mimikatz 2.0

Benjamin DELPY `gentilkiwi`
Our little story

`whoami`, why am I doing this?
mimikatz 2.0 & sekurlsa
Focus on Windows 8.1 et 2012r2
Kerberos & strong authentication

Questions / Answers

And of course, some demos during the session
(and stickers ;)

09/07/2014 Benjamin DELPY `gentilkiwi` @ 15th RMLL/LSM benjamin@gentilkiwi.com ; blog.gentilkiwi.com
`whoami`? Why `mimikatz`?

Benjamin DELPY `gentilkiwi`:
- Kiwi addict, I code, but when it’s done, I tweet about it: @gentilkiwi
- lazy efficient
- I don’t work as pentester/searcher/technical guy, I do it as a Kiwi (nights)
- I use Windows (but also OpenBSD)
  - is the enemy of your enemy your friend? ;)

`mimikatz`:
- born 2007
- is not a hacking tool (seriously)
- is coded for my personal needs
- can demonstrate some security concept
  - Have you ever try to demonstrate “theoretical” risks and to obtain reaction? acts? (budgets?)
- try to follow Microsoft's evolution (who’s the cat/mouse?)
- is not enough documented! (I know, but I work on it on GitHub...)
mimikatz 2.0

- fully recoded in C, with system’s runtimes (≠ VC9, 10...)
  - strict code (no *goto* ;))
  - smaller (~180 kb)
  - Deal relatively transparently with *memory/process/dumps*, and with *registry/hives*.

- Works on *XP/2003, Vista/2008, Seven/2008r2, 8/2012* and *8.1/2012r2*
  - x86 & x64 ;)
  - *Windows 2000 support dropped with 1.0 version*

- Two other components, **not mandatory**:
  1. *mimidrv*; a driver to interact with the Windows Kernel (hooks, tokens, process...)
  2. *mimilib*; a library with some goodies:
     - AppLocker bypass ;
     - Authentication Package (SSP) ;
     - Password filter ;
     - mimikatz::sekurlsa for *WinDBG.*
mimikatz :: sekurlsa
LSA (level \textcolor{red}{PLAYSKOOL})

Authentication Packages
\begin{itemize}
\item msv1_0
\item tspkg
\item wdigest
\item livessp
\item kerberos
\end{itemize}

Authentication
\begin{itemize}
\item msv1_0
\item kerberos
\end{itemize}

SAM

user:domain:password

Challenge Response
mimikatz :: sekurlsa
LSA (level "PLAYSKOOL")

- Authentication packages:
  - take user’s credentials;
  - do their job (hash, asking for ticket...);
  - keep enough data in memory to compute the answers to the challenges (Single Sign On).
    - Not in all case, eg: LiveSSP provider does not keep data for a SmartCard authentication

- If we can get data, and inject it in another session of LSASS, we avoid authentication part.

- If we put data in right places, we can still answer to the challenges.

- This is the principle of « Pass-the-hash »
  - In fact, of « Pass-the-* »
mimikatz :: sekurlska
demo! - sekurlska::logonpasswords
mimikatz :: sekurlsa
what is it?

This module of mimikatz read data from SamSs service (known as LSASS process) or from a memory dump!

sekurlsa module can retrieve:

- MSV1_0 hash & keys (dpapi)
- TsPkg password
- WDigest password
- LiveSSP password
- Kerberos password, ekeys, tickets & pin
- SSP password

And also:

- pass-the-hash
- overpass-the-hash / pass-the-(e)key
  - RC4 (ntlm), AES128 & AES256
- pass-the-ticket (official MSDN API !)
mimikatz :: sekurlsa workflow

for each session

search list/AVL for LUID

module!symbol

typedef struct _KIWI_struct {
    LUID LocallyUniqueIdentifier;
    [...] 
    LSA_UNICODE_STRING UserName;
    LSA_UNICODE_STRING Domaine;
    LSA_UNICODE_STRING Password;
    [...] 
} KIWI_struct, *PKIWI_struct;

Key NT 5 Symbols
RC4 lsasrv!g_randomKey
DES lsasrv!hDESKey
AES lsasrv!hAESKey

Key NT 6 Symbols
lsasrv!/InitializationVector
3DES lsasrv!h3DESKey
AES lsasrv!hAESKey

Credentials in clear!

