

Exploitation des tokens Windows

Introduction au pentest AD et aux Internals Windows

whoami /all



Éditer le profil

Aurélien Chalot
@Defte_

Hacker, sysadmin and security researcher @OrangeCyberdef 💻
Calisthenic enthusiast 💪
100 Hide&Sec 100
[Traduire la biographie](#)



 The grid  blog.whiteflag.io  A rejoint Twitter en novembre 2017









399 abonnements **224** abonnés

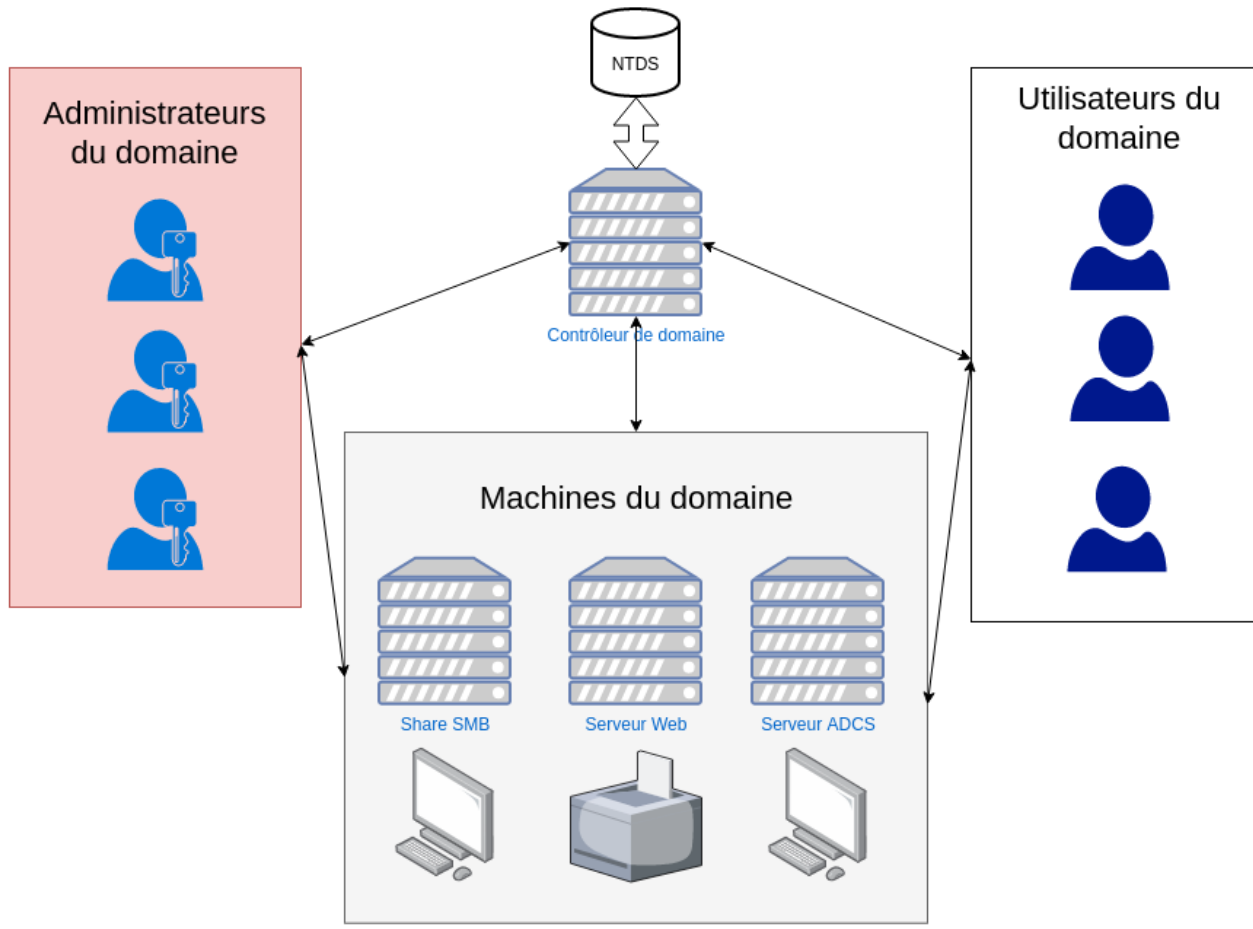
Vous avez dit Active Directory ?

