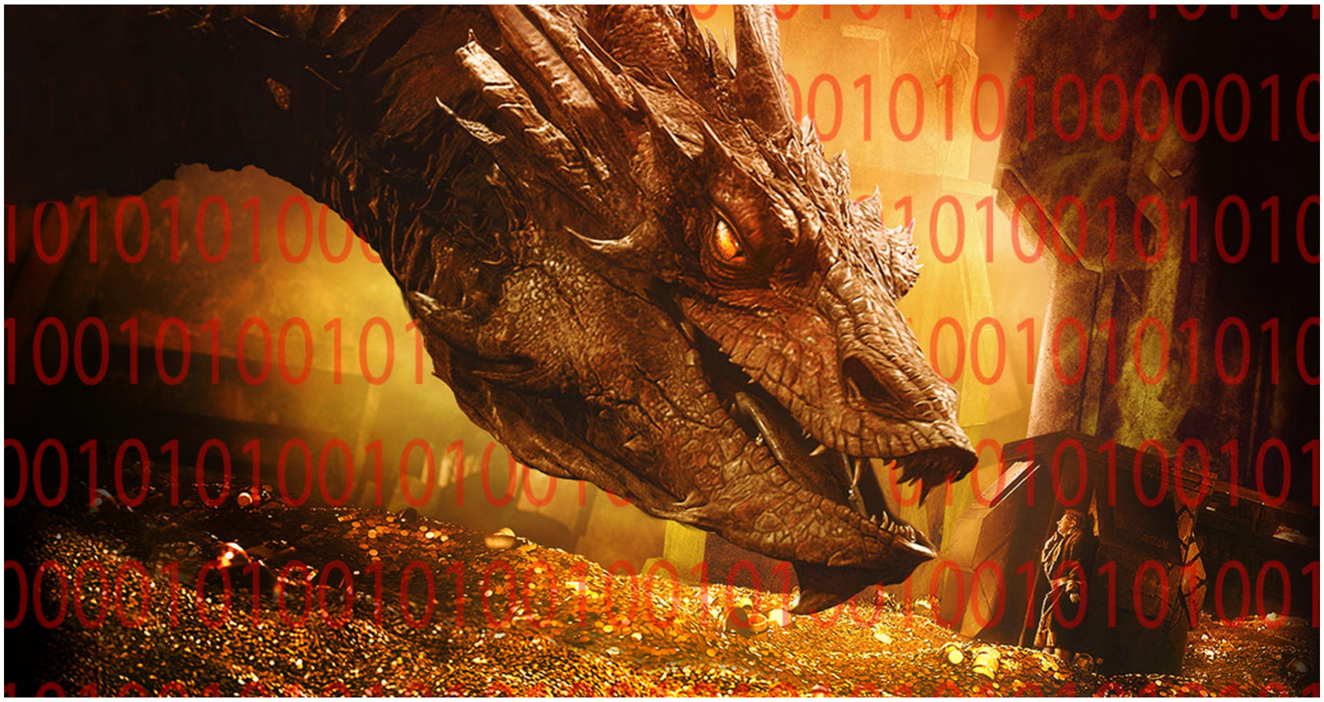
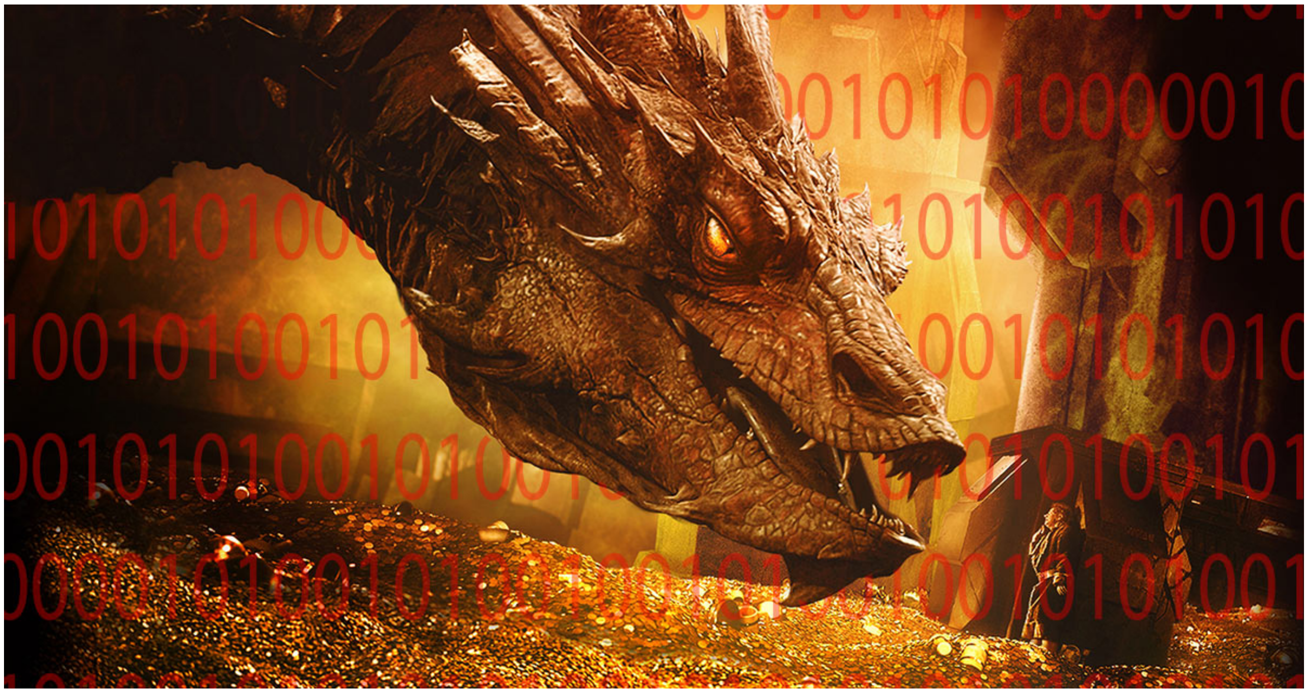
Anomali Threat Research Releases First Public Analysis of Smaug Ransomware as a Service

anomali.com/blog/anomali-threat-research-releases-first-public-analysis-of-smaug-ransomware-as-a-service



Research | August 10, 2020

by Anomali Threat Research



Authored by: Joakim Kennedy and Rory Gould

Anomali ThreatStream customers can find Indicators of Compromise (IOCs), signatures, and more information about this threat here.

Introduction

Threat actors and cybercriminals that don't have the ability to develop their own ransomware for malicious campaigns can turn to the Smaug Ransomware as a Service (RaaS) offering, which is available via a Dark Web Onion site. At least two threat actors are operating the site, providing ransomware that can be used to target Windows, macOS, and Linux machines. The site is built with ease of use in mind. To launch an attack, threat actors simply need to sign up, create a campaign, and then start distributing the malware. The site also handles decryption key purchasing and tracking for victims.