LsaUnprotectMemory

typedef struct _LIST_ENTRY {
    union {
        LUID _LUID; 
        ULONG 0;
    } _LIST_ENTRY;
    ... 
} LIST_ENTRY, *PLIST_ENTRY;

09/07/2014 Benjamin DELPY `gentilkiwi` @ 15th RMLL/LSM
benjamin@gentilkiwi.com ; blog.gentilkiwi.com
mimikatz :: sekurlsa

* memo *

**Security Packages**

<table>
<thead>
<tr>
<th>Package</th>
<th>Symbols</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>tspkg</td>
<td>tspkg!TSGlobalCredTable</td>
<td>RTL_AVL_TABLE</td>
</tr>
<tr>
<td>wdigest</td>
<td>wdigest!l_LogSessList</td>
<td>LIST_ENTRY</td>
</tr>
<tr>
<td>livessp</td>
<td>livessp!LiveGlobalLogonSessionList</td>
<td>LIST_ENTRY</td>
</tr>
<tr>
<td>kerberos (nt5)</td>
<td>kerberos!KerbLogonSessionList</td>
<td>LIST_ENTRY</td>
</tr>
<tr>
<td>kerberos (nt6)</td>
<td>kerberos!KerbGlobalLogonSessionTable</td>
<td>RTL_AVL_TABLE</td>
</tr>
<tr>
<td>msv1_0</td>
<td>lsasrv!LogonSessionList</td>
<td>LIST_ENTRY</td>
</tr>
<tr>
<td></td>
<td>lsasrv!LogonSessionListCount</td>
<td>ULONG</td>
</tr>
<tr>
<td>ssp</td>
<td>msv1_0!SspCredentialList</td>
<td>LIST_ENTRY</td>
</tr>
</tbody>
</table>

**Protection Keys**

<table>
<thead>
<tr>
<th>Key NT 5</th>
<th>Symbols</th>
<th>Key NT 6</th>
<th>Symbols</th>
</tr>
</thead>
</table>
| RC4      | lsasrv!g_cbRandomKey
lsasrv!g_pRandomKey | 3DES                | lsasrv!h3DesKey                              |
| DESx     | lsasrv!g_pDESXKey
lsasrv!g_Feedback | AES                 | lsasrv!hAesKey                               |
|          |                                |                     | lsasrv!InitializationVector                 |
All credentials in memory are encrypted, but in a reversible way to be used (ok, not ~all~ are encrypted)

Encryption is **symmetric**, keys are in the memory of the **LSASS** process

- It’s like sending an encrypted ZIP with the password in the same email...
- Encrypt works with **LsaProtectMemory**, decrypt with **LsaUnprotectMemory**

**Both deal with LsaEncryptMemory**

*Depending on the secret size, algorithm is different:*

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lsasrv</td>
<td>Lsasrv</td>
<td>Lsasrv</td>
</tr>
<tr>
<td>LSASS</td>
<td>LSASS</td>
<td>LSASS</td>
</tr>
<tr>
<td>NT5</td>
<td>NT6</td>
<td>NT6</td>
</tr>
<tr>
<td>g_cbRandomKey</td>
<td>InitializationVector</td>
<td>hAesKey</td>
</tr>
<tr>
<td>g_pRandomKey</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g_pDESXKey</td>
<td>h3DesKey</td>
<td></td>
</tr>
<tr>
<td>g_Feedback</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

09/07/2014 Benjamin DELPY `gentilkiwi` @ 15th RMLL/LSM

benjamin@gentilkiwi.com ; blog.gentilkiwi.com
mimikatz :: sekurlsa

demo! - sekurlsa::logonpasswords
mimikatz
Focus on Windows 8.1 & 2012r2

- After a lot of customers cases, time, credentials stolen...Microsoft had to react! (a little bit, ok ;))
  
  “In Windows Server 2012 R2 and Windows 8.1, new credential protection and domain authentication controls have been added to address credential theft.”
  

