

TellYouThePass Ransomware Analysis Reveals a Modern Reinterpretation Using Golang

Anmol Maurya



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

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.

<u>Golang's popularity among malware developers</u> 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:

		Ţ	goiang_org_x_crypto_ssn_init	.text	Ē.		f	github_com_pkg_sftp_internal_encoding_ssh_filexfer	.text
		f	golang_org_x_crypto_sshptr_truncatingMAC_BlockS	.text			f	github_com_pkg_sftp_init	.text
		f	golang_org_x_crypto_sshptr_truncatingMAC_Reset	.text	4	L	HOB	38Z38Q	
		f	crypto ptr Hash HashFunc 1	.text			f	H088Z38Q_X9Tv77KG	.text
		f	golang org x crypto ssh ptr truncatingMAC Sum	.text			f	H088Z38Q_EQXK0rAG	.text
		f	golang org x crypto ssh ptr truncatingMAC Write	text			f	HO88Z38Q_WGg49HAt	.text
		VPD					ţ	HO88238Q_B4hPffnJ	.text
-		ybb					Ţ	HO88238Q_MUXEMSPZ	.text
		Ĵ	VBBmScQI_W7JL3RN3	.text			J	HU88Z38Q_CDS_JXK6	.text
		f	yBBmScQI_iriD4Vpz	.text			ţ	HO88Z38Q_Ckalq0WC	.text
		f	sync Once doSlow dwrap 1 53	.text			f	HO88Z38Q_BmeKdiYE	.text
		Ŧ	vBBmScOI init 0	text			f	HO88Z38Q_MWyliM43	.text
		1	PBmB-QI_init_0	text			f	HO88Z38Q_Y5vHL73Y	.text
	-	Ĵ	ABBuzco1_IUIt	.text			f	HO88Z38Q_UfDiTxaZ	.text
4		ma	in				f	H088Z38Q_UoIvvHyS	.text
		f	main_main	.text			f	H088Z38Q_G0_8Vtz0	.text
	f	rt	D_amd64	.text			f	syncOnce_doSlow_dwrap_1_42	.text
	f	cm	body	.text			f	HO88Z38Q_Vy7DSDba	.text
	Ŧ	COL	unthody.	text			f	HO88Z38Q_Vy7DSDba_dwrap_2	.text
	1.4	-00	incood,	100700			1.0	10007300	A

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

As we have <u>previously</u> 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.

<pre> loc_56DFA5: lea rax, RTYPE_string lea rbx, off_5F49C0 ; "already enc done" call runtime_gopanic</pre>
<pre>newfstatat(AT_FDCWD, "/root/showkey.txt", {st_mode=S_IFREG 0644, st_size=1708,}, 0) = 0 namecloop([tv_soc=0, tv_psoc=1000000], NULL) = 0</pre>
nanosleep({tv_sec=0, tv_nsec=1000000}, NULL) = 0
write(2, <mark>"panic: ", 7panic:) = 7</mark>
<pre>write(2, "already enc done", 16already enc done) = 16</pre>
715839a8012c1e1154a38984.exe
panic: already enc done
goroutine 1 [running]:
H088Z380.E9rNM3uj(0xc00004e300, 0xc00001e380) ullboUZo0.oo:1 +0x2f8
H088Z380.LKyfn1gq(0xc00001e380, 0xc00004e1e0)
S9zU4b7E.go:1 +0x5f main.main()
LŸj0b5Ph.go:1 +0x67

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" 	o "/root/showkey.txt"

o "C:\\ProgramData/public.txt"

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

The sample uses the <u>Golang Crypto Packages</u> for RSA key — some of them are <u>crypto x509 MarshalPKCS1PublicKey</u>, <u>crypto x509 MarshalPKCS1PrivateKey</u>, <u>encodin</u> <u>g pem EncodeToMemory and crypto rsa GenerateMultiPrimeKey</u>.

As seen in Figure 4, crypto_x509_ MarshalPKCS1PrivateKey converts the RSA private key to PKCS #1, ASN.1 DER form. Then, <u>the encoding pem EncodeToMemory</u> 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).

