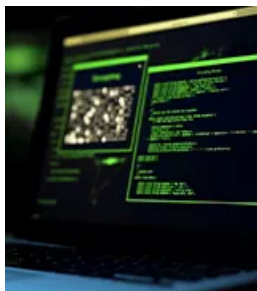


LeslieLoader Malware Sparkrat Cyber Risk Kroll

kroll.com/en/insights/publications/cyber/leslieloader-undocumented-loader-observed

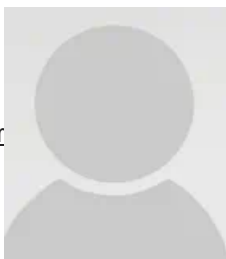


Cyber

Wed, Mar 13, 2024

LESLIELOADER – Undocumented Loader Observed

Marc Messer



Marc Messer

Key Takeaways

- Kroll has observed a new loader for SPARKRAT malware used in ongoing campaigns.
- While SPARKRAT development has officially ceased, unofficially it has continued to be modified by threat actors as needed.
- One of these new changes is a previously undocumented loader, identified by the AES key “LeslieCheungKwok”.

Summary

Kroll observed the use of SPARKRAT in conjunction with a previously undocumented loader written in Golang. The loader assists in the initial infection and deployment of the malicious payload, enabling SPARKRAT to execute on a system. This process allows the payload to reach the target system undetected and unquarantined. The loader achieves its goal by decoding and decrypting a secondary payload binary, then injecting it into a notepad.exe instance. This injection allows the malware to blend with legitimate system activity as it shares the memory space of a legitimate application. Despite detection tools’ ability to mitigate process injections, they remain a common evasion tactic.

GitHub developer XZB-1248 wrote and released SPARKRAT, a Golang binary compiled for multiple platforms, as an open-source, feature-rich remote admin tool on GitHub on March 18, 2022. Even though the developer abandoned the project in February 2023, intrusion investigations continue to discover modified versions of SPARKRAT. The “DRAGONSPARK” campaign, notorious for its attacks against organizations in East Asia, frequently uses this malware. SPARKRAT interprets its embedded Golang source code at runtime, which complicates analysis and static detections.

A source code repository very similar to LESLIELOADER has been identified alongside with instructions on how to utilize the loader for any payload necessary, originally timestamped June 7, 2022. Steps include generating a shellcode payload, AES encrypting the payload, and generating the executable. Additionally, the author posted proofs of running the samples against various antivirus and sandbox tools. However, there are some key differences from LESLIELOADER. Unlike the samples observed by Kroll, this loader does not beacon out a network connection. Additionally, this loader does not use process injection for code execution or a secondary file for payload.

```
func DecrptogAES(src, key []byte) []byte {
    block, err := aes.NewCipher(key)
    if err != nil {
        fmt.Println(err)
        return nil
    }
    blockMode := cipher.NewCBCDecrypter(block, key)
    blockMode.CryptBlocks(src, src)
    src = UnPaddingText1(src)
    return src
}

func main() {
    str := "payload"
    key := []byte("LeslieCheungKwok")
    base_byte, _ := base64.StdEncoding.DecodeString(str)
    build(string(DecrptogAES(base_byte, key)))
}
```

Figure 1: Similar Code Repository

Later forks of this repository show modified versions of the initial loader, which begin to implement Base64 decoding. Continued modification of this original source code likely resulted in the version covered within this article.

```

func main() {
    //payload替换
    str := "base64-payload"
    //密钥长度16
    key := []byte("LeslieCheungKwok")
    src := EncryptogAES([]byte(str), key)
    //fmt.Println(src)
    base64Str := base64.StdEncoding.EncodeToString(src)
    fmt.Println("加密后的数据为:", base64Str)
}

```

Figure 2: Forked Repository Showing Addition of Base64 Decoding Stage

Kroll has identified and triaged additional LESLIELOADER samples and has observed them to contain Cobalt Strike configurations as well as other payloads, so this loader is not limited to SPARKRAT.

