Demystifying the Playboy RaaS

Gijs Rijnders
24 september 2025

POLITIE

whoami

- Malware reverse engineer
- CTI analyst

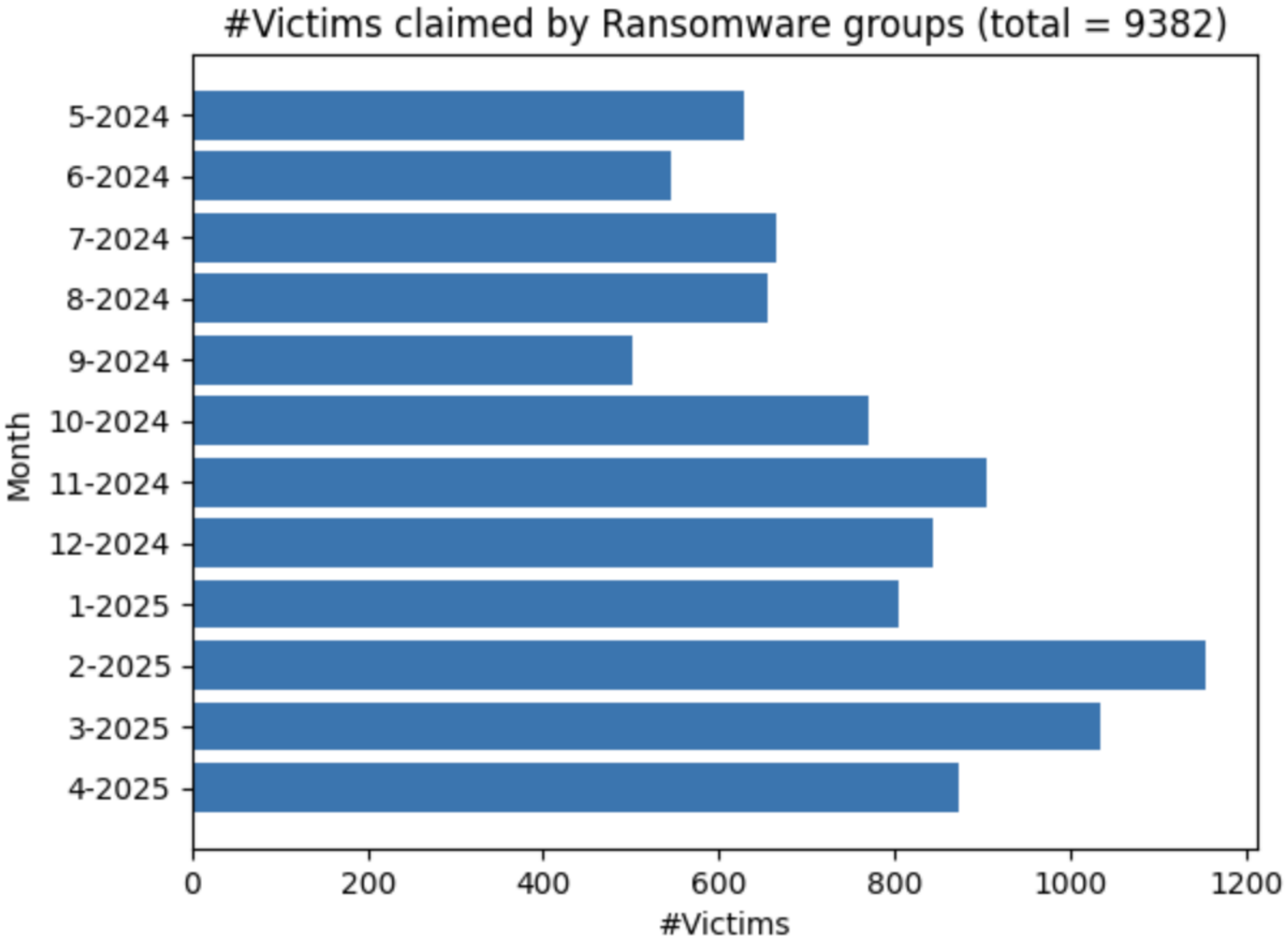
- Specializes in ransomware
- Finding & exploiting weaknesses to build decryptors & disrupt botnets