M 🕅	
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	<pre>rsi, aRsaPrivateKey ; "RSA PRIVATE KEY"</pre>
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 59 2D 2D 2D 2D RIVATE KEY 000000C000202020 4D 49 43 58 67 49 42 41 41 48 42 67 51 43 72	^
000000C000202000 2D 2D 2D 2D 2D 2D 2D 42 45 47 49 4E 20 52 53 41 20 50 000000C000202010 52 49 56 41 54 45 20 4B 45 59 2D 2D 2D 2D 2D 2D 0A 000000C000202020 4D 49 49 43 58 67 49 42 41 41 4B 42 67 51 43 72	
000000C000202010 52 49 56 41 54 45 20 48 45 59 2D 2D 2D 2D 2D 0A RIVATE KEY 000000C000202020 4D 49 49 43 58 67 49 42 41 41 48 42 67 51 43 72	
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 28	
000000C000202040 75 39 75 37 49 28 28 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	
000000C0002020/0 3/ /8 3/ 30 4F /3 6E /0 6/ 69 6B 5/ /6 6/ 6C 2F	
000000C00202080 64 56 72 65 42 70 30 32 58 78 34 76 63 6E 75 52	
000000000202090 4F 38 30 2F 2F 0C 7A 4E 04 42 30 09 08 74 04 48	
00000000202040 / 8 04 49 45 5/ / 8 06 44 55 50 / 4 54 06 04 / 9 46	
000000000000000000000000000000000000000	
000000C0002020E0 41 42 0A 41 6E 47 41 43 47 34 67 70 72 6A 6A 4C	
000000C0002020E0 72 6E 71 36 32 70 32 78 52 56 4C 53 6A 6E 4D 45 rng62p2x8VL510ME	
000000C000202100 4E 49 69 6F 2F 74 4D 6F 41 50 65 49 4A 4E 53 54 NIJO/TMOAPETINST	
000000C000202110 52 56 6A 62 72 62 4B 55 42 75 6B 69 6E 33 4A 54 RVibrbKUBukin3JT	
000000C000202120 61 65 59 0A 31 46 79 64 49 42 53 6D 51 59 57 64 aev.1FydIBsmQYwd	
000000c000202130 65 70 32 52 71 33 31 48 5A 55 52 63 4A 53 6B 43 ep2rg31HzURCJSkc	
000000c000202140 47 44 54 74 67 66 2F 66 45 38 58 32 64 45 71 41 GDTtgf/fE8x2dEqA	
000000C000202150 78 36 73 65 4C 43 6D 42 4C 76 45 45 67 30 76 41 x6seLCmBLvEEgOVA	
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 69P0Z. XwgP8MZGUW	
000000C0002021B0 5A 4D 4F 55 52 51 48 /A 2F 57 43 /3 33 68 41 39 ZMOURQHZ/WCS3hA9	
000000C0002021C0 41 33 51 65 47 31 77 4F 78 66 45 31 58 47 4C 77 A30eG1w0xtE1XGLw	
000000C002021D0 /1 62 /8 35 56 /2 46 31 46 28 6A /8 5/ 68 6C 35 qDX5V7F1F+JXWh15	-
00000000000000000000000000000000000000	-
	•

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 packa ges from Golang, as shown in Figure 6.

