Fiche technique

Clé de registre suspecte

David Maciejak

Degré de difficulté

Cet article présente les manières de détecter les logiciels malveillants. Les méthodes peuvent être automatiques et se dérouler à l'aide d'un scanneur des failles de la sécurité qui supporte le test des scripts au local, comme Nessus.

Malware ou PuP (*Potentially unwanted Program* – programme potentiellement non-souhaité) doit stocker certaines données dans le système afin de ne pas entrer en interaction avec ce système. Cette démarche est nécessaire pour pouvoir créer et modifier les services, se connecter automatiquement lors du bootage et se connecter à Internet Explorer. Il est possible de détecter ces opérations qui ont lieu dans le système, et plus précisément dans le registre Windows.

La clé suspecte est un exemple d'un programme potentiellement non-souhaité via les modules *Browser Helper Object* pour l'application Internet Explorer.

Browser Helper Object (BHO) est un module *DLL*, conçu en tant que plug-in pour le navigateur Internet Microsoft Internet Explorer afin de fournir des fonctionnalités supplémentaires. Les modules BHO ont été ajoutés à IE en octobre 1997 avec la sortie de la version 4 de Internet Explorer. La plupart de modules BHO est chargé une fois par chaque nouvelle instance de Internet Explorer (*http://en.wikipedia.org/wiki/Browser_Helper_Object*).

Exemple d'utilisation : vous souhaitez détecter *CursorZone*, *Grip Toolbar*, un spyware connu recueillant des informations sur les visites et modifiant la page de recherche par défaut. Afin d'apprendre davantage à ce sujet, consultez :

- *http://www.castlecops.com/tk28918-gripcz4_dll.html*,
- *http://research.sunbelt-software.com/threatdisplay.aspx?name=Grip%20Toolbar&threatid=14981*.

Cet article explique...

- Comment détecter les traces des infections, en particulier sur une plate-forme Microsoft Windows.
- Comment écrire nos propres plug-ins Nessus à l'aide de NASL.

Ce qu'il faut savoir...

- Comment utiliser l'outil Nessus.
- Avoir des notions de NASL et/ou savoir écrire des scripts.
- Connaître les systèmes Microsoft Windows et Linux.

Listing 1. *Vérification de la clé du registre Grip Toolbar*

```
if(description) {
   script_id(16314);
   script_version("$Revision: 1.6 $");
   name["english"] = "Potentially unwanted software";
   script_name(english:name["english"]);
   desc["english"] = " This script checks for the presence of files and programs which might have been installed without
                     the consent of the user of the remote host. Verify each of softwares found to see if they are
                     compliant with you security policy. Solution : See the URLs which will appear in the report Risk
                     factor : High";
   script_description(english:desc["english"]);
   summary["english"] = "Checks for the presence of differents dll on the remote host";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2005 David Maciejak and Tenable Network Security");
   family["english"] = "Windows";
   script_family(english:family["english"]);
   script_dependencies("smb_hotfixes.nasl");
   script_require_keys("SMB/Registry/Enumerated");
   script_require_ports(139, 445);
   exit(0);
}
#load specific SMB function
include("smb_func.inc");
include("smb_hotfixes.inc");
if ( get_kb_item("SMB/samba") ) exit(0);
global_var handle;
#get port info from knowledge base
port = kb_smb_transport();
if(!port)exit(0);
#get credential and domain name from knowledge base
if(!get_port_state(port))return(FALSE);
login = kb_smb_login();
pass = kb_smb_password();
domain = kb_smb_domain();
soc = open_sock_tcp(port);
if(!soc)exit(0);
session_init(socket:soc, hostname:kb_smb_name());
#connect to IPC$ share
ret = NetUseAdd(login:login, password:pass, domain:domain, share:"IPC$");
if ( ret != 1 ) exit(0);
#open an handle on registry entry KKEY_CLASS_ROOT
handle = RegConnectRegistry(hkey:HKEY_CLASS_ROOT);
if ( isnull(handle) ) exit(0);
function check_reg(name, url, key, item, exp) {
   local_var key_h, value, sz;
   #open the registry key
   key_h = RegOpenKey(handle:handle, key:key, mode: MAXIMUM_ALLOWED);
   if( ! isnull(key_h) ) {
      #query the subkey if item != NULL
      value = RegQueryValue(handle:key_h, item:item);
      RegCloseKey(handle:key_h);
      if ( ! isnull(value) ) sz = value[1];
      else return 0;
   } else return 0;
   #if subkey equals the exp value that we expected
   if(exp == NULL || tolower(exp) >< tolower(sz)) {
      report = string("'", name, "' is installed on the remote host.\n", "Make sure that the user of the remote host
                     intended to install this software and that its use matches your corporate security policy.\n\n",
                     "Solution : ", url, "\n", "Risk factor : High");
      #report specific data based on this malware
      security_hole(port:kb_smb_transport(), data:report);
   }
}
i = 0;
```