First Loader Stage

The loader begins with two files, Ntmssvc.dll and RemovableStorage.dll. Upon execution of Ntmssvc.dll with the /runcode flag, RemovableStorage.dll is read from C:\Windows\System32\Ntmssvc.dll contains the loader functionality, and RemovableStorage.dll functions as the payload.

decrypt_Removable_Storage	XREF[2]:	6e327525(j),
fcn.Spark_SvchostDLL_cmd.RunCode		fcn.Spark_SvchostDLL_cmd.Run:6e3...

```

LEA    R12=>local_10,[RSP + -0x10]

CMP    R12,qword ptr [R14 + 0x10]
JBE    LAB_6e32751f

SUB    RSP,0x90

MOV    qword ptr [RSP + local_8],RBP

LEA    RBP=>local_8,[RSP + 0x88]

LEA    RAX,[s_C:\Windows\System32\RemovableSto_6e3f85... = "C:\Windows\System32\Remova...

MOV    EBX,0x28

CALL    fcn.os.ReadFile                                undefined fcn.os.ReadFile(undefi...

```

Figure 3: RemovableStorage.dll File Read into Memory

Ntmssvc.dll initially attempts to beacon out to 209.141.50[.]215:443, however, this execution can be skipped in favor of overwriting the instruction pointer to the storage decoding function.

Figure 4: HTTP Beacon Attempt

Figure 5: Jumping to the Loader Function

4/11

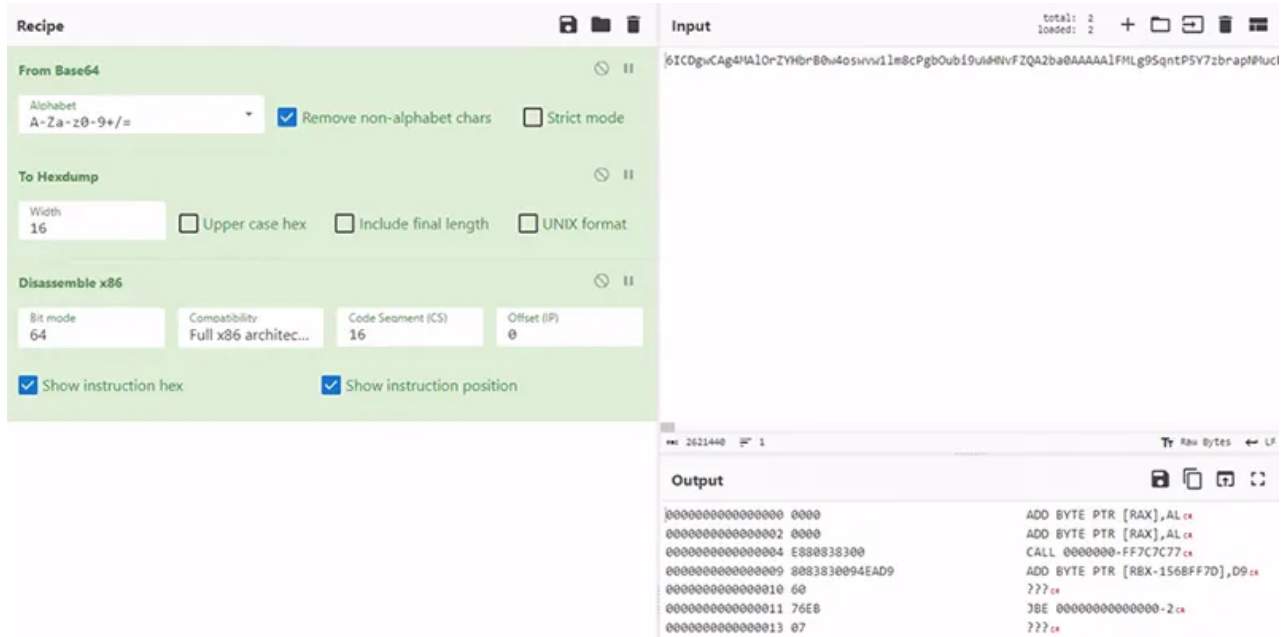


Figure 8: CyberChef Recipe and Output

To continue observing the loader's behavior to the memory pages containing these payloads, we set hardware breakpoints for memory access to them. Ultimately, we can observe the final payload for process injection dynamically calculated to have the size of 83DA50 in registers RCX and RDX.

