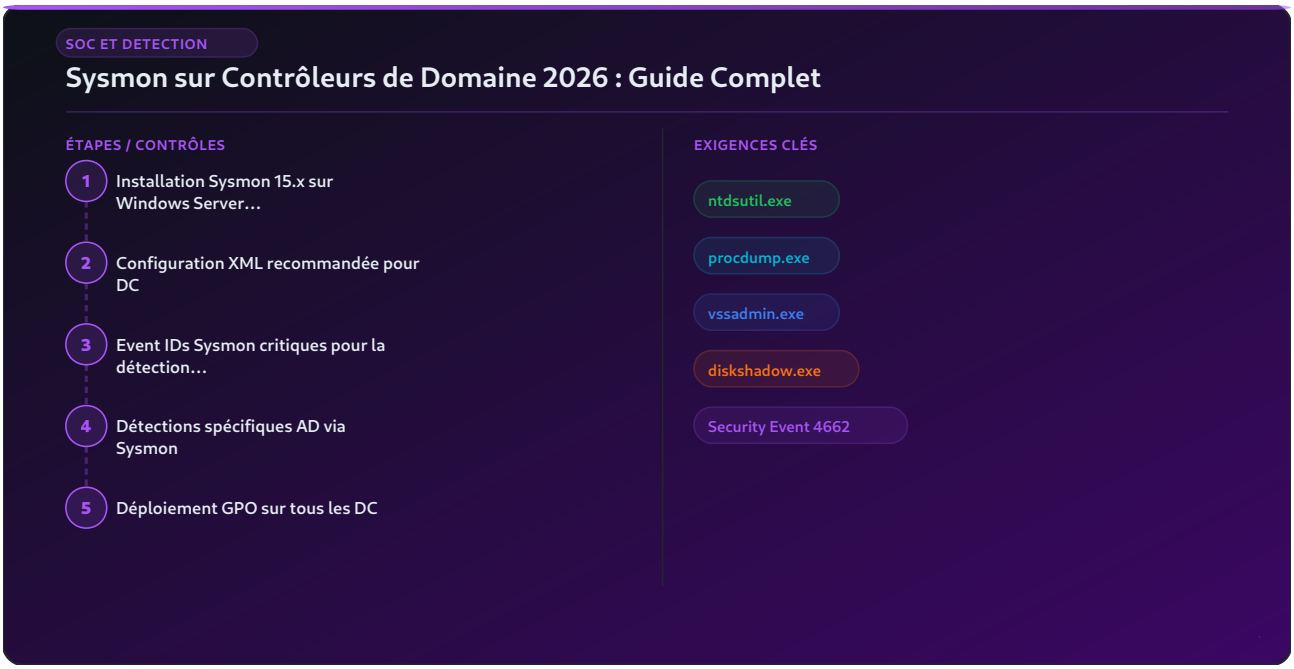
Sysmon sur Contrôleurs de Domaine : Configuration et Détection 2026

> Déployer et configurer Sysmon sur vos DC Windows Server 2025 : config SwiftOnSecurity, règles de détection AD, intégration Wazuh et Sentinel. Guide expert.

Les contrôleurs de domaine Active Directory sont la cible prioritaire de toute APT ciblant une infrastructure Windows. Mimikatz, DCSync, procdump sur LSASS, extraction de ntds.dit — ces techniques d'attaque ont en commun qu'elles laissent des traces observables si et seulement si vous disposez de la bonne télémétrie. Les journaux Windows natifs (Security Event Log) sont nécessaires mais insuffisants : ils ne capturent pas les accès processus à LSASS, les injections de DLL, ni les connexions réseau initiées par des services système compromis. Sysmon (System Monitor), développé par Sysinternals/Microsoft, comble précisément ce manque. Avec une configuration adaptée aux DC, il génère des événements de haute valeur pour détecter les techniques MITRE ATT&CK les plus fréquemment observées sur les environnements AD français en 2026 : T1003.003 (OS Credential Dumping — NTDS), T1003.001 (LSASS Memory), T1207 (Rogue Domain Controller — DCSync). Ce guide s'adresse aux RSSI et équipes SOC qui souhaitent déployer Sysmon sur leurs DC Windows Server 2025 de manière industrielle, avec une configuration XML éprouvée, un déploiement GPO, et une intégration SIEM (Wazuh, Sentinel). L'objectif : une visibilité maximale sur les attaques AD avec un overhead minimal sur les contrôleurs de domaine en production. Nous couvrons également la maintenance, les mises à jour et les pièges courants rencontrés en environnement réel.