Vous connaissez GUID (comme *Globally Unique Identifier*, un nombre de 128-bits unique créé par le système Windows ou par certaines applications Windows afin d'identifier un composant concret, une application, un fichier, une entrée dans la base de données et/ou un utilisateur) (`{4E7BD74F-2B8D-469E-A6E4-FC7CBD87BD7D}`) et l'un des noms de fichiers que vous ajoutez au système (*gripcz4.dll*).

Le script (Listing 1) se connecte au registre distant afin de vérifier si les GUID et le nom du fichier existent (c'est un fragment de *smb_suspicious_files.nasl*).

Le Listing 2 présente un exemple plus simple. Il n'y est pas nécessaire de tester les valeurs de l'entrée de la clé mais il faut seulement apprendre si une telle clé existe. Microsoft permet de se protéger contre la faille qui consiste à lancer le code en *Vector Markup Language* (afin d'apprendre davantage, consultez CVE-2006-4868). Les utilisateurs doivent se désinscrire de *vgx.dll*. Le plug-in vérifie si le fichier *DLL* a été désinscrit et il n'est plus pris en compte dans le registre.

Un démarrage automatique suspect lors du lancement du système

Les logiciels malveillants utilisent souvent les clés suivantes pour y ajouter des virus et lancer lors du démarrage du système :

- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run* – destiné au programme qui doit être lancé lors de chaque bootage. Remarquez que root path peut exister sous forme de *HKEY_CURRENT USER*,
- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices* – destiné au service qui doit être lancé lors de chaque bootage,
- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce* – destiné au service qui doit être lancé une fois lors du prochain bootage. Toutes ces trois clés et leurs entrées sont chargées de manière asynchrone.
- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce* – destiné au programme qui doit être lune fois lors du prochain bootage ; elles sont chargées de manière synchrone avec un ordre aléatoire. Remarquez que root path peut exister sous forme de *HKEY_CURRENT USER* (détails à l'adresse : *http://support.microsoft.com/kb/158022/en-us*)
- *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx* – concerne Windows 98 et Millennium.

Listing 1. *Vérification de la clé du registre Grip Toolbar - suite*

```
##############################################################################
name = make_list();
name[i] = "Grip Toolbar";
url[i] = "http://www.giantcompany.com/antispyware/research/spyware/spyware-Grip-Toolbar.aspx";
key[i] = "CLSID\{4E7BD74F-2B8D-469E-A6E4-FC7CBD87BD7D} \InprocServer32";
item[i] = NULL;
exp[i] = "gripcz4.dll";
##############################################################################
for(i=0;name[i];i++) {
   #call the check_reg function
   check_reg(name:name[i], url:url[i], key:key[i], item:item[i], exp:exp[i]);
}
RegCloseKey(handle:handle);
NetUseDel();
```

Tableau 1. *Valeur initiale des services Windows*

Type de lancement	Loader	Signification
0x0 (Boot)	Kernel	Il représente une partie de la pile pour le volume booté (lancé) et doit être donc chargé via Boot Loader.
0x1 (System)	I/O subsystem	Il représente le pilote chargé lors de l'initialisation du noyau.
0x2 (Auto load)	Service Control Manager	Chargé ou lancé automatiquement pour tous les démarrages, quelque soit le service.
0x3 (Load on demand)	Service Control Manager	Disponible indépendamment du type, mais lancé uniquement par un utilisateur (par exemple, en cliquant sur l'icône Outils dans le Panneau de bord).
0x4 (disabled)	Service Control Manager	A NE PAS LANCER A AUCUN CAS.