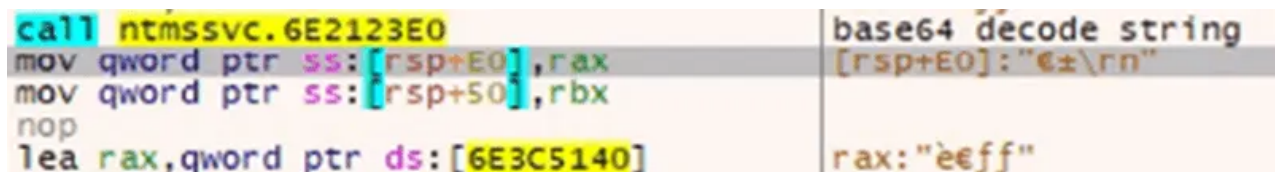


Figure 9: Preparing for process injection.

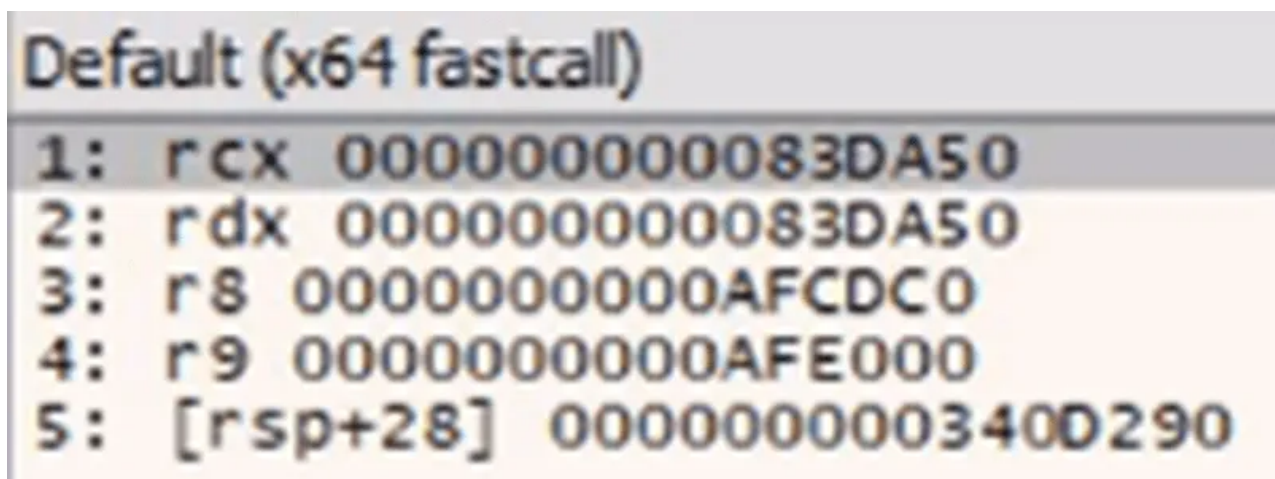


Figure 10: Final Payload Size in RAX and RDX

Process injection to notepad.exe begins with a matching payload size in register r8 when memory is being allocated.

Notepad.exe is launched as a suspended process, however, there are additional artifacts of note within the loader indicating other processes such as calc.exe and cmd.exe. While they do not appear to be used in this sample for injection, the analysis of the use of these processes is out of scope.