Installation Sysmon 15.x sur Windows Server 2025

Sysmon est distribué via Sysinternals Suite par Microsoft. La version 15.x apporte des améliorations de performance notables sur Windows Server 2025, notamment la réduction de l'overhead sur les événements DNS (Event 22) qui peuvent être très fréquents sur un DC.

Téléchargement et vérification de signature

```
 # Télécharger Sysmon depuis Sysinternals
$sysmonUrl = "https://live.sysinternals.com/Sysmon64.exe"
$sysmonPath = "C:\Tools\Sysmon\Sysmon64.exe"
New-Item -ItemType Directory -Force -Path "C:\Tools\Sysmon"
Invoke-WebRequest -Uri $sysmonUrl -OutFile $sysmonPath

# Vérifier la signature Authenticode Microsoft
$sig = Get-AuthenticodeSignature -FilePath $sysmonPath
if ($sig.Status -ne "Valid" -or $sig.SignerCertificate.Subject -notlike "*Microsoft*") {
    throw "Signature invalide — ne pas installer !"
}
Write-Host "Signature valide : $($sig.SignerCertificate.Subject)"
```

Installation silencieuse avec configuration

```
 # Installation avec configuration XML DC
Start-Process -FilePath $sysmonPath -ArgumentList "-accepteula -i C:\Tools\Sysmon\sysmon-dc-confi

# Vérifier l'installation
Get-Service Sysmon64 | Select Name, Status, StartType
# Attendu : Name=Sysmon64, Status=Running, StartType=Automatic
```

Configuration XML recommandée pour DC

La configuration XML est le cœur de Sysmon. Sur un DC, elle doit être plus ciblée que sur un poste de travail : moins de bruit, mais aucune attaque critique non détectée. La configuration ci-dessous est adaptée de SwiftOnSecurity et MSTIC pour les DC Windows Server 2025.

⚡ Retour terrain

Pour un groupe de services financiers qui recevait 2 millions d'alertes SIEM par jour et en traitait 3 %, j'ai réalisé une analyse de la valeur des règles de détection sur 6 mois de logs. Résultat : 73 % des alertes provenaient de 4 règles mal calibrées. La suppression ou le réajustement de ces 4 règles a réduit le bruit de 65 % sans manquer aucun incident réel lors des 3 mois de validation. Le ROI d'un tuning SIEM est parmi les meilleurs investissements défensifs possibles.

```
<!-- Configuration Sysmon pour DC — extrait -->
<Sysmon schemaversion="4.90">
  <EventFiltering>
    <!-- Event 1: Processus suspects depuis DC -->
    <ProcessCreate onmatch="include">
      <Image condition="is">C:\Windows\System32
tdsutil.exe</Image>
      <Image condition="is">C:\Sysinternals\procdump.exe</Image>
      <Image condition="contains">mimikatz</Image>
      <CommandLine condition="contains">lsass</CommandLine>
    </ProcessCreate>

    <!-- Event 10: Accès à LSASS (signature Mimikatz) -->
    <ProcessAccess onmatch="include">
      <TargetImage condition="end with">lsass.exe</TargetImage>
      <GrantedAccess condition="is">0x1010</GrantedAccess>
      <GrantedAccess condition="is">0x1438</GrantedAccess>
    </ProcessAccess>

    <!-- Event 11: NTDS.dit copié hors emplacement normal -->
    <FileCreate onmatch="include">
      <TargetFilename condition="contains">ntds.dit</TargetFilename>
      <TargetFilename condition="not">C:\Windows\NTDS
tds.dit</TargetFilename>
    </FileCreate>
  </EventFiltering>
</Sysmon>
```

Pour une configuration complète de production, ajoutez les exclusions des processus légitimes à haut volume. Sans exclusions, un DC peut générer plusieurs milliers d'événements Sysmon par minute, saturant les journaux et masquant les vrais incidents.

