From Water to Wine: An Analysis of WINELOADER | Splunk

By Splunk Threat Research Team

Introduction

In late February 2024, Mandiant identified APT29, a Russian state-sponsored threat group, deploying a new backdoor called WINELOADER to target German political parties. This campaign marks a significant shift in APT29's targeting, as they have traditionally focused on government and diplomatic entities. The expansion to political parties suggests an evolution in the group's intelligence gathering priorities, likely influenced by the current geopolitical climate.

The attack chain begins with a spear-phishing email containing a malicious link to a ZIP file hosted on a compromised website. The ZIP file contains an HTML Application (HTA) file that, when executed, initiates a multi-stage infection process ultimately leading to the delivery of the WINELOADER backdoor.

This blog post provides a detailed analysis of the tactics, techniques, and procedures (TTPs) employed by APT29 in this campaign, focusing on two key aspects:

Initial Access: We'll examine the spear-phishing email, the compromised website hosting the malicious ZIP file, and the HTA file responsible for the initial stages of the infection chain.

WINELOADER Analysis: We'll dive deep into the WINELOADER backdoor, exploring its capabilities, command and control (C2) communication, and evasion techniques.

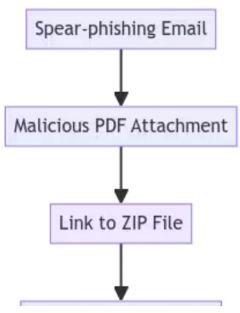
Furthermore, we'll showcase the Splunk security content developed by the Splunk Threat Research Team to help defenders detect and respond to this threat.

As APT29 continues to adapt and evolve their tactics, it is crucial for organizations to stay informed and prepared. By understanding the TTPs and malware employed in this campaign, security teams can enhance their detection capabilities and better protect their organizations from this sophisticated threat.

Initial Access

This section dives deeper into the TTPs employed by APT29 in the initial access stage of the WINELOADER campaign. By examining the spear-phishing attachment and the various components of the infection chain, we aim to provide defenders with the knowledge needed to identify and mitigate this threat.

The Initial Access TTPs



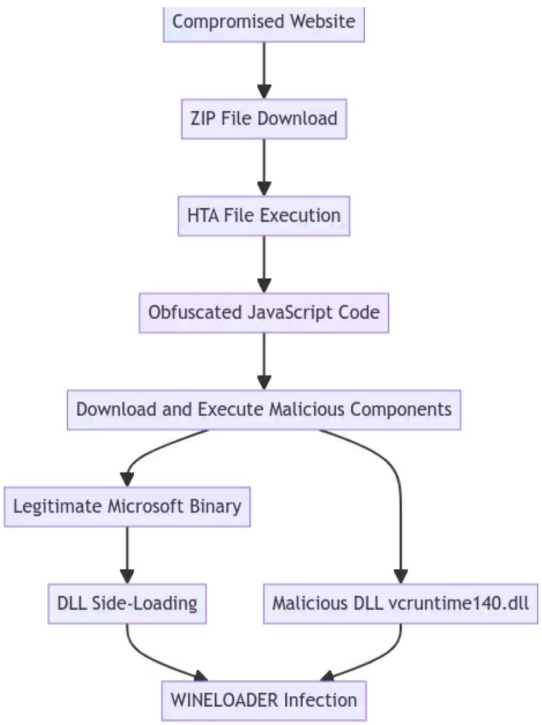


Figure 01: Attack Chain

The attack chain begins with a spear-phishing email containing a malicious PDF attachment. This PDF file, masquerading as an invitation to a wine tasting event, includes a link to a ZIP file hosted on a compromised website. The ZIP file contains an HTML Application (HTA) file named "wine.hta" or "invite.hta", depending on the sample analyzed.

Upon executing the HTA file, obfuscated JavaScript code initiates the next stage of the infection chain. The obfuscation technique used in this code matches patterns associated with the publicly available obfuscator "obfuscator.io". The HTA file downloads and executes additional malicious components, including:

A legitimate Microsoft-signed binary named "sqlwriter.exe" or "sqldumper.exe", which is vulnerable to DLL side-loading.

A malicious DLL named "vcruntime140.dll", crafted by the threat actor to be side-loaded by the legitimate binary.

The successful execution of the malicious DLL marks the beginning of the WINELOADER infection.

Assessing Detection Coverage with Atomic Red Team

To help defenders assess their detection coverage against the TTPs used in this campaign, the Splunk Threat Research Team has developed an Atomic Red Team test. This test provides a safe and controlled environment for security teams to evaluate their defenses and identify potential gaps in their detection capabilities.

The Atomic Red Team test developed by the Splunk Threat Research Team covers the following aspects of the initial access stage:

HTA with base64 encoded invite.txt file

Write invite.txt and decode the base64 to invite.zip

Extract the invite.zip, which contains Atomic Red Team T1574.002 gup.exe DLL side load

After extraction the HTA will then run gup.exe to simulate the DLL side load

We tried to mimic this as close to the WINELOADER infection chain, only not using SQLWriter or SQLdumper. During our testing, however, we enhanced our Atomic by embedding sqlwriter.exe with the malicious sample of vcruntime140.dll to emulate the behaviors.

By running these tests and analyzing the results, security teams can gain visibility into their detection and response to the TTPs employed by APT29 in the WINELOADER campaign. This can then be used to:

Fine-tune analytics.

