Everything Starts From CyCraft

CYCRAFT

Prometheus-Decryptor

Prometheus-Decryptor is an project to decrypt files encrypted by Prometheus ransomware.

Command Arguments

```
Usage of ./bin/prometheus_decrypt:
  -b string
        Custom search with byte value. (i.e. \xde\xad\xbe\xef -> deadbeef)
        Please use ?? to match any byte (i.e. de??beef)
  -c    Use current tickcount. (only support in Windows)
  -d int
        Decrypt size when guessing. The default size is 100, and you can specify your own size corresponding to your search pattern.
        0 stands for the guessing file size, and -1 stands for the max header size 100 except for Microsoft documents. (default -1)
  -e string
        Search file extension.
  -f int
        Found candidate. (default 1)
  -i string
        Input encrypted file.
  -k string
        Decrypt with this key.
  -m int
        Move backward m minutes from the current decrypted seed when guessing the next sample. (default 30)
  -o string
        Output decrypted file.
  -p int
        Use n thread. (default 1)
  -r    Reversed tickcount.
  -s string
        Custom search with regular expression.
  -t int
        Start tickcount.
```

Usage

Guess password

Guess the password of a png image from tickcount 0.

```
./prometheus_decrypt -i ./sample/CyCraft.png.PROM\[prometheushelp@mail.ch\] -o ./output/CyCraft.png -e png -p 16
```

In this command, there are 4 arguments: - i: input encrypted file - o: output file - e: search file format - p: thread count

Reversed Tickcount

Guess the password of a png image from tickcount 100000 in reversed order.

```
./prometheus_decrypt -i ./sample/CyCraft.png.PROM\[prometheushelp@mail.ch\] -o ./output/CyCraft.png -e png -p 16 -t 100000 -r
```

There are 2 additional arguments: - t: start from 100000 - r: reversed order (100000...0)

Guess from current tickcount (only for Windows)

Guess the password of a png image from the current tickcount in reversed order. This feature is usually used with reversed order.

```
./prometheus_decrypt -i ./sample/CyCraft.png.PROM\[prometheushelp@mail.ch\] -o ./output/CyCraft.png -e png -p 16 -c -r
```

There is an additional argument: - c: start from the current tickcount

Decrypt (Encrypt) with a key

Decrypt (Encrypt) a file with a provided key.

```
./prometheus_decrypt -i ./sample/CyCraft.png.PROM\[prometheushelp@mail.ch\] -o ./output/CyCraft.png -k "+@[%T-mZSh+E[^^i{W:dpwnhdL4<b8D4}]]"
```

There is an additional argument: - k: provided key

Guess password with custom format (regular expression)

Guess the password of a text file with a known string "we had another great".

```
./prometheus_decrypt -i ./sample/test.txt.enc -o ./output/test.txt -p 16 -s "we had another great" -d 0
```

There are two additional arguments: - s: regular expression to match the decrypted file - d: the decrypted size when guessing. It's default value is 100. Since the custom search pattern is not limited to first 100 bytes, we use 0 here to decrypt the whole files.

Guess password with custom format (bytes pattern)

Guess the password of a png file with its header in hex.

```
./prometheus_decrypt -i ./sample/CyCraft.png.PROM[prometheushelp@mail.ch] -o ./output/CyCraft.png -p 16 -b '89??4e??0d??1a0a??00' -d 10
```

There is an additional argument: - b: PNG header in hex format. - The full bytes are "8950 4e47 0d0a 1a0a 0000". - We can use ?? to match any byte. - d: since the pattern is the first 10 bytes of png files, we can specify 10 here to enhance the drcryption speed.

Custom search with bytes pattern is much more convenient than regular expression, since there are lots of file format that it can't be performed by visible characters.

Guess password for a directory

Guess the password of a png file with its header in hex.

```
./prometheus_decrypt -i ./sample -o ./output -p 16 -m 1 -f 2
```

There are two additional arguments: - m: Move backward m minutes from the current decrypted seed when guessing the next sample. (default 30) - Use `seed-m*60*1000` as the start tickcount. - f: Found candidate. (default 1) - Limit the candidates found. There may be several candidates to a file, limit its candidates can save time.

Since there are lots of files to decrypt, you can press `Ctrl-c` to skip the current guessing file.

