



# 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 [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 —

460b096aaef535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a38984 (Windows )

5c8710638fad8eeac382b0323461892a3e1a8865da3625403769a4378622077e (Linux)

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

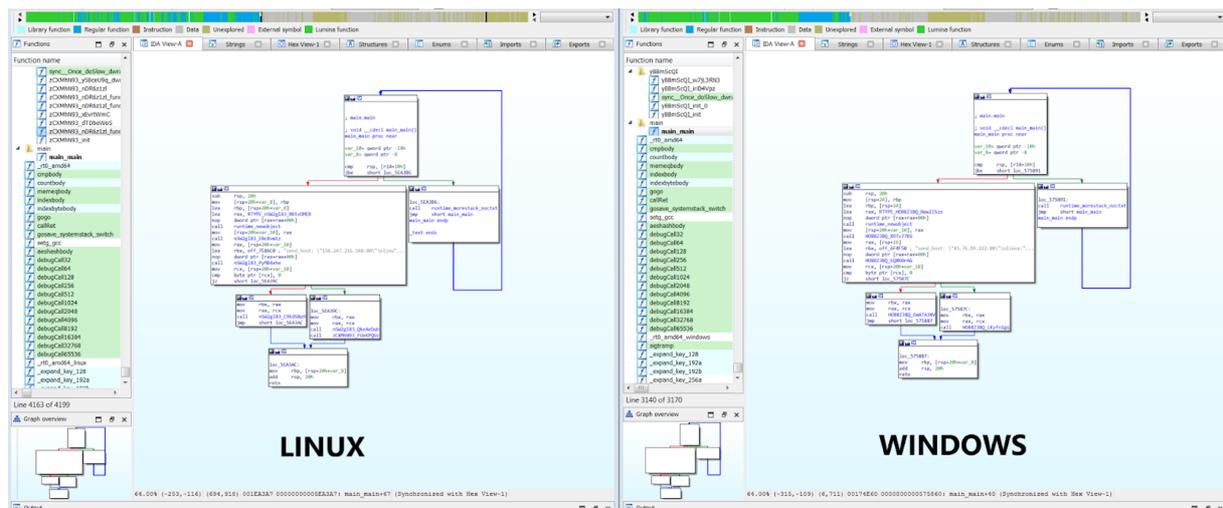


Figure 1. The “main.” functions for both Windows and Linux samples are almost identical (Click to enlarge)

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

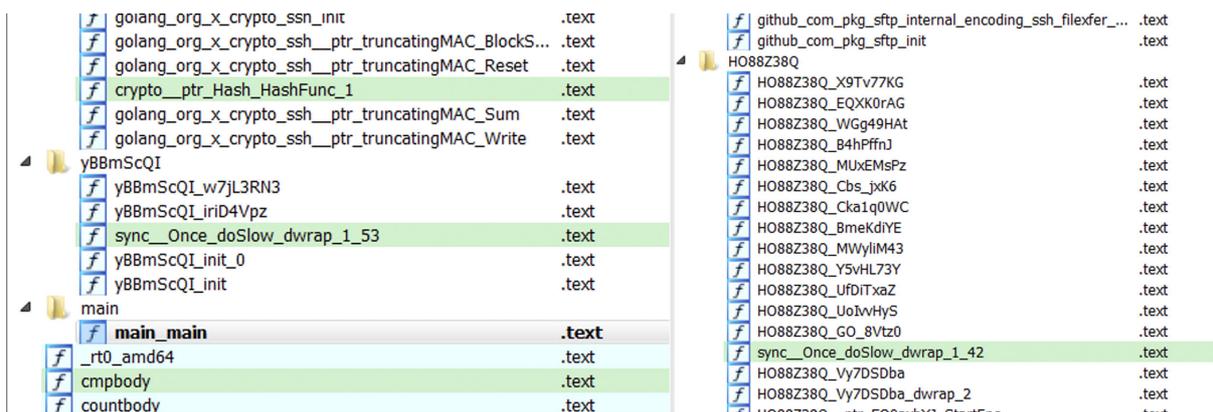


Figure 2. TellYouThePass ransomware functions for the Windows sample in IDA Pro (Click to enlarge)

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.

