Threat Hunter Playbook

Search this book...

Remote DCOM IErtUtil DLL Hijack

Hypothesis

Threat actors might be copying files remotely to abuse a DLL hijack opportunity found on the DCOM InternetExplorer.Application Class.

Technical Context

Offensive Tradecraft

A threat actor could use a known DLL hijack vulnerability on the DCOM InternetExplorer.Application Class while instantiating the object remotely. When the object instantiate, it looks for `iertutil.dll` in the `c:\Program Files\Internet Explorer\` directory. That DLL does not exist in that folder. Therefore, a threat actor could easily copy its own DLL in that folder and execute it by instantiating an object via the DCOM InternetExplorer.Application Class remotely. When the malicious DLL is loaded, there are various approaches to hijacking execution, but most likely a threat actor would want the DLL to act as a proxy to the real DLL to minimize the chances of interrupting normal operations. One way to do this is by cloning the export table from one DLL to another one. One known tool that can help with it is Koppeling.

Pre-Recorded Security Datasets

Metadata	Value
docs	https://securitydatasets.com/notebooks/atomic/windows/lateral_movement/SDWIN-201009183000.html
link	https://raw.githubusercontent.com/OTRF/Security-Datasets/master/datasets/atomic/windows/lateral_movement/host/covenant_dcom_iertutil_dll_hijack.zip

Download Dataset

```
import requests
from zipfile import ZipFile
from io import BytesIO

url = 'https://raw.githubusercontent.com/OTRF/Security-Datasets/master/dat
zipFileRequest = requests.get(url)
zipFile = ZipFile(BytesIO(zipFileRequest.content))
datasetJSONPath = zipFile.extract(zipFile.namelist()[0])
```

Read Dataset

```
import pandas as pd
from pandas.io import json

df = json.read_json(path_or_buf=datasetJSONPath, lines=True)
```

Analytics

A few initial ideas to explore your data and validate your detection logic:

Analytic I

Look for non-system accounts SMB accessing a `C:\Program Files\Internet Explorer\iertutil.dll` with write (0x2) access mask via an administrative share (i.e C$).

Data source	Event Provider	Relationship	Event
File	Microsoft-Windows-Security-Auditing	User accessed File	5145

Logic

```
SELECT `@timestamp`, Hostname, ShareName, SubjectUserName, SubjectLogonId,
FROM dataTable
WHERE LOWER(Channel) = "security"
    AND EventID = 5145
    AND RelativeTargetName LIKE '%Internet Explorer\\iertutil.dll'
    AND NOT SubjectUserName LIKE '%$'
    AND AccessMask = '0x2'
```

Pandas Query

```
(
df[['@timestamp','Hostname','ShareName','SubjectUserName','SubjectLogonId'

[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 5145)
    & (df['RelativeTargetName'].str.lower().str.endswith('internet explore
    & (df['AccessMask'] == '0x2')
    & (~df['SubjectUserName'].str.endswith('$', na=False))
]
.head()
)
```

Analytic II

Look for `C:\Program Files\Internet Explorer\iertutil.dll` being accessed over the network with write (0x2) access mask via an administrative share (i.e C$) and created by the System process on the target system.

Data source	Event Provider	Relationship	Event
File	Microsoft-Windows-Security-Auditing	User accessed File	5145
File	Microsoft-Windows-Sysmon/Operational	Process created File	11

Logic

```
SELECT `@timestamp`, Hostname, ShareName, SubjectUserName, SubjectLogonId,
FROM dataTable b
INNER JOIN (
    SELECT LOWER(REVERSE(SPLIT(TargetFilename, '\'))[0]) as TargetFilename
    FROM dataTable
    WHERE Channel = 'Microsoft-Windows-Sysmon/Operational'
        AND Image = 'System'
        AND EventID = 11
        AND TargetFilename LIKE '%Internet Explorer\\iertutil.dll'
) a
ON LOWER(REVERSE(SPLIT(RelativeTargetName, '\'))[0]) = a.TargetFilename
WHERE LOWER(b.Channel) = 'security'
    AND b.EventID = 5145
    AND b.AccessMask = '0x2'
```

