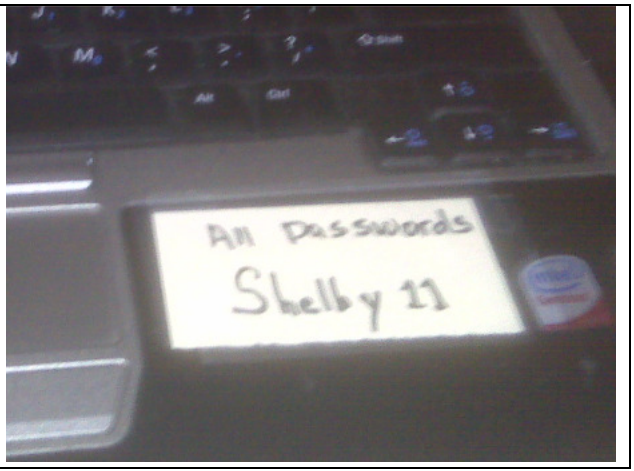
Homegrown Dictionary/Rainbow Table Password Discovery

Author: brian.wuchner [@] enterpriseadmins.org

Sometimes it is really easy to figure out a user's password. Just check out that picture to the right. Other times the password is less obvious but could come from a list of passwords you already know. For example, if your helpdesk always resets passwords to the same few values, many users could be using one of those default passwords. Here is a simple process for those who would like to audit your directory for such accounts.	

Files Needed:

File Name	Description
`CreatePassControlUsers.vbs`	VBscript that creates users with known passwords from passlist.txt.
`Passlist.txt`	The list of passwords you would like to include for comparison.
`PwDump.exe`	NTLM and LanMan password grabber utility obtained from http://www.foofus.net/fizzgig/pwdump/.
`Pwd-dump.txt`	The exported password hash list we generated using PwDump.exe
`BuildPassControlDict.vbs`	Reads pwd-dump.txt and generates output code for the AnalyzeForKnownPasswords.vbs script.
`AnalyzeForKnownPasswords.vbs`	Reads pwd-dump.txt and generates list of accounts with known passwords.
`Results.txt`	The file containing our final product: user names and passwords which were compromised.

THIS PROCESS AND ACCOMPANYING SCRIPTS WERE DESIGNED FOR EDUCATIONAL PURPOSES IN A NON-PRODUCTION ENVIRONMENT. USE AT YOUR OWN RISK.

Using CreatePassControlUsers.vbs and Passlist.txt

Edit line 32 of CreatePassControlUsers.vbs to include the NetBIOS domain/workstation/server name where the control users should be created. For this example, I'm creating test users in a 406PASSWORD active directory domain specifically created for this exercise.

Create a text file with one password per line. We will create a user account for each of these passwords. This will generate a password hash stored in the Windows SAM database. I have selected 15 known passwords from the 10,000 accounts and created earlier and included them in my passlist.txt file.

When ready run "**cscript CreatePassControlUsers.vbs**" to create your test/control users:

```
C:\Users\Administrator\Desktop\CreateUsers>cscript CreatePassControlUsers.vbs
Created user account for passctrl-mamo#1 with password mamo#1
Created user account for passctrl-anmo#1 with password anmo#1
Created user account for passctrl-maro#1 with password maro#1
Created user account for passctrl-anro#1 with password anro#1
Created user account for passctrl-maha#1 with password maha#1
Created user account for passctrl-maca#1 with password maca#1
Created user account for passctrl-maal#1 with password maal#1
Created user account for passctrl-anca#1 with password anca#1
Created user account for passctrl-anha#1 with password anha#1
Created user account for passctrl-anal#1 with password anal#1
Created user account for passctrl-jomo#1 with password jomo#1
Created user account for passctrl-jemo#1 with password jemo#1
Created user account for passctrl-samo#1 with password samo#1
Created user account for passctrl-damo#1 with password damo#1
Created user account for passctrl-stmo#1 with password stmo#1
```

Using PwDump.exe to export hashes

pwdump is the name of various Windows programs that output the LM and NTLM password hashes of local user accounts from the Security Account Manager (SAM). In order to work, it must be run under an Administrator account, or be able to access an Administrator account on the computer where the hashes are to be dumped; so pwdump does not compromise security. I should mention, however, that many antivirus products see this file as a threat.

