

TellYouThePass Ransomware Analysis Reveals Modern Reinterpretation Using Golang

crowdstrike.com/blog/tellyouthepass-ransomware-analysis-reveals-modern-reinterpretation-using-golang/

Anmol Maurya

January 11, 2022



- TellYouThePass ransomware, discovered in 2019, recently re-emerged compiled using Golang
- Golang's popularity among malware developers makes cross-platform development more accessible
- TellYouThePass ransomware was recently associated with Log4Shell post-exploitation, targeting Windows and Linux
- The CrowdStrike Falcon® platform protects customers from Golang-written TellYouThePass ransomware using the power of machine learning and behavior-based detection

The TellYouThePass ransomware family was recently reported as a post-exploitation malicious payload used in conjunction with a remote code execution vulnerability in Apache Log4j library, dubbed Log4Shell.

TellYouThePass was first reported in early 2019 as a financially motivated ransomware designed to encrypt files and demand payment for restoring them. Targeting both Windows and Linux systems, TellYouThePass ransomware re-emerged in mid-December 2021 along with other ransomware like Khonsari. This lesser-known ransomware family came back into the spotlight as a post-exploitation payload associated with the Log4Shell. The remote code execution vulnerability is estimated to expose affected organizations to a wave of cybersecurity risks.

Previously known TellYouThePass ransomware samples were written in traditional programming languages like Java or .Net., but two new recent samples reported in public repositories have been rewritten and compiled in Golang.

Golang's popularity among malware developers has steadily increased over the past years. It allows them to use the same codebase and compile it for all major operating systems, making cross-platform development work more accessible.

What follows is a deeper dive into the new Golang-written TellYouThePass ransomware samples for Windows and Linux and how the CrowdStrike Falcon platform protects against them.

Setting Up the Analysis

We first check the binary for the “Go build id” string to identify the Golang build used for compiling it. In recent campaigns of Go-written malware, especially in ransomware cases, attackers patch the binary to remove this string, making it difficult for researchers to use string-based signatures to detect the binary as Go.

Going through the two samples —

`460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a38984` (Windows)

`5c8710638fad8eeac382b0323461892a3e1a8865da3625403769a4378622077e` (Linux)

— we noticed that more than 85% of code in the Windows and Linux versions are almost the same:

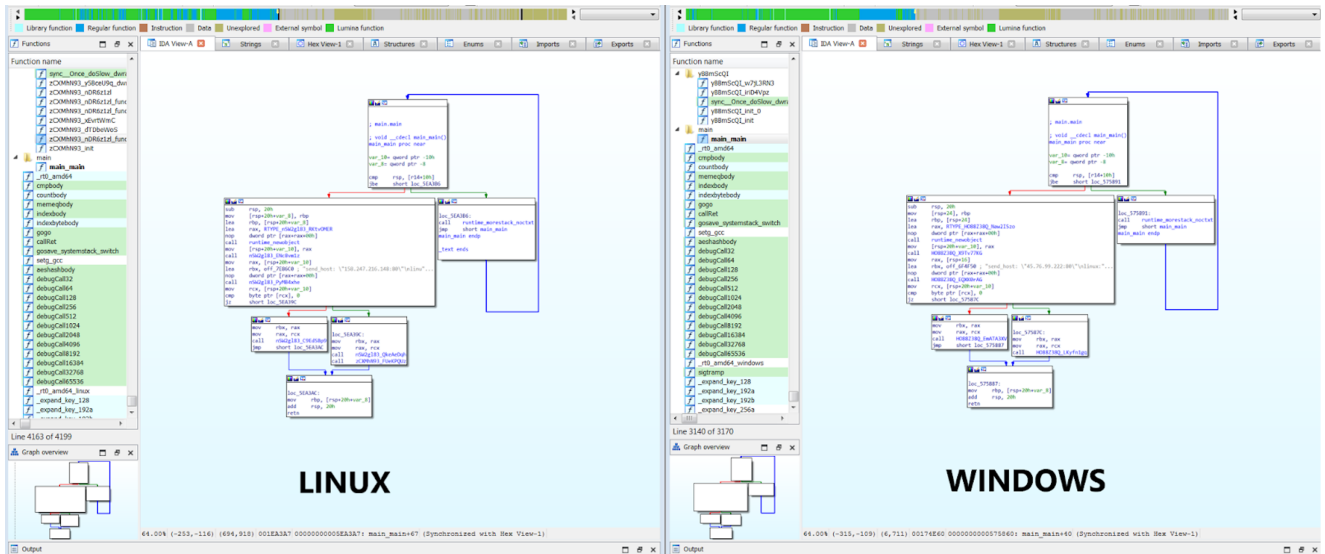


Figure 1. The “main.” functions for both Windows and Linux samples are almost identical (Click to enlarge)

A deeper dive into the some of the ransomware’s functions:

