Subject: Re: Protect package - A starting copy protection system
Posted by dolik.rce on Sun, 19 Sep 2010 22:50:38 GMT
View Forum Message <> Reply to Message

Hi Max!

This is a very interesting package. I couldn't resist and played with it for a while My conclusions: It works very well and it would definitely stop me from stealing the app.

Few observations:

It works as well with flags "GUI .PROTECT" which cause less recompilation when switching between protected and unprotected mode. (If you want to have encrypted code in other packages than main and still use this trick, you can add PROTECT into 'Accepts' field in package manager.)

You can't declare variables inside the encrypted block, because 'jump to label __end crosses initialization of ...'. It is somewhat cryptic, so it should be probably mentioned in docs. Work around is to put another pair of '{}' between the calls to PROTECT_... macros to limit the scope of variables declared inside.

You can't have two encrypted blocks in one function, as it results into redeclaring variables. Even if the blocks are in different scopes, it fails on duplicate labels. This could be fixed easily, but it is probably not important for real-life usage.

The 'return' in PROTECT_END_FUNC prevents using the macro in functions returning a value. Omitting it causes runtime error, but ugly fix comes to my mind:#define

```
//only for posix here, for win it is similar
PROTECT_END_FUNC(RETURN) \
 RETURN; \
 __end: \
 asm volatile ( \
  "\t.ascii \""PROTECT_END_MARKER"\"\n" \
 )
//and calling it like this (stupid example):
int testfn1(void){
 int j=0;
 PROTECT_START_FUNC(Decrypt);
 PromptOK("testfn1 DECRYPTED SUCCESSFULLY!!!");
 return j; // <- returning in between wouldn't hurt
 PROTECT_END_FUNC(return j);
}
```

One question at the end: Do I understand it right, that the decryption is performed only on first call of the function? So it modifies only the program loaded in memory? If so, I'll seriously consider calling it a voodoo

Good job!
Honza