Listing 2. *Vérification de la désinscription de dll en Vector Markup Language*

```
if(description) {
   script_id(90001);
   script_cve_id("CVE-2006-4868");
   script_bugtraq_id(20096);
   script_version("$Revision: 1.0 $");
   name["english"] = "Vulnerability in Vector Markup Language Could Allow Remote Code Execution (925568)";
   script_name(english:name["english"]);
   desc["english"] = "
         Synopsis
         Arbitrary code can be executed on the remote host through the email client or the web browser.
         Description
         The remote host is running a version of Internet Explorer or Outlook Express which is vulnerable to a bug
         in the $ Vector Markup Language handling routine which may allow an attacker execute arbitrary code on the
         $ remote host by sending a specially crafted email.
         Solution
         A workaround is availabe at: http://www.microsoft.com/technet/security/advisory/925568.mspx.
         Risk factor
         High / CVSS Base Score: 8
         (AV:R/AC:H/Au:NR/C:C/A:C/I:C/B:N)";
   script_description(english:desc["english"]);
   summary["english"] = "Checks if VML is registered";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2006 Tenable Network Security");
   family["english"] = "Windows : Microsoft Bulletins";
   script_family(english:family["english"]);
   script_dependencies("smb_hotfixes.nasl");
   script_require_keys("SMB/Registry/Enumerated");
   script_require_ports(139, 445);
   exit(0);
}
#load smb modules
include("smb_func.inc");
include("smb_hotfixes.inc");
include("smb_hotfixes_fcheck.inc");
#get credential and hostname from kb
login = kb_smb_login();
pass = kb_smb_password();
domain = kb_smb_domain();
port = kb_smb_transport();
#exit if port is closed
if (!get_port_state(port))exit(1);
#open socket
soc = open_sock_tcp(port);
if (!soc) exit(1);
#init session, connect to IPC$ share
session_init(socket:soc, hostname:kb_smb_name());
r = NetUseAdd(login:login, password:pass, domain:domain, share:"IPC$");
if ( r != 1 ) exit(1);
hkcr = RegConnectRegistry(hkey:HKEY_CLASS_ROOT);
if ( isnull(hkcr) ) {
   NetUseDel();
   exit(1);
}
registered = 0;
#try to open given CLSID
key_h = RegOpenKey(handle:hkcr, key:"CLSID\{10072CEC-8CC1-11D1- 986E-00A0C955B42E}\InprocServer32",
      $ mode:MAXIMUM_ALLOWED);
if (!isnull(key_h)) {
   registered = 1;
}
RegCloseKey(handle:hkcr);
NetUseDel();
#if key was found
if (registered) security_hole (port);
```

Listing 3. *Vérification de l'entrée de la sous-clé du registre du ver NetSky*

```
if(description) {
   script_id(12070);
   script_version("$Revision: 1.1 $");
   name["english"] = "Netsky.B";
   vscript_name(english:name["english"]);
   desc["english"] = "
         This system appears to be infected by Netsky.B which is a mass-mailing worm that uses its own SMTP engine to $
         distribute itself to the email addresses it collects when probing local hard drives or remote mapped drives.
         Solution: Update your Anti-virus definitions file and perform a complete system scan.
         See also:
         http://vil.nai.com/vil/content/v_101034.htm,
         http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_NETSKY.B.
         Risk factor: High";
   script_description(english:desc["english"]);
   summary["english"] = "Detects Netsky.B Registry Key";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2003 Renaud Deraison modified by c.houle@bell.ca");
   family["english"] = "Windows";
   script_family(english:family["english"]);
   script_dependencie("netbios_name_get.nasl", "smb_login.nasl","smb_registry_access.nasl");
   script_require_keys("SMB/name", "SMB/login", "SMB/password", "SMB/domain","SMB/transport");
   script_require_ports(139, 445);
   exit(0);
}
include("smb_nt.inc");
#return item found under key on HKEY_LOCAL_MACHINE root
version = registry_get_sz(key:"SOFTWARE\Microsoft\Windows\CurrentVersion\Run", item:"service");
if ( ! version ) exit(0);
if("services.exe -serv" >< version ) security_hole(port);
```