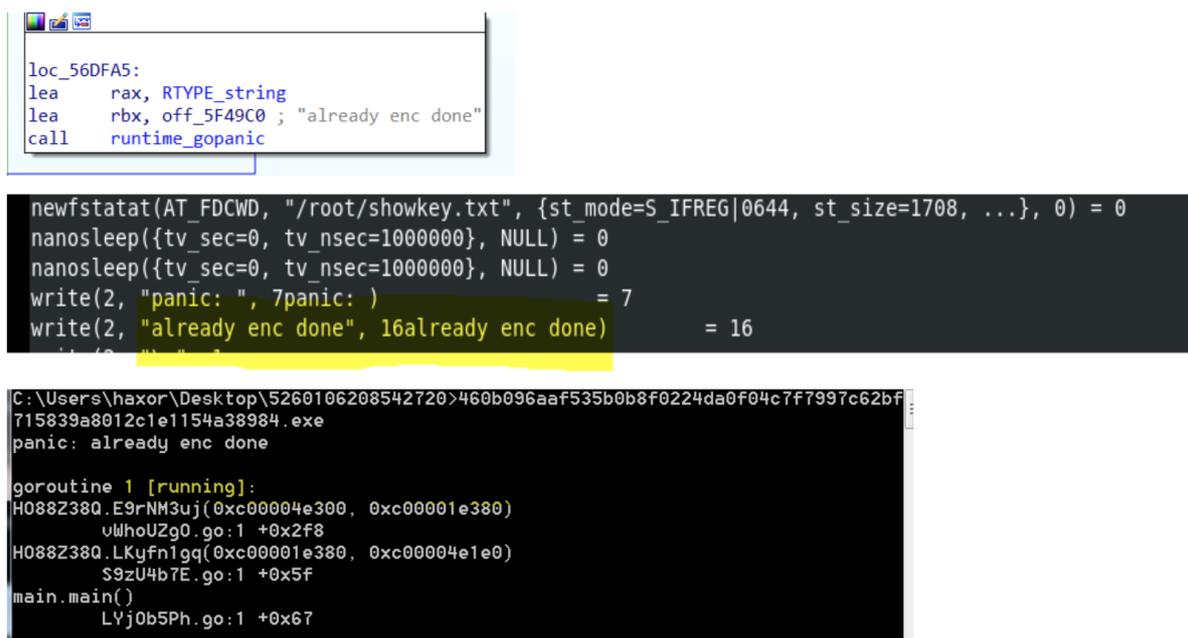


Figure 3. Encryption function followed by successful encryption for both Linux and Windows (Click to enlarge)

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
o “C:\ProgramData/showkey.txt”	o “/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 4B 45 59 2D 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	4F 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	rnq6zpzXRVL5j0ME
000000C000202100	4E 49 69 6F 2F 74 4D 6F 41 50 65 49 4A 4E 53 54	NIjo/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/fE8X2dEqA
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/H59azxHlgvt
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 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	A3QeG1w0xfE1XGLw
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

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.



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:

“ABCDEF GHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/”

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”:

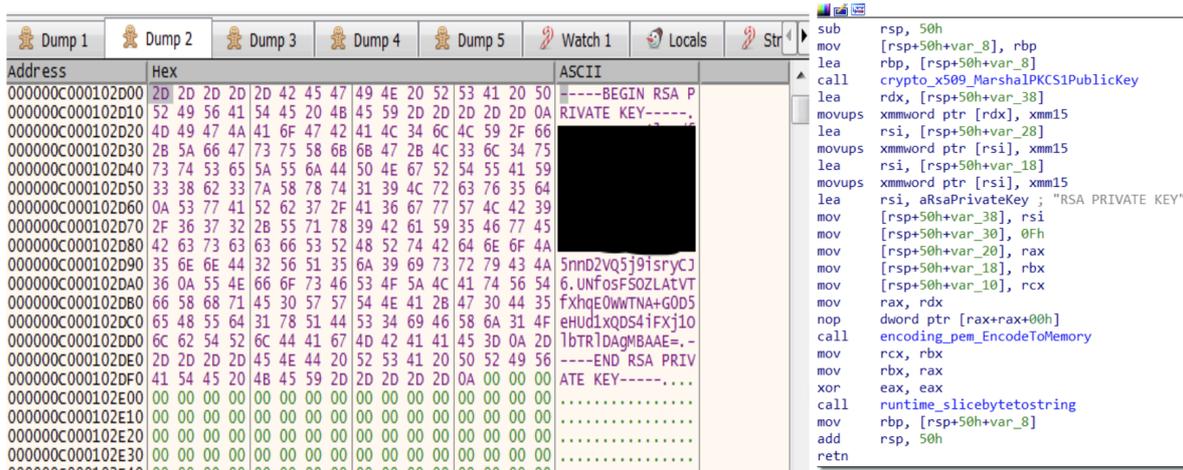


Figure 7. Key generation function (Click to enlarge)

