Qealler Infostealer static analysis – Part 0x1

securityinbits.com/malware-analysis/pyrogenic-infostealer-static-analysis-part-0x1/

January 6, 2020

Cow.tour.tooi	29	Belond hyperparasitism().
cow.tour.zone	20	Controlists provide the control of t
run.sake.boy	50	centralists.protradstrut = new class[]{outpayment.tueoptastics(); Superimposure.sorot(();;
Fun.sake.cup	31	<pre>Irines.desmopathy();</pre>
tun.sake.gang	32	}
> 🌐 run.sake.gay	33	
run.sake.kid	340	<pre>public void spl() throws Exception {</pre>
run.sake.off	35	<pre>Inconceivableness.exhortative();</pre>
run.sake.oil	36	Commenced.ankerhold();
run.sake.port	37	Sylvius.dubber();
run.sake.post	38	Chrysopoetic. <i>awan</i> ();
run.sake.wood	39	Allegretto.vitalised();
	40	Concephalus.hyponymic();
	41	Allegretto.diminished();
	42	Stereoptican.acraldehyde();
	43	Pleasantest.loveling = Unimperative.interbreeds();
	⇒44	Centralists.stoppled = Superimposure.donis.doFinal(Outpayment.minseito);
	45	(new Troglodytes()).unpromotable():
	46	1

Qealler is heavily obfuscated Java based Infostealer which is quite active based on ANY.RUN submission. This will be a three part blog series, this post will focus on Qealler/Pyrogenic static analysis, next <u>part 0x2</u> we learn unpacking using Java agent and in the last <u>part 0x3</u> we find similarity between Qealler/Pyrogenic variants based on static code analysis . You may download the BankPaymAdviceVend_LLCRep.jar from <u>ANY.RUN</u> (MD5: F0E21C7789CD57EEBF8ECDB9FADAB26B) and follow along or download the latest Qealler sample from ANY.RUN submission.

CONTENTS

Overview

It's currently targeting different regions e.g. Australian companies^[1], Africa and the Middle East^[2] based on the references. I will be using <u>Bytecode Viewer</u> to decompile Jar using FernFlower Java Decompiler. Let's start with quick dynamic analysis. Our main goal for the blog series is to unpack this jar so we can analyse the capability and compare it with Qealler.

Quick Dynamic Analysis

- Connect to CC 157.245.160[.]150 at port 80 and create the below process.
 cmd.exe /c chcp 1252 > NUL & powershell.exe -ExecutionPolicy Bypass NoExit -NoProfile -Command -
- Drop these two clean files sqlitejdbc.dll (MD5: a4e510d903f05892d77741c5f4d95b5d) and jnidispatch.dll (MD5: d2f0da769204b8c45c207d8f3d8fc37e) but it deletes these two file before exiting.
- Connect to bot.whatismyipaddress.com to get the public IP of the infected system.
- Steal credential from different applications

Packed Pyrogenic static analysis

1. Open the jar file in BCV (Bytecode Viewer), you will see multiple class files in different packages. Below pic shows the main entry point of the jar file.



2. For this sample, I found out that FernFlower decompiled the source code correctly. Select View -> Pane 1 -> FernFlower -> Java in BCV as shown below.



Bytecode Viewer FernFlower selection

3. If you browse the different class files in BCV, you will find many encrypted class files which don't translate to Java src code. e.g. one of them is shown below