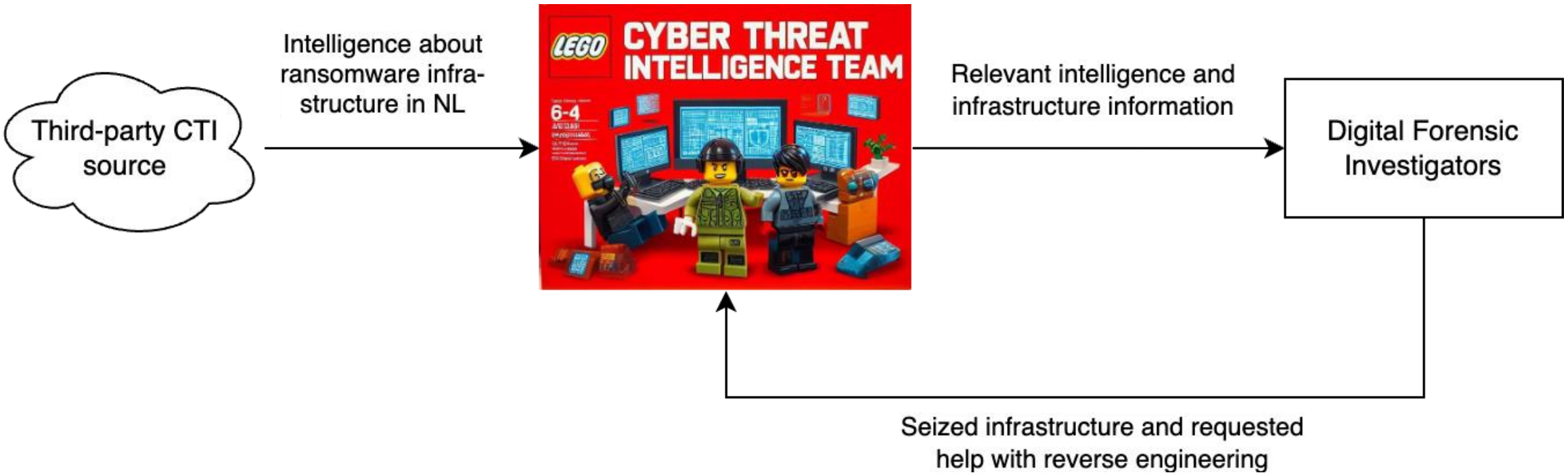
in/gijs-rijnders/

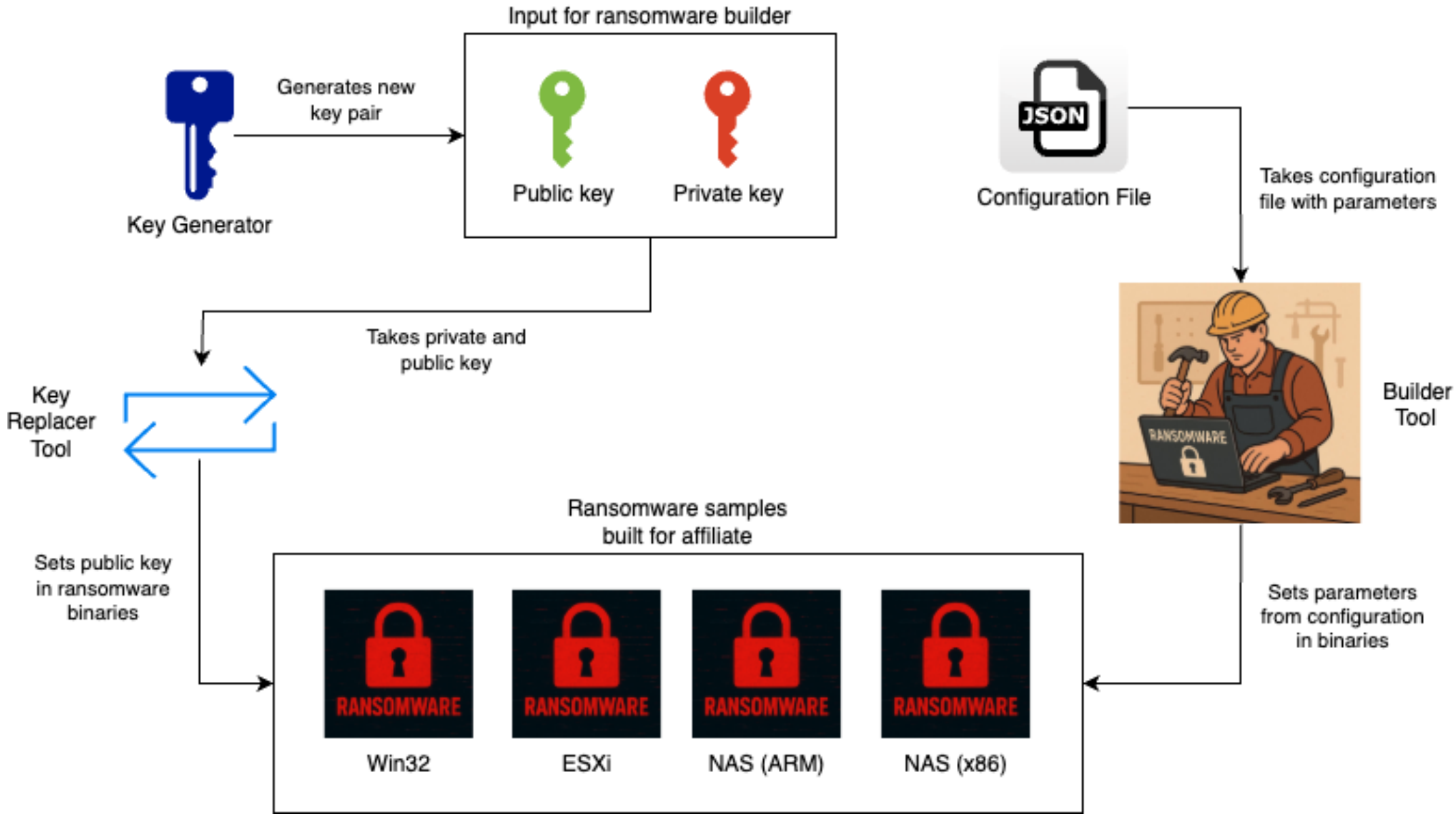
crysearch.nl


LEGO
CYBER THREAT
INTELLIGENCE TEAM
6-4

NO MORE RANSOM


#Victims claimed by Ransomware groups (total = 9382)
Month
5-2024
6-2024
7-2024
8-2024
9-2024
10-2024
11-2024
12-2024
1-2025
2-2025
3-2025
4-2025
0
200
400
600
800
1000
1200
#Victims


Third-party CTI source
Intelligence about ransomware infra-structure in NL
LEGO
CYBER THREAT INTELLIGENCE TEAM
6-4
Relevant intelligence and infrastructure information
Digital Forensic Investigators
Seized infrastructure and requested help with reverse engineering