## 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
<ul style="list-style-type: none"> <li>“taskkill /f/im msftesql.exe “</li> <li>“schtasks /delete /tn WM /F “</li> <li>“taskkill /f/im sqlagent.exe “</li> <li>“taskkill /f/im sqlbrowser.exe “</li> <li>“taskkill /f/im sqlservr.exe “</li> <li>“taskkill /f/im sqlwriter.exe “</li> <li>“taskkill /f/im oracle.exe “</li> <li>“taskkill /f/im ocssd.exe “</li> <li>“taskkill /f/im dbsnmp.exe “</li> <li>“taskkill /f/im synctime.exe “</li> <li>“taskkill /f/im mydesktopqos.exe “</li> </ul>	<ul style="list-style-type: none"> <li>“service mysql stop”</li> <li>“/etc/init.d/mysqld stop”</li> <li>“service oracle stop”</li> <li>“systemctl disable \”postgresql*\””</li> <li>“systemctl disable \”mysql*\””</li> <li>“systemctl disable \”oracle*\””</li> </ul>

<ul style="list-style-type: none"> <li>○ “taskkill /f /im agntsvc.exeisqlplussvc.”</li> <li>○ “taskkill /f /im xfssvccon.exe “</li> <li>○ “taskkill /f /im mydesktopservice.exe “</li> <li>○ “taskkill /f /im ocautoupds.exe “</li> <li>○ “taskkill /f /im agntsvc.exeagntsvc.exe “</li> <li>○ “taskkill /f /im agntsvc.exeencsvc.exe “</li> <li>○ “taskkill /f /im firefoxconfig.exe “</li> <li>○ “taskkill /f /im tbirdconfig.exe “</li> <li>○ “taskkill /f /im ocomm.exe “</li> <li>○ “taskkill /f /im mysqld.exe “</li> <li>○ “taskkill /f /im mysqld-nt.exe “</li> <li>○ “taskkill /f /im mysqld-opt.exe “</li> <li>○ “taskkill /f /im dbeng50.exe “</li> <li>○ “taskkill /f /im sqbcoreservice.exe “</li> <li>○ “taskkill /f /im excel.exe “</li> <li>○ “taskkill /f /im infopath.exe “</li> <li>○ “taskkill /f /im msaccess.exe “</li> <li>○ “taskkill /f /im mspub.exe “</li> <li>○ “taskkill /f /im onenote.exe “</li> <li>○ “taskkill /f /im outlook.exe “</li> <li>○ “taskkill /f /im powerpnt.exe “</li> <li>○ “taskkill /f /im steam.exe “</li> <li>○ “taskkill /f /im sqlservr.exe “</li> <li>○ “taskkill /f /im thebat.exe “</li> <li>○ “taskkill /f /im thebat64.exe “</li> <li>○ “taskkill /f /im thunderbird.exe “</li> <li>○ “taskkill /f /im visio.exe “</li> <li>○ “taskkill /f /im winword.exe “</li> <li>○ “taskkill /f /im wordpad.exe”</li> <li>○ “taskkill /f /im tnslnr.exe”</li> </ul>	
--	--

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
<ul style="list-style-type: none"> <li>○ EFI.Boot</li> <li>○ EFI.Microsoft</li> <li>○ Windows</li> <li>○ Program Files</li> </ul>	<ul style="list-style-type: none"> <li>○ /bin</li> <li>○ /boot</li> <li>○ /sbin</li> <li>○ /tmp</li> </ul>