- Un Active Directory c'est un annuaire qui contient les informations relatives aux ressources d'une entreprise:

- Ordinateurs / serveurs
- Imprimantes
- Dossiers partagés (share SMB)
- Utilisateurs

Nom	
 SERVEUR	
 WIN-7US6VSMRGR6	

Nom	Type	Description
 Administrateur	Utilisateur	Compte d'utilisateur d'a...
 Administrateurs clés	Groupe de sécurité - Global	Les membres de ce grou...
 Administrateurs clés Entreprise	Groupe de sécurité - Universel	Les membres de ce grou...
 Administrateurs de l'entreprise	Groupe de sécurité - Universel	Administrateurs désigné...
 Administrateurs du schéma	Groupe de sécurité - Universel	Administrateurs désigné...
 Admins du domaine	Groupe de sécurité - Global	Administrateurs désigné...
 Contrôleurs de domaine	Groupe de sécurité - Global	Tous les contrôleurs de ...
 Contrôleurs de domaine clonab...	Groupe de sécurité - Global	Les membres de ce grou...
 Contrôleurs de domaine d'entr...	Groupe de sécurité - Universel	Les membres de ce grou...



Le contrôleur de domaine (DC) est la pièce centrale qui permet de manager l'ensemble du réseau Active Directory

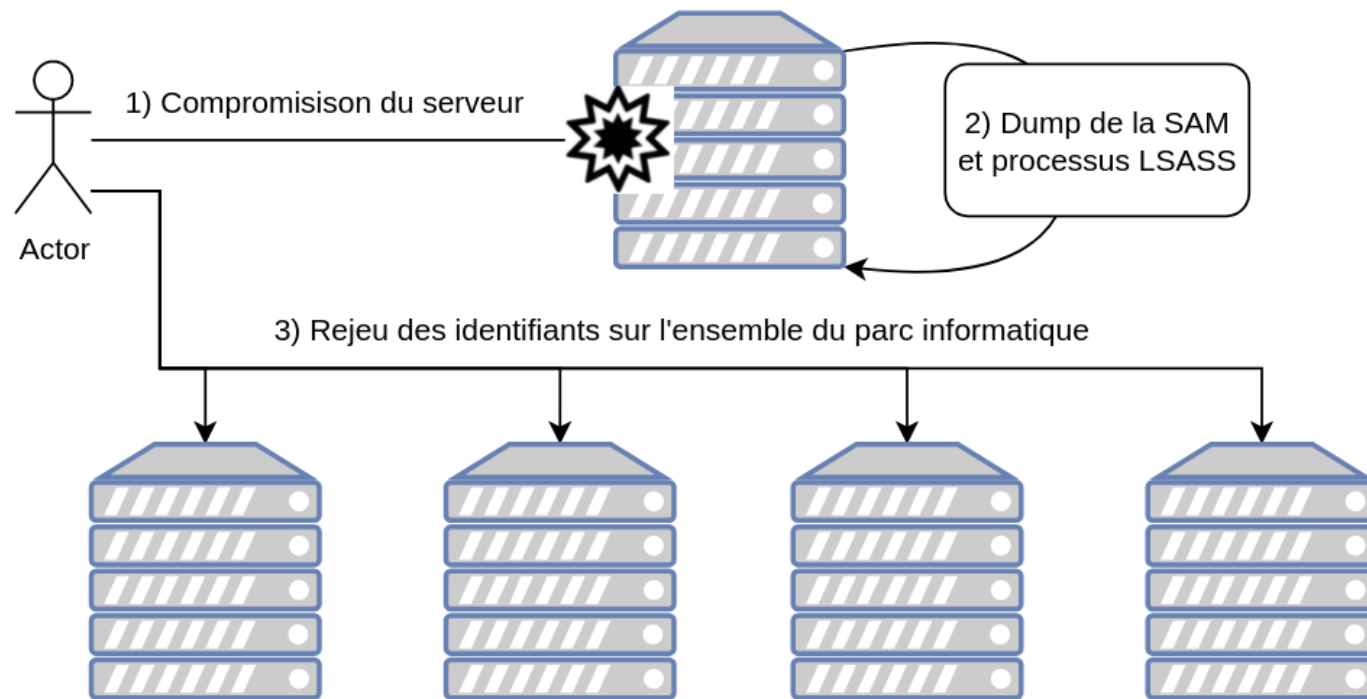
Test d'intrusion interne

Compromettre l'Active Directory -> être administrateur du domaine

Pour cela il existe plusieurs techniques:

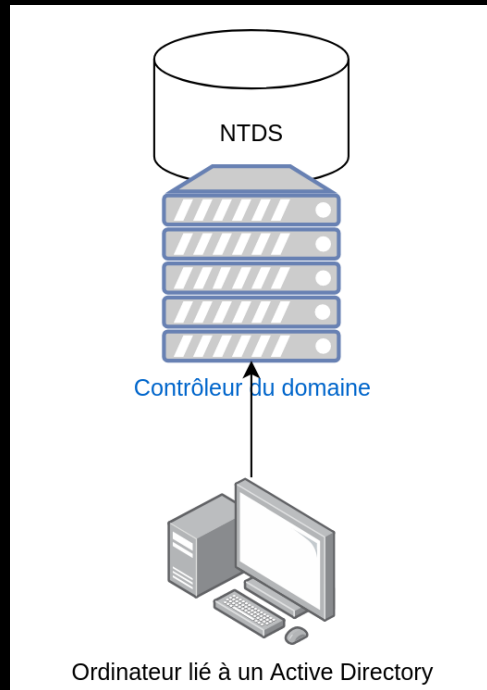
- Comptes utilisateurs avec des mots de passe faibles
- Exploitation de défauts de configuration Active Directory
- Exploitation de serveurs / services vulnérables et rebond

Scénario classique



SAM, LSASS, NTDS, dafuk ?

NTDS (NT Directory Services): base de données des comptes sur un réseau Active Directory



Fichier présent sur les contrôleurs du domaine

System32

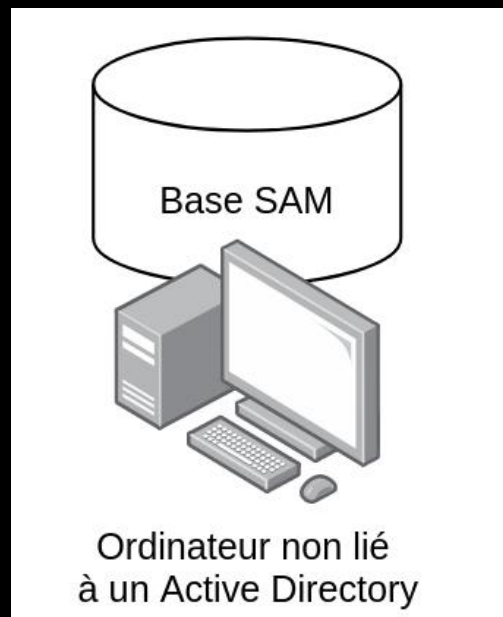
Fichier Accueil Partage Affichage

Ce PC > Disque local (C:) > Windows > System32

Nom	Modifié le	Type	Taille
nsisvc.dll	10/07/2010 13:10	extension de l'app...	50 Ko
nslookup	16/07/2016 15:18	Application	85 Ko
ntasn1.dll	16/07/2016 15:18	Extension de l'app...	232 Ko
ntdll.dll	02/02/2018 19:28	Extension de l'app...	1 844 Ko
ntds.dit	13/01/2022 11:44	Fichier DIT	12 288 Ko
ntdsu.dll	13/01/2022 11:44	Extension de l'app...	94 Ko
ntdsai.dll	13/01/2022 11:44	Extension de l'app...	3 866 Ko

SAM, LSASS, NTDS, dafuk ?

SAM (Security Account Manager): base de données des comptes **locaux** sur un système Windows

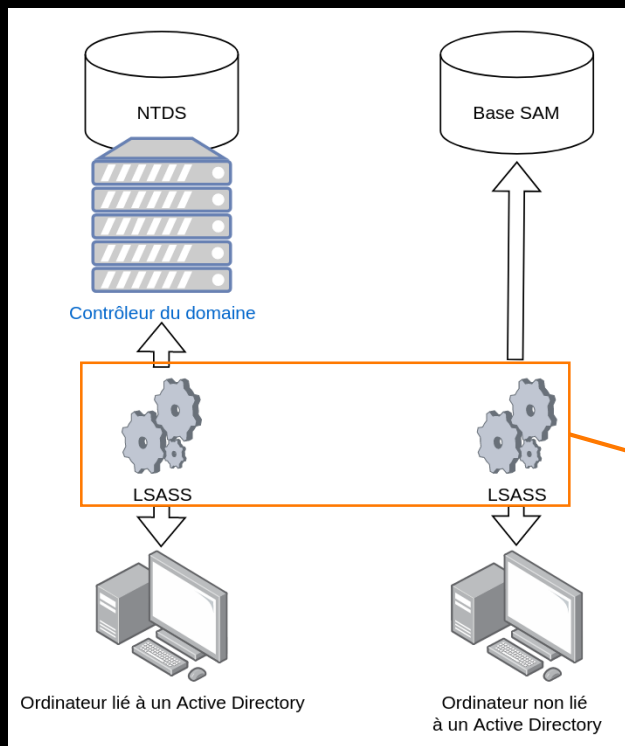


Clé de registre présente sur toutes les machines Windows

Clé de registre présente sur toutes les machines Windows			
Nom	Type	Données	
(par défaut)	REG_SZ	(valeur non définie)	
C	REG_BINARY	08 00 01 00 00 00 00 00 d0 00 00 03 00 01 00 01 0...	
ServerDomainUpdates	REG_BINARY	fe ff 01	

SAM, NTDS, LSASS, dafuk ?