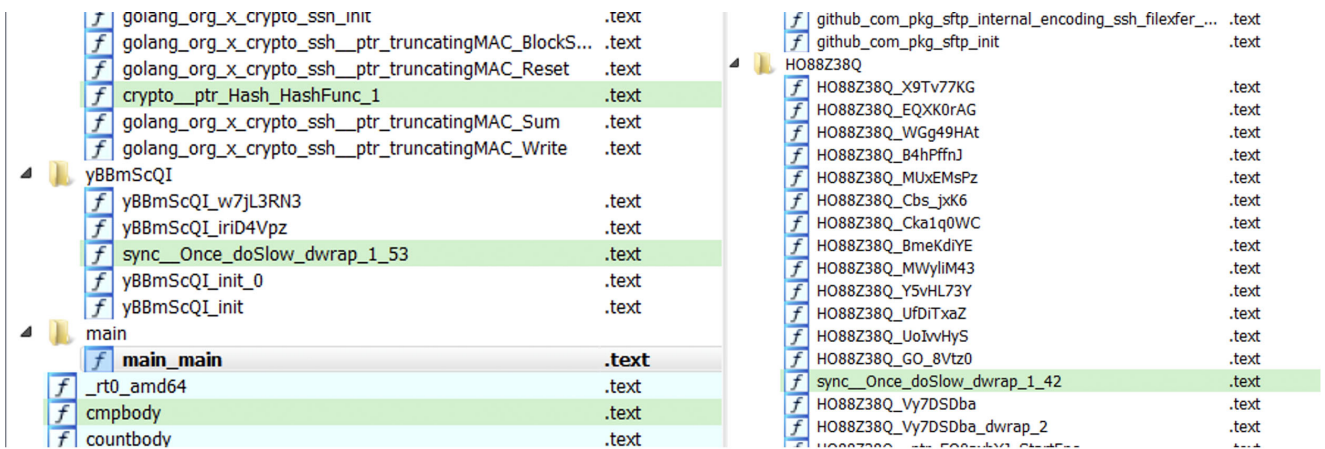


Figure 2. TellYouThePass ransomware functions for the Windows sample in IDA Pro (Click to enlarge)

As we have previously discussed, we start by focusing on the “main.” functions in Golang. We notice in this case that the malware authors have left only one main function and changed the other functions to random names, making analysis difficult.

The sample checks the existence of the files “ showkey.txt ” and “ public.txt ” with the help of OS.Getenv, using “ ALLUSERSPROFILE ” and “ HOMEDRIVE ” as keys in Windows and Home and /tmp/ in Linux. If it is present, it means encryption occurred, and it exists using runtime_gopanic ; otherwise, it creates them.

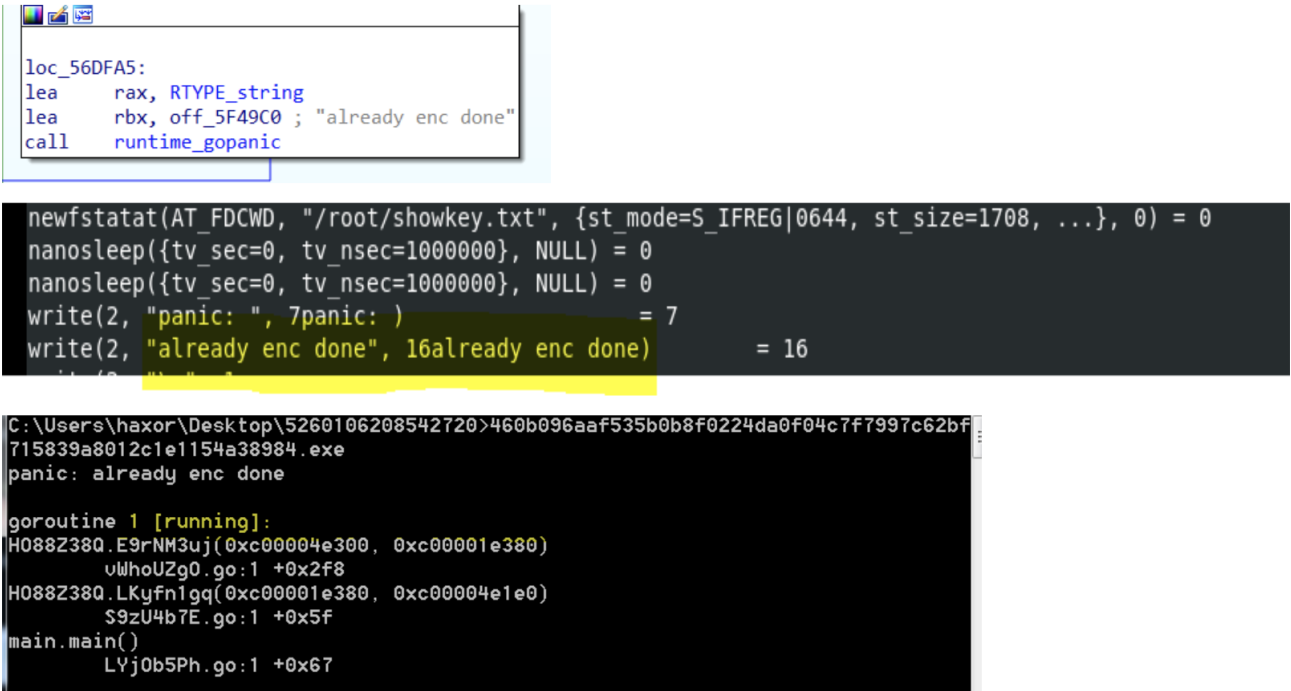


Figure 3. Encryption function followed by successful encryption for both Linux and Windows (Click to enlarge)

For Windows, the return is “ C:\\ProgramData “ and /root/ directory in Linux. Using path.join to join “ showkey.txt “ and “ public.txt ” with the directories results in:

Windows	Linux
• “ C:\\ProgramData/showkey.txt ”	• “ /root/showkey.txt ”
• “ C:\\ProgramData/public.txt ”	• “ /root/public.txt ”

Table 1. Directories for saving showkey.txt and public.txt

The sample uses the Golang Crypto Packages for RSA key — some of them are crypto_x509_MarshalPKCS1PublicKey, crypto_x509_MarshalPKCS1PrivateKey, encoding_pem_EncodeToMemory and crypto_rsa_GenerateMultiPrimeKey.

