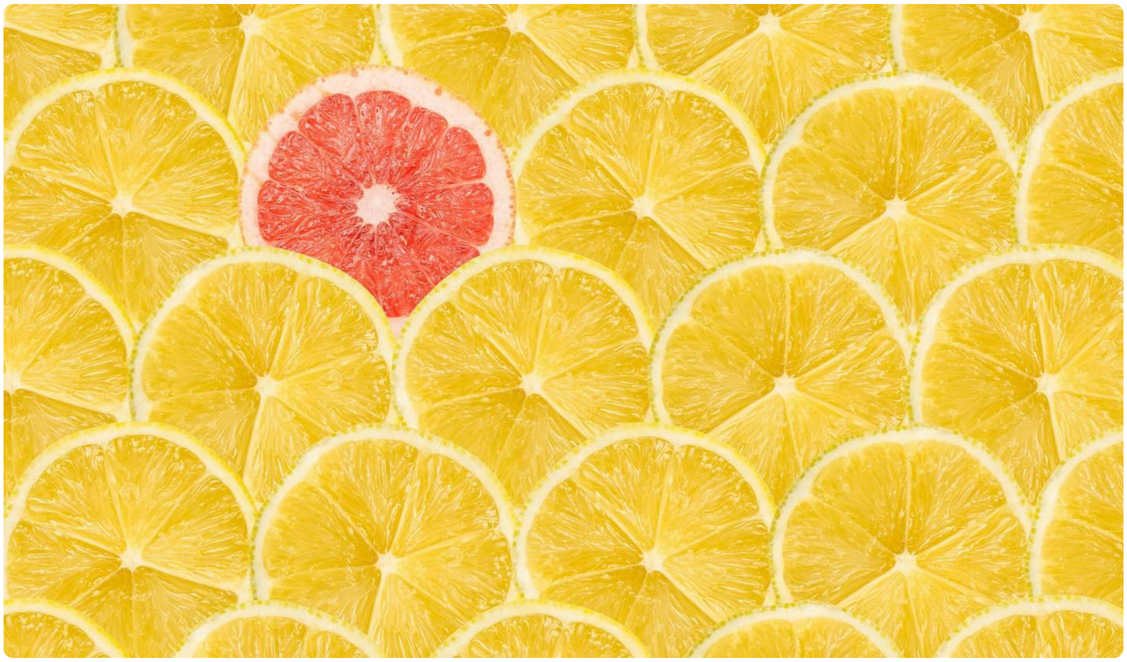
# REF2924: how to maintain persistence as an (advanced?) threat

Subscribe



## Preamble

In recent months, there has been a noticeable shift in the nature of the incidents being tracked under REF2924. Initially, the attacker employed custom, purpose-built malware. As the attack evolved, we observed the same group resorting to the use of open source tools or publicly available source code as a basis for developing new capabilities.

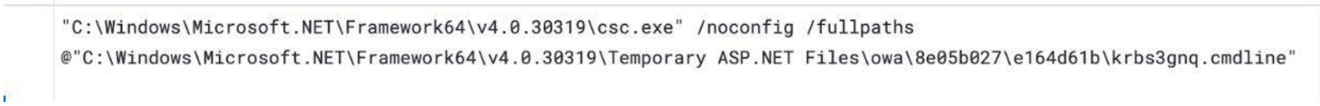
## Key takeaways

The attacker has shifted from using custom malware to open source tools or publicly available source code to develop new capabilities.

- The attacker has also deployed open source tools like TFirewall and AdFind in the victim's environment.
- In order to maintain persistence the attacker has deployed multiple different tools and techniques.

## .NET Webshell

On February 16th, 2023 Elastic Security Labs observed the Microsoft .NET compiler ( `csc.exe` ) being used to compile a DLL file,. The output was identified by Elastic Defend as a malicious file. Analysts who may have observed dynamic runtime compilation of .NET web shells should note that this was performed by the operator, not automatically by the system.