LSASS (Local Security Authority Subsystem): processus en charge de l'authentification sur un système Windows



Gestionnaire des tâches

Fichier Options Affichage

Processus Performance Utilisateurs Détails Services

Nom	7% Processeur	24% Mémoire
> Hôte de service : service local (8)	0%	5,9 Mo
> Hôte de service : service local (aucun réseau) (4)	0%	5,6 Mo
> Hôte de service : service local (réseau restreint)	0%	1,1 Mo
> Hôte de service : service local (réseau restreint) (4)	0%	9,4 Mo
> Hôte de service : service réseau (5)	0%	5,4 Mo
> Hôte de service : service réseau (réseau restreint)	0%	1,0 Mo
> Hôte de service : système local (14)	0%	16,6 Mo
> Hôte de service : système local (réseau restreint) (4)	0%	6,6 Mo
Interruptions système	0,8%	0 Mo
Local Security Authority Process (6)	0,5%	27,6 Mo
Processus d'exécution client-serveur	0%	1,1 Mo
Processus d'exécution client-serveur	0%	1,1 Mo

SAM, NTDS, LSASS, dafuk ?

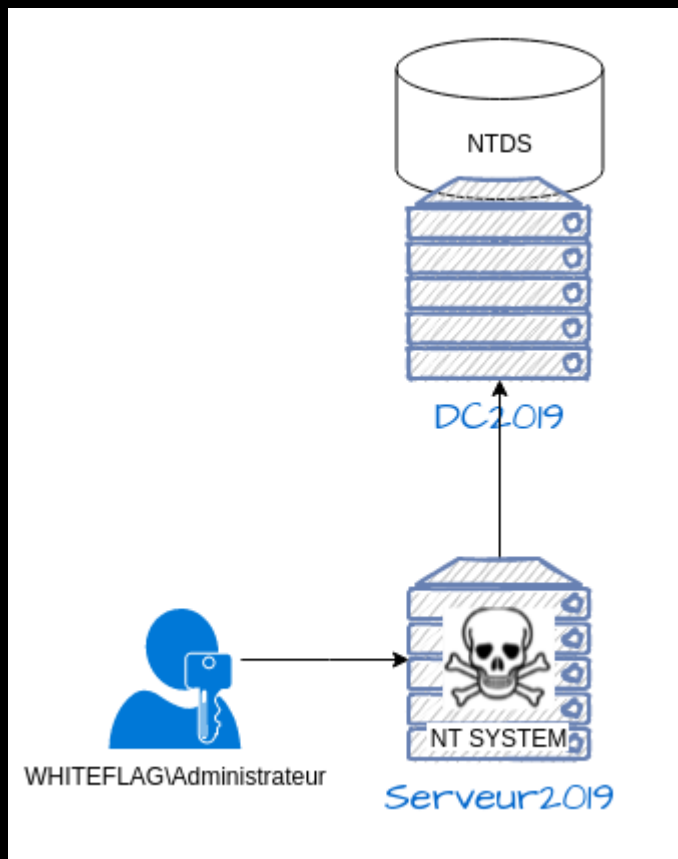
Le processus LSASS stocke les informations d'authentications telles que:

- Des mots de passe en clair ou sous la forme d'un hash NTLM
- Des tickets Kerberos (CF authentication Kerberos)

Pour cette raison il est **extrêmement important** de dumper le contenu du processus (la RAM allouée au processus LSASS) de manière à en extraire le contenu.