Exclusions essentielles pour réduire le bruit

```
 <!-- Exclure les accès LSASS légitimes connus -->
<ProcessAccess onmatch="exclude">
  <SourceImage condition="is">C:\Windows\system32\wbem\wmiprvse.exe</SourceImage>
  <SourceImage condition="is">C:\Windows\System32\svchost.exe</SourceImage>
  <SourceImage condition="is">C:\Windows\System32\MsMpEng.exe</SourceImage>
  <GrantedAccess condition="is">0x1000</GrantedAccess>
  <GrantedAccess condition="is">0x100000</GrantedAccess>
</ProcessAccess>

<!-- Exclure créations de processus légitimes à haut volume -->
<ProcessCreate onmatch="exclude">
  <ParentImage condition="is">C:\Windows\System32\services.exe</ParentImage>
  <Image condition="begin with">C:\Windows\System32\conhost.exe</Image>
</ProcessCreate>
```

Event IDs Sysmon critiques pour la détection sur DC

Sur un DC, six Event IDs Sysmon sont particulièrement critiques pour la détection des attaques Active Directory.

Event 1 — Process Creation

L'Event 1 enregistre toute création de processus avec le hash SHA-256 de l'exécutable, le chemin complet, la ligne de commande et le processus parent. Sur un DC, surveillez spécifiquement :

ntdsutil.exe : utilisé pour l'extraction de ntds.dit via la commande "ifm". Toute exécution hors maintenance planifiée est suspecte.

procdump.exe avec `-ma lsass` dans la ligne de commande : dump mémoire de LSASS.

vssadmin.exe ou **diskshadow.exe** : peuvent être utilisés pour accéder à ntds.dit via shadow copy.

Toute exécution depuis `%TEMP%`, `%APPDATA%` ou des chemins UNC inhabituels.

Event 3 — Network Connection

Les connexions réseau sortantes initiées depuis le DC sont anormales dans la grande majorité des cas. Un DC ne doit pas initier de connexions vers des IP externes. Surveillez :

Connexions sortantes depuis lsass.exe, ntds.exe ou svchost.exe vers des IP non-RFC1918.

Connexions sur des ports inhabituels (443, 80, 8080) depuis des processus système.

Volumes inhabituels de réplication DRSUAPI (port 49152+) — signature possible de DCSync depuis un client non-DC.

Event 7 — Image Loaded

L'Event 7 enregistre chaque DLL chargée par un processus. Sur lsass.exe, toute DLL non signée Microsoft ou chargée depuis un chemin inhabituel est un signal fort d'injection. Attention : Event 7 peut être très verbeux sur un DC — filtrez aggressivement pour ne garder que les DLL suspectes (non signées, chargées depuis %TEMP%, etc.).

Event 10 — Process Access

C'est l'Event le plus important pour détecter Mimikatz et les techniques de credential dumping. Le champ `GrantedAccess` indique les droits obtenus sur le processus cible :

GrantedAccess	Technique associée	Sévérité
0x1010	Mimikatz sekurlsa::logonpasswords (read + query info)	Critique
0x1438	Mimikatz sekurlsa (variante)	Critique
0x143A	Credential dumping avancé	Critique
0x40	Accès lecture mémoire générique	Élevé
0x1FFFFF	Accès complet à lsass (PROCESS_ALL_ACCESS)	Critique

Event 11 — File Create

La création de ntds.dit hors de son emplacement standard (`C:\Windows\NTDS\`) est une signature directe d'extraction de base de données AD. Les shadow copies accessibles via `\?`

`\GLOBALROOT\Device\HarddiskVolumeShadowCopy*\Windows\NTDS tds.dit` sont également à surveiller.

Event 22 — DNS Query

Les requêtes DNS inhabituelles depuis un DC (résolution de domaines C2 connus, requêtes DNS vers des TLD non standards) peuvent signaler une compromission. Cet Event doit être filtré avec soin sur un DC — les DC font eux-mêmes beaucoup de résolution DNS interne.

Détections spécifiques AD via Sysmon

DCSync (T1003.006)

DCSync abuse du protocole DRSUAPI pour répliquer les secrets AD (hashes NTLM, tickets Kerberos) sans accès physique aux DC. La détection via Sysmon combine :

Event 10 sur lsass.exe avec GrantedAccess 0x1010 depuis un processus non-SYSTEM sur un DC

Corrélation avec **Security Event 4662** (opération sur un objet AD) avec GUID de réplication DS

Sur le réseau : trafic DRSUAPI (RPC, port 49152+) depuis une IP non-DC

Mimikatz sekurlsa (T1003.001)

```
 <!-- Règle Sysmon pour Mimikatz sekurlsa -->
<ProcessAccess onmatch="include">
  <TargetImage condition="end with">lsass.exe</TargetImage>
  <GrantedAccess condition="is">0x1010</GrantedAccess>
  <GrantedAccess condition="is">0x1438</GrantedAccess>
  <GrantedAccess condition="is">0x143A</GrantedAccess>
  <GrantedAccess condition="is">0x1FFFFF</GrantedAccess>
</ProcessAccess>
```

Procdump sur LSASS (T1003.001 variante)

Procdump est un outil Sysinternals légitime détourné pour dumper lsass.exe. La détection combine Event 1 (ProcessCreate avec procdump.exe + argument lsass) et Event 10 (ProcessAccess sur lsass depuis procdump).