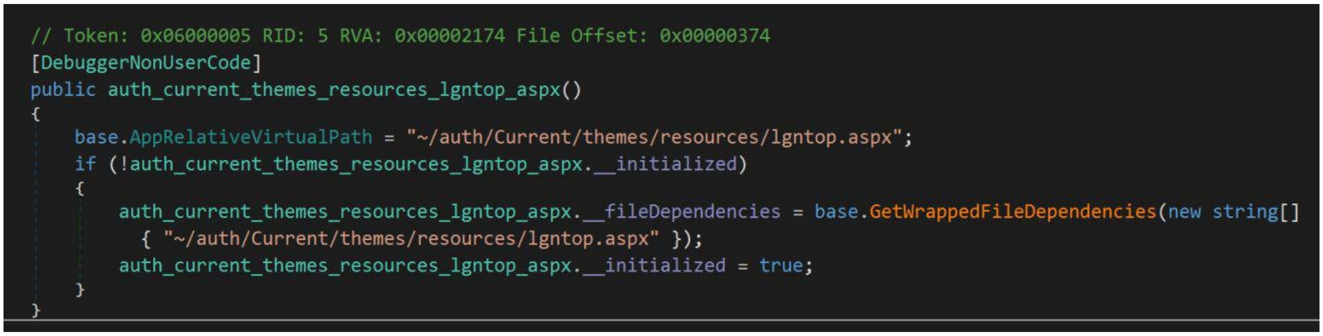


```
"C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe" /noconfig /fullpaths
@"C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Temporary ASP.NET Files\owa\8e05b027\e164d61b\krbs3gnq.cmdline"
```

The resulting output file was named `App_Web_lgntop.aspx.ec688436.pkx46see.dll` (a50ca8df4181918fe0636272f31e19815f1b97cce6d871e15e03b0ee0e3da17b) and was the subject of malware analysis.

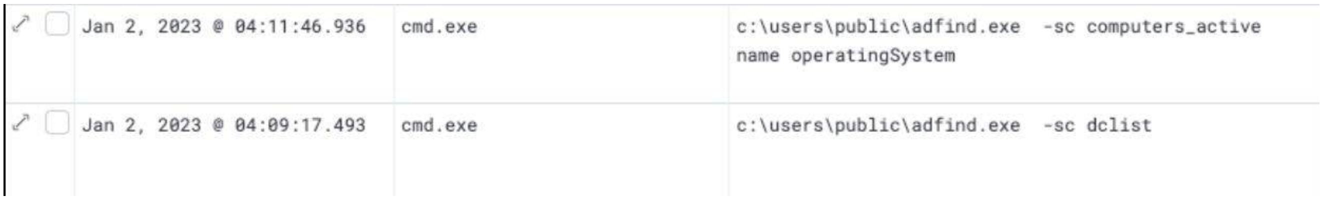
### Analysis

The web shell requires a small amount of pre-configuration to ensure it listens for the correct URI. In this case the path will be " `~/auth/Current/themes/resources/lgntop.aspx`".



```
// Token: 0x06000005 RID: 5 RVA: 0x00002174 File Offset: 0x00000374
[DebuggerNonUserCode]
public auth_current_themes_resources_lgntop_aspx()
{
    base.AppRelativeVirtualPath = "~/auth/Current/themes/resources/lgntop.aspx";
    if (!auth_current_themes_resources_lgntop_aspx.__initialized)
    {
        auth_current_themes_resources_lgntop_aspx.__fileDependencies = base.GetWrappedFileDependencies(new string[]
          { "~/auth/Current/themes/resources/lgntop.aspx" });
        auth_current_themes_resources_lgntop_aspx.__initialized = true;
    }
}
```

This path is expected on Microsoft Exchange Outlook Web Access (OWA) sites, so it was likely selected to blend in with the OWA service that is running on the target server. Once a web request is received it is processed by the following method.



| | | | |
|---|---|---|---|
| ☐ | Jan 2, 2023 @ 04:11:46.936 | cmd.exe | c:\users\public\adfind.exe -sc computers_active name operatingSystem |
| ☐ | Jan 2, 2023 @ 04:09:17.493 | cmd.exe | c:\users\public\adfind.exe -sc dclist |

This method checks if a specific HTTP header named XFF is present in the request headers. If it is present and its value, after passing through an MD5 hash function and a substring function, matches the string " 19267E61029B4546", then the method proceeds to execute the rest of the code. The string is likely used as an authentication key to prevent others from using the webshell.

