TP2 - Script de collecte de configuration Windows pour une analyse à froid

Adrien Fontenil, Léo Menudé

14/10/2022

Table des matières

1 Script de collecte VBScript **3**
1.1 Utilisation du script 3
1.2 Fonctionnement du script 3
1.3 Éléments récupérés pour l'analyse à froid 4
1.3.1 Configuration système 4
1.3.2 Utilisateurs, Groupes et Permissions 6
1.3.3 Services et processus 7
1.3.4 Applications 7
1.4 Stockage des informations 8

2 Liens utiles **9**

3 Annexes **10**

1 Script de collecte VBScript

Notre objectif est de réaliser un script autoporteur de collecte de configuration d'une machine `Windows`, dans le but d'effectuer une analyse à froid, afin de détecter de potentielles failles de sécurité. Ce script est écrit en `VBScript` et a été tester sur une machine `Windows Server 2012 R2`.

1.1 Utilisation du script

Ce script doit être exécuté par un administrateur, depuis `cmd.exe` avec la commande suivante :

```
cscript audit_dump.vbs
```

Après exécution, vous pourrez retrouver un fichier archive cabinet (`.cab`) contenant les différentes informations extraites du système depuis lequel le script est lancé. Il se situera au même endroit que le script.

1.2 Fonctionnement du script

Notre script peut s'exécuter exclusivement avec `cmd.exe` car aucune commande `PowerShell` n'est exécuté. En effet, le script doit pouvoir être exécuté sur tout système Windows et donc sans certains modules. Le script s'articule autour de 4 étapes majeures :

- **Initialisation**
- **Collecte**
- **Archivage**
- **Nettoyage**

L'initialisation débute par une section de déclaration ainsi qu'une fonction `init()` qui permet d'initialiser nos objets et mettre en place une arborescence de dossier, dans laquelle seront stocké les résultats de la récupération. un schéma de l'arborescence est disponible en annexe.

La collecte se fait sur différents types d'élément, le système (`audit_system()`), les utilisateurs (`audit_user()`), les services et processus (`audit_svc_proc(()`) et enfin les applications. (`audit_applications()`) Les différentes informations et leurs utilités de récupération sont explicitées à la section suivante.

Nous avons ajouté la notion d'archivage `cab` dans notre script. Cependant notre méthode d'archivage ne permet pas de garder l'arborescence, cela nous obligerait à ajouter un fichier de configuration sur la machine client ce qui empêche d'ajouter facilement la gestion des sous-dossiers au sein de notre script. Il y a donc la possibilité de commenter

à la fin du script les appels aux fonctions `createCab()` et `clearDirectory()` pour désactiver l'archivage.

Pour finir, le script nettoie dans un premier temps, les fichiers temporaires cab et le dossier non archivé pour laisser uniquement notre archive puis ensuite va libérer les ressources allouées aux différents objets VBS utilisés

1.3 Éléments récupérés pour l'analyse à froid

Au début de ce TP, nous avons établi une liste des éléments à cibler et l'intérêt de leur récupération.

Afin de récupérer les différentes informations et fichiers, nous avons développé une fonction `run(command, i, filename)` qui utilise un objet `WScript.Shell` afin de lancer des commandes `command`, spécifiant le sous-dossier de stockage `i` et le nom à donner au fichier créé `filename`. Également, la gestion des sous-dossiers se fait grâce au calcul relatif à la machine des différents chemins via les fonctions `getScriptPath` et `getAuditFolderPath`. Cette gestion se fait également par la fonction `createAuditFolder()` qui utilise un objet `Scripting.FileSystemObject` afin de créer, supprimer, modifier les différents fichiers et dossiers.

Il est important de noter que l'ensemble des commandes exécutées sont des commandes `cmd` dans un souci de portabilité.