Configuration du laboratoire



- Un administrateur du domaine
- Un contrôleur du domaine (dc2019)
- Un serveur Windows 2019 (déjà compromis)

Demo time (Windows Defender désactivé)



Kernel

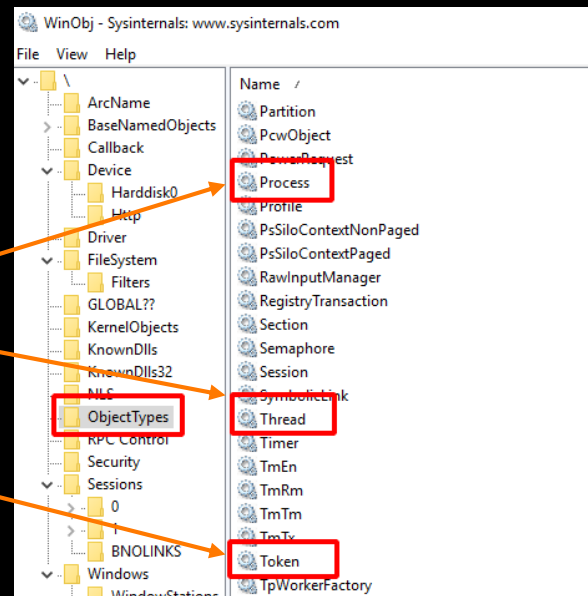
Le Kernel c'est le cœur du système d'exploitation Sa fonction est de:

- Gérer les ressources du système (mémoire RAM, processus etc.)
- Gérer les accès sécurisés à ces ressources
- Permettre la communication entre les logiciels et le hardware

Pour assurer cette fonction il dispose de plusieurs **objets**.

Par exemple:

- Un objet de type **Process** utilisé pour manager un processus
- Un objet de type **Thread** utilisé pour manager un Thread
- Ou encore un objet de type **Token**



Token

Token: objet Windows qui décrit le contexte de sécurité d'un processus ou d'un thread.

Au sein de ce token on trouve:

- L'identité de l'utilisateur qui détient ce token
- Les groupes dans lequel se trouve l'utilisateur
- Un identifiant logon SID qui identifie la session de connexion actuel (le contexte)
- L'ensemble des privilèges attribués à l'utilisateur sur le système

Propriétés de : powershell.exe (4356)

Memory	Environment	Handles	Job	.NET assemblies
.NET performance	GPU	Disk and Network	Comment	
General	Statistics	Performance	Threads	Token

User: WHITEFLAG\Administrateur

User SID: S-1-5-21-1254471023-1136857918-2392298254-500

Session: 1 Elevated: N/A Virtualized: Not allowed

App container SID: N/A

Name	Flags
AUTORITE NT\Cette organisation	Mandatory (de
AUTORITE NT\INTERACTIF	Mandatory (de
AUTORITE NT\Utilisateurs authentifiés	Mandatory (de
BUILTIN\Administrateurs	Mandatory (de
BUILTIN\Utilisateurs	Mandatory (de

Name	Status	Description
SeBackupPrivilege	Disabled	Sauvega...
SeChangeNotifyPrivilege	Default Enabled	Contour...
SeCreateGlobalPrivilege	Default Enabled	Créer de...
SeCreatePagefilePrivilege	Disabled	Créer un...
SeCreateSymbolicLinkPrivilege	Disabled	Créer de...
SeDebugPrivilege	Enabled	Débogu...
SeDelegateSessionUserImpersonatePrivilege	Disabled	Obtenir ...

To view capabilities, claims and other attributes, click Advanced.

Integrity Advanced

Privilèges Windows

Il existe 37 privilèges sous un système Windows. Ces privilèges permettent à celui qui les détient d'effectuer des actions sur le système.

Par exemple:

- **SeLoadDriverPrivilege**: permet de charger un driver
- **SeShutdownPrivilege**: permet d'arrêter la machine
- **SeDebugPrivilege**: permet de déboguer n'importe quel processus du système

```
C:\Users\Administrateur>whoami /priv
```