Output

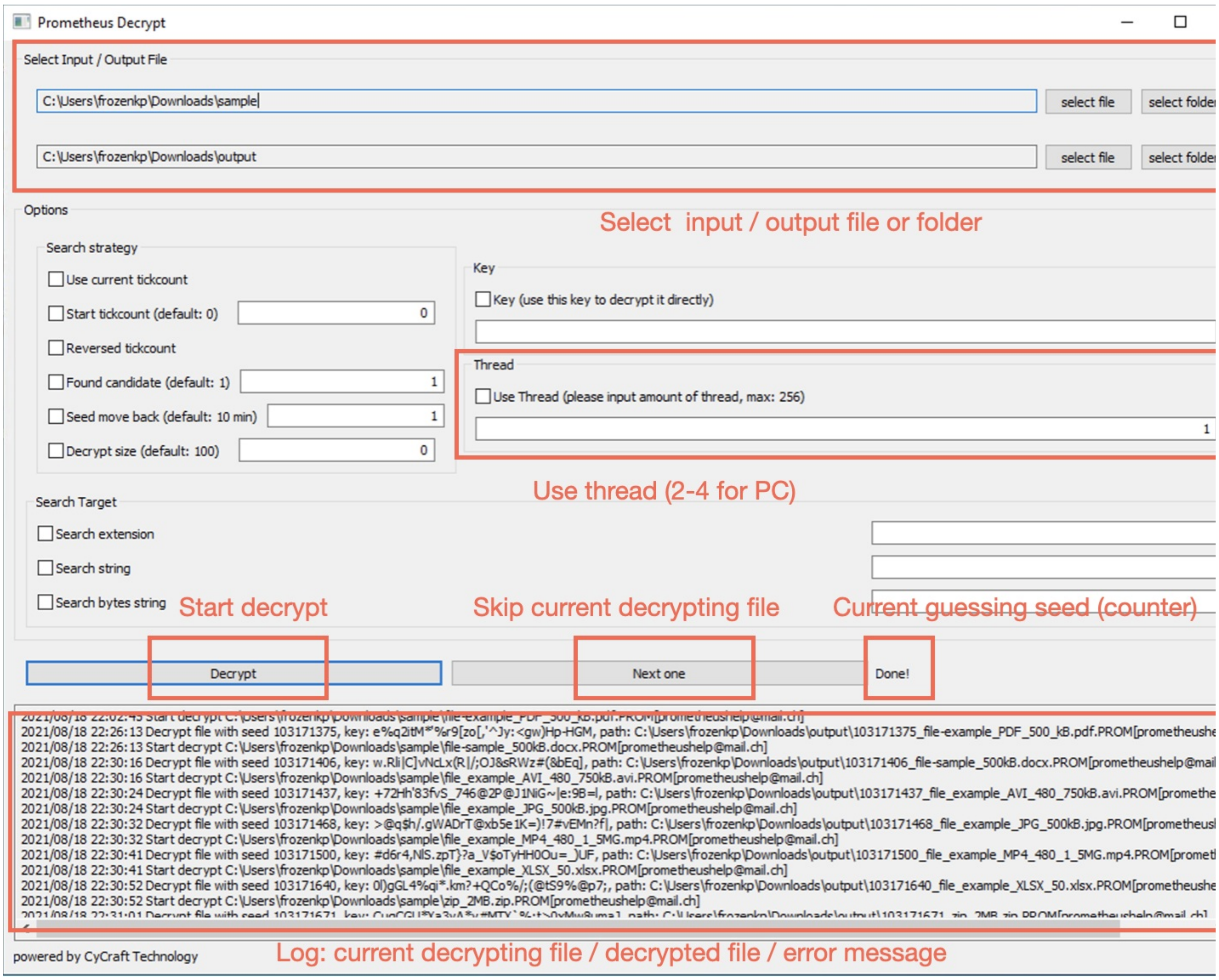
The output should like this. Since we match the file with magic number, it might be matched even a wrong key is provided. Therefore, we keep the decryption process continued to guess. You can terminate it anytime if you find the correct decrypted file.

```
% ./prometheus_decrypt -i ./sample/test.txt.enc -o ./output/test.txt -p 16 -s "we had another great"
Decrypt file with seed 615750, key: +@[%T-mZSh+E[^^i{W:dpwnhdL4<b8D4, path: ./output/615750_test.txt
2795306...
```

