

Forum: [Hack Forums](#)

Thread: [AMSI Bypass](#)

gigajew - January 27, 2020, 7:24 pm

This is my version of AMSI bypass only utilizing 2 Windows APIs (VirtualProtect and LoadLibrary). Thanks to RASTA MOUSE for creating the original sauce. If you don't know what AMSI is, read more here <https://docs.microsoft.com/en-us/windows/win32/amsi/antimalware-scan-interface-portal> I'm sure you've seen this error message before.

Anyway, here's the sauce:

Paskowsky - January 28, 2020, 12:41 am

what function is at base + 0x44a0 ?

I know using "magic offsets" is not a good practice :P

Would a binary search be more resistant to microsoft updates?

Regards,
Paskowsky

FuzzyInteger - January 28, 2020, 12:55 am

are you patching function ?

```
mov  eax,0x80070057
ret  0x18
```

```
mov  eax,0x80070057
ret
```

anyways this is so wrong VirtualProtect(addr, buffer.Length, 0x04, out int old)

remember memory page is 4096 byte each one

this code is very bad coded, instead of patch the function (wich is not good at all a lot of hardcoded address)

it's better hook them :) ... but of course u require to shellcoding :)

(Jan 28, 2020 01:41 AM)Paskowsky Wrote: [\[CITE\]](#) what function is at base + 0x44a0 ?

I know using "magic offsets" is not a good practice :P

Would a binary search be more resistant to microsoft updates?

Regards,
Paskowsky

seems to patch something in this export AmsiInitialize

gigajew - January 28, 2020, 5:15 am

(Jan 28, 2020 01:41 AM)Paskowsky Wrote: [\[CITE\]](#) what function is at base + 0x44a0 ?

I know using "magic offsets" is not a good practice :P

Would a binary search be more resistant to microsoft updates?

Regards,

Paskowsky

AmsiScanBuffer is the function that gets patched

(Jan 28, 2020 01:55 AM)FuzzyInteger Wrote: [\[CITE\]](#) are you patching function ?

```
mov  eax,0x80070057
ret  0x18
```

```
mov  eax,0x80070057
ret
```

anyways this is so wrong VirtualProtect(addr, buffer.Length, 0x04, out int old)

remember memory page is 4096 byte each one

this code is very bad coded, instead of patch the function (wich is not good at all a lot of hardcoded address)

it's better hook them :) ... but of course u require to shellcoding :)

seems to patch something in this export AmsiInitialize

If you read Rasta Mouses post he goes through the process very clearly, he determined that a clean scan returns 0x80070057 so the patch is basically just making the function return CLEAN upon a scan

<https://rastamouse.me/2018/11/amsiscanbuffer-bypass---part-3/>

FuzzyInteger - January 28, 2020, 5:55 am

(Jan 28, 2020 06:15 AM)gigajew Wrote: [\[CITE\]](#) If you read Rasta Mouses post he goes through the process very clearly, he determined that a clean scan returns 0x80070057 so the patch is basically just making the function return CLEAN upon a scan

<https://rastamouse.me/2018/11/amsiscanbuffer-bypass---part-3/>

interesting article, but as i say, patching a function it's not a very good idea, it's much better hook the function ?

also the function on amsi64bit is on imagebase + 2540 on amsi32bit is on imagebase + 45B0
so i think the source code you posted, do not work on a new win10 system (all patched)

also since Amsiscanbuffer is a function exported by the dll, why did u not get the pointer ? by the export table ? and hook the function ?

so, this code can be improved a lot, even future patch of microsoft will not solve the problem ?

This line have no sense:

```
Code IntPtr addr = new IntPtr(LoadLibrary("amsi.dll").ToInt64() + ((IntPtr.Size * 8 == 32) ? 0x44A0: 0x23E0));
```

instead of adding 0x44A0 32bit?: 0x23E0 64bit? why did u not get the pointer by passing

```
Code GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer")
```

?

HFDone - January 28, 2020, 8:24 am

(Jan 28, 2020 06:55 AM)FuzzyInteger Wrote: [\[CITE\]](#) interesting article, but as i say, patching a function it's not a very good idea, it's much better hook the function ?

also the function on amsi64bit is on imagebase + 2540 on amsi32bit is on imagebase + 45B0 so i think the source code you posted, do not work on a new win10 system (all patched)

also since Amsiscanbuffer is a function exported by the dll, why did u not get the pointer ? by the export table ? and hook the function ?

so, this code can be improved a lot, even future patch of microsoft will not solve the problem ?

