BianLian Ransomware Encrypts Files in the Blink of an Eye

blogs.blackberry.com/en/2022/10/bianlian-ransomware-encrypts-files-in-the-blink-of-an-eye

The BlackBerry Research & Intelligence Team

Minutes make the difference to defenders in responding to a ransomware attack on a victim’s network. BianLian ransomware raises the cybercriminal bar by encrypting files with exceptional speed.

Threat actors built the new BianLian ransomware in the Go programming language (aka Golang). Despite the large size of files created in Go, threat actors are turning to this “exotic” programming language more often for a variety of reasons, particularly its robust support for concurrency. This is the ability for various malicious functions to run independently of each other, which speeds up the attack.

In addition to an overview of the BianLian ransomware, this post also highlights some of the unique aspects of Golang that makes it an increasingly popular choice for malware authors.

Operating System

Windows	MacOS	Linux	Android
Yes	No	No	No

Risk & Impact

Impact	High
Risk	Medium

What is BianLian?

Bian Lian is an ancient dramatic art that originates in China. Artists move about the stage in brightly colored outfits and colored masks. The performers are so quick to change the masks that, with the swipe of a fan or the blink of an eye, their costume's face completely changes. We can now add a new meaning to the term "Bian Lian," because a ransomware group took the name and made it their own.

Research from Cyble found that this threat group targets many different industry sectors. Their targets have historically included manufacturing, education, healthcare, professional services, energy, banking, financial services, and insurance (BFSI), and the entertainment industry.

As of September 20, 2022, the group's leak site includes 23 victims. BlackBerry researchers analyzed the list of victims and determined that this group targets corporations rather than specific countries. The listed victims have varied origins, including the United States, Australia, and the United Kingdom.

Why do these operators target English-speaking countries? It's likely the threat actor is financially motivated rather than politically or geographically orientated. And at this point, the group has not claimed any affiliation with any nation state or agenda.

BianLian ransomware shares its name with a malicious Android package (APK) application that was previously hosted on the Google Play™ store, but it has since been removed. This malicious application was also dubbed "BianLian" by ThreatFabric.

This identically named malware used a dropper from the Google Play store to install a malicious file from the infamous Anubis Banker Trojan. From there, the threat actor would use a messaging service to deliver command-and-control (C2) commands and steal user credentials. At the time of writing, no one has claimed any relation between these two malware families aside from the name they've been given.

Go For Speed in Malware

BianLian ransomware is written in Golang. As we discussed in a recent whitepaper, Golang is an open-source programming language designed by Google employees. The language's official first release was in March 2012, and it quickly became a mainstay language for large industry organizations such as Apple, Google, and IBM.

Golang comes with a large standard library, garbage collection cleanup, and concurrency support. Concurrency means that multiple computations can take place at the same time through a process called multithreading. To enable this, Go uses "Goroutines", which allow for asynchronous execution of functions or methods independently from each other. This concurrency allows for quicker encryption of the target system.

Go can compile code for Windows®, Linux®, and OS X®. This feature makes it possible for malware authors to create threats that impact all the major operating systems, if they choose to.

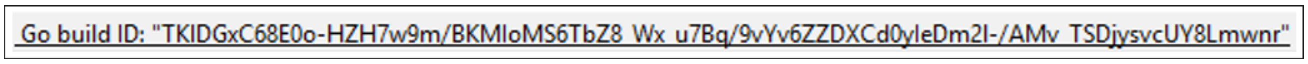
Go libraries are statically linked, which means all the necessary libraries are included in the compiled binary. In languages where this is not the case, developers will either include the libraries separately from the main executable, or they will have to hope that the target machine has the needed libraries already installed on their machine. Including these libraries makes for a larger file that is harder to distribute, but larger files might also be ignored by antivirus (AV) engines that are trying to optimize for speed.

Technical Analysis of BianLian Ransomware

The sample we will be analyzing in this post is named anabolic.exe (SHA256: 46d340eaf6b78207e24b6011422f1a5b4a566e493d72365c6a1cace11c36b28b). This file is a 64-bit executable compiled with Golang version 1.18.3.