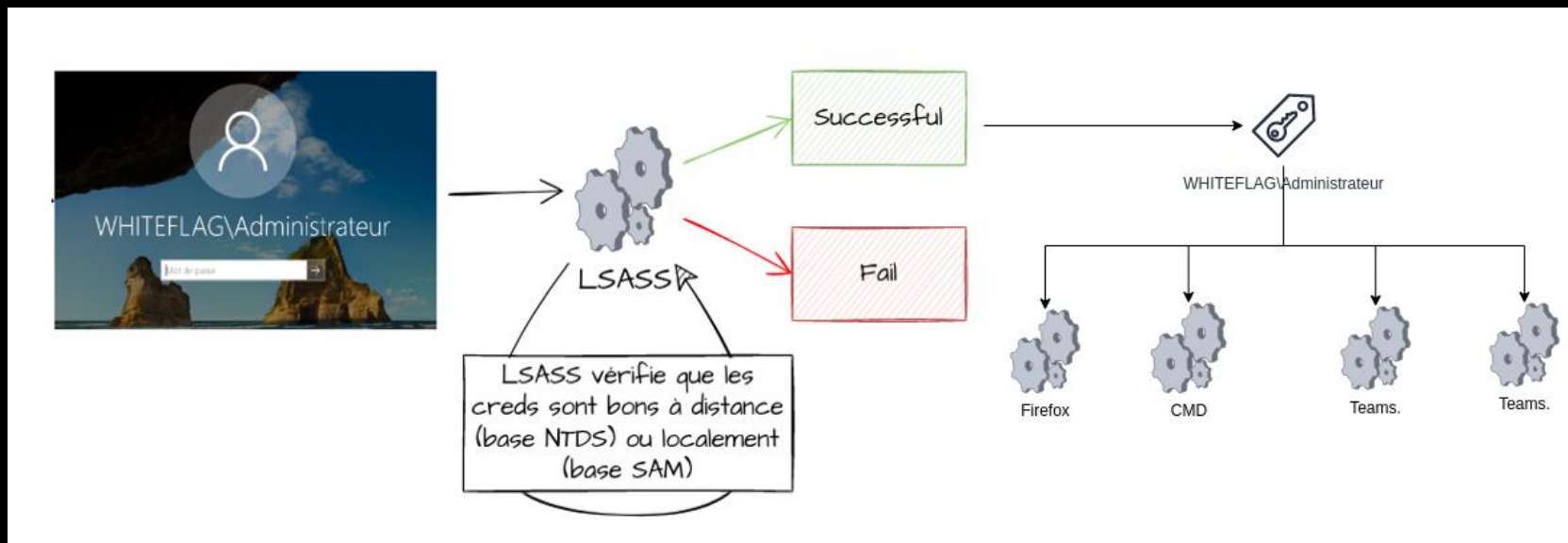
```
Informations de privilèges
```

```
-----
```

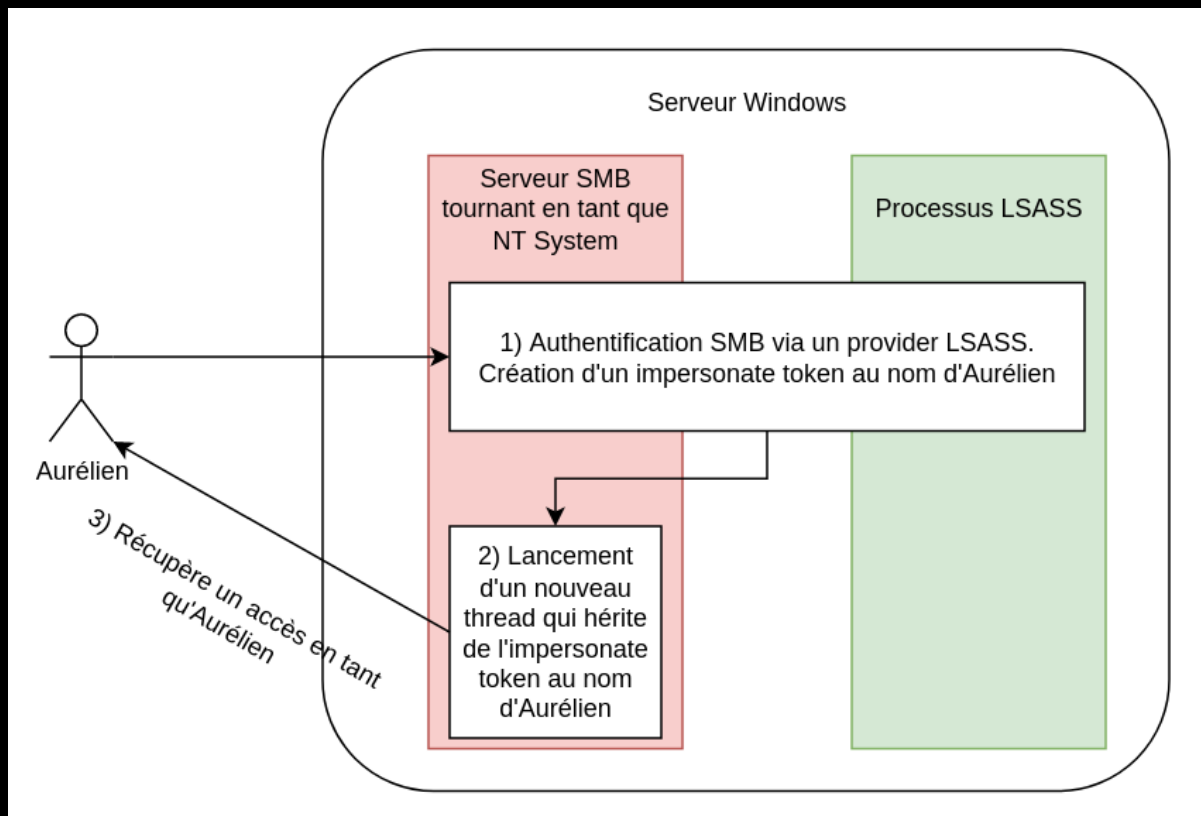
Nom de privilège	Description
=====	=====
SeIncreaseQuotaPrivilege	Ajuster les quotas de mémoire pour un processus
SeMachineAccountPrivilege	Ajouter des stations de travail au domaine
SeSecurityPrivilege	Gérer le journal d'audit et de sécurité
SeTakeOwnershipPrivilege	Prendre possession de fichiers ou d'autres objets
SeLoadDriverPrivilege	Charger et décharger les pilotes de périphériques
SeSystemProfilePrivilege	Performance système du profil
SeSystemtimePrivilege	Modifier l'heure système
SeProfileSingleProcessPrivilege	Processus unique du profil
SeIncreaseBasePriorityPrivilege	Augmenter la priorité de planification
SeCreatePagefilePrivilege	Créer un fichier d'échange
SeBackupPrivilege	Sauvegarder les fichiers et les répertoires
SeRestorePrivilege	Restaurer les fichiers et les répertoires
SeShutdownPrivilege	Arrêter le système
SeDebugPrivilege	Déboguer les programmes
SeSystemEnvironmentPrivilege	Modifier les valeurs de l'environnement du microproc

