Reversing Reprise License Manager, by KANGALOOJ

An enterprise-class license management that made to be reversed easily.

About this tutorial

After a quick review on Synkro tutorial on RLM reversing that helps me a lot to know RLM, I found some critical errors in Synkro's tutorial like public key length that he said is always 225 or 226, but it's wrong, my research on RLM shows that it can vary more widely (I reached 224 to 227). So I decided to write another tutorial to crack RLM. Anyway I really want to say thanks to **Synkro** for his tutorial on RLM.

Introduction to RLM

Wiki says: "The original FLEXlm development team moved on to develop the Reprise License Manager (RLM) in 2006."

RLM uses "Public key/Private key" strategy for licensing just like FLEXlm but key pairs are longer than FLEXlm that introduce more security in licensing. Don't panic! It's a joke! My experience with FLEXlm and RLM says that FLEXlm was more complex to reverse in comparison with RLM.

After reading this tutorial, you will be able to crack any RLM protected app which made until now.

✓ Here I teach you to reverse Win32 (x86) targets, but for x64 versions and other platforms you can use this tutorial with small modifications.

What you need?

RLM Helper (ver. 2.0), that I created myself! This tool needed only if you want to automate some works and nothing else needed if you use this amazing tool!

OllyDbg (I use ver. 2.01), for x86 executable, you can use IDA Pro for other PEs.

RLM SDK, to make rlmsign.exe

Visual Studio 6.0 or 2003 or 2005 or 2008 or 2010 to compile RLM SDK!

How to Crack?

There is two ways to crack RLM target:

Ultra Easy Way:

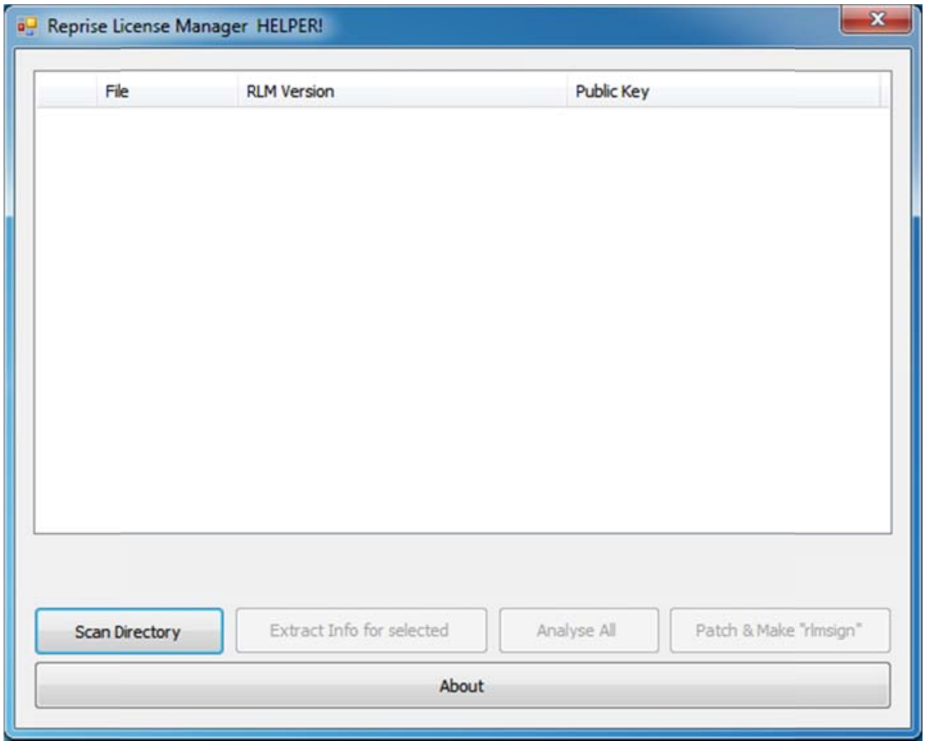
Using RLM Helper v2.0, crack procedure is so easy:

Note: You **don't** need OllyDbg/IDA, RLM SDK and Visual Studio in this way!

1) Scan target directory with RLM helper to find files which must be patched!

2) Analyze all founded files to extract needed info!

3) Hit "Patch & Make **rlmsign.exe**" button and enjoy!



Another Way:

Using OllyDbg, RLM SDK, and Visual Studio!

RLM Public Keys always starts with "`30-81-??-02-40`" or "`30-81-??-02-41`"!
Open target in OllyDbg and find occurrences of hex string "`30-81-??-02-40`" or "`30-81-??-02-41`", "`??`" is depended on Public Key length. You can find public key length by adding 3 to "`??`"! For example public key length of "`30-81-DD-02-40-…`" is 0xDD+3 = 224!