lea rax,qword ptr ds:[6E3F5402]	000000006E3F5402:"C:\\windows\\System32\\calc.exeCanal
mov qword ptr ds:[6E368180],rax	000000006E368180:&"C:\\windows\\System32\\notepad.exe
jmp ntmssvc.6E32643F	
lea rdi,qword ptr ds:[6E368180]	000000006E368180:&"C:\\windows\\System32\\notepad.exe
lea rax,qword ptr ds:[6E3F5402]	000000006E3F5402:"C:\\windows\\System32\\calc.exeCanal
call ntmssvc.6E125E40	
mov ebx,1C	
call ntmssvc.6E3260E0	
test al,al	
jne <ntmssvc.get pointer for notepad.exe	
mov qword ptr ds:[6E368188],1B	
cmp dword ptr ds:[6E635600],0	
nop	
jne ntmssvc.6E326472	
lea rax,qword ptr ds:[6E3F4821]	000000006E3F4821:"C:\\windows\\System32\\cmd.exeCertE
mov qword ptr ds:[6E368180],rax	000000006E368180:&"C:\\windows\\System32\\notepad.exe
jmp ntmssvc.6E326485	
lea rdi,qword ptr ds:[6E368180]	000000006E368180:&"C:\\windows\\System32\\notepad.exe
lea rax,qword ptr ds:[6E3F4821]	000000006E3F4821:"C:\\windows\\System32\\cmd.exeCertE
call ntmssvc.6E125E40	
mov ebx,1B	
call ntmssvc.6E3260E0	
test al,al	
jne <ntmssvc.get pointer for notepad.exe	
mov qword ptr ds:[6E368188],1F	
cmp dword ptr ds:[6E635600],0	
jne ntmssvc.6E3264B7	
lea rcx,qword ptr ds:[6E3F6D68]	000000006E3F6D68:"C:\\windows\\System32\\svchost.exeC
mov qword ptr ds:[6E368180],rcx	000000006E368180:&"C:\\windows\\System32\\notepad.exe
jmp <ntmssvc.get pointer for notepad.exe	
lea rdi,qword ptr ds:[6E368180]	000000006E368180:&"C:\\windows\\System32\\notepad.exe
lea rcx,qword ptr ds:[6E3F6D68]	000000006E3F6D68:"C:\\windows\\System32\\svchost.exeC
call ntmssvc.6E125F40	
mov rax,qword ptr ds:[6E368180]	000000006E368180:&"C:\\windows\\System32\\notepad.exe

Figure 11: Acquiring Pointer to notepad.exe

Once notepad.exe is created as a suspended process, Ntmssvc.dll overwrites the process memory for notepad.exe to manipulate the entry point. Prior to overwriting the entry point, notepad.exe proceeds as follows:

sub rsp,28	EntryPoint
call notepad.7FF6852D63A4	
add rsp,28	
jmp notepad.7FF6852D58B0	
int3	
int3	
int3	
int3	

Figure 12: Original Entry Point

Once the notepad.exe entry point has been overwritten, the memory address of our SPARKRAT payload is loaded into RAX and jumped to, beginning execution of the malware.

mov rax,1F311360000	1F311360000:"èëff"
jmp rax	
sub cl,ch	
outsb	

Figure 13: Modified entry point

This now allows for the injected payload to be executed as notepad.exe.

File Information

Filename Ntmssvc.dll

MD5	a8164d4d4c7c0669f01f0cec11db5c23
SHA1	524d1042f6834b5d5af6fc8361d25eb3c5e0925c
SHA-256	ec45da0ca70a9b71652cc95d51665f7ad568294bd5652c395a119bccd613e9b4
SSDeep	49152:NzPXvD233tOrb/TgvO90d7HjmAFd4A64nsfJaZmq59X4uv+J9mqBAhmm9tHaF90V:o3l/1MEUkz6h+
File Size	5.2M
Go Version	1.19.6
Go Build ID	CkXRLYLv5G2q6zLSX2ow/XDMcRiYChHDh4RhVcfIB/AvlcOAK9US3h4x7V9sBS/D-1jdZnavCSSXOMBjnmk

Filename RemovableStorage.dll

MD5	d8ed87bccf4ff55b98ee6fd229427a45
SHA1	1e0d10e17bf5fc2b1dd7cfc717767d87e5fd4489
SHA-256	b8cab11421eb4731c16cf3c34ca2b3f2a758d5e112f877b90a18b3e146c8add0
SSDeep	49152:wTJSBSikp+WKZvBSVAYk5d606sKhdu6XAPf48ud5cFCz0/cef1Y0Ei7Mt3iyEiYM:O
File Size	15M

Additional Samples

During this investigation, Kroll identified other samples with indicators common to this loader. Several of these files notably have indications of Cobalt Strike and shellcode injection payloads. VirusTotal currently has the following list of hashes that have been identified with the yara rule listed below:

SHA-256	VT Submissions
f63511dc1d46742a9afb09e51294bcefa6a950326688c6e67b13facbf8ecdccf	1
0b77f2dc04fcb24034feb26ed8bb0721da93f55d74ea8347373946abdf245f37	1
e70aac286c3b320c7cf19a74904c3ff5ecd1795915c98c56c9ff3915875595da	1
75a49cd2762810d0e9bd0187e8a29a0ad3e84281028e8aec44723249a5da2363	1