🏂 Dump 1	R	Dum	p 2		£	Dum	np 3		£	Dun	np 4		R	Dun	np 5		2	Watch 1	🔊 Locals	3	Str	∢ ►
Address		He	< .															ASCII				
0000000005c	6630	4C	53	30	74	4C	53	31	43	52	55	64	4A	54	69	42	53	LS0tLS1C	RUDJTIBS			
0000000005c	6640	55	30	45	67	55	46	56	43	54	45	6C	44	49	45	74	46	U0EgUFVC	TEIDIETE			
0000000005c	6650	57	53	30	74	4C	53	30	74	43	6B	31	4A	53	55	4A	4A	WS0tLS0t	ck135033			
0000000005c	6660	61	6B	46	4F	51	6D	64	72	63	57	68	72	61	55	63	35	akFOQmdr	cWhraUc5			
0000000005C	6670	64	7A	42	43	51	56	46	46	52	6B	46	42	54	30	4E	42	dzBCQVFF	RKFBTONB			
0000000005c	6680	55	54	68	42	54	55	6C	4A	51	6B	4E	6E	53	30	4E	42	UThBTU]]	QKNNSONB			
0000000005C	6690	55	55	56	42	65	58	6B	34	4C	31	4E	35	63	48	46	6B	UUVBeXk4	L1N5CHFk			
0000000005C	66A0	4B	30	56	32	4E	32	35	55	63	32	56	58	62	32	73	4B	K0V2N25U	c2vxb2sK			
0000000005c	66B0	53	48	68	56	61	32	70	68	4D	55	6C	47	4F	46	4E	52	SHhVa2ph	MUIGOFNR			
0000000005C	66C0	52	46	63	32	59	6E	4E	31	59	6A	42	6C	62	47	46	43	RFC2YnN1	YJBIbGFC			
0000000005C	66D0	53	7A	5A	32	52	6A	68	47	63	32	70	34	62	31	5A	72	SZZ2RjhG	c2p4b1Zr			
0000000005C	66E0	4F	45	35	34	52	6A	59	30	51	6C	52	31	53	6D	4E	30	OE54RJY0	QIRISMNO			
0000000005C	66F0	63	32	11	35	59	30	39	6D	56	46	6C	44	SA	33	56	35	C2W5Y09m	VF IDZ 3V5			
000000000050	6700	DA 5A	48	42	22	22	6/	6F	18	65	22	78	4D	2/	22	39	6B	ZHBRUGOX	eUXMWU9K			
000000000050	6720	22	19	39	47	24	00	22	43	04	30	39	69	24	47	74	20	RYSGIMRC	dugingep			
000000000050	6720	61	OB	20	40	03	40	4E	55	22	40	OC C	DA 6F	10	44	22	54	eKVLCEN3	DF IZ aD04			
000000000050	6740	67	40	04	22	40	32	40	20	64	05	40	DE	4E	33	22	23	AFORL2FP	dCT4dpEc			
0000000000000000	6750	62	20	45	54	24	45	46	51	51	41	49	60	42	OE CP	40	10	DUNTWEGY				
000000000000000000000000000000000000000	6760	50	6A	42	57	56	22	40	76	61	12	70	6P	45	21	40	20	WIIBVUZEV	AND MICKED			
000000000000000000000000000000000000000	6770	50	64	56	16	10	19	40	22	5.4	40	16	40	64	60	20	21	VIVENHM2				
000000000050	6780	61	6B	38	30	55	30	34	30	52	60	56	54	62	68	54	77	ak800040	SmVThkZw			
000000000050	6790	57	60	46	46	63	31	45	53	63	6B	34	79	55	58	46	44	WINNC1NS				
000000000050	6740	4R	32	44	46	61	79	74	46	55	32	39	71	64	60	44	36	K210avtN	1290dm16			
000000000050	67B0	63	45	6F	4R	54	32	68	78	62	64	56	70	61	55	56	61	CEOKT2hx	hivnallva			
000000000050	67C0	59	55	46	51	40	60	44	34	40	56	54	53	55	6B	77	35	YUFOMm14	MVZSUkw5			
00000000005c	67D0	65	48	4D	77	65	48	49	35	4E	54	52	4C	56	74	49	72	eHMweHT5	NTRLVZTr			
00000000005c	67E0	55	6E	6C	71	5A	33	52	30	4B	30	74	58	56	48	46	61	Unlaz 3RO	KOTXVHFa			
00000000005c	67F0	56	48	46	4C	57	47	67	31	51	54	64	79	4E	7A	4E	69	VHFLWGq1	OTdVNZNi			
0000000005c	6800	51	57	4E	79	57	69	39	4E	64	77	70	72	56	44	4A	76	QWNVWi9N	dwprVDJV			
0000000005c	6810	4E	6E	55	77	55	57	4A	31	4F	56	4A	55	62	54	4A	42	NNUWUW31	оузиртзв			
0000000005c	6820	64	55	64	54	55	45	52	55	52	7A	4E	4E	62	57	6C	49	dudtueru	RZNNbwli			
0000000005c	6830	62	57	6C	78	63	46	5A	75	54	48	6C	74	51	57	35	4D	bwlxcFZu	THITQW5M			
0000000005c	6840	4F	56	6C	47	55	46	56	57	51	55	56	73	63	45	46	79	OVIGUEVW	QUVSCEFY			
0000000005c	6850	55	48	46	75	4C	7A	4A	43	61	54	68	57	5A	46	52	6F	UHFULZJC	aThwZFRO			
0000000005c	6860	43	6D	56	52	53	55	52	42	55	55	46	43	43	69	30	74	CmVRSURB	UUFCCi0t			
0000000005c	6870	4C	53	30	74	52	55	35	45	49	46	4A	54	51	53	42	51	LSOTRU5E	IFJTQSBQ			
0000000005c	6880	56	55	4A	4D	53	55	4D	67	53	30	56	5A	4C	53	30	74	VUJMSUMg	50VZLSOt			Ŧ
•			-	-		111	-	-								2.					Þ	



