Blog Home

Defending Against Scheduled Task Attacks in Windows Environments

Qualys
June 20, 2022 - 11 min read

Last updated on: *December 22, 2022*

Table of Contents

Scheduling tasks is one of the most popular attack techniques used by threat actors to establish persistence on a victim's machine. The Qualys Research Team investigated different ways that attackers could use to conceal scheduled tasks. In this blog, we *describe three new techniques to hide and delete scheduled tasks in a Microsoft Windows environment.*

Adversaries abuse task scheduling functionality in Microsoft Windows environments to facilitate initial or recurring execution of malicious code at system startup or on a scheduled basis for persistence. In fact, the MITRE ATT&CK framework lists it as one of the most popular techniques used by threat actors, since the ability to schedule programs or scripts is a common utility across operating systems.

Recently, security researchers at Microsoft published an article that documented how the Chinese state-sponsored group Hafnium concealed scheduled tasks by deleting the Security Descriptor (SD) value within the Windows registry path:

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\TASK_NAME.
```

Following the disclosure by Microsoft, the Qualys Research Team wondered if there are other ways of concealing scheduled tasks and decided to investigate further. The objective of this blog is to communicate our research findings.

Our most important finding is that the `Index` value within the Windows Registry path

```
(HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\TASK_NAME)
```

can also be abused to hide and delete any scheduled task.

First, let's briefly describe the technique used by Hafnium and others to hide a scheduled task. Next, we give a detailed description of new techniques being used to hide a scheduled task in Microsoft environments.

How Threat Actors Hide Scheduled Tasks

According to Microsoft's blog, with the creation of every scheduled task, the following two registry subkeys get created: one within the Tree path and the other within the Tasks path.

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\TASK_NAME
```

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{GUID}
```

The first subkey TASK_NAME, created within the Tree path, matches the name of the scheduled task. The values created within it (i.e. Id, Index, and SD) contain metadata for task registration within the system.

The second subkey {GUID}, created within the Tasks path, matches the Id value in the Tree subkey. The values created within it (i.e. Actions, Path, Triggers, etc.) contain the basic parameters necessary to facilitate the execution of the task.

In the case of Hafnium, the threat actor created a scheduled task named "WinUpdate" to re-establish any dropped connections to their command & control infrastructure. This resulted in the creation of subkeys within the Tree path and Tasks path. Subsequently, the threat actor acquired SYSTEM privileges (via token theft) and deleted the SD value within the Tree subkey. Removal of the SD value resulted in the task "disappearing" from the *Task*

Scheduler app and the output of *schtasks /query* command, thereby concealing the scheduled task from any traditional means of identification.

Our investigation revealed that modifying or deleting the Index value within the Tree subkey also hides scheduled tasks. Now we'll review our findings in more detail, but first a quick description of our lab conditions.

The Qualys Research Team Experimental Setup Environment

Our experiments were conducted on Windows 10 Pro (v10.0.19043), Windows 10 Enterprise (v10.0.19044) and Windows 2016 server. On every machine, we first performed the following two steps:

A. Configure Object Auditing in the Local Security Policy's advanced auditing options to get events corresponding to scheduled task creation (4698), deletion (4699) and updating (4702) in the Windows Security event log.

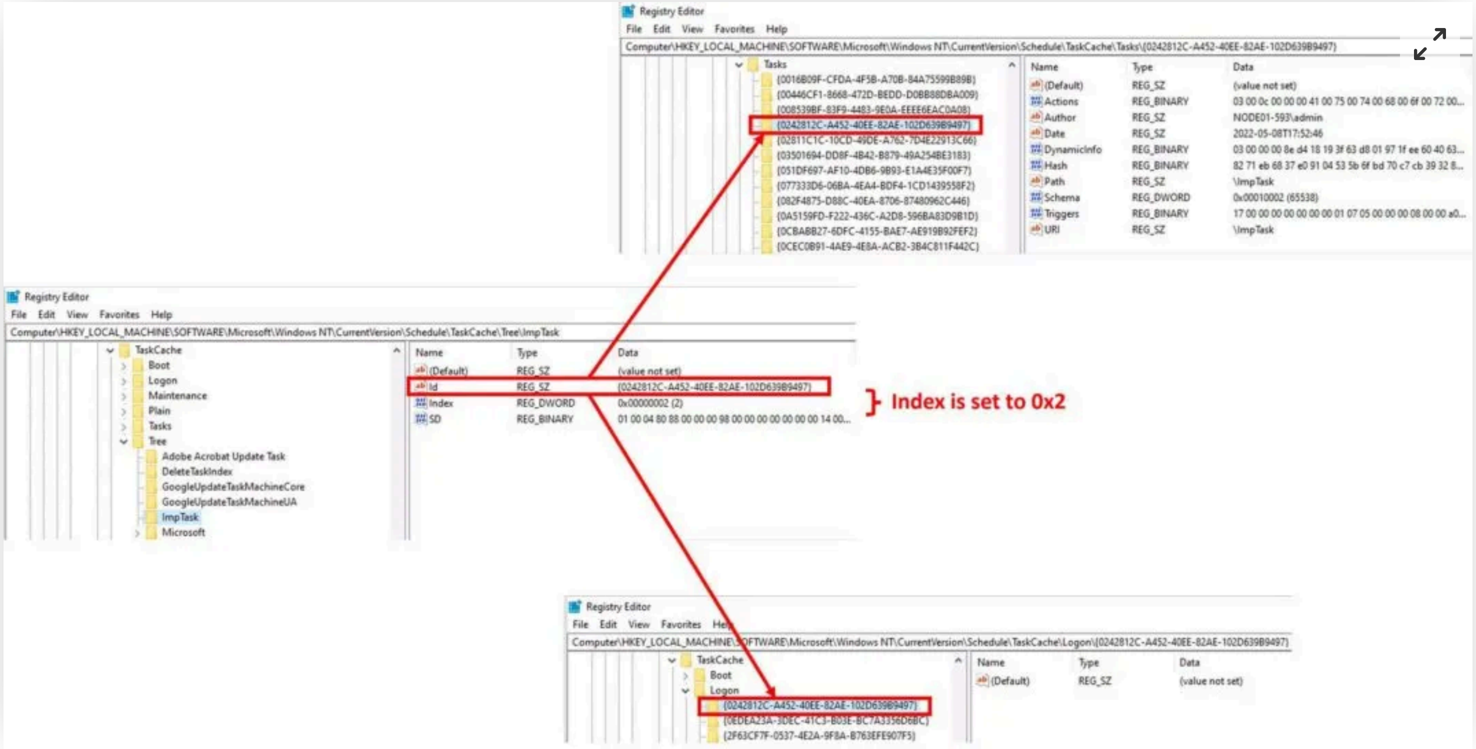
B. Create a scheduled task named *ImpTask* that executes after user login.
```
schtasks /create /tn ImpTask /tr cmd.exe /sc onlogon /rl highest
```

Once the *schtasks /create* command is executed, the following three subkeys corresponding to the newly created ImpTask are created (refer Figure 1).

1. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\ImpTask`
2. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{GUID}`

3. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Logon\{GUID}`

The Index value within ImpTask subkey is set to 0x2 (see Figure 1) as the {GUID} subkey for this task is created within the Logon path (since the task is scheduled to run after user login).



