


RansomEXX Trojan attacks Linux systems

SL securelist.com/ransomexx-trojan-attacks-linux-systems/99279/



Authors

- **Expert** [Fedor Sinitsyn](#)

-  [Vladimir Kuskov](#)

We recently discovered a new file-encrypting Trojan built as an ELF executable and intended to encrypt data on machines controlled by Linux-based operating systems.

After the initial analysis we noticed similarities in the code of the Trojan, the text of the ransom notes and the general approach to extortion, which suggested that we had in fact encountered a Linux build of the previously known ransomware family RansomEXX. This malware is notorious for attacking large organizations and was most active earlier this year.

RansomEXX is a highly targeted Trojan. Each sample of the malware contains a hardcoded name of the victim organization. Moreover, both the encrypted file extension and the email address for contacting the extortionists make use of the victim's name.

Several companies have fallen victim to this malware in recent months, including the [Texas Department of Transportation](#) (TxDOT) and [Konica Minolta](#).

Technical description

The sample we came across – [aa1ddf0c8312349be614ff43e80a262f](#) – is a 64-bit ELF executable. The Trojan implements its cryptographic scheme using functions from the open-source library mbedtls.

When launched, the Trojan generates a 256-bit key and uses it to encrypt all the files belonging to the victim that it can reach using the AES block cipher in ECB mode. The AES key is encrypted by a public RSA-4096 key embedded in the Trojan's body and appended to each encrypted file.

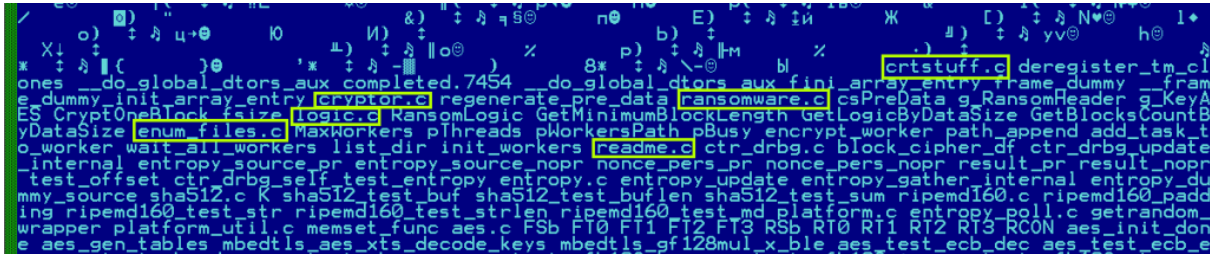
Additionally, the malware launches a thread that regenerates and re-encrypts the AES key every 0.18 seconds. However, based on an analysis of the implementation, the keys actually only differ every second.

Apart from encrypting the files and leaving ransom notes, the sample has none of the additional functionality that other threat actors tend to use in their Trojans: no C&C communication, no termination of running processes, no anti-analysis tricks, etc.

```
if ( a1 )
{
    v10 = strlen(s) + 277;
    v2 = alloca(16 * ((v10 + 23LL) / 0x10uLL));
    dest = (16 * ((&s + 7) >> 4));
    if ( dest )
    {
        strcpy(dest, s);
        strcat(dest, ".");
        v3 = rand();
        sprintf(src, "%08x", v3);
        strcat(dest, src);
        if ( fsize(dest) == -1 )
        {
            stream = fopen64(s, "r+");
            if ( stream )
            {
                v9 = fsize(s);
                if ( v9 )
                {
                    if ( v9 > 15 )
                    {
                        mbedtls_aes_init(aes_ctx);
                        pthread_mutex_lock(&csPreData);
                        memcpy(ptr, &g_RansomHeader, 0x200uLL);
                        mbedtls_aes_setkey_enc(aes_ctx, &g_KeyAES, 256LL);
                        pthread_mutex_unlock(&csPreData);
                        if ( !fseek(stream, 0LL, SEEK_END)
                            && fwrite(ptr, 1uLL, 0x200uLL, stream)
                            && !fseek(stream, -512 - v9, 1)
                            && ProcessFileHandleWithLogic(stream, aes_ctx, a2, v9, CryptOneBlock) )
                        {
                            v13 = 1;
                        }
                    }
                }
            }
        }
    }
}
```

Fragment of the file encryption procedure pseudocode; variable and function names are saved in the debug information and must match the original source code

Curiously, the ELF binary contains some debug information, including names of functions, global variables and source code files used by the malware developers.



