Extracting Whitelisted Paths from Windows Defender ASR Rules

adamsvoboda.net/extracting-asr-rules

June 22, 2022

This blog post was made possible by the fantastic work and research done by @commail which you can read here.

Background

Recently I was presented with a scenario where I wanted to dump lsass.exe on a machine protected by Microsoft Defender for Endpoint (MDE/ATP) with the ASR rule to prevent lsass.exe dumps enabled.

For many red teams, lsass dumps may have fallen out of popularity with the plethora of other options we have to acquire credentials and perform lateral movement. When I need to make it happen, my first choice for lsass dumps these days is usually HandleKatz, because it's still pretty undetected by many EDR products (MDE/BitDefender/Cylance/etc).

So, how do you go about dumping lsass.exe on a box protected with MDE and ASR? Well, fortunately you can leverage a variety of whitelisted paths within the Defender ASR rules that help you achieve this. After finding a whitelisted exclusion path for the ASR rule you want to bypass, simply run your executable from that path!

Extracting Whitelisted Exclusions from Defender Signature Updates

Windows Defender signatures/rules are stored in VDM containers. Many of them are just Lua script files. It's possible to use a tool such as WDExtract to decrypt and extract all the PE images from these containers. By analyzing the extracted VDM you can pull whitelisted exclusion paths for ASR rules.

I will now demonstrate a very quick, hacky way to quickly get an updated list of potential exclusion paths for particular ASR rules.

Let's pick on the ASR rule for "Block credential stealing from the Windows local security authority subsystem".

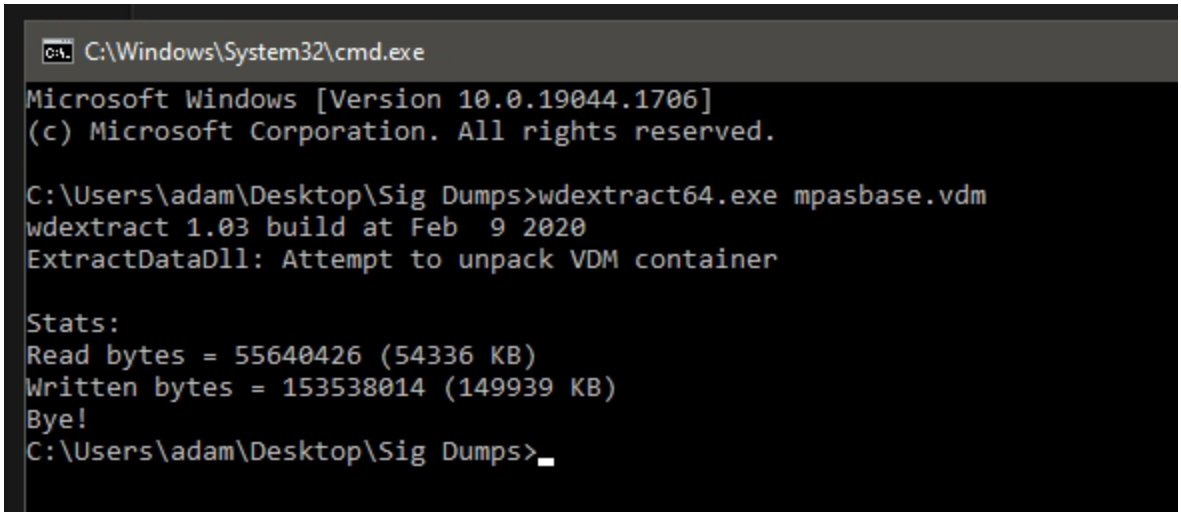
Here is a link to the particular ruleset on MSDN. Here you can see that the ASR rule is tied to a particular GUID, in this case `9e6c4e1f-7d60-472f-ba1a-a39ef669e4b2`.

This rule can be enabled on your machine with the following PowerShell script: `Set-MpPreference -AttackSurfaceReductionRules_Ids 9e6c4e1f-7d60-472f-ba1a-a39ef669e4b2 -AttackSurfaceReductionRules_Actions Enabled`

First, we need to locate the Defender signature files. You can usually find these in the following location: `C:\ProgramData\Microsoft\Windows Defender\Definition Updates\Backup`