Déploiement GPO sur tous les DC

Pour un déploiement uniforme sur tous les contrôleurs de domaine, utilisez une GPO avec un script de démarrage. Cette méthode garantit que Sysmon est installé et configuré même sur les nouveaux DC ajoutés à l'environnement.

```
 # Installation Sysmon via GPO (script démarrage)
$sysmonPath = "\dc01\NETLOGON\Sysmon\sysmon64.exe"
$configPath = "\dc01\NETLOGON\Sysmon\sysmon-dc-config.xml"
if (-not (Get-Service "Sysmon64" -ErrorAction SilentlyContinue)) {
    Start-Process -FilePath $sysmonPath -ArgumentList "-accepteula -i $configPath" -Wait
} else {
    Start-Process -FilePath $sysmonPath -ArgumentList "-c $configPath" -Wait
}

# Vérifier que Sysmon tourne
Get-Service Sysmon64 | Select Name, Status, StartType
Get-WinEvent -LogName "Microsoft-Windows-Sysmon/Operational" -MaxEvents 10 | Select TimeCreated,
```

Configuration GPO

Copiez Sysmon64.exe et sysmon-dc-config.xml dans `\domain\NETLOGON\Sysmon\`.

Créez une GPO ciblant l'OU des contrôleurs de domaine (Domain Controllers OU).

Computer Configuration → Windows Settings → Scripts (Startup/Shutdown) → Startup → ajoutez le script PowerShell.

Activez "Run scripts asynchronously" : Non (le script doit terminer avant le login).

Liez la GPO à l'OU Domain Controllers avec **Enforced** pour éviter le blocage d'héritage.

Pour les mises à jour de configuration (nouvelle version de sysmon-dc-config.xml), le script détecte automatiquement Sysmon déjà installé et applique uniquement la nouvelle configuration (`-c $configPath`).

Intégration avec Wazuh

Wazuh ingère nativement les événements Sysmon via le canal Windows Event Log. La configuration se fait dans ossec.conf sur l'agent Wazuh déployé sur chaque DC.

```
 <!-- ossec.conf — agent Wazuh sur DC -->
<ossec_config>
  <localfile>
    <location>Microsoft-Windows-Sysmon/Operational</location>
    <log_format>eventchannel</log_format>
  </localfile>

  <!-- Journaux Windows Security obligatoires en complément -->
  <localfile>
    <location>Security</location>
    <log_format>eventchannel</log_format>
    <query>Event/System[EventID=4624 or EventID=4625 or EventID=4662 or EventID=4768 or EventID=4
  </localfile>
</ossec_config>
```

Wazuh inclut des règles prédéfinies pour Sysmon dans `/var/ossec/rules/0595-win-sysmon_rules.xml` . Pour les détections AD spécifiques (DCSync, Mimikatz), ajoutez des règles custom :

```
<!-- /var/ossec/rules/local_rules.xml — règles Sysmon DC -->
<group name="sysmon,active_directory,">
  <rule id="100200" level="15">
    <if_group>sysmon_event_10</if_group>
    <field name="win.eventdata.targetImage" type="pcre2">(?i)lsass\.exe</field>
    <field name="win.eventdata.grantedAccess" type="pcre2">0x1010|0x1438|0x143a|0x1fffff</field>
    <description>Possible Mimikatz/DCSync - accès LSASS suspect (GrantedAccess $(win.eventdata.gr
    <mitre>
      <id>T1003.001</id>
    </mitre>
  </rule>

  <rule id="100201" level="15">
    <if_group>sysmon_event_11</if_group>
    <field name="win.eventdata.targetFilename" type="pcre2">(?i)ntds\.dit</field>
    <description>ntds.dit créé hors emplacement standard — extraction AD suspectée</description>
    <mitre>
      <id>T1003.003</id>
    </mitre>
  </rule>
</group>
```

Pour en savoir plus sur l'intégration Wazuh avec Active Directory, consultez notre article dédié sur la détection des attaques Active Directory avec Wazuh.