Listing 4. *Vérification de la présence d'un service suspect : ver Sdbot*

```
if(description) {
   script_id(95000);
   script_version("$Revision: 1.0 $");
   name["english"] = "W32/Sdbot.worm.gen.by";
   script_name(english:name["english"]);
   desc["english"] = "
         This system appears to be infected by W32/Sdbot.worm.gen.by which is a worm spreading using multiple attacking
                     vector.
         Solution: Update your Anti-virus definitions file and perform a complete system scan.
         See also: http://vil.mcafeesecurity.com/vil/content/v_133133.htm.
         Risk factor: High";
   script_description(english:desc["english"]);
   summary["english"] = "Detects W32/Sdbot.worm.gen.by";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2006 D. Maciejak");
   family["english"] = "Windows";
   script_family(english:family["english"]);
   script_dependencie("netbios_name_get.nasl", "smb_login.nasl","smb_registry_access.nasl");
   script_require_keys("SMB/name", "SMB/login", "SMB/password",  "SMB/domain","SMB/transport");
   script_require_ports(139, 445);
   exit(0);
}
include("smb_nt.inc");
#return item found under key on HKEY_LOCAL_MACHINE root
filepath = registry_get_sz(key:"system\currentcontrolset\services\rpcsvc", item:"imagepath");
if ( ! filepath ) exit(0);
if("\System32\rpcsvc.exe" >< filepath ) security_hole(port);
```

Listing 5. *Vérification si le service anti-virus Nod32 n'a pas été désactivé.*

```
if(description) {
   script_id(95001);
   script_version ("$Revision: 1.0 $");
   name["english"] = "Nod32 antivirus disabled";
   script_name(english:name["english"]);
   desc["english"] = "
         Synopsis :
         Remote system is not configured for running Nod32 antivirus.
         Description:
         The remote host does not have Nod32 antivirus enabled.
         Solution:
         Enable Nod32 service on this host
         Risk factor:
         None";
   script_description(english:desc["english"]);
   summary["english"] = "Determines if Nod32 antivirus is disabled";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2007 David Maciejak");
   family["english"] = "Windows";
   script_family(english:family["english"]);
   script_dependencies("netbios_name_get.nasl", "smb_login.nasl", "smb_registry_access.nasl");
   script_require_keys("SMB/transport", "SMB/name", "SMB/login", §  "SMB/password", "SMB/registry_access");
   script_require_ports(139, 445);
   exit(0);
}
include("smb_func.inc");
port = get_kb_item("SMB/transport");
if(!port)port = 139;
#load credential and host data from kb
name = kb_smb_name(); if(!name)exit(0);
login = kb_smb_login();
pass = kb_smb_password();
domain = kb_smb_domain();
port = kb_smb_transport();
#test if port is closed
if ( ! get_port_state(port) ) exit(0);
#open socket
soc = open_sock_tcp(port);
if ( ! soc ) exit(0);
#init session to host
session_init(socket:soc, hostname:name);
#open share with credential
r = NetUseAdd(login:login, password:pass, domain:domain, share:"IPC$");
if ( r != 1 ) exit(0);
#connect to HKEY_LOCAL_MACHINE registry
hklm = RegConnectRegistry(hkey:HKEY_LOCAL_MACHINE);
if ( isnull(hklm) ) {
   NetUseDel();
   exit(0);
}
key = "SYSTEM\CurrentControlSet\Services\NOD32krn";
item = "Start";
#open NOD32 key
key_h = RegOpenKey(handle:hklm, key:key, mode:MAXIMUM_ALLOWED);
if ( ! isnull(key_h) ) {
   #catch the value
   value = RegQueryValue(handle:key_h, item:item);
   #test if value is set to disable mode
   if (!isnull (value) && ((value[1] == 3)||(value[1] == 4))
         #if so it's stange, we report it
         security_note (port);
   RegCloseKey (handle:key_h);
}
RegCloseKey (handle:hklm);
NetUseDel ();
```

Les entrées et les sections sont triées de manière alphabétique afin de forcer un ordre déterministe ; un processus séparé n'est pas créé pour chaque entrée. La liste de dépendances des modules dll est chargée (afin d'apprendre davantage, consultez *http://support.microsoft.com/?scid=kb%3Benus%3B232487&x=18&y=13*). Exemple d'utilisation : le ver *Netsky.B* se propage et vous voulez savoir si les stations de travail ont été infectées. Après la lecture de Trend-Micro (*http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_NETSKY.B*), vous savez qu'il ajoute une entrée de la sous-clé appelée service avec la valeur services.exe -serv à la clé du registre *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run*, afin d'effectuer un démarrage automatique lors du bootage.

