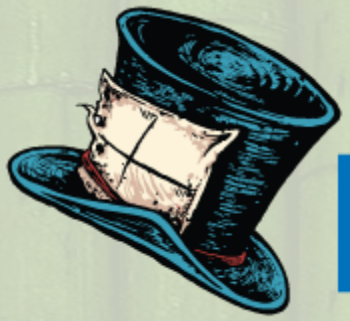
From mimikatz to kekeo

passing by new Microsoft Security Technologies



BlueHat IL

Benjamin DELPY `***gentilkiwi***`

`whoami` ?

- **Benjamin DELPY - @gentilkiwi**

  - –Security researcher at night (*it's not my work*)
  - –Author of `mimikatz`
    - *This little program that I wrote to learn C*
    - And **kekeo**, for my (your ?) personal usage ;)
  - –I'm not:
    - **Bachelor, CISSP, CISA, OSCP, CHFI, CEH, ISO*, MCSA, CHFI, PASSI, […]**
  - –Memes count: **>15** – "Cyber*": **0**

`who are we`?

- **When I come to Microsoft, I'm not really alone**
  - A lots of people in the infosec community are behind me and inspire me
  - Especially these 2 crazy guys who made the foundation of SSO research
    - **Alva `Skip` Duckwall (@passingthehash)**
    - **Chris Campbell (@obscuresec)**
  - Yes, t-shirts are about Mark Russinovich ;)

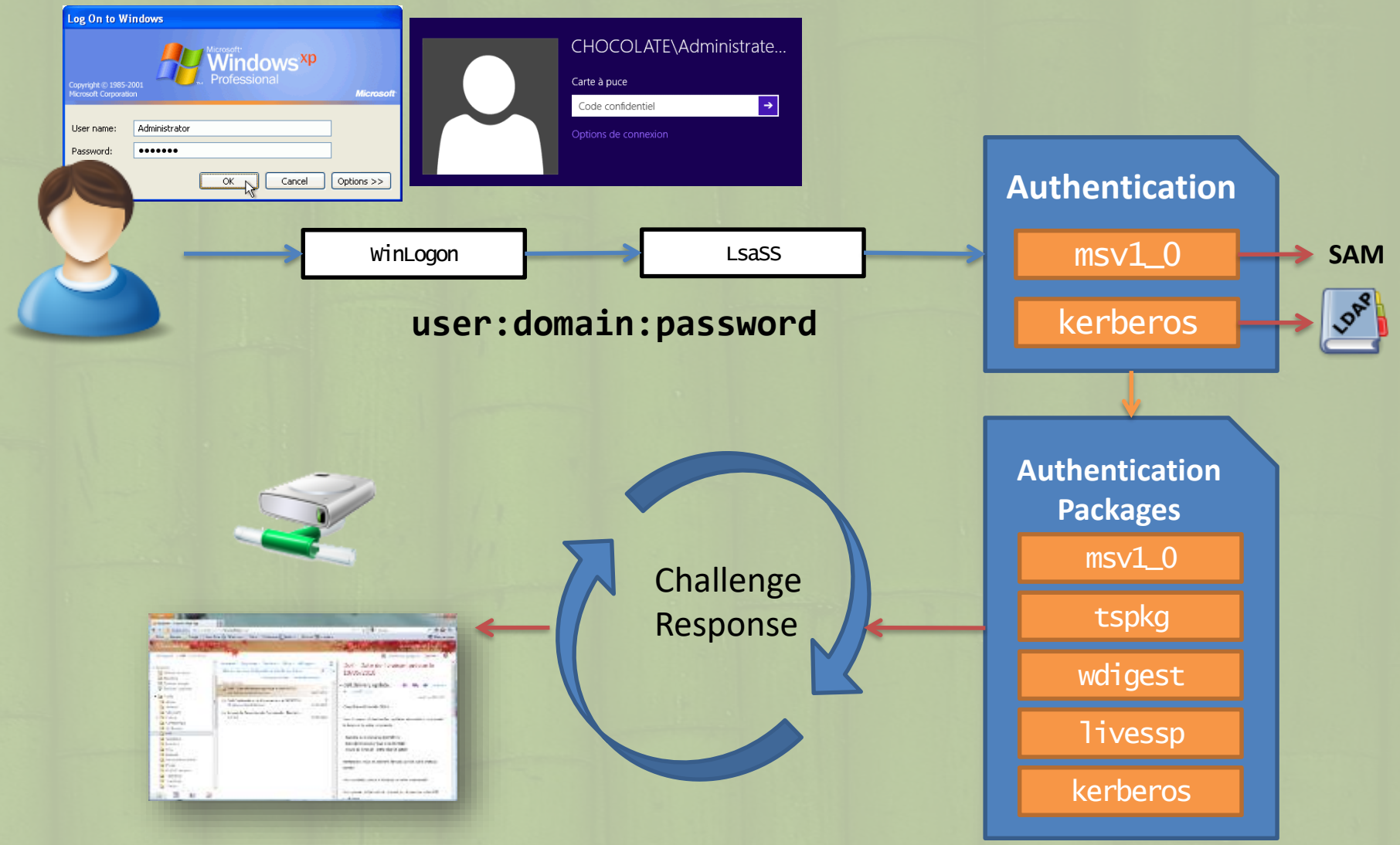
Ho, by the way…

- Remember #askpth ?

- Initially, I won…, but s
had to think that Frer
pass-the-hash to the xBox….

mimikatz :: sekurlsa

LSA (PLAYSKOOL level)



Little retrospective

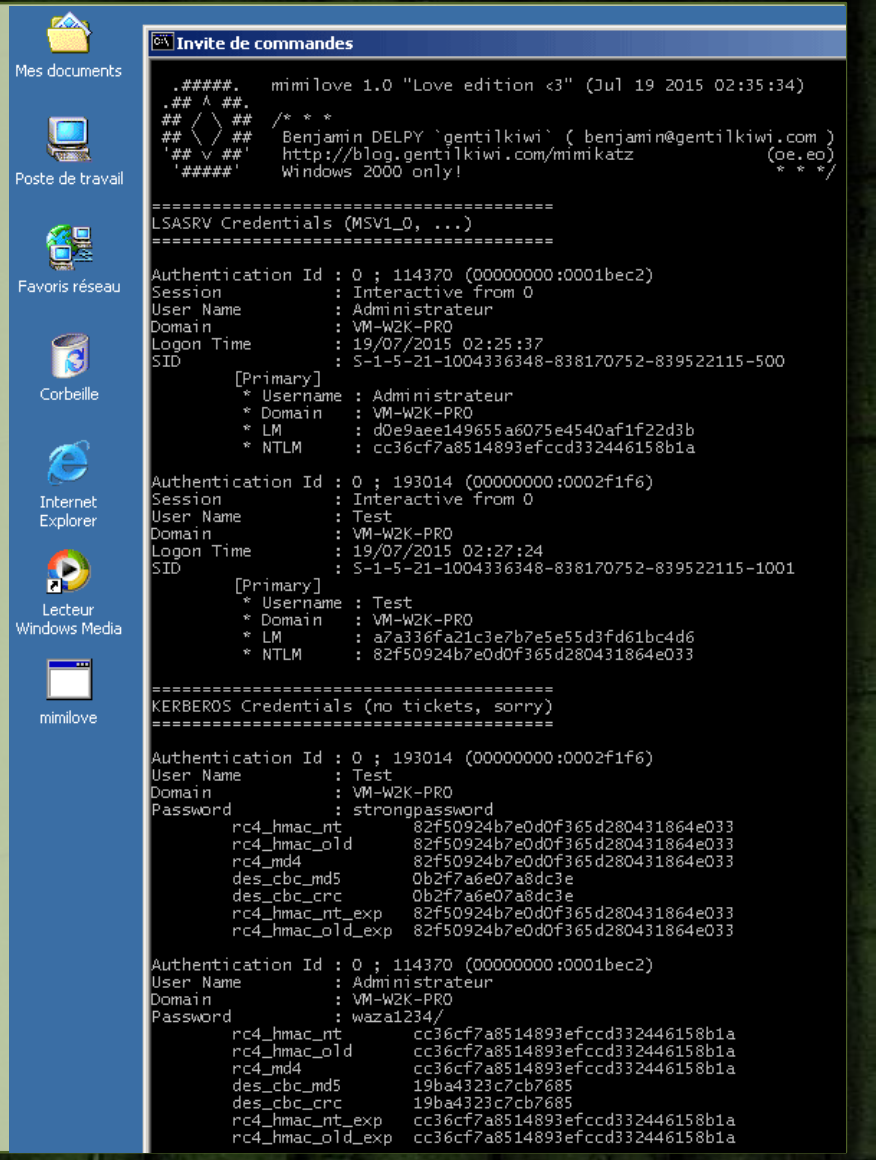
Windows 2000

- Yes, it still exist on some AD environments
  - Who said ‘even DC’ ? ☺
- Credentials can be in multiple places, but in LSASS we can easily find:
  - LM & NTLM hashes (no encryption at all) ;
  - Kerberos:
    - keys (no encryption at all) ;
    - Not tested: tickets and session key, but it seems logical ;
    - Passwords (if not already consumed) ;
- Passwords are “encrypted” in memory

```
hash = *((PBYTE) &session.Password.Length + 1);
*((PBYTE) &session.Password.Length + 1) = 0;
RtlRunDecodeUnicodeString(hash, &session.Password);
```

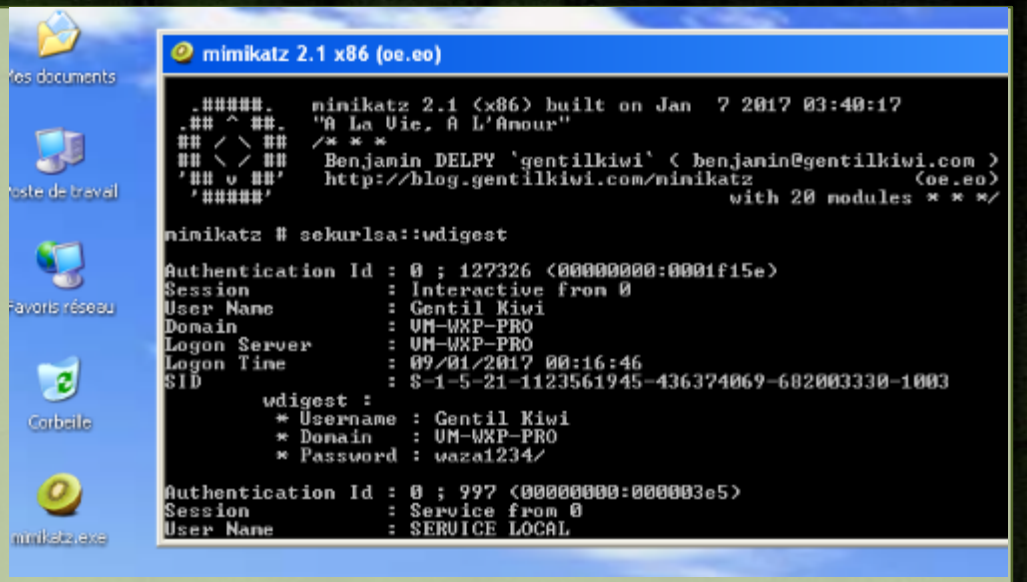
basically, a XOR with a 1 byte ‘key’ !