When a Golang program is built, it generates a BuildID. A BuildID in Golang is a unique representation of the file and its contents. The BuildID is in the first 32 kb of the binary file, though the exact position can vary depending on the operating system it's compiled for. The file in focus here is a portable executable (PE) file.

Looking closer, a BuildID is comprised of a hash of the filenames and the contents of the application, and it is segmented into two parts: actionID and ContentID. The actionID is a hash of the inputs that produced the packages or binary, and the contentID is the hash of the action output. This action output is the binary itself. The buildID of anabolic.exe is shown below in Figure 1.



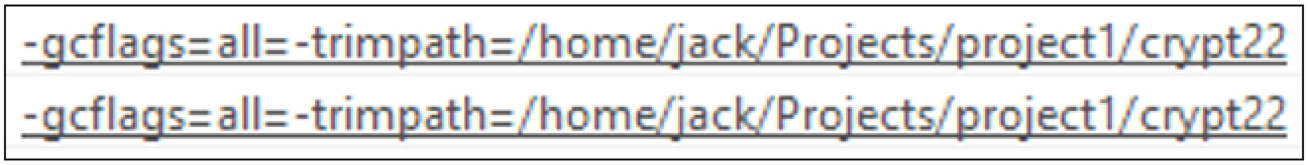
```
Go build ID: "TKIDGxC68E0o-HZH7w9m/BKMloMS6TbZ8_Wx_u7Bq/9vYv6ZZDXCd0yleDm2I-/AMv_TSDjysvcUY8Lmwnr"
```

Figure 1: Build ID from 46d340eaf6b78207e24b6011422f1a5b4a566e493d72365c6a1cace11c36b28b

The command used to compile the program is also stored in the binary. To compile a program with Go, you use the “Go build” command. This command compiles all the packages and dependencies necessary for the application.

For the sample we’re analyzing in this blog, this build path was given two arguments, shown below in Figure 2. The argument commands given are **gcflags** and **trimpath**. -Gcflags relates to flags passed to the compiler, and -trimpath removes all absolute file system paths from the executable. This is an attempt to remove user path directories.

It’s useful to note the last part of the path here: “crypt22.” This can mark iterations of development from the actor, as this number has changed between samples in the wild.

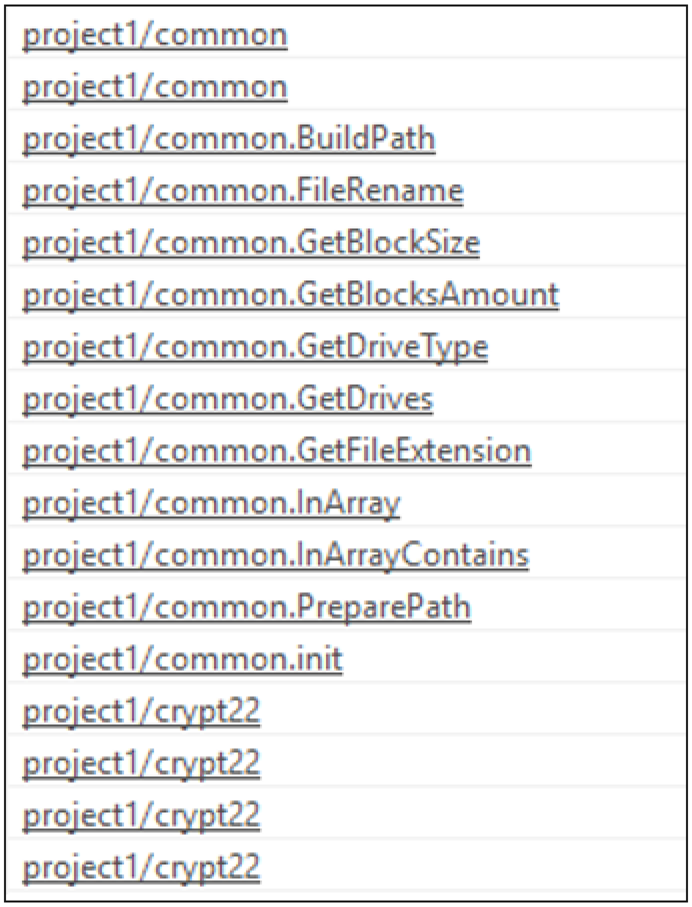


```
-gcflags=all=-trimpath=/home/jack/Projects/project1/crypt22
-gcflags=all=-trimpath=/home/jack/Projects/project1/crypt22
```