<ul style="list-style-type: none"> <li>○ All Users</li> <li>○ Boot</li> <li>○ IEidcache</li> <li>○ ProgramData</li> <li>○ desktop.ini</li> <li>○ autorun.inf</li> <li>○ netuser.dat</li> <li>○ iconcache.db</li> <li>○ thumbs.db</li> <li>○ Local Settings</li> <li>○ bootfont.bin</li> <li>○ System Volume Information</li> <li>○ AppData</li> <li>○ Recycle.Bin</li> <li>○ Recovery</li> </ul>	<ul style="list-style-type: none"> <li>○ /etc</li> <li>○ /lib</li> <li>○ /proc</li> <li>○ /dev</li> <li>○ /sys</li> <li>○ /usr/include</li> <li>○ /usr/java</li> </ul>
--	--

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, crawl, 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, 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 encrypted 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 :

wVpNQcCHvOWGdNdDaOSoyus4zAqE5egyi6BOiYHZWFz/p7Q3zN0BsY7PrfbrQtOp5IQR2R05/h4THwJ5rDQcpvrGdLr/6vxLby2ZGukPy+pz9vOzxE0KWRj  
WJ/6VDbHCvnyrSCHpLdtGycePFX+pAAqCUxvrNgU676USwTUiIhAcxRMazDyFZuCFqjV6ao2r40MzfSB2Q+k9vvt3eE3m1855qp6AxBaJZ+VdQHCEkxWvC  
vRp3EKEDA3vHEWWCjnoQ5lnskNf69r1P9GU51Wrwiv78rGlp0furn7CFARQ984M/gWhVNBjOzIR9grOkW7DMQy1i6Tr2Sv4u9Zzn8GzbhwFi78NWKqv71E  
AeuZVRpnMNIFpUefTEraF2ulXtUoDVhjn8GpbB3IG4YWoLk0ZvRFIT0pzgELGhCvPHsO0ersotb/SIMX1Nd1bUIDA681nW85GUv5ENaqnQRSaczCU84YWv  
dcF+nF98gzpsXxEFOVtkQh94dwWEAYy8JcNm9TMLxpY4FrGga/L1AXUkfcJlyHDNf7Dv+biDJwrjefQxkbnWwGaDmdcRKvbuEUT10bCLWdxByiX63YI3I  
SLbP2Z71FM7QovvCu/2hlg9YT4jTT6PDeCZKN4fndKe/4/tADvNRJl71Rc15ROZRJfXZCkCMNP+8DnuC5RaJbF//EoEY57Y5231oQerjW1qWi8hDGqxZmJ3D  
70WqC6xQkAInmDflevNuJTTYNtNLasQ7yfvWruobpM3e5e3c6JF24h/rXcX2R38LMrHKrMVB02glQNAEFD8ibd3HIGDXN5C7JVo2YYRMoSmRLtsngaXxv  
oJeQRIRzHHkH0HD6BFxGYOAq7flosdIrqy/PAFDw3UZJFqmSeqpDN1pGIVzNtE411WwkNicMYPq2By9PQd2Ag2+2RA2wvq7xLlilRmdDNMJs1GdlhvkKQ:

### 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 [bc1qqxck7kpgzvod7v2hfyk55yr45fnnl4rmt3jasz](https://blockchain.info/address/bc1qqxck7kpgzvod7v2hfyk55yr45fnnl4rmt3jasz).
3. send your btc transfer screenshots and your personid to my email [service@goodluckday.xyz](mailto: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 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 decryption tools.

Anything you want to help . please send mail to my email [service@goodluckday.xyz](mailto:service@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.