- “Restricted Admin mode for Remote Desktop Connection”
  
  ✔ Avoid user credentials to be sent to the server (and stolen)
  
  ✗ Allow authentication by *pass-the-hash, pass-the-ticket* & *overpass-the-hash* with *CredSSP*

- “LSA Protection”
  
  ✔ Deny memory access to *LSASS* process (protected process)
  
  ✗ Bypassed by a driver or another protected process (remember? *mimikatz* has a driver ;))

- “Protected Users security group”
  
  ✔ No more *NTLM, WDigest, CredSSP*, no delegation nor SSO... Strengthening *Kerberos* only!
  
  ✗ Kerberos tickets can still be stolen and replayed (and smartcard/pin code is in memory =))
### mimikatz

**Focus on Windows 8.1 & 2012r2**

---

<table>
<thead>
<tr>
<th>Primary</th>
<th>Credential Keys</th>
<th>tspkg</th>
<th>wdigest</th>
<th>kerberos</th>
<th>livessp</th>
<th>ssp</th>
<th>dpapi</th>
<th>ccredman</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LM</td>
<td>NTLM</td>
<td>SHA1</td>
<td>LM</td>
<td>NTLM</td>
<td>SHA1</td>
<td>Root</td>
<td>DPAPI</td>
</tr>
<tr>
<td>Local Account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain Account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local Account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain Account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain Protected Users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

#### Windows XP/2003

- Local Account: 2
- Domain Account: 2

#### Windows Vista/2008 & 7/2008r2

- Local Account: 5
- Domain Account: 5

#### Windows 8/2012

- Local Account: 7
- Domain Account: 7

#### Windows 8.1/2012r2

- Microsoft Account: 3
- Local Account: 3
- Domain Account: 3
- Domain Protected Users: 3

---

### Windows 8.1 vault for user’s authentication

<table>
<thead>
<tr>
<th>PIN</th>
<th>Picture</th>
<th>Fingerprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>pass</td>
<td>gestures</td>
</tr>
</tbody>
</table>

- not applicable
- PIN code when SmartCard used for native Logon
- PIN code is NOT encrypted in memory (XP/2003)

---

1. can need an unlock on NTS, not available with smartcard
2. tspkg is not installed by default on XP, not available on 2003
3. tspkg is off by default (but needed for SSO with remoteapps/ts), wdigest too
5. When accessed/used by owner
6. When local admin, UAC and after unlock

---

Last version on: [http://1drv.ms/1fW4hu](http://1drv.ms/1fW4hu)
mimikatz
Focus on Windows 8.1 & 2012r2

06/12/2012 - Mitigating Pass-the-Hash-Attacks and Other Credential Theft

13/05/2014 - KB2871997 - Backport of Windows 8.1/2012r2 nice stuff to 7/2008r2 & 8/2012

Kerberos is a computer network authentication protocol which works on the basis of 'tickets' to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner.


Two kinds of ticket:
- TGT: for account in the domain;
- TGS: to access a service on a node, for one user.

Some resources more accurate than me:
mimikatz :: kerberos 1/3 authentication

Kerberos (level PLAYSKOOL)

1. **AS-REQ**
   I would like a ticket for 'Administrateur' on the domain 'chocolate'

2. **AS-REP**
   Here is a TGT ticket for 'Administrateur' on the domain 'chocolate'
   If you have its credentials (good passwords, so good keys), you can use it to ask me TGS, thanks to the session key

**Username**

<table>
<thead>
<tr>
<th>username</th>
<th>password</th>
<th>ntlm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrateur</td>
<td>waza1234/</td>
<td>cc36cf7a8514893efcccd332446158b1a</td>
</tr>
</tbody>
</table>

**KDC**

**Start/End/MaxRenew**

- krbtgt / chocolate.local
- Administrateur @ chocolate.local

**Session key + metadata**

- SID: 5-1-5-21-a-b-c
- User RID: 500 (Administrateur)
- Groups RID: 520, 512, 519, 518, 572 (Admins du domaine, entreprise, ...)
- Dernier changmt: 04/02/2014 23:21:07
- Expire: Jamais
- Modifiable: 05/02/2014 23:21:07

** krbtgt**

**username**

<table>
<thead>
<tr>
<th>username</th>
<th>password</th>
<th>ntlm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrateur</td>
<td>waza1234/</td>
<td>cc36cf7a8514893efcccd332446158b1a</td>
</tr>
</tbody>
</table>