- *And this “key” is stored in the secret structure*



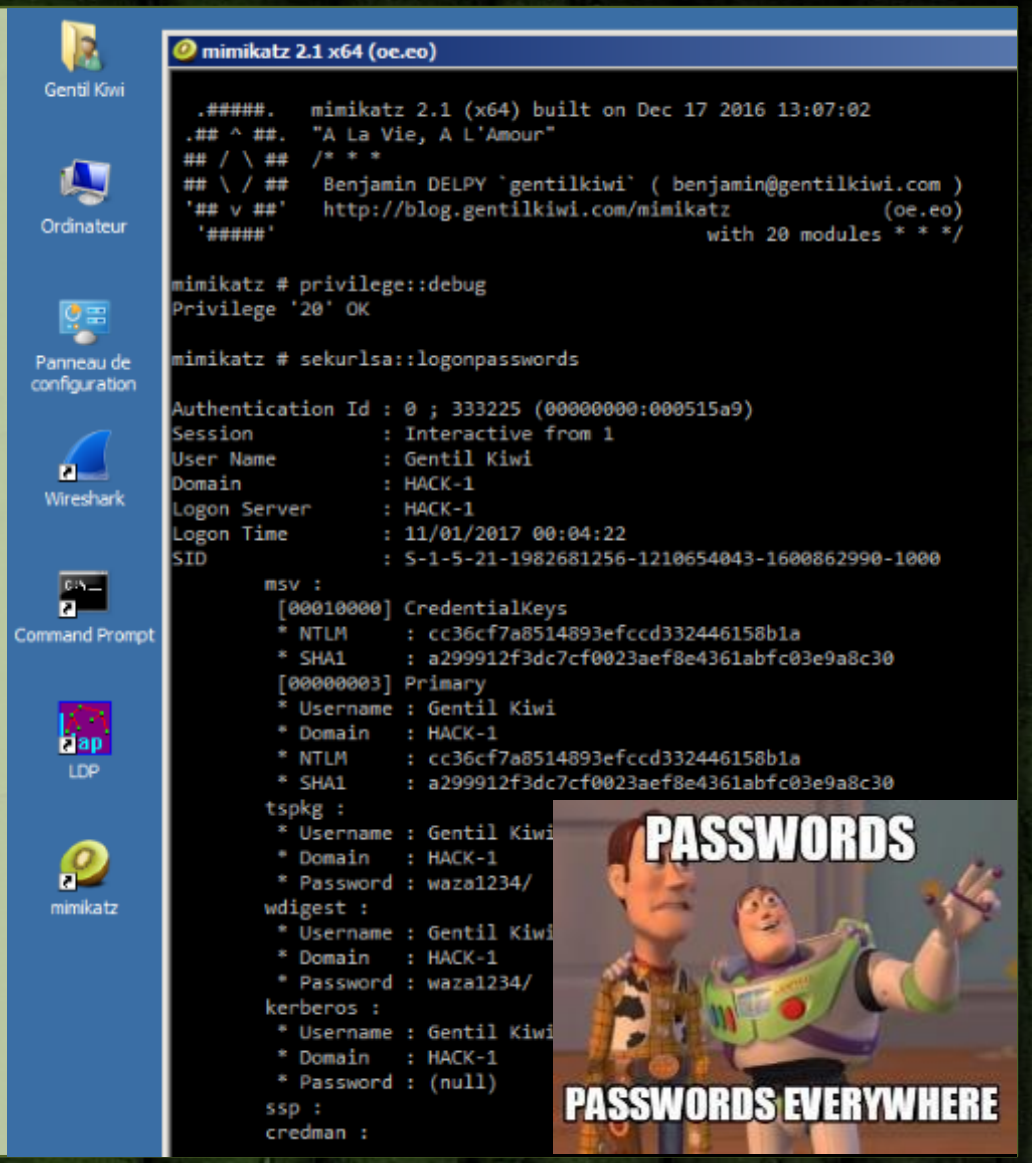
Windows XP/2003

- This time, we have a gift: **WDigest** provider
  - *Authentication to Web Site, SASL, LDAP scenarios…*
  - https://technet.microsoft.com/library/cc778868.aspx
  - https://www.ietf.org/rfc/rfc2617.txt
- We now have constantly passwords in memory
  - Someone at Microsoft had to think it was dangerous
- LSA SSO secrets are now protected by **LsaEncryptMemory**
  - and unprotected by **LsaUnprotectMemory**
  - Crypto: **RC4** or **DESx** algorithms (depending on the secret size)
- *Keys and 'IV' are stored near the secret (in LSASS process)*
- *First Windows version supported by mimikatz to dump Kerberos tickets and associated session key*
- In "bonus", **TsPkg** provider can be added manually in Windows XP (but let's go on)



Windows Vista/7

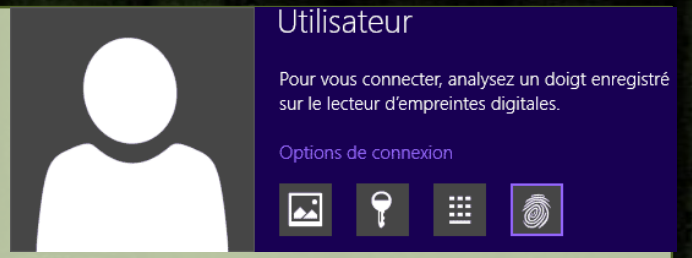
- Other new gifts
  - Older still here, of course ;)
  - ***TsPkg*** *(CredSSP support)*
    - *Credential Delegation for Terminal Server/PowerShell/Double hop, etc…*
  - ***LiveSSP*** *(ok, not really a gift, but it's here too)*
- We now have constantly <u>several passwords</u> in memory
  - Someone at Microsoft had to think it was more and more dangerous
- LSA SSO secrets are still protected by **`LsaEncryptMemory`**
  - and of course unprotected by **`LsaUnprotectMemory`**
  - Crypto: **3DES** or **AES** algorithms (depending on the secret size)
- *Keys and IV are stored near the secret (in LSASS process)*



Windows 8/8.1

“Sometimes it is necessary to hit the rock bottom to better bounce”

- Still no change… a lots of passwords, keys, tickets



- Some new crazy stuff at logon:
  - PIN Code
  - Picture
    - guys, really?
  - Fingerprints

Domain credentials in CLEAR in system vault !

Offline access is possible

« lol »

- Default “(over)pass-the-hash” or “pass-the-ticket” to Remote Desktop

Windows 8/8.1

“Sometimes it is necessary to hit the rock bottom to better bounce”

Windows 8

- Windows 8.1 introduced a new “clean base”
  - **No password by default in memory**
    - WDigest is now “off” by default ☺
  - **LSA logon session cache cleaner**
    - it’s more difficult to get credentials after logoff
  - **“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 =))
- KB 2975625
    - **Restricted Admin is now disabled by default ;)**
- Bad stuff still here for compatibility ☹

Windows 8/8.1

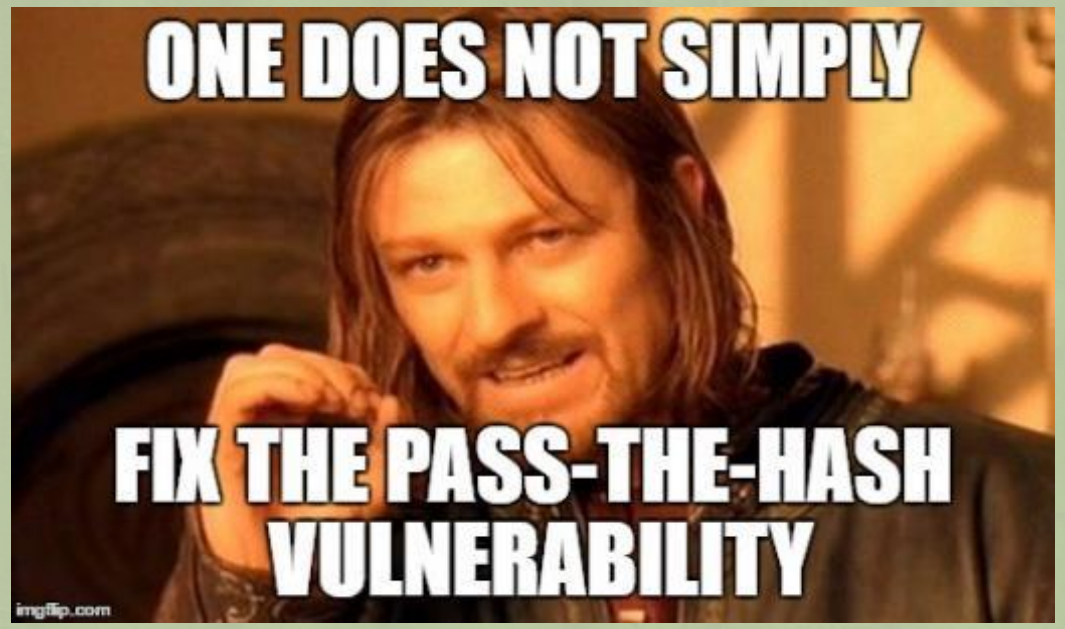
“Sometimes it is necessary to hit the rock bottom to better bounce”