```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\{0242812C-A452-40EE-82AE-102D639B9497}
```

```
Tasks
  {0016B09F-CFDA-4F5B-A70B-84A75599B89B}
  {00446CF1-8668-472D-BEDD-D0BB88DBA009}
  {008539BF-83F9-4483-9E0A-EEEE6EAC0A08}
  {0242812C-A452-40EE-82AE-102D639B9497}
  {02811C1C-10CD-49DE-A762-7D4E22913C66}
  {03501694-DD8F-4B42-B879-49A254BE3183}
  {051DF697-AF10-4DB6-9B93-E1A4E35F00F7}
  {077333D6-06BA-4EA4-BDF4-1CD1439558F2}
  {082F4875-D88C-40EA-8706-87480962C446}
  {0A5159FD-F222-436C-A2D8-596BA83D9B1D}
  {0CBABB27-6DFC-4155-BAE7-AE919B92FEF2}
  {0CEC0B91-4AE9-4E8A-ACB2-3B4C811F442C}
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
Actions	REG_BINARY	03 00 0c 00 00 00 41 00 75 00 74 00 68 00 6f 00 72 00...
Author	REG_SZ	NODE01-593\admin
Date	REG_SZ	2022-05-08T17:52:46
DynamicInfo	REG_BINARY	03 00 00 00 8e d4 18 19 3f 63 d8 01 97 1f ee 60 40 63...
Hash	REG_BINARY	82 71 eb 68 37 e0 91 04 53 5b 6f bd 70 c7 cb 39 32 8...
Path	REG_SZ	\ImpTask
Schema	REG_DWORD	0x00010002 (65538)
Triggers	REG_BINARY	17 00 00 00 00 00 00 00 01 07 05 00 00 00 08 00 00 a0...
URI	REG_SZ	\ImpTask

```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\ImpTask
```

```
TaskCache
  Boot
  Logon
  Maintenance
  Plain
  Tasks
  Tree
    Adobe Acrobat Update Task
    DeleteTaskIndex
    GoogleUpdateTaskMachineCore
    GoogleUpdateTaskMachineUA
    ImpTask
    Microsoft
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
Id	REG_SZ	{0242812C-A452-40EE-82AE-102D639B9497}
Index	REG_DWORD	0x00000002 (2)
SD	REG_BINARY	01 00 04 80 88 00 00 00 98 00 00 00 00 00 00 00 14 00...

} Index is set to 0x2