Le script (Listing 3) ouvre le registre, vérifie si une clé suspecte est présente et teste la valeur de la sous-clé.

Service suspect

Les logiciels malveillants s'installent parfois comme un service, afin de se lancer automatiquement lors du bootage. Pour ce faire, ces programmes doivent ajouter une entrée au registre dans *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\servicename*.

Ensuite, ils doivent paramétrer l'entrée de la sous-clé appelée `ImagePath` contenant le chemin aux données exécutables qui seront chargées dans la mémoire (Listing 9). `driverName` ou `serviceName` y ont la même valeur que le nom lié de la sous-clé `Services` (dans notre cas servicename) (afin d'apprendre davantage, consultez *http://support.microsoft.com/kb/103000*).

Le script de détection détermine l'entrée dans le registre. Regardez le Listing 4.

Service suspect chargé

Toute entrée dans *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services* est un service et dispose d'une clé de démarrage (constante `REG_DWORD`). Cette clé définit les valeurs pour les services donnés. Regardez le Tableau 1.

La valeur initiale est ignorée par les adaptateurs. Si le `Type` est une valeur *Win32 Service*, la valeur initiale doit être paramétrée à `Auto`, `Demand`, ou `Disabled`. Consultez également : *http://support.microsoft.com/kb/103000 for details*.

Exemple d'utilisation : ver appelé *W32/Sdbot.worm.gen.by* (regardez la description à l'adresse *http://vil.mcafeesecurity.com/vil/content/v_133133.htm*) ajoute une entrée suspecte de la clé à `hkey_local_machine\system\currentcontrolset\services\rpcsvc\imagepath="%WinDir%\System32\rpcsvc.exe"`.

Service suspect désactivé

Conformément à la table ci-dessus, la valeur de la clé paramétrée à `0x3` ou `0x4` peut éveiller des soupçons. Une action recommandée consiste à vérifier si le service anti-virus n'a pas été désactivé.

Le Listing 5 présente le fonctionnement du plug-in.

Compte d'utilisateurs/groupes suspect, vol du mot de passe

Nessus est doté des mécanismes qui servent à détecter les noms de compte suspects, les comptes sans mots de passe et les comptes/mots de passe connus.

Nous vous présentons ci-après comment fonctionne le plug-in, chargé de vérifier si un compte appelé *hax0r* avec un mot de passe vide existe dans le système. La démarche repose sur la fonction `check_account`, qui tente de se connecter à la cible distante à l'aide des protocoles telnet ou ssh. Regardez le Listing 6.

Listing 6. *Vérification la présence du compte hax0r avec un mot de passe vide*

```
if(description) {
   script_id(11253);
   script_version ("$Revision: 1.10 $");
   script_cve_id("CVE-1999-0502");
   script_name(english:"Unpassworded hax0r account");
   desc["english"] = "
         The account 'hax0r' has no password set.
         An attacker may use it to gain further privileges on this system
         Risk factor: High
         Solution: Set a password for this account or disable it";
   script_description(english:desc["english"]);
   script_summary(english:"Logs into the remote host");
   script_category(ACT_GATHER_INFO);
   script_family(english:"Default Unix Accounts");
   script_copyright(english:"This script is Copyright (C) 2003 Renaud
                     Deraison");
   script_dependencie("find_service.nes", "ssh_detect.nasl");
   script_require_ports("Services/telnet", 23, "Services/ssh", 22);
   script_require_keys("Settings/ThoroughTests");
   exit(0);
}
#load needed functions
include("default_account.inc");
include("global_settings.inc");
if ( ! thorough_tests ) exit(0);
account = "hax0r";
#if no optional 2nd parameter is given in check_account(login, password), ←
                     it #check against an empty password
port = check_account(login:account);
if(port)security_hole(port);
```