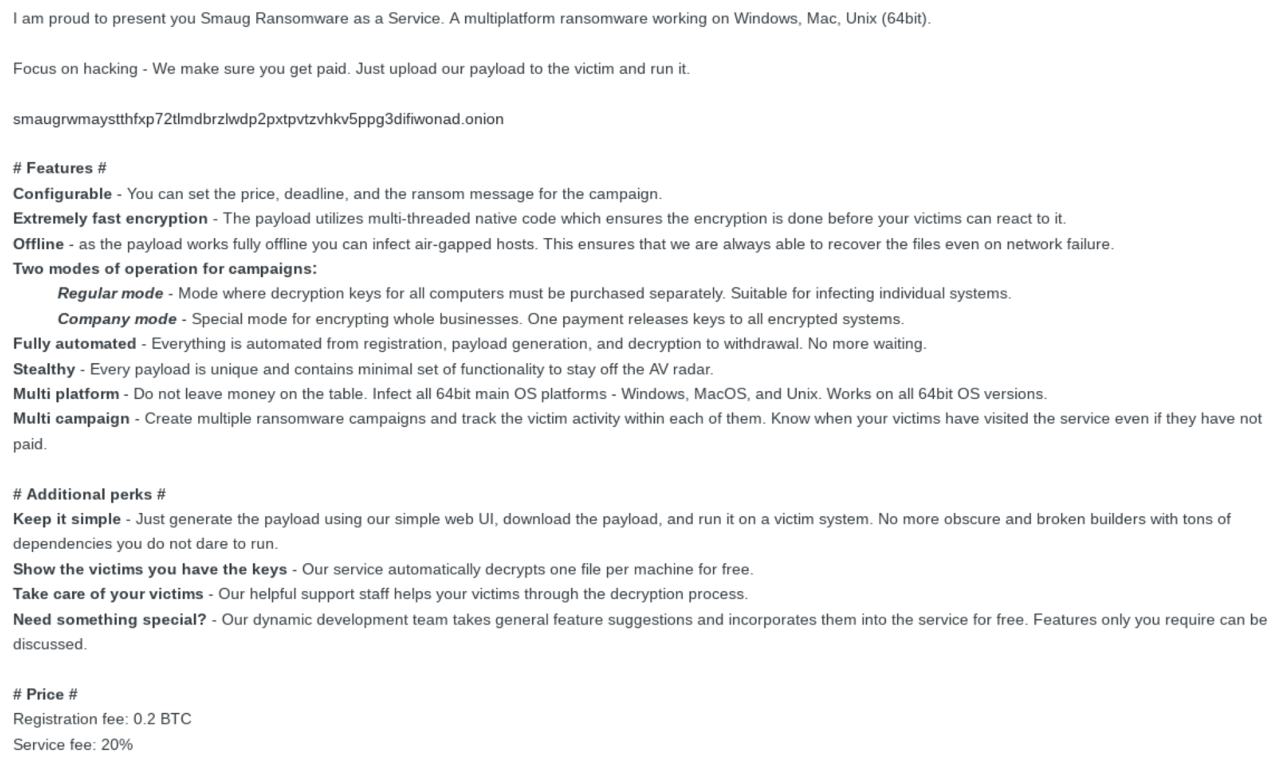
Threat Actor

The Smaug RaaS appears to be operated by one to two threat actors that are both active on the criminal underground. One of the threat actor's online handles on the forums has been identified. The second operator still is unknown.

Forum Activity

On May 5, 2020, an actor named **corinda** posted on the 'Exploit.in' forum advertising a new RaaS dubbed 'Smaug.' The post (figure 1.) showcased Smaug's features and included screenshots of the Smaug UI, this is detailed in the **Panel** section below. The post also directed users as to how they could avail of the service: contacting smaug-ransomware@protonmail.com with a registration fee of 0.2 BTC (appx $1,900 (USD) at time of posting) and subsequent service fees of 20%. The actor was willing to waive the registration fee for the first five users who could demonstrate their skills with the product; this was likely to garner reputation on the forum given that **corinda's** reputation score was zero.

Corinda continued to interact with posters before the topic was locked by a forum admin on May 14, as **corinda** had not deposited money on the forum. There has been no subsequent activity from **corinda** and the profile has not been active since June 6.


I am proud to present you Smaug Ransomware as a Service. A multiplatform ransomware working on Windows, Mac, Unix (64bit).

Focus on hacking - We make sure you get paid. Just upload our payload to the victim and run it.

smaugrwmaystthfxp72tlmdbrzlwdp2pxtpvtzvhkv5ppg3difiwonad.onion

# Features #
Configurable - You can set the price, deadline, and the ransom message for the campaign.
Extremely fast encryption - The payload utilizes multi-threaded native code which ensures the encryption is done before your victims can react to it.
Offline - as the payload works fully offline you can infect air-gapped hosts. This ensures that we are always able to recover the files even on network failure.
Two modes of operation for campaigns:
Regular mode - Mode where decryption keys for all computers must be purchased separately. Suitable for infecting individual systems.
Company mode - Special mode for encrypting whole businesses. One payment releases keys to all encrypted systems.
Fully automated - Everything is automated from registration, payload generation, and decryption to withdrawal. No more waiting.
Stealthy - Every payload is unique and contains minimal set of functionality to stay off the AV radar.
Multi platform - Do not leave money on the table. Infect all 64bit main OS platforms - Windows, MacOS, and Unix. Works on all 64bit OS versions.
Multi campaign - Create multiple ransomware campaigns and track the victim activity within each of them. Know when your victims have visited the service even if they have not paid.

# Additional perks #
Keep it simple - Just generate the payload using our simple web UI, download the payload, and run it on a victim system. No more obscure and broken builders with tons of dependencies you do not dare to run.
Show the victims you have the keys - Our service automatically decrypts one file per machine for free.
Take care of your victims - Our helpful support staff helps your victims through the decryption process.
Need something special? - Our dynamic development team takes general feature suggestions and incorporates them into the service for free. Features only you require can be discussed.

# Price #
Registration fee: 0.2 BTC
Service fee: 20%

Figure 1 *- Initial Smaug offering on Exploit.in*

Account History

Corinda's Exploit account was created on May 17, 2019 as a paid registration (paid registrations are completely anonymous and do not link to other accounts on 'friendly forums'), and remained dormant until Oct. 2019. On Oct. 6, 2019 **corinda** created their first post:

'I look for front end dev. Must be able to design professional websites
Must be able to show previous work or prove skills otherwise
Fulent english mandatory
Budget $2000 in BTC
PM for details'

The only reply to the topic was from **corinda** three days later on 10/9:

‘I am still looking. Do not be shy’

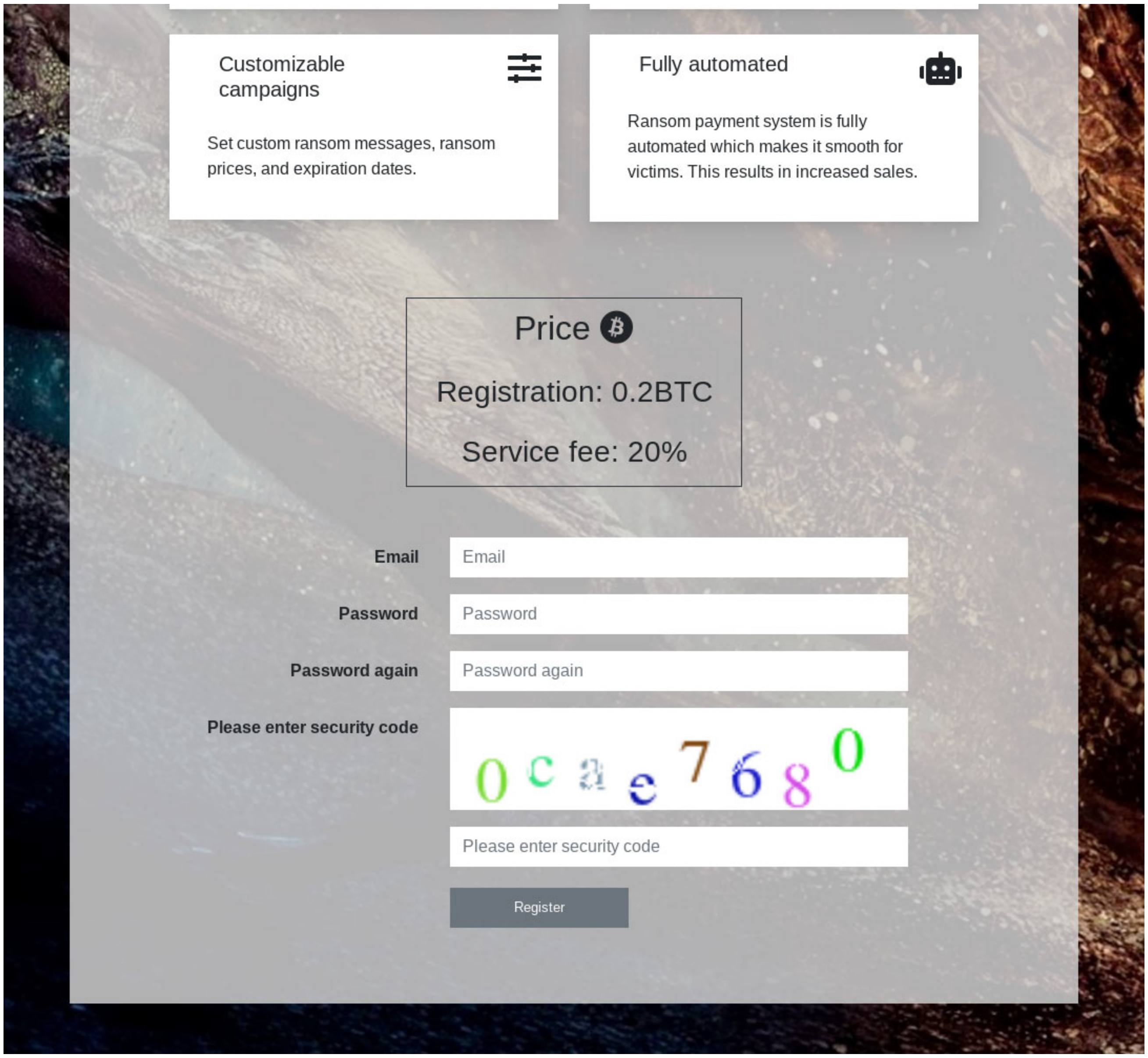
These posts were recovered through Sixgill as the original topic has since been deleted and is no longer available on the forum.

