Analyzing the attack surface of kernel registry filters

Matthieu Suiche, MoonSols

Bio

- Founder of MoonSols
- Kernel developer
- Forensic/Memory/ utilities author
  - Windd, DumpIt, SandMan etc.
- Private training about Memory Forensics
- Microsoft MVP

Agenda

- Introduction
- Filters
- Surface
- Testing
- Q&A

MoonSols

Potential Threats

- Race condition
- User-land pointers
- Null pointers

Registry filter

- Windows XP and later
  - CmRegisterCallback
- Windows Vista and later
  - CmRegisterCallbackEx
- CmUnRegisterCallback
- CmSetCallbackObjectContext

Registry Callback

- Pre and Post callback functions.
- REG_NOTIFY_CLASS
  - OpenKey, CreateKey, RenameKey, DeleteKey, SetValueKey, etc.
- Pre callback information structure
  - REG_*_KEY_INFORMATION
- Post callback information structure
  - REG_POST_OPERATION_INFORMATION
  - Contains a pointer to the structure above.

Registry Callback

- Maximum of 58 registry notify class (Pre, Post)
- 24 unique functions
- 24 + 1 unique structures
  - Including REG_POST_OPERATION_INFORMATION

REG_NOTIFY_CLASS value	Structure Type
RegNtDeleteKey	REG_DELETE_KEY_INFORMATION
RegNtSetValueKey	REG_SET_VALUE_KEY_INFORMATION
RegNtDeleteValueKey	REG_DELETE_VALUE_KEY_INFORMATION
RegNtSetInformationKey	REG_SET_INFORMATION_KEY_INFORMATION
RegNtRenameKey	REG_RENAME_KEY_INFORMATION
RegNtEnumerateKey	REG_ENUMERATE_KEY_INFORMATION
RegNtEnumerateValueKey	REG_ENUMERATE_VALUE_KEY_INFORMATION
RegNtQueryKey	REG_QUERY_KEY_INFORMATION
RegNtQueryValueKey	REG_QUERY_VALUE_KEY_INFORMATION
RegNtQueryMultipleValueKey	REG_QUERY_MULTIPLE_VALUE_KEY_INFORMATION
RegNtPreCreateKey	REG_PRE_CREATE_KEY_INFORMATION
RegNtPreCreateKeyEx	REG_CREATE_KEY_INFORMATION
RegNtPreOpenKey	REG_PRE_OPEN_KEY_INFORMATION
RegNtPreOpenKeyEx	REG_OPEN_KEY_INFORMATION
RegNtKeyHandleClose	REG_KEY_HANDLE_CLOSE_INFORMATION

REG_NOTIFY_CLASS value	Structure Type
RegNtPreFlushKey	REG_FLUSH_KEY_INFORMATION
RegNtPreLoadKey	REG_LOAD_KEY_INFORMATION
RegNtPreUnLoadKey	REG_UNLOAD_KEY_INFORMATION
RegNtPreQueryKeySecurity	REG_QUERY_KEY_SECURITY_INFORMATION
RegNtPreSetKeySecurity	REG_SET_KEY_SECURITY_INFORMATION
RegNtCallbackContextCleanup	REG_CALLBACK_CONTEXT_CLEANUP_INFORMATION
RegNtPreRestoreKey	REG_RESTORE_KEY_INFORMATION
RegNtPreSaveKey	REG_SAVE_KEY_INFORMATION
RegNtPreReplaceKey	REG_REPLACE_KEY_INFORMATION

Case #1

```
USHORT RegNotifyClass = (USHORT)Argument1;

    if (CallbackTable[RegNotifyClass] == NULL)
    {
        return STATUS_SUCCESS;
    }

    return (*(CallbackTable[RegNotifyClass]))
(CallbackContext, Argument1, Argument2);
```

REG_CREATE_KEY_INFORMATION	
PUNICODE_STRING	CompleteName
PVOID	RootObject
PVOID	ObjectType
ULONG	CreateOptions
PUNICODE_STRING	Class
PVOID	SecurityDescriptor
PVOID	SecurityQualityOfService
ACCESS_MASK	DesiredAccess
ACCESS_MASK	GrantedAccess
PULONG	**Disposition**
PVOID	*ResultObject
PVOID	CallContext
PVOID	RootObjectContext
PVOID	Transaction
PVOID	Reserved

Case #2

- Disposition is always going to be a kernel-pointer but…
  - Even if the PreviousMode is UserMode
- Developer would need to use ARGUMENT_PRESENT() macro.