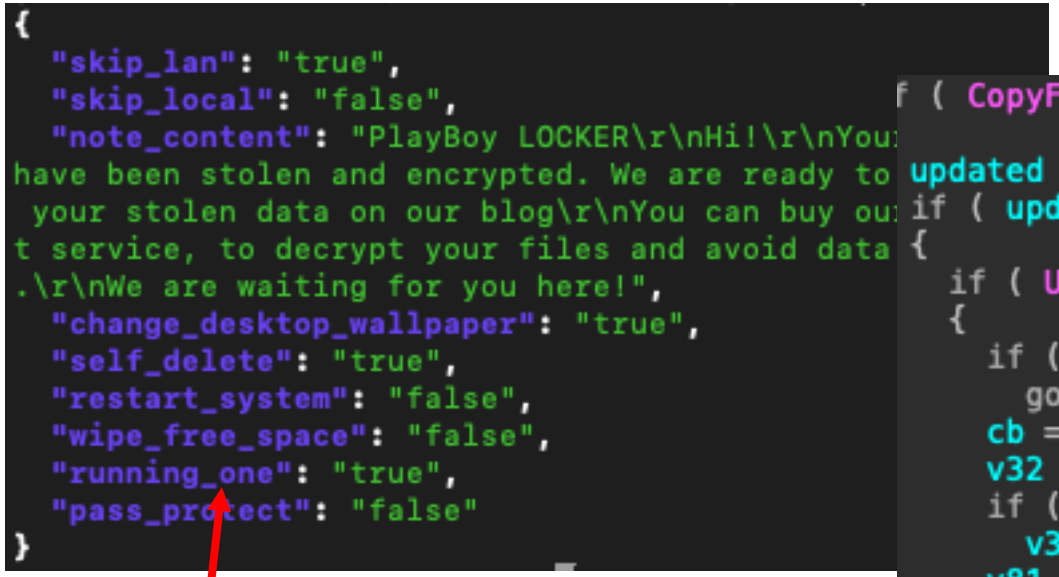

Input for ransomware builder
Generates new key pair
Key Generator
Public key
Private key
JSON
Configuration File
Takes configuration file with parameters
Takes private and public key
Key Replacer Tool
RANSOMWARE
Builder Tool
Ransomware samples built for affiliate
Sets public key in ransomware binaries
Sets parameters from configuration in binaries
RANSOMWARE
RANSOMWARE
RANSOMWARE
RANSOMWARE
Win32
ESXi
NAS (ARM)
NAS (x86)

```
CryptGenRandom(phProv[0], 0x20u, &private_key);
private_key &= 0xF8u;
last_byte_of_private_key = last_byte_of_private_key & 0x3F | 0x40;
v12 = *(_OWORD *)&private_key;
LOBYTE(v12) = private_key;
v13 = unk_420390;
HIBYTE(v13) = last_byte_of_private_key & 0x3F | 0x40;
sub_401A40(v9);
sub_402C70(v11, v7, (int)&v12, v9);
sub_402FF0(v10, v7);
sub_401000(v10);
sub_4014D0(v8);
sub_4015B0(v8);
v7[0] = v8[0];
v7[1] = v8[1];
v7[2] = v8[2];
v7[3] = v8[3];
v7[4] = v8[4];
sub_401D70((int)&public_key, (int *)v7);
sub_404430((int)"Curve25519 keys generated.\n");
sub_4044A0("Public Key", &public_key);
sub_4044A0("Private Key", &private_key);
phProv[0] = (HCRYPTPROV)"C:\\Users\\administrator\\Desktop\\Software\\Encryptors";
phProv[1] = (HCRYPTPROV)"C:\\Users\\administrator\\Desktop\\Software\\Decryptors";
for ( i = 0; i < 2; ++i )
{
```