Smaug Panel

Smaug developers shared screenshots of the product on their Onion site, it offers some insight as to how the product is to be used.

Registration

The site offers a registration form replete with captcha. Upon completion, users are directed to pay 0.2 BTC to a specified Bitcoin wallet. The actor claims that accounts will be activated within 10 minutes of payment after the payment has received six confirmations.


Customizable campaigns

Set custom ransom messages, ransom prices, and expiration dates.

Fully automated

Ransom payment system is fully automated which makes it smooth for victims. This results in increased sales.

Price

Registration: 0.2BTC

Service fee: 20%

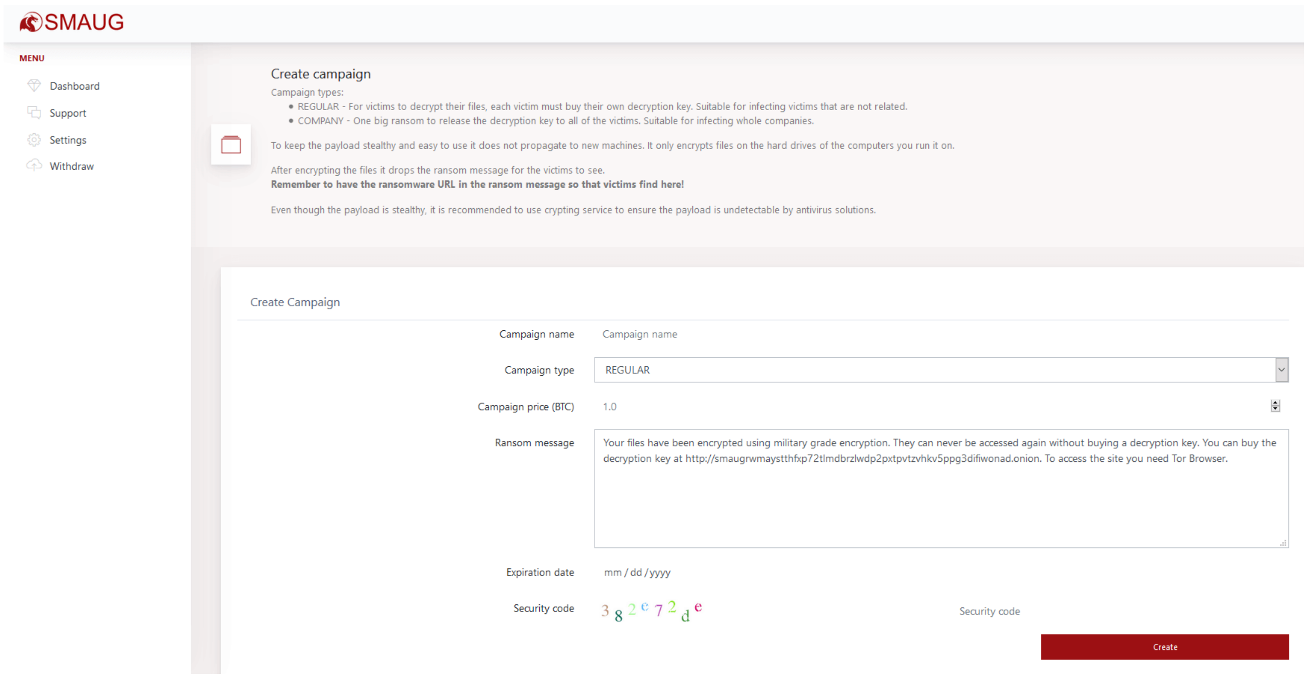
Email	Email
Password	Password
Password again	Password again
Please enter security code	0cae7680
	Please enter security code

Figure 2 *- Registration form*

Dashboard

Create Campaign

After successful registration, the user is taken to the Smaug dashboard, which appears as a clean, utilitarian UI with clear directions. From this page, a user can create campaigns with custom ransom messages and expiration dates; Smaug also allows users to distinguish between ‘Regular’ and ‘Company’ campaigns. A ‘Regular’ campaign will necessitate a decryption key for every victim infected, while a ‘Company’ campaign will allow a single key to decrypt all computers affected by the campaign.



Create campaign

Campaign types:

- REGULAR - For victims to decrypt their files, each victim must buy their own decryption key. Suitable for infecting victims that are not related.
- COMPANY - One big ransom to release the decryption key to all of the victims. Suitable for infecting whole companies.

To keep the payload stealthy and easy to use it does not propagate to new machines. It only encrypts files on the hard drives of the computers you run it on.

After encrypting the files it drops the ransom message for the victims to see.
Remember to have the ransomware URL in the ransom message so that victims find here!

Even though the payload is stealthy, it is recommended to use crypting service to ensure the payload is undetectable by antivirus solutions.

