PAMS library protection

11.06.2012

LIEBHERR

Agenda

- Definition of security level
- Usage of PAMS library
- “Man in the middle” attack
- How to prevent “Man in the middle” attack
- Obfuscation discussion with Mrs. Karl Zerlauth & Andreas Jung