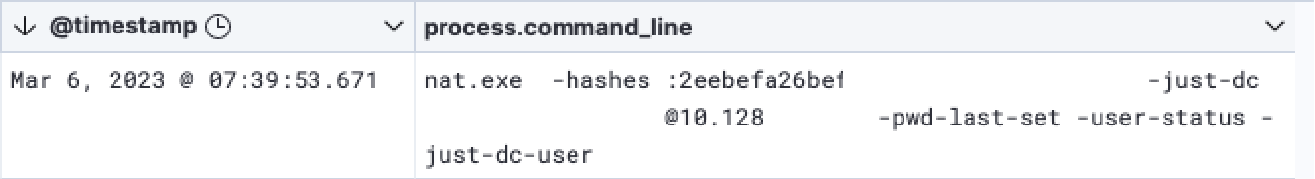
Within the if statement, the method reads the binary data from the request body using the BinaryRead method and stores it in a byte array. It then creates a string containing the fully qualified name of a .NET type that the code wants to load and gets a reference to that type using the Type.GetType method. The byte array in the image is the ASCII code representation of the text “ System.Reflection.Assembly ”. This way of presenting the code is done in order to avoid string-based detection. The System.Reflection.Assembly class provides methods and properties to load, examine, and manipulate assemblies at runtime.

The code obtains a reference to a method named Load in the loaded type and invokes it using the Invoke method. The Load method takes a byte array as a parameter, which the code decrypts using a Decrypt method (not shown in this publication). The result of the Load method invocation is stored in an object variable.

The code then gets a reference to another method named CreateInstance in the loaded type and invokes it using the Invoke method. The CreateInstance method takes a string as a parameter, which the code constructs from a byte array containing the ASCII codes for the string U. The result of the CreateInstance method invocation is stored in an object variable.

Finally, the code calls the Equals method on the object, passing in the current object. Because Equals will call GetType on the object, this approach is a way to indirectly call functions covertly.

The Encrypt and Decrypt functions include a hard-coded key.



| @timestamp | process.command_line |
|---|---|
| Mar 6, 2023 @ 07:39:53.671 | nat.exe  -hashes :2eebefa26bef  -just-dc  @10.128  -pwd-last-set -user-status -just-dc-user |

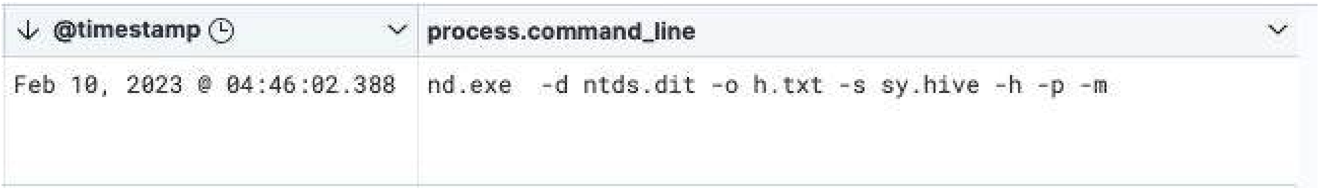
## Sources

The key " e45e329feb5d925b" is the result of taking the first half of the MD5 hash of the string "rebeyond". The string “rebeyond” refers to the developer of the Behinder web shell framework. This refers to the developer of the Behinder webshell framework. This key is also the default value when you generate a shell template using the Behinder or derivative Godzilla webshell frameworks.