Listing 7. *Vérification des droits de la clé du registre*

```
if(description) {
   script_id(11867);
   script_bugtraq_id(2065);
   script_version ("$Revision: 1.8 $");
   script_cve_id("CVE-2001-0047");
   name["english"] = "SMB Registry : permissions of the Microsoft Transaction Server key";
   script_name(english:name["english"]);
   desc["english"] = "
         Synopsis :
         A local user can gain additional privileges.
         Description:
         The registry key HKLM\SOFTWARE\Microsoft\Transaction Server\Packages
         can be modified by users not in the admin group.
         Write access to this key allows an unprivileged user to gain additional
         privileges.
         See also:
         http://www.microsoft.com/technet/security/bulletin/ms00-095.mspx
         Solution:
         Use regedt32 and set the permissions of this key to:
         - admin group: Full Control
         - system: Full Control
         - everyone: Read
         Risk factor:
         Medium / CVSS Base Score: 5
         (AV:L/AC:L/Au:NR/C:P/A:P/I:P/B:N)";
   script_description(english:desc["english"]);
   summary["english"] = "Determines the access rights of a remote key";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2003 Tenable Network Security");
   family["english"] = "Windows";script_family(english:family["english"]);
   script_dependencies("netbios_name_get.nasl", "smb_login.nasl", "smb_registry_access.nasl");
   script_require_keys("SMB/transport", "SMB/name", "SMB/login", "SMB/password", "SMB/registry_access");
   script_require_ports(139, 445);
   exit(0);
}
#load SMB function
include("smb_func.inc");
access = get_kb_item("SMB/registry_access");
if(!access)exit(0);
port = get_kb_item("SMB/transport");
if(!port)port = 139;
#get credential and hostname from kb
name = kb_smb_name(); if(!name)exit(0);
login = kb_smb_login();
pass = kb_smb_password();
domain = kb_smb_domain();
port = kb_smb_transport();
#check if port is still open
if ( ! get_port_state(port) ) exit(0);
#open network connection
soc = open_sock_tcp(port);
if ( ! soc ) exit(0);
#init session to hostname
session_init(socket:soc, hostname:name);
#using given credential
r = NetUseAdd(login:login, password:pass, domain:domain, share:"IPC$");
if ( r != 1 ) exit(0);
#connect to registry key
hklm = RegConnectRegistry(hkey:HKEY_LOCAL_MACHINE);
#exit if key does not exist
if ( isnull(hklm) ) {
   NetUseDel();
   exit(0);
}
```

Listing 7. *Vérification des droits de la clé du registre - suite*

```
key = "SOFTWARE\Microsoft\Transaction Server\Packages";
#open key SOFTWARE\Microsoft\Transaction Server\Packages
key_h = RegOpenKey(handle:hklm, key:key, mode:MAXIMUM_ALLOWED |ACCESS_
                    SYSTEM_SECURITY);
#check if key exist
if(!isnull(key_h)) {
   #get key privilege
   rep = RegGetKeySecurity (handle:key_h, type: DACL_SECURITY_INFORMATION
         | SACL_SECURITY_INFORMATION | § GROUP_SECURITY_INFORMATION |
         OWNER_SECURITY_INFORMATION);
   #check if key is writable by non admin users
   if(!isnull(rep) && registry_key_writeable_by_non_admin(security_
                    descriptor:rep)) {
      #if so it's a flaw
      security_warning(port);
   }
   #close registry connection
   RegCloseKey (handle:key_h);
}
#close registry connection
RegCloseKey (handle:hklm);
#close network connection
NetUseDel();
```

Listing 8. *Vérification des droits du shell*

```
#check if Nessus version is up to date enough (>= 2.1.1) to know the §
pread #function
if ( ! defined_func("pread") ) exit(0);
if(description) {
   script_id(95002);
   script_version ("1.0");
   name["english"] = "Shell SUID permission";
   script_name(english:name["english"]);
   desc["english"] = "
         This plugin checks wrong permission on /bin/bash and /bin/zsh
         Risk factor: High";
   script_description(english:desc["english"]);
   summary["english"] = "Check SUID bit on shells";
   script_summary(english:summary["english"]);
   script_category(ACT_GATHER_INFO);
   script_copyright(english:"This script is Copyright (C) 2007 §  David
         Maciejak");
   family["english"] = "Policy Compliance";
   script_family(english:family["english"]);
   script_dependencies("ping_host.nasl", "ssh_settings.nasl");
   exit(0);
}
#load SSH functions
include("ssh_func.inc");
buf = "";
port=kb_ssh_transport();
# On the local machine, just run the command
if (islocalhost()) {
   #execute command with process read
   buf = pread(cmd: "ls", argv: make_list("ls", "-l", "/bin/bash", §
         "/bin/ksh"));
} else {
   #try to open or reuse an existing SSH connection
   sock = ssh_login_or_reuse_connection();
   if (! sock) exit(0);
   #if socket is open, execute command remotely
   buf = ssh_cmd(socket:sock, cmd:"/bin/ls -l /bin/bash /bin/ksh",
         §  timeout:60);
   ssh_close_connection();
```


À propos de l'auteur

David Maciejak habite en France et il est spécialiste de protection. Il a passé une partie de son temps libre à travailler sur les projets opens source, comme Nessus, Metasploit et Snort.