REG_CREATE_KEY_INFORMATION	
PUNICODE_STRING	**CompleteName**
PVOID	**RootObject**
PVOID	ObjectType
ULONG	CreateOptions
PUNICODE_STRING	Class
PVOID	SecurityDescriptor
PVOID	SecurityQualityOfService
ACCESS_MASK	DesiredAccess
ACCESS_MASK	GrantedAccess
PULONG	Disposition
PVOID	*ResultObject
PVOID	CallContext
PVOID	RootObjectContext
PVOID	Transaction
PVOID	Reserved

Case #3

- Both fields, CompleteName and RootObject, are complementary

```
LONG WINAPI RegOpenKeyEx(
        HKEY hKey,
        LPCTSTR lpSubKey,
```

ObQueryNameString(RootObject) + CompleteName = **Potential buffer overflow**

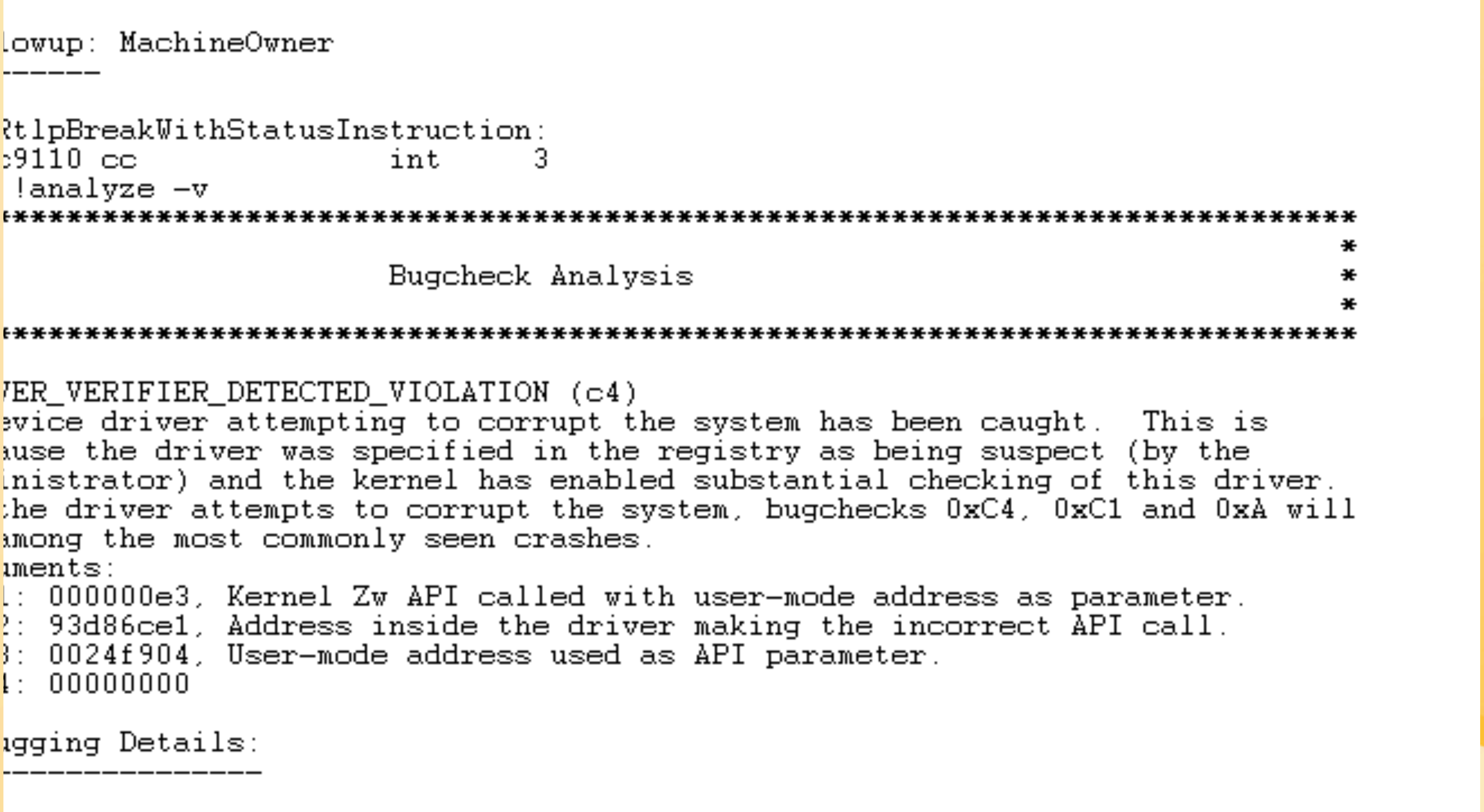
REG_POST_OPERATION_INFORMATION	
PVOID	Object
NTSTATUS	Status
PVOID	**PreInformation**
NTSTATUS	ReturnStatus
PVOID	CallContext
PVOID	**ObjectContext**
PVOID	Reserved