## Persistence module

On February 13, 2023, we observed a new persistent malware called `kavUpdate.exe` written in .NET with an exceptionally small footprint (about 6Kb compiled). We believe this software was developed specifically for this environment by the threat. Elastic Security Labs observed this binary persisting via a Scheduled Task, though other mechanisms would likely be compatible.

### Analysis



| @timestamp | process.command_line |
|---|---|
| Feb 10, 2023 @ 04:46:02.388 | nd.exe  -d ntds.dit -o h.txt -s sy.hive -h -p -m |

This code is designed with the sole purpose of executing a set of predefined commands. The malware checks the current day and hour, and if it is Monday or Thursday at 5am, it will execute a series of commands:

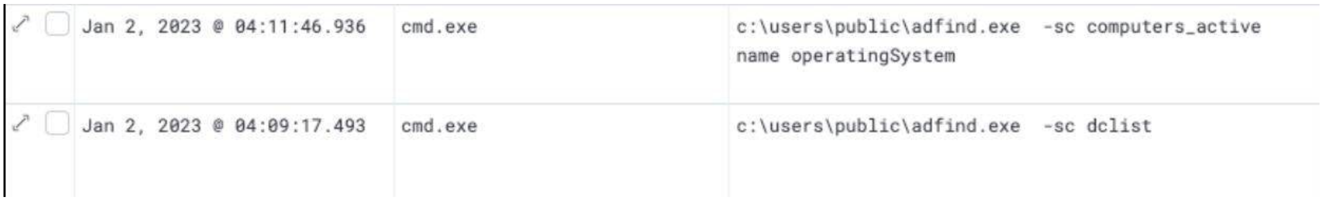
1. Delete the user 'norshasa'
2. Add the user 'norshasa' with the password 'P@ssw0rd123...'
3. Activate the user 'norshasa'
4. Add the user 'norshasa' to the Domain Admins group
5. Add the user 'norshasa' to the Remote Desktop Users group
6. Create a full backup of NTDS in the `C:\ProgramData\temp` folder
7. On the same days of the week, one hour later at 6am, delete the user 'norshasa.'

## Open source tools

On January 2nd, 2023 the threat deployed TFirewall in the victim's environment. TFirewall is a testing tool designed to evaluate whether hosts can establish a SOCKS5 proxy within an intranet environment while allowing for outbound network communication through specific ports. Developed using Golang, TFirewall is comprised of a client and server component and is compatible with multiple operating systems.

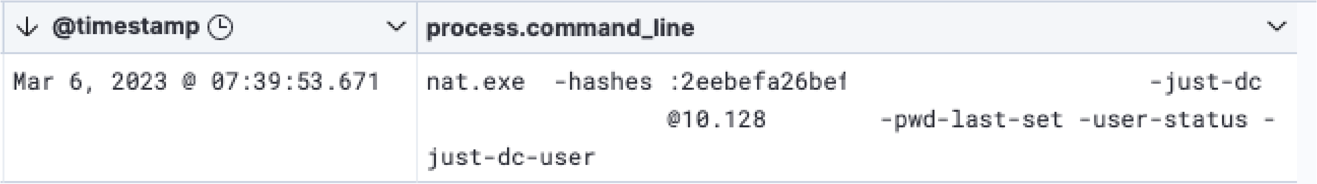
Along with TFirewall, we observed that the attacker used the free tool AdFind. `AdFind` is a command line utility for querying Active Directory and other directory services. AdFind can be run on Windows 7 or newer and requires no special security permissions beyond the ability to launch executables. It’s written in C++ and compiled with Visual Studio 2022. The source code is not available.