Les utilisateurs de l'application Nessus 3 peuvent profiter de l'option permettant de vérifier la conformité. Ils peuvent vérifier la configuration des périphériques des systèmes UNIX et Windows. Ils peuvent vérifier la durée de validité minimale/maximale du mot de passe, la longueur minimale du mot de passe, la composition du mot de passe et disposent de nombreuses autres possibilités.

Exemple :

```
<item>
   name: "min_password_length"
   description: "Minimum password
                        length"
   value: "14..MAX"
</item>
```

Cet audit vérifie si la longueur minimale du mot de passe dans le système UNIX est égale à 14 caractères.

Pour apprendre davantage, consultez : *http://cgi.tenablesecurity.com/nessus_compliance_checks.pdf*.

Privilèges et droits d'objets suspects

Le privilège dans le registre SMB peut être non conforme à la politique de sécurité, ce qui peut se traduire par la divulgation des informations, permettant à un utilisateur non autorisée d'accéder aux données.

Le script (Listing 7) vérifie *HKLM\SOFTWARE\Microsoft\Transaction Server\Packages*, regardez aussi *MS00-095*,

Il est parfois possible de vérifier dans la famille des systèmes UNIX les droits à l'aide des plug-ins locaux. Le script (Listing 8.) teste si les shells ordinaires ont un *SUID* de *root* ; ce type de faille peut donner potentiellement à un attaquant un accès depuis le niveau de root.

Listing 8. *Vérification des droits du shell -suite*

```
}
if (! buf) {
   display("could not execute command\n");
   exit(0);
}
#split each line from the buffer
lines=split(buf);
shells = "";
foreach line (lines) {
   #typical line is -rwsr-xr-x 1 root root 664084 2006-04-22 00:51 /bin/bash
   #check for SUID
   if (eregmatch(pattern: '^-..s', string: line, icase: 0)) {
      buf2=split(line, sep:' ');
      #check file owner
      if (chomp(buf2[2]) >< "root") {
         #shellname is the last from ls command
         shellname=buf2[max_index(buf2)-1];
         shells += '\n' + shellname;
      }
   }
}

if (shells >!< "") {
   report = desc["english"] + '\n\nPlugin output :\n\nThe shells below have §
   incorrect permission :\n' + shells;
   security_hole(port:port, data:report);
}
exit(0);
```

Listing 9. *Fragment du registre du système Windows*

```
ImagePath REG_DWORD Path and
      filename
Specifies a path name.
For adapters, this value
      is ignored.
Default: For a driver:
%WinDir%\SYSTEM32\DRIVERS\
      driverName.SYS
For a service:
%WinDir%\SYSTEM32\
      § serviceName.EXE
```

Listing 10. *Exemple d'un audit*

```
<custom_item>
   sytem: "Linux"
   type: FILE_CHECK
   description: "Permission and
   ownership check for /bin/bash"
   file: "/bin/bash"
   owner: "root"
   group: "root"
   mode: "-rwxr-xr-x"
</item>
```

Remarquez que ce type du script qui utilise les commandes externes doit être signé par l'équipe Nessus pour des raisons de sécurité (via la commande `nasl -S`), avant d'être utilisé. Il faut paramétrer l'option `nasl_no_signature_check` à la valeur yes w nessusd.conf, ce qui fait en sorte que le serveur Nessus fait un détour de toutes les signatures de scripts et il charge/exécute les scripts indépendamment de l'authenticité des signatures.

En ce qui concerne les systèmes Windows, ce plug-in peut être également exécuté par une combinaison de la connexion SMB ou l'installation du système SSH.

Si vous disposez de Nessus 3 direct feed, vous devriez vérifier la conformité chez vous.

L'audit `FILE_CHECK` est utilisé pour détecter un fichier donné et ses paramètres. Le Listing 10 présente l'équivalent de ce qui fait le script. ●