GUI

We provide a GUI version for windows users. All features is supported in the GUI version. If you know nothing about programming, please follow the steps below to decrypt your files:

1. Choose a file or folder to decrypt.
2. Choose the output file name or output folder.
3. Select "Use thread" and fill in 2-4 for PC. (Threads usually make the decryption routine faster, but it actually depends on amount of your cpu cores)
4. Click decrypt.
5. There is a counter, which shows the current guessing tickcount.
6. The decrypting result will show in the text block below. (There may be multiple possible key, so the decryption routine will continue to decrypt even find a possible key. You can press "Next one" to skip the current file).



Prometheus Decrypt
Select Input / Output File
C:\Users\frozenkp\Downloads\sample
select file
select folder
C:\Users\frozenkp\Downloads\output
select file
select folder
Select input / output file or folder
Options
Search strategy
Use current tickcount
Start tickcount (default: 0) 0
Reversed tickcount
Found candidate (default: 1) 1
Seed move back (default: 10 min) 1
Decrypt size (default: 100) 0
Key
Key (use this key to decrypt it directly)
Thread
Use Thread (please input amount of thread, max: 256)
1
Use thread (2-4 for PC)
Search Target
Search extension
Search string
Search bytes string
Start decrypt
Skip current decrypting file
Current guessing seed (counter)
Decrypt
Next one
Done!
powered by CyCraft Technology
Log: current decrypting file / decrypted file / error message

```
2021/08/18 22:02:45 Start decrypt C:\Users\frozenkp\Downloads\sample\file-example_PDF_500_kB.pdf.PROM[prometheushelp@mail.ch]
2021/08/18 22:26:13 Decrypt file with seed 103171375, key: e%q2itM*'%r9[zo[,'^Jy:<gw)Hp-HGM, path: C:\Users\frozenkp\Downloads\output\103171375_file-example_PDF_500_kB.pdf.PROM[prometheushe
2021/08/18 22:26:13 Start decrypt C:\Users\frozenkp\Downloads\sample\file-sample_500kB.docx.PROM[prometheushelp@mail.ch]
2021/08/18 22:30:16 Decrypt file with seed 103171406, key: w.Rli|C]vNcLx(R|/;OJ&sRWz#(&bEq], path: C:\Users\frozenkp\Downloads\output\103171406_file-sample_500kB.docx.PROM[prometheushelp@mail
2021/08/18 22:30:16 Start decrypt C:\Users\frozenkp\Downloads\sample\file_example_AVI_480_750kB.avi.PROM[prometheushelp@mail.ch]
2021/08/18 22:30:24 Decrypt file with seed 103171437, key: +72Hh'83fvS_746@2P@J1NiG~|e:9B=l, path: C:\Users\frozenkp\Downloads\output\103171437_file_example_AVI_480_750kB.avi.PROM[promethe
2021/08/18 22:30:24 Start decrypt C:\Users\frozenkp\Downloads\sample\file_example_JPG_500kB.jpg.PROM[prometheushelp@mail.ch]
2021/08/18 22:30:32 Decrypt file with seed 103171468, key: >@q$h/.gWADrT@xb5e1K=)!7#vEMn?f|, path: C:\Users\frozenkp\Downloads\output\103171468_file_example_JPG_500kB.jpg.PROM[prometheus
2021/08/18 22:30:32 Start decrypt C:\Users\frozenkp\Downloads\sample\file_example_MP4_480_1_5MG.mp4.PROM[prometheushelp@mail.ch]
2021/08/18 22:30:41 Decrypt file with seed 103171500, key: #d6r4,NlS.zpT}?a_V$oTyHH0Ou=_)UF, path: C:\Users\frozenkp\Downloads\output\103171500_file_example_MP4_480_1_5MG.mp4.PROM[promet
2021/08/18 22:30:41 Start decrypt C:\Users\frozenkp\Downloads\sample\file_example_XLSX_50.xlsx.PROM[prometheushelp@mail.ch]
2021/08/18 22:30:52 Decrypt file with seed 103171640, key: 0l)gGL4%qi*.km?+QCo%/;(@tS9%@p7;, path: C:\Users\frozenkp\Downloads\output\103171640_file_example_XLSX_50.xlsx.PROM[prometheushe
2021/08/18 22:30:52 Start decrypt C:\Users\frozenkp\Downloads\sample\zip_2MB.zip.PROM[prometheushelp@mail.ch]
2021/08/18 22:31:01 Decrypt file with seed 103171671, key: CuqCGU*Xa3vA*v#MTY`%:t>0xMw8uma], path: C:\Users\frozenkp\Downloads\output\103171671_zip_2MB.zip.PROM[prometheushelp@mail.ch]
```

