Raul Pastor

Remediation

Nella fase di remediation ho deciso di risolvere le seguenti vulnerabilità critiche evidenziate in giallo:

Sev	CVSS	VPR	EPSS	Name	Family	Count
CRITICAL	10.0			Canonical Ubuntu Linux SEoL (8.04.x)	General	1
CRITICAL	10.0 *			VNC Server 'password' Password	Gain a shell remotely	1
CRITICAL	9.8	8.9	0.9447	Apache Tomcat AJP Connector Request Injection (Ghostcat)	Web Servers	1
CRITICAL	9.8			SSL Version 2 and 3 Protocol Detection	Service detection	2
CRITICAL	9.8			Bind Shell Backdoor Detection	Backdoors	1
CRITICAL	...	...	...	SSL (Multiple Issues)	Gain a shell remotely	3

Vulnerabilità 1:

VNC Server – Password “password” (Default Credential).

Il servizio VNC esposto permette il controllo remoto del desktop (come il remote desktop di windows). È ancora impostata la password predefinita “password”, facilmente indovinabile.

Il rischio maggiore è che un attaccante prenda il totale controllo del sistema con un semplice brute force.

La soluzione in questo caso è quella di cambiare la password ‘password’ in una più sicura, così da rendere più difficile l’attacco sulla macchina.

```
msfadmin@metasploitable:~$ vncpasswd
Using password file /home/msfadmin/.vnc/passwd
VNC directory /home/msfadmin/.vnc does not exist, creating.
Password:
Warning: password truncated to the length of 8.
Verify:
Passwords do not match. Please try again.

Password:
Verify:
Passwords do not match. Please try again.

Password:
Verify:
Would you like to enter a view-only password (y/n)? n
msfadmin@metasploitable:~$ _
```

Ho utilizzato il comando ‘vncpasswd’ per settare la password in una piu complessa, nonostante troncasse la password a un massimo di 8 caratteri.
Ho comunque cercato di rendere la password poco vulnerabile includendo caratteri alfanumerici, maiuscole e caratteri speciali ‘Lin@97_u’.

Raul Pastor

Vulnerabilità 2:

Apache Tomcat AJP Connector Request Injection (Ghostcat – CVE-2020-1938).

Tomcat è un application server che serve per eseguire applicazioni web Java.
In particolare Gestisce richieste HTTP che contengono logica Java, Tiene in esecuzione applicazioni web e Comunica con altri server tramite AJP.

Il rischio qui è che un attaccante possa inviare indisturbato dei pacchetti AJP manipolati in modo da leggere o scrivere file interni del server senza la nostra approvazione oppure ottenere esecuzione di codice malevolo.

Qui ho ipotizzato se Tomcat potesse richiedere una password ad ogni tentativo di connessione, ma purtroppo la versione è troppo obsoleta per abilitare il flag secretRequired="true" nel file /etc/tomcat5.5/server.xml; servirebbe la versione 7.0 o una più recente.
In questo caso sono state aggiunte due regole firewall tramite iptables che permettono l'accesso alla porta AJP (8009) solo dalla macchina locale. Tutti i tentativi di connessione provenienti da host esterni vengono bloccati. Questo mantiene il servizio attivo, ma elimina la possibilità di sfruttare la vulnerabilità da remoto.

- sudo iptables -A INPUT -p tcp --dport 8009 -s 192.168.1.99 -j ACCEPT
- sudo iptables -A INPUT -p tcp --dport 8009 -j DROP

Vulnerabilità 3:

SSL Version 2 e Version 3 Abilitati (Protocol Obsolete).

SSL (Secure Sockets Layer) è un protocollo che serve a cifrare le comunicazioni su Internet (sostituito successivamente da TLS).

Il sistema accetta connessioni cifrate tramite SSLv2 e SSLv3. Entrambi i protocolli sono deprecati e presentano vulnerabilità critiche, permettendo attacchi MITM e potenziale decifratura del traffico.