Original names of source files embedded in the trojan's body



Execution log of the trojan in Kaspersky Linux Sandbox

Similarities with Windows builds of RansomEXX

Despite the fact that previously discovered PE builds of RansomEXX use WinAPI (functions specific to Windows OS), the organization of the Trojan's code and the method of using specific functions from the mbedtls library hint that both ELF and PE may be derived from the same source code.

In the screenshot below, we see a comparison of the procedures that encrypt the AES key. On the left is the ELF sample aa1ddf0c8312349be614ff43e80a262f; on the right is the PE sample fcd21c6fca3b9378961aa1865bee7ecb used in the TxDOT attack.

Despite being built by different compilers with different optimization options and for different platforms, the similarity is quite obvious.

```
14 v19 = mbedtis_mpi_read_string(
15     v16,
16     16LL,
17     "BF02A208837E9B96A9ABFFCCED10868865672540E5408BED9811F87CAEE140999EAD9889D9066F9886
18     *FB82AF0374886F68C38C59E2AD48F3707CF3A5130CB40D9FACDD15F388A166001DA092244AF7A74B7F
19     *6F5E3825A5E2032C60178C671E45F4408C5FC034AF18F985E473EF20E09E73C288562450F0CB050346
20     *1F9FC3553E1E9924E4111C1E779F48E7649850E3E0F9F7298E2D061D828F0943665E293821A248276
21     *81B988C5CAA331FAE039F47D7D501FE1D518E49684818E8AC679ECB8600405283586EAC1A8D003F161
22     *D0AACD40C7CF0899F06921A8ECC4CF89E529A5A706738497C7150314E63C2BCD7085C21F20548436905
23     *6DAA8C69C42E221E0761870E4E05E0998B9622885DE004F63258A09929024D478769AC7FE383A44CFD32
24     *80BF2228E142400399F9419F84C1E50673708728C82889D09878C49395FE4D18C8D45CE93D445098
25     *1857A4A8FE383104216D45AB71D8F9862600D67861C9617D73C9790BA07244F0759E88FF3AE00F8EE39
26     *090B40FA8F3508507CE6E9544AA237");
27
28 if ( v19 )
29     return v21;
30 v19 = mbedtis_mpi_read_string(&v17, 16LL, "010001");
31 if ( v19 )
32     return v21;
33 v15 = (mbedtis_mpi_bitlen(v16) + 7) >> 3;
34 v19 = mbedtis_rsa_pkcs1_encrypt(&v16, mbedtis_ctr_drbg_random, v12, 0, 32LL, v8, v18);
35 if ( v19 )
36     return v21;
37 pthread_mutex_lock(&csPRefData);
38 *g_KeyAES = *v8;
39 *g_KeyAES[8] = v9;
40 *g_KeyAES[16] = v10;
41 *g_KeyAES[24] = v11;
42 _memp_memp(g_RansomwareHeader, v18, v15);
43 pthread_mutex_unlock(&csPRefData);
44 v21 = 1;
45 mbedtis_rsa_free(&v16);
46 mbedtis_ctr_drbg_free(v12);
47 mbedtis_entropy_free(v13);
48 return v21;
49
50 14 sub_401000
51 15 sub_401000
52 16 sub_401000
53 17 sub_401000
54 18 sub_401000
55 19 sub_401000
56 20 sub_401000
57 21 sub_401000
58 22 sub_401000
59 23 sub_401000
60 24 sub_401000
61 25 sub_401000
62 26 sub_401000
63 27 sub_401000
64 28 sub_401000
65 29 sub_401000
66 30 sub_401000
67 31 sub_401000
68 32 sub_401000
69 33 sub_401000
70 34 sub_401000
71 35 sub_401000
72 36 sub_401000
73 37 sub_401000
74 38 sub_401000
75 39 sub_401000
76 40 sub_401000
77 41 sub_401000
78 42 sub_401000
79 43 sub_401000
80 44 sub_401000
81 45 sub_401000
82 46 sub_401000
83 47 sub_401000
84 48 sub_401000
85 49 sub_401000
86 50 sub_401000
87 51 sub_401000
88 52 sub_401000
89 53 sub_401000
90 54 sub_401000
91 55 sub_401000
92 56 sub_401000
93 57 sub_401000
94 58 sub_401000
95 59 sub_401000
96 60 sub_401000
97 61 sub_401000
98 62 sub_401000
99 63 sub_401000
100 64 sub_401000
101 65 sub_401000
102 66 sub_401000
103 67 sub_401000
104 68 sub_401000
105 69 sub_401000
106 70 sub_401000
107 71 sub_401000
108 72 sub_401000
109 73 sub_401000
110 74 sub_401000
111 LABEL_27:
112     if ( !v8 )
113     {
114         EnterCriticalSection(&crit_section);
115         memcpy(&aes_key, v16, 0x20u);
116         memcpy(&aes_key_encrypted, v16, v20[1]);
117         LeaveCriticalSection(&crit_section);
118         v12 = 1;
119         mbedtis_rsa_free(v20);
120         memset(v19, 0, 0x100u);
121         memset(v23, 0, sizeof(v23));
122         v24 = 0;
123         memset(v25, 0, 0x190u);
124         v22 = 0;
125     }
126     goto LABEL_29;
127 }
```