As seen in Figure 4, crypto_x509_MarshalPKCS1PrivateKey converts the RSA private key to PKCS #1, ASN.1 DER form. Then, the encoding_pem_EncodeToMemory returns the PEM (Privacy Enhanced Mail) encoding, and after that, runtime_slicebytetostring converts bytes to string, resulting in the conversion of bytes to string (see Figure 5).

```
sub    rsp, 50h
mov    [rsp+50h+var_8], rbp
lea    rbp, [rsp+50h+var_8]
call   crypto_x509_MarshalPKCS1PrivateKey
lea    rdx, [rsp+50h+var_38]
movups xmmword ptr [rdx], xmm15
lea    rsi, [rsp+50h+var_28]
movups xmmword ptr [rsi], xmm15
lea    rsi, [rsp+50h+var_18]
movups xmmword ptr [rsi], xmm15
lea    rsi, aRsaPrivateKey ; "RSA PRIVATE KEY"
mov    [rsp+50h+var_38], rsi
mov    [rsp+50h+var_30], 0Fh
mov    [rsp+50h+var_20], rax
mov    [rsp+50h+var_18], rbx
mov    [rsp+50h+var_10], rcx
mov    rax, rdx
nop    dword ptr [rax+rax+00h]
call   encoding_pem_EncodeToMemory
mov    rcx, rbx
mov    rbx, rax
xor    eax, eax
call   runtime_slicebytetostring
mov    rbp, [rsp+50h+var_8]
add    rsp, 50h
retn
```

Figure 4. Function that generates the RSA private key

Address	Hex	ASCII
000000C000202000	2D 2D 2D 2D 2D 42 45 47 49 4E 20 52 53 41 20 50	-----BEGIN RSA P
000000C000202010	52 49 56 41 54 45 20 48 45 59 2D 2D 2D 2D 0A	RIVATE KEY-----.
000000C000202020	4D 49 49 43 58 67 49 42 41 41 4B 42 67 51 43 72	
000000C000202030	65 56 55 56 51 63 65 33 54 73 6F 38 66 71 6E 2B	
000000C000202040	75 39 75 37 49 2B 2B 7A 6C 4E 71 42 30 70 6E 45	
000000C000202050	6D 35 54 75 65 66 4F 6A 4C 47 42 4F 57 2B 34 64	
000000C000202060	0A 5A 50 55 30 64 79 6F 51 36 6D 31 54 4F 53 70	
000000C000202070	37 78 57 50 4F 75 6E 70 67 69 6B 57 76 67 6C 2F	
000000C000202080	64 56 72 65 42 70 30 32 58 78 34 76 63 6E 75 52	
000000C000202090	4E 58 50 2F 2F 6C 7A 4E 64 42 56 69 68 74 64 48	
000000C0002020A0	78 0A 49 45 37 7A 6F 44 55 36 74 34 6E 64 79 4B	
000000C0002020B0	54 71 35 50 6F 6A 6D 48 30 48 58 75 54 54 30 6C	
000000C0002020C0	32 54 37 51 79 53 54 6F 41 35 36 61 4C 33 76 50	
000000C0002020D0	52 56 4C 6A 79 2F 6C 58 6B 57 54 77 49 44 41 51	
000000C0002020E0	41 42 0A 41 6F 47 41 43 47 34 67 70 72 6A 6A 4C	
000000C0002020F0	72 6E 71 36 32 70 32 78 52 56 4C 53 6A 6F 4D 45	rng6zpxRVL5jome
000000C000202100	4E 49 69 6F 2F 74 4D 6F 41 50 65 49 4A 4E 53 54	NIio/tmoAPEIjNST
000000C000202110	52 56 6A 62 72 62 4B 55 42 75 6B 69 6E 33 4A 54	RVjbrbkUBukin3JT
000000C000202120	61 65 59 0A 31 46 79 64 49 42 53 6D 51 59 57 64	aeY.1FydIBSmQYwd
000000C000202130	65 70 32 52 71 33 31 48 5A 55 52 63 4A 53 6B 43	ep2Rq31HZURCjSkC
000000C000202140	47 44 54 74 67 66 2F 66 45 38 58 32 64 45 71 41	GDTtgf/fe8X2deEq
000000C000202150	78 36 73 65 4C 43 6D 42 4C 76 45 45 67 30 76 41	x6seLCmBLVEEg0vA
000000C000202160	37 31 70 31 0A 78 69 38 71 6E 37 7A 54 6D 74 4E	71p1.xi8qn7zTmtN
000000C000202170	6F 35 4B 59 58 33 68 71 58 50 34 68 76 79 34 42	o5KYX3hqXP4hvy4B
000000C000202180	74 58 56 44 2F 48 35 39 61 7A 78 48 49 67 76 74	tXVD/H59azxHIGvt
000000C000202190	4A 78 32 45 43 51 51 44 68 6F 4B 54 6E 54 77 44	Jx2ECQQDhokTnTwD
000000C0002021A0	36 39 50 30 5A 0A 58 77 67 50 38 4D 5A 47 55 77	69POZ.XwgP8MZGUw
000000C0002021B0	5A 4D 4F 55 52 51 48 7A 2F 57 43 73 33 68 41 39	ZMOURQHz/wcs3ha9
000000C0002021C0	41 33 51 65 47 31 77 4F 78 66 45 31 58 47 4C 77	A3QeG1wOxfE1XGLw
000000C0002021D0	71 62 78 35 56 72 46 31 46 2B 6A 78 57 68 6C 35	qbx5vrF1F+jxwh15
000000C0002021E0	41 65 4B 74 31 66 0A 53 4A 32 74 4D 4C 66 6E 41	Aekt1f.Sj2tMLfna