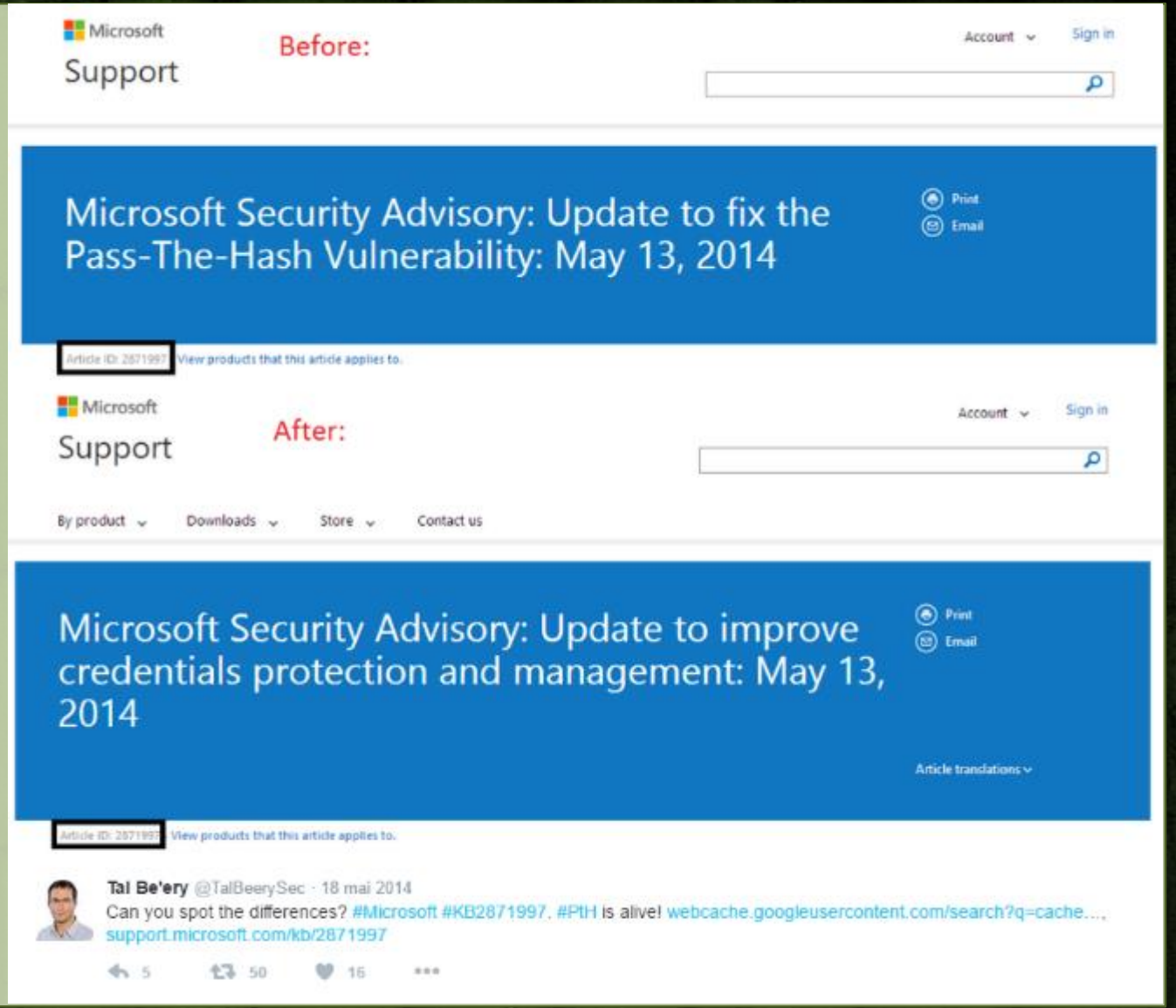
- Created KB 2871997 from ~* previous measures
  - Backported to Windows 7

Windows 8/8.1

“Sometimes it is necessary to hit the rock bottom to better bounce”

- But management and marketing was here
  - *Management is always here… (Tal too)*

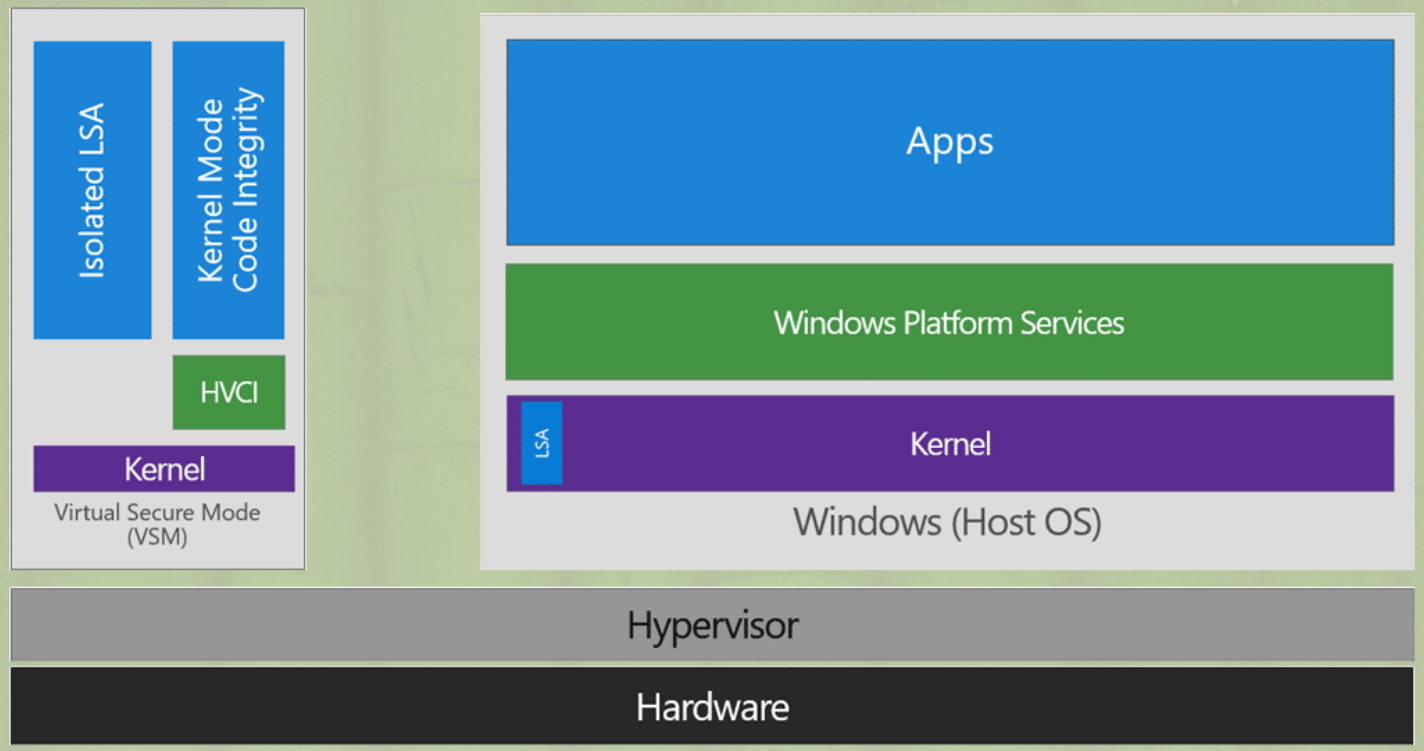




Windows 10

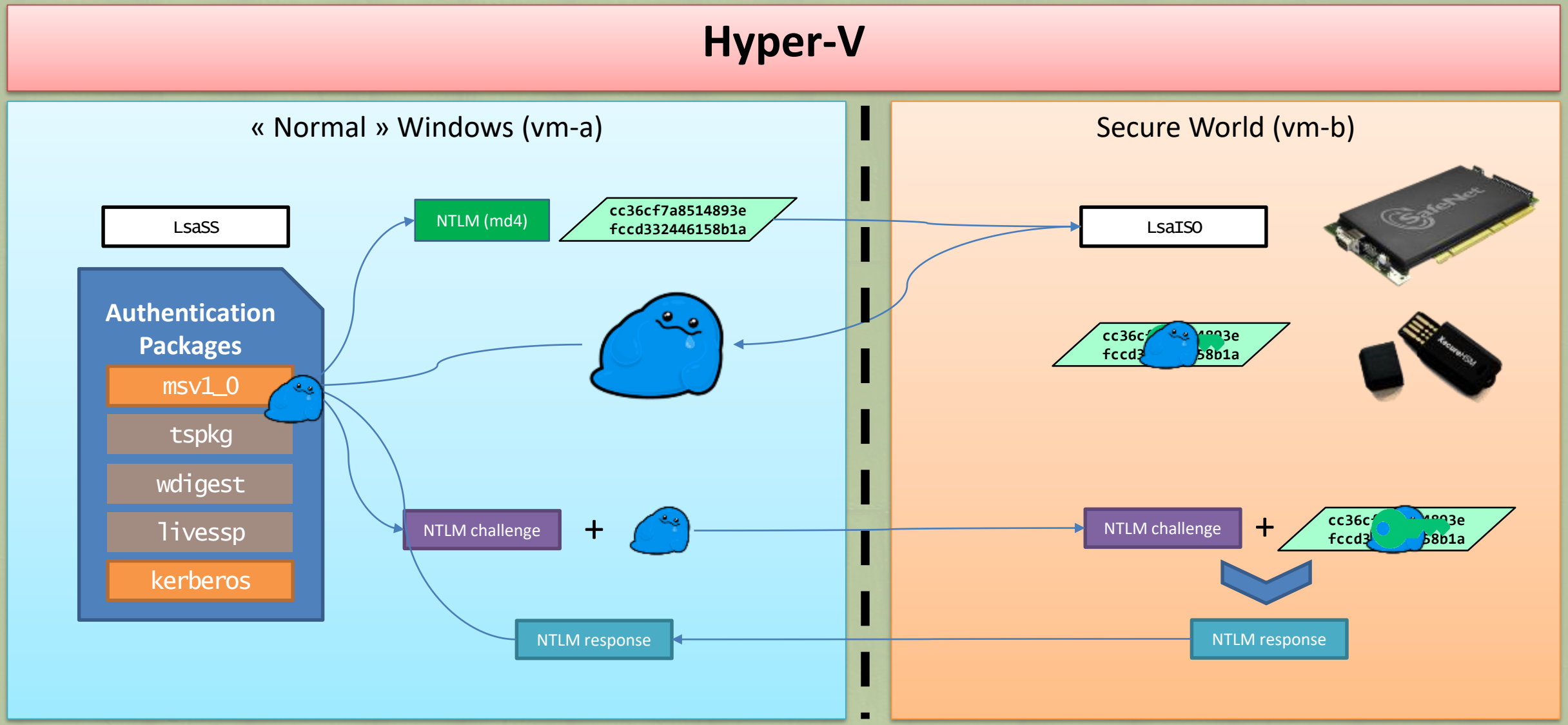
"A new hope"

- We keep all the good things and we will add a secret ingredient: VSM
  - *But only for the rich, for the others: telemetry, Cortana…*
- A very good move *approaching* Crypto HSM architecture !



https://blogs.technet.microsoft.com/ash/2016/03/02/windows-10-device-guard-and-credential-guard-demystified/

Windows 10 - Credential Guard
NTLM - *without B.S.*



Windows 10 - Credential Guard
Kerberos - *without B.S.*

Hyper-V

« Normal » Windows (vm-a)

LsaSS

Authentication Packages

- msv1_0
- tspkg
- wdigest
- livessp
- kerberos

Authentication Key
(password derived)

rc4_hmac_nt (NTLM/md4)	cc36cf7a8514893e fccd332446158b1a
aes128_hmac	8451bb37aa6d7ce3 d2a5c2d24d317af3
aes256_hmac	1a7ddce7264573ae 1f498ff41614cc78 001cbf6e3142857c ce2566ce74a7f25b