Figure 2: Crypt project path

The author packaged all the ransomware's functionalities into a common package. A package is a collection of source files in the same directory that are compiled together. Static analysis of the strings in pestudio provides us with an idea of what the sample’s capabilities are.

From this information, we can see it will likely rename files, and it can also chunk files into smaller blocks for processing, query drive information, and check file extensions. Project pathing for this functionality is shown in Figure 3.

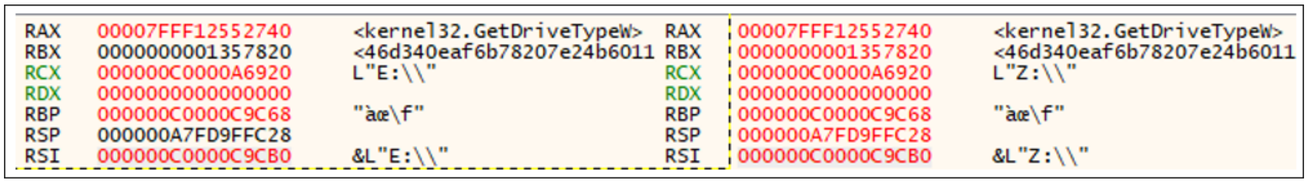


```
project1/common
project1/common
project1/common.BuildPath
project1/common.FileRename
project1/common.GetBlockSize
project1/common.GetBlocksAmount
project1/common.GetDriveType
project1/common.GetDrives
project1/common.GetFileExtension
project1/common.InArray
project1/common.InArrayContains
project1/common.PreparePath
project1/common.init
project1/crypt22
project1/crypt22
project1/crypt22
project1/crypt22
```

Figure 3. Common package functions

Upon execution of the file, the application searches the host machine for all possible drive names. To do this, the ransomware uses GetDriveTypeW from the kernell32 library. This functionality is accessed via the GetProcAddress API call, which retrieves the address of the function. The purpose of this action is to ensure that the ransomware can encrypt all connected or potential media. The call starts from "A:\\" and checks all the way to "Z:\\."

Debugger output demonstrating a snippet of this activity is shown in Figure 4.



```
RAX   00007FFF12552740    <kernel32.GetDriveTypeW>  RAX   00007FFF12552740    <kernel32.GetDriveTypeW>
RBX   0000000001357820    <46d340eaf6b78207e24b6011 RBX   0000000001357820    <46d340eaf6b78207e24b6011
RCX   000000C0000A6920    L"E:\\"                   RCX   000000C0000A6920    L"Z:\\"
RDX   0000000000000000                              RDX   0000000000000000
RBP   000000C0000C9C68    "àœ\f"                    RBP   000000C0000C9C68    "àœ\f"
RSP   000000A7FD9FFC28                              RSP   000000A7FD9FFC28
RSI   000000C0000C9CB0    &L"E:\\"                  RSI   000000C0000C9CB0    &L"Z:\\"
```