Command: Commands are comma separated (like assembly instructions): mov eax, ebx

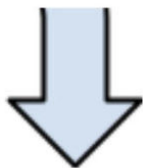
Figure 5. The generated RSA key (Click to enlarge)

The RSA public key is generated using the

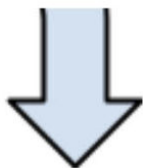
`encoding_base64_ptr_Encoding_DecodeString` and `encoding_pem_encode` packages from Golang, as shown in Figure 6.

Address	Hex	ASCII
00000000005C6630	4C 53 30 74 4C 53 31 43 52 55 64 4A 54 69 42 53	LS0tLS1CRUdJTiBS
00000000005C6640	55 30 45 67 55 46 56 43 54 45 6C 44 49 45 74 46	U0EgUfVCTE1DIETF
00000000005C6650	57 53 30 74 4C 53 30 74 43 6B 31 4A 53 55 4A 4A	WS0tLS0tck1JSUJJ
00000000005C6660	61 6B 46 4F 51 6D 64 72 63 57 68 72 61 55 63 35	akFOQmdrcwHrauc5
00000000005C6670	64 7A 42 43 51 56 46 46 52 6B 46 42 54 30 4E 42	dzBCQVFRkFBT0NB
00000000005C6680	55 54 68 42 54 55 6C 4A 51 6B 4E 6E 53 30 4E 42	UthbTUlJQknS0NB
00000000005C6690	55 55 56 42 65 58 6B 34 4C 31 4E 35 63 48 46 6B	UUVBexk4L1N5CHF
00000000005C66A0	4B 30 56 32 4E 32 35 55 63 32 56 58 62 32 73 4B	K0V2N25uc2vxbsk
00000000005C66B0	53 48 68 56 61 32 70 68 4D 55 6C 47 4F 46 4E 52	SHhva2pHmU1GOFNR
00000000005C66C0	52 46 63 32 59 6E 4E 31 59 6A 42 6C 62 47 46 43	RFc2Ynn1YjblbgFC
00000000005C66D0	53 7A 5A 32 52 6A 68 47 63 32 70 34 62 31 5A 72	Szz2Rjhgc2p4b1Zr
00000000005C66E0	4F 45 35 34 52 6A 59 30 51 6C 52 31 53 6D 4E 30	OE54RjY0Q1R1SmN0
00000000005C66F0	63 32 77 35 59 30 39 6D 56 46 6C 44 5A 33 56 35	c2w5Y09mVf1Dz3V5
00000000005C6700	5A 48 42 52 55 67 6F 78 65 55 78 4D 57 55 39 6B	ZHBRUgoxelUXMwU9k
00000000005C6710	52 79 39 47 54 6D 52 43 64 30 39 69 54 47 74 70	Ry9GTmRcd09iTGtp
00000000005C6720	65 6B 56 4C 63 46 4E 33 55 46 6C 5A 61 44 55 34	ekVLcFN3UF1Zadu4
00000000005C6730	61 46 64 52 4C 32 46 50 62 6B 6C 6E 4E 33 52 53	aFdRL2FPbk1n3RS
00000000005C6740	62 30 4E 54 57 45 67 79 64 47 49 34 64 6E 46 73	b0NTwEgydGI4dnFs
00000000005C6750	57 6E 42 56 55 7A 46 51 51 6C 5A 68 43 6B 46 42	wnBVuzFQq1ZchckFB
00000000005C6760	59 6A 4A 57 56 33 45 76 61 48 70 6B 4D 31 51 30	YjJwV3EvaHpkM1Q0
00000000005C6770	59 6A 56 46 4D 48 4D 32 5A 48 46 4D 64 6D 39 31	YjVFMHM24HFMDm91
00000000005C6780	61 6B 38 30 55 30 34 30 53 6D 56 54 62 6B 5A 77	ak80U040SmvTbkZw
00000000005C6790	57 6C 4E 4E 63 31 4E 53 63 6B 34 79 55 58 46 4A	w1nnc1nscK4yUXFJ
00000000005C67A0	4B 32 4A 4F 61 79 74 4E 55 32 39 71 64 6D 4A 36	K2JoaytNU29qdmJ6
00000000005C67B0	63 45 6F 4B 54 32 68 78 62 6A 56 70 61 55 56 61	ceOkT2hxbjvpaUva
00000000005C67C0	59 55 46 51 4D 6D 4A 34 4D 56 5A 53 55 6B 77 35	YUFQmMj4MvZsukw5
00000000005C67D0	65 48 4D 77 65 48 49 35 4E 54 52 4C 56 7A 49 72	eHMweHI5NTRLVzIr
00000000005C67E0	55 6E 6C 71 5A 33 52 30 4B 30 74 58 56 48 46 61	un1qz3R0K0tXVHfa
00000000005C67F0	56 48 46 4C 57 47 67 31 51 54 64 79 4E 7A 4E 69	VHFLwGg1QTdyNzNi
00000000005C6800	51 57 4E 79 57 69 39 4E 64 77 70 72 56 44 4A 76	QWnywi9NdwprVDJv