We also observe resemblances in the procedure that encrypts the file content, and in the overall layout of the code.

What's more, the text of the ransom note is also practically the same, with the name of the victim in the title and equivalent phrasing.

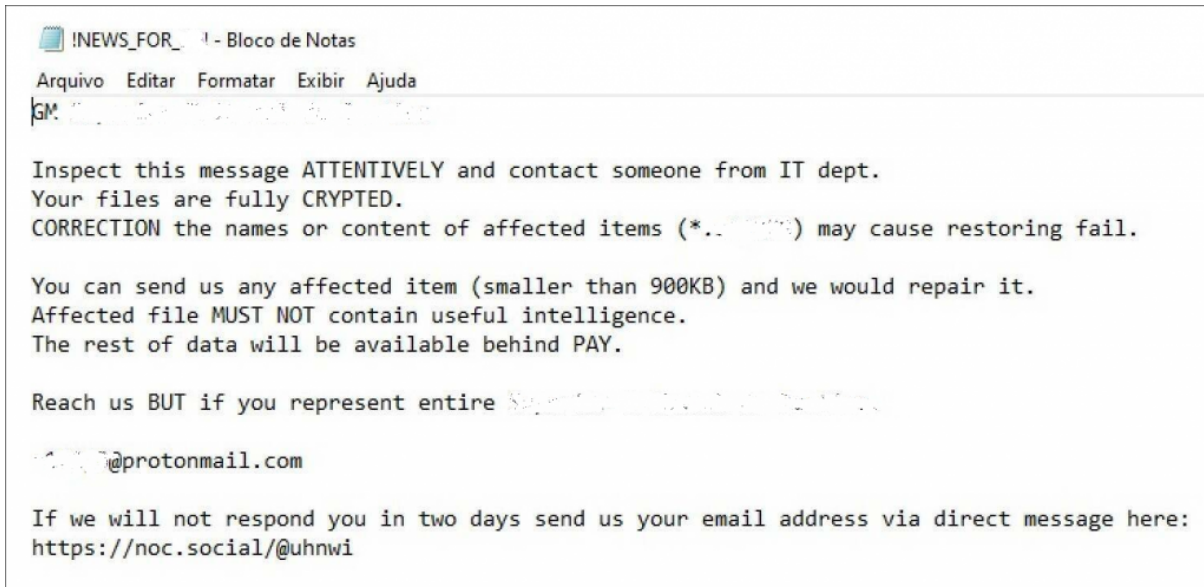
Parallels with a recent attack in Brazil

As reported by the media, one of the country's government institutions has just been attacked by a targeted ransomware Trojan.