Create Campaign

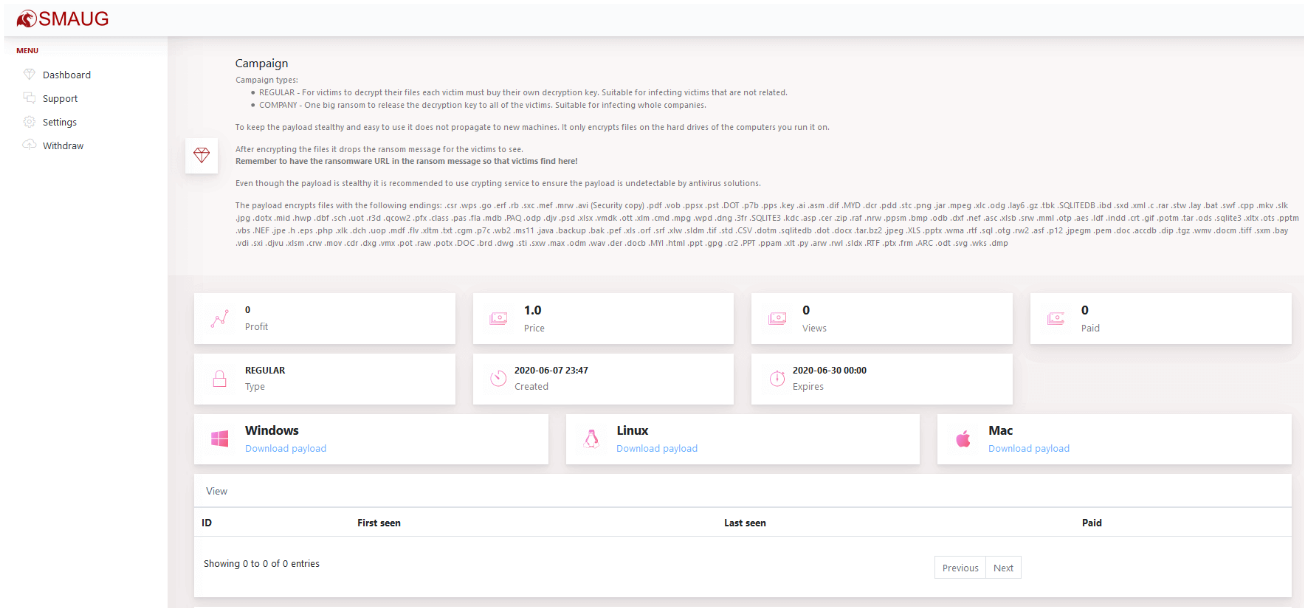
Campaign name	Campaign name
Campaign type	REGULAR
Campaign price (BTC)	1.0
Ransom message	Your files have been encrypted using military grade encryption. They can never be accessed again without buying a decryption key. You can buy the decryption key at http://smaugrwmaystthfxp72tlmdbrzlwdp2pxtpvtzvhkv5ppg3difiwonad.onion. To access the site you need Tor Browser.
Expiration date	mm / dd / yyyy
Security code	382e72de

Figure 3 *- Campaign creation*

After the expiration date set by the user has passed, the campaign will expire and the victims can no longer recover their files.

Manage Campaign

With a campaign created, a user can download the relevant payload for the system(s) they are attempting to infect; Mac, Linux and Windows are all available. The dashboard also will keep track of how many systems have been infected, this is done through allowing a victim one free decryption of a single file, this is then recorded and updates the dashboard. The interface allows the user to track the campaign dates and profits they have accumulated.



Campaign

Campaign types:

- REGULAR - For victims to decrypt their files each victim must buy their own decryption key. Suitable for infecting victims that are not related.
- COMPANY - One big ransom to release the decryption key to all of the victims. Suitable for infecting whole companies.

To keep the payload stealthy and easy to use it does not propagate to new machines. It only encrypts files on the hard drives of the computers you run it on.

After encrypting the files it drops the ransom message for the victims to see.
Remember to have the ransomware URL in the ransom message so that victims find here!

Even though the payload is stealthy it is recommended to use crypting service to ensure the payload is undetectable by antivirus solutions.

The payload encrypts files with the following endings: .csr .wps .go .erf .rb .sxc .mef .mrw .avi (Security copy) .pdf .vob .ppsx .pst .DOT .p7b .pps .key .ai .asm .dif .MYD .dcr .pdd .stc .png .jar .mpeg .xlc .odg .lay6 .gz .tbk .SQLITEDB .ibd .sxd .xml .c .rar .stw .lay .bat .swf .cpp .mkv .slk .jpg .dotx .mid .hwp .dbf .sch .uot .r3d .qcow2 .pfx .class .pas .fla .mdb .PAQ .odp .djv .psd .xlsx .vmdk .ott .xlm .cmd .mpg .wpd .dng .3fr .SQLITE3 .kdc .asp .cer .zip .raf .nrw .ppsm .bmp .odb .dxf .nef .asc .xlsb .srw .mml .otp .aes .ldf .indd .crt .gif .potm .tar .ods .sqlite3 .xltx .ots .pptm .vbs .NEF .jpe .h .eps .php .xlk .dch .uop .mdf .flv .xltm .txt .cgm .p7c .wb2 .ms11 .java .backup .bak .pef .xls .orf .srf .xlw .sldm .tif .std .CSV .dotm .sqlitedb .dot .docx .tar.bz2 .jpeg .XLS .pptx .wma .rtf .sql .otg .rw2 .asf .p12 .jpegm .pem .doc .accdb .dip .tgz .wmv .docm .tiff .sxm .bay .vdi .sxi .djvu .xlsm .crw .mov .cdr .dxg .vmx .pot .raw .potx .DOC .brd .dwg .sti .sxw .max .odm .wav .der .docb .MYI .html .ppt .gpg .cr2 .PPT .ppam .xlt .py .arw .rwl .sldx .RTF .ptx .frm .ARC .odt .svg .wks .dmp

0 Profit	1.0 Price	0 Views	0 Paid
REGULAR Type	2020-06-07 23:47 Created	2020-06-30 00:00 Expires	

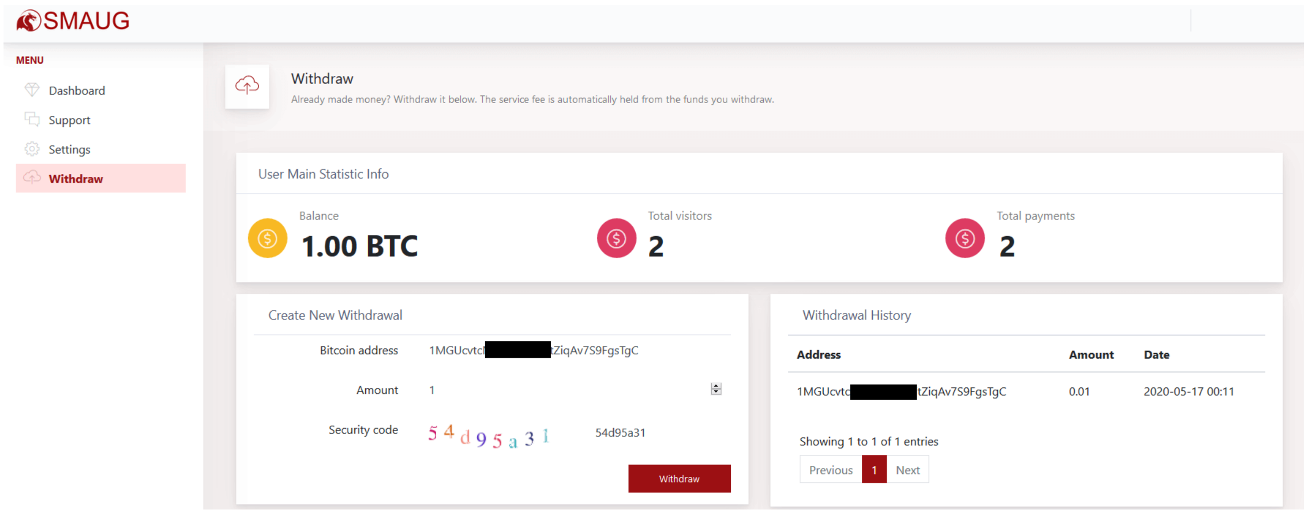
ID	First seen	Last seen	Paid

Showing 0 to 0 of 0 entries

Figure 4 *- Campaign management*

Fund Withdrawal

Once a victim has paid the ransom, the user is able to navigate to the 'Withdrawal' page. This dashboard will present the user with their current balance (the 20% fee has been automatically withheld) and number of payments made to them. From here, the user simply creates a withdrawal to a specified BTC wallet and removes the funds.



Withdraw
Already made money? Withdraw it below. The service fee is automatically held from the funds you withdraw.