Intégration avec Microsoft Sentinel

Pour les environnements utilisant Microsoft Sentinel comme SIEM, les événements Sysmon remontent via le connecteur Windows Security Events (Azure Monitor Agent).

Configuration Azure Monitor Agent

```
# Vérifier que l'Azure Monitor Agent collecte le canal Sysmon
# Dans Azure Portal : Log Analytics Workspace → Data Sources → Windows Event Logs
# Ajouter : Microsoft-Windows-Sysmon/Operational (level: Information)
```

Requêtes KQL pour Sentinel

```
// Détection Mimikatz via Event 10 Sysmon
Event
| where Source == "Microsoft-Windows-Sysmon"
| where EventID == 10
| parse EventData with * '<Data Name="TargetImage">' TargetImage '</Data>' *
                       '<Data Name="GrantedAccess">' GrantedAccess '</Data>' *
                       '<Data Name="SourceImage">' SourceImage '</Data>' *
| where TargetImage has "lsass.exe"
| where GrantedAccess in ("0x1010", "0x1438", "0x143a", "0x1fffff")
| project TimeGenerated, Computer, SourceImage, TargetImage, GrantedAccess
| sort by TimeGenerated desc
```

Performance — overhead sur DC en production

L'overhead de Sysmon sur un DC dépend directement de la configuration XML. Voici les mesures observées sur un DC Windows Server 2025 avec 500 utilisateurs actifs :

Configuration	CPU overhead	RAM	Événements/min
SwiftOnSecurity non modifiée	3-5%	~120 Mo	800-1200
Config DC optimisée (ce guide)	1-2%	~80 Mo	150-300
Config minimale (Events 1,10,11 seulement)	<0.5%	~50 Mo	20-50

Les Event IDs les plus coûteux en performance sur un DC sont Event 3 (Network Connection — très fréquent sur un DC) et Event 7 (Image Loaded). Si la performance est contrainte,

commencez par une config minimale axée sur Events 1, 10 et 11, qui couvrent les attaques les plus critiques.

Maintenance et mises à jour Sysmon

Mise à jour de la version Sysmon

```
 # Mettre à jour Sysmon sans interruption de service
# Sysmon supporte la mise à jour en place
Start-Process -FilePath "\dc01\NETLOGON\Sysmon\sysmon64.exe" -ArgumentList "-u" -Wait
Start-Process -FilePath "\dc01\NETLOGON\Sysmon\sysmon64.exe" -ArgumentList "-accepteula -i \dc01\
```

Mise à jour de la configuration XML

```
 # Appliquer une nouvelle configuration sans redémarrer le service
$sysmonPath = "C:\Tools\Sysmon\Sysmon64.exe"
$configPath = "\dc01\NETLOGON\Sysmon\sysmon-dc-config.xml"
Start-Process -FilePath $sysmonPath -ArgumentList "-c $configPath" -Wait
# Vérifier que la nouvelle config est active
& $sysmonPath -s
```

La mise à jour de configuration est instantanée et non disruptive — Sysmon continue de fonctionner pendant l'application de la nouvelle configuration.

Analyse forensique des événements Sysmon post-incident

Sysmon n'est pas seulement un outil de détection en temps réel — il constitue également une ressource précieuse lors des investigations forensiques post-compromission. Les journaux Sysmon conservés en dehors du DC (via Wazuh, Sentinel ou un collecteur SIEM) permettent de

reconstituer la chronologie d'une attaque avec une précision que les journaux Windows natifs ne peuvent pas atteindre seuls.

Reconstruction de la chaîne d'exécution avec Event 1

L'Event 1 Sysmon enregistre le PPID (Process Parent ID) en plus du PID, ce qui permet de reconstruire l'arbre de processus complet d'une attaque. Pour une investigation DCSync ou Mimikatz :

```
 # Exporter les Event 1 Sysmon des dernières 72h vers CSV
$StartTime = (Get-Date).AddHours(-72)
Get-WinEvent -LogName "Microsoft-Windows-Sysmon/Operational" -FilterXPath "*[System[EventID=1 and
    ForEach-Object {
        $xml = [xml]$_.ToXml()
        [PSCustomObject]@{
            Time = $_.TimeCreated
            ProcessName = ($xml.Event.EventData.Data | Where Name -eq 'Image').InnerText
            CommandLine = ($xml.Event.EventData.Data | Where Name -eq 'CommandLine').InnerText
            ParentProcess = ($xml.Event.EventData.Data | Where Name -eq 'ParentImage').InnerText
            Hash = ($xml.Event.EventData.Data | Where Name -eq 'Hashes').InnerText
            User = ($xml.Event.EventData.Data | Where Name -eq 'User').InnerText
        }
    } | Export-Csv -Path "C:\Forensics\sysmon-processes-72h.csv" -NoTypeInformation -Encoding UTF
```