Improve incident response procedures.

Ultimately strengthen the organization's overall security posture.

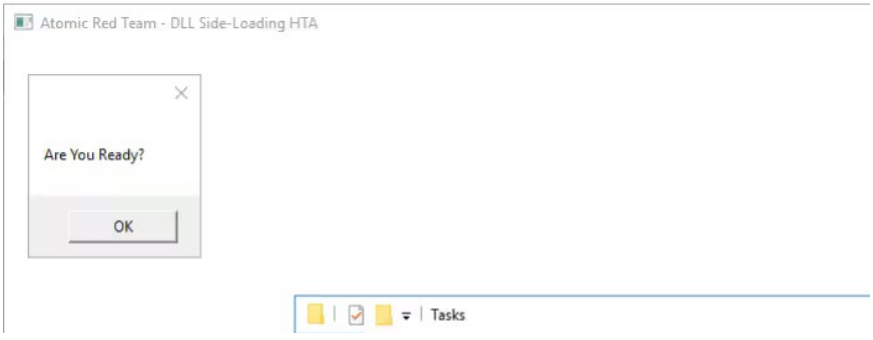
In our example, the HTA file writes the Base64 encoded content of `invite.zip` to a file at `C:\Windows\Tasks\invite.txt`. It then decodes this file from Base64 to a ZIP file and unzips it. After that, it runs `gup.exe` and displays a message box saying "DLL Side-Load Operation Completed."

The HTA is simple:

```
<html>
<head>
<title>Atomic Red Team - DLL Side-Loading HTA</title>
<HTA:APPLICATION ID="AtomicSideLoad" APPLICATIONNAME="AtomicSideLoad" BORDER="thin" BORDERSTYLE="normal" ICON="shell32.dll,4" >
<script language="VBScript">
Dim shell
Set shell = CreateObject("Wscript.Shell")

' Base64 encoded content of invite.zip
Dim base64EncodedContent
base64EncodedContent = "UEsDBBQAAAAIAC2FglgNVmbioloFAJAUCwAHAAAAR1VQLmV4ZdR9e3wTVfb4JGnatLRMeAQq8ggYNVjFYkWLLZjQpEwkhfKuPOvyqgpSIYEir0JabXYYt+t7fS27vl8rrsoCKrYU2yKv

' Path to write the encoded file and later, the decoded zip
Dim filePath
filePath = "C:\Windows\Tasks\invite.txt"

' Write the base64 encoded content to a file
Dim fso, textFile
Set fso = CreateObject("Scripting.FileSystemObject")
Set textFile = fso.OpenTextFile(filePath, 2, True)
textFile.Write base64EncodedContent
textFile.Close

' Decode the file from base64 to zip
shell.Run "certutil -decode " & filePath & " " & Replace(filePath, ".txt", ".zip"), 0, True

' Use PowerShell to unzip the file (alternative to tar)
Dim unzipPath
unzipPath = "C:\Windows\Tasks"
shell.Run "powershell -command Expand-Archive -Path " & Replace(filePath, ".txt", ".zip") & " -DestinationPath " & unzipPath, 0, True

MsgBox "Are You Ready?"
' Run gup.exe
shell.Run "C:\Windows\Tasks\gup.exe", 0, True

MsgBox "DLL Side-Load Operation Completed."
</script>
</head>
<body>
<h2>Atomic Test HTA</h2>
<img src="https://www.redcanary.com/wp-content/uploads/image2-25.png" alt="Atomic Red Team Logo" width="200" height="200">
<p>This Atomic Red Team test is brought to you by: <a href="https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1574.002/T1574.002.md#atomic-test-1-
</body>
```

Figure 02: malicious .HTA

Upon running the HTA file, most everything will occur in the background until a prompt occurs to notify that the gup.exe is ready to run. In this screenshot you can see the “Are You Ready?” prompt. Below the prompt is the c:\windows\tasks directory with the files ready to load.



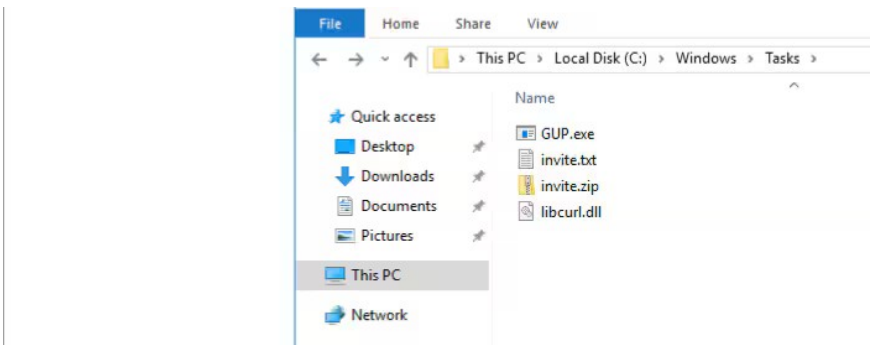


Figure 03.1: Simulation Attack

Upon clicking “OK,” the test will be completed by spawning calc.exe and a final message box from the HTA.