SHA-256	VT Submissions
e8a08771552c29c573f55f6d444d9b9282f331e703fda76110deb124b801d730	1
f28f9a11926cfbd08896ee669607ea9ce10f69e217ce10f5ed74db688687cba5	1
b5ba48002f13975f23851a3ec80547aa71d6953ce7aad42af3ba4a87699c80dd	1
05ecd6749aea5cade72b028994e719b3c3315103ef8ba6c46d4dba72575080e0	2
c9b36dbca2f3260c62c1180720be599886e56191be9d4b1822dacec4d8dd3e99	1
4741ddd031d4741e9ef92f606753ea4e95b9e161173a2a82c18ffd1f8206d852	1
fd557bc84742fa76efd394f0f5f648a889648bd01487c489aa395251b61a7d60	1
b5d46ffed1ec57b24867499bd15f5a0487443e1f0f4c1fa578d917e851560ba4	1
24c197844f1275df73f45803b1a740d4dae565f6de06bac388f2cd6df454469c	1
fb728d019149bb3dd696c1b417ceea008b88a18baf61b2640bae224ed8148cbe	1
0c3cb0a8314e668a65df60800c8075e34b95d1d18fe72fe269f938454543bca9	1
983e03a0f46cd40a861f7dc3e4efd1397b9b474168dc322a75a551bfb3fa8ded	1
db42d41a59e3e5c9bb234479ce05cd1923ef9754911cc14e9b3e5d500e191511	1
94445af999055bf7d7cddc0d1d5183ab2776d85285f0522a28fac6c5a6101906	1
fdea8b01b2597ceafe6f08b5fd12cc603b1e3ce2037731c0b6defde6935b1ce0	1
25bfed3eb9ebc12604edee0aeb4c8bc2f2a37e9fd1c77ebda33b5756509a726d	1
2df61720ffee7a6730c029da45bddba2ec9eace7359e1ee964b44560e9062bb9	1
12afe1b18b0e7dcc4903fff3a7dc2a2b6c3d111c2f836e7b4d13ed082b8dea89	1
73d1d42b76e1051c6ad0c832dc1c6bcd7158f4a0687f1f7d84c9c3c513b0e43e	1
4ca0d50308013957a3da3fc60a62649c405fd9dbf7b184b18e73333ddf23e0bc	1

Additionally, the below Yara rule may be used to search for indications of this loader:

```
import "pe"

rule leslieloader
{
  meta:
    author = "Marc Messer"
    description = "Checks for the inline string 'LeslieCheungKwok' with instructions."
    truepositive = "ec45da0ca70a9b71652cc95d51665f7ad568294bd5652c395a119bccd613e9b4"
  strings:
    $a = {48 b9 4c 65 73 6c 69 65 43 68 48 89 08 48 b9 65 75 6e 67 4b 77 6f 6b}
  condition:
    pe.is_pe and $a
}
```

[Stay Ahead with Kroll >](#)

Cyber Threat Intelligence

Threat intelligence are fueled by frontline incident response intel and elite analysts to effectively hunt and respond to threats.

[Cyber Threat Intelligence](#)

Malware Analysis and Reverse Engineering

Kroll's Malware Analysis and Reverse Engineering team draws from decades of private and public-sector experience, across all industries, to deliver actionable findings through in-depth technical analysis of benign and malicious code.

[Malware Analysis and Reverse Engineering](#)

Cyber and Data Resilience

Incident response, digital forensics, breach notification, security strategy, managed security services, discovery solutions, security transformation.

[Cyber and Data Resilience](#)

Kroll Responder MDR

Stop cyberattacks. Kroll Responder managed detection and response is fueled by seasoned IR experts and frontline threat intelligence to deliver unrivaled response.

[Kroll Responder MDR](#)

Penetration Testing Services

Validate your cyber defenses against real-world threats. Kroll's world-class penetration testing services bring together front-line threat intelligence, thousands of hours of cyber security assessments completed each year and a team of certified cyber experts — the foundation for our sophisticated and scalable approach.

Penetration Testing Services

Cyber Risk Assessments

Kroll's cyber risk assessments deliver actionable recommendations to improve security, using industry best practices & the best technology available.

Cyber Risk Assessments