Figure 4. GetDriveTypeW call with drives

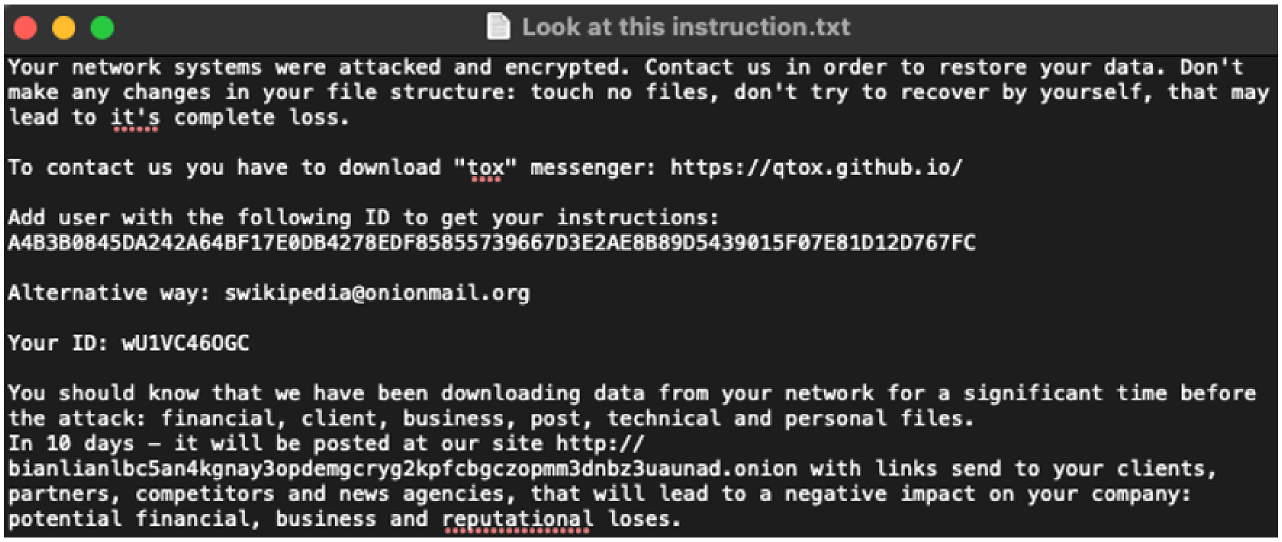
Once all the drives are populated, the threat begins its ransom process. The ransomware encrypts files using the standard library crypto package in Go. These packages are open-source libraries used to provide cryptographic functionality, like the base CryptoAPI provided in Windows environments.

BianLian Ransomware Note and Demands

The ransomware drops a .TXT file into the directory where the file was run, named “Look at this instruction.txt.” The ransom note is shown in its entirety in Figure 5.

This text file contains information regarding how to contact the threat actor to restore your data. To contact the threat actors, victims can either download the peer-to-peer encrypted chat service application Tox messenger, or they can email the group directly.

The threat actor gives their victims 10 days before publicly releasing their stolen data on the operator’s “.onion” site. The note does not indicate how much money the threat actors want for payment.