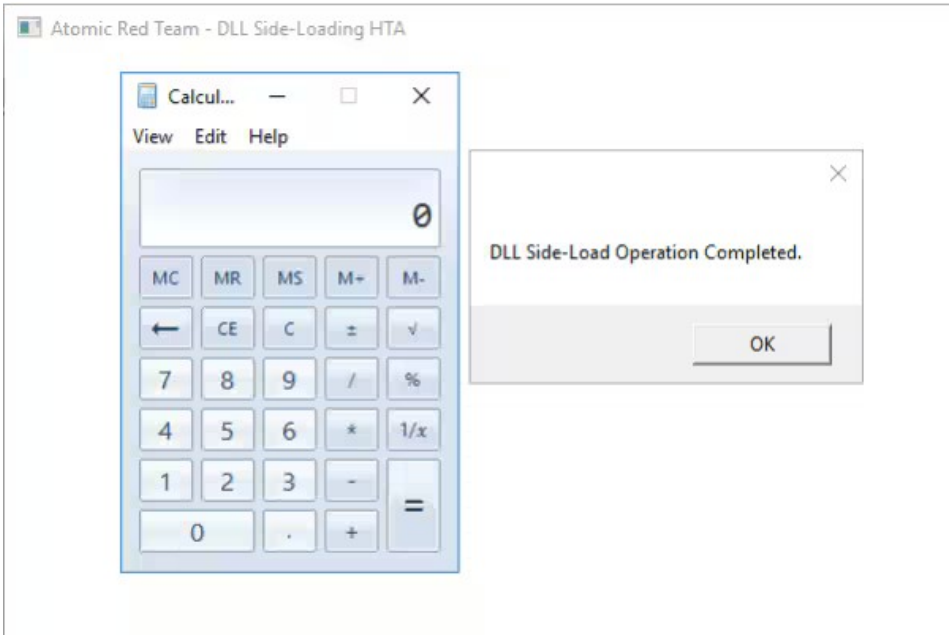


Figure 03.2: Simulation Attack

The last “OK” click will lead to the Atomic logo along with the reference to the DLL sideload test with Gup.exe

Next, check out this video for a live demonstration of our HTA. To try out the HTA, it is hosted on a GIST here.

WINELOADER Breakdown

The next section takes a closer look at the WINELOADER malware itself, examining its:

Capabilities

Communication methods

Evasion techniques

This variant of WINELOADER employs DLL side-loading techniques to execute its malicious payload. It achieves this by initiating the execution of either legitimate SQLWriter.exe or SQLDumper.exe, which in turn automatically loads a specially crafted vcruntime140.dll residing in the same directory as these applications.

SQLWriter.exe is a vital component of Microsoft SQL Server developed by Microsoft Corporation. SQLWriter installs a service facilitating backup/restore operations for Microsoft SQL Server via the Windows VSS infrastructure.

Alternatively, the malware may utilize legitimate Sqldumper.exe, responsible for generating dump files essential for Watson error reporting and debugging tasks.

In the WINELOADER samples analyzed by Zscaler and Mandiant, the Splunk Threat Research Team, observed that the specially crafted vcruntime140.dll exports 'memset' and '_set_se_translator', signaling the beginning of the code execution process.

This code segment is responsible for decrypting a block of 0x8028 bytes using the RC4 algorithm. The RC4 key is positioned after the code setup within the aforementioned export function.