For our example we will be using the following command line switches:

-n = No History

-o = Output file name

localhost = the machine name used for the export. This tool can be used remotely by specifying a remote hostname instead of localhost.

```
C:\pwdump6-2.0.0-beta-exe-only>PwDump.exe -n -o pwd-dump.txt localhost

pwdump6 Version 2.0.0-beta-2 by fizzgig and the mighty group at foofus.net
```

```
** THIS IS A BETA VERSION! YOU HAVE BEEN WARNED. **
Copyright 2009 foofus.net

This program is free software under the GNU
General Public License Version 2 (GNU GPL), you can redistribute it and/or
modify it under the terms of the GNU GPL, as published by the Free Software
Foundation.  NO WARRANTY, EXPRESSED OR IMPLIED, IS GRANTED WITH THIS
PROGRAM.  Please see the COPYING file included with this program
and the GNU GPL for further details.

No history available
Waiting for remote service to terminate...
Servers with many user accounts can take several minutes
.................
Completed.
```

Running this command produces a 1.7mb file in our test lab. Opening the file in notepad shows several thousand lines like the following:

```
Administrator:500:NO PASSWORD*********************:1BCD4DC36E2AD8C964D73D7DD96701A3:::
Guest:501:NO PASSWORD*********************:NO PASSWORD*********************:::
krbtgt:502:NO PASSWORD*********************:1EE9E3D7EA563517759DA31336D08F19:::
johnsmith:1103:NO PASSWORD*********************:1888C2DF7CFDEB0C0DD94280FA172622:::
johnjones:1104:NO PASSWORD*********************:293C96B09B9CA2575DC2C704730F6CD3:::
johnjohnson:1105:NO PASSWORD*********************:293C96B09B9CA2575DC2C704730F6CD3:::
johnlee:1106:NO PASSWORD*********************:7511B9D3D99D3C5CC025F5FB6878F310:::
johnbrown:1107:NO PASSWORD*********************:921F6119D74A2A6AD1861BCDBBD08D73:::
johnwilliams:1108:NO PASSWORD*********************:0F20C565D86031AACEF5E324E7235646:::
```

Using BuildPassControlDict.vbs

My 1.7mb pwd-dump.txt file has over 10,000 user accounts. Mixed in this list are 15 password control hashes that I'm interested in for another script. Instead of reading through this huge file, I've created a script to find the passctrl- User IDs that I'm interested in.

You start this process by running "**`cscript BuildPassControlDict.vbs`**"

```
dict.add "13F9750B80A0DA7AF0FF8FB6B5B16BCA","mamo#1"
dict.add "F09BD538580DDDFF2021565F906FD36C","anmo#1"
dict.add "32C9CD701FED94C35EFFC7E963D74C24","maro#1"
```

```
dict.add "E920FE3B21D574CD8B4A782CCCC92F56","anro#1"
dict.add "51E00BF26CC0D6A8337A05231E2D190A","maha#1"
dict.add "682A7300FA111765DD710F1F7D24B119","maca#1"
dict.add "71F3E3094AE6DA90A5CD8160BE4310DA","maal#1"
dict.add "F06391B5DE3704A1775F6A14EDD2A354","anca#1"
dict.add "E2D7039C74F6C5ED2EA5CB8C4639BBDC","anha#1"
dict.add "FF195E1A6CECECD9744B3012B60CA113","anal#1"
dict.add "00EEB464DE357124C47CE4CBF5B5B454","jomo#1"
dict.add "CC3F89FD1D2753F0BD826C3267921154","jemo#1"
dict.add "DB4779936218D24743F82B2D7E2CC6E9","samo#1"
dict.add "75443FD3D002F304C177FFE1C3CB48AB","damo#1"
dict.add "5AC954BC72A98623439A6782ED612D0D","stmo#1"
```

The script that reads my 1.7mb file looking for passctrl- users actually does one other thing. It rearranges the values (by splitting on the colon character) and prefixes the password information with dict.add. This will allow me to paste my dictionary into another VBscript. See the next section for more detail.

Using AnalyzeForKnownPasswords.vbs

Edit the AnalyzeForKnownPasswords.vbs file. On line 8 (after the Scripting.Dictionary object has been defined but before the While/Wend loop) insert your password control values. You are now ready to discover user accounts that have known password values. Depending on the size of your list this may take a few seconds. When complete you should be able to see accounts that have known passwords in results.txt:

```
388 accounts processed in 1 seconds.
```

The next audit

The first time you use this process you'll probably need to follow all four steps. This will allow you to generate your very own "control accounts" with known passwords, export account information, build your "rainbow table" and audit the accounts. For subsequent audits you can probably get away with just half the amount of work. For the next audit you'll really only need to follow the sections "Using PwDump.exe to export hashes" and "Using AnalyzeForKnownPasswords.vbs".