User Main Statistic Info

Balance	Total visitors	Total payments
1.00 BTC	2	2

Create New Withdrawal

Bitcoin address	1MGUcvtc[redacted]ZiqAv7S9FgsTgC
Amount	1
Security code	5 4 d 9 5 a 3 1 54d95a31

Withdraw

Withdrawal History

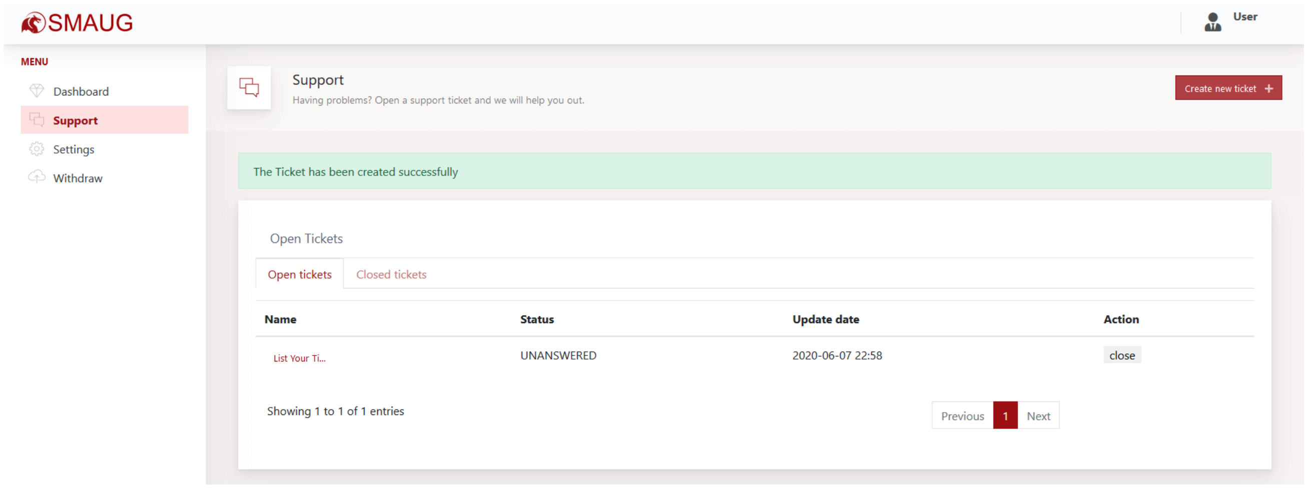
Address	Amount	Date
1MGUcvtc[redacted]tZiqAv7S9FgsTgC	0.01	2020-05-17 00:11

Showing 1 to 1 of 1 entries
Previous 1 Next

Figure 5 *- Fund overview*

Support

The Smaug developers claim to run a support service to assist with bugs and other issues. They also claim to take feature requests, they will create them for a fee or incorporate them for free if enough users request it.



Support
Having problems? Open a support ticket and we will help you out.

The Ticket has been created successfully

Open Tickets

Open tickets | Closed tickets

Name	Status	Update date	Action
List Your Ti...	UNANSWERED	2020-06-07 22:58	close

Showing 1 to 1 of 1 entries
Previous 1 Next

Figure 6 *- Support service*

Settings

The settings page is bare-boned and allows for users to change passwords.

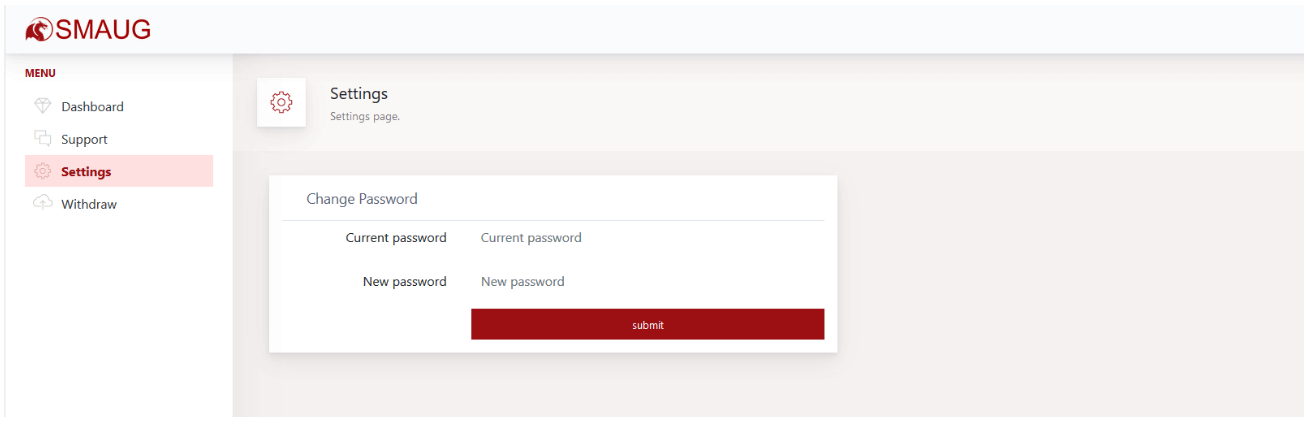


Figure 7 *- Settings page*

The Ransomware

The Smaug ransomware is written in Go (Golang). Anomali Threat Research has identified samples targeting both Windows and Linux machines. The malware has a relative simple design implemented in about 300 lines of code. Its source code structure is shown in Figure 8 below. The malware creates a unique key for each machine that is used to encrypt the files. The key is encrypted by the threat actor's RSA public key which means the machine key can only be encrypted by a threat actor's private key. The ransomware is designed to run completely offline without the requirement of a network connection.

```
Package main: src/Lock/cmd/payload
File: main.go
     main Lines: 17 to 54 (37)
     mainfunc1 Lines: 54 to 71 (17)
Package Lock/internal/payload/payloadutil:
src/Lock/internal/payload/payloadutil
File: payloadutil.go
     GenKey Lines: 14 to 29 (15)
     Drop Lines: 29 to 32 (3)
Package Lock/internal/pkg/util: src/Lock/internal/pkg/util
File: utils.go
     VisitFile Lines: 9 to 21 (12)
File: vars.go
     init Lines: 21 to 21 (0)
Package Lock/internal/pkg/encryption: src/Lock/internal/pkg/encryption
File: aes.go
     EncryptFile Lines: 16 to 178 (162)
     pad Lines: 178 to 184 (6)
File: rsa.go
     RsaEncrypt Lines: 15 to 79 (64)
```

Figure 8 *- Extracted source code structure from a malware sample.*

Go binaries do not include a compilation timestamp. Consequently, it is difficult to gauge the age of a sample. The analyzed sample was compiled by version 1.14.2 of the official Go compiler. This compiler was released on April 8, 2020 and was uploaded to a public malware repository on May 31, 2020. At the time of it being uploaded to the public repository, approximately 11% of the antivirus engines detected it as malicious. The earliest sample was uploaded on May 11, 2020 and was only detected by one antivirus engine, while the Linux sample was uploaded on May 19, 2020 and as of this writing date is not detected by any engines.

Key Generation

Smaug generates a unique encryption key for each infected machine. First, the malware gets a timestamp of the current time. The Unix time in nanoseconds of this timestamp is used to set the global seed for the "math/rand" package in the standard library for Go.[1] This action has no effect on the randomness as the malware. after this action, generates a 32 byte slice of randomness via the call to "crypt/rand.Read".[2] This function on Windows uses the "CryptGenRandom" API to generate the random data. The random data is encrypted with the threat actor's 2048 bit key using RSA Optimal Asymmetric Encryption Padding (OAEP).