BankPaymAdviceVend_LLCRep.jar cow/tour/away/Sobbing.class
• • • • cow • • • • • • • • • • • • • • • • • • •
i tour 00 01 02 03 04 05 05 07 05 09 08 06 04 06 01 02 03 04 05 05 05 05 06 07 05 09 08 06 <t< td=""></t<>
Belord.dass 00000000 al d3 37 a0 52 ad 68 11 3e b7 ca 81 4f 4e 10 26 Belord.dass Mopeders.dass Mopeders.dass 00000000 6a 98 d4 6e ff 5e 66 96 bc bc dd f4 1f 98 aa ba Sobbing.dass 00000000 5e e1 7b fd e8 ef 66 2f f8 3d e7 cb d5 9f 77 fd Sobbing.dass 00000000 5e e1 7b fd e8 ef 66 2f f8 3d e7 cb d5 df
below 00000001 6a 98 d4 6e ff 5e 86 96 bc bc dd ff 1f 98 aa ba Mopeders.dass Sobbing.dass 00000002 8c c7 cc fd 09 a9 a3 2c f5 87 f9 43 03 2c 5a 0f Sobbing.dass 00000000 5e e1 7b fd e8 ef 46 e2 f8 3d e7 cb d5 9f 77 fd Sobbing.dass 00000000 5e e1 7b fd e8 ef 46 e2 f8 3d e7 cb d5 d4 d6 d3 d1 aa 58 b4 ca d5 d5 d2 d2 f4 c6 d2 f8 d6 d1 d1 d0 d4
Modecters. class 00000002 8c c7 cc fd 09 a9 a3 2c f5 87 f9 43 03 2c 5a 0f Shwa.class Sobbing.class 00000003 5e e1 7b fd e8 ef 66 e2 f8 3d e7 cb d5 9f 77 fd Spira.class 00000004 77 83 12 6e b0 d3 01 aa 58 b4 8c 03 6e 1a de Spira.class 00000005 95 2d 4e c6 82 78 00 98 61 12 80 14 19 03 44 25 Thundersquall.class 00000007 cc 71 e8 52 d4 f5 b6 76 65 22 1d 98 b6 ba 92 33 82 b7 65 28 67 66 65 22 1d 98 b6 b4 90
Sniva.class 00000003 Se e1 7b fd e8 ef 46 e2 f8 3d e7 cb d5 9f 77 fd Sobbing.class Sobbing.class 00000004 77 83 12 6e b0 d3 01 aa 58 b4 8c 03 6e 1a de Sylvius.class 00000005 95 2d 4e c6 82 78 00 98 61 12 80 14 19 03 a4 25 Thundersqual.class 00000007 cc 71 e8 52 d4 f5 be 7c fd a4 00 Uncontrovertibly.class 00000007 cc 71 e8 52 d4 f5 bd fd bd bd bd bd a4 00 Wily.class 00000000 83 1a 54 8e 19 20 6b f3 bd a4 31 4b 07 22 65 <t< td=""></t<>
Sooding.class 00000004 77 83 12 6e b d3 01 aa 58 b4 8c 36 03 6e 1a de Spira.class Sylvius.class 00000005 95 2d 4e c6 82 78 00 98 61 12 80 14 19 03 a4 25 Thundersquall.class 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 ld 98 b1 b6 a4 00 Uncontrovertibly.class 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 ld b8 b6 a9 23 Wily.class 00000008 3b 1a 54 8e 19 20 6b f3 bd d4 31 4b b7 86 f5 f6 f5 s0 e6 f5 c5 s0 e6 f6 c5 s0 e6 f6 <t< td=""></t<>
Spiral-dass 00000005 95 2d 4e c6 82 78 00 98 61 12 80 14 19 03 a4 25 Sylvius.dass D0000006 89 1b dc c4 70 95 93 38 22 be 34 19 d6 a4 00 Duncontrovertibly.dass 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 1d 98 bb 6b a9 23 Wily.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd o4 05 a0 e0 94 2f 82 Wily.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd o4 05 a0 e0 94 2f 82 Wily.dass 00000000 b5 17 68 5e 52 8c c4 04 4d 31 4b