call enc call enc

encoding_base64__ptr_Encoding_DecodeString
encoding_pem_Decode

ſ

																					_
🏂 Dump 1	2	Dum	p 2		Å	Dum	р 3		R	Dun	np 4		R	Dur	np 5		2	Watch 1	🐨 Local	s 🛛 🐉 Str	4
Address		Hex	C															ASCII			
000000c00020	3FD6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				-
000000c00020	3FE6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				
000000c00020	3FF6	00	00	00	00	00	00	00	00	00	00	2D	2D	2D	2D	2D	42		B		
000000c00020	4006	45	47	49	4E	20	52	53	41	20	50	55	42	4C	49	43	20	EGIN RSA	PUBLIC		
000000c00020	4016	4B	45	59	2D	2D	2D	2D	2D	0A	4D	49	49	42	49	6A	41	KEY	MIIBIIA		
000000c00020	4026	4E	42	67	6B	71	68	6B	69	47	39	77	30	42	41	51	45	NBakahki	G9W0BAQE		
000000c00020	4036	46	41	41	4F	43	41	51	38	41	4D	49	49	42	43	67	4B	FAAOCAQ8	AMIIBCOK		
000000c00020	4046	43	41	51	45	41	79	79	38	2F	53	79	70	71	64	2B	45	CAQEAVV8	/Sypad+E		
000000c00020	4056	76	37	6E	54	73	65	57	6F	6B	0A	48	78	55	6B	6A	61	v7nTseWo	k.Hxukja		
000000c00020	4066	31	49	46	38	53	51	44	57	36	62	73	75	62	30	65	6C	1IF8SQDW	6bsub0e1		
000000c00020	4076	61	42	4B	36	76	46	38	46	73	6A	78	6F	56	6B	38	4E	aBK6vF8F	sixovk8N		
000000c00020	4086	78	46	36	34	42	54	75	4A	63	74	73	6C	39	63	4F	66	xF64BTuJ	cts19cof		
000000c00020	4096	54	59	43	67	75	79	64	70	51	52	0A	31	79	4C	4C	59	TYCguydp	QR.1YLLY		
000000c00020	40A6	4F	64	47	2F	46	4E	64	42	77	4F	62	4C	6B	69	7A	45	OdG/FNdB	wObLkizE		
000000c00020	40B6	4B	70	53	77	50	59	59	68	35	38	68	57	51	2F	61	4F	KpSwPYYh	58hwq/a0		
000000c00020	40C6	6E	49	67	37	74	52	6F	43	53	58	48	32	74	62	38	76	nIg7tRoC	SXH2tb8v		
000000c00020	40D6	71	6C	5A	70	55	53	31	50	42	56	61	0A	41	41	62	32	qlzpus1P	BVa.AAb2		
000000c00020	40E6	56	57	71	2F	68	7A	64	33	54	34	62	35	45	30	73	36	vwg/hzd3	T4b5E0s6		
000000c00020	40F6	64	71	4C	76	6F	75	6A	4F	34	53	4E	34	4A	65	53	6E	dqLvouj0	4SN4JeSn		
000000c00020	4106	46	70	5A	53	4D	73	53	52	72	4E	32	51	71	49	2B	62	FpZSMsSR	rN2QqI+b		
000000c00020	4116	4E	6B	2B	4D	53	6F	6A	76	62	7A	70	4A	0A	4F	68	71	Nk+MSojv	bzpJ.Ohq		
000000c00020	4126	6E	35	69	69	45	5A	61	41	50	32	62	78	31	56	52	52	n5iiEZaA	P2bx1VRR		
000000c00020	4136	4C	39	78	73	30	78	72	39	35	34	4B	57	32	2B	52	79	L9xs0xr9	54KW2+Ry		
000000c00020	4146	6A	67	74	74	2B	4B	57	54	71	5A	54	71	4B	58	68	35	jgtt+KWT	qZTqKXh5		
000000c00020	4156	41	37	72	37	33	62	41	63	72	5A	2F	4D	77	0 A	6B	54	A7r73bAc	rZ/Mw.kT		
000000c00020	4166	32	бF	36	75	30	51	62	75	39	52	54	6D	32	41	75	47	206u0Qbu	9RTm2AuG		
000000c00020	4176	53	50	44	54	47	33	4D	6D	69	48	6D	69	71	70	56	6E	SPDTG3Mm	iHmiqp∨n		
000000c00020	4186	4C	79	6D	41	6E	4C	39	59	46	50	55	56	41	45	6C	70	LymAnL9Y	FPUVAElp		
000000c00020	4196	41	72	50	71	6E	2F	32	42	69	38	56	64	54	68	0A	65	ArPqn/2B	i8vdTh.e		
000000c00020	41A6	51	49	44	41	51	41	42	0A	2D	2D	2D	2D	2D	45	4E	44	QIDAQAB.	END		
000000c00020	41B6	20	52	53	41	20	50	55	42	4C	49	43	20	4B	45	59	2D	RSA PUB	LIC KEY-		
000000c00020	41C6	2D	2D	2D	2D	0A	00	00	00	00	00	00	00	00	00	00	00				
000000c00020	41D6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				
000000c00020	41E6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				
000000c00020	41F6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				
000000c00020	4206	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00				
000000000000000000000000000000000000000	4216	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		and the second second		

