Andrew Chung

Incident Response - Sofacy

Key evolutions in campaign

Across all four iterations of the campaign, the Sofacy group predominantly used similar tactics and techniques to facilitate their attacks, namely:

- Their reliance on **Spear Phishing Attachment (T1193)** as the initial attack vector.
- **User Execution (T1023)** to run malicious code.
- **Process Discovery (T1057)**, **Screen Capture (T1113)** and **System Information Discovery(T1082)** to gather system-specific information.
- **Remote File Copy (T1105)** and **Standard Application Layer Protocol (T1071)** to interact with and copy files from the C2 server.

However, some key evolutions involve their efforts to avoid being detected. By the last iteration, the Sofacy group had heavily employed **Template Injection (T1221)** and **Multi-Stage Channels (T1104)** to evade static detection as no typical indicators are present until after the malicious payload is fetched and to obfuscate the C2 channel. Likewise, instead of extending the functionality of their **Zebrocy (S0251)** and **Cannon (S0351)** tools, the group opted to focus on delivering the Trojan in variant programming languages in an effort to make detection more difficult. Furthermore, their experimentation with **Standard Application Layer Protocol (T1071)** is also quite interesting with their usage of an email-based C2 communication channel which would be a difficult C2 channel to detect and act against due to encryption and legitimacy of email services.

Datasets/feeds for detection

Detection of	Data Sources
Initial Access	Detonation chamber, Email gateway, File monitoring, Mail server, Network intrusion detection system, Packet capture
Execution	Anti-virus, Authentication logs, DLL monitoring, Email gateway, File monitoring, Loaded DLLs, Netflow/Enclave netflow, Network intrusion detection system, PowerShell logs, Process command-line parameters, Process monitoring, Web logs, Windows event logs.
Defense Evasion	Binary file metadata, File monitoring, Process command-line parameters, Process monitoring.
Discovery	API monitoring, AWS CloudTrail logs, Azure activity logs, PowerShell logs, Process command-line parameters, Process monitoring, Stackdriver logs.
Collection	API monitoring, Data loss prevention, File monitoring, Process command-line parameters, Process monitoring.
Command and Control	DNS records, File monitoring, Host network interface, Netflow/Enclave netflow, Network device logs, Network protocol analysis, Packet capture, Process command-line parameters, Process monitoring, Process use of network, SSL/TLS inspection.
Exfiltration	Netflow/Enclave netflow, Packet capture, Process monitoring, Process use of network.

Static correlation vs User/Entity Behaviour Analysis

The challenge with static correlation lies in the fact that due to the sheer amount of logs generated, SOCs are inundated with noise and false alerts which consequently make detection of adversaries difficult. Using both static correlation and U/EBA aims to overcome these limitations and reduce false positives, helping to eliminate alert fatigue and allowing focus on credible, high-alert risks.

Examples of static correlation rules:

Trigger an alert if:

- A malicious event is detected once an attachment is opened. (such as a Microsoft Word document reaching out to the internet or spawning Powershell.exe)
- PowerShell/change in policy to run PowerShell is executed.
- Command-line arguments for actions that could be taken to gather system and network information are executed. (e.g. tasklist, systeminfo, wmic)
- Changes are made to the registry that do not correlate to known software, patch cycles, etc.
- An unusual process performs sequential file opens and copy actions to another location on the file system for many files at once.
- Data flow is uncommon. (e.g. significantly more data sent than data received)

For U/EBA, we would be interested in defining behaviours akin to those that trigger the alerts listed above (e.g. Trigger an alert if a user executes PowerShell/change in policy to run PowerShell). Furthermore, although privileged users are the key population of interest within the environment, non-privileged users should not be discounted, as standard accounts are often escalated in privileges.

In regards to responding to these rules, a fair share of these rules are tailored to the Sofacy attacks and can be seen as somewhat effective to respond to in the sense that they mostly trigger on suspect circumstances specific to the nature of the Sofacy attacks. However, some of these rules (e.g. PowerShell and Windows Management Instrumentation) lose their effectiveness if these activities are already commonly used in an environment. In these

