Decrypt EFS-encrypted files without a cert backup

https://tinyapps.org/docs/decrypt-efs-without-cert-backup.html

tinyapps.org / docs / Decrypt EFS-encrypted files without a cert backup

---

Windows users may unintentionally enable EFS encryption (even from just unpacking a ZIP file created under macOS), resulting in errors like these when trying to copy files from a backup or offline system, even as root:

- Windows:
  - File Access Denied
  - Access is denied.
- macOS:
  - The operation can’t be completed because you don’t have permission to access some of the items.
  - Permission denied
- Linux:
  - Error splicing file: Permission denied
  - Permission denied

Despite popular perception ("*If you don't have a copy of the certificate then your files are forever lost.*", "*If you didn't export the encryption certificates from the computer that encrypted the files then the data in those files is gone forever*", etc.), it may be possible to create the necessary certificate from an offline system or backup thanks to

Benjamin Delpy's mimikatz and his guide *howto ~ decrypt EFS files*. Here is an abbreviated (and by turns amplified) version:

0. Copy necessary files

From the offline system, copy these folders and paste them into the directory containing mimikatz.exe on a running system:

- %USERPROFILE%\AppData\Roaming\Microsoft\
  - SystemCertificates\
  - Crypto\
  - Protect\

If the password is unknown, copy these two files as well:

- %WINDIR%\system32\config\
  - SAM
  - SYSTEM

1. Retrieve certificate thumbprint from one of the encrypted files

```
cipher /c "D:\Users\foo\Pictures\secret.jpg"

...
    Certificate thumbprint: 096B A4D0 21B5 0F5E 78F2 B985 4A74 6167 8EDA A006
  No recovery certificate found.
  Key information cannot be retrieved.
The specified file could not be decrypted.
```

2. Export certificate and its public key to DER

```
mimikatz #
```

```
crypto::system /file:"SystemCertificates\My\Certificates\096BA4D021B50F5E78F2B9854A7461678EDAA006"
/export

...
        Key Container  : d209e940-6952-4c9d-b906-372d5a3dbd50
        Provider       : Microsoft Enhanced Cryptographic Provider v1.0
...
  Saved to file: 096BA4D021B50F5E78F2B9854A7461678EDAA006.der
```

3. Find the master key

Check files within Crypto\RSA*SID*\ to find the one containing a pUniqueName which matches the key container found in step 2, e.g.,

```
mimikatz #
dpapi::capi /in:"Crypto\RSA\S-1-5-21-3425643682-3879794161-2639006588-
1000\43838b0ac634d4f965f7c24f0fa91b2b_a55eeef9-ab65-4716-a466-adfc937caecd"

...
  pUniqueName        : d209e940-6952-4c9d-b906-372d5a3dbd50
...
  guidMasterKey      : {92f17fce-aae6-488b-9fd8-7774c6c3eb16}
```

4. Recover NTLM hash if necessary

If the password is unknown, recover the NTLM hash:

```
mimikatz #
lsadump::sam /system:SYSTEM /SAM:SAM

...
RID  : 000003e8 (1000)
User : foo
  Hash NTLM: 31d6cfe0d16ae931b73c59d7e0c089c0
```

For domain accounts, you'll only need the NTLM hash (`/hash:xx`); for local accounts, you'll need *either* the corresponding password (`/password:xx`) or its SHA1 hash (`/hash:xx`), which means knowing, cracking, or looking it up:[1]

- Lookup online:
  - Hashes.com
  - CrackStation
  - Ntlm() Encrypt & Decrypt
  - HashKiller

- Lookup offline:
  - Rainbow Crackalack
  - FreeRainbowTables.com
- Crack via hashcat or similar

5. Decrypt the master key

In this example, we have a local account with an NTLM hash

of 31d6cfe0d16ae931b73c59d7e0c089c0, which corresponds to a blank password and a SHA1 hash of da39a3ee5e6b4b0d3255bfef95601890afd80709:

```
mimikatz #
 dpapi::masterkey /in:"Protect\S-1-5-21-3425643682-3879794161-2639006588-1000\92f17fce-aae6-488b-9fd8-
 7774c6c3eb16" /hash:da39a3ee5e6b4b0d3255bfef95601890afd80709

...
[masterkey] with hash: da39a3ee5e6b4b0d3255bfef95601890afd80709 (sha1 type)
  key : 6e24723a56a885fc957f25d4872cbbf10589b1f08033d32174ef3618a192f0e101e41196ca76d68905773742
  sha1: 4505118da94b7df471bbbcf6d2c6c744a612e62b
```

