Abusing cloud apps 101: Command and Control

Dagmawi Mulugeta
Whoami

Currently a Threat researcher @ Netskope

Previously
- Researcher @ Cyrisk
- Software Engineer @ Sift Security
- Developer @ ECFMG

MSc in Cybersecurity from Drexel University

Interests: CTFs, exploit development, and cloud apps
Outline of takeaways

What is Cloud C2?

Which cloud apps are abused for C2 in the wild?

How can you simulate this technique in your corporate networks?

What defences can be put in place?
What is Cloud C2?
Command and Control

Stage in the Cyber Kill Chain

Traditionally, involves a compromised device polling a server for commands

Via mediums like HTTPs and DNS directly to an attacker controlled server

Example frameworks include Cobalt Strike and PowerShell Empire

Cloud Command and Control (Cloud C2)

**Traditional C2**
- Attackers setting up their own servers, domains, etc.
- Tough to detect, but can be identified via IP / domain blocklists

**Cloud C2**
- (ab)Use a cloud applications as a command and control channel
- Very minimal setup
- Even tougher to detect since traffic blends in with normal app usage
Abusing Web services for Command and Control (T1102)

Contains three sub-techniques

Sub-technique 1) Bidirectional Communication (Primary focus for this talk)
- Send commands to and receive output from a compromised system over the cloud app
Sub-technique 2) **One-Way Communication**
- Send commands to **without receiving** output from a compromised system over the cloud app

Sub-technique 3) **Dead Drop Resolver**
- Abuse the cloud app to **host information that points to additional C2** infrastructure; victims will reach out and be redirected by these resolvers

Source: https://blog.bushidotoken.net/2021/04/dead-drop-resolvers-espionage-inspired.html
Abused cloud apps
Which cloud apps are abused?

Some examples of malware and cloud apps they abuse:

- BoxCaon, Nimble Mamba, and Crutch have used DropBox for C2 communications
- Graphite and BLUELIGHT abuse OneDrive for C2
- Aclip abused messenger application Slack’s API for C2
- BLACKCOFFEE and Lazarus abused Github to obfuscate its C2 traffic
- Pawn Storm abuses Google Drive via a RAT
- CozyCar and ROKRAT abuse Twitter as a main and backup C2 channel
- Comnie uses Tumblr and BlogSpot to mask C2 traffic
- FIN7 used services like Google Docs, Google Scripts, and Pastebin for C2
- MuddyWater abused OneHub to distribute remote access tools
- Sandworm abused the Telegram Bot API to send and receive commands

A more detailed list can be found on MITRE's page
Which cloud apps are abused?

- 23 known and reported instances of Cloud C2 in the April 2020 to April 2022 time period
Which cloud apps are abused?

- Apps in Cloud Storage category tend to be preferred by attackers
### Features abused

<table>
<thead>
<tr>
<th>Category</th>
<th>Example apps</th>
<th>Abused features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Storage</td>
<td>Dropbox, OneDrive</td>
<td>Upload file, Download file, Delete file</td>
</tr>
<tr>
<td>Social</td>
<td>Telegram</td>
<td>Bot usage, Read message, Write message</td>
</tr>
<tr>
<td>Collaboration</td>
<td>Slack</td>
<td>Create Channels, Read message, Write message, Reply to message</td>
</tr>
<tr>
<td>CDN</td>
<td>CloudFront</td>
<td>Proxy traffic via a CDN network</td>
</tr>
<tr>
<td>Web Design</td>
<td>Google Sites</td>
<td>DGA to pass data to different URL everyday</td>
</tr>
<tr>
<td>PaaS</td>
<td>Glitch</td>
<td>Create custom applications, Upload files</td>
</tr>
<tr>
<td>Development Tools</td>
<td>GitHub</td>
<td>Create a repository, Add commits, Delete commits</td>
</tr>
<tr>
<td>Webmail</td>
<td>Google Mail</td>
<td>Write drafts, Write emails, Attach task results as documents</td>
</tr>
</tbody>
</table>
Taking a closer look
Taking a closer look

Empire + Dropbox

- Empire is a PowerShell and Python 3 post-exploitation framework

We will look at

- What the cloud application is meant to do?
- Why attackers might prefer this cloud application?
- A real world example of abuse
- A detailed walk through of how to simulate this technique in red team engagements
- Behind the scenes of how the cloud app is abused
Background

- Category is Cloud Storage

- Tend to be abused by uploading, downloading, deleting encrypted / encoded files

- Flexible app development interface and very easy to get setup

- Exist as both an enterprise and personal cloud further enabling stealth for the attacker
Real world example