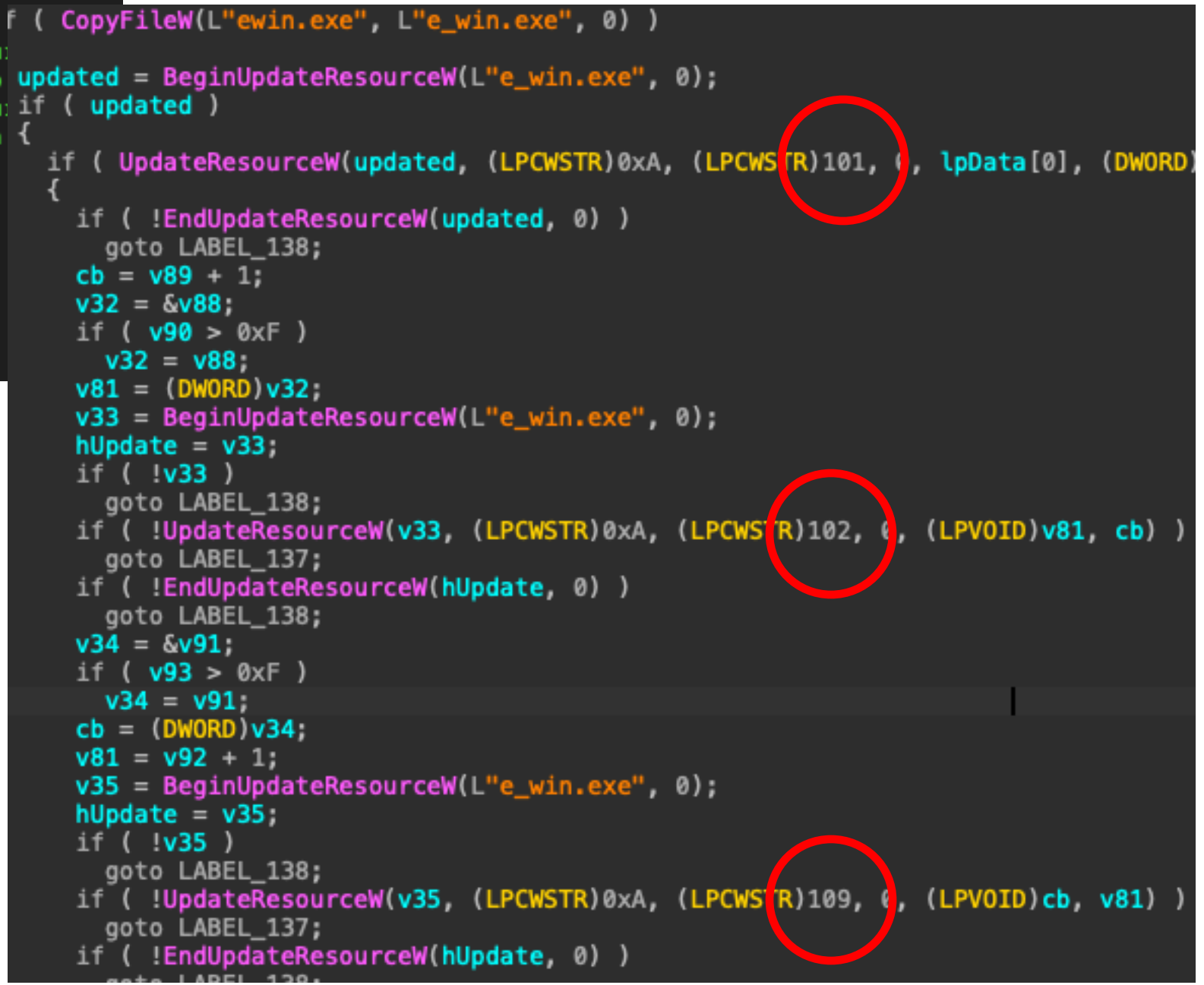
https://cr.yp.to/ecdh.html

```
FileSize = GetFileSize(FileA, 0);
decryptor_buf = (char *)malloc(FileSize);
ReadFile(v6, decryptor_buf, FileSize, &NumberOfBytesRead, 0);
CloseHandle(v6);
v9 = 0;
if ( FileSize )
{
  while ( 2 )
  {
    i = 0;
    ptr = &decryptor_buf[v9];
    while ( ptr[i] == aCurvpattern[i] )        // Looks for pattern 'curvpattern' and writes private key directly after.
    {
      if ( (unsigned int)++i >= 11 )
      {
        if ( !ptr )
          goto LABEL_11;
        *(_OWORD *)ptr = *private_key;
        *((_OWORD *)ptr + 1) = private_key[1];
        v12 = CreateFileA(Buffer, 0x40000000u, 0, 0, 2u, 0x80u, 0);
        if ( v12 == (HANDLE)-1 )
        {
          error_func("Can't create output file %s, bye!\n");
          free(decryptor_buf);
          ExitProcess(0);
        }
        WriteFile(v12, decryptor_buf, FileSize, &NumberOfBytesRead, 0);
        CloseHandle(v12);
        v13 = "\"%s\" created with modified content!\n";
```

Two 16-byte writes after the curvpattern



```
{
  "skip_lan": "true",
  "skip_local": "false",
  "note_content": "PlayBoy LOCKER\r\nHi!\r\nYou
have been stolen and encrypted. We are ready to
 your stolen data on our blog\r\nYou can buy ou
t service, to decrypt your files and avoid data
.\r\nWe are waiting for you here!",
  "change_desktop_wallpaper": "true",
  "self_delete": "true",
  "restart_system": "false",
  "wipe_free_space": "false",
  "running_one": "true",
  "pass_protect": "false"
}
```