Timeline forensique multi-events

Pour une investigation complète sur un DC compromis, combinez les events Sysmon avec les logs Security dans une timeline unifiée. La méthode recommandée avec Wazuh est d'utiliser les requêtes Wazuh API pour extraire les événements corrélés autour d'un timeframe suspect :

T-5min à T+0 : Event 3 (connexion réseau entrante inhabituelle) → indicateur de mouvement latéral initial.

T+0 à T+2min : Event 1 (création processus cmd.exe, PowerShell, ou outil d'attaque) → exécution de payload.

T+2min à T+5min : Event 10 (accès à lsass.exe) → tentative d'extraction de credentials.

T+5min à T+15min : Event 11 (création de fichiers dans %TEMP% ou chemin réseau) → exfiltration potentielle.

Préservation des journaux Sysmon

Par défaut, le journal Windows Microsoft-Windows-Sysmon/Operational est limité à 64 Mo. Sur un DC actif, ce journal peut être saturé en quelques heures avec une configuration verbeuse. Augmentez la taille maximale du journal et activez l'archivage automatique :

```
 # Augmenter la taille du journal Sysmon à 512 Mo et archiver à la saturation
wevtutil sl "Microsoft-Windows-Sysmon/Operational" /ms:536870912 /rt:false /ab:true

# Vérifier la configuration
wevtutil gl "Microsoft-Windows-Sysmon/Operational"
```

En parallèle, configurez votre agent SIEM (Wazuh ou Azure Monitor Agent) pour collecter les événements en temps réel — ne comptez pas uniquement sur les journaux locaux pour la rétention long terme. Sur les DC soumis à NIS2, une rétention minimum de 12 mois des logs de sécurité est recommandée par l'ANSSI.

Indicateurs de faux positifs courants

Sur un DC Windows Server 2025, certains processus légitimes génèrent des Event 10 sur lsass.exe avec des GrantedAccess bénins. Sans exclusion, ces processus déclenchent des alertes SIEM parasites :

Windows Defender / MsMpEng.exe : accède régulièrement à lsass avec GrantedAccess 0x1000 (PROCESS_QUERY_LIMITED_INFORMATION) — à exclure explicitement.

Velociraptor, CrowdStrike, SentinelOne : agents EDR qui accèdent légitimement à lsass pour la surveillance — leurs chemins d'installation doivent être exclus.

Task Manager (taskmgr.exe) : accès GrantedAccess 0x400 (PROCESS_QUERY_INFORMATION) pour afficher les processus — bénin.

LSASS lui-même : accès intra-processus avec GrantedAccess 0x1fffff — normal, à exclure si source = target.

Documentez ces faux positifs dans votre configuration XML et dans les règles de suppression de votre SIEM pour éviter la fatigue d'alerte. Un SOC qui reçoit trop de faux positifs finit par ignorer les vraies alertes — c'est le risque opérationnel principal d'une configuration Sysmon non optimisée sur un DC.

Lecture complémentaire sur la sécurité AD

Sysmon est un composant d'une stratégie de détection plus large. Pour comprendre les techniques que vous cherchez à détecter, consultez nos articles sur Mimikatz et l'extraction de credentials AD, l'attaque DCSync et ses défenses, et la protection de ntds.dit contre l'extraction. Pour situer ces techniques dans un cadre tactique, notre article sur le framework MITRE ATT&CK et les TTPs 2026 offre la contextualisation nécessaire.

À RETENIR

Points clés à retenir

Event 10 (ProcessAccess) sur lsass.exe avec GrantedAccess 0x1010/0x1438 est la signature la plus fiable de Mimikatz.

Event 11 (FileCreate) sur ntds.dit hors de C:\Windows\NTDS\ signale une extraction de la base AD.

Les exclusions dans la config XML sont aussi importantes que les inclusions — sans elles, le volume de bruit masque les vraies alertes.