CmSetCallbackObjectContext

- Per object, developer-defined context structure !
- Usually called during a Create/Open key callback function.
- Initialization of this structure can be a goldmine of mistakes done by developers.

REG_QUERY_KEY_INFORMATION	
PVOID	Object
KEY_INFORMATION_CLASS	KeyInfomationClass
PVOID	**KeyInformation**
ULONG	Length
PULONG	ResultLength
PVOID	CallContext
PVOID	**ObjectContext**
PVOID	Reserved

- KeyInformation, and like most of UNICODE_STRING used by SetValueKey, DeleteKey etc.
- Is a user-mode pointer if the PreviousMode is equal to UserMode
- ☺



```
lowup: MachineOwner
-----

RtlpBreakWithStatusInstruction:
c9110 cc                int     3
 !analyze -v
***************************************************************************
                                                                          *
                     Bugcheck Analysis                                    *
                                                                          *
***************************************************************************

VER_VERIFIER_DETECTED_VIOLATION (c4)
evice driver attempting to corrupt the system has been caught.  This is
ause the driver was specified in the registry as being suspect (by the
inistrator) and the kernel has enabled substantial checking of this driver.
the driver attempts to corrupt the system, bugchecks 0xC4, 0xC1 and 0xA will
among the most commonly seen crashes.
uments:
1: 000000e3, Kernel Zw API called with user-mode address as parameter.
2: 93d86ce1, Address inside the driver making the incorrect API call.
3: 0024f904, User-mode address used as API parameter.
4: 00000000

ugging Details:
--------------
```

- Verifier is pretty useful to spot that kind of bugs
- ProbeForRead/ProbeForWrite

```
__try
    {
        ProbeForRead(KeyInfo->NewName->Buffer,
                     KeyInfo->NewName->Length, 1);
        RtlCopyMemory(Buffer,
                       KeyInfo->NewName->Buffer,
                       KeyInfo->NewName->Length);
    }
    __except (EXCEPTION_EXECUTE_HANDLER)
    {
        NtStatus = STATUS_INVALID_PARAMETER;
    }
```

Automated tests ?

- RegCbTestCtrl.exe
  - http://msdn.microsoft.com/en-us/library/gg607497%28v=vs.85%29.aspx

Altitude Conflict Test

CreateKey Block Test

CreateKey Bypass Test

CreateKey Override Access Denied Test

CreateKey Override Block Test

SetKeySecurity Bypass Test

Transacted CreateKey Bypass Test

Transacted CreateKey Bypass (No Commit) Test

Unregister Close Race Test

Save Restore Replace Test

References

- Fermin J. Serna – Windows Secure Kernel Development
  - http://zhodiac.hispahack.com/my-stuff/security/Windows_Secure_Kernel_Development.pptx
- Most of Tarjei Mandt talks. ☺

Q&A
msuiche@moonsols.com

MoonSols