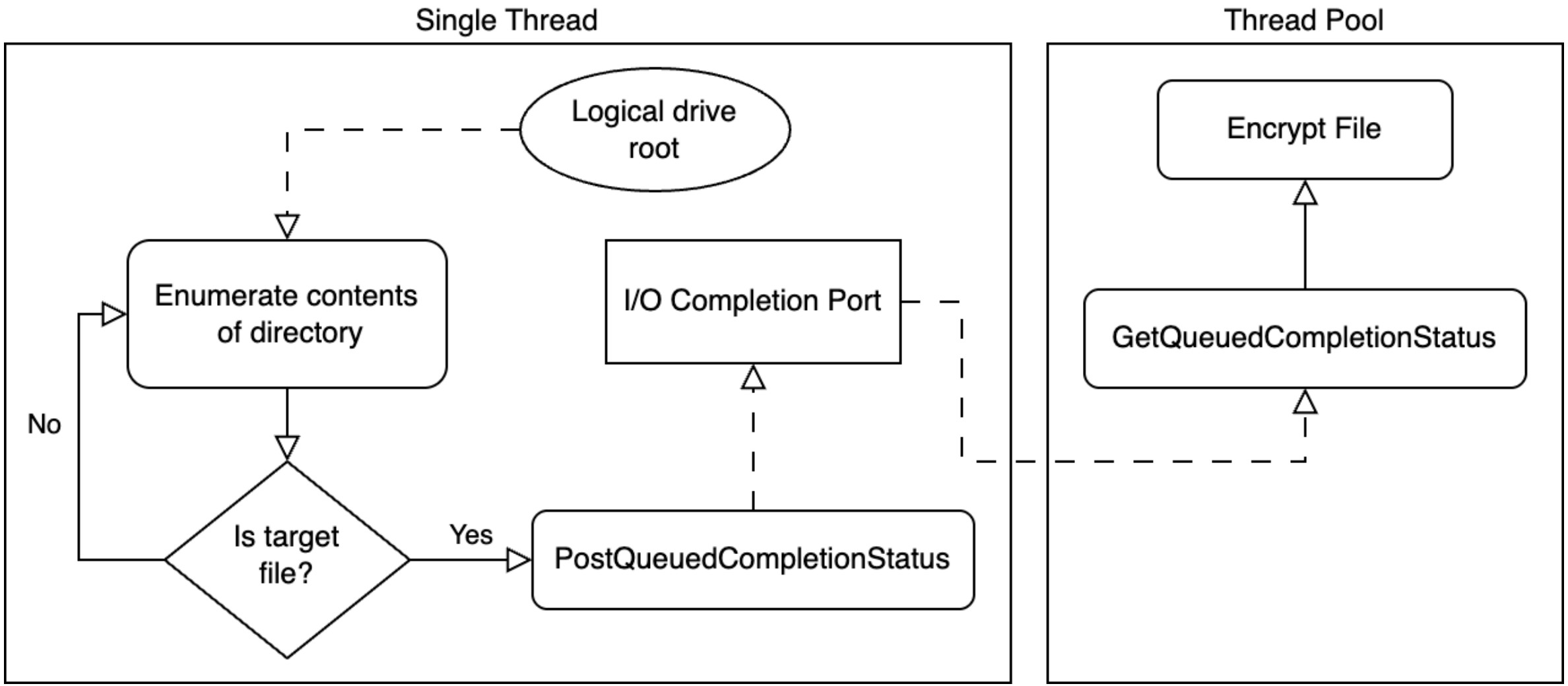
```
f ( CopyFileW(L"ewin.exe", L"e_win.exe", 0) )

updated = BeginUpdateResourceW(L"e_win.exe", 0);
if ( updated )
{
  if ( UpdateResourceW(updated, (LPCWSTR)0xA, (LPCWSTR)101, 0, lpData[0], (DWORD)
  {
    if ( !EndUpdateResourceW(updated, 0) )
      goto LABEL_138;
    cb = v89 + 1;
    v32 = &v88;
    if ( v90 > 0xF )
      v32 = v88;
    v81 = (DWORD)v32;
    v33 = BeginUpdateResourceW(L"e_win.exe", 0);
    hUpdate = v33;
    if ( !v33 )
      goto LABEL_138;
    if ( !UpdateResourceW(v33, (LPCWSTR)0xA, (LPCWSTR)102, 0, (LPVOID)v81, cb) )
      goto LABEL_137;
    if ( !EndUpdateResourceW(hUpdate, 0) )
      goto LABEL_138;
    v34 = &v91;
    if ( v93 > 0xF )
      v34 = v91;
    cb = (DWORD)v34;
    v81 = v92 + 1;
    v35 = BeginUpdateResourceW(L"e_win.exe", 0);
    hUpdate = v35;
    if ( !v35 )
      goto LABEL_138;
    if ( !UpdateResourceW(v35, (LPCWSTR)0xA, (LPCWSTR)109, 0, (LPVOID)cb, v81) )
      goto LABEL_137;
    if ( !EndUpdateResourceW(hUpdate, 0) )
```