6. Decrypt the private key

```
mimikatz #
 dpapi::capi /in:"Crypto\RSA\S-1-5-21-3425643682-3879794161-2639006588-
 1000\43838b0ac634d4f965f7c24f0fa91b2b_a55eeef9-ab65-4716-a466-adfc937caecd"
 /masterkey:4505118da94b7df471bbbcf6d2c6c744a612e62b

...
        Private export : OK - 'raw_exchange_capi_0_d209e940-6952-4c9d-b906-372d5a3dbd50.pvk'
```

7. Build PFX certificate

with OpenSSL:[2]

```
openssl.exe x509 -inform DER -outform PEM -in 096BA4D021B50F5E78F2B9854A7461678EDAA006.der -out
public.pem
```

```
openssl.exe rsa -inform PVK -outform PEM -in raw_exchange_capi_0_d209e940-6952-4c9d-b906-372d5a3dbd50.pvk -out private.pem

writing RSA key

openssl.exe pkcs12 -in public.pem -inkey private.pem -password pass:bar -keyex -CSP "Microsoft Enhanced Cryptographic Provider v1.0" -export -out cert.pfx
```

8. Install PFX certificate

```
certutil -user -p bar -importpfx cert.pfx NoChain,NoRoot

Certificate "user" added to store.
CertUtil: -importPFX command completed successfully.
```

9. Access your files!

Your files should now be accessible, but you may want to take this opportunity to decrypt them:

```
cipher /d "D:\Users\foo\Pictures\secret.jpg"


cipher /d /s:"D:\Users\foo\Pictures\"
```

(or right click → Advanced → uncheck "Encrypt contents to secure data" → OK).

Footnotes

1. Benjamin mentions a few other possibilities: domain backup key, CREDHIST, and extracting NTLM & SHA1 hashes along with masterkeys from a full memory dump.
2. 3gstudent suggests using cert2spc.exe and pvk2pfx.exe instead of openssl.exe:

   ```
   cert2spc.exe 096BA4D021B50F5E78F2B9854A7461678EDAA006.der public.spc
   pvk2pfx.exe -pvk raw_exchange_capi_0_d209e940-6952-4c9d-b906-372d5a3dbd50.pvk -pi test -spc public.spc -pfx cert.pfx -f
   ```

   A potential downside of this approach is having to download the 810MB Windows 10 SDK rather than a 2MB OpenSSL binary; on the other hand, you don't have to trust a third

party. Mount the Windows 10 SDK ISO and extract cert2spc.exe and pvk2pfx.exe via lessmsi; find cert2spc.exe in Installers\Windows SDK Signing Tools-x86_en-us.msi (ARM, x64, and x86 versions included) and pvk2pfx.exe in Installers\Windows SDK Desktop Tools x86-x86_en-us.msi, Installers\Windows SDK Desktop Tools x64-x86_en-us, and Installers\Windows SDK Desktop Tools arm64-x86_en-us.msi.

Sources

- howto ~ decrypt EFS files
- Retrieving lost Windows 10 password, using Kali Linux, mimikatz and hashcat

Related

- Search for EFS-encrypted files: `cipher /u /n`
- View or backup existing certs via reykeywiz.exe or certmgr.msc
- Advanced EFS Data Recovery "helps recovering the encrypted files under various circumstances.
  - EFS-protected disk inserted into a different PC
  - Deleted users or user profiles
  - User transferred into a different domain without EFS consideration
  - Account password reset performed by system administrator without EFS consideration
  - Damaged disk, corrupted file system, unbootable operating system
  - Reinstalled Windows or computer upgrades
  - Formatted system partitions with encrypted files left on another disk"
- Encrypting File System
- About EFS (Encryption File System)
- So my dad asked me to help regain access to some "encrypted files"...
- encrypted file system recovery
- Files remain encrypted after you copy the files from an encrypted folder to a WebDAV share if the files are copied by using a computer that is running Windows 7 or Windows Server 2008 R2
- Encrypting File System (EFS) files appear corrupted when you open them
- HOW TO: Prevent Files from Being Encrypted When Copied to a Server
- To Create A Personal Information Exchange (PFX) File
- MCTS 70-680: Encrypting File System (EFS)

- EFS and decrypting a file:

> *If you have your original profile, you can use "reccerts" tool to retrieve the private key to recovery EFS file.*
> ...
> `reccerts.exe -path: "profile path" -password:<password>`
> *But you have to contact to Microsoft Support to get this tool.*

---

created: 2019.10.18, updated: 2022.11.19

Revision #1
Created 23 December 2023 15:11:23 by ColtM
Updated 13 June 2024 01:28:02 by ColtM