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:

"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

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":

		-				🗾 🖆 🖼	
🏂 Dump 1	💂 Dump 2	🏂 Dump 3	🏂 Dump 4	🏂 Dump 5 🛛 🖁	🖗 Watch 1 🔮 Locals 🛛 🖉 Str	sub mov	rsp, 50h [rsp+50h+var_8], rbp
Address	Hex				ASCII	lea	<pre>rbp, [rsp+50h+var_8] cmunto_v500_ManshalBKCS1DublicKov</pre>
000000c000102p	00 2D 2D 2	D 2D 2D 42 4	15 47 49 4E 2	0 52 53 41 20 50	BEGIN RSA P	- 102	rdy [ren±50b±var 38]
000000c000102p	10 52 49 5	6 41 54 45	0 4B 45 59 2	D 2D 2D 2D 2D 04	A RTVATE KEY	movups	vmmuord ntr [rdv] vmm15
000000000000000000000000000000000000000	20 40 49 4	7 44 41 65	7 42 41 40 3	4 6C 4C 50 2E 66		lea	rsi [rsn+50b+var 28]
000000000000000000000000000000000000000	20 28 54 6	6 47 72 75	8 68 68 47 2	P AC 22 6C 24 79	El I	movups	xmmword ptr [rsi], xmm15
000000000000000000000000000000000000000	40 73 74 5	2 65 54 55	50 00 00 47 2	7 52 54 55 41 50		lea	rsi, [rsp+50h+var 18]
000000000000000000000000000000000000000	50 22 28 6	5 05 JA 55	79 74 21 20 4	C 72 62 76 25 64		movups	xmmword ptr [rsi], xmm15
00000000001020	50 04 52 7	7 41 52 62			*	lea	rsi, aRsaPrivateKey ; "RSA PRIVATE KEY"
00000000001020	00 UA 53 /	7 41 52 62	37 ZF 41 30 0	7 77 57 40 42 39		mov	[rsp+50h+var_38], rsi
00000000001020	0/0 ZF 36 3	7 32 28 55	1 /8 39 42 6	1 59 35 46 77 45		mov	[rsp+50h+var_30], 0Fh
000000C000102D	80 42 63 /	3 63 63 66	3 52 48 52 7	4 42 64 6E 6F 4A		mov	[rsp+50h+var_20], rax
000000C000102D	90 35 6E 6	E 44 32 56	51 35 6A 39 6	9 73 72 79 43 4A	A 5nnD2VQ5j9isryCJ	mov	[rsp+50h+var_18], rbx
000000c000102D	0AO 36 OA 5	5 4E 66 6F	73 46 53 4F 5	A 4C 41 74 56 54	1 6.UNfosFSOZLAtVT	mov	[rsp+50h+var_10], rcx
000000c000102D	BO 66 58 6	8 71 45 30	57 57 54 4E 4	1 2B 47 30 44 35	5 fxhqE0wwTNA+G0D5	mov	rax, rdx
000000c000102D	CO 65 48 5	5 64 31 78	51 44 53 34 6	9 46 58 6A 31 4F	eHUd1xQD54iFXj10	nop	dword ptr [rax+rax+00h]
000000c000102p	DO 6C 62 5	4 52 6C 44 4	1 67 4D 42 4	1 41 45 3D 0A 2D	D 1btr1DAgMBAAE=	call	encoding_pem_EncodeToMemory
