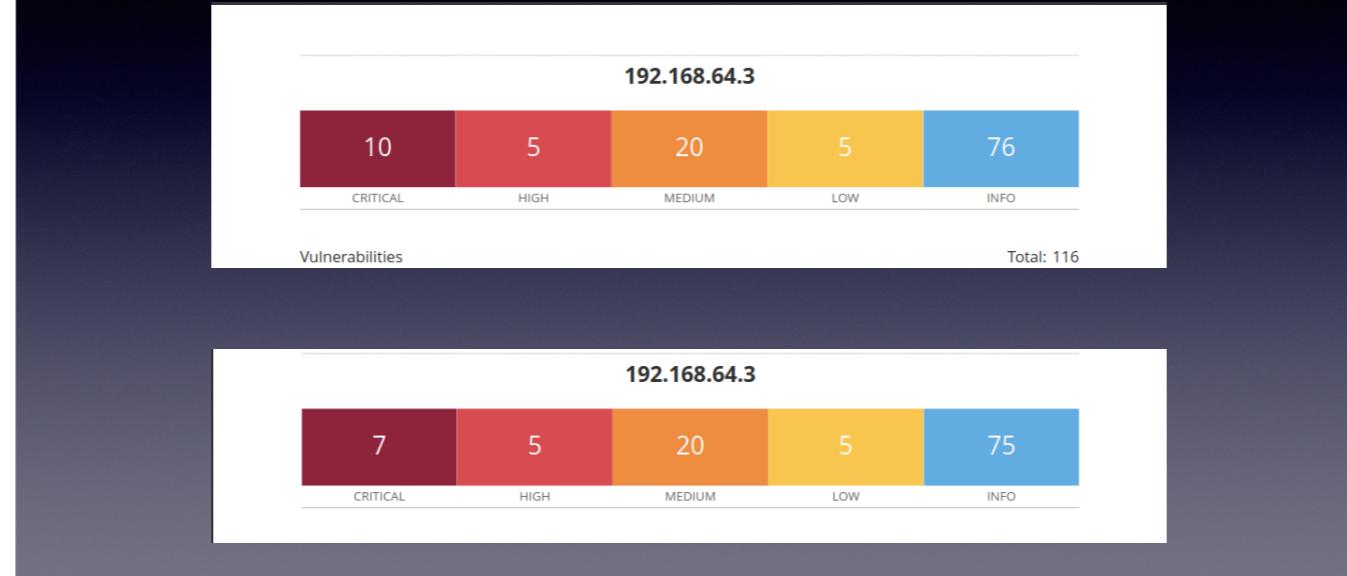


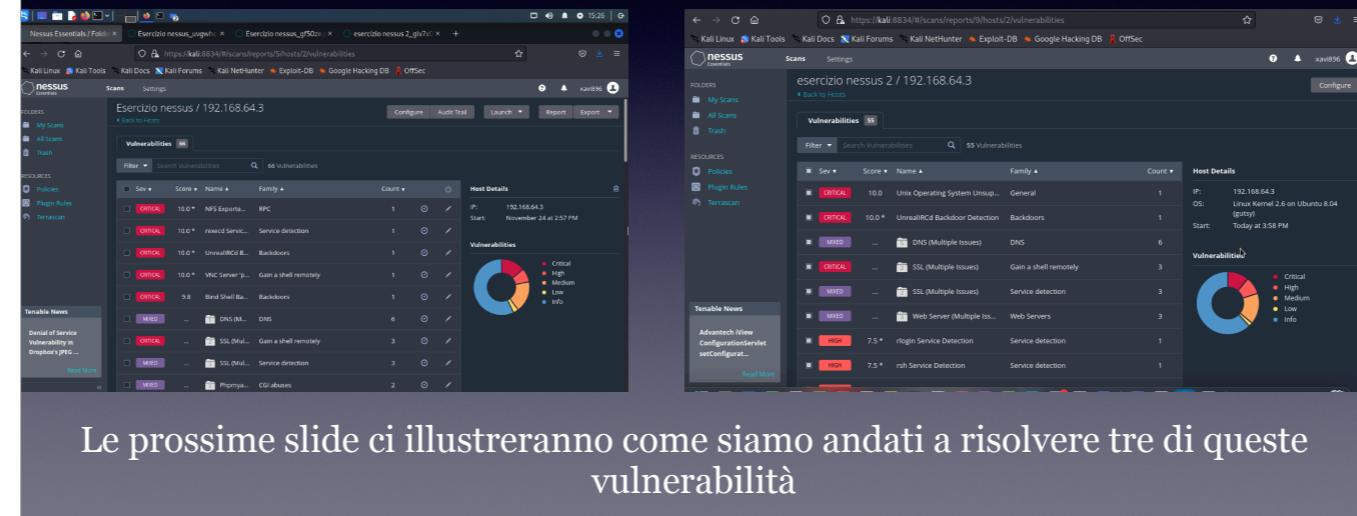
# **REPORT NESSUS**

Questo report ha come scopo di illustrare come come  
risolvere eventuali vulnerabilità riscontrate nelle scansioni  
del programma nessus.