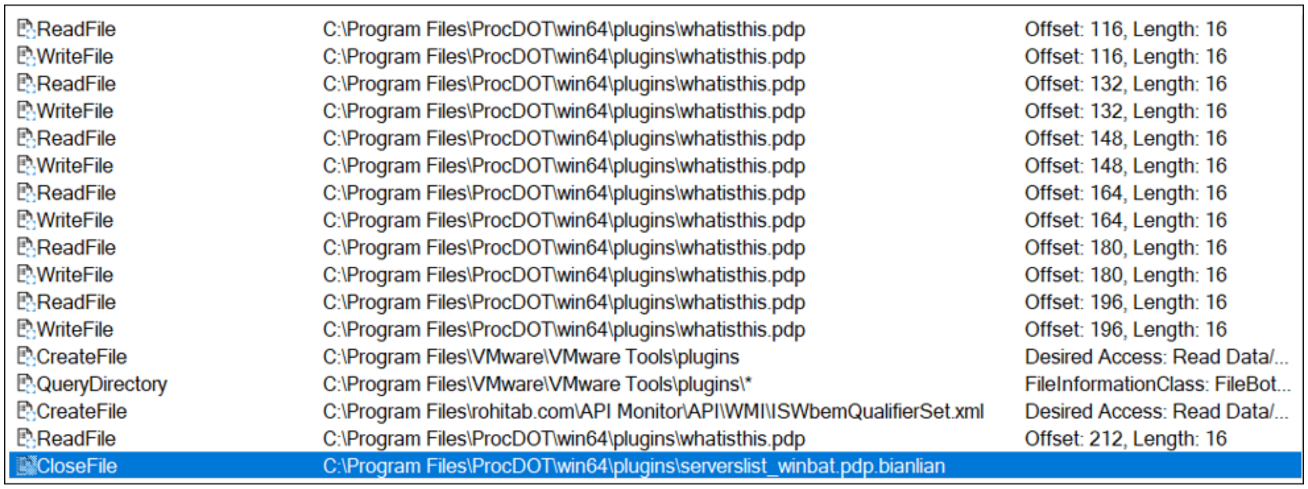
```
Your network systems were attacked and encrypted. Contact us in order to restore your data. Don't
make any changes in your file structure: touch no files, don't try to recover by yourself, that may
lead to it's complete loss.

To contact us you have to download "tox" messenger: https://qtox.github.io/

Add user with the following ID to get your instructions:
A4B3B0845DA242A64BF17E0DB4278EDF85855739667D3E2AE8B89D5439015F07E81D12D767FC

Alternative way: swikipedia@onionmail.org

Your ID: wU1VC46OGC

You should know that we have been downloading data from your network for a significant time before
the attack: financial, client, business, post, technical and personal files.
In 10 days – it will be posted at our site http://
bianlianlbc5an4kgnay3opdemgcryg2kpfcbgczopmm3dnbz3uaunad.onion with links send to your clients,
partners, competitors and news agencies, that will lead to a negative impact on your company:
potential financial, business and reputational loses.
```

Figure 5. BianLian ransom note

From here, this process is repeated in multiple threads to ensure a speedy encryption.

The ransomware targets any drive found on the system, including mounted drives, and encrypts anything that is not an executable, driver, or text file. These exclusions are meant to avoid encrypting either the ransom note, or anything that might cause the system to malfunction. The excluded files types are located within their directories by using FindFirstFileW and FindNextFileW. The lpFindFileData return value holds the information on the directory found, and FindNextFileW is used to step through the files returned with FindFirstFileW.

The files are cut into small chunks and the encryption is multithreaded to increase the speed of this operation. Figure 6 shows a snapshot of the read/write execution steps the ransomware takes to encrypt the files on the system. The output shows the read/write length of the buffer at 16 bytes, which is kept consistent throughout the execution of the file. After encryption, filenames are appended with a “.bianlian” extension and closed.



```
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 116, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 116, Length: 16
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 132, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 132, Length: 16
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 148, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 148, Length: 16
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 164, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 164, Length: 16
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 180, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 180, Length: 16
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 196, Length: 16
WriteFile       C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 196, Length: 16
CreateFile      C:\Program Files\VMware\VMware Tools\plugins                         Desired Access: Read Data/...
QueryDirectory  C:\Program Files\VMware\VMware Tools\plugins\*                       FileInformationClass: FileBot...
CreateFile      C:\Program Files\rohitab.com\API Monitor\API\WMI\ISWbemQualifierSet.xml   Desired Access: Read Data/...
ReadFile        C:\Program Files\ProcDOT\win64\plugins\whatisthis.pdp                Offset: 212, Length: 16
CloseFile       C:\Program Files\ProcDOT\win64\plugins\serverslist_winbat.pdp.bianlian
```

Figure 6. Ransomware read/write procmon

We did not observe any network interaction in this sample. This means the threat actor could have deployed the sample directly on the system with remote access.

As highlighted by cybersecurity services firm called [redacted], it is likely the BianLian threat group's initial access is gained via the ProxyShell vulnerability chain or a SonicWall VPN firmware vulnerability. From here, the threat actor moves laterally to find targets of interest, escalates their privileges, and deploys the BianLian ransomware. Then, using dropped copies of WinSCP and 7-Zip to archive and transfer chosen files, data is extracted and sent back to the threat actor. Additionally, threat operators might install backdoors on the systems to maintain access to the infected system.