**Molerats abuses Dropbox for command-and-control** - January 2022

- Threat actor known for stealth
- Have previously used this TTP
- Used multiple (5+) accounts

Source: https://thehackernews.com/2022/01/molerats-hackers-hiding-new-espionage.html
Attack Simulation using DropBox

- Open source tools: DropboxC2, C3, Empire

- We will use Empire below

- Steps
  - Step 1) Create an attacker controlled Dropbox account
  - Step 2) Setup an Empire listener with an access token from the Dropbox account
  - Step 3) Generate a malicious payload and deliver it to the victim machine to simulate a compromise
  - Step 4) Interact with the compromised device by tunneling commands through the Dropbox account
Attack Simulation using Dropbox

- Step 1: Create an attacker controlled Dropbox account with an access token
Attack Simulation using DropBox

- Step 2: Setup an Empire listener with an access token from the Dropbox account.

```plaintext
Empire: uselistener/dbx > execute
  $ listener dropbox successfully started

Empire: uselistener/dbx >
```
Attack Simulation using Dropbox

- Result from Step 2: This will create a folder to be used for the C2 channel in the attacker’s Dropbox account.
Attack Simulation using DropBox

- Step 3: Deliver the payload to the victim machine to simulate a compromise
Attack Simulation using DropBox

- Step 4: Interact with the compromised device
Attack Simulation using DropBox

- Results from Step 5: Interaction by uploading / downloading encrypted task files

```python
def aes_encrypt(key, data):
    """
    Generate a random IV and new AES cipher object with the given key, and return IV + encryptedData.
    """
    if isinstance(key, str):
        key = bytes(key, "UTF-8")
    if isinstance(data, str):
        data = bytes(data, "UTF-8")
    backend = default_backend()
    IV = os.urandom(16)
    cipher = Cipher(algorithms.AES(key), modes.CBC(IV), backend=backend)
    encryptor = cipher.encryptor()
    ct = encryptor.update(pad(data)) + encryptor.finalize()
    return IV + ct
```
DropBox for C2: summary

- Cloud Storage app abused by uploading, downloading, and deleting encrypted files
  - Similar to OneDrive and Google Drive

- Can simulate a threat actor using this technique using tools like DropboxC2, C3, and Empire

- TTPs based on sophistication
  - Low / unsophisticated:
    - default configurations using a tool like Empire / C3

  - Medium:
    - custom configuration with a tool like Empire / C3

  - High (targeted attacks in the real world)
    - multiple accounts with data transfer distributed among them
Defences
Abusing cloud apps 101: Command and Control

Why is this hard to detect?

<table>
<thead>
<tr>
<th>Benign</th>
<th>Cloud C2</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET githubdesktop:5892</td>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET relay_x64_c691_victi...</td>
</tr>
<tr>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET githubdesktop:5892</td>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET relay_x64_c691_victi...</td>
</tr>
<tr>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET githubdesktop:5892</td>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET relay_x64_c691_victi...</td>
</tr>
<tr>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET githubdesktop:5892</td>
<td><img src="https://api.github.com/repos/..." alt="Request" /> HTTP/1.1 GET relay_x64_c691_victi...</td>
</tr>
</tbody>
</table>

1. Both malicious and benign traffic is going to the same domain
2. The domain is a valid cloud provider domain
3. The traffic to the domain is encrypted using the cloud provider’s certificate
Defences

Identify and block known malware [Endpoint] [Network]

Disallow non corporate cloud applications [Endpoint] [Network]

Adequate logging and monitoring - OWASP Top 10 [Endpoint] [Network] [29]

Regular beaconing behaviour [Network]
Defences

Unsigned applications making network connections [Endpoint]

User agents associated with known malware [Network]

Packet count anomalies [Network][1]

Detect data exfiltration over the C2 channel [Network][1]
Conclusion

What is Cloud C2? *Command and Control via a Cloud Application*

Which apps are abused for C2? *Vast majority of them can be abused*

How can you simulate this in your network? *Four steps when used with Empire/C3/Covenant*

What defences can be put in place? *Numerous controls can aid to detect Cloud C2*
Contact

Twitter: @dagmulu

Linkedin: dmulugeta

Future updates on our blog
THANK YOU!
References

[5] https://attack.mitre.org/techniques/T1102/
[9] https://sans.org.egnyte.com/dl/4mdnX7iS0V
[18] https://github.com/Coalfire-Research/Slackor
[21] https://github.com/Arno0x/DBC2
[22] https://github.com/FSSecureLABS/C3
[23] https://github.com/3xpl01tc0d3r/Callidus
[26] https://github.com/BC-SECURITY
[27] https://github.com/BC-SECURITY/Empire
[28] https://aws.amazon.com/workspaces/