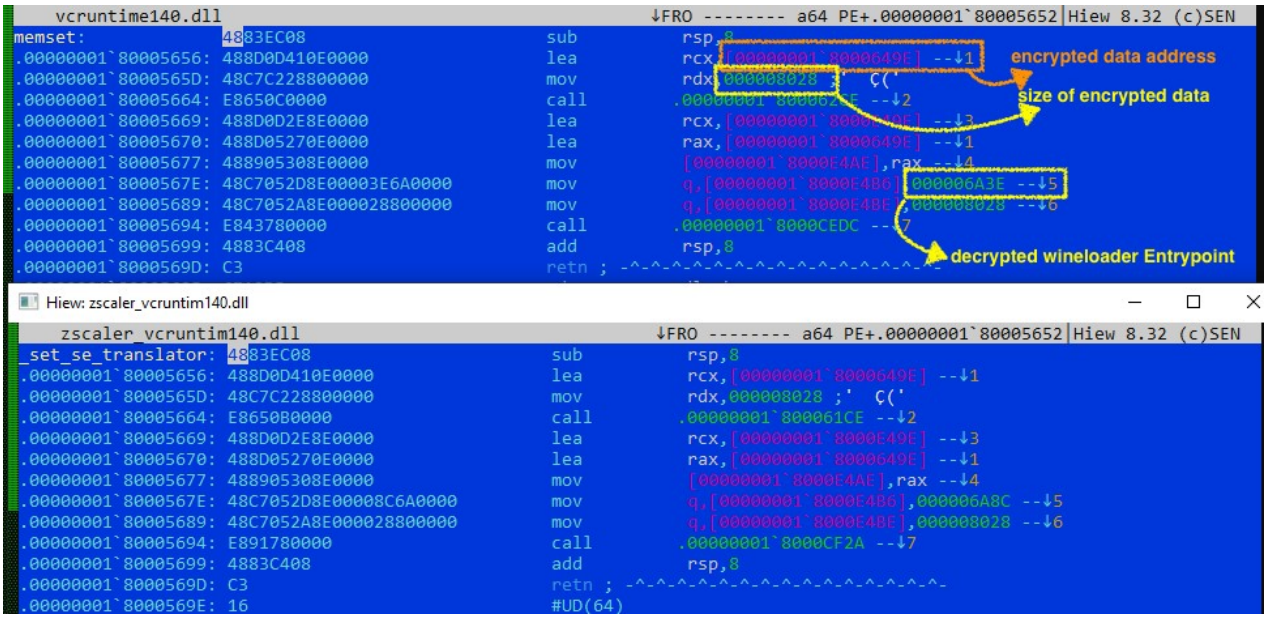


Figure 04: Export Functions

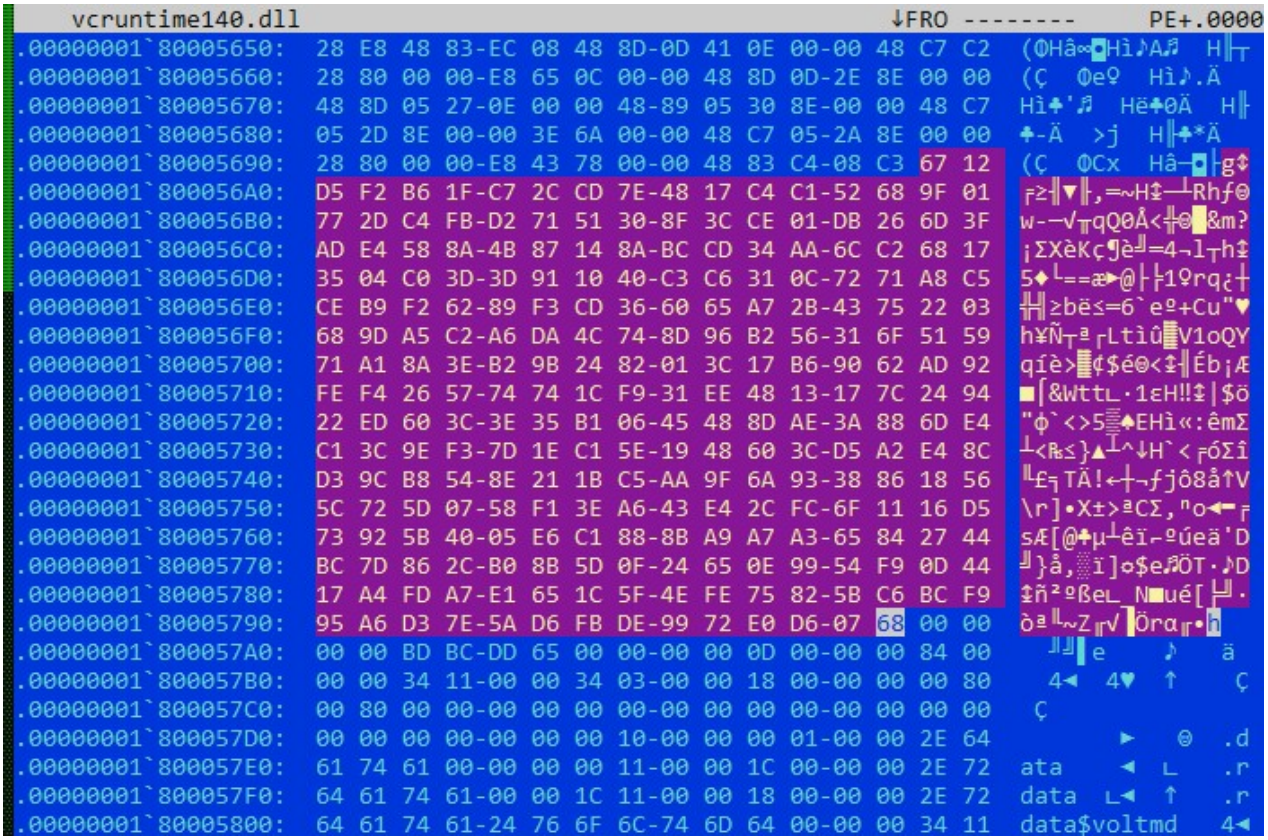


Figure 05: One of the RC4 Key

The decrypted data blob typically comprises a headless WINELOADER or shellcode, meticulously encrypted, especially regarding critical APIs and strings essential for its operations. This encryption strategy aims to thwart static analysis of its code.

The decryption routine employed by this WINELOADER variant uses yet another RC4 algorithm, with the RC4 key positioned at offset 0x20 within the decrypted headless WINELOADER.

The figure below shows the potential structure of the headless WINELOADER, highlighting key components such as the RC4 key and the encrypted strings table.