TGT Session Key

TGT

Secure World (vm-b)

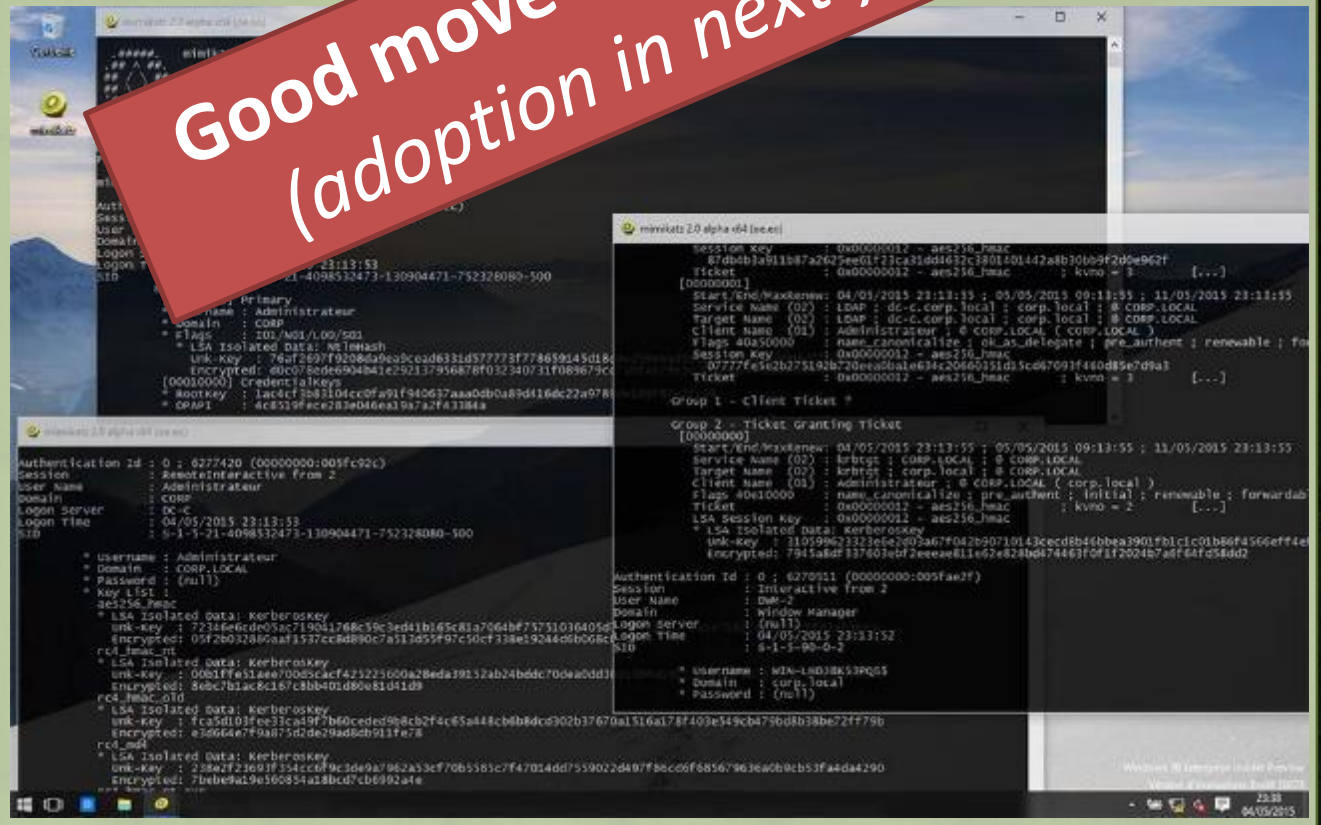
LsaISO

Windows 10 - Credential Guard
without B.S.

- **Prerequisites** (at this time)
  - Windows 10 Enterprise
  - ~~Domain (works only with domain accounts)~~
  - SecureBoot
  - Virtualization
    - Don’t start VMWare in the « guest » Windows OS after ;)
  - ~~*TPM 2*~~

- **Limitations**
  - ~~No local accounts~~
  - No Kerberos TGS session keys (why ?)
  - Not really available in Virtual Machine (for now)
  - Not enable by default
  - Rely on Hyper-V and Hardware protection *(Intel © love to flash bad firmware ;))*

Good move from Microsoft !
(adoption in next years?)



- **A lot of potential**
  - Remote Credential Guard (cool ☺)
    - Kerberos
    - NTLM ?
  - SmartCard PIN code ?
  - DPAPI ? (no, not the NG)
    - Private Key (even legacy ones)
    - Domain Passwords ?
  - SAM ?
    - Yep, even DSRM,
  - Domain controller sensitive operations ?
    - Kerberos PAC Signature ?

What now ?

- Previous credentials stealing techniques can't work anymore in the same way
  - I'm lazy (*attackers too*)
- There are several ways to obtain information or gain privileges by normal protocols/operations and logical approach
  - **[MS-DRSR]** - Directory Replication Service
  - **[MS-NRPC]** - Netlogon Remote Protocol
  - **[MS-BKRP]** - BackupKey Remote Protocol
  - **[MS-LSAD]** - Local Security Authority (Domain Policy) Remote Protocol
  - **[MS-PAC]** - Privilege Attribute Certificate Data Structure
  - Kerberos
    - **[MS-KILE]** - Kerberos Protocol Extensions
    - **[MS-SFU]** - Kerberos Protocol Extensions: Service for User and Constrained Delegation Protocol
  - …
- By example: Duqu 2.0 @ Kaspersky relied on **a very logical attack** on Microsoft PAC Signature in Kerberos tickets (MS14-068 – user to Domain Admin/Enterprise Admin)
  - https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyberespionage_actor_returns.pdf

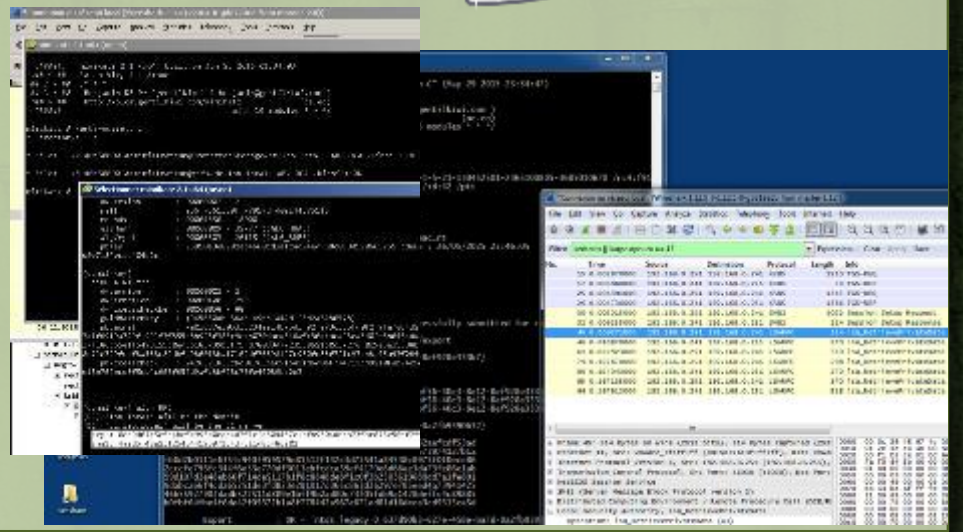
A lots of sexy names!

Various methods

- Golden tickets
  - Access to all the domain with 1 secret ☺
  - *Who said « forest »?*

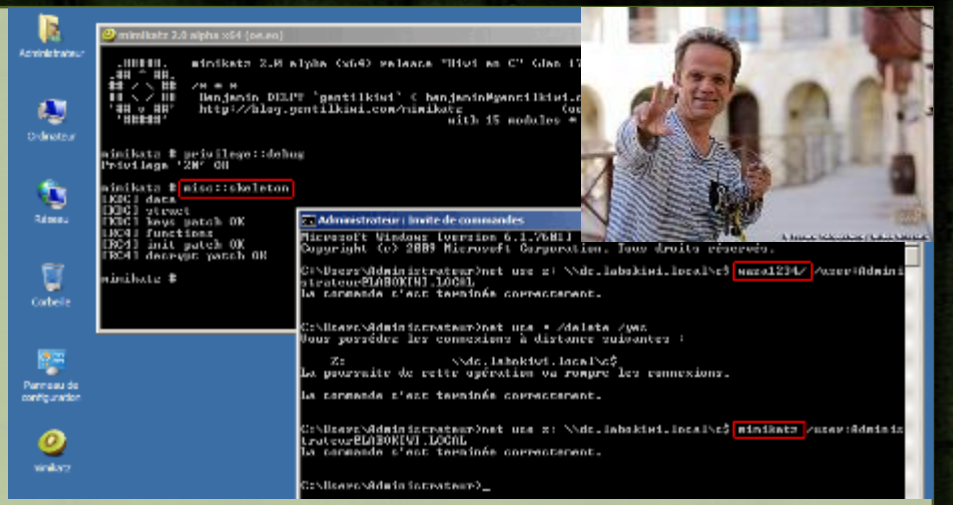
- Silver tickets
  - Access to a service (usualy a server/computer)
  - Silver is the new ~~black~~ pass-the-hash

- DPAPI Backup key:
  - Offline: by retrieving the "***~never changed master DPAPI key***"
    - Access to secrets of all users of the domains
  - Online: accessing our own secrets without knowing our password
    - Domain Passwords too
    - Useful for malware ☺



Various methods

- Skeleton Key: why steel passwords when you can use your own everywhere?
  - https://www.secureworks.com/research/skeleton-key-malware-analysis



- Play with your SID !
  - With sidHistory : be “Domain admins” without being “Domain admins”
  - Transform yourself as a domain controller to avoid logs ☺

mimikatz :: DCSync

- **DCSync** was originally a standalone program co-developed with **Vincent LE TOUX** (http://www.mysmartlogon.com)
- It was followed by
  - Impacket, **Alberto Solino** (https://github.com/CoreSecurity/impacket)
  - DSInternals, **Michael Grafnetter** (https://www.dsinternals.com/en/)
    - **These guys rock !**
- **DCSync** is now fully integrated in **mimikatz** (*lsadump::dcsync*) to take benefit of all its functions

https://twitter.com/subTee/status/637469004987129856

mimikatz :: DCSync

- **DCSync** is a new way to get Credential from the Active Directory…
  - Without file on disk or memory
  - Without shellcode in memory
  - Without process dumping
  - Without filebackup/shadow copy/physical access.