Pandas Query

```
fileCreateDf = (
df[['@timestamp','Hostname','Image','TargetFilename']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 11)
    & (df['Image'].str.lower() == 'system')
    & (df['TargetFilename'].str.lower().str.endswith('internet explorer\\i
]
)

fileCreateDf['Filename'] = fileCreateDf['TargetFilename'].str.split('\\').

fileAccessedDf = (
df[['@timestamp','Hostname','ShareName','SubjectUserName','SubjectLogonId'
[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 5145)
    & (df['AccessMask'] == '0x2')
]
)

fileAccessedDf['Filename'] = fileAccessedDf['RelativeTargetName'].str.spli

(
pd.merge(fileCreateDf, fileAccessedDf,
    on = 'Filename', how = 'inner')
)
```

Analytic III

Look for `C:\Program Files\Internet Explorer\iertutil.dll` being accessed over the network with write (0x2) access mask via an administrative share (i.e C$), created by the System process and loaded by `C:\Program Files\Internet Explorer\iexplore.exe`. All happening on the target system.

Data source	Event Provider	Relationship	Event
File	Microsoft-Windows-Security-Auditing	User accessed File	5145
File	Microsoft-Windows-Sysmon/Operational	Process created File	11
File	Microsoft-Windows-Sysmon/Operational	Process loaded Dll	7

Logic

```
SELECT `@timestamp`, Hostname, ShareName, SubjectUserName, SubjectLogonId,
FROM dataTable d
INNER JOIN (
    SELECT LOWER(REVERSE(SPLIT(TargetFilename, '\'))[0]) as TargetFilename
    FROM dataTable b
    INNER JOIN (
        SELECT ImageLoaded
        FROM dataTable
        WHERE Channel = 'Microsoft-Windows-Sysmon/Operational'
            AND EventID = 7
            AND LOWER(Image) LIKE '%iexplore.exe'
            AND ImageLoaded LIKE '%Internet Explorer\\iertutil.dll'
    ) a
    ON b.TargetFilename = a.ImageLoaded
    WHERE b.Channel = 'Microsoft-Windows-Sysmon/Operational'
        AND b.Image = 'System'
        AND b.EventID = 11
) c
ON LOWER(REVERSE(SPLIT(RelativeTargetName, '\'))[0]) = c.TargetFilename
WHERE LOWER(d.Channel) = 'security'
    AND d.EventID = 5145
    AND d.AccessMask = '0x2'
```

Pandas Query

```
fileCreateDf = (
df[['@timestamp','Hostname','Image','TargetFilename']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 11)
    & (df['Image'].str.lower() == 'system')
    & (df['TargetFilename'].str.lower().str.endswith('internet explorer\\i
]
)

imageLoadedDf = (
df[['@timestamp','Hostname','ImageLoaded','Image']]

[(df['Channel'] == 'Microsoft-Windows-Sysmon/Operational')
    & (df['EventID'] == 7)
    & (df['Image'].str.lower().str.endswith('iexplore.exe', na=False))
    & (df['ImageLoaded'].str.lower().str.endswith('internet explorer\\iert
]
)

firstJoinDf = (
pd.merge(fileCreateDf, imageLoadedDf,
    left_on = 'TargetFilename', right_on = 'ImageLoaded', how = 'inner')
```

```
)

firstJoinDf['Filename'] = firstJoinDf['TargetFilename'].str.split('\\').st

fileAccessedDf = (
df[['@timestamp','Hostname','ShareName','SubjectUserName','SubjectLogonId'

[(df['Channel'].str.lower() == 'security')
    & (df['EventID'] == 5145)
    & (df['AccessMask'] == '0x2')
]
)

fileAccessedDf['Filename'] = fileAccessedDf['RelativeTargetName'].str.spli

(
pd.merge(firstJoinDf, fileAccessedDf,
    on = 'Filename', how = 'inner')
)
```

Known Bypasses

False Positives

Hunter Notes

- Baseline your environment to identify normal activity. Document all accounts creating files over the network via administrative shares.
- Baseline iexplore.exe execution and modules loaded (i.e signed and un-signed)

Hunt Output

Type	Link
Sigma Rule	https://github.com/SigmaHQ/sigma/blob/master/rules/windows/builtin/security/win_dcom_iertutil_dll_hijack.yml
Sigma Rule	https://github.com/SigmaHQ/sigma/blob/master/rules/windows/sysmon/sysmon_dcom_iertutil_dll_hijack.yml

References

- https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-dcom/64af4c57-5466-4fdf-9761-753ea926a494

Previous
Remote WMI ActiveScriptEventConsumers

Next
Remote WMI Wbemcomn DLL Hijack