000000c000102p	EO 2D 2D 2	D 2D 45 4E	4 20 52 53 4	1 20 50 52 49 56	5END RSA PRTV	mov	rcx, rbx
00000000001020	FO 41 54 4	5 20 4B 45	9 2D 2D 2D 2	D 2D 0A 00 00 00	ATE KEY	mov	rbx, rax
00000000001025	00 00 00 0					xor	eax, eax
00000000000000002020					S	call	runtime_slicebytetostring
000000000000000000000000000000000000000						mov	rbp, [rsp+50h+var_8]
000000000000000000000000000000000000000	20 00 00 0					add	rsp, søn
000000000000000002E						retn	

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 <u>os exec command</u> 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 " 	 "service mysql stop" "/etc/init.d/mysqld stop" "service oracle stop" "systemctl disable \"postgresql*\"" "systemctl disable \"mysql*\"" "systemctl disable \"oracle*\""

0	"taskkill /f /im agntsvc.exeisqlplussvc."
0	"taskkill /f /im xfssvccon.exe "
0	"taskkill /f /im mydesktopservice.exe "
0	"taskkill /f /im ocautoupds.exe "
0	"taskkill /f /im agntsvc.exeagntsvc.exe
0	"taskkill /f /im agntsvc.exeencsvc.exe
0	"taskkill /f /im firefoxconfig.exe
0	"taskkill /f /im tbirdconfig.exe "
0	"taskkill /f /im ocomm.exe "
0	"taskkill /f /im mysqld.exe "
0	"taskkill /f /im mysqld-nt.exe "
0	"taskkill /f /im mysqld-opt.exe "
0	"taskkill /f /im dbeng50.exe "
0	"taskkill /f /im sqbcoreservice.exe "
0	"taskkill /f /im excel.exe "
0	"taskkill /f /im infopath.exe "
0	"taskkill /f /im msaccess.exe "
0	"taskkill /f /im mspub.exe "
0	"taskkill /f /im onenote.exe "
0	"taskkill /f /im outlook.exe "
0	"taskkill /f /im powerpnt.exe "
0	"taskkill /f /im steam.exe "
0	"taskkill /f /im sqlservr.exe "
0	"taskkill /f /im thebat.exe "
0	"taskkill /f /im thebat64.exe "
0	"taskkill /f /im thunderbird.exe "
0	"taskkill /f /im visio.exe "
0	"taskkill /f /im winword.exe "
0	"taskkill /f /im wordpad.exe"
0	"taskkill /f /im tnslsnr.exe"

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 ; "ABCDEFGHIJKLMNOPQRSTUVWXYZ'
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 EFI.Microsoft Windows Program Files 	 /bin /boot /sbin /tmp