The binary is quickly identifiable based on its hash (114b37df703d46a44de0bc96afab8b8590e59a3c389558dd531298e5dd275acb). During execution, we recognized the use of AdFind-specific command line flags and parameters:



```
Jan 2, 2023 @ 04:11:46.936   cmd.exe   c:\users\public\adfind.exe  -sc computers_active
                                       name operatingSystem
Jan 2, 2023 @ 04:09:17.493   cmd.exe   c:\users\public\adfind.exe  -sc dclist
```

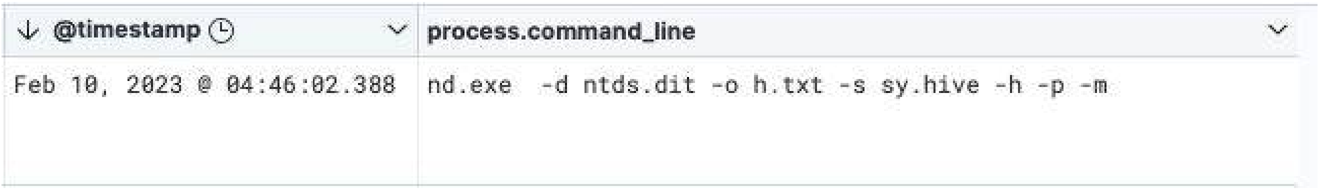
On March 6th, 2023 we observed a process named `nat.exe`. Initially, the file was only identified as generically malicious. However, if we take a closer look at the command line parameters that are used during execution, we have a hint for which tool the attacker is using.



```
@timestamp                   process.command_line
Mar 6, 2023 @ 07:39:53.671   nat.exe  -hashes :2eebefa26bef                     -just-dc
                                              @10.128        -pwd-last-set -user-status -
                             just-dc-user
```

Based on these arguments, we can safely conclude it's a packed version of the Impacket tool secretsdump. Impacket contains a collection of Python classes for working with network protocols. Impacket is commonly used to carry out a variety of tasks related to network security and penetration testing, though it may also be abused by threat actors.

Using the same approach (examining the command line parameters), we identified the use of the tool called NTDSDumpEx which exhibited the same command line arguments employed by this tool:



```
@timestamp                   process.command_line
Feb 10, 2023 @ 04:46:02.388  nd.exe  -d ntds.dit -o h.txt -s sy.hive -h -p -m
```

`NTDSDumpEx` is capable of extracting data from the Active Directory NTDS.dit database in its offline state, meaning the database does not have to be running. It can extract information such as user accounts, group memberships, access control lists, and other directory objects.

## Background

Throughout the attack we witnessed a combination of TTPs that provide a recognizable fingerprint. For example, the way the attacker exported mailboxes is described in detail in this blog post. We also see a strong resemblance in the way credentials from LSASS are being exported, as described here. The majority of the commands and tools deployed by the attacker are well described on the same GitHub users' tips repository.

We also note that the technique used to deploy NAPLISTENER is described here and the deployment method for malicious IIS modules like DOORME can be found in this blog post. And lastly, a post on Godzilla and Behinder web shells in exchange servers closely reflects how these capabilities were implemented within targeted environments.

During malware analysis of the SIESTAGRAPH, NAPLISTENER, and SOMNIRECORD families, we also identified open source repositories that minimally served as the inspiration for these payloads and which have been described in other publications from Elastic Security Labs.

We conclude that the attackers are at the very least regular consumers of blogs and open source repositories, both of which have contributed to the rapid pace of this threat's activities.

## Detection logic

The following prebuilt protections are available from Elastic: - AdFind Command Activity

## YARA

Elastic Security has created YARA rules to identify this activity. Below are YARA rules to identify the Behinder web shell.

```
rule Windows_Trojan_Behinder { meta: author = "Elastic Security" creation_date
= "2023-03-02" last_modified = "2023-03-02" description = "Web shell found in
REF2924, related to Behinder or Godzilla" os = "Windows" arch = "x86"
category_type = "Trojan" family = "Behinder" threat_name =
"Windows.Trojan.Behinder" License = “Elastic License v2” reference_sample =
"a50ca8df4181918fe0636272f31e19815f1b97cce6d871e15e03b0ee0e3da17b" strings:
$load = { 53 79 73 74 65 6D 2E 52 65 66 6C 65 63 74 69 6F 6E 2E 41 73 73 65 6D
62 6C 79 } $key = "e45e329feb5d925b" ascii wide condition: all of them }
```