Il existe deux types de token: primary token et impersonate token

Quand obtient-on un primary token?



Quand obtient-on un impersonate token ?



Primary token vs Impersonate token

	Primary token	Impersonate token
Est attribué à:	Un processus	Un thread
Est obtenu suite à:	Une authentification interactive	Principalement via une authentification réseau
Les identifiants de stockage sont stockés dans LSASS?	Oui	Non

Where is this going ?

Les tokens sont des **objets Windows** disponibles au sein du système d'exploitation que l'on peut **manipuler** à condition de disposer des privilèges suffisants (notamment SeDebugPrivilege)



Manipuler un token

On sait que les tokens sont des objets Windows. Ils sont donc représentés par une structure

(Le project Vergilius tient à jour les structures des objets Windows pour l'ensemble des versions Windows. C'est très utile quand il faut dev via la WinAPI :P <https://www.vergiliusproject.com>)

```
//0x498 bytes (sizeof)
struct _TOKEN
{
    struct _TOKEN_SOURCE TokenSource; //0x0
    struct _LUID TokenId; //0x10
    struct _LUID AuthenticationId; //0x18
    struct _LUID ParentTokenId; //0x20
    union _LARGE_INTEGER ExpirationTime; //0x28
    struct _ERESOURCE* TokenLock; //0x30
    struct _LUID ModifiedId; //0x38
    struct _SEP_TOKEN_PRIVILEGES Privileges; //0x40
    struct _SEP_AUDIT_POLICY AuditPolicy; //0x58
    ULONG SessionId; //0x78
    ULONG UserAndGroupCount; //0x7c
    ULONG RestrictedSidCount; //0x80
    ULONG VariableLength; //0x84
    ULONG DynamicCharged; //0x88
    ULONG DynamicAvailable; //0x8c
    ULONG DefaultOwnerIndex; //0x90
    struct _SID_AND_ATTRIBUTES* UserAndGroups; //0x98
    struct _SID_AND_ATTRIBUTES* RestrictedSids; //0xa0
    VOID* PrimaryGroup; //0xa8
    ULONG* DynamicPart; //0xb0
    struct _ACL* DefaultDacl; //0xb8
    enum _TOKEN_TYPE TokenType; //0xc0
    enum _SECURITY_IMPERSONATION_LEVEL ImpersonationLevel; //0xc4
    ULONG TokenFlags; //0xc8
    UCHAR TokenInUse; //0xcc
    ULONG IntegrityLevelIndex; //0xd0
    ULONG MandatoryPolicy; //0xd4
    struct _SEP_LOGON_SESSION_REFERENCES* LogonSession; //0xd8
    struct _LUID OriginatingLogonSession; //0xe0
    struct _SID_AND_ATTRIBUTES_HASH SidHash; //0xe8
    struct _SID_AND_ATTRIBUTES_HASH RestrictedSidHash; //0xf8
    struct _AUTHZBASEP_SECURITY_ATTRIBUTES_INFORMATION* pSecurityAttributes; //0x308
    VOID* Package; //0x310
    struct _SID_AND_ATTRIBUTES* Capabilities; //0x318
    ULONG CapabilityCount; //0x320
    struct _SID_AND_ATTRIBUTES_HASH CapabilitiesHash; //0x328
    struct _SEP_LOWBOX_NUMBER_ENTRY* LowboxNumberEntry; //0x438
    struct _SEP_CACHED_HANDLES_ENTRY* LowboxHandlesEntry; //0x440
    struct _AUTHZBASEP_CLAIM_ATTRIBUTES_COLLECTION* pClaimAttributes; //0x448
    VOID* TrustLevelSid; //0x450
    struct _TOKEN* TrustLinkedToken; //0x458
    VOID* IntegrityLevelsSidValue; //0x460
    struct _SEP_SID_VALUES_BLOCK* TokenSidValues; //0x468
    struct _SEP_LUID_TO_INDEX_MAP_ENTRY* IndexEntry; //0x470
    struct _SEP_TOKEN_DIAG_TRACK_ENTRY* DiagnosticInfo; //0x478
    struct _SEP_CACHED_HANDLES_ENTRY* BnoIsolationHandlesEntry; //0x480
    VOID* SessionObject; //0x488
    ULONGLONG VariablePart; //0x490
};
```