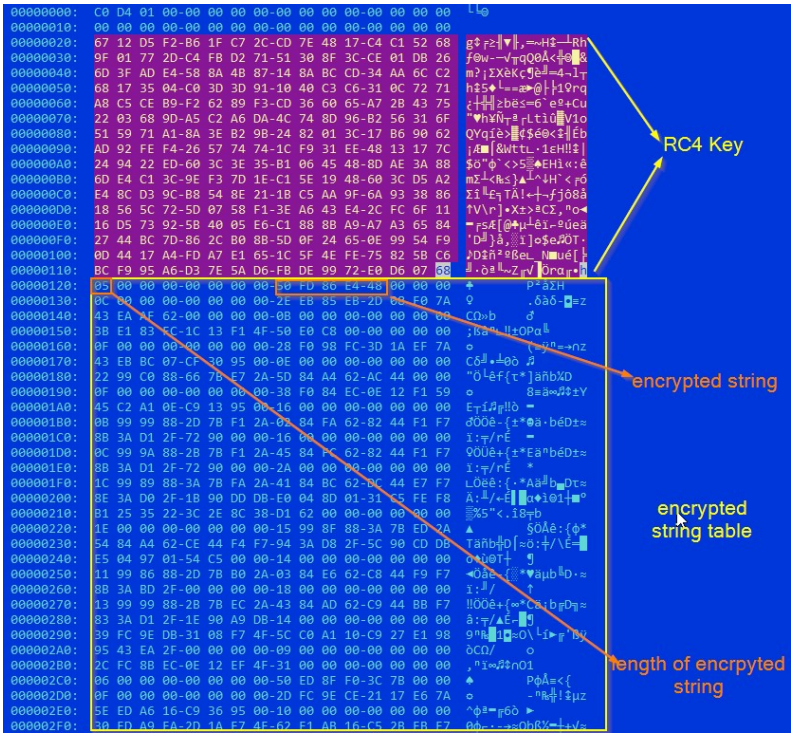


Figure 06: Decrypted WINELOADER

The figure below illustrates the following decrypted C2 information associated with the two WINELOADER variants we analyzed:

C2 Domains and Landing pages:

castechtools[.]com/api[.]php

siestakeying[.]com/auth[.]php

User Agents:

Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.1) Gecko/20100101 Firefox/86.1

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)

Figure 07: C2, User Agent & Landing Page

All of the C2 URLs are already down, so they aren't available to further analyze the WINELOADER infection chain.

Next, this brief video demonstrates how the information previously shared helped us in creating a simple tool to extract the headless WINELOADER from the specially crafted vcruntime140.dll for further analysis and TTP extraction.

The simplified version of this python tool is available here.

IOC

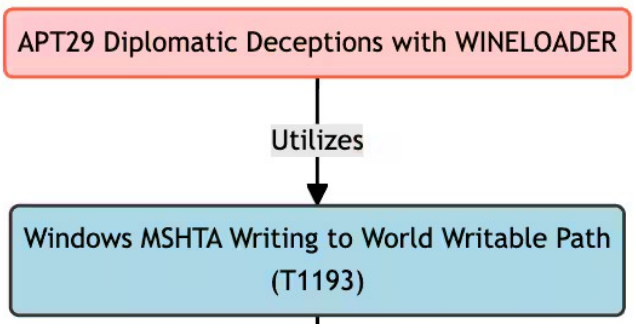
FileName	SHA256
vcruntime140.dll	72b92683052e0c813890caf7b4f8bfd331a8b2afc324dd545d46138f677178c4 d0a8fa332950b72968bdd1c8a1a0824dd479220d044e8c89a7dea4434b741750

YARA Rule:

```
import "pe"
rule possible_wine_loader_export_function {
   meta:
      author =  "@tccontre18 - Br3akpoint"
      description = "possible wine loader export function setup code"
      date =  "2024-04-03"
      sha256 = "72b92683052e0c813890caf7b4f8bfd331a8b2afc324dd545d46138f677178c4"
   strings:
      $exp_loader = {48 83 EC 08 48 8D 0D ?? ?? ?? ?? 48 C7 C2 28 80 00 00 E8 ?? ?? ?? ?? 48 8D 0D ?? ?? ?? ?? 48 8D 05 ?? ?? ?? ?? 48 89 05 ?? ?? ?? ?? 48 C7 05 ?? ?? ?? ?? ?? ?? 00 00 48 C7 05 ?? ?? ?? ?? 28 80 00 00 E8 ?? ?? 00 00 48 83 C4 08 C3}
   condition:
      uint16(0) == 0x5a4d and $exp_loader and pe.number_of_exports != 0
   }
```

Splunk Security Content

The Splunk Threat Research Team has released a new analytic story covering this campaign. Below is a breakdown of the related security content.