Syrvisuass 00000006 89 1b dc c4 70 95 93 38 22 be 34 13 19 d6 a4 00 Image: Syrvisuass 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 1d 98 bb 6b a9 23 Image: Wily.dass 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 1d 98 bb 6b a9 23 Image: Wily.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd 64 a4 00 Image: Wily.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd 64 a4 31 4b b0 72 Image: Wily.dass 00000000 b5 17 68 5e 28 cf cf 58 ae ee f8 3e 91 c2
Incontrovertibly.dass 00000007 cc 71 e8 52 d4 f5 be 7c e5 22 1d 98 bb 6b a9 23 Incontrovertibly.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd 04 05 a0 e0 94 2f 82 Incontrovertibly.dass 00000008 3b 1a 54 8e 19 20 6b f3 bd 04 05 a0 e0 94 2f 82 Incontrovertibly.dass 00000008 85 17 68 5e 52 8c c4 04 04 9b ba 4d 31 4b b0 72 Incontrovertibly.dass 00000000 be e9 45 2b f7 6c f5 58 ae ee f8 3e 91 c2 08 Incontrovertibly.dass 00000000 0e ab d2 de 63 66 ef 52
00000008 3b 1a 54 8e 19 20 6b 61 bd 04 05 a0 e0 94 2f 82 00000008 3b 1a 54 8e 19 20 6b f3 bd 04 05 a0 e0 94 2f 82 00000009 83 22 16 60 85 2d 26 6c 07 c4 c7 8d 67 8e 6f c5 00000000 b6 e9 45 2b f7 6c f5 58 ae ee f8 3e 91 c2 08 00000000 b6 e9 45 2b f7 6c f5 58 ae ee f8 3e 91 c2 08 00000000 0e ab d2 4c 63 66 ef 5 69 39 78 1c 34 f6 de 00000000 13 ad 0a fc
Image: Construction of the construc
Image: Indication of the second of the se
B
B
Image: Solution of the state of the sta
a - b pass b0000000 13 at 0 a 10 bit 0 10 c 35 72 ba bb 0a 47 6a as es es b - b sky b0000000 1 e 62 26 e4 as 51 c b 51 c 09 ce 7b 73 d7 b2 c - b sky 00000000 1 e 0c 2b c6 25 e4 d0 78 55 8e 6e 13 5b 04 69 f0 c - b son 00000000 1 e 7b 73 d7 b2
B B B B C D C D C D C D C D C D C D C D C D C D <thd< th=""> <thd< th=""> <thd< th=""> <thd< th=""></thd<></thd<></thd<></thd<>
UUUUUUII 6a II 7d d3 60 66 60 08 6d 62 6T 86 d0 e6 38 e6
00000012 8c 41 b9 4a fb e9 93 7e 5b dc e9 5e 6c 20 8d 15
00000013 9d 9c 62 74 fe 2f a7 cc 75 0d 41 af 28 0a a2 71
00000014 9c 5a 2d 3c cd f8 ee 49 e9 02 99 31 al 08 6a 50
00000015 6b 41 10 fb da bc 59 39 59 bc c6 bf c7 d3 59 2d
00000016 6a 4c 01 dc de 69 72 78 78 37 25 32 df f2 b9 b0
00000017 26 fl 94 b2 df 4e 6b ad 57 19 45 99 46 51 60 al
00000018 3c 9e 2c 39 c7 02 77 fe 49 3c e0 19 27 24 db 24
00000019 56 80 b9 69 05 7a 81 43 0d 04 13 bf 26 ec a2 be
0000001a 27 19 d8 82 9c c4 d8 08 b7 55 57 eb 5e 08 ab 8b
0000001b f7 3e b2 67 03 b4 eb c0 b2 45 f8 5f ac df e6 b8
0000001c 68 81 31 f6 dd 04 5a 21 12 41 ef fb 67 7b f1 16
0000001d e2 46 fl c3 9e b2 b1 42 de 2b 85 9d 4c be 4e ed
0000001e a8 5a 70 1a c9 5a 16 2e b1 87 4e fc e9 31 b0 6d
0000001f 84 88 92 64 a2 9d 38 65 22 44 1c 92 e1 aa 35 2a
Disk flagested (as flagested in a starting) 00000020 45 79 e3 fl e3 e1 90 fb b9 61 5e ba 22 ab cf 3a
00000021 81 9e 7b 46 1a 63 be 63 35 f5 56 36 76 8f 0f 9a
Exact + 00000022 af 81 82 f0 03 9d bd d2 56 e4 ff de 58 59 43 97
00000023 7a 89 7e 30 a3 f5 ac b1 0e 69 af 43 bf fd c9 d0
Search 00000024 42 f0 22 ab 51 13 fc 48 f3 a5 69 91 24 3c de 06
Search from All Classes