Queste immagini riportano il risultato delle scansioni effettuate prima e dopo.



Queste immagini ci riportano due tipi di scansioni, la prima a sinistra ci illustra una scansione iniziale, dove la macchina bersaglio ci espone tutte le sue vulnerabilità. La seconda a destra ci mostra come la situazione sia cambiata dopo che abbiamo risolto tre vulnerabilità critiche.



Le prossime slide ci illustreranno come siamo andati a risolvere tre di queste vulnerabilità

## Vulnerabilità da analizzare e risolvere:

CRITICAL 10.0\* 11356 NFS Exported Share Information Disclosure

CRITICAL 10.0\* 61708 VNC Server 'password' Password

CRITICAL 9.8 51988 Bind Shell Backdoor Detection

**CRITICAL**    10.0\*    11356    NFS Exported Share Information Disclosure

**Esercizio nessus / Plugin #11356**

[Back to Vulnerabilities](#)

Hosts 1    Vulnerabilities 66    VPR Top Threats    History 2

**CRITICAL** NFS Exported Share Information Disclosure

**Description**  
At least one of the NFS shares exported by the remote server could be mounted by the scanning host. An attacker may be able to leverage this to read (and possibly write) files on remote host.

**Solution**  
Configure NFS on the remote host so that only authorized hosts can mount its remote shares.

**Output**

```
GNU nano 2.0.7           File: /etc/hosts.allow
# /etc/hosts.allow: list of hosts that are allowed to access the system.
# See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:  ALL: LOCAL #some_netgroup
#          ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "portmap" for the
# daemon name. Remember that you can only use the keyword "ALL" and IP
# addresses (NOT host or domain names) for the portmapper, as well as for
# rpc.mountd (the NFS mount daemon). See portmap(8) and rpc.mountd(8)
# for further information
#
# ALL:DEMY

ALL:DEMY

[ Read 14 lines ]  G Get Help  W WriteOut  R Read File  P Prev Page  C Cut Text  C Cur Pos
X Exit  J Justify  S Where Is  N Next Page  U UnCut Text  T To Spell
```

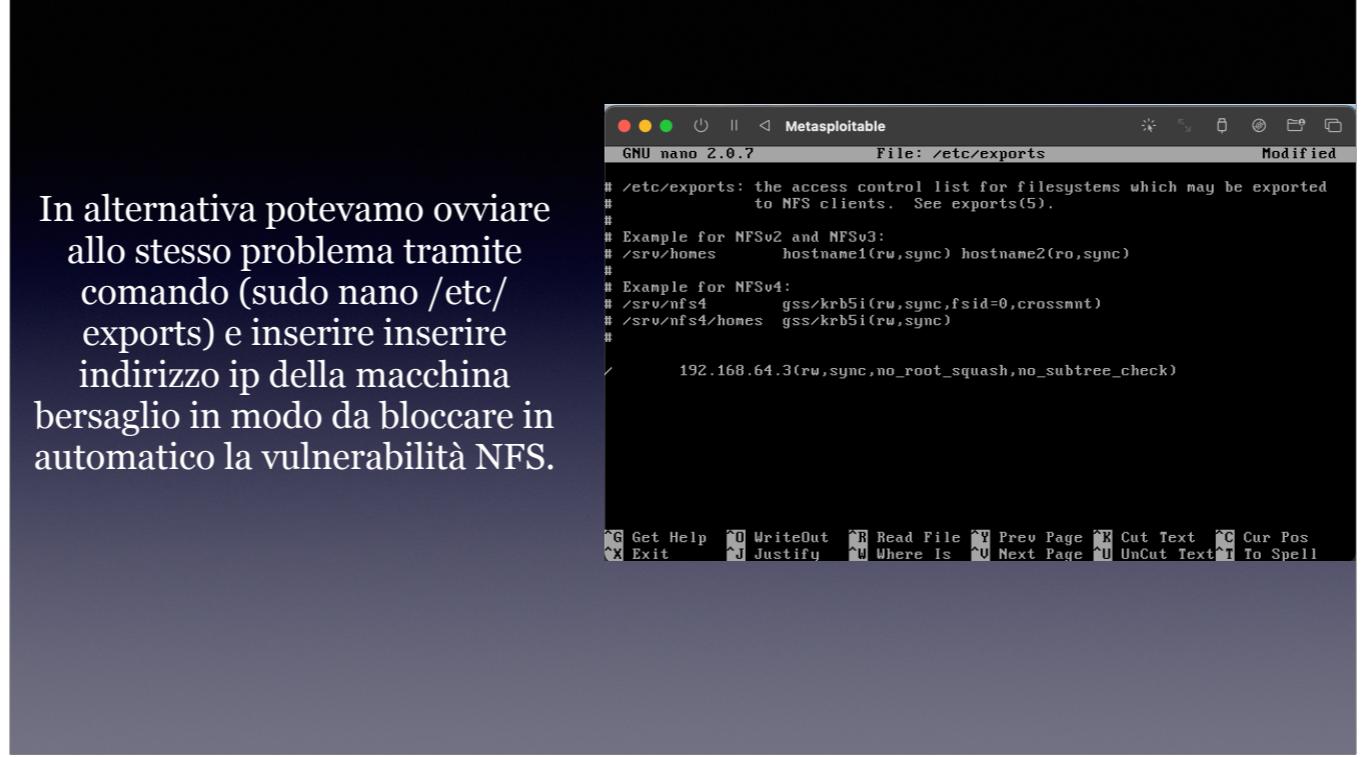
  

```
GNU nano 2.0.7           File: /etc/hosts.deny
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
# See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:  ALL: some.host.name, .some.domain
#          ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "portmap" for the
# daemon name. Remember that you can only use the keyword "ALL" and IP
# addresses (NOT host or domain names) for the portmapper, as well as for
# rpc.mountd (the NFS mount daemon). See portmap(8) and rpc.mountd(8)
# for further information.
#
# The PRAMOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
#
# ALL:ALL

[ Read 19 lines ]  G Get Help  W WriteOut  R Read File  P Prev Page  C Cut Text  C Cur Pos
X Exit  J Justify  S Where Is  N Next Page  U UnCut Text  T To Spell
```