- Like others, it needs some specific rights
  - Administrator
  - Domain controller

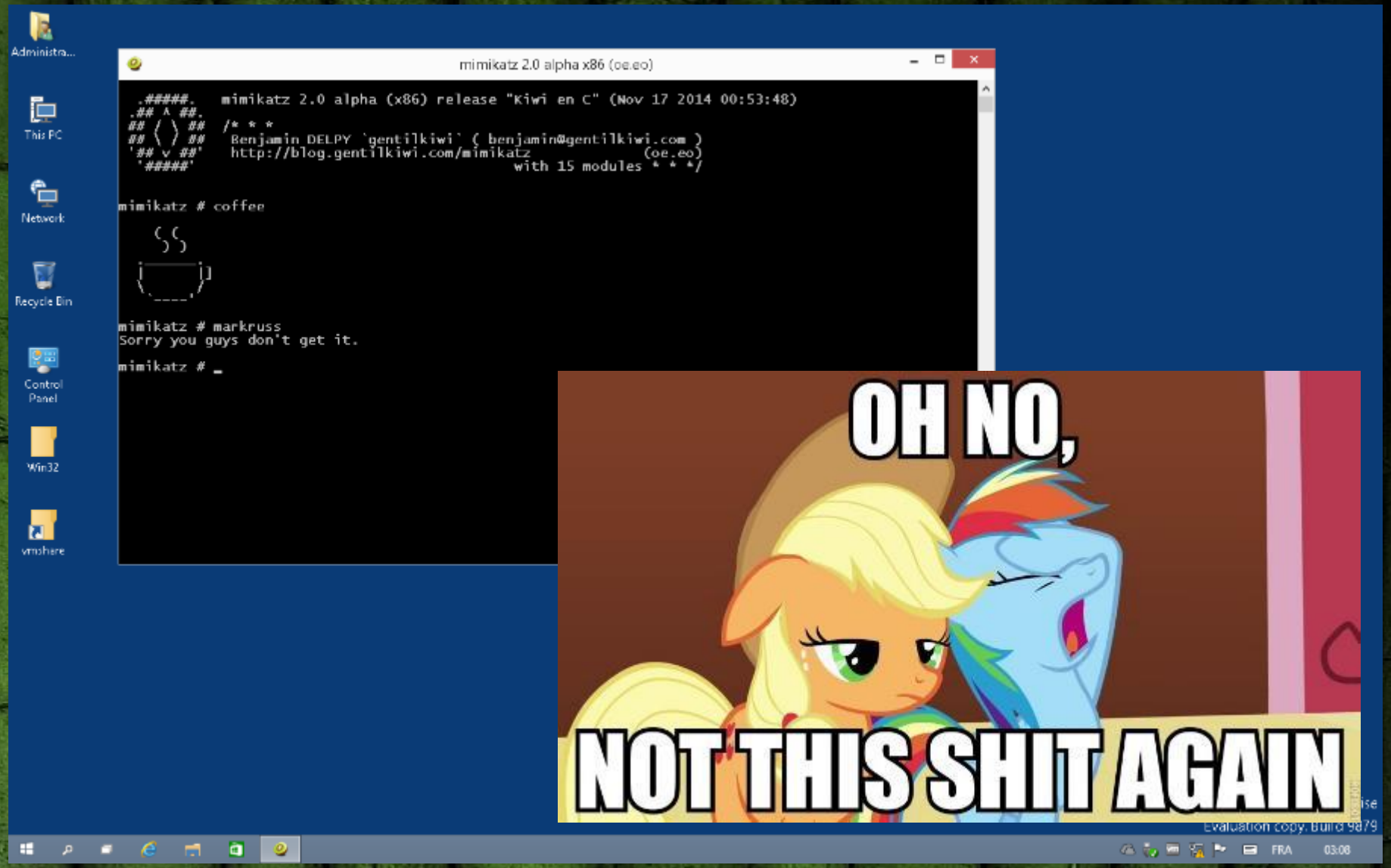
- **All is made remotely, by official RPC**
  - **default: no LOG when using a DC account !**

mimikatz :: DCSync

- **DCSync** will “mimic” API calls used in DC synchronization (**MS-DRSR**).
  - DCs exchange objects modification. One modification can be, of course, sensitive data

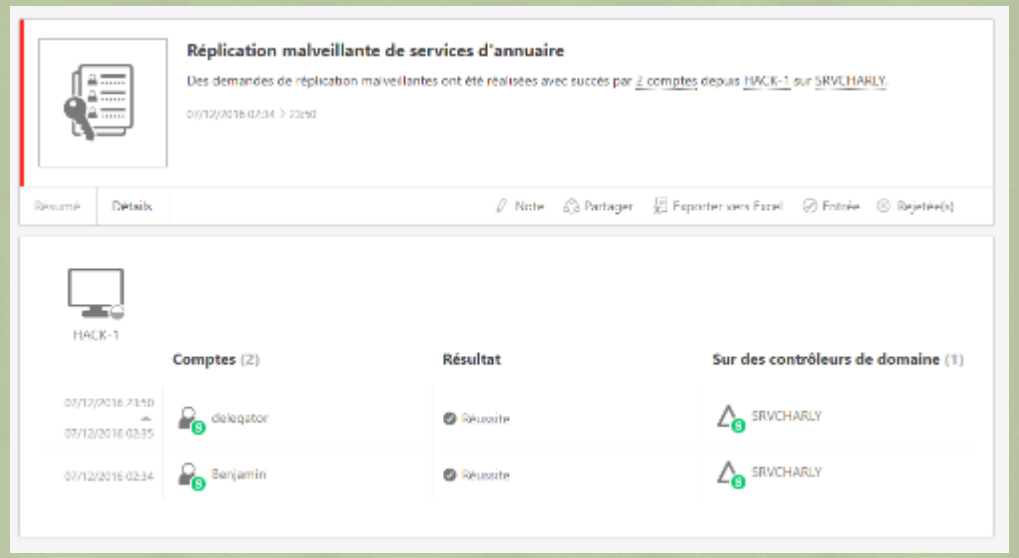
DRSBind (standard)	• To get a handle to the DC as a normal user (it’s enough here)
DRSDomainControllerInfo	• To get NtdsDsaObjectGuid of one DC in the domain
DRSCrackNames	• To translate the user/object name in a GUID
DRSUnbind (standard)	• To release the standard handle (we don’t need it anymore)
DRSBind (DC)	• To get a handle to the DC as another DC (using NtdsDsaObjectGuid)
DRSGetNCChanges	• To obtain all change of the targeted object (using its GUID)
DRSUnbind (DC)	• To release the DC handle (we don’t need it anymore)

Demo !

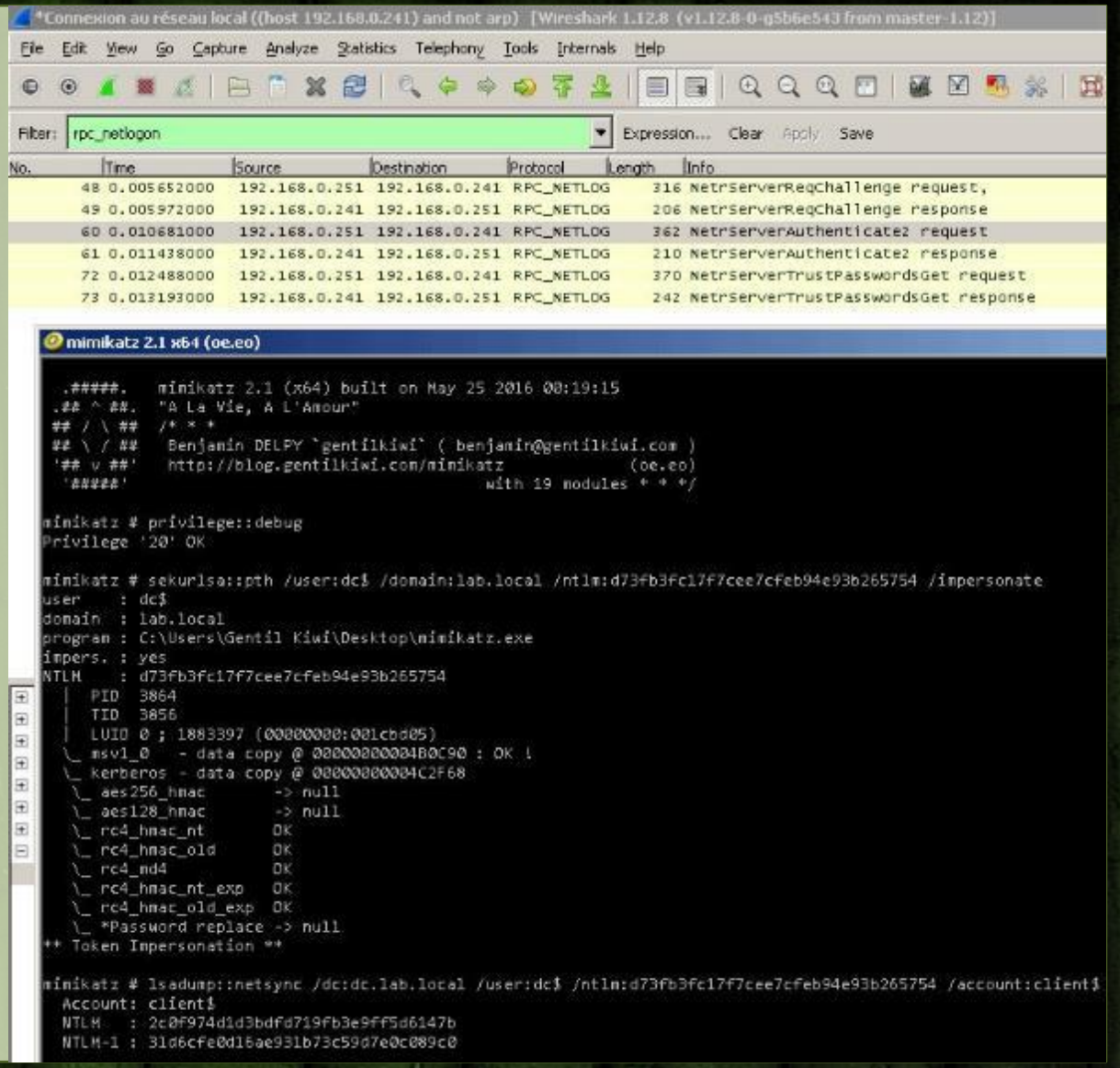

mimikatz 2.0 alpha x86 (oe.eo)
.#####. mimikatz 2.0 alpha (x86) release "Kiwi en C" (Nov 17 2014 00:53:48)
.## ^ ##.
/ \ ## /* * *
\ / ## Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'#####' with 15 modules * * */
mimikatz # coffee
mimikatz # markruss
Sorry you guys don't get it.
mimikatz # _
OH NO,
NOT THIS SHIT AGAIN

mimikatz :: netsync

- **[MS-DRSR]** DCSync is easy to spot:



- **[MS-NRPC]** What about NetLogon ? ☺
  - ***Now*** limited to 'not user accounts', but:
    - Silver tickets are so cool
    - Account type can be modified, even temporary



kekeo

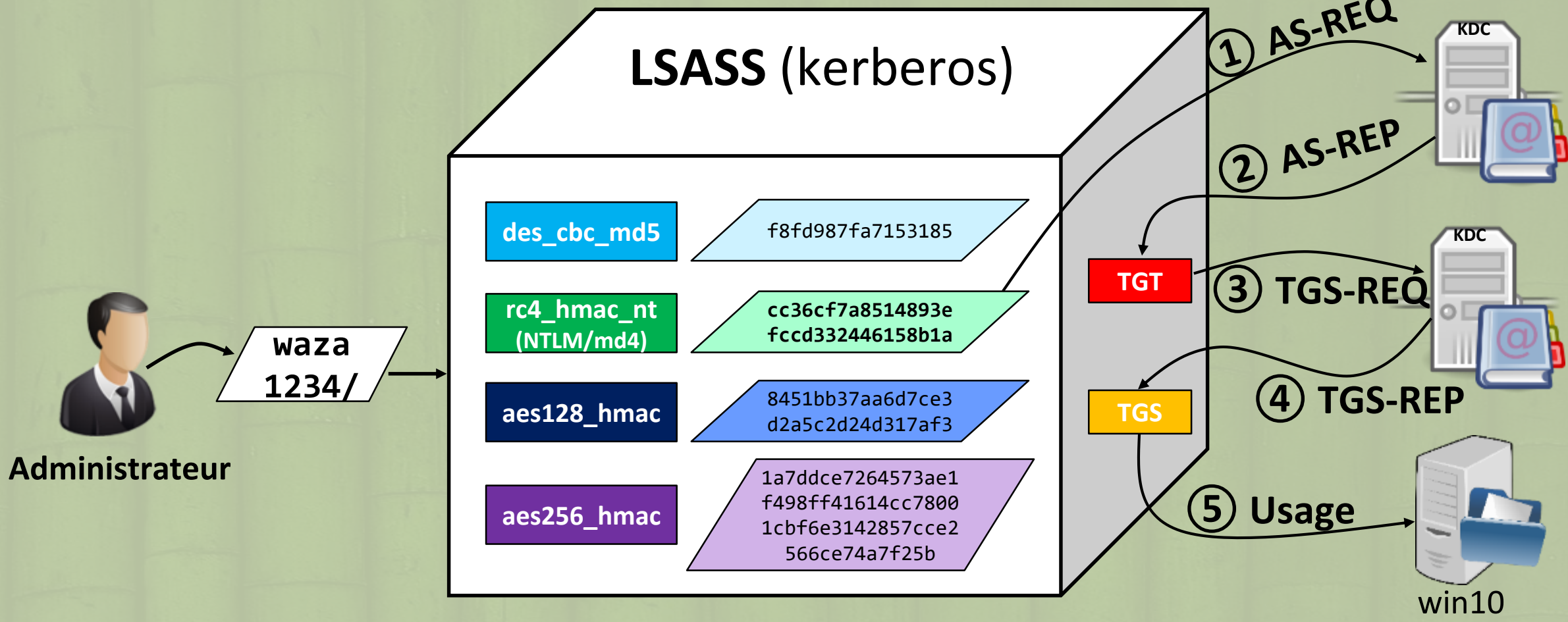
- **kekeo** is a Kerberos tools suite ☺

- I've made it external to **mimikatz** for two reasons
  - I hate to code network related stuff ;
  - It uses an external commercial ASN.1 library inside
    - It was hard for me to make it works with others…but **OSS/Nokalva** teams was kind enough to offer me a 'limited' license ☺
    - It seems quite close to some Microsoft code ;)

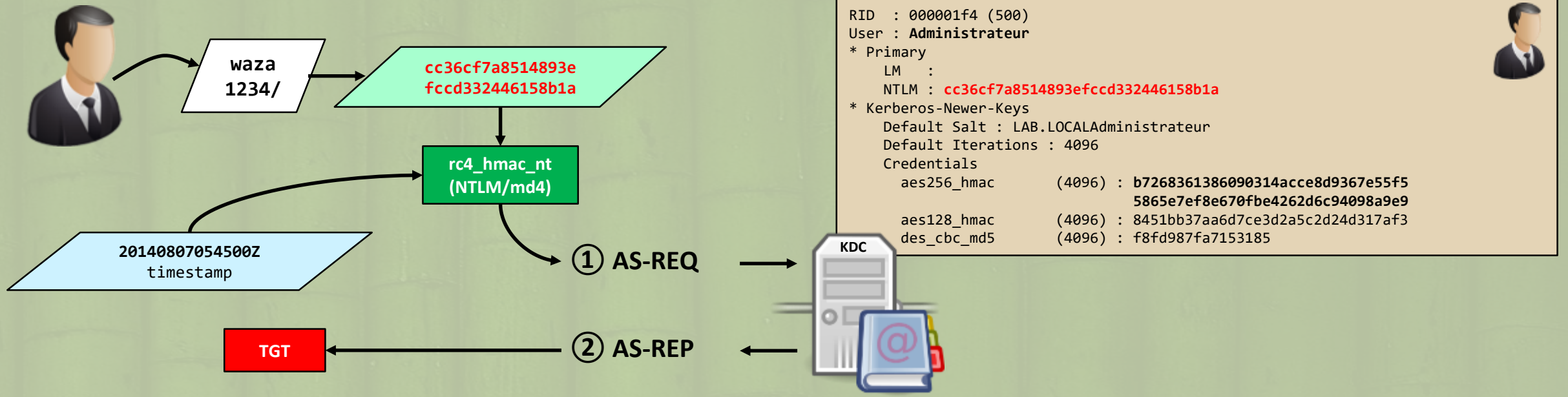
- It includes
  - **ms14068** to exploit MS14-068 (authenticated user to DA/EA/DC$ avoiding KDC signature check)
  - **ms11013** to exploit MS11-013 (authenticated user to DA/EA/DC$ avoiding * signature check)
  - **asktgt** to authenticate against a KDC with various methods (*including* **pkinitmustiness** *stuff*), and get a TGT
  - **asktgs** to request service tickets from a TGT (or renew)
  - **aoratopw** to change a password for a user from its keberos key (not reseting it)
  - **kirbikator** to convert tickets between kirbi/wce/ccache/LSA format
  - **pkinitmustiness** ☺
  - **s4u** to play with delegation (S4U2Self & S4U2Proxy)
  - **kerbstorm** /

A little reminder about Kerberos authentication

- How does it works ?



A little reminder about Kerberos authentication



- The **KDC** will validate the authentication if it can decrypt the timestamp with the long-term user key (for **RC4**, the **NTLM** hash of the user password)
- It issues a **TGT** representing the user in the domain, for a specified period

A little reminder about Kerberos authentication

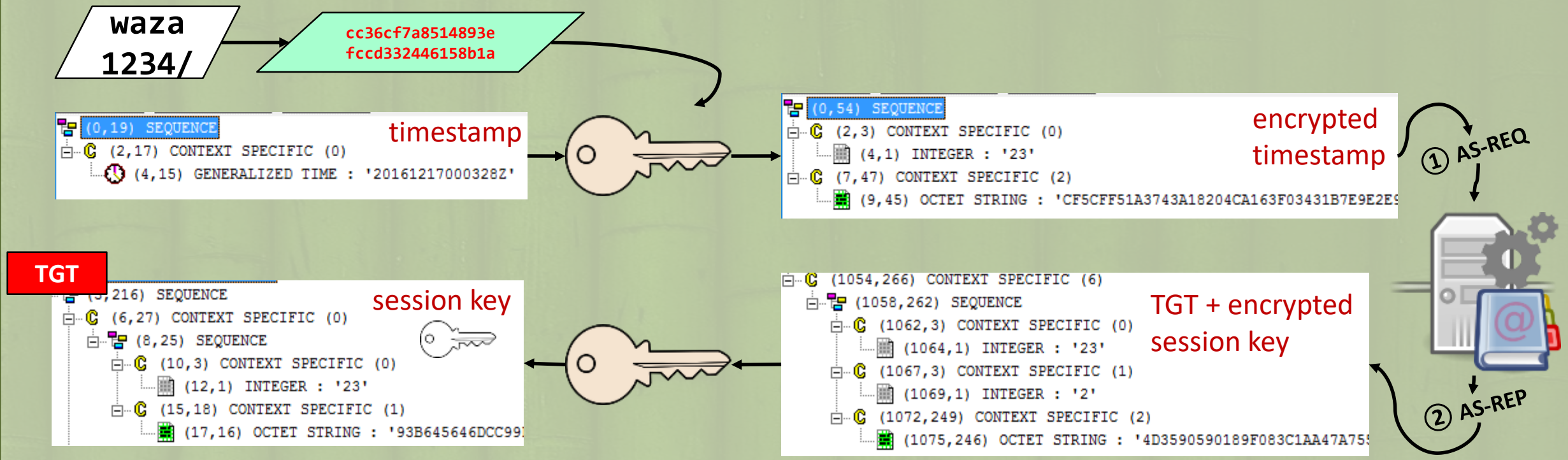
- But what about Smartcard, Token, …

Kerberos authentication

Kerberos authentication

Password mode

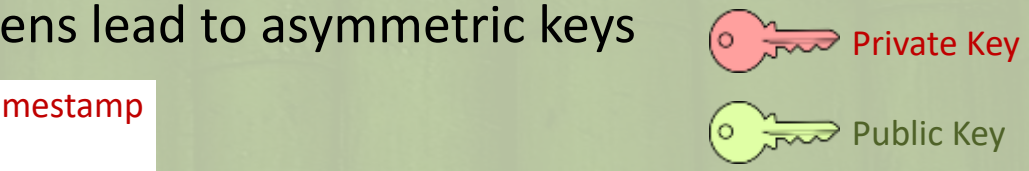
- Passwords lead to symmetric keys



Kerberos authentication

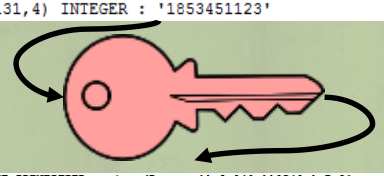
RSA Mode

- Smartcards/tokens lead to asymmetric keys



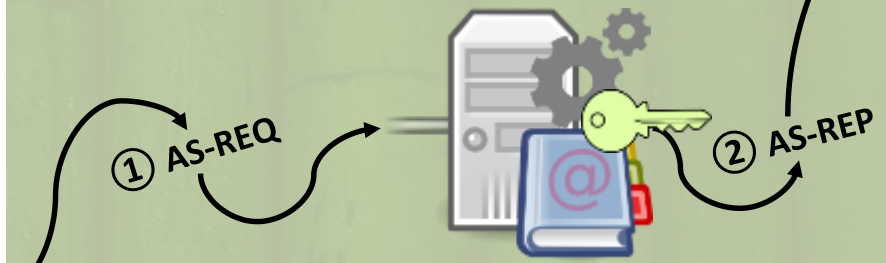

```
(58,77) SEQUENCE
  (60,30) CONTEXT SPECIFIC (0)
  (92,11) CONTEXT SPECIFIC (1)
  (105,3) CONTEXT SPECIFIC (2)
  (110,17) CONTEXT SPECIFIC (3)
    (112,15) GENERALIZED TIME : '201612162231262'
  (129,6) CONTEXT SPECIFIC (4)
    (131,4) INTEGER : '1853451123'
```