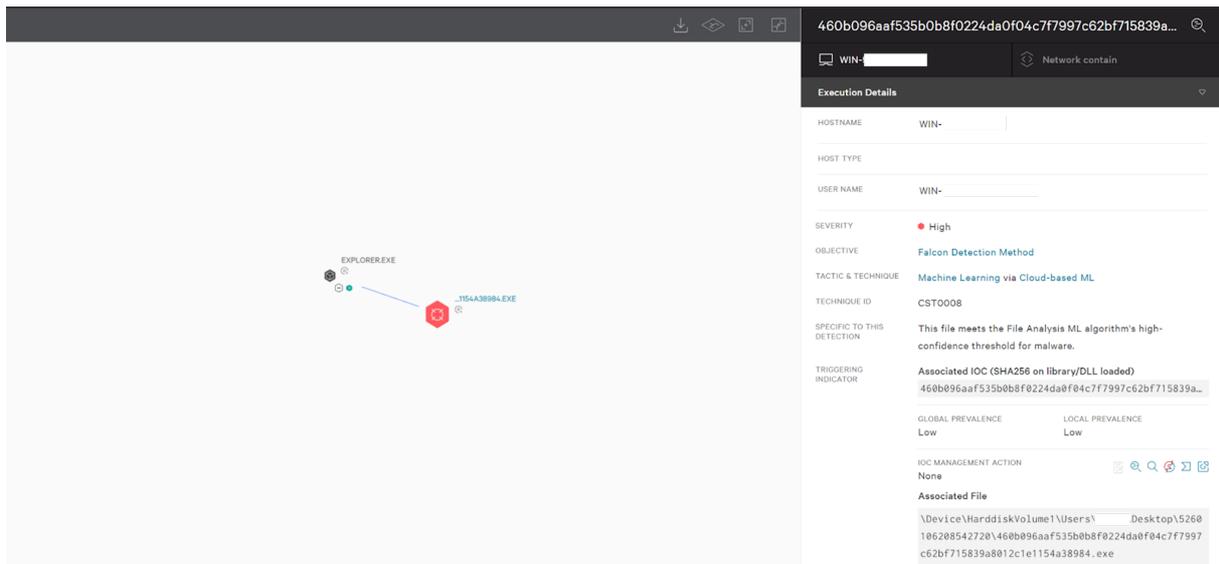


Figure 10. Falcon detection of Golang-written Windows TellYouThePass ransomware sample (Click to enlarge)

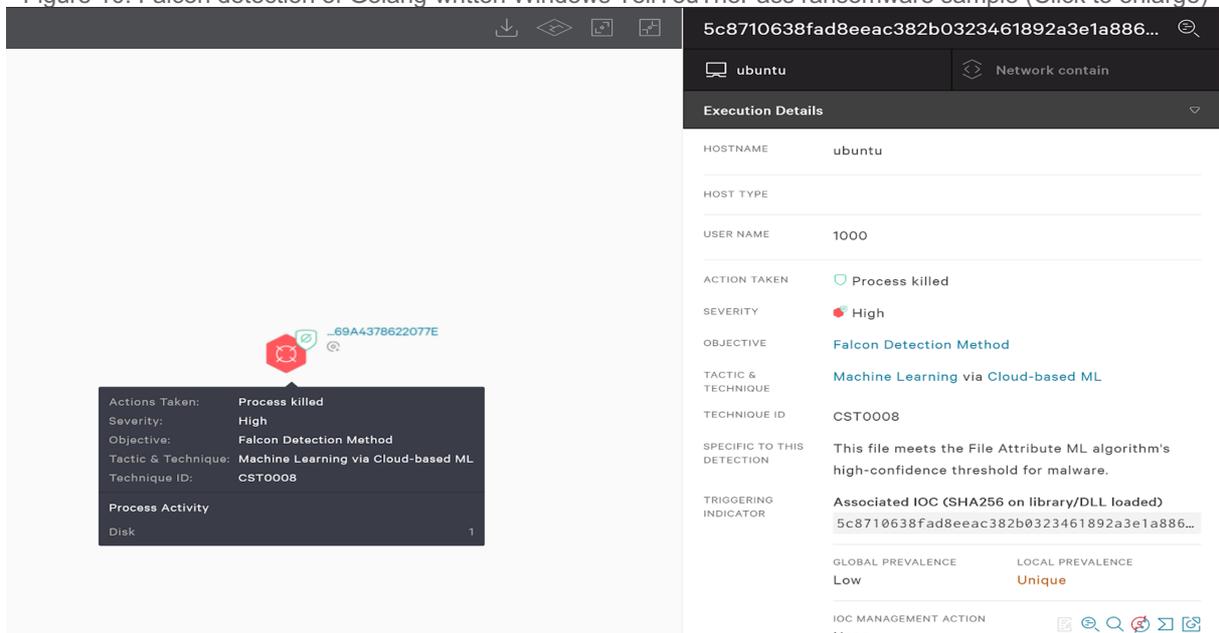


Figure 11. Falcon detection of Golang-written Linux TellYouThePass ransomware sample (Click to enlarge)

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	460b096aaf535b0b8f0224da0f04c7f7997c62bf715839a8012c1e1154a38984

Linux	5c8710638fad8eeac382b0323461892a3e1a8865da3625403769a4378622077e
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.*