIALE



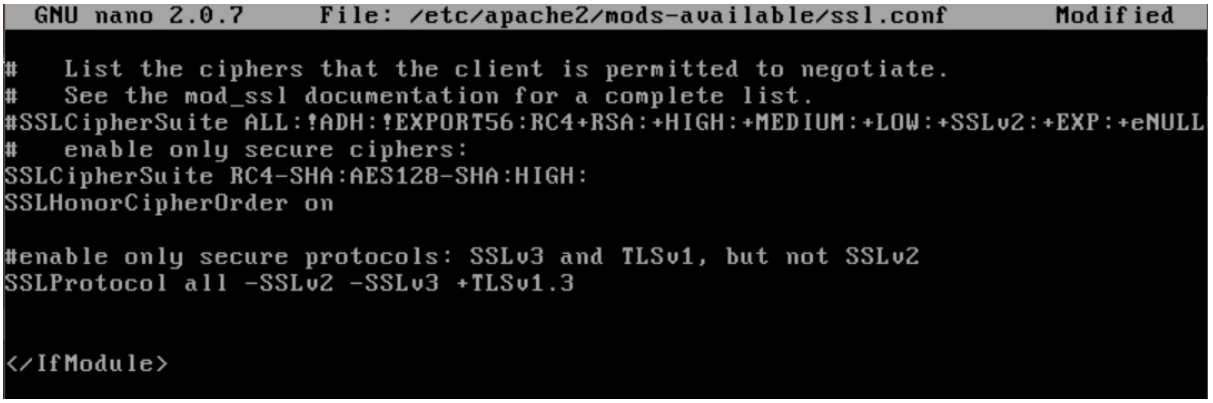
REPORT FINALE



Alla fine di queste soluzioni le criticità sono diminuite come si può vedere in modo approfondito dal file di report lasciato in allegato.

BONUS:

SSL Version 2 and 3 Protocol Detection: Il servizio remoto accetta connessioni crittografate utilizzando SSL 2.0 e/o SSL 3.0. Queste versioni di SSL sono affette da diversi difetti crittografici. La soluzione è passare a TLS ma la versione di OPENSSL su metasploitable è la 0.9 e TLSv1.1 è supportato dalla versione 1.0.1. L'unica soluzione in questo caso è aggiornare OPENSSL e di conseguenza rimuovere il protocollo SSL e abilitare TLS come mostrato sotto.



Unix Operating System Unsupported Version Detection: Quel numero di versione è basato sul TTL, basta cambiare quel valore per nascondere l'OS version. La seconda soluzione è quella di aggiornare ad una versione OS di UNIX che sia supportata attualmente.

Ho modificato il file /etc/sysctl.conf e aggiunto la seguente riga

net.ipv4.ip_default_ttl = 199

Samba Badlock Vulnerability: Un utente malintenzionato man-in-the-middle in grado di intercettare il traffico tra un client e un server che ospita un database SAM può sfruttare questa falla per

forzare un downgrade del livello di autenticazione, che consente l'esecuzione di chiamate di rete Samba arbitrarie nel contesto dell'utente intercettato, ad esempio visualizzare o modificare dati sensibili di sicurezza nel database di Active Directory (AD) o disabilitare servizi critici.
La soluzione è quella di fare l’upgrade a Samba version 4.2.11 / 4.3.8 / 4.4.2 o successive.