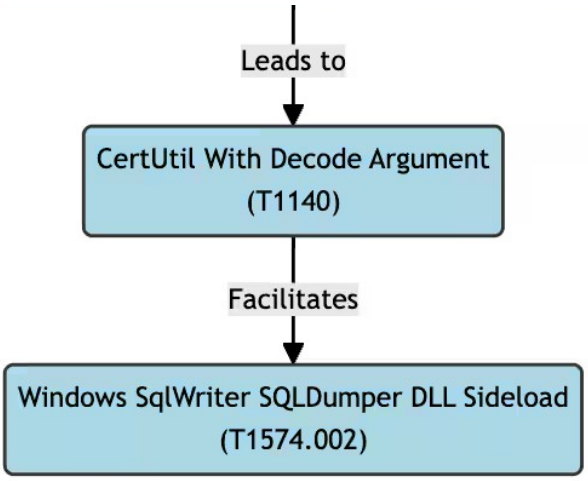


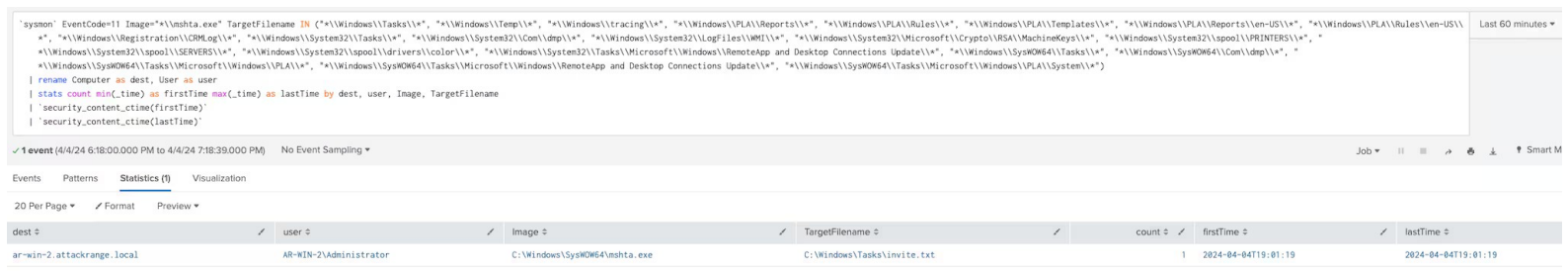
Figure 08: Security Content Detection Coverage

[Windows MSHTA Writing to World Writable Path]

This detection identifies instances of the Windows utility `mshta.exe` being used to write files to world-writable directories, a technique commonly leveraged by adversaries to execute malicious scripts or payloads.

Starting on February 26, 2024, APT29 has been observed distributing phishing attachments that lead to the download and execution of the ROOTSAW dropper via a compromised website. The ROOTSAW payload, using obfuscated JavaScript, downloads a file named `invite.txt` to the `C:\Windows\Tasks` directory. This file is then decoded and decompressed to execute a malicious payload

```
`sysmon` EventCode=11 Image="*\\mshta.exe" TargetFilename IN ("*\\Windows\\Tasks\\*", "*\\Windows\
\Temp\\*", "*\\Windows\\tracing\\*", "*\\Windows\\PLA\\Reports\\*", "*\\Windows\\PLA\\Rules\\*", "*\
\Windows\\PLA\\Templates\\*", "*\\Windows\\PLA\\Reports\\en-US\\*", "*\\Windows\\PLA\\Rules\\en-
US\\*", "*\\Windows\\Registration\\CRMLog\\*", "*\\Windows\\System32\\Tasks\\*", "*\\Windows\
\System32\\Com\\dmp\\*", "*\\Windows\\System32\\LogFiles\\WMI\\*", "*\\Windows\\System32\
\Microsoft\\Crypto\\RSA\\MachineKeys\\*", "*\\Windows\\System32\\spool\\PRINTERS\\*", "*\
\Windows\\System32\\spool\\SERVERS\\*", "*\\Windows\\System32\\spool\\drivers\\color\\*", "*\
\Windows\\System32\\Tasks\\Microsoft\\Windows\\RemoteApp and Desktop Connections Update\\*", "*\
\Windows\\SysWOW64\\Tasks\\*", "*\\Windows\\SysWOW64\\Com\\dmp\\*", "*\\Windows\
\SysWOW64\\Tasks\\Microsoft\\Windows\\PLA\\*", "*\\Windows\\SysWOW64\\Tasks\\Microsoft\
\Windows\\RemoteApp and Desktop Connections Update\\*", "*\\Windows\\SysWOW64\\Tasks\
\Microsoft\\Windows\\PLA\\System\\*")
 | rename Computer as dest, User as user
 | stats count min(_time) as firstTime max(_time) as lastTime by dest, user, Image, TargetFilename
 | `security_content_ctime(firstTime)`
 | `security_content_ctime(lastTime)`
```

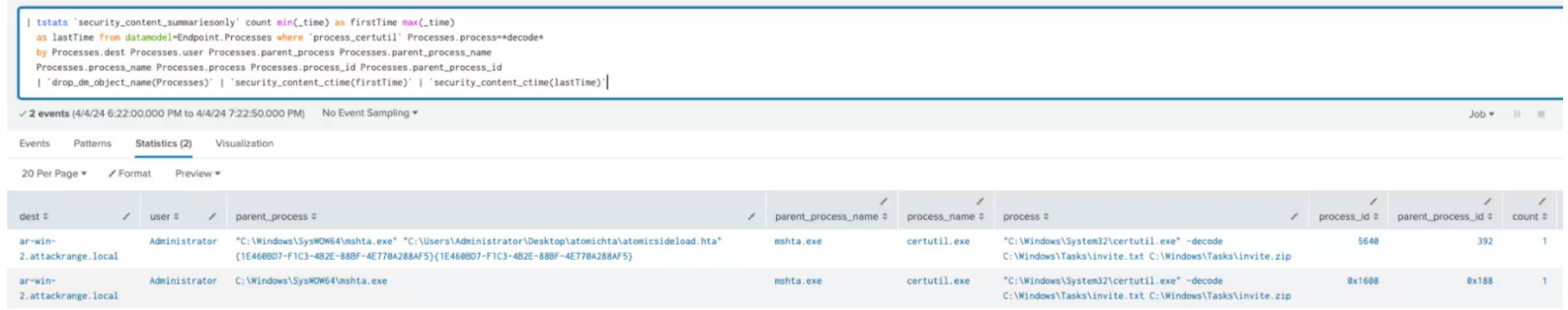


(Get this content: Windows MSHTA Writing to World Writable Path.)

CertUtil with Decode Argument

CertUtil.exe may be used to `encode` and `decode` a file, including portable executables and script code. Malicious usage will include decoding an encoded file that was downloaded.