0	All Hears	0	/etc
0		0	/11
0	Boot	0	/11b
0	IEidcache	0	/proc
0	ProgramData	0	/dev
0	desktop.ini	0	/sys
0	autorun.inf	0	/usr/include
0	netuser.dat	0	/usr/java
0	iconcache.db		
0	thumbs.db		
0	Local Settings		
0	bootfont.bin		
0	System Volume Information		
0	AppData		
0	Recycle.Bin		
0	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, 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, fxg, 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, jspx, 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, pag, 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, vhdx, 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, xlsm, xlsx, xlt, xltm, xltx, xlw, xml, ycbcra, 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 encryptd 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 decrytion tool . else you can delete your encrypted data or reinstall your system.

Your personid :

wVpNQcCHvOWGdNdDaOSoyus4zAqE5egyi6BOiYHZWFz/p7Q3zN0BsY7PrfbrQtOp5IQR2R05/h4THwJ5rDQcpvrGdLr/6vxLby2ZGukPy+pz9vOzxE0KWkjWJ/VDbHCVnyr5CHpLdtGycePFX+pAAqCUxyrNgU676USwTUihAcxRMAzDyFZuCfQ)V6ao2r40MzfSB2Q+k9gvi3eE3m1855qp6AxBaJZ+VdQHCekxWvCvRp3EkeDA3vHEWWCjnoQ5lnskNf69r1P9GU5IWrwir78rGlp0fuRN7CFARQ984M/gWhVNBJozIR9grOkW7DMQy1i6Tr2Sv4u9Zzn8GzbhwFi78NWKqjv71EAuZVRpnMNIFpUefTEraF2ulXtUoDVhjn8GpbB3IG4YWoLk0ZvRFiT0pzgELGhCvPHsO0crsotb/5IMX1Nd1bU1DA681nW85GUv5ENaquQRSaczCU84YWvdcF+nF98gzpsXxEFOVTkQh94dwWEAYy8JcNm9TMLxpY4FrGga/L1AXUkfcJlyHDNf7Dv+biDJwrbjefQxkBnWwGaDmdcRKvbuEUT10bCLWdxByiX63Yl31SLbP2Z71FM7QovvCu/zhIg9YT4jTT6PDeCZKV4fndK/(A1DvNRJ7IRc15R0ZRJFxZCKCMNP+8DnuC5RaJF/EoEY57Y5231oQerjW1qWi8hDGqzzm33U oJeQRIRZHHkH0HD6BFxGV0Aq7flosdIrgv/PAFDw3UZJFqmSeqDN1pGIVzNtE411WwkNicMYPq2By9PQID2Ag2+2RA2wvq7xLliRmdDNMJs1GtllhvIKQ:

Decrytion do as follows:

if you not own bitcoin, you can buy it online on some websites. like https://localbitcoins.net/ or https://www.coinbase.com/.
 send 0.05 btc to my wallet address bc1qqxck7kpzgvud7v2hfyk55yr45fnml4rmt3jasz.
 send your btc transfer screenshots and your persionid to my email service@goodluckday.xyz.i will send you decrytion tool.

Tips:

1.don't rename your file2.you can try some software to decrytion . but finally you will kown 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 decrytion tools.

Anything you want to help , please send mail to my email $\mathsf{service}@\mathsf{goodluckday.xyz}.$ Have a nice day .

Figure 9. TellYouThePass ransom note (Click to enlarge)

CrowdStrike Falcon Protection

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 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/Hos t	sha256
Window	460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a3
s	8984

Linux	5c8710638fad8eeac382b0323461892a3e1a8865da3625403769a4378622 077e
Window s 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	Scheduled Task/Job
	Privilege Escalation	
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: <u>Golang Malware Is More than a Fad:</u> <u>Financial Motivation Drives Adoption</u>
- Learn about another ransomware variant that uses a Golang packer: <u>New Ransomware</u> <u>Variant Uses Golang Packer</u>
- Visit the product website to learn how the powerful <u>CrowdStrike Falcon platform</u> provides comprehensive protection across your organization, workers and data, wherever they are located.
- <u>Get a full-featured free trial of CrowdStrike Falcon Prevent</u>[™] and see how true next-gen AV performs against today's most sophisticated threats.