00000000005C6810	4E 0E 55 77	55 57 4A 31	4F 56 4A 55	62 54 4A 42	NNUWUJ10VJUDTJB
00000000005C6820	64 55 64 54	55 45 52 55	52 7A 4E 4E	62 57 6C 49	dudTUERURzNNbwI
00000000005C6830	62 57 6C 78	63 46 5A 75	54 48 6C 74	51 57 35 4D	bw1xcFZuTh1tqw5M
00000000005C6840	4F 56 6C 47	55 46 56 57	51 55 56 73	63 45 46 79	ov1GUFVWQUVsCEFY
00000000005C6850	55 48 46 75	4C 7A 4A 43	61 54 68 57	5A 46 52 6F	UHFuLzJcAThwZFRo
00000000005C6860	43 6D 56 52	53 55 52 42	55 55 46 43	43 69 30 74	CmVRSURBUUFCCi0t
00000000005C6870	4C 53 30 74	52 55 35 45	49 46 4A 54	51 53 42 51	LS0tRU5EIFJTQSBQ
00000000005C6880	56 55 4A 4D	53 55 4D 67	53 30 56 5A	4C 53 30 74	VUJMSUMgSOVZLS0t



```
call encoding_base64_ptr_Encoding_DecodeString
call encoding_pem_Decode
;
```



Address	Hex	ASCII
000000C000203FD6	00 00 00 00
000000C000203FE6	00 00 00 00
000000C000203FF6	00 00 00 00
000000C000204006	45 47 49 4E	-----B
000000C000204016	48 45 59 2D	EGIN RSA PUBLIC
000000C000204026	4E 42 67 6B	KEY-----MIIBIjA
000000C000204036	46 41 41 4F	NBqkqhkiG9w0BAQE
000000C000204046	43 41 51 45	FAAOCAQ8AMIIBCgk
000000C000204056	76 37 6E 54	CAQEAYy8/Sypqd+E
000000C000204066	31 49 46 38	v7nTsewok.HxUkja
000000C000204076	61 42 4B 36	1IF8SQDw6bsub0e1
000000C000204086	78 46 36 34	aBK6vF8FsJxovk8N
000000C000204096	54 59 43 67	xF64BTuJcts19coF
000000C0002040A6	4F 64 47 2F	TYCguydpQR.1yLLY
000000C0002040B6	4B 70 53 77	odG/FndBwoblkiZe
000000C0002040C6	6E 49 67 37	KpSwPYyh58hwQ/aO
000000C0002040D6	71 6C 5A 70	nIgt7tRocSXH2tb8v
000000C0002040E6	56 57 71 2F	qLZpus1PBva.AAb2
000000C0002040F6	64 71 4C 76	Vwq/hzd3T4b5E0s6
000000C000204106	46 70 5A 53	dqLvouj04SN4Jesn
000000C000204116	4E 6B 2B 4D	FpZSMsSRrN2QqI+b
000000C000204126	6E 35 69 69	Nk+MSojvbzpj.Ohq
000000C000204136	4C 39 78 73	n5iIEZaAP2bx1VRR
000000C000204146	6A 67 74 74	L9xs0xr954kw2+ry
000000C000204156	41 37 72 37	jgtt+kwTq2Tqkxh5
000000C000204166	32 6F 36 75	A7r73bAcrZ/Mw.kT
000000C000204176	53 50 44 54	2o6u0Qbu9RTm2Aug
000000C000204186	4C 79 6D 41	SPDTG3MmiHmtqpvn
000000C000204196	41 72 50 71	LymAnL9YFPUVAElp
000000C0002041A6	51 49 44 41	ArPqn/2Bi8vdTh.e
000000C0002041B6	20 52 53 41	QIDAQAB.-----END
000000C0002041C6	2D 2D 2D 2D	-----RSA PUBLIC KEY-
000000C0002041D6	00 00 00 00	-----
000000C0002041E6	00 00 00 00
000000C0002041F6	00 00 00 00
000000C000204206	00 00 00 00
000000C000204216	00 00 00 00

Figure 6. Base64 decoding (Click to enlarge)

After that, the PERSON_ID stores the encoding generated by “`encoding_base64_ptr_Encoding_EncodeToString`” (in this case:

“ `ABCDEFGHIJKLMN0PQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789 +/` ”

as array for Base64 std encoding) every time the sample runs, saving it into “ `showkey.txt` ”.

Afterward, another key is generated using the function below (Figure 7), also saving it into “ `public.txt` ”:

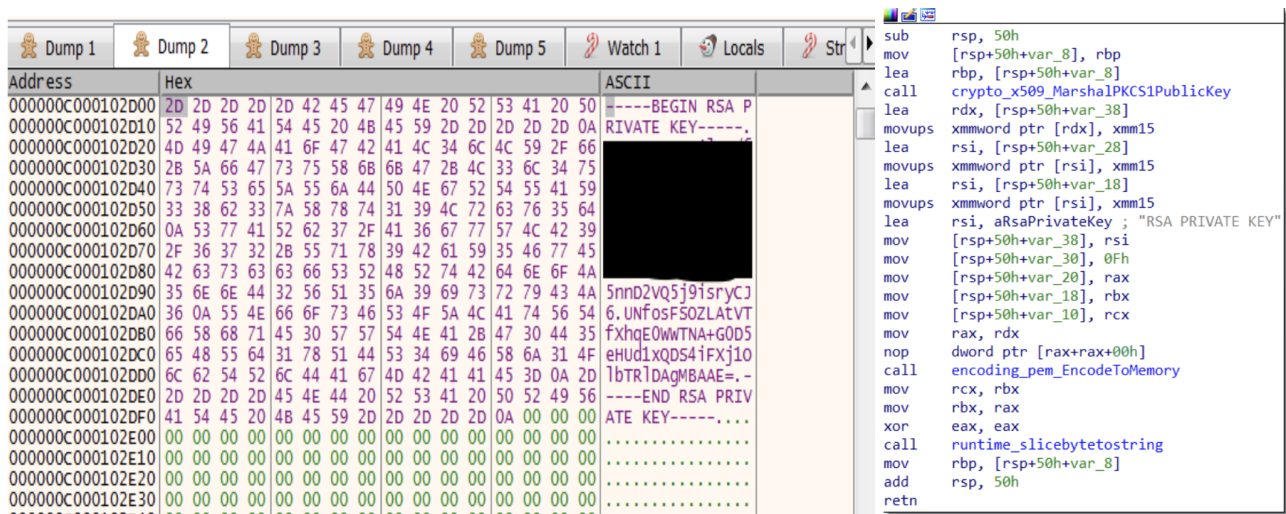


Figure 7. Key generation function (Click to enlarge)

Ransomware Behavior Prior to Encryption

TellYouThePass ransomware tries to kill some tasks and services before initiating the encryption routine, as shown in Table 2 below. However, in Linux, it requires root privilege to do that. Targeted applications include various email clients, database applications, web servers and document editors.

It runs various commands using `cmd.exe` to kill tasks in Windows, and in Linux, it takes the `os_exec_command` Go package to execute different commands using `/bin/bash/` :

Windows

Linux

-
- "taskkill /f /im msftesql.exe "
 - "schtasks /delete /tn WM /F "
 - "taskkill /f /im sqlagent.exe "
 - "taskkill /f /im sqlbrowser.exe "
 - "taskkill /f /im sqlservr.exe "
 - "taskkill /f /im sqlwriter.exe "
 - "taskkill /f /im oracle.exe "
 - "taskkill /f /im ocssd.exe "
 - "taskkill /f /im dbsnmp.exe "
 - "taskkill /f /im synctime.exe "
 - "taskkill /f /im mydesktopqos.exe "
 - "taskkill /f /im agntsvc.exeisqlplussvc."
 - "taskkill /f /im xfssvcon.exe "
 - "taskkill /f /im mydesktopservice.exe "
 - "taskkill /f /im ocautoupds.exe "
 - "taskkill /f /im agntsvc.exeagntsvc.exe "
 - "taskkill /f /im agntsvc.exeencsvc.exe "
 - "taskkill /f /im firefoxconfig.exe "
 - "taskkill /f /im tbirdconfig.exe "
 - "taskkill /f /im ocomm.exe "
 - "taskkill /f /im mysqld.exe "
 - "taskkill /f /im mysqld-nt.exe "
 - "taskkill /f /im mysqld-opt.exe "
 - "taskkill /f /im dbeng50.exe "
 - "taskkill /f /im sqbcoreservice.exe "
 - "taskkill /f /im excel.exe "
 - "taskkill /f /im infopath.exe "
 - "taskkill /f /im msaccess.exe "
 - "taskkill /f /im mspub.exe "
 - "taskkill /f /im onenote.exe "
 - "taskkill /f /im outlook.exe "
 - "taskkill /f /im powerpnt.exe "
 - "taskkill /f /im steam.exe "
 - "taskkill /f /im sqlservr.exe "
 - "taskkill /f /im thebat.exe "
 - "taskkill /f /im thebat64.exe "
 - "taskkill /f /im thunderbird.exe "
 - "taskkill /f /im visio.exe "
 - "taskkill /f /im winword.exe "
 - "taskkill /f /im wordpad.exe "
 - "taskkill /f /im tnslsnr.exe "
 - "service mysql stop"
 - "/etc/init.d/mysqld stop"
 - "service oracle stop"
 - "systemctl disable \"postgresql*\""
 - "systemctl disable \"mysql*\""
 - "systemctl disable \"oracle*\""

Table 2. TellYouThePass commands that try to terminate some tasks and services before initiating the encryption routine

After that, it iterates through all directories from **A to Z** and encrypts the files.

```

loc_56A857:
mov     [rsp+64], rcx
lea     rdx, aAbcdefghijklmn ; "ABCDEFGHJKLMNOPQRSTUVWXYZ"
movzx   ebx, byte ptr [rdx+rcx]

```

Both the Windows and the Linux versions have a list of directory exclusions for encryption, shown in Table 3.

Windows	Linux
• EFI.Boot	• /bin
• EFI.Microsoft	• /boot
• Windows	• /sbin
• Program Files	• /tmp
• All Users	• /etc
• Boot	• /lib
• IEidcache	• /proc
• ProgramData	• /dev
• desktop.ini	• /sys
• autorun.inf	• /usr/include
• netuser.dat	• /usr/java
• iconcache.db	
• thumbs.db	
• Local Settings	
• bootfont.bin	
• System Volume Information	
• AppData	
• Recycle.Bin	
• Recovery	

Table 3. TellYouThePass directory exclusions for encryption

The TellYouThePass ransomware focuses on encrypting popular media and file extensions, saving their paths in the “ `encfile.txt` ” text file, located in the same folder as “ `public.txt` ” and “ `showkey.txt` ”.