1.3.1 Configuration système

Dans un premier temps, nous récupérons des informations des données système assez variées. Une configuration globale est disponible grâce à l'exécution de la commande :

```
systeminfo
```

Elle récupère des informations globales comme des informations sur l'OS, le BIOS ou encore le CPU permettant d'avoir une vision global sur le système, obligatoire pour mener une analyse lors de la phase d'analyse à froid afin de connaître les version ou encore les mises à jour possibles. Un autre point important est qu'on peut retrouver l'utilisation mémoire et vérifier s'il y a une utilisation anormale de cette dernière.

Configuration réseau

Ensuite il faut récupérer la configuration réseau. On récupère cela avec la commande :

```
ipconfig /all
```

On pourra vérifier ici s'il n'y a pas de problème de configuration des interfaces ou bien des interfaces qui ne devrait pas exister. On pourra également visualiser les informations relatives au DHCP, DNS ou encore à la configuration IP.

Disques et lecteurs

Ensuite, on récupère l'organisation des périphériques de stockage pour visualiser l'ensemble des lecteurs (physiques, réseaux, ...). On fait cela avec la commande :

```
wmic logicaldisk get deviceid, volumename, description
```

Ces informations nous permettront de vérifier si les périphériques branchés sont légitimes. Par exemple, on pourra voir si des périphériques USB ou bien des lecteurs réseaux sont montés.

Ports

Nous avons jugé intéressant de récupérer l'ensemble des ports en écoute et les services derrières. Ces informations sont récupérées par la commande :

```
netstat -ab
```

Cette liste nous permet de vérifier que tous les ports ouverts sont bien ceux voulus par l'utilisateur. Par exemple, on pourra vérifier si un service malveillant est en écoute (Ex : bindshell) ou simplement des services non voulus accessible depuis l'extérieur.

Règles de pare-feu

Un autre point important, les règles de pare-feu. On obtient cela avec la commande :

```
netsh advfirewall firewall show rule name=all verbose
```

Avec ces règles, nous pourrons vérifier s'il y a un problème de configuration, avec par exemple une règle trop filtrante ou au contraire qui laisse passer des connexions non souhaités.

Police de sécurité

Nous avons également récupéré la police de sécurité globale du système avec la commande :

```
% secedit /export /areas SECURITYPOLICY
```

Cela permet de vérifier certains points imposés par cette police comme la complexité des mots de passe ou encore leur temps de validité. Il y a aussi d'autres points plus spécifiques comme le mécanisme protégeant du "brute-force" et sa configuration.

Mises à jour système

Les failles de sécurité sont patchées grâce aux mises à jour système. Il faut donc s'assurer que les dernières disponibles soient bien installées. Il faut également vérifier que seules les personnes autorisées puissent exécuter leur installation, afin d'empêcher d'installer des codes malicieux. Pour cela, notre script exporte le résultat de la commande suivante :

```
wmic qfe list full
```

Cette commande donne des informations sur les mises à jour installées comme par exemple :

- L'identifiant de la mise à jour
- La date d'installation
- L'utilisateur ayant fait l'installation

Variables d'environnements En ce qui concerne les variables d'environnement, elles sont récupérées différemment. En effet, on les récupère immédiatement avec notre objet `Shell` par l'intermédiaire du champ `Environment` (`WScript.Shell.Environment(type)`). On récupère quatre types de variables d'environnements :

- `System` : Vérification du PATH.
- `User` : Problème de configuration pour l'utilisateur.
- `Volatile` : Informations sur le compte courant. (compte administrateur dans notre cas)
- `Process` : Nombreuses informations de configurations (PATHs, CPU, systemroot, windir, ...)

1.3.2 Utilisateurs, Groupes et Permissions

Dans l'optique d'obtenir le maximum d'informations sur le comptes de la machine, nous utilisons 2 commandes :

- `wmic UserAccount`

  Cette commande permet de récupérer les noms des utilisateurs, leur identifiant ainsi que si le compte est désactivé ou non.
- `net user XXXXXXX`

  Cette commande est exécutée sur tous les comptes récupérés et permet d'obtenir des informations détaillées, notamment la date de dernière connexion, la date de dernier changement de mot de passe, les groupes de l'utilisateur, l'emplacement des scripts de connexion,...