```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Logon\{0242812C-A452-40EE-82AE-102D639B9497}
```

```
TaskCache
  Boot
  Logon
    {0242812C-A452-40EE-82AE-102D639B9497}
    {0EDEA23A-3DEC-41C3-B03E-BC7A3356D6BC}
    {2F63CF7F-0537-4E2A-9F8A-B763EFE907F5}
```

Name	Type	Data
(Default)	REG_SZ	(value not set)

Figure 1. Three registry keys associated with the scheduled task ImpTask.

New Methods to Hide a Scheduled Task

We observed that, when a scheduled task is created, in addition to the Tree and Tasks subkeys, one more subkey gets created. This third subkey is created depending on whether the task is scheduled to run:

- At startup, as indicated by */sc onstart* parameter in *schtasks /create* command

- During user logon, as indicated by */sc onlogon* parameter *in schtasks /create* command
- At a time, other than boot up or logon (e.g., */sc daily /st 09:00*)

The third subkey is created within one of the following paths:

1. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Boot\{GUID}`
2. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Logon\{GUID}`
3. `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Plain\{GUID}`

The name of the third subkey {GUID} matches with the Id value found in the Tree subkey. We further observed that the Index value within the Tree subkey is also related to this third subkey associated with the scheduled task. We found that the Index value is set to either 0x1 or 0x2 or 0x3. Specifically,

1. All tasks registered within the path `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Boot` have an Index value of 0x1
2. All tasks registered within the path `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Logon` have an Index value of 0x2
3. All tasks registered within the path `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\(Plain or Maintenance)` have an Index value of 0x3

The Qualys Research Team wrote a Python script and ran it across different Windows machines to confirm this behavior. Because every scheduled task is a part of either Boot or Logon or Plain or Maintenance, so there seems to be only three possible values for Index: 0x1, 0x2, or 0x3. Our investigation did not find any online documentation describing the purpose of the Index value associated with the scheduled task. However, we were able to manipulate the Index value to obtain the following outcomes.

1. **Hide a specific scheduled task:** We found that setting the Index value to 0x0 within the Tree subkey hides the task from the *Task Scheduler* app and the output of *schtasks /query*. However, the task continues to run as per its scheduled time, even across system restarts. The resulting behavior is exactly the same as what Hafnium threat actors achieved after deleting the SD value. Further, if we try to modify the task after its Index value is set to 0x0 using *schtasks /change* command, the task gets deleted. However, the event id 4699 corresponding to the scheduled task deletion does not get reported to the Windows Security Event log.
2. **Hide all scheduled tasks:** We also found that deletion of the Index value causes the *Task Scheduler* app and *schtasks /query* to fail with an error message “Internal error occurred” that effectively hides all scheduled tasks. The existing tasks continue to run and new tasks can still be created.

Setting the index to any other value (0x4, 0xffff etc.), does not hide the scheduled task and the task continues to run as scheduled.

Now let’s examine the two outcomes when the index value was manipulated.

Hide Scheduled Task

In this first scenario, we create another scheduled task named *ModifyIndexTask* that executes once with SYSTEM privileges – after the creation of ImpTask –

and set its Index value to 0x0. The command is as follows:

```
schtasks /create /tn ModifyIndexTask /tr "reg.exe add
\"HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\ImpTask\" /v Index /d 0x0 /t REG_DWORD
/f" /ru "NT AUTHORITY\SYSTEM" /rl highest /sc once /st <time later than creation
time of ImpTask>
```

Figure 2. The Index value of ImpTask is modified to 0x0.

Once ModifyIndexTask is executed, it sets the Index value of ImpTask to 0 (Figure 2). As a result, ImpTask disappears from both the *Task Scheduler* app(Figure 3) and the output of *schtasks /query* command (Figure 4). However, ImpTask continues to run even after the system restarts (Figure 5). Although ImpTask does not appear in the output of *schtasks /query* command, Figure 5 shows that it is possible to get the status of the task using *schtasks /query* command by specifying the task name using the parameter */tn*.

Figure 3. ImpTask disappears from the *Task Scheduler* app once Index value is modified to 0x0.

Figure 4. ImpTask disappears from output of *schtasks /query* once Index value is modified to 0x0.

Figure 5. After setting Index value to 0x0, ImpTask continues to run.

The Qualys Research Team was able to reproduce this issue on every Windows 10 machine that we experimented with, which was across a total of five boxes.

Another interesting observation was that, if we try to change the program name within ImpTask (with Index value 0x0) using *schtasks /change /tr* command, the task gets deleted as shown in Figure 6. It is executed without reporting event id 4699: scheduled task deletion, or event id 4702: scheduled task update, in the Windows Security Event log. However, event id 4699 *is* reported if we use *schtasks /delete* command to delete ImpTask.

Figure 6. Deletion of ImpTask using *schtasks /change /tr leaves* no trace in Windows Security log.

Hide All Scheduled Tasks

In this second first scenario, we create another scheduled task that executes with SYSTEM privileges and deletes the Index value within the ImpTask subkey. The command is as follows:

```
schtasks /create /tn ModifyIndexTask /tr “reg.exe delete \”HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\ImpTask\” /v Index /f” /ru “NT AUTHORITY\SYSTEM” /rl highest /sc once /st <time later than creation time of ImpTask>
```

Once the Index value within the ImpTask subkey is deleted (Figure 7), all scheduled tasks disappear from the *Task Scheduler* app (Figure 8) and the output of *schtasks /query* command (Figure 9), instead we get an error message saying, “An internal error occurred”. Even specifying the task name ImpTask within the parameter */tn* also doesn’t work.

Figure 7. Index value deleted from the ImpTask subkey.

Figure 8. After Index value deletion, all scheduled tasks disappear from Task scheduler app, and error message displayed.

Figure 9. Specifying the task name using schtasks /query /tn command also doesn’t work.

Although scheduled tasks are not displayed, they are executed as per their scheduled time. The inability to view scheduled tasks persists even after the system reboots. Modifying ImpTask using *schtasks /change* command causes the Index value to be generated again, after which the execution of *schtasks /query* command is successful (Figure 10).

Figure 10. Modifying ImpTask using schtasks /change command causes Index value to be generated again, following which the execution of schtasks /query command is successful.

After deleting the Index value, we tried to delete the ImpTask using *schtasks /delete*. Interestingly, the command failed with an error message. When we next tried to change ImpTask using *schtasks /change* command, the Index value within the ImpTask subkey was restored. All tasks reappeared in the *Task Scheduler* app and the execution of *schtasks /query* was also successful. Note that the Index value is restored only when *schtasks /delete* precedes *schtasks /change*. When we executed *schtasks /change* without first running *schtasks /delete*, the Index value was not restored, and we continued to get an error message on executing *schtasks /query*.

Conclusion

An investigation by Qualys Research Team found that the Index value along with the SD value within the Tree subkey of a scheduled task plays an important role and both can be abused by attackers. In this blog, we described three new techniques to hide and delete scheduled tasks:

1. Hide a scheduled task from the *Task Scheduler* app and the output of *schtasks /query* command by setting its Index value to 0x0
2. Delete a scheduled task by first setting its Index value to 0x0 and then using *schtasks /change /tr* command which effectively deletes the task without leaving any trace in the Windows Security Event log
3. Hide all scheduled tasks from the *Task Scheduler* app and the output of *schtasks /query* command by deleting the Index value of any scheduled task

Any of these new techniques can be used to hide a scheduled task in Microsoft environments. Therefore, it is important to monitor modifications to both Index and SD values of scheduled tasks. These changes could alert on the facilitation of malicious code execution either at system startup or on a scheduled basis for persistence.

Contributors

Mayuresh Dani, Threat Research Manager, Qualys
mdani@qualys.com

Written by
Qualys
Write to Qualys at webmaster@qualys.com

Like Share

Related content
Microsoft, scheduled task

SHARE YOUR COMMENTS

Join the discussion today!

Learn more about Qualys and industry best practices.

Share what you know and build a reputation.

Secure your systems and improve security for everyone.

Start a discussion

Qualys

Qualys.com

Qualys Community Edition

Qualys Merchandise Store

Qualys Communities

Vulnerability Management

Policy Compliance

PCI Compliance

Web App Scanning

Web App Firewall

Continuous Monitoring

Security Assessment Questionnaire

Threat Protection

Asset Inventory

AssetView

CMDB Sync

Endpoint Detection & Response

Security Configuration Assessment

File Integrity Monitoring

Cloud Inventory

Certificate Inventory

Container Security

Cloud Security Assessment

Certificate Assessment

Out-of-band Configuration Assessment

Patch Management

Developer API

Cloud Agent

Dashboards & Reporting

Discussions

All discussions

Global IT Asset Management

IT Security

Compliance

Cloud & Container Security

Web App Security

Certificate Security & SSL Labs

Developer API

Training

Overview

Certified Courses

Video Library

Instructor-led Training

Docs

Overview

Release Notes

Blog

All posts

Qualys Insights

Product and Tech

Vulnerabilities and Threat Research

Release Notifications

Support

Support Portal