Pyrogenic encrypted class

- 4. Based on the above encrypted class file, you can guess that there should be some decryption algorithm used to decrypt those files.
- 5. Decryption algorithms can be custom or well known e.g. AES. Study this example java code ^[3] which encrypt/decrypt using AES.

```
public class AES {
    private static SecretKeySpec secretKey;
   private static byte[] key;
   public static void setKey(String myKey)
    {
        MessageDigest sha = null;
        trv {
           key = myKey.getBytes("UTF-8");
           sha = MessageDigest.getInstance("SHA-1");
           key = sha.digest(key);
           key = Arrays.copyOf(key, 16);
           secretKey = new SecretKeySpec(key, "AES");
        }
        catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
        catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
    }
    public static String encrypt(String strToEncrypt, String secret)
    {
        try
        {
            setKey(secret);
           Cipher cipher = Cipher.getInstance(<u>"AES/ECB/PKCS5Padding</u>");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey);
            return Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes("UTF-8")));
        }
        catch (Exception e)
        {
            System.out.println("Error while encrypting: " + e.toString());
        3
        return null;
    }
   public static String decrypt(String strToDecrypt, String secret)
    {
        try
        {
            setKey(secret);
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
           cipher.init(Cipher.DECRYPT_MODE, secretKey);
           return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
        }
        catch (Exception e)
        {
            System.out.println("Error while decrypting: " + e.toString());
        }
        return null:
    }
}
```

AES Java Encrypt Decrypt Example

Some of the keyword mentioned in the above java code e.g. **getInstance** can help us to find the encryption algorithm and **doFinal** can point to final decryption result.

6. Let's search for AES references after importing the decompiled src code to Eclipse IDE.

\triangleright	cow.tour.kill
\triangleright	cow.tour.pass
\triangleright	cow.tour.sky
\triangleright	cow.tour.son
\triangleright	cow.tour.tool
\triangleright	cow.tour.zone
\triangleright	run.sake.boy
\triangleright	run.sake.cup
\triangleright	run.sake.gang
\triangleright	run.sake.gay
\triangleright	run.sake.kid
\triangleright	run.sake.off
\triangleright	run.sake.oil
\triangleright	run.sake.port
\triangleright	run.sake.post
\triangleright	run.sake.wood

```
55
            Prechordal.unnourishable();
 56
        }
 57
 58⊝
        public static void furtherer() {
 59
           Belord.vesuvianite();
 60
           aciculated();
            Existlessness.cockpit = Psychoacoustic.floorwalkers.getEncoded();
 61
$62
         Carole.overprolifically = new SecretKeySpec(Existlessness.cockpit, "AES");
 63
            Romaunts.overwave();
 64
            Immi.interterritorial();
 65
            (new Unexceptable()).grossulariaceous();
 66
        }
 67
  <mark>68</mark>⊖
        public static boolean aciculated() {
 69
           return Overclose.shabbiness;
 70
        }
 71
        public static boolean chumawi() {
 72<del>0</del>
           return Overclose.shabbiness;
 73
        }
 74
     1
Commenced.java
                     🚺 Unexceptable.java 🔀
  1 package run.sake.off;
  3⊕ import javax.crypto.Cipher;
  8 public class Unexceptable {
  9
        public static int ominous = 177682096;
 10
        public static int outremer() {
 11⊝
 12
           return ominous;
 13
        }
 14
 15⊝
        public static int outsoars() {
           return Prechordal.progravid;
 16
        }
 17
 18
        public void grossulariaceous() {
 190
 20
 21
22⊖
⇒23
        public static void myonosus() throws Exception {
           Superimposure.donis = Cipher.getInstance("AES/ECB/PKCS5Padding");
        N
24
 25
        public static boolean thrill() {
 26⊖
           return Overclose.shabbiness;
 27
 28
        }
 29 }
 30
     .
🖹 Problems 🛛 🖉 Javadoc 🖳 Declaration 🔗 Search 🔀
'AES' - 5 matches in workspace
⊿ 📂 Packed
  🔺 🧁 src
     🗅 🗁 cow
     🔺 🗁 run
       🔺 🗁 sake
          🔺 🗁 kid
            Stereoptican.java
          a 🗁 off
            D Carole.java
            🔺 🚺 Unexceptable.java
                 23: Superimposure.donis = Cipher.getInstance("AES/ECB/PKCS5Padding");
            Vivified.java
                 62: Carole.overprolifically = new SecretKeySpec(Existlessness.cockpit, "AES");
```

```
Pyrogenic AES references
```