You will find two or three public keys in the target, which last one is **ISV**’s public key and others are for RLM internal use, so don’t patch them at all.

Now open visual studio console and switch to RLM SDK directory and then make some preliminary apps like **rlmgenkeys.exe** by command “**nmake**“! Now you can make key-pairs using **rlmgenkeys.exe**!

You must make a key-pair that it’s public key length matches target public key, so you may need to run rlmgenkeys.exe several times to reach desired public key length.

After producing appropriate key pair, patch ISV public key in the target (last occurrence of public key in target).

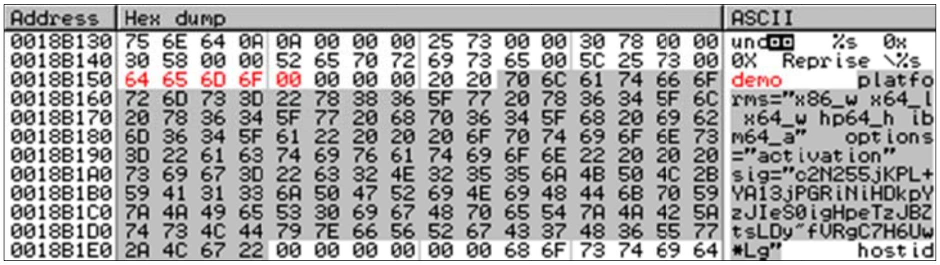
You must find **ISV name** from target and **LICENSE_TO_RUN** string from target! You can do it using **OllyDbg**, load target in **OllyDbg** and search for all referenced strings in target. Then search for string ‘**sig=”**’ and you will find two or three occurrences of this string, one of them is like XML strings <...sig=”..”..> and other one is something like:

```
platforms="x86_w x64_l x64_w hp64_h ibm64_a"   options="activation"   sig="c2N25…..”
```

This is **LICENSE_TO_RUN** string of **ISV**, copy it to “license_to_run.h” file located in “**src**” folder of RLM SDK, and modify #define RLM_LICENSE_TO_RUN …. line with **LICENSE_TO_RUN** string you found before! But remember to scape quotation marks with “\”! Here is an example of modified one:

```
#define RLM_LICENSE_TO_RUN “platforms=\"x86_w x64_l x64_w hp64_h ibm64_a\" \
options=\"activation\"   sig=\"c2N25…..\"”
```

Just before **LICENSE_TO_RUN** string in target, you can find **ISV** name:



```
Address  Hex dump                                         ASCII
0018B130 75 6E 64 0A 0A 00 00 00 25 73 00 00 30 78 00 00 und□□   %s  0x
0018B140 30 58 00 00 52 65 70 72 69 73 65 00 5C 25 73 00 0X  Reprise \%s
0018B150 64 65 6D 6F 00 00 00 00 20 20 70 6C 61 74 66 6F demo      platfo
0018B160 72 6D 73 3D 22 78 38 36 5F 77 20 78 36 34 5F 6C rms="x86_w x64_l
0018B170 20 78 36 34 5F 77 20 68 70 36 34 5F 68 20 69 62  x64_w hp64_h ib
0018B180 6D 36 34 5F 61 22 20 20 20 6F 70 74 69 6F 6E 73 m64_a"   options
0018B190 3D 22 61 63 74 69 76 61 74 69 6F 6E 22 20 20 20 ="activation"
0018B1A0 73 69 67 3D 22 63 32 4E 32 35 35 6A 4B 50 4C 2B sig="c2N255jKPL+
0018B1B0 59 41 31 33 6A 50 47 52 69 4E 69 48 44 6B 70 59 YA13jPGRiNiHDkpY
0018B1C0 7A 4A 49 65 53 30 69 67 48 70 65 54 7A 4A 42 5A zJIeS0igHpeTzJBZ
0018B1D0 74 73 4C 44 79 7E 66 56 52 67 43 37 48 36 55 77 tsLDy~fVRgC7H6Uw
0018B1E0 2A 4C 67 22 00 00 00 00 00 00 68 6F 73 74 69 64 *Lg"      hostid
```

Here you can see ISV name “demo” in red color just before **LICENSE_TO_RUN**!

Modify **#define RLM_ISV_NAME "demo"** line in “license_to_run.h” file to match your target ISV name.

Now edit “**makefile**” in main SDK folder and change line “**ISV = demo**” with your target ISV name!

Run “**nmake**” in Visual Studio console and make SDK with new **ISV** name and **RLM_LICENSE_TO_RUN** string.

Now you can sign your license with **rlmsign.exe** produced with nmake!

Enjoy!