Sysmon et les événements Windows Security sont complémentaires — une stratégie SOC DC nécessite les deux.

Le déploiement GPO via NETLOGON garantit la cohérence sur tous les DC y compris les nouveaux.

Checklist Sysmon sur contrôleurs de domaine

[ ] Télécharger Sysmon 15.x et vérifier la signature Authenticode Microsoft

[ ] Préparer la configuration XML adaptée DC (Events 1, 3, 7, 10, 11, 22)

[ ] Inclure les règles de détection : lsass access 0x1010/0x1438, ntds.dit création, ntdsutil/procdump/mimikatz

[ ] Ajouter les exclusions des processus légitimes à haut volume

[ ] Tester la configuration sur un DC non-critique avant déploiement général

[ ] Copier Sysmon64.exe + config XML dans \domain\NETLOGON\Sysmon\

[ ] Créer la GPO avec script de démarrage ciblant l'OU Domain Controllers

[ ] Vérifier l'installation sur tous les DC (Get-Service Sysmon64)

[ ] Mesurer l'overhead CPU/RAM avant et après déploiement

[ ] Configurer Wazuh (ossec.conf) ou Sentinel (Azure Monitor Agent) pour ingérer le canal Sysmon

[ ] Ajouter les règles de détection SIEM (Mimikatz Event 10, ntds.dit Event 11)

[ ] Tester les alertes avec des scénarios simulés (Invoke-AtomicTest si disponible)

[ ] Documenter la procédure de mise à jour (version Sysmon + rotation config XML)

[ ] Planifier une revue trimestrielle de la configuration XML

FAQ — Sysmon sur contrôleurs de domaine

Quel overhead CPU/mémoire Sysmon génère-t-il sur un DC ?

Avec une configuration bien optimisée, Sysmon génère entre 1% et 3% d'overhead CPU sur un DC en charge normale, et consomme environ 50 à 150 Mo de RAM selon le volume d'événements. Les configurations trop permissives peuvent monter à 5-10% CPU. La clé est

l'exclusion agressive des processus légitimes à haut volume. La configuration SwiftOnSecurity est un bon point de départ mais doit être adaptée au profil de charge de chaque DC.

Quelle config Sysmon utiliser sur un DC Windows Server 2025 ?

Pour un DC, activez systématiquement Event 10 ciblant lsass.exe pour détecter Mimikatz et DCSync, Event 11 pour les accès à ntds.dit hors emplacement normal, et Event 7 pour les DLL injections dans LSASS. Excluez les processus système à haut volume (services.exe, svchost.exe avec parents légitimes) pour réduire le bruit. La config MSTIC DC-specific sur GitHub est une excellente base complémentaire à SwiftOnSecurity.

Sysmon peut-il détecter Mimikatz en temps réel ?

Sysmon Event 10 avec GrantedAccess 0x1010 ou 0x1438 ciblant lsass.exe est la signature la plus fiable de Mimikatz sekurlsa. Cette détection fonctionne en temps réel dès que Mimikatz ouvre un handle sur lsass.exe. Attention : Mimikatz peut obfusquer ces appels via des techniques kernel. Coupler Sysmon Event 10 avec Windows Defender Credential Guard et un EDR kernel offre une détection multicouche bien plus robuste.

Comment envoyer les événements Sysmon vers Wazuh ?

Dans ossec.conf sur l'agent Wazuh du DC, ajoutez un bloc localfile ciblant le canal Microsoft-Windows-Sysmon/Operational avec log_format eventchannel. Wazuh intègre nativement des règles de décodage pour Sysmon. Pour les alertes spécifiques DC, ajoutez des règles custom dans /var/ossec/rules/local_rules.xml ciblant les Event IDs 1, 10, 11 avec les attributs GrantedAccess appropriés.

Faut-il des événements Sysmon ET les événements Windows natifs ?

Oui, les deux sont complémentaires. Les événements Windows Security couvrent les authentifications (4624/4625/4648), les changements AD (5136/5137) — informations que Sysmon ne capture pas. Sysmon complète avec la visibilité processus, réseau, accès fichiers et

accès processus — informations absentes des journaux natifs. Pour une détection efficace des attaques AD (DCSync, Pass-the-Hash, Kerberoasting), vous avez besoin des deux canaux.

Sources

ANSSI CERT-FR — Publications et alertes

MITRE ATT&CK — Framework des techniques d'attaque

CISA — Advisories de cybersécurité