La VM Metasploitable utilizza versioni obsolete di OpenSSL e Apache che non supportano protocolli TLS moderni, quindi ai fini del nostro test in laboratorio possiamo solo limitare l'accesso ai servizi tramite firewall, considerando però che in un ambiente reale/lavorativo è imperativo disabilitare il servizio SSL per abilitare il servizio TLS aggiornato (v1.2 o v1.3).
Limito quindi il traffico HTTPS e HTTP rispettivamente sulle porte 443 e 80 al solo host locale.

- sudo iptables -A INPUT -p tcp --dport 443 -s 192.168.1.99 -j ACCEPT
- sudo iptables -A INPUT -p tcp --dport 443 -j DROP

- sudo iptables -A INPUT -p tcp --dport 80 -s 192.168.1.99 -j ACCEPT
- sudo iptables -A INPUT -p tcp --dport 80 -j DROP

Vulnerabilità 4:

Bind Shell Backdoor Detection.

Una bind shell è una shell di sistema che viene messa in ascolto su una porta di rete, aprendo una porta TCP e permettendo il collegamento direttamente sul sistema target.

Nessus si riferisce alla Bind Shell come Backdoor perché la porta non richiede autenticazione/credenziali e non passa da SSH.

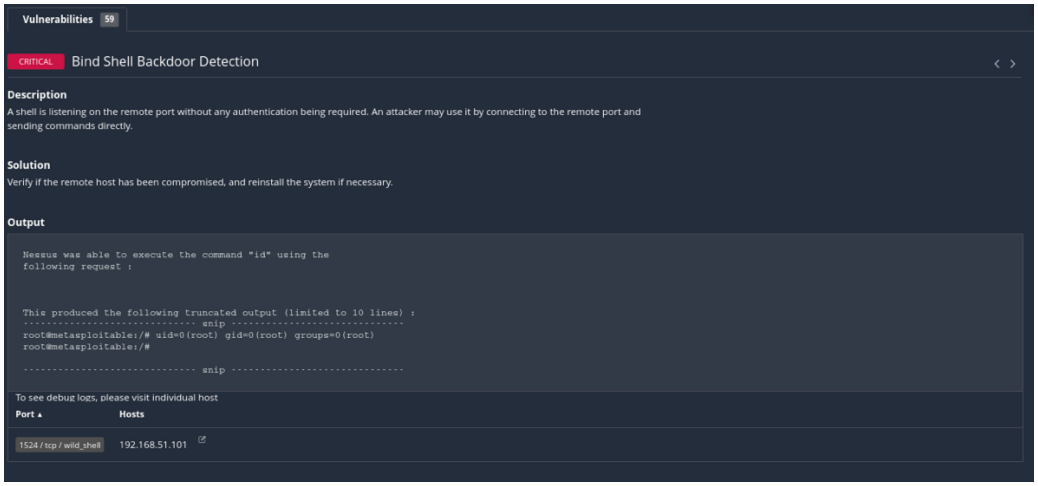
Le criticità di questa vulnerabilità è l'accesso remoto non autenticato(e persistenza) di un attaccante con la conseguente possibile esecuzione di malware e potenziale compromissione del sistema.

Raul Pastor

La soluzione migliore per questo tipo di vulnerabilità è il ripristino completo del sistema in quanto, se presente una backdoor qualcuno ha gia avuto accesso al sistema remoto e il sistema operativo potrebbe gia essere compromesso.

per scopi puramente didattici mi atterrò a bloccare la porta 1524 con un'ulteriore regola di firewall…

- sudo iptables -A INPUT -p tcp --dport 1524 -j DROP



Voglio precisare che, In questo contesto, le regole di firewall sono state applicate esclusivamente come misura di mitigazione a scopo didattico, al fine di dimostrare la capacità di contenere l'esposizione di una vulnerabilità critica mantenendo comunque attivi i servizi richiesti.
Per ciascuna delle vulnerabilità analizzate ho comunque indicato le remediation corrette da applicare in un ambiente di produzione, distinguendo chiaramente tra mitigazioni applicabili nel laboratorio e soluzioni definitive adottabili in un sistema reale.