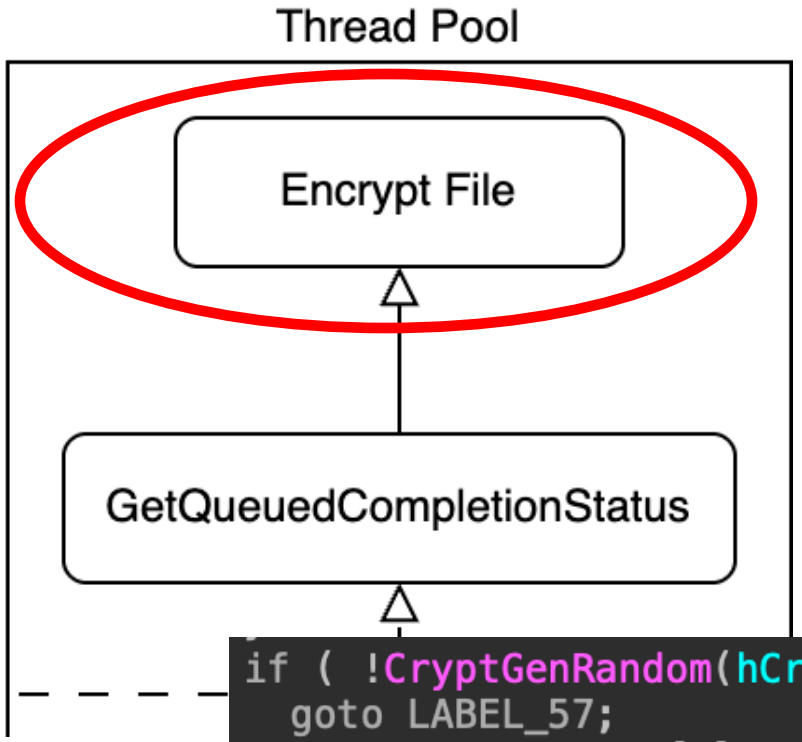
Optional mutex check

PEB + 188 = NtGlobalFlag

```
ModuleHandleA = GetModuleHandleA("ntdll.dll");
NtQueryInformationProcess = (NTSTATUS (__stdcall *)(HANDLE, PROCESSINFOCLASS, PVOID, ULONG, PULONG))GetProcAddress(ModuleHandleA, "NtQueryInformationProcess");
if ( !NtQueryInformationProcess
  || (v7 = GetCurrentProcess(), NtQueryInformationProcess(v7, ProcessBasicInformation, &Block.field_8, 24, 0) < 0)
  || (*(_BYTE *)(Block.field_C + 188) & 0x70) == 0 )
{
  v8 = GetCurrentProcess();
  pbDebuggerPresent = 0;
  v9 = GetModuleHandleA("ntdll.dll");
  ProcAddress = (NTSTATUS (__stdcall *)(HANDLE, PROCESSINFOCLASS, PVOID, ULONG, PULONG))GetProcAddress(
                                                                                        v9,
                                                                                        "NtQueryInformationProcess");
  if ( !ProcAddress || ProcAddress(v8, ProcessDebugPort, &pbDebuggerPresent, 4, 0) < 0 || !pbDebuggerPresent )
  {
    memset(&Context.Dr0, 0, 0x2C8u);
    Context.ContextFlags = 65552;
    CurrentThread = GetCurrentThread();
    if ( !GetThreadContext(CurrentThread, &Context) || !Context.Dr0 && !Context.Dr1 && !Context.Dr2 && !Context.Dr3 )
    {
```

Checks for any hardware breakpoints set


Single Thread
Thread Pool
Logical drive root
Enumerate contents of directory
I/O Completion Port
Encrypt File
GetQueuedCompletionStatus
No
Is target file?
Yes
PostQueuedCompletionStatus




```
{
  read_size = 10485760;
}
if ( !ReadFile(hFile, read_buffer, read_size, &bytesRead, 0) )
  break;
if ( !bytesRead )
  goto LABEL_56;
hc128_in_place(&encrypt_state, read_buffer, read_buffer, bytesRead);
v31.HighPart = -(bytesRead != 0);
v31.LowPart = -bytesRead;
if ( !SetFilePointerEx(hFile, v31, 0, FILE_CURRENT)
  || !WriteFile(hFile, read_buffer, bytesRead, &bytes_written, 0) )
```

```
if ( !CryptGenRandom(hCryptProv, 0x20u, keys.FileSecretKey) )// Generates secret key here
  goto LABEL_57;
keys.FileSecretKey[0] &= 0xF8u;                 // Secret key trimming
keys.FileSecretKey[31] = keys.FileSecretKey[31] & 0x3F | 0x40;
curve25519_point_multiply(footer.FooterFileKey, keys.FileSecretKey, &base_point);
curve25519_point_multiply(&keys, keys.FileSecretKey, AttackerPublicKey);
memset(&footer.HashLength, 0, 20);
*(_OWORD *)footer.sha256_state = sha_constants_0;
*(_OWORD *)&footer.sha256_state[16] = sha_constants_1;
*(_OWORD *)&footer.sha256_state[32] = sha_constants_2;
*((_QWORD *)&v17 + 1) = *((_QWORD *)&sha_constants_3 + 1);
*(_OWORD *)&footer.sha256_state[48] = sha_constants_3;
compute_sha256(&footer, &keys, 0x20u);          // Derive encryption key from shared secret.
```

```
00000000
00000000 footer_struct   struc ; (sizeof=0x48, copyof_144)
00000000 FilePublicKey   db 32 dup(?)
00000020 Crc32           dd ?
00000024 unknown         dd ?
00000028 Marker          db 32 dup(?)
00000048 footer_struct   ends
00000048
```

```
qmemcpy(&footer_struc.Marker, "Tis coolis diffuse very andomly!", 32);
```

```
ptr = (unsigned __int8 *)out;
checksum = 'gnod';
counter = 64;
do
{
  c = *ptr++;
  checksum = crc32_table[c ^ HIBYTE(checksum)] ^ (checksum << 8);
  --counter;
}
while ( counter );
footer_struc.Crc32 = checksum;
```

```
checksum = 'gnod';
i = 64;
do
{
  c = (unsigned __int8)*ptr++;
  checksum = crc32_table[c ^ HIBYTE(checksum)] ^ (checksum << 8);
  --i;
}
while ( i );
if ( Buffer[8] != checksum )
{
  MessageBoxW(0, this, L"Key broken!", 0);
  goto LABEL_32;
}
```

```
SetFilePointerEx(FileW, (LARGE_INTEGER)(FileSize.QuadPart - 72), 0, FILE_BEGIN);
ReadFile(FileW, Buffer, 72u, &NumberOfBytesRead, 0);
if ( Buffer[10] != ' siT'
  || Buffer[11] != 'looc'
  || Buffer[12] != 'd si'
  || Buffer[13] != 'uffi'
  || Buffer[14] != 'v es'
  || Buffer[15] != ' yre'
  || Buffer[16] != 'odna'
  || Buffer[17] != '!ylm' )
{
```