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time)
 as lastTime from datamodel=Endpoint.Processes where `process_certutil` Processes.process=*decode*
 by Processes.dest Processes.user Processes.parent_process Processes.parent_process_name
 Processes.process_name Processes.process Processes.process_id Processes.parent_process_id
 | `drop_dm_object_name(Processes)` | `security_content_ctime(firstTime)` |
`security_content_ctime(lastTime)`
```



(Get this content: CertUtil with Decode Argument.)

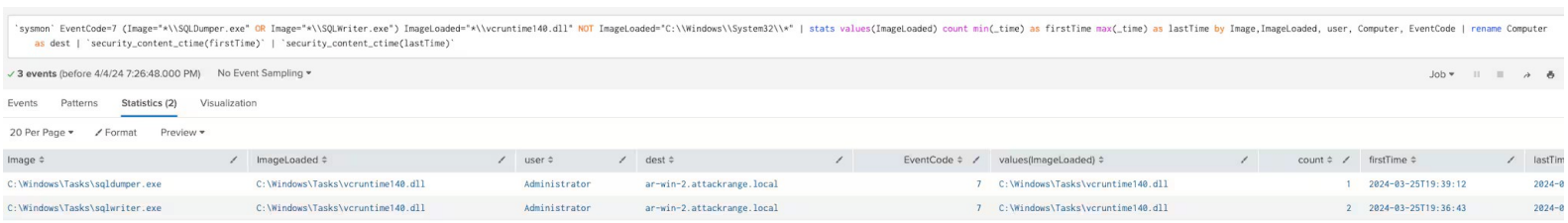
Windows SQLWriter SQLDumper DLL Sideload

This analytic identifies the abuse of SqlWriter and SQLDumper executables to sideload the vcruntime140.dll library. This technique is commonly used by adversaries to load malicious code into a legitimate process. The analytic:

Searches for EventCode 7 from Sysmon logs where the Image is either SQLDumper.exe or SQLWriter.exe and the ImageLoaded is vcruntime140.dll.

Filters out the legitimate loading of vcruntime140.dll from the System32 directory to reduce false positives.

```
`sysmon` EventCode=7 (Image="*\\SQLDumper.exe" OR Image="*\\SQLWriter.exe") ImageLoaded="*\
\vcruntime140.dll" NOT ImageLoaded="C:\\Windows\\System32\\*" | stats values(ImageLoaded) count
min(_time) as firstTime max(_time) as lastTime by Image,ImageLoaded, user, Computer, EventCode | rename
Computer as dest | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)`
```



(Get this content: [Windows SqlWriter SQLDumper DLL Sideload](#).)

Windows Unsigned MS DLL Side-Loading

The following analysis identifies potential DLL side-loading instances involving unsigned DLLs with a company detail signature mimicking Microsoft. This technique is frequently exploited by adversaries to execute malicious code automatically by running a legitimate process.

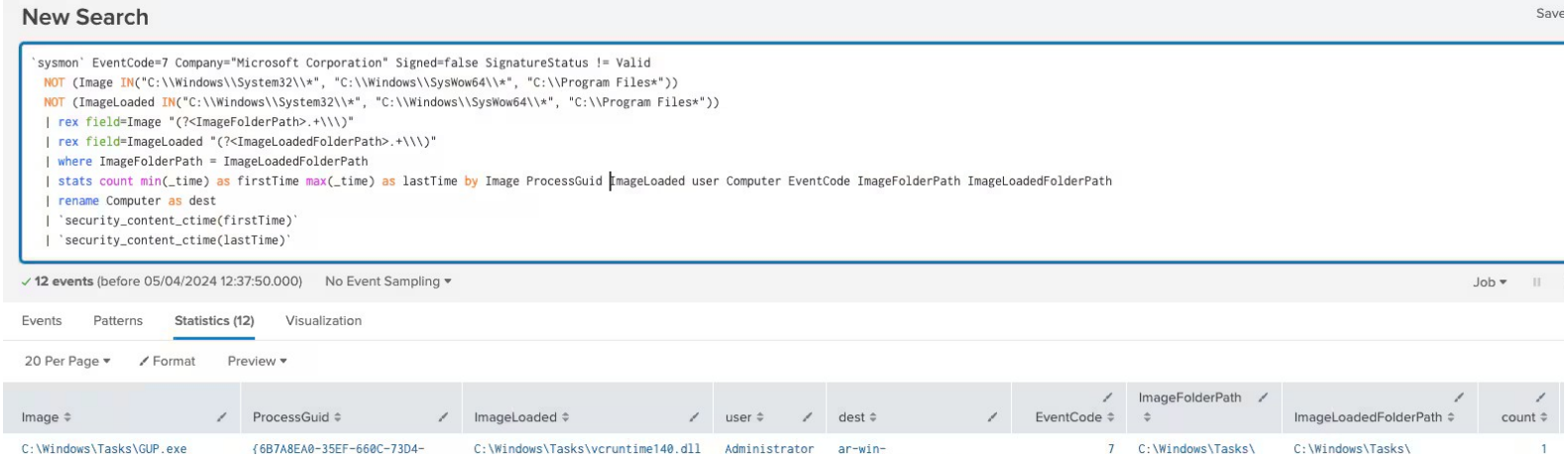
The analytics involves:

Searching Sysmon logs for Event Code 7, where both the `Image` and `ImageLoaded` paths do not match system directories (`system32`, `syswow64`, and `programfiles`).

Verifying whether the loaded DLL is signed and checking if the folder paths of the `Image` and `ImageLoaded` are identical.

This anomaly detection mechanism serves as a valuable indicator for identifying suspicious processes that load unsigned DLLs. Add other paths based on org hunting.