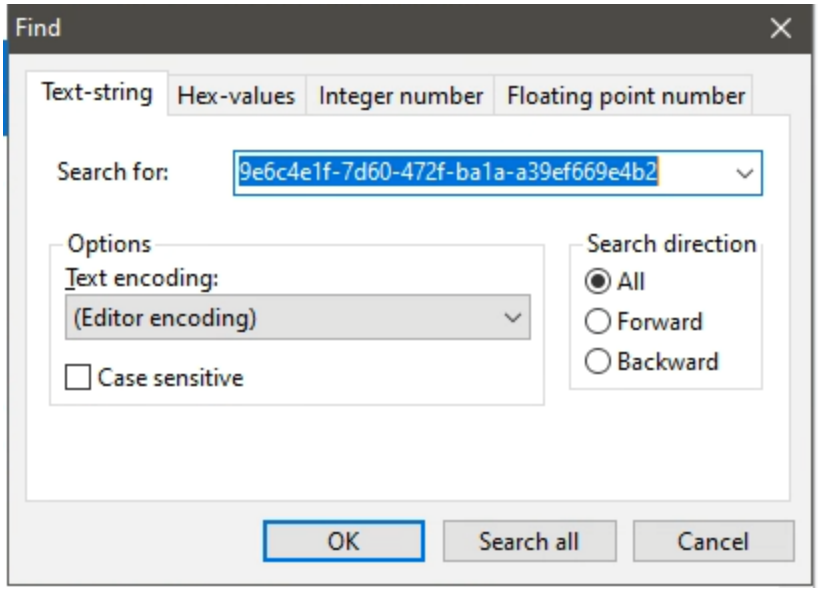
In our case, we are primarily interested in the `mpasbase.vdm` file.

Let's extract the file using WDExtract: `wdextract64.exe mpasbase.vdm`

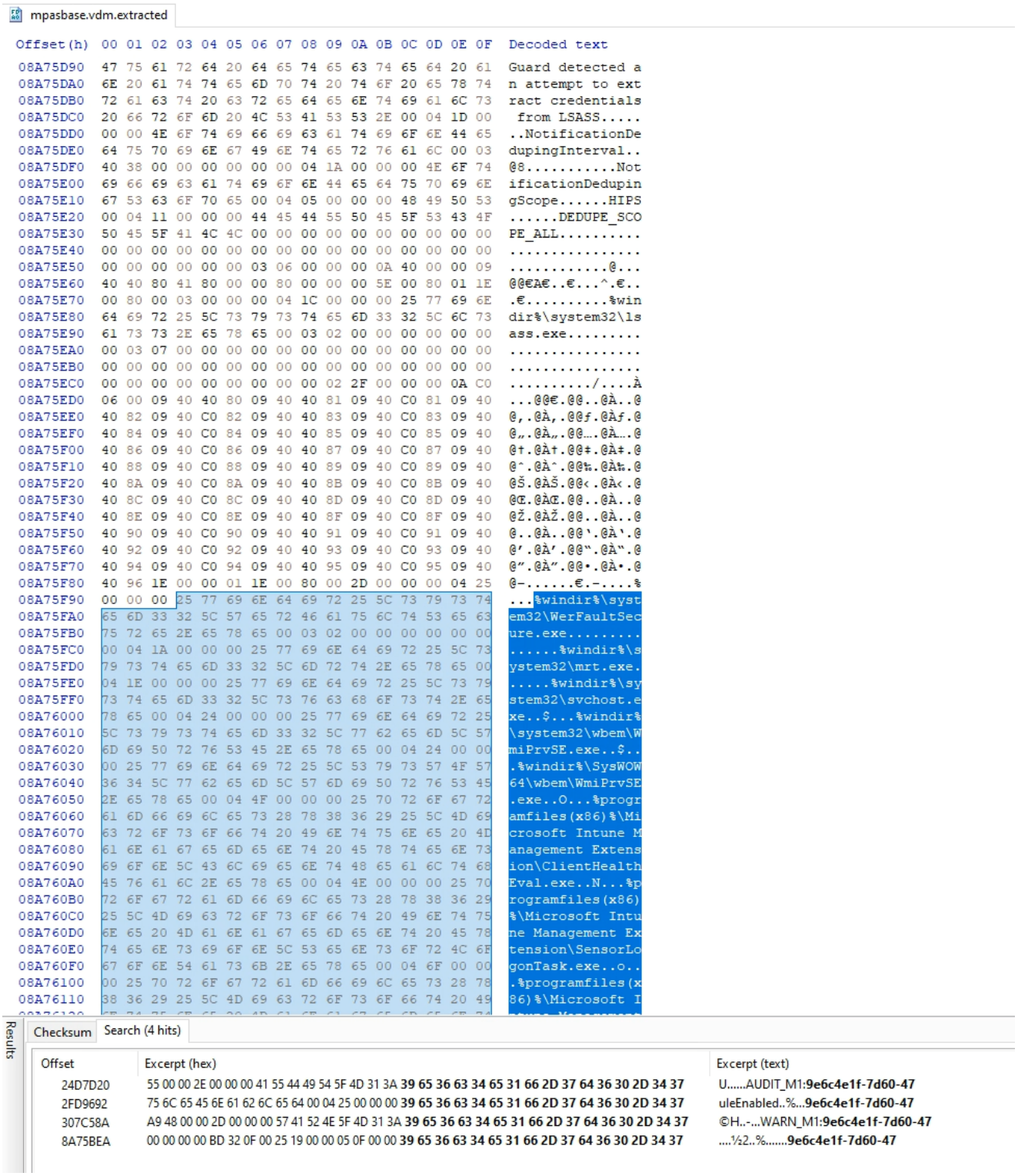


Open the extracted file `mpasbase.vdm.extracted` in a Hex Editor, such as HxD.