Marker in footer

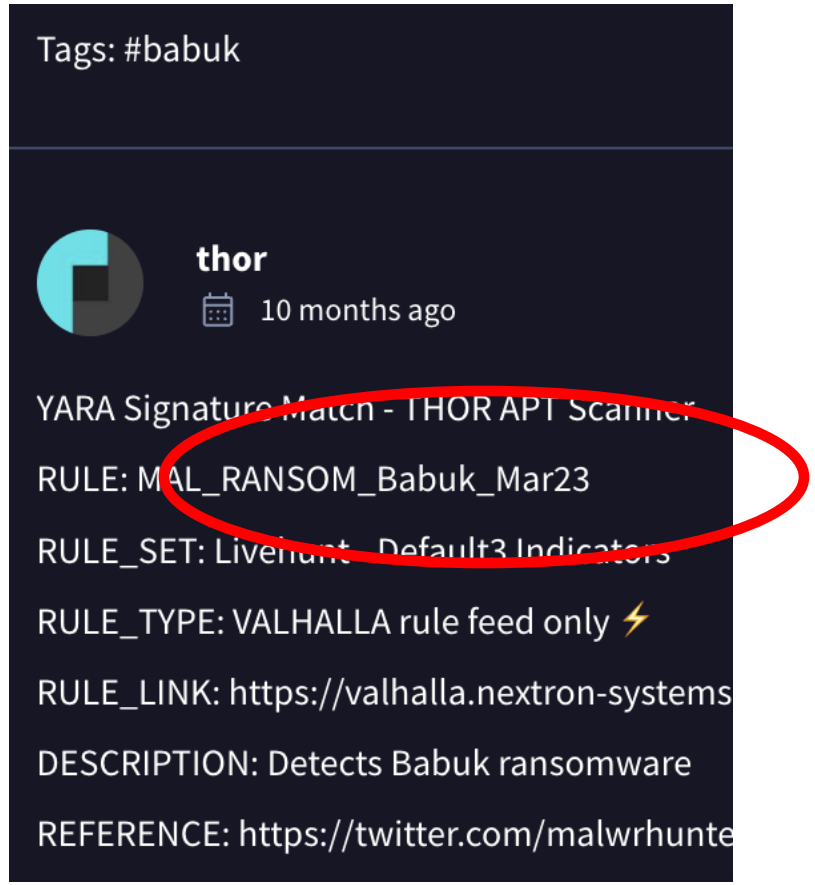
```
marker[0] = 0xAB;
marker[1] = 0xBC;
marker[2] = 0xCD;
marker[3] = 0xDE;
marker[4] = 0xEF;
marker[5] = 0xF0;
v37 = (_BYTE *)runtime_makeslice((int)RTYPE_uint8, 32, 32);
v15 = io_ReadAtLeast(dword_81BFFD8, dword_81BFFDC, (int)v37, 32, 32, 32);
if ( v37 != v19 )
  runtime_memmove((int)v19, (int)v37, 32);
golang_org_x_crypto_curve25519_ScalarBaseMult((int)v18, (int)v19);
golang_org_x_crypto_curve25519_ScalarMult((int)v17, (int)v19, (int)&unk_81B8400);
v12 = runtime_concatstring2(0, v41, *(int *)v42, (int)".plboy", 6);
if ( os_rename(v41, *(int *)v42, v12, v15)
  || (v13 = runtime_concatstring2(0, v41, *(int *)v42, (int)".plboy", 6),
      v11 = (os_File *)os_OpenFile(v13, v15, 2, 0),
      v13) )
```

```
v8.ptr = v22;
*(_QWORD *)&v8.len = 0x2000000020LL;
crypto_sha256_Sum256(v8, v10);
((void (*)(void))loc_80B5F80)();
((void (*)(void))loc_80B5A96)();
DWORD1(v15) = golang_org_x_crypto_chacha20_newUnauthenticatedCipher((int)&v31, (int)v22, 32, 32, (int)v21, 12, 22);
if ( DWORD2(v15) )
  goto LABEL_14;
```

```
v2 = fopen("/dev/urandom", "r");
if ( !v2 )
{
  perror("fopen");
  exit(1);
}
v3 = v2;
if ( a2 != fread(ptr, 1uLL, a2, v2) )
{
  perror("fread");
  fclose(v3);
  exit(1);
}
return fclose(v3);
```

```
get_random_bytes(&secret_key, 32);
secret_key.m128i_i8[0] &= 0xF8u;
HIBYTE(v14) = HIBYTE(v14) & 0x3F | 0x40;
curve25519_point_multiply(file_pubkey, &secret_key, (int *)curve25519_base_point);
curve25519_point_multiply(v15, &secret_key, (int *)aCurvpattern);
secret_key = 0LL;
v14 = 0LL;
init_sha256_state(v7);
sha256_0((unsigned int *)v7, v15, 0x20u);
sub_7CE0((__int64)v7, (__int64)v16);
memset(v7, 0, 0x68uLL);
sha256((__int64)v11, (__int64)v16, 32LL);
v3 = 0;
sub_A830(v8, v11, 0LL, 0LL);
memset(v16, 0, sizeof(v16));
while ( 1 )
{
  v5 = fread(v2, 1uLL, 0xA00000uLL, v1);
  v6 = v5;
  if ( !v5 )
  {
    st_size = buf.st_size;
    goto LABEL_11;
  }
  v3 += v5;
  sosemanuk_crypt(v8, v2, v2, v5);
  if ( fseek(v1, -(__int64)v6, 1) )
  {
    log_entry((__int64)"fseek failed for file: %s\n", ptr);
    st_size = buf.st_size;
    goto LABEL_11;
  }
  if ( fwrite(v2, 1uLL, v6, v1) != v6 )
    break;
```

```
rule Playboy_ESXi_Encryptor
{
  meta:
    author = "Gijs Rijnders"
    description = "ESXi encryptor Playboy"
    target_entity = "file"
  strings:
    $sosemanuk_unum32 = { 00 00 00 00 13 CF 9F E1 26 37 97 6B 35 F8 08 8A }
    $fopen = "fopen"
    $rb = "r+b"
    $vmdk = ".vmdk"
    $vmem = ".vmem"
    $vswp = ".vswp"
    $vmsn = ".vmsn"
  condition:
    all of them and (uint32(0x0) == 0x464c457f)
}
```



```
qmemcpy(
  data,
  "~~~ LockBit 3.0 the world's fastest ransomware since 2019~~~\n"
  "\n"
  ">>>> Your data are stolen and encrypted\n"
  "\n"
  ">>>> What guarantees that we will not deceive you? \n"
  "\n"
  "\tWe are not a politically motivated group and we do not need anything other
  "    \n"
  "\tIf you pay, we will provide you the programs for decryption and we will del
  "\tLife is too short to be sad. Be not sad, money, it is only paper.\n"
  "    \n"
  "\tIf we do not give you decrypters, or we do not delete your data after payme
  "future. \n"
  "\tTherefore to us our reputation is very important. We attack the companies w
  "ed victim after payment.\n"
  "    \n"
  "\tYou can obtain information about us on twitter https://twitter.com/hashtag/
  "    \n"
  "\n"
  ">>>> you can contact us in mail or tox.\n"
  "\t\n"
  "\tTox ID LockBitSupp: CCF12BE4CF49692C4AF295EA10DE1910EDC0CA55E1359B9F66506A0
  "\tmail Support: xndxaxia@proton.me\n"
  "\t\n"
  ">>>> Your personal DECRYPTION ID: B7568014A48684D6D525F3F3722638C4\n"
  "\n"
  ">>>> Warning! Do not DELETE or MODIFY any files, it can lead to recovery prob
  "\n"
  ">>>> Warning! If you do not pay the ransom we will attack your company repeat
```