Based on the ransom note, which is almost identical to the one in the sample we described, and the news article mentioned above, there is a high probability that the target is the victim of another variant of RansomEXX.

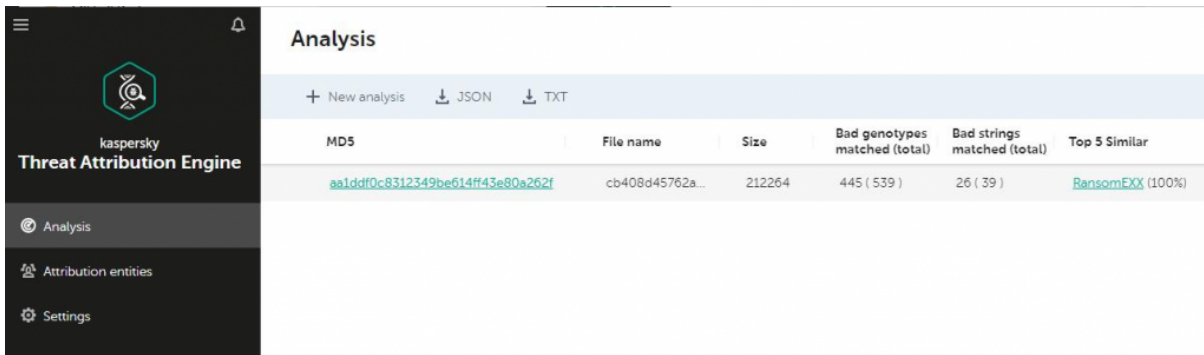
```
db 'Greetings ██████ !!!',0Dh,0Ah
; DATA XREF: ReadMeStoreForDir+119↑o
db 0Dh,0Ah
db 'Study this message REGARDFULLY and call administrator from techni'
db 'cal division.',0Dh,0Ah
db 'Yours information is securely ENCRYPTED.',0Dh,0Ah
db 'CHANGING content or names of crypded files (*.█████) can make rec'
db 'overing failure.',0Dh,0Ah
db 0Dh,0Ah
db 'You can mail us one crypded document (not bigger than 700KB) and '
db 'we would restore it.',0Dh,0Ah
db 'Encrypted file MUST NOT have rich data.',0Dh,0Ah
db 'All other data will be your behind the PAYMENT.',0Dh,0Ah
db 0Dh,0Ah
db 'Reach us SOLELY if you represent all affected network.',0Dh,0Ah
db 0Dh,0Ah
db '████████@protonmail.com',0
align 20h
```

Ransom note from the sample aa1ddf0c8312349be614ff43e80a262f



Ransom note from the Bleeping Computer post about the most recent attack in Brazil

Our products protect against this threat and detect it as Trojan-Ransom.Linux.Ransomexx



Kaspersky Threat Attribution Engine identifies Ransomexx malware family

Indicators of compromise

Recent Linux version: [aa1ddf0c8312349be614ff43e80a262f](#)

Earlier Windows version: [fcd21c6fca3b9378961aa1865bee7ecb](#)

- [Encryption](#)
- [Linux](#)
- [Malware Descriptions](#)
- [Ransomware](#)
- [Targeted attacks](#)
- [Trojan](#)

Authors

-  **Expert** Fedor Sinitsyn
-  Vladimir Kuskov

RansomEXX Trojan attacks Linux systems

Your email address will not be published. Required fields are marked *