Below is the full list of targeted extensions for encryption:

1cd, 3dm, 3ds, 3fr, 3g2, 3gp, 3pr, 602, 7z, ps1, 7zip, aac, ab4, accdb, accde, accdr, accdt, ach, acr, act, adb, adp, ads, aes, agdl, ai, aiff, ait, al, aoi, apj, arc, arw, asc, asf, asm, asp, aspx, asx, avi, awg, back, backup, backupdb, bak, bank, bat, bay, bdb, bgt, bik, bin, bkp, blend, bmp, bpw, brd, c, cdf, cdr, cdr3, cdr4, cdr5, cdr6, cdrw, cdx, ce1, ce2, cer, cfg, cgm, cib, class, cls, cmd, cmt, conf, config, contact, cpi, cpp, cr2, craw, crt, crw, cs, csh, csl, csr, css, csv, dac, dat, db, db3, db_journal, dbf, dbx, dc2, dch, dcr, dcs, ddd, ddoc, ddrw, dds, der, des, design, dgc, dif, dip, dit, djv, djvu, dng, doc, docb, docm, docx, dot, dotm, dotx, drf, drw, dtd, dwg, dxb, dxf, dxg, edb, eml, eps, erbsql, erf, exf, fdb, ffd, fff, fh, fhd, fla, flac, flf, flv, flvv, fpx, frm, fxx, gif, gpg, gray, grey, groups, gry, gz, h, hbk, hdd, hpp, html, hwp, ibank, ibd, ibz, idx, iif, iiq, incpas, indd, jar, java, jnt, jpe, jpeg, jpg, jsp, jsp, ashx, js, kc2, kdbx, kdc, key,

kpdx, kwm, laccdb, lay, lay6, ldf, lit, log, lua, m, m2ts, m3u, m4p, m4u, m4v, mapimail, max, mbx, md, mdb, mdc, mdf, mef, mfw, mid, mkv, mlb, mml, mmw, mny, moneywell, mos, mov, mp3, mp4, mpeg, mpg, mrw, ms11, msg, myd, myi, nd, ndd, ndf, nef, nk2, nop, nrw, ns2, ns3, ns4, nsd, nsf, nsg, nsh, nvram, nwb, nx2, nxl, nyf, oab, obj, odb, odc, odf, odg, odm, odp, ods, odt, ogg, oil, orf, ost, otg, oth, otp, ots, ott, p12, p7b, p7c, pab, pages, paq, pas, pat, pcd, pct, pdb, pdd, pdf, pef, pem, pfx, php, pif, pl, plc, plus_muhd, png, pot, potm, potx, ppam, pps, ppsm, ppsx, ppt, pptm, pptx, prf, ps, psafe3, psd, pspimage, pst, ptx, pwm, py, qba, qbb, qbm, qbr, qbw, qbx, qby, qcow, qcow2, qed, r3d, raf, rar, rat, raw, rb, rdb, rm, rtf, rvt, rw2, rwl, rwz, s3db, safe, sas7bdat, sav, save, say, sch, sd0, sda, sdf, sh, sldm, sldx, slk, sql, sqlite, sqlite3, sqlitedb, sr2, srf, srt, srw, st4, st5, st6, st7, so, st8, stc, std, sti, stm, stw, stx, svg, swf, sxc, sxd, sxg, sxi, sxm, sxw, tar, tar.bz2, tbk, tex, tga, tgz, thm, tif, tiff, tlg, txt, uop, uot, vb, vbox, vbs, vdi, vhd, vhd, vmdk, vmsd, vmx, vmxf, vob, wab, wad, wallet, war, wav, wb2, wk1, wks, wma, wmv, wpd, wps, x11, x3f, xis, xla, xlam, xlc, xlk, xlm, xlr, xls, xlsb, xism, xlsx, xlt, xltm, xltx, xlw, xml, ybcra, yuv, zip.

Finally, the ransom note contains information about the encryption algorithm used to encrypt the files, specifically RSA-1024 and AES-256. It also includes the personid, used for identifying the victim. Following 0.05 bitcoin transfer into a designated and hardcoded wallet, attackers promise to provide victims with the decryption tool to recover all files.

I am so sorry ! All your files have been encryptedd by RSA-1024 and AES-256 due to a computer security problems.
If you think your data is very important .The only way to decrypt your file is to buy my decryption tool .
else you can delete your encrypted data or reinstall your system.

Your personid :

wVpNqCCHvOWGdNdDaOSoyus4zAqE5egy6BOiYHZWFz/p7Q3zN0BsY7PrfbrQtOp5IQR2R05/h4THwJ5rDQcpvrGdLr/6vxlby2ZGukPy+pz9vOzxE0KWRJ
WJ/6VDbHCvnyrSCHpLdtGycePEFX+pAAqCUxyrNgU676USwTUihAcxRMAdyFZuCFQjV6ao2r40MzfSB2Q+k9gvt3eE3m1855qp6AxBaJZ+VdQHcekxWvC
vRp3EKEDA3vHEWwCjnoQ5lnskNf69r1P9GU5IWrwiv78rGlp0fuRN7CFARQ984M/gWhVNBJozIR9grOkW7DMQy1i6Tr2Sv4u9Zzn8GzbhwFi78NWKqjv71E
AeuZVRpnMNIFpUefTEraF2ulXtUoDVhjn8GpbB3IG4YWoLk0ZvRFiT0pZgELGhCvPHs00ersotb/SIMX1Nd1bU1DA681nW85GUv5ENaqnQRSaczCU84YWv
dcF+nF98gzpsXxEF0VTKqH94dwWEAYy8JcNm9TMLxpY4FrGga/L1AXUkfcJlyHDNf7Dv+biDJwrbjefQxkBnWwGaDmdcRKvbuEUT10bCLWdxByiX63Y13I
SLbP2Z71FM7QovvCu/2hIg9YT4jTT6PDeCZKN4fndKe/4/fADvNRJ17Rc15ROZRJFxCkCMNP+8DnuC5RaJbF//EoEY57Y5231oQerjW1qWi8hDGqxZmJ3D
70WqC6xQkAlnmDfleVNuJTTYntLasQ7yfyfjWvruobpM3c5e3c6JF24h/rXcX2R38LMrHKrMVB02glQNAEFD8ibid3HIGDXN5C7JVo2YYRMoSmRLtsngaXxv-
oJeQRIRzHHkH0HD6BFxGYOAq7flosdIrry/PAFDw3UZJFqmSeqpDN1pGIVzNtE411WwkNicMYPq2By9PQfd2Ag2+2RA2wvq7xLliRmdDNMJs1GtlhviKQ:

Decryption do as follows:

1. if you not own bitcoin,you can buy it online on some websites. like <https://localbitcoins.net/> or <https://www.coinbase.com/> .
2. send 0.05 btc to my wallet address bc1qqxck7kpgzgvud7v2hfyk55yr45fuml4rmt3jasz.
3. send your btc transfer screenshots and your personid to my email service@goodluckday.xyz . i will send you decryption tool.

Tips:

- 1.don't rename your file
- 2.you can try some software to decryption . but finally you will know it's vain .
- 3.if any way can't to contact to me .you can try send me bitcoin and paste your email in the transfer information. i will contact you and send you decryption tools.

Anything you want to help . please send mail to my email service@goodluckday.xyz.
Have a nice day .

Figure 9. TellYouThePass ransom note (Click to enlarge)

The Falcon platform automatically detects and protects against this type of Golang-written malware using the power of the cloud, on-sensor and in-the-cloud machine learning, and indicators of attack (IOAs) to detect the threat. As Figure 10 shows, Falcon's cloud-based machine learning detects both Golang-written ransomware samples for TellYouThePass, immediately protecting Windows and Linux environments.

CrowdStrike Falcon leverages machine learning to identify known and unknown malware or threats by understanding malicious intent. Both on-sensor and cloud-based machine learning can detect and prevent post-exploitation threats leveraging exploits such as Log4Shell to protect against malware, including the new Golang-written TellYouThePass ransomware.

The screenshot displays the CrowdStrike Falcon console interface. On the left, a network diagram shows a connection between 'EXPLORER.EXE' and a file named '_1154A38984.EXE'. The file is highlighted with a red star icon, indicating a high-severity detection. On the right, the 'Execution Details' panel provides the following information:

- HOSTNAME:** WIN-
- HOST TYPE:**
- USER NAME:** WIN-
- SEVERITY:** High
- OBJECTIVE:** Falcon Detection Method
- TACTIC & TECHNIQUE:** Machine Learning via Cloud-based ML
- TECHNIQUE ID:** CST0008
- SPECIFIC TO THIS DETECTION:** This file meets the File Analysis ML algorithm's high-confidence threshold for malware.
- TRIGGERING INDICATOR:** Associated IOC (SHA256 on library/DLL loaded)
460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a...
- GLOBAL PREVALENCE:** Low
- LOCAL PREVALENCE:** Low
- IOC MANAGEMENT ACTION:** None
- Associated File:**
\\Device\HarddiskVolume1\Users\Desktop\5260106208542720\460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a38984.exe

Figure 10. Falcon detection of Golang-written Windows TellYouThePass ransomware sample (Click to enlarge)

Figure 11. Falcon detection of Golang-written Linux TellYouThePass ransomware sample (Click to enlarge)

The CrowdStrike Falcon platform provides protection against threats and visibility for all hosts in Windows, Linux and macOS, regardless of their location. The Falcon sensor can detect and prevent threats ranging from ransomware, cryptocurrency miners, trojans and botnets to stop today’s most sophisticated threats.

Indicators of Compromise (IOCs)

File/Host	sha256
Windows	<u>460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a38984</u>
Linux	<u>5c8710638fad8eeac382b0323461892a3e1a8865da3625403769a4378622077e</u>
Windows host	45[.]76[.]99[.]222[:]:]80
Linux Host	158[.]247[.]216[.]148[:]:]80

MITRE ATT&CK® Framework Mapping

Attack Id	Tactic	Description
T1059	Execution	Command and Scripting Interpreter

T1053	Execution Persistence Privilege Escalation	Scheduled Task/Job
T1027	Defense Evasion	Obfuscated Files or Information
T1140	Defense Evasion	Deobfuscate/Decode Files or Information
T1083	Discovery	File and Directory Discovery
T1057	Discovery	Process Discovery
T1560	Collection	Archive Collected Data
T1486	Impact	Data Encrypted for Impact

Additional Resources

- *Read more about Golang malware in this blog: [Golang Malware Is More than a Fad: Financial Motivation Drives Adoption](#)*
- *Learn about another ransomware variant that uses a Golang packer: [New Ransomware Variant Uses Golang Packer](#)*
- *Visit the product website to learn how the powerful [CrowdStrike Falcon platform](#) provides comprehensive protection across your organization, workers and data, wherever they are located.*
- *[Get a full-featured free trial of CrowdStrike Falcon Prevent™](#) and see how true next-gen AV performs against today's most sophisticated threats.*