```
`sysmon` EventCode=7 Company="Microsoft Corporation" Signed=false SignatureStatus != Valid
 NOT (Image IN("C:\\Windows\\System32\\*", "C:\\Windows\\SysWow64\\*", "C:\\Program Files*"))
 NOT (ImageLoaded IN("C:\\Windows\\System32\\*", "C:\\Windows\\SysWow64\\*", "C:\\Program
Files*"))
 | rex field=Image "(?<ImageFolderPath>.+\\\)"
 | rex field=ImageLoaded "(?<ImageLoadedFolderPath>.+\\\)"
 | where ImageFolderPath = ImageLoadedFolderPath
 | stats count min(_time) as firstTime max(_time) as lastTime by Image ProcessGuid ImageLoaded user
Computer EventCode ImageFolderPath ImageLoadedFolderPath Company Description Product Signed
SignatureStatus
 | rename Computer as dest
 | `security_content_ctime(firstTime)`
 | `security_content_ctime(lastTime)`
 | `windows_unsigned_ms_dll_side_loading_filter`'
```



	040000000F03}			2.attackrange.local				
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-3683-660C-93D4-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-3B04-660C-08D5-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-3B63-660C-22D5-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-4033-660C-98D5-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-4208-660C-CFD5-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-42D2-660C-EFD5-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-8B66-660D-66F3-040000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\GUP.exe	{6B7A8EA0-E418-660E-3618-050000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\sqldumper.exe	{6B7A8EA0-D2E0-6601-BEE6-030000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\sqlwriter.exe	{6B7A8EA0-D24B-6601-A6E6-030000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1
C:\Windows\Tasks\sqlwriter.exe	{6B7A8EA0-D2E4-6601-BFE6-030000000F03}	C:\Windows\Tasks\vcruntime140.dll	Administrator	ar-win-2.attackrange.local	7	C:\Windows\Tasks\	C:\Windows\Tasks\	1

Summary

APT29 has launched a new campaign targeting political parties using the WINELOADER backdoor. Our detailed analysis of the TTPs employed by APT29, focused on the initial access stage and the WINELOADER malware itself. To help organizations detect and respond to this threat, The Splunk Threat Research Team has:

Developed an Atomic Red Team test.

Released a new analytic story.

As APT29 continues to evolve, it is important for security teams to stay informed and enhance their detection capabilities to protect against sophisticated threats.

Learn More

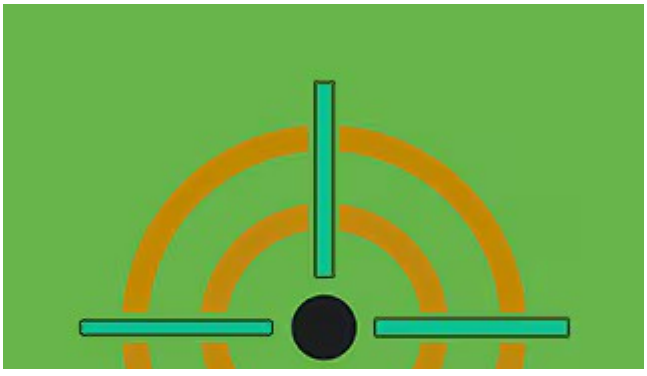
Visit research.splunk.com to view the Splunk Threat Research Team's complete security content repository. You can implement this content using the Enterprise Security Content Updates app or the Splunk Security Essentials app.

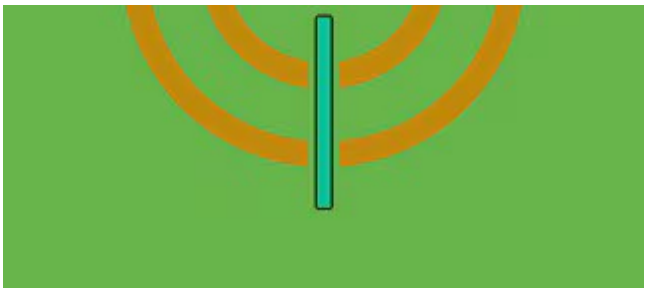
We would like to thank Teoderick Contreras and Michael Haag for authoring this post and the entire Splunk Threat Research Team for their contributions.

References

https://www.mandiant.com/resources/blog/apt29-wineloader-german-political-parties

https://www.zscaler.com/blogs/security-research/european-diplomats-targeted-spikedwine-wineloader





The Splunk Threat Research Team is an active part of a customer's overall defense strategy by enhancing Splunk security offerings with verified research and security content such as use cases, detection searches, and playbooks. We help security teams around the globe strengthen operations by providing tactical guidance and insights to detect, investigate and respond against the latest threats. The Splunk Threat Research Team focuses on understanding how threats, actors, and vulnerabilities work, and the team replicates attacks which are stored as datasets in the Attack Data repository.

Our goal is to provide security teams with research they can leverage in their day to day operations and to become the industry standard for SIEM detections. We are a team of industry-recognized experts who are encouraged to improve the security industry by sharing our work with the community via conference talks, open-sourcing projects, and writing white papers or blogs. You will also find us presenting our research at conferences such as Defcon, Blackhat, RSA, and many more.

Read more Splunk Security Content.