**krbtgt**

Start/End/MaxRenew
krbtgt / chocolate.local
Administrateur @ chocolate.local
Session key + metadata

**SID**: S-1-5-21-a-b-c
**User RID**: 500 (Administrateur)
**Groups RID**: 520, 512, 519, 518, 572 (Admins du domaine, entreprise, ...)
**Dernier changmt**: 04/02/2014 23:21:07
**Expire**: Jamais
**Modifiable**: 05/02/2014 23:21:07
mimikatz :: kerberos 2/3 asking for service
Kerberos (level PLAYSKOOL)

③ TGS-REQ
I would like a ticket for the ‘cifs’ service on ‘serveur’ of ‘chocolate’ domain. Here is my TGT and some information encrypted with session key. I know it, because I’m really ‘Administrateur’.

④ TGS-REP
Here is a TGS for ‘cifs/serveur’ on the ‘chocolate’ domain. If you know initial session key, you can decrypt TGS session key and use it for communicate with ‘serveur’.

<table>
<thead>
<tr>
<th>rid</th>
<th>username</th>
<th>ntlm</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Administrateur</td>
<td>cc36cf7a8514893efccd332446158b1a</td>
</tr>
<tr>
<td>502</td>
<td>krbtgt</td>
<td>310b643c5316c8c3c70a10cfb17e2e31</td>
</tr>
<tr>
<td>1106</td>
<td>Equipement</td>
<td>57a087d98bfac9df10df27a564b77ad6</td>
</tr>
<tr>
<td>1107</td>
<td>Utilisateur</td>
<td>8e3a18d453ec2450c321003772d678d5</td>
</tr>
<tr>
<td>1108</td>
<td>serveur$</td>
<td>77d4b1409b7e5b97263b0f0230f73041</td>
</tr>
</tbody>
</table>
Hello ‘serveur’, here is a TGS for you. It show that the KDC knows me as ‘Administrateur’ on the ‘chocolate’ domain for using your ‘cifs’ service. All that with all the benefits that the KDC me recognize me (groups, privileges, time...) You can check this ticket because you know the secret key of this ticket (it’s your secret), so you check session key of the request.
mimikatz :: kerberos
pass-the-ticket

**TGS theft** – *access to a service on a server for 10h (can vary)*

**TGT theft** – *full identity of one user for 10h (can vary)*

---

09/07/2014  Benjamin DELPY `gentilkiwi` @ 15th RMLL/LSM  benjamin@gentilkiwi.com ; blog.gentilkiwi.com
eKey theft – full identity of one user for password lifetime on the domain

AS-REQ
I would like a ticket for ‘Administrateur’ on the domain ‘chocolate’

AS-REP
Here is a TGT ticket for ‘Administrateur’ on the domain ‘chocolate’
wait? I can obtain a Kerberos ticket with a NTLM hash? Like in “pass-the-hash”?

- Only a hash?
- Yeah, you can =)

So what is that?

- Preauth & first data are encrypted with user key, but what is that key?
- For RC4, the key is the NTLM hash!

Domain: CHOCOLATE / S-1-5-21-130452501-2365100805-3685010670

RID: 00001f4 (500)
User: Administrateur

* Primary
  LM: :
  NTLM: cc36cf7a8514893efccd332446158b1a

* Kerberos
  Default Salt: CHOCOLATE.LOCALAdministrateur
  Credentials
    des_cbc_md5: f8fd987fa7153185

* Kerberos-Newer-Keys
  Default Salt: CHOCOLATE.LOCALAdministrateur
  Default Iterations: 4096
  Credentials
    aes256_hmac (4096): b7268361386090314acce8d9367e55f55865e7ef8e670fbe4262d6c94098a9e9
    aes128_hmac (4096): 8451bb37aa6d7ce3d2a5c2d24d317af3
    des_cbc_md5 (4096): f8fd987fa7153185
mimikatz :: kerberos
demo ! - sekurlsa::tickets

mimikatz 2.0 alpha x86
.
#
## #
## 
## #
##
#
##
## #
##
#
## #
## 
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
## #
##
#
Golden Ticket

- **TGT** are limited to 10 hours and can be renewed — configurable time

- **TGT** are nothing more than **TGS** for a service named ‘**krbtgt**’ for all **KDC** in a domain

For that, they’re encrypted with a common key for each KDC. With **RC4**, the **NTLM** hash of the fictive account ‘**krbtgt**’ (or AES)

I don’t really know why, but this key is “never” renewed (only when migrating to >= 2008 functional level domain)

- However, using the passwords history (2) of this account, a full renew can be done in two moves.