thor

5 months ago

Match - THOR APT Scanner

SP_RANSOM_LockBit_RansomNote_Feb24

ehunt - Suspicious3 Indicators

ommunity

tps://github.com/Neo23x0/signature-base/search?q=MAL_SUSP_RANSOM_LockB

Detects the LockBit ransom note file 'LockBit-DECRYPT.txt' which is a sign of a Lo

tps://www.huntress.com/blog/slashandgrab-screen-connect-post-exploitation-i

5 months ago

Match - THOR APT Scanner

buk_Ransomware_Feb24

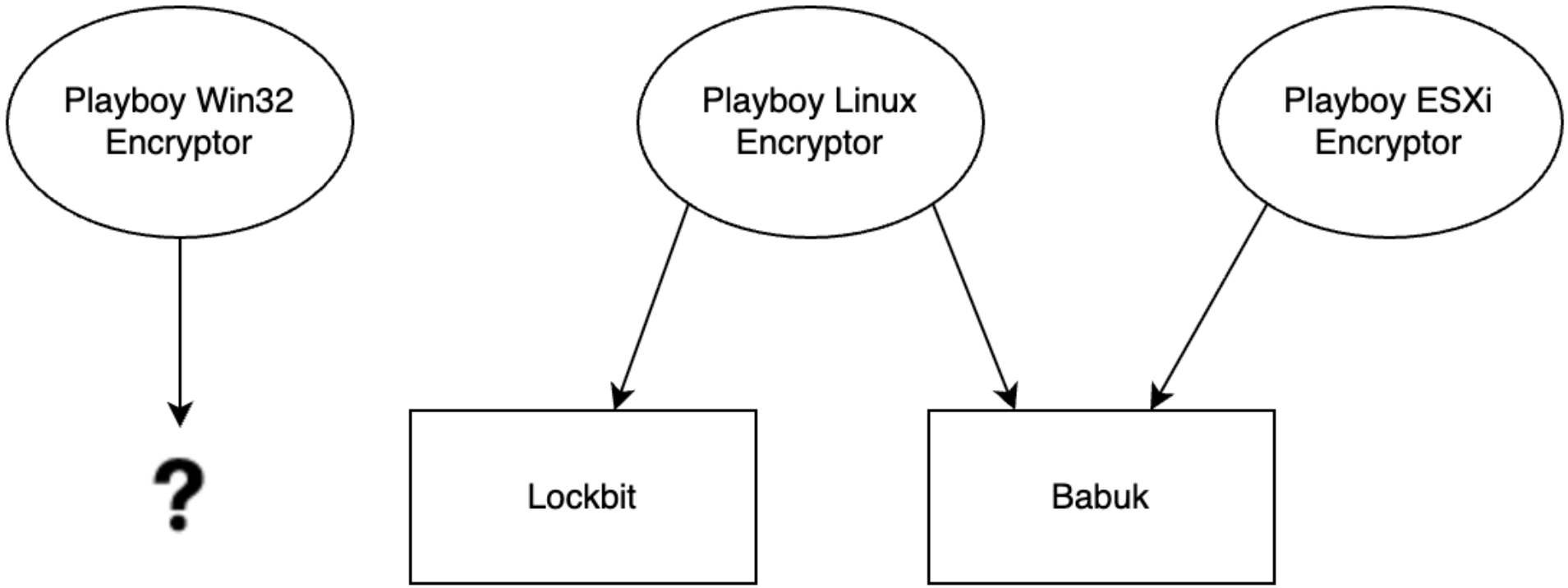
ehunt - Default3 Indicators

LHALLA rule feed only ⚡

RULE_LINK: https://valhalla.nextron-systems.com/info/rule/MAL_Babuk_Ransomware_Feb24

DESCRIPTION: Detects Babuk ransomware

https://blogs.npav.net/blogs/post/ransomware-gangs-impersonate-lockbit-to-intimidate-victims-and-leverage-aws-in-latest-attacks


Playboy Win32
Encryptor
?
Playboy Linux
Encryptor
Playboy ESXi
Encryptor
Lockbit
Babuk