File Encryption

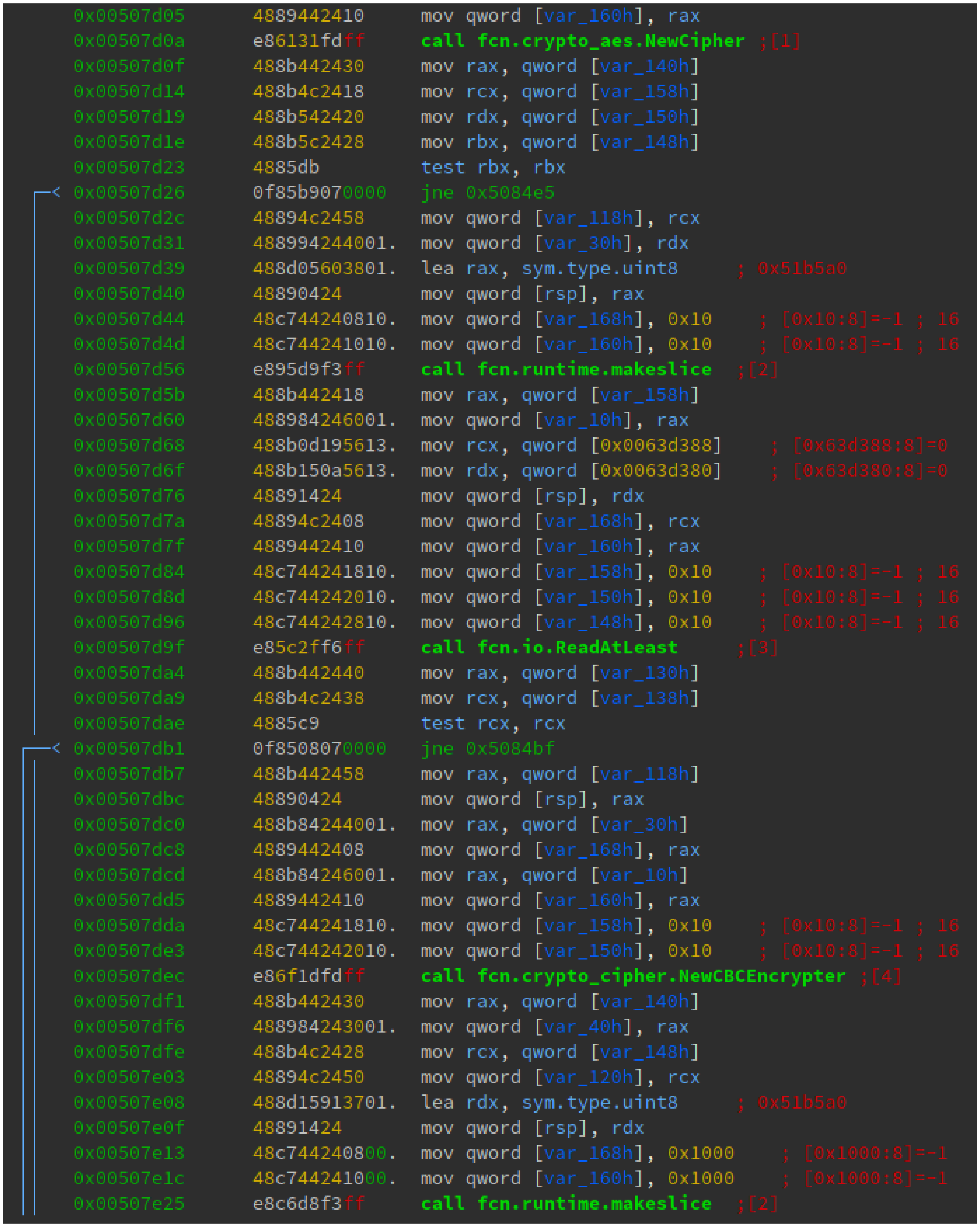
The malware uses a pair of "goroutines" for each disk partition for the encryption of files. A goroutine is Go's implementation of coroutines for concurrency. The first routine scans the partition for all files and compares the file extension against the list of extensions shown in Figure 9 below. If the file has an extension that is in the list, the file is added to a list of files to be encrypted.

```
 *.3fr *.accdb *.aes *.ai *.ARC *.arw *.asc *.asf *.asm *.asp *.avi
*.backup *.bak *.bat *.bay *.bmp *.brd *.c *.cdr *.cer *.cgm *.class *.cmd
*.cpp *.cr2 *.crt *.crw *.csr *.CSV *.dbf *.dch *.dcr *.der *.dif *.dip
*.djv *.djvu *.dng *.doc *.DOC *.docb *.docm *.docx *.dot *.DOT *.dotm
*.dotx *.dwg *.dxf *.dxg *.eps *.erf *.fla *.flv *.frm *.gif *.go *.gpg
*.gz *.h *.html *.hwp *.ibd *.indd *.jar *.java *.jpe *.jpeg *.jpegm *.jpg
*.kdc *.key *.lay *.lay6 *.ldf *.max *.mdb *.mdf *.mef *.mid *.mkv *.mml
*.mov *.mpeg *.mpg *.mrw *.ms11 *.MYD *.MYI *.nef *.NEF *.nrw *.odb *.odg
*.odm *.odp *.ods *.odt *.orf *.otg *.otp *.ots *.ott *.p12 *.p7b *.p7c
*.PAQ *.pas *.pdd *.pdf *.pef *.pem *.pfx *.php *.png *.pot *.potm *.potx
*.ppam *.pps *.ppsm *.ppsx *.ppt *.PPT *.pptm *.pptx *.psd *.pst *.ptx *.py
*.qcow2 *.r3d *.raf *.rar *.raw *.rb *.rtf *.RTF *.rw2 *.rwl *.sch *
(Security copy) *.sldm *.sldx *.slk *.sql *.sqlite3 *.SQLITE3 *.sqlitedb
*.SQLITEDB *.srf *.srw *.stc *.std *.sti *.stw *.svg *.swf *.sxc *.sxd
*.sxi *.sxm *.sxw *.tar *.tar.bz2 *.tbk *.tgz *.tif *.tiff *.txt *.uop
```

```
*.uot *.vbs *.vdi *.vmdk *.vmx *.vob *.wav *.wb2 *.wks *.wma *.wmv *.wpd
*.wps *.xlc *.xlk *.xlm *.xls *.XLS *.xlsb *.xlsm *.xlsx *.xlt *.xltm
*.xltx *.xlw *.xml *.zip
```

Figure 9 *- List of file extensions the malware uses to determine if the file should be encrypted or not.*

The second routine encrypts all the files in the list identified by the other routine. Each file is encrypted with AES in CBC mode. Figure 10 below shows the setup of the encryption handler for each file. A new AES cipher handler is constructed, an Initialization Vector (IV) is generated by reading 16 bytes from “crypt/rand”, before a CBC handler is created.