This line have no sense:

```
Code IntPtr addr = new IntPtr(LoadLibrary("amsi.dll").ToInt64() + ((IntPtr.Size * 8 == 32) ? 0x44A0: 0x23E0));
```

instead of adding 0x44A0 32bit?: 0x23E0 64bit? why did u not get the pointer by passing

```
Code GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer")
```

?

dont know why he decide to change it the owner code do it

<https://github.com/rasta-mouse/AmsiScanBufferBypass/blob/master/ASBBypass/Program.cs>

Edit - maybe he got problem with GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer") in the powershell script

<https://rastamouse.me/2018/10/amsiscanbuffer-bypass---part-1>

FuzzyInteger - January 28, 2020, 9:07 am

(Jan 28, 2020 09:24 AM)HFDone Wrote: [\[CITE\]](#) Edit - maybe he got problem with GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer") in the powershell script

<https://rastamouse.me/2018/10/amsiscanbuffer-bypass---part-1>

the GetProcAddress can be replaced, just by walking the export table, LoadLibrary too can be replaced without any problems ?

doing a LoadLibrary + offset it's really weird and unsafe...

gigajew - January 29, 2020, 6:48 am

(Jan 28, 2020 10:07 AM)FuzzyInteger Wrote: [\[CITE\]](#) the GetProcAddress can be replaced, just by walking the export table, LoadLibrary too can be replaced without any problems ?

doing a LoadLibrary + offset it's really weird and unsafe...

You should make a thread with some samples I'm sure it'd get a lot of people's attention

FuzzyInteger - January 29, 2020, 9:37 am

(Jan 29, 2020 07:48 AM)gigajew Wrote: [\[CITE\]](#) You should make a thread with some samples I'm sure it'd get a lot of people's attention

since u can use api, just use some api with callback (callwindowproc example) now make a shellcode wich get the LOADED DLL from PEB

```
Code mov eax, Dword[fs:30h] ; PEB
    mov eax, Dword[eax + 0Ch] ; PEB->Ldr
    mov eax, Dword[eax + 14h] ; PEB->Ldr.InMemoryOrderModuleList.Flink
now loop since u get the AMSI unicode NAME
```

```
Code mov eax, Dword[eax] ; Next entry
    mov ebx, [eax+0x28] ; Check Unicode Dll Name
```

Now get the imagebase

```
Code mov eax, Dword[eax + 10h] ; entry's address
now u need just to check the EAT by doing
```

$e_lfanew + PeHeader.EAT.VirtualAddress + EAT.NumberOfNames * EAT.AddressOfFunctions + EAT.AddressOfNameOrdinals + EAT.AddressOfNames$

Loop on that and BOOOM u done....

simple as fuck ?

gigajew - January 29, 2020, 1:32 pm

(Jan 29, 2020 10:37 AM)FuzzyInteger Wrote: [\[CITE\]](#) since u can use api, just use some api with callback (callwindowproc example) now make a shellcode wich get the LOADED DLL from PEB

```
Code mov eax, Dword[fs:30h] ; PEB
    mov eax, Dword[eax + 0Ch] ; PEB->Ldr
    mov eax, Dword[eax + 14h] ; PEB->Ldr.InMemoryOrderModuleList.Flink
now loop since u get the AMSI unicode NAME
```

```
Code mov eax, Dword[eax] ; Next entry
    mov ebx, [eax+0x28] ; Check Unicode Dll Name
```

Now get the imagebase

```
Code mov eax, Dword[eax + 10h] ; entry's address
now u need just to check the EAT by doing
```

$e_lfanew + PeHeader.EAT.VirtualAddress + EAT.NumberOfNames * EAT.AddressOfFunctions + EAT.AddressOfNameOrdinals + EAT.AddressOfNames$

Loop on that and BOOOM u done....

simple as fugg ?

I'm sure it's simple enough but I'm saying make a thread and teach regular people what you know get some +rep

FuzzyInteger - January 29, 2020, 11:09 pm

(Jan 29, 2020 02:32 PM)gigajew Wrote: [\[CITE\]](#) I'm sure it's simple enough but I'm saying make a thread and teach regular people what you know get some +rep
i dont want +rep