😫 Package Explorer 💥 📃 🗖	🕐 Thuribles.java 🙁 🖉 Superimposure.java 🌓 Centralists.java	
F 😓 🝃 🔻	1 package cow.tour.how;	
A 🛱 Packed	2	
▶ ■ JRE System Library [JavaSE-10]	3 [®] import cow.tour.away.Belord:□	
4 (¹⁰ , src	20	
b # cow tour away	21 public class Thuribles (
b = contour belt	2) public class find fores (
E contourbit	22 public static int regrant = -1502955970;	
E cow tour how		
b device and the second sec	public static void atdecno() {	
Cow tour pass	<pre>25 Uncontrovertibly.dermopathic();</pre>	
b de contoursky	26 Intelligencer.ravendom();	
b = contoursep	27 Melodramatic.overexuberant();	
b = contoursel	28 Superimposure.omasum = Key.class;	
	<pre>29 Belord.hyperparasitism();</pre>	
E run sake boy	<pre>30 Centralists.proindustrial = new Class[]{Outpayment.ideoplastics(), Superimposure.sorbic()};</pre>	
Full sake cun	31 Trines.desmopathy();	
Full sake gang	32 }	
Funisakeigang	33	
Funisakeigay	34 ^e public void spl() throws Exception {	
Funisake.off	35 Inconceivableness.exhortative():	
run.sake.oil	36 Commenced.ankerhold():	
tun.sake.port	37 Svlvius.dubber():	
tun.sake.post	38 Chrysphoetic.awa():	
trun.sake.wood	39 Allegretto vitalized():	
· •	A Concentalus hypopulis():	
	11 Allogatta diministrat();	
	A Aregretto. diministrat();	
	42 Stel copical devices - Harmonstive interpresed();	
	4.3 Freadances: tove ting = Onimperative: interpreeds();	
	(art Treat the date of the super imposure above and suber in a contract (outpayment in issue);	
	45 (new Proglodytes()).unpromotable();	
	48 ^e public static int deordination() {	
	49 return Phlebotomies.fyke;	
	50 }	
	5 1 }	
	52	
	🖹 Problems @ Javadoc 😡 Declaration 🔗 Search 🕄	
	'doFinal' - 1 match in workspace	
	a 👺 Packed	
	a 🗁 src	
	a 🗁 cow	
	a 😂 tour	
	⊿ ≥ how	
	A 🚺 Thurblesjava	
	44: Centralists.stoppled = Superimposure.donis. <u>doFinal</u> (Outpayment.minseito);	

Pyrogenic doFinal reference

```
) Commenced.java 🛛 Superimposure.java 🗍 Centralists.java 🖉 Unexceptable.java 🖉 Carole.java
                                                                             🕽 Lamellaria.java
                                                                                           Thuribles.java
20
 21 public class Commenced {
       public static int antishipping = -1420882016;
 22
 23
24⊖
       public static int ahamkara() {
 25
          return Echinospermum.encyclicals;
26
       }
 27
 28⊖
       public static boolean predated() {
 29
          return Overclose.shabbiness;
       }
 30
 31
 32⊖
       public static void unjolly() throws Exception {
         Existlessness.tranchant = new PBEKeySpec(Carole.hamburg, Chyack.ruffianism, 10000, 128);
 33
 34
          Zoothecia.interrena();
          Immi.cinchonidine();
 35
⇒36
          Outpayment.uncome = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1");
 37
          Spira.pinwheels();
 38
          Metalworker.democratization();
 39
          Trines.penecontemporaneous();
 40
          Melodramatic.starstone();
 41
          Assize.uncookable();
 42
          Psychoacoustic.floorwalkers = Outpayment.uncome.generateSecret(Existlessness.tranchant);
 43
          Melodramatic.lymphedema();
 44
          Metalworker.accountantship();
          Zeuzeridae.unshewed();
 45
 46
       }
```

Pyrogenic PBKDF2WithHmacSHA1 reference

- Based on the above images which shows multiple references, we can confirm that this sample uses the algo "AES/ECB/PKCS5Padding" and key may be generated using "PBKDF2WithHmacSHA1". So it confirmed that it doesn't use any custom decryption algorithm.
- 8. We can add our code to write the data to file after **doFinal** call and execute the sample in IDE to get the dumped class file Then we can decompile the class file using BCV and continue analysis. But it can be multiple layer obfuscation which can make our analysis harder and slower.

Conclusion

This above static analysis method to find the encryption routine and interesting breakpoint (doFinal) while debugging is very useful in Java Malware analysis. Using this approach you will not miss any code path but this requires more time and effort. So in the upcoming <u>part</u> 0x2, we will unpack this malware using **Java agent** which will speed up our analysis.

Hope you enjoyed this post, please <u>Follow @Securityinbits</u> **me** on Twitter to get the latest update about my malware analysis & DFIR journey. Happy Reversing \cong