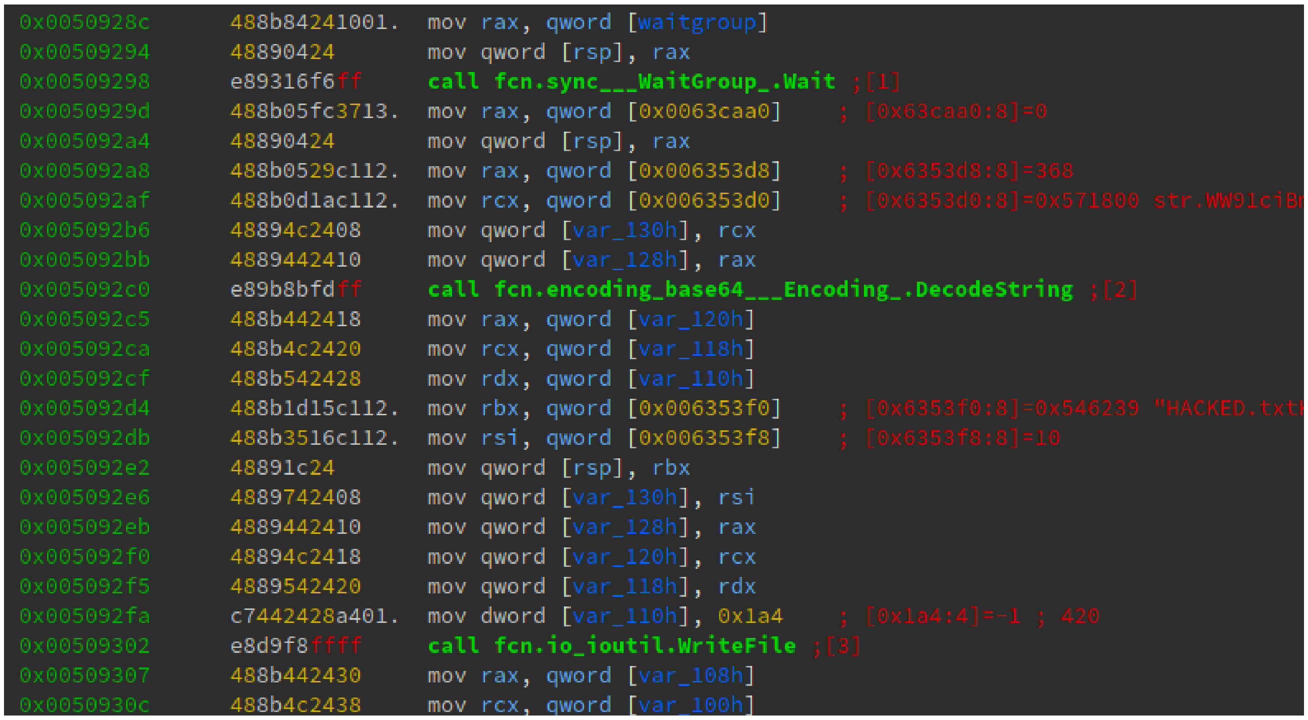


```
    0x00507d05      4889442410     mov qword [var_160h], rax
    0x00507d0a      e86131fdff     call fcn.crypto_aes.NewCipher ;[1]
    0x00507d0f      488b442430     mov rax, qword [var_140h]
    0x00507d14      488b4c2418     mov rcx, qword [var_158h]
    0x00507d19      488b542420     mov rdx, qword [var_150h]
    0x00507d1e      488b5c2428     mov rbx, qword [var_148h]
    0x00507d23      4885db         test rbx, rbx
 ┌─< 0x00507d26      0f85b9070000   jne 0x5084e5
 │   0x00507d2c      48894c2458     mov qword [var_118h], rcx
 │   0x00507d31      488994244001.  mov qword [var_30h], rdx
 │   0x00507d39      488d05603801.  lea rax, sym.type.uint8      ; 0x51b5a0
 │   0x00507d40      48890424       mov qword [rsp], rax
 │   0x00507d44      48c744240810.  mov qword [var_168h], 0x10     ; [0x10:8]=-1 ; 16
 │   0x00507d4d      48c744241010.  mov qword [var_160h], 0x10     ; [0x10:8]=-1 ; 16
 │   0x00507d56      e895d9f3ff     call fcn.runtime.makeslice   ;[2]
 │   0x00507d5b      488b442418     mov rax, qword [var_158h]
 │   0x00507d60      488984246001.  mov qword [var_10h], rax
 │   0x00507d68      488b0d195613.  mov rcx, qword [0x0063d388]     ; [0x63d388:8]=0
 │   0x00507d6f      488b150a5613.  mov rdx, qword [0x0063d380]     ; [0x63d380:8]=0
 │   0x00507d76      48891424       mov qword [rsp], rdx
 │   0x00507d7a      48894c2408     mov qword [var_168h], rcx
 │   0x00507d7f      4889442410     mov qword [var_160h], rax
 │   0x00507d84      48c744241810.  mov qword [var_158h], 0x10     ; [0x10:8]=-1 ; 16
 │   0x00507d8d      48c744242010.  mov qword [var_150h], 0x10     ; [0x10:8]=-1 ; 16
 │   0x00507d96      48c744242810.  mov qword [var_148h], 0x10     ; [0x10:8]=-1 ; 16
 │   0x00507d9f      e85c2ff6ff     call fcn.io.ReadAtLeast      ;[3]
 │   0x00507da4      488b442440     mov rax, qword [var_130h]
 │   0x00507da9      488b4c2438     mov rcx, qword [var_138h]
 │   0x00507dae      4885c9         test rcx, rcx
┌──< 0x00507db1      0f8508070000   jne 0x5084bf
││   0x00507db7      488b442458     mov rax, qword [var_118h]
││   0x00507dbc      48890424       mov qword [rsp], rax
││   0x00507dc0      488b84244001.  mov rax, qword [var_30h]
││   0x00507dc8      4889442408     mov qword [var_168h], rax
││   0x00507dcd      488b84246001.  mov rax, qword [var_10h]
││   0x00507dd5      4889442410     mov qword [var_160h], rax
││   0x00507dda      48c744241810.  mov qword [var_158h], 0x10     ; [0x10:8]=-1 ; 16
││   0x00507de3      48c744242010.  mov qword [var_150h], 0x10     ; [0x10:8]=-1 ; 16
││   0x00507dec      e86f1dfdff     call fcn.crypto_cipher.NewCBCEncrypter ;[4]
││   0x00507df1      488b442430     mov rax, qword [var_140h]
││   0x00507df6      488984243001.  mov qword [var_40h], rax
││   0x00507dfe      488b4c2428     mov rcx, qword [var_148h]
││   0x00507e03      48894c2450     mov qword [var_120h], rcx
││   0x00507e08      488d15913701.  lea rdx, sym.type.uint8      ; 0x51b5a0
││   0x00507e0f      48891424       mov qword [rsp], rdx
││   0x00507e13      48c744240800.  mov qword [var_168h], 0x1000    ; [0x1000:8]=-1
││   0x00507e1c      48c744241000.  mov qword [var_160h], 0x1000    ; [0x1000:8]=-1
││   0x00507e25      e8c6d8f3ff     call fcn.runtime.makeslice   ;[2]
```

Figure 10 *- Screenshot of the routine creating the encryption handler. The files are encrypted with AES in CBC mode. The last four instructions create a 4k buffer that is used to encrypt the file 4k bytes at the time.*

The files are encrypted 4k bytes at a time. If the file is smaller or the read returns less than 4k, the malware adds padding to the end of the file to make the total bytes a multiple of 16 bytes as required by AES. The padding format used is PKCS#7. The encrypted blocks are written to a new file that has the original filename with an unique identifier added to the end of the filename. The original file is removed after the file has been encrypted.

The ransomware writes the ransom note in all the folders with encrypted files. In Figure 11 below, the logic for writing the note is shown. In the analyzed sample, the ransom note's name is "HACKED.txt".



```
0x0050928c      488b84241001.  mov rax, qword [waitgroup]
0x00509294      48890424       mov qword [rsp], rax
0x00509298      e89316f6ff     call fcn.sync___WaitGroup_.Wait ;[1]
0x0050929d      488b05fc3713.  mov rax, qword [0x0063caa0]    ; [0x63caa0:8]=0
0x005092a4      48890424       mov qword [rsp], rax
0x005092a8      488b0529c112.  mov rax, qword [0x006353d8]    ; [0x6353d8:8]=368
0x005092af      488b0d1ac112.  mov rcx, qword [0x006353d0]    ; [0x6353d0:8]=0x571800 str.WW91ciBr
0x005092b6      48894c2408     mov qword [var_130h], rcx
0x005092bb      4889442410     mov qword [var_128h], rax
0x005092c0      e89b8bfdff     call fcn.encoding_base64___Encoding_.DecodeString ;[2]
0x005092c5      488b442418     mov rax, qword [var_120h]
0x005092ca      488b4c2420     mov rcx, qword [var_118h]
0x005092cf      488b542428     mov rdx, qword [var_110h]
0x005092d4      488b1d15c112.  mov rbx, qword [0x006353f0]    ; [0x6353f0:8]=0x546239 "HACKED.txtH
0x005092db      488b3516c112.  mov rsi, qword [0x006353f8]    ; [0x6353f8:8]=10
0x005092e2      48891c24       mov qword [rsp], rbx
0x005092e6      4889742408     mov qword [var_130h], rsi
0x005092eb      4889442410     mov qword [var_128h], rax
0x005092f0      48894c2418     mov qword [var_120h], rcx
0x005092f5      4889542420     mov qword [var_118h], rdx
0x005092fa      c7442428a401.  mov dword [var_110h], 0x1a4    ; [0x1a4:4]=-1 ; 420
0x00509302      e8d9f8ffff     call fcn.io_ioutil.WriteFile ;[3]
0x00509307      488b442430     mov rax, qword [var_108h]
0x0050930c      488b4c2438     mov rcx, qword [var_100h]
```

Figure 11 *- Logic for writing the ransom note to the disk.*

The ransom note is stored in the binary as a Base64 encoded string. Figure 12 shows the decoded ransom note.