Build

```
make win32    # windows 32 bits
make win64    # windows 64 bits
make linux    # linux
make win32GUI # windows 32 bits GUI (built on windows)
make win64GUI # windows 64 bits GUI (build on windows)
```

Supported File Format

We match the magic number with https://github.com/h2non/filetype. Here is the file type we currently support:

Image

- **jpg** - `image/jpeg`
- **png** - `image/png`
- **gif** - `image/gif`
- **webp** - `image/webp`
- **cr2** - `image/x-canon-cr2`
- **tif** - `image/tiff`
- **bmp** - `image/bmp`
- **heif** - `image/heif`
- **jxr** - `image/vnd.ms-photo`
- **psd** - `image/vnd.adobe.photoshop`
- **ico** - `image/vnd.microsoft.icon`
- **dwg** - `image/vnd.dwg`

Video

- **mp4** - `video/mp4`
- **m4v** - `video/x-m4v`
- **mkv** - `video/x-matroska`
- **webm** - `video/webm`

- **mov** - `video/quicktime`
- **avi** - `video/x-msvideo`
- **wmv** - `video/x-ms-wmv`
- **mpg** - `video/mpeg`
- **flv** - `video/x-flv`
- **3gp** - `video/3gpp`

Audio

- **mid** - `audio/midi`
- **mp3** - `audio/mpeg`
- **m4a** - `audio/m4a`
- **ogg** - `audio/ogg`
- **flac** - `audio/x-flac`
- **wav** - `audio/x-wav`
- **amr** - `audio/amr`
- **aac** - `audio/aac`

Archive

- **epub** - `application/epub+zip`
- **zip** - `application/zip`
- **tar** - `application/x-tar`
- **rar** - `application/vnd.rar`
- **gz** - `application/gzip`
- **bz2** - `application/x-bzip2`
- **7z** - `application/x-7z-compressed`
- **xz** - `application/x-xz`
- **zstd** - `application/zstd`
- **pdf** - `application/pdf`
- **exe** - `application/vnd.microsoft.portable-executable`
- **swf** - `application/x-shockwave-flash`
- **rtf** - `application/rtf`
- **iso** - `application/x-iso9660-image`
- **eot** - `application/octet-stream`
- **ps** - `application/postscript`
- **sqlite** - `application/vnd.sqlite3`
- **nes** - `application/x-nintendo-nes-rom`
- **crx** - `application/x-google-chrome-extension`
- **cab** - `application/vnd.ms-cab-compressed`
- **deb** - `application/vnd.debian.binary-package`
- **ar** - `application/x-unix-archive`
- **Z** - `application/x-compress`
- **lz** - `application/x-lzip`
- **rpm** - `application/x-rpm`
- **elf** - `application/x-executable`
- **dcm** - `application/dicom`

Documents

- **doc** - `application/msword`
- **docx** - `application/vnd.openxmlformats-officedocument.wordprocessingml.document`
- **xls** - `application/vnd.ms-excel`
- **xlsx** - `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet`
- **ppt** - `application/vnd.ms-powerpoint`
- **pptx** - `application/vnd.openxmlformats-officedocument.presentationml.presentation`

Font

- **woff** - `application/font-woff`
- **woff2** - `application/font-woff`
- **ttf** - `application/font-sfnt`
- **otf** - `application/font-sfnt`

Application

- **wasm** - `application/wasm`
- **dex** - `application/vnd.android.dex`
- **dey** - `application/vnd.android.dey`

How it work ?

Prometheus ransomware use salsa20 with a tickcount-based random password to encrypt. The size of the random password is 32 bytes, and every character is visible character. Since the password use tickcount as the key, we can guess it brutally.

CYCRAFT

Everything Starts From Security