Groupes

Afin de pouvoir vérifier les droits des utilisateurs, nous devons disposer des groupes présents sur la machine. Nous utilisons donc la commande suivante, qui nous donne la liste des groupes et leur identifiant :

```
wmic group
```

Privilèges des groupes et utilisateurs

Pour garantir la sécurité d'un système, il faut porter une attention particulière aux droits des différents utilisateurs, et ainsi limité au maximum les actions qu'ils peuvent faire. Pour vérifier que des utilisateurs ne puissent pas effectuer des actions qu'ils ne seraient pas censés pouvoir faire, nous exportons la liste des privilèges. Pour chacun d'eux, nous obtenons les identifiants des groupes et des utilisateurs autorisés.

```
secedit /export /areas USER_RIGHTS
```

Robustesse des mots de passe

Tous les comptes doivent disposer d'un mot de passe robuste (respectant au moins les recommandations de l'ANSSI) [1]. Il est donc nécessaire d'extraire les hashes des mots de passe des utilisateurs afin d'essayer de les casser. Pour cela, il faut extraire 2 parties du registre système : `SAM` (contenant les hashes chiffrés) et `System` (contenant la clef de déchiffrement).

```
reg save hklm\sam
reg save hklm\system
```

1.3.3 Services et processus

Afin de détecter la présence d'un malware, nous pouvons analyser les services activés et l'utilisation de la mémoire par les processus. Nous utilisons donc `sc query` dans le but de récupérer tous les services actifs sur la machine, et `tasklist` pour récupérer la liste de tous les processus et leur utilisation de la mémoire. Ces listes permettent de déterminer si un indésirable (connu) est en cours d'exécution.

1.3.4 Applications

Applications lancées au démarrage

Certaines applications sont lancées au démarrage du système, sans interaction de la part de l'utilisateur. Cela peut permettre notamment d'exécuter du code afin de récupérer le contrôle de la machine ou bien, par exemple faire sortir des informations vers un serveur externe. Notre script va donc lister les différentes applications lancées au démarrage, avec notamment le chemin vers l'exécutable lancé et l'utilisateur utilisé.

```
wmic startup
```

Applications installées

Certains logiciels ne sont jamais mis à jour sur les machines, et permettent l'exploitation de failles qui ne sont pas patchées. Notre script récupère donc la liste de toutes les applications installées avec notamment leur version.

```
wmic product
```

On pourra également vérifier la présence d'antivirus ou d'autre applications de sécurité au sein de cette liste.

1. `https://www.ssi.gouv.fr/uploads/2014/11/RGS_v-2-0_B2.pdf`

1.4 Stockage des informations

Les informations sont stockées dans les fichiers suivants :

Fichiers systèmes

- `audit_dumps\system\system_info.txt`
- `audit_dumps\system\network_info.txt`
- `audit_dumps\system\disks_info.txt`
- `audit_dumps\system\open_ports_info.txt`
- `audit_dumps\system\firewall_info.txt`
- `audit_dumps\system\security_policy.txt`
- `audit_dumps\system\updates.txt`
- `audit_dumps\system\environment_info.txt`

Fichiers utilisateurs, groupes et permissions

- `audit_dumps\user\groups_info.txt`
- `audit_dumps\user\users_info.txt`
- `audit_dumps\user\users_names.txt`
- `audit_dumps\user\users_full_details.txt`
- `audit_dumps\user\privileges.txt`
- `audit_dumps\user\sam_system_dump\sam`
- `audit_dumps\user\sam_system_dump\system`

Fichiers services et processus

- `audit_dumps\services_processes\services_info.txt`
- `audit_dumps\services_processes\process_info.txt`

Fichiers applications

- `audit_dumps\applications\launch_on_startup.txt`
- `audit_dumps\applications\applications.txt`