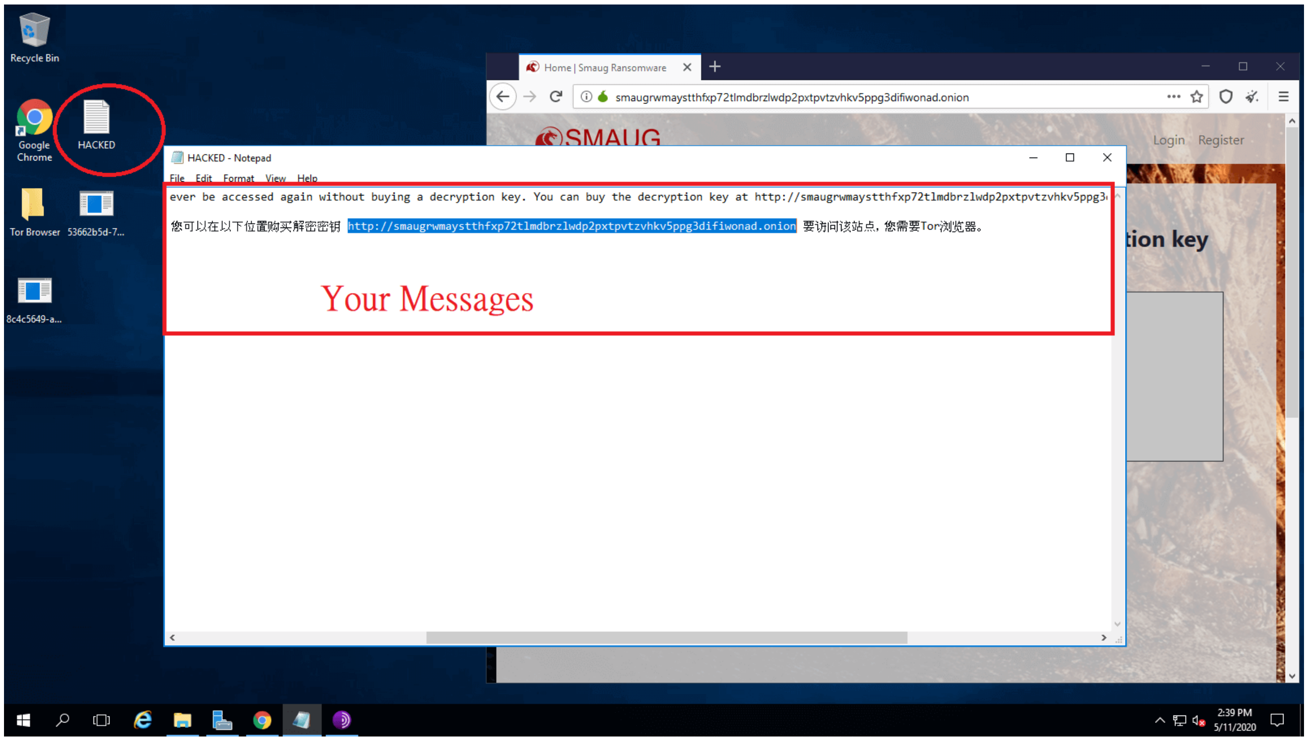
```
Your files have been encrypted using military grade encryption. They can never be accessed again without buying a decryption key. You can buy the decryption key at http://smaugrwmaystthfxp72tlmdbrzlwdp2pxtpvtzvhkv5ppg3difiwonad.onion. To access the site you need Tor Browser.
```

Figure 12 *- Base64 decoded ransom note stored in the analyzed sample.*

Analysis

The marked difference in quality of English between **corinda's** contracting post and the Smaug offering would indicate two different authors. The original post was likely created by the core individual(s) who were developing Smaug, whilst the Smaug offering post was likely created by a fluent English-speaking contractor.

The spelling and grammar mistakes present in the initial post and the need for a fluent English-speaking front end dev would indicate that this actor does not come from an English-speaking country. **Corinda** specified that targeting Commonwealth of Independent States (CIS) countires was forbidden and would result in a ban from the service. However, this does not necessarily indicate that the actor's location is in a CIS state as "Exploit.in" is a Russian language cybercrime site that bans the targeting of CIS countries. It is worth noting that on the Smaug Onion site a screenshot of a ransomware message is presented (Figure 13) with Mandarin characters predominantly displayed.


```
ever be accessed again without buying a decryption key. You can buy the decryption key at http://smaugrwmaystthfxp72tlmdbrzlwdp2pxtpvtzvhkv5ppg3

您可以在以下位置购买解密密钥 http://smaugrwmaystthfxp72tlmdbrzlwdp2pxtpvtzvhkv5ppg3difiwonad.onion 要访问该站点，您需要Tor浏览器。
```

Figure 13 *- Mandarin characters visible on ransom note*

Corinda's offer of Smaug did not appear to gain much traction on the forum leading to a follow up discount of a 15-day trial to reputable forum members, likely to boost reputation. The only subsequent activity on the thread was other users asking if anyone could vouch for the legitimacy of the product. After ten days (5/14/20), a forum admin locked the thread and requested that **corinda** deposit $8,000 to the forum's escrow account, this is a common practice to protect forum members from exit scams. **Corinda** did not comply and the thread has remained inactive since, however, the contact details are still available for users to purchase Smaug from **corinda**. Due to a lack of history, interactions and feedback, we cannot corroborate **corinda's** claim and rate it as *unreliable*.

The Smaug ransomware is relatively-simple compared to other active ransomware. For example, it lacks the functionality of stopping running processes which are used by ransomware to release file locks, allowing for encryption of locked files. It also does not delete backups and shadow copy files on Windows. This flaw will make it possible to recover the encrypted files from the shadow copy if the service is enabled. The simplicity of the malware allows the author to target multiple operating systems from the same code base. The Windows and Linux samples have been compiled from the same source code with no modifications, except the shim-layers provided by Go’s standard library.

Conclusion

Smaug is a RaaS that makes it easy for threat actors to use ransomware to achieve objectives. The ransomware can run on all the three major operating systems that opens up the potential for broader targeting. The actual ransomware is relatively simple compared to common ransomware currently being used by other threat actors. The malware’s only functionality is the encryption of files. Threatstream Enterprise users can find Indicators of Compromise (IOCs), signature, and more information about this threat here.

How Anomali Helps

Anomali Threat Research provides actionable threat intelligence that helps customers, partners, and the security community to detect and mitigate the most serious threats to their organizations. The team frequently publishes threat research in the form of white papers, blogs, and bulletins that are made available to the security community, general public, and news organizations. Intelligence and bulletins about threat actors and related Indicators of Compromise (IOCs) are integrated directly into Anomali customers’ security infrastructures to enable faster and more automated detection, blocking, and response. For more information on how Anomali customers gain integrated access to threat research, visit: https://www.anomali.com/products.

Mitre ATT&CK

T1486 - Data Encrypted for Impact

Endnotes

1. “rand package · pkg.go.dev”, accessed July 7, 2020, https://pkg.go.dev/math/rand?tab=doc#Seed
2. “rand package · pkg.go.dev”, accessed July 7, 2020, https://pkg.go.dev/crypto/rand?tab=doc#Read

Topics: Research