Search for the GUID of the ASR rule you want to investigate:



Scroll down slightly to see the list of exclusions and extract the data:



It's important to keep in mind that the list of paths you may see here in the hex dump are not always exclusions. They can be part of other paths listed for ASR rules such as Monitored Locations. You'll need to do some testing/investigating to confirm if you are just naivley using content from the hex dump.

Ultimately, this gives us a list of excluded paths that are allowed to perform lsass.exe dumps even with the ASR rule enabled:

```
%windir%\system32\WerFaultSecure.exe
%windir%\system32\mrt.exe
%windir%\system32\svchost.exe
%windir%\system32\wbem\WmiPrvSE.exe
%windir%\SysWOW64\wbem\WmiPrvSE.exe
%programfiles(x86)%\Microsoft Intune Management Extension\ClientHealthEval.exe
%programfiles(x86)%\Microsoft Intune Management Extension\SensorLogonTask.exe
%programfiles(x86)%\Microsoft Intune Management
Extension\Microsoft.Management.Services.IntuneWindowsAgent.exe
%programdata%\Microsoft\Windows Defender Advanced Threat
Protection\DataCollection\*\OpenHandleCollector.exe
%programfiles%\WindowsApps\Microsoft.GamingServices_*\gamingservices.exe
%programfiles(x86)%\Cisco\Cisco AnyConnect Secure Mobility Client\vpnagent.exe
%programfiles(x86)%\Zoom\bin\CptHost.exe
%programfiles(x86)%\Microsoft\EdgeUpdate\MicrosoftEdgeUpdate.exe
%programfiles(x86)%\Google\Update\GoogleUpdate.exe
%programfiles(x86)%\Splunk\bin\splunkd.exe
%programfiles%\Avecto\Privilege Guard Client\DefendpointService.exe
%programfiles%\Intel\SUR\QUEENCREEK\x64\esrv_svc.exe
%programfiles%\Microsoft Monitoring Agent\Agent\HealthService.exe
%programfiles%\Microsoft Monitoring Agent\Agent\MOMPerfSnapshotHelper.exe
%programfiles%\Nexthink\Collector\Collector\nxtsvc.exe
%programfiles%\Splunk\bin\splunkd.exe
%programfiles%\Azure Advanced Threat Protection
Sensor\*\Microsoft.Tri.Sensor.Updater.exe
%windir%\CCM\CcmExec.exe
%windir%\CCM\SensorLogonTask.exe
%windir%\Temp\Ctx-*\Extract\TrolleyExpress.exe
%programdata%\Citrix\Citrix Receiver*\TrolleyExpress.exe
%programdata%\Citrix\Citrix Workspace *\TrolleyExpress.exe
%programfiles(x86)%\Citrix\Citrix Workspace *\TrolleyExpress.exe
%temp%\Ctx-*\Extract\TrolleyExpress.exe
%programfiles%\Quest\ChangeAuditor\Agent\NPSrvHost.exe
%programfiles%\Quest\ChangeAuditor\Service\ChangeAuditor.Service.exe
%windir%\system32\DriverStore\FileRepository\hpqkbsoftwarecompnent.inf_amd64_*\HotKeyS

%windir%\system32\CompatTelRunner.exe
%programfiles(x86)%\Printer Properties Pro\Printer Installer
Client\PrinterInstallerClient.exe
%programfiles%\Printer Properties Pro\Printer Installer
Client\PrinterInstallerClient.exe
%programfiles(x86)%\Zscaler\ZSATunnel\ZSATunnel.exe
%programfiles%\Zscaler\ZSATunnel\ZSATunnel.exe
%programfiles(x86)%\ManageSoft\Security Agent\mgssecsvc.exe
%programfiles%\ManageSoft\Security Agent\mgssecsvc.exe
%programfiles(x86)%\Snow Software\Inventory\Agent\snowagent.exe
%programfiles%\Snow Software\Inventory\Agent\snowagent.exe
c:\windows\system32\WerFaultSecure.exe
c:\windows\system32\wbem\WmiPrvSE.exe
c:\windows\SysWOW64\wbem\WmiPrvSE.exe
```

To take it a step further, you can actually read the lua scripts by decompiling them after extraction with a tool such as MpLua converter. This will allow you to more clearly see how the rule logic works.

I strongly recommend exploring the great research by commail here for more details!

Credits

Theme Moonwalk