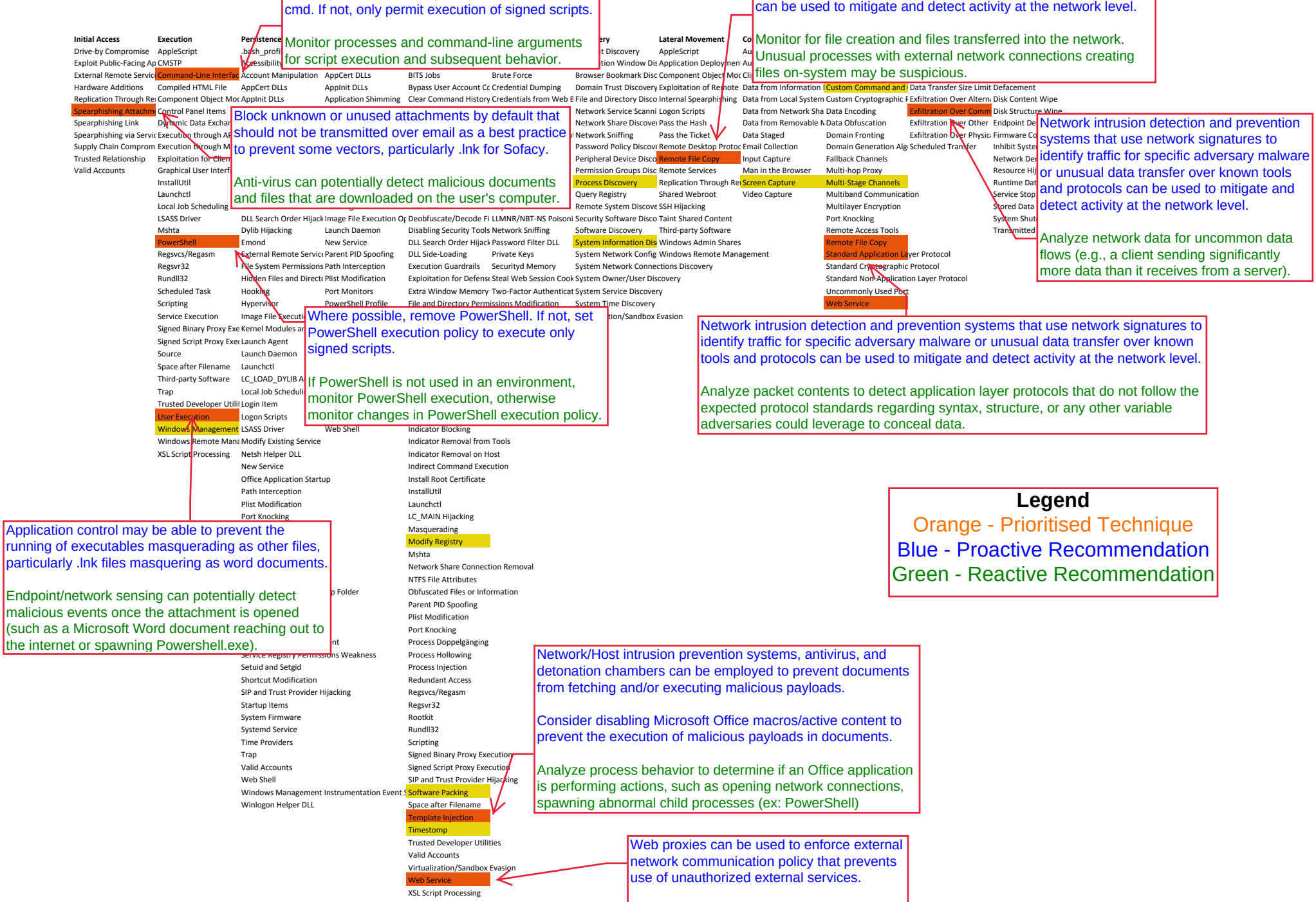
circumstances, these rules should be seen as something that influences a larger data model and to increase confidence of malicious activity, data and events should not be viewed in isolation but as part of a chain of behaviour that could lead to other activities.

As for the interaction between static correlation and U/EBA, in our case, we could administer static correlation rules for prioritised techniques employed by the Sofacy group and leverage U/EBA to detect unknown behaviour unable to be detected by static correlation such as new advances in the Sofacy campaign. By combining both of these approaches into a hybrid analytic we are provided insight into patterns of behaviour and an additional context around known and unknown threats, in conjunction with a more accurate identification of threats.

Mitigation

In order of priority,

1. Prevent initial access into your network. The Sofacy group has a heavy reliance on **Spear Phishing Attachment (T1193)** as the initial attack vector. In the best case, initial access would be thwarted and there would be no subsequent activities.
2. Prevent the execution of malicious code. Likewise, the Sofacy group has a heavy reliance on **User Execution (T1023)** and the **Command and Scripting Interpreter (T1059)** to run malicious code, in which both play a key part in allowing the attack to progress. Preventing execution can thwart further activities.
3. Mitigate the impact of **Zebrocy (S0251).** The Trojan is a routine key tool in the Sofacy attack for the collection and exfiltration of data. As it's unfeasible to prevent its execution due to its dynamic nature, the next best course of action would be to downplay its impact either by preventing its capabilities or allowing early detection.


Where possible, disable/prevent the execution of cmd. If not, only permit execution of signed scripts.
Monitor processes and command-line arguments for script execution and subsequent behavior.
Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known tools and protocols can be used to mitigate and detect activity at the network level.
Monitor for file creation and files transferred into the network. Unusual processes with external network connections creating files on-system may be suspicious.
Block unknown or unused attachments by default that should not be transmitted over email as a best practice to prevent some vectors, particularly .lnk for Sofacy.
Anti-virus can potentially detect malicious documents and files that are downloaded on the user's computer.
Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known tools and protocols can be used to mitigate and detect activity at the network level.
Analyze network data for uncommon data flows (e.g., a client sending significantly more data than it receives from a server).
Where possible, remove PowerShell. If not, set PowerShell execution policy to execute only signed scripts.
If PowerShell is not used in an environment, monitor PowerShell execution, otherwise monitor changes in PowerShell execution policy.
Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known tools and protocols can be used to mitigate and detect activity at the network level.
Analyze packet contents to detect application layer protocols that do not follow the expected protocol standards regarding syntax, structure, or any other variable adversaries could leverage to conceal data.
Legend
Orange - Prioritised Technique
Blue - Proactive Recommendation
Green - Reactive Recommendation
Application control may be able to prevent the running of executables masquerading as other files, particularly .lnk files masquering as word documents.
Endpoint/network sensing can potentially detect malicious events once the attachment is opened (such as a Microsoft Word document reaching out to the internet or spawning Powershell.exe).
Network/Host intrusion prevention systems, antivirus, and detonation chambers can be employed to prevent documents from fetching and/or executing malicious payloads.
Consider disabling Microsoft Office macros/active content to prevent the execution of malicious payloads in documents.
Analyze process behavior to determine if an Office application is performing actions, such as opening network connections, spawning abnormal child processes (ex: PowerShell)
Web proxies can be used to enforce external network communication policy that prevents use of unauthorized external services.
Analyze network data for uncommon data flows (e.g., a client sending significantly more data than it receives from a server).