What could we do with a permanent key, which allow creating TGT?

<table>
<thead>
<tr>
<th>rid</th>
<th>username</th>
<th>type</th>
<th>key</th>
</tr>
</thead>
<tbody>
<tr>
<td>502</td>
<td>krbtgt</td>
<td>rc4</td>
<td>310b643c5316c8c3c70a10cfb17e2e31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>aes128</td>
<td>Da3128afc899a298b72d365bd753dbfb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>aes256</td>
<td>15540cac73e94028231ef86631bc47bd5c827847ade468d6f6f739eb00c68e42</td>
</tr>
</tbody>
</table>
mimikatz :: kerberos
Golden Ticket – TGT Create (extract)

- Client name: Administrateur
- Service name: krbtgt/chocolate.local
- Validity
  - Start Time: 09/07/2014 10:25:00
  - End Time: 09/07/2024 10:25:00
- Authorization data Microsoft (PAC)
  - Username: Administrateur
  - Domain SID
    - S-1-5-21-130452501-2365100805-3685010670
  - User ID
    - 500 Administrateur
  - Groups ID
    - 512 Admins du domaine
    - 519 Administrateurs de l'entreprise
    - 518 Administrateurs du schéma
    - ...
  - ...

rid | username     | ntlm
---|--------------|----------------
502| krbtgt      | 310b643c5316c8c3c70a10cfb17e2e31
mimikatz :: kerberos
demo! - kerberos::golden

09/07/2014 Benjamin DELPY `gentilkiwi` @ 15th RMLL/LSM
benjamin@gentilkiwi.com ; blog.gentilkiwi.com
mimikatz :: sekurlsa
What we can do?

Basics
- No physical access to computer / servers
  - Volume/disk encryption
- No admin rights! (even for VIP) – no Debug privilege!
- Disable local admin accounts
  - **Strong passwords** *(haha, it was a joke, so useless 😊)*
- For privileged account, network login instead of interactive (when possible)
- Audit ; pass the **hash** keeps traces and can lock accounts
- Use separated network (or forest) for privileged tasks

More in depth
- Force strong authentication (SmartCard & Token) : $ / €
- Short validity for Kerberos tickets
- No delegation
- Disable LM & NTLM (force Kerberos)
- No exotic biometric!
- Let opportunities to stop retro compatibility

To study
- **TPM** on Windows 8.1
  - Virtual SmartCard seems promising
- Verify TPM CSP/KSP of specific provider (Lenovo, Dell,...)
  - Remember biometric? ;)

Use HSM / Kerberos Box for crypto operations
mimikatz
what else?

- Retrieve system/users secrets (like saved passwords)
- Export keys/certificates, even those that are not exportable (software CAPI & CNG)
- Stop event monitoring...
- Bypass Applocker / SRP
- Manipulate some Handles
- Patch Terminal Server
- Basic GPO bypass
- Driver
  - Play with Tokens & Privileges
  - Display SSDT x86 & x64
  - List MiniFilters
  - List Notifications (process/thread/image/registry)
  - List hooks et and procedures of Objects
Thanks’ to / Merci à :

– RMLL / LSM & partners ;
  • Especially Christian for his invitation!
– Microsoft to change some behaviors! 😊 ;
– Community for ideas (∞) ;
– Folks, friends supporting me every day (oe.eo) ;
– You, for your attention and your nice messages!

Questions, remarks?

➔ Please! Don’t be shy!
Blog, Source Code & Contact

- Blog: [blog.gentilkiwi.com](http://blog.gentilkiwi.com)
- mimikatz: [http://blog.gentilkiwi.com/mimikatz](http://blog.gentilkiwi.com/mimikatz)
- Source: [https://github.com/gentilkiwi/mimikatz](https://github.com/gentilkiwi/mimikatz)
- Contact: benjamin@gentilkiwi.com