2 Liens utiles

VBS : https://repository.root-me.org/Programmation/VB/FR%20-%20Le%20language%20VBScript.pdf

VBS Doc : https://ss64.com/vb/

Variables d'environnement : https://stackoverflow.com/questions/904739/can-i-pick-up-environment-variables-in-vbscript-wsh-script

wmic : https://learn.microsoft.com/fr-fr/windows/win32/wmisdk/wmic

net user : https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc771865(v=ws.11)

net localgroup : https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc725622(v=ws.11)

systeminfo : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/systeminfo

ipconfig : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/ipconfig

netstat : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/netstat

netsh : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/netsh

secedit : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/secedit

reg save : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/reg-save

sc query : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/sc-query

tasklist : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/tasklist

makecab : https://learn.microsoft.com/fr-fr/windows-server/administration/windows-commands/makecab

3 Annexes

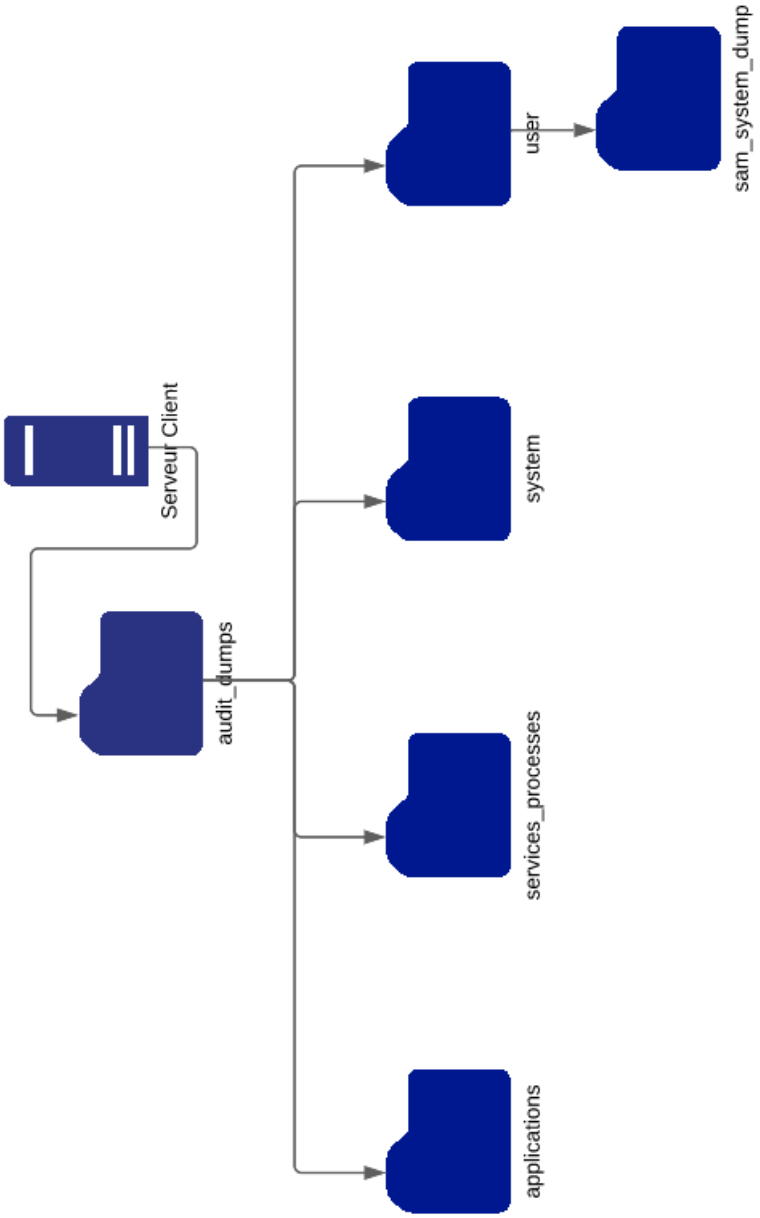

FIGURE 1 – Arborescence de dossier de dump