timestamp



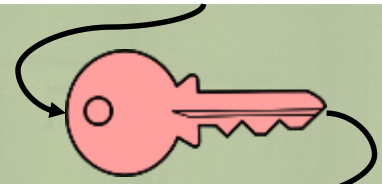
```
(4,9) OBJECT IDENTIFIER : signedData : '1.2.840.113549.1.7.2'
(15,2008) CONTEXT SPECIFIC (0)
  (19,2004) SEQUENCE
    (23,1) INTEGER : '3'
    (26,11) SET
      (39,96) SEQUENCE
        (41,7) OBJECT IDENTIFIER :  : '1.3.6.1.5.2.3.1'
        (50,85) CONTEXT SPECIFIC (0)
          (52,83) OCTET STRING
            (54,81) SEQUENCE
              (56,79) CONTEXT SPECIFIC (0)
                (58,77) SEQUENCE
                  (60,30) CONTEXT SPECIFIC (0)
                  (92,11) CONTEXT SPECIFIC (1)
                  (105,3) CONTEXT SPECIFIC (2)
                  (110,17) CONTEXT SPECIFIC (3)
                    (112,15) GENERALIZED TIME : '201612162231262'
                  (129,6) CONTEXT SPECIFIC (4)
                    (131,4) INTEGER : '1853451123'
    (137,1568) CONTEXT SPECIFIC (0)
      (1709,314) SET
        (1713,310) SEQUENCE
          (1717,1) INTEGER : '1'
          (1720,85) SEQUENCE
          (1807,9) SEQUENCE
          (1818,61) CONTEXT SPECIFIC (0)
            (1820,22) SEQUENCE
            (1844,35) SEQUENCE
              (1846,9) OBJECT IDENTIFIER : messageDigest : '1.2.840.113549.1.9.4'
              (1857,22) SET
                (1859,20) OCTET STRING : '0F38C624432E4F16D4B028DB46B5F34FEF444'
          (1881,13) SEQUENCE
            (1883,9) OBJECT IDENTIFIER : rsaEncryption : '1.2.840.113549.1.1.1'
            (1894,0) NULL
          (1896,128) OCTET STRING : 'D6951131B3BC00AEE4D930C9E4D573394A22CAE68DA9
```

signed timestamp + public key




```
(4,9) OBJECT IDENTIFIER : envelopedData : '1.2.840.113549.1.7.3'
(15,2584) CONTEXT SPECIFIC (0)
  (19,2580) SEQUENCE
    (23,1) INTEGER : '0'
    (26,239) SET
      (29,236) SEQUENCE
        (32,1) INTEGER : '0'
        (35,85) SEQUENCE
        (122,13) SEQUENCE
          (124,9) OBJECT IDENTIFIER : rsaEncryption : '1.2.840.113549.1.1.1'
          (135,0) NULL
        (137,128) OCTET STRING : '7E0B66BE537C6CA09E8A3594E45981C87496443FC91F
      (268,2331) SEQUENCE
        (272,9) OBJECT IDENTIFIER : signedData : '1.2.840.113549.1.7.2'
        (283,26) SEQUENCE
          (285,8) OBJECT IDENTIFIER : rc2CBC : '1.2.840.113549.3.2'
          (295,14) SEQUENCE
            (297,2) INTEGER : '160'
            (301,8) OCTET STRING : '0B19B9038EE43D38'
        (311,2288) CONTEXT SPECIFIC (0) : 'E82390E82436C2AA6476DB1E5BD4D74A2E94A6
```

TGT + encrypted session key



TGT

```
(6,27) CONTEXT SPECIFIC (0)
  (8,25) SEQUENCE
    (10,3) CONTEXT SPECIFIC (0)
      (12,1) INTEGER : '23'
    (15,18) CONTEXT SPECIFIC (1)
      (17,16) OCTET STRING : '93B645646DCC99
```

session key

Kerberos authentication

*RSA Mode + **Diffie–Hellman***

- Smartcards/tokens lead to asymmetric keys

Private Key

Public Key

timestamp + DH Parameters

```
(64,58) CONTEXT SPECIFIC (0)
(66,56) SEQUENCE
  (68,3) CONTEXT SPECIFIC (0)
    (70,1) INTEGER : '0'
  (73,17) CONTEXT SPECIFIC (1)
    (75,15) GENERALIZED TIME : '20161216223445Z'
  (92,6) CONTEXT SPECIFIC (2)
    (94,4) INTEGER : '1853451123'
  (100,22) CONTEXT SPECIFIC (3)
    (102,20) OCTET STRING : '0000000000000000000000000000000000000000'
(124,289) CONTEXT SPECIFIC (1)
(128,285) SEQUENCE
  (132,147) SEQUENCE
    (135,7) OBJECT IDENTIFIER : dhPublicNumber : '1.2.840.10046.2.1'
    (144,135) SEQUENCE
      (147,129) INTEGER : '00FFFFFFFFFFFFFFFFC90FDAA22168C234C4C6628B
      (279,1) INTEGER : '2'
  (282,132) BIT STRING UnusedBits: 0
    (286,128) INTEGER : '0DAA4406C282BE625A40B4A0D663598A625686BD1DE4F
```

Diffie-Hellman

① AS-REQ

② AS-REP

```
(42,7) OBJECT IDENTIFIER :  : '1.3.6.1.5.2.3.2'
(51,172) CONTEXT SPECIFIC (0)
  (54,169) OCTET STRING
    (57,166) SEQUENCE
      (60,136) CONTEXT SPECIFIC (0)
        (63,133) BIT STRING UnusedBits: 0
          (67,129) INTEGER : '00C5E191E0BAC80442EF5789A5
      (199,6) CONTEXT SPECIFIC (1)
        (201,4) INTEGER : '1853451123'
      (207,17) CONTEXT SPECIFIC (2)
        (209,15) GENERALIZED TIME : '201612162359192'
```

TGT + encrypted session key

Diffie-Hellman

signed timestamp + DH Parameters + public key

```
(1717,1) INTEGER : '1'
(1720,85) SEQUENCE
(1807,9) SEQUENCE
(1818,61) CONTEXT SPECIFIC (0)
  (1820,22) SEQUENCE
  (1844,35) SEQUENCE
    (1846,9) OBJECT IDENTIFIER : messageDigest : 1.2.840.113549.1.9.
    (1857,22) SET
      (1859,20) OCTET STRING : '0F38C624432E4F16D4B
(1881,13) SEQUENCE
  (1883,9) OBJECT IDENTIFIER : rsaEncryption : '1.
  (1894,0) NULL
(1896,128) OCTET STRING : 'D6951131B3BC00AEE4D930C9E4D573394A22CAE68DA9
```

TGT

session key

```
(-) SessionKey:
f41ec16389147c43a8dc423c5079eb3b19ef59b719c148f10cf964d6d6bc7af0
07f5a77b6bada41e94bd3308d0433dace3771965963f745d3fd32
0009bc9f9f68362eb319692f88d3a77113df5fbfd37c667f7c91d360f9fec576
4e8126020f57d5665651db95180e7a5228a1be4d6d761e690879d4e55199cb68

(-) Kerberos key (aes256_hmac):
5533c212ac890763bfb6a6d476e3e3ed394924815b35310ba4d9c78bf4c93d2e
```

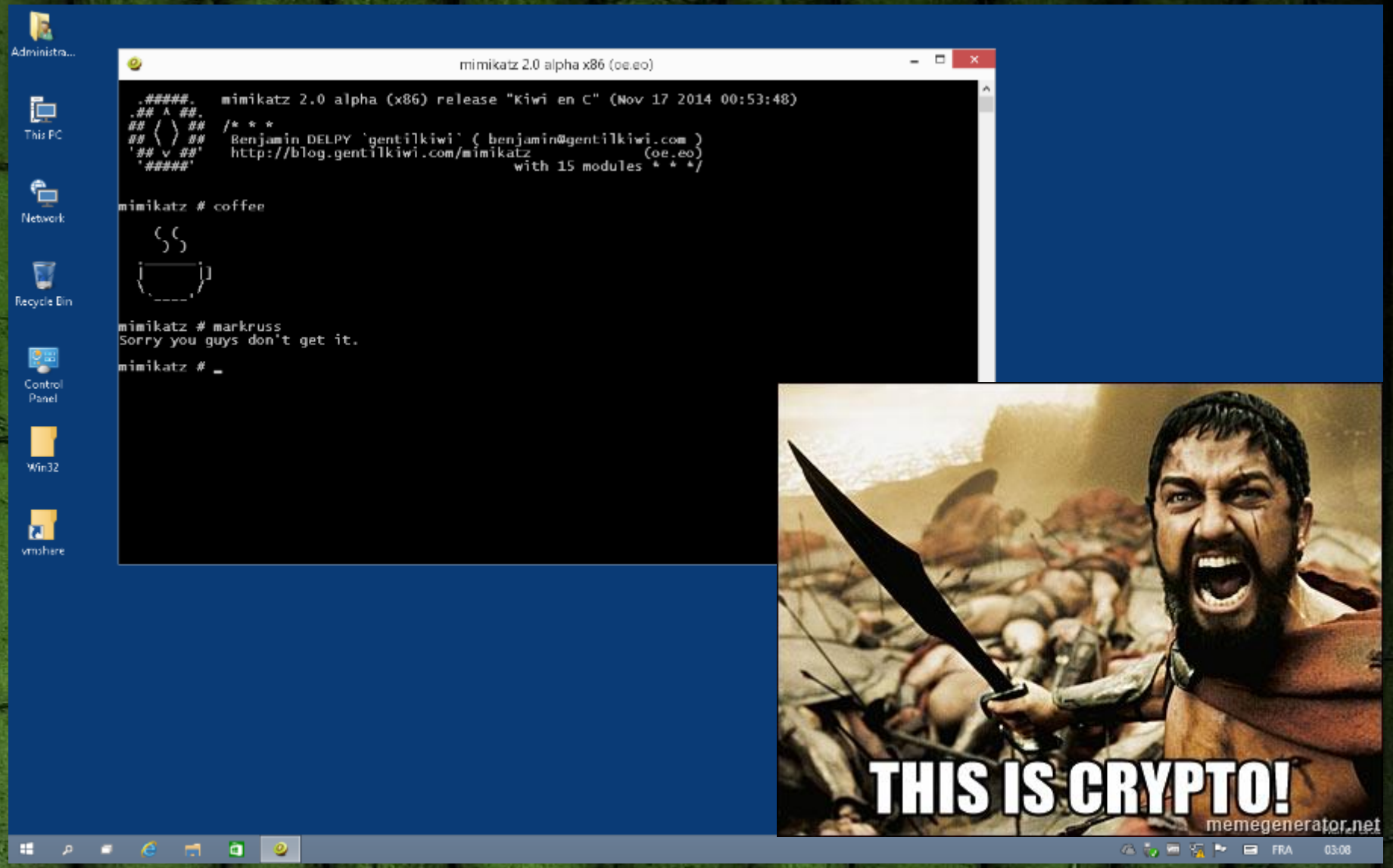
Kerberos authentication

Mode	Secret needed to encode AS-REQ	Secret needed to decode AS-REP
Password / Key	YES	YES
RSA	YES	YES
RSA with Diffie-Hellman	YES	**NO**