Per risolvere questo problema sono dovuto andare nelle cartelle hosts.allow e deny, e modificare i parametri.  
 Nella sezione .allow ho inserito deny mentre nella sezione deny ho inserito all, in poche parole ho fatto l'azione opposta per poter risolvere questa vulnerabilità.

In alternativa potevamo ovviare allo stesso problema tramite comando (sudo nano /etc/exports) e inserire inserire indirizzo ip della macchina bersaglio in modo da bloccare in automatico la vulnerabilità NFS.



```
Metasploitable
GNU nano 2.0.7          File: /etc/exports      Modified

# /etc/exports: the access control list for filesystems which may be exported
#           to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes    hostname1(rw,sync) hostname2(ro,sync)
#
# Example for NFSv4:
# /srv/nfs4     gss/krb5i(rw,sync,fsid=0,crossmnt)
# /srv/nfs4/homes  gss/krb5i(rw,sync)
#
# / 192.168.64.3(rw,sync,no_root_squash,no_subtree_check)

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

The screenshot shows the Nessus interface with a critical vulnerability identified. The top bar displays "CRITICAL" in red, "10.0\*" in blue, "61708" in light blue, and "VNC Server 'password' Password". The main window title is "Esercizio nessus / Plugin #61708" and the sub-title is "Back to Vulnerabilities". The navigation bar includes "Hosts 1", "Vulnerabilities 66", "VPR Top Threats 0", and "History 2". The current view is on the "Vulnerabilities" tab, specifically the "CRITICAL" section for the "VNC Server 'password' Password" entry. The "Description" section states: "The VNC server running on the remote host is secured with a weak password. Nessus was able to login using VNC authentication and a password of 'password'. A remote, unauthenticated attacker could exploit this to take control of the system." The "Solution" section advises: "Secure the VNC service with a strong password." To the right, a terminal window titled "Metasploitable" shows a root shell session: "root@metasploitable:/home/msfadmin# vncpasswd" followed by a password prompt and verification step.

Questa vulnerabilità è stata semplice da risolvere in quanto non c'era una password di accesso e quindi la macchina attaccante poteva effettuare login molto facilmente.  
In risposta a ciò ho creato una password complessa in modo che nessun'altra macchina possa effettuare l'accesso.

The screenshot shows the Nessus interface with the following details:

- CRITICAL** 9.8 51988 Bind Shell Backdoor Detection
- Esercizio nessus / Plugin #51988
- Hosts: 1 Vulnerabilities: 66 VPR Top Threats: 0 History: 2
- Description: A shell is listening on the remote port without any authentication being required. An attacker may use it by connecting to the remote port and sending commands directly.
- Solution: Verify if the remote host has been compromised, and reinstall the system if necessary.
- Terminal output (right side):

```
nsfadmin@metasploitable:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
nsfadmin@metasploitable:~$ sudo iptables -I INPUT -p tcp --s 192.168.64.2 --dport 1524 -j DROP
nsfadmin@metasploitable:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP      tcp   --  192.168.64.2          anywhere            tcp dpt:ingreslock
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
nsfadmin@metasploitable:~$
```

Text overlay on the right side of the screenshot area:

Per risolvere il problema delle backdoor in entrata sono andato a modificare il firewall tramite il comando iptables (sudo iptables -I INPUT -p tcp IP target – dport 1524 - DROP).  
In questo modo la macchina bersaglio non accetta comunicazioni con la macchina attaccante.



In conclusione, possiamo dedurre che le scansioni hanno come scopo di mostrare eventuali vulnerabilità e risolvere quante più possibili in modo da ridurre potenziali rischi e attacchi da parte di malintenzionati.

**GRAZIE**

EPCODE