Un peu de code

Demo time (avec Windows Defender activé)

Test d'intrusion réel

```
[*] Listing available tokens
[ID: 0][TokenPrimary][ ] Owner
[ID: 1][TokenImpersonation][S
[ID: 2][TokenImpersonation][S
[ID: 3][TokenPrimary][Securit
[ID: 4][TokenImpersonation][S
[ID: 5][TokenImpersonation][S
[ID: 6][TokenImpersonation][S
[ID: 7][TokenImpersonation][S
[ID: 8][TokenImpersonation][S
[ID: 9][TokenImpersonation][S
[ID: 10][TokenImpersonation][
[ID: 11][TokenImpersonation][
[ID: 12][TokenImpersonation][
[ID: 13][TokenImpersonation][
[ID: 14][TokenPrimary][Securi
[ID: 15][TokenPrimary][Securi
[ID: 16][TokenImpersonation][
[ID: 17][TokenPrimary][Securi
[ID: 18][TokenImpersonation][
[ID: 19][TokenPrimary][Securi
[ID: 20][TokenPrimary][Securi
[ID: 21][TokenPrimary][Securi
[ID: 22][TokenImpersonation][
[ID: 23][TokenImpersonation][
[ID: 24][TokenImpersonation][
[ID: 25][TokenImpersonation][
[ID: 26][TokenImpersonation][
[ID: 27][TokenPrimary][Securi
[ID: 28][TokenPrimary][Securi
[ID: 29][TokenPrimary][Securi
[ID: 29][TokenPrimary][SecurityImpersonation] Owner: AUTORITE NT\SERVICE LOCAL | User: AUTORITE NT\SERVICE LOCAL
[*] Impersonating [redacted] Administrateur and launching command [cmd.exe /c net group 'Domain Admins' [redacted] /Add /doma
La demande sera traitée sur contrôleur de domaine du domaine [redacted]

C:\Users\Administrateur>
```

En résumé

L'authentification Windows est réalisée par le processus LSASS:

- Localement via la base SAM
- A distance en contactant le contrôleur du domaine (base NTDS.dit)

Une fois l'authentification validée, un token est créé au nom de l'utilisateur. Ce token contient:

- Le nom de l'utilisateur
- Les groupes dans lequel il est présent
- Les privilèges associés à l'utilisateur

Les tokens sont des objets Windows que l'on peut manipuler si on dispose des bons privilèges.

En dupliquant le token d'un administrateur du domaine on peut usurper son identité et donc compromettre un domaine. Comme c'est un mécanisme interne de Windows, il est très compliqué pour un AV de détecter l'attaque. Cette technique est donc bien plus furtive qu'un dump de LSASS via Mimikatz.

Remédiations

- Il est possible d'analyser de statiquement les exécutables pour voir quelles fonctions ils utilisent
- Analyser dynamiquement le comportement de l'exécutable (certains EDR's arrivent à détecter l'attaque c'est loin d'être le cas pour tous)
- Mais surtout:



Des questions ?

Mail: aurelien.chalot@orange.com

Twitter: <https://twitter.com/Defte>

Mon blog: <https://blog.whiteflag.io>

<https://cyberdefense.orange.com>