Conclusion

BianLian is a relatively new threat actor that targets a wide range of industries. As they are likely financially motivated, they will continue their efforts to exploit systems and networks they gain access to. Their Golang-based ransomware utilizes goroutines and encrypts files in chunks to quickly ransom an infected system. The threat actor targets multiple industries in multiple countries. Their deployment method is manual infiltration of the system, and they use living-off-the-land (LotL) binaries to explore the networks and systems themselves. Once they find all the information they want, they deploy their ransomware.

Who is Targeted by BianLian Ransomware?

To date, this ransomware group has targeted the following industries: professional services, manufacturing, healthcare, energy, media, banks, and education sectors. Their targets thus far reside in the United States, Australia, and the United Kingdom. There is no indication that they are necessarily limited to these industries or countries.

Mitigation Tips

File Carving (D3-FC) – Use the file carving technique to examine files sent over the network.

File Access Pattern Analysis (D3-FAPA) – Identify the way an application accesses files; target could be the multiple read/writes on files that ransomware employs.

Remote Terminal Session Detection (D3-RTSD) – Detect unauthored remote sessions through network traffic.

File Creation Analysis (D3-FCA) – Ransomware creates ransom notes; this behavior can be detected.

YARA Rule for BianLian Ransomware

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
rule BianLian_Go_Ransomware{
    meta:
        description = "Detects BianLian ransomware"
        author = "BlackBerry Threat Research Team"
        date = "2022-09-13"
        license = "This Yara rule is provided under the Apache License 2.0
(https://www.apache.org/licenses/LICENSE-2.0) and open to any user or organization, as
long as you use it under this license and ensure originator credit in any derivative to the
BlackBerry Research & Intelligence Team"
    strings:
        $s1 = "trimpath=/home/jack/Projects/project1/"
        $s2 = "common.BuildPath"
        $s3 = "common.GetBlocksAmount"
        $s4 = "common.GetDrives"
        $s5 = "common.GetBlockSize"
        $s6 = "common.FileRename"
        $s7 = "common.GetFileExtension"
        $s8 = "exec.(*Cmd).Start.func1"
        $s9 = "exec.(*Cmd).Start.func2"
        $s10 = "exec.(*Cmd).Start.func3"
        $s11 = "CryptBlocks"
    condition:
        uint16(0) == 0x5a4d and all of them
}
```

Indicators of Compromise (IoCs)

46d340eaf6b78207e24b6011422f1a5b4a566e493d72365c6a1cace11c36b28b

117a057829cd9abb5fba20d3ab479fc92ed64c647fdc1b7cd4e0f44609d770ea

1fd07b8d1728e416f897bef4f1471126f9b18ef108eb952f4b75050da22e8e43

eaf5e26c5e73f3db82cd07ea45e4d244ccb3ec3397ab5263a1a74add7bbcb6e2

cbab4614a2cdd65eb619a4dd0b5e726f0a94483212945f110694098194f77095

References

Redacted blog: https://redacted.com/blog/bianlian-ransomware-gang-gives-it-a-go/
Cyble: https://blog.cyble.com/2022/08/18/bianlian-new-ransomware-variant-on-the-rise/
Go.dev documentation: https://go.dev/src/cmd/go/internal/work/buildid.go
Go Routines: https://golangbot.com/goroutines/
Go.dev Crypto Documentation: https://pkg.go.dev/crypto
Microsoft Documentation: https://learn.microsoft.com/en-us/
ThreatFabric: https://www.threatfabric.com/blogs/bianlian_from_rags_to_riches_the_malware_dropper_that_had_a_dream.html

BlackBerry Assistance

If you're battling this malware or a similar threat, you've come to the right place, regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants dedicated to handling response and containment services for a wide range of incidents, including ransomware and Advanced Persistent Threat (APT) cases.

We have a global consulting team standing by to assist you, providing around-the-clock support where required, as well as local assistance. Please contact us here: https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

Related Reading

About The BlackBerry Research & Intelligence Team

The BlackBerry Research & Intelligence team examines emerging and persistent threats, providing intelligence analysis for the benefit of defenders and the organizations they serve.

Back