- **Once we have access to the Smartcard/Token, even for a short time, we can generate multiple pre-signed AS-REQ for future usage ☺**
  - as long as the source certificate validity (usually seen « years »)

- Do you remember ? Windows LSA service **keeps PIN code in memory**
  - Useful on Terminal Server where LSASS can control remote Smartcards (even virtual ones) ;)

Demo !


mimikatz 2.0 alpha x86 (oe.eo)
.#####. mimikatz 2.0 alpha (x86) release "Kiwi en C" (Nov 17 2014 00:53:48)
.## ^ ##.
/ \ ## /* * *
\ / ## Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'#####' with 15 modules * * */
mimikatz # coffee
mimikatz # markruss
Sorry you guys don't get it.
mimikatz #
THIS IS CRYPTO!
memegenerator.net

Kerberos authentication

RSA Mode + Diffie–Hellman

- Is this Windows specific : **NO**
  - RFC 4556 :

```
3.1.1.  Required Algorithms

   All PKINIT implementations MUST support the following algorithms:

   o  AS reply key enctypes: aes128-cts-hmac-sha1-96 and aes256-cts-
      hmac-sha1-96 [RFC3962].

   o  Signature algorithm: sha-1WithRSAEncryption [RFC3370].

   o  AS reply key delivery method: the Diffie-Hellman key delivery
      method, as described in Section 3.2.3.1.
```

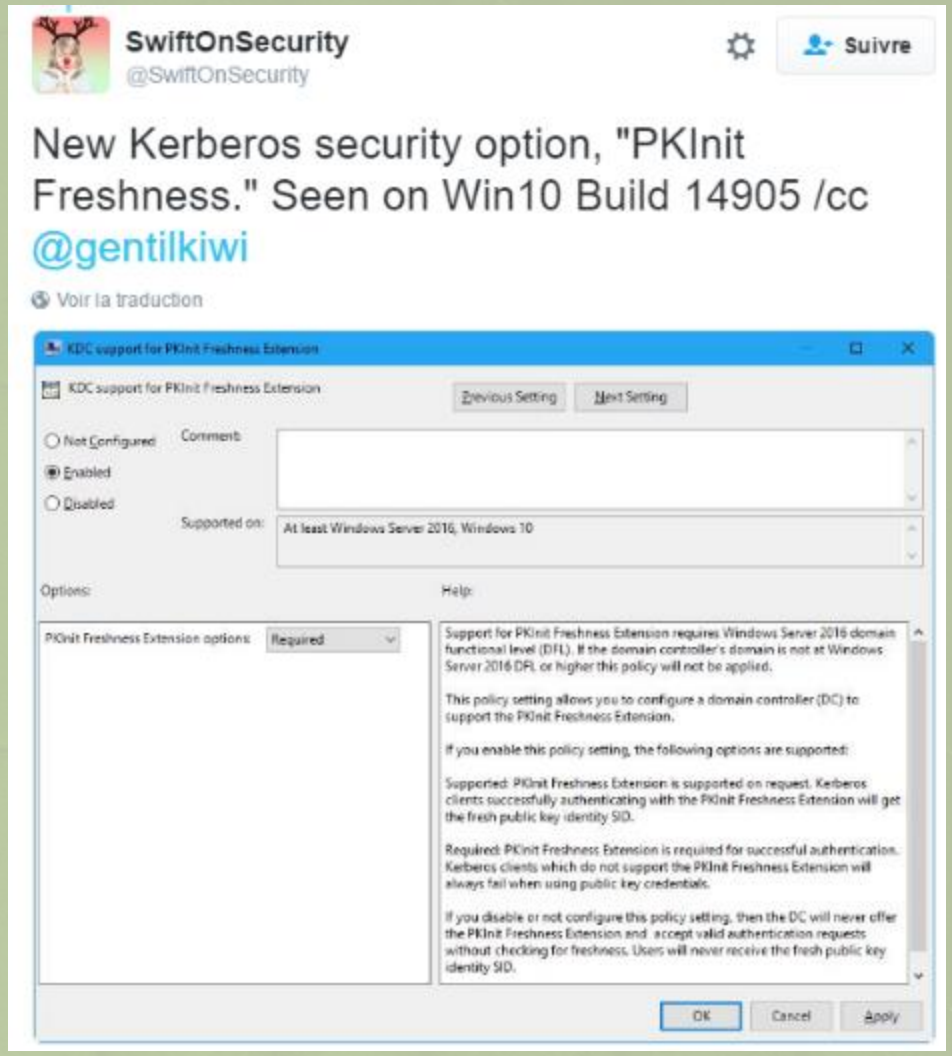
  - RFC 5349

```
   This document describes the use of Elliptic Curve certificates,
   Elliptic Curve signature schemes and Elliptic Curve Diffie-Hellman
   (ECDH) key agreement within the framework of PKINIT
```

Kerberos authentication

RSA Mode + Diffie–Hellman

- And what we can do?
  - Microsoft try to improve current Kerberos protocol by RFC draft:
    - https://datatracker.ietf.org/doc/draft-ietf-kitten-pkinit-freshness/
    - https://www.ietf.org/proceedings/91/slides/slides-91-kitten-1.pdf
  - They already implemented GPO for that (not tested) :
    - **But you must have a full network aware of it… (>= Windows 10 & 2016)**
  - Unless you use ECC certificates, it's not common to use DH with RSA certificates in Windows environment
    - Push some IPS rules to inspect AS-REQ… it's signed, NOT encrypted!
    - Windows Event Log does not seems to make differences between RSA and RSA/DH ☹



Kekeo bonus question?

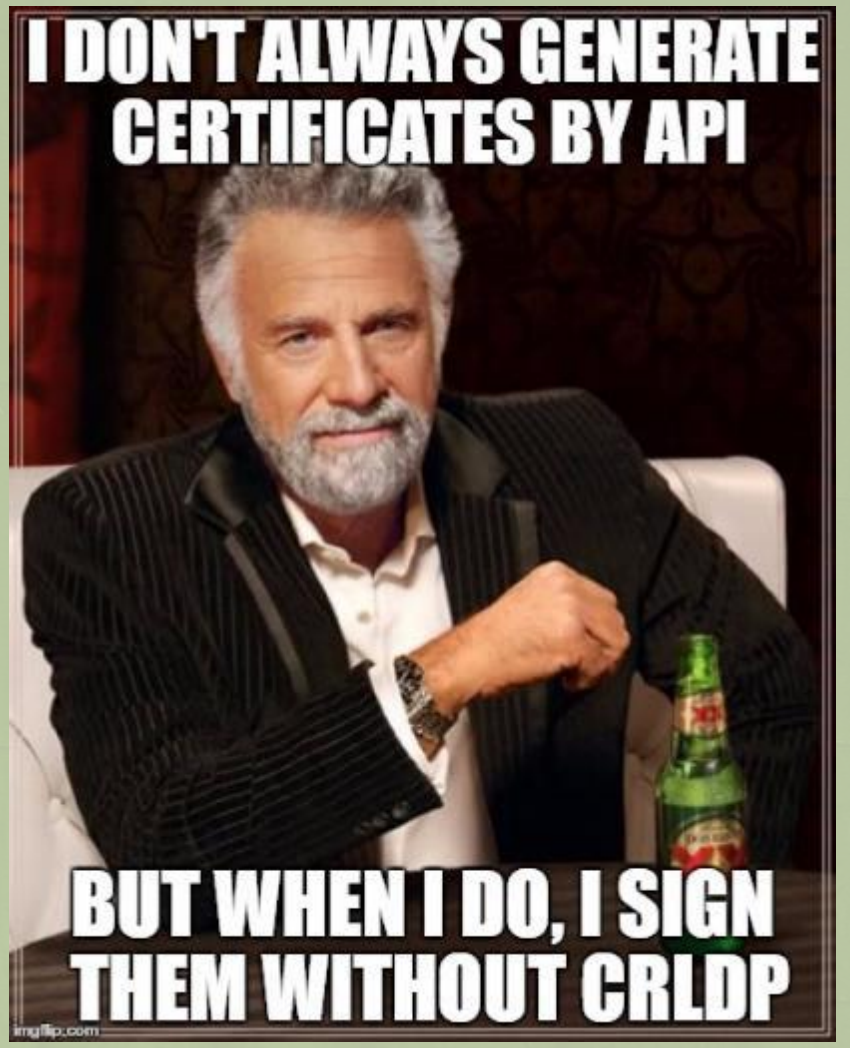
- Why do I use in my code:

```
if(CryptAcquireContext(&keyInfo->hProv, NULL, MS_ENH_DSS_DH_PROV, PROV_DSS_DH, CRYPT_VERIFYCONTEXT))
{
        ((PDWORD) (((PULONG_PTR) keyInfo->hProv)[28] ^ 0xa2491d83))[5] = 1; // :)
        ...
```

kekeo / mimikatz

a little bonus!

- Do you have a Windows PKI inside ?
- Do you know what's inside your **NTAuth** certificate store ?
- Is your certificate authority private key exportable ?
  - If software one: yes
  - No "but": **it is**.
- Who can access the Windows PKI Server ?
  - Think about backups ;)
- You think low level crypto operations make PKI logs ?
- What if I generate an unknown long term certificate…
  - without CRLDP?
  - How do you revoke it? ☺



That's all Folks!

- mimikatz http://blog.gentilkiwi.com/mimikatz (French)
- source https://github.com/gentilkiwi ← go here for **English** Wiki & Release (kekeo too) ;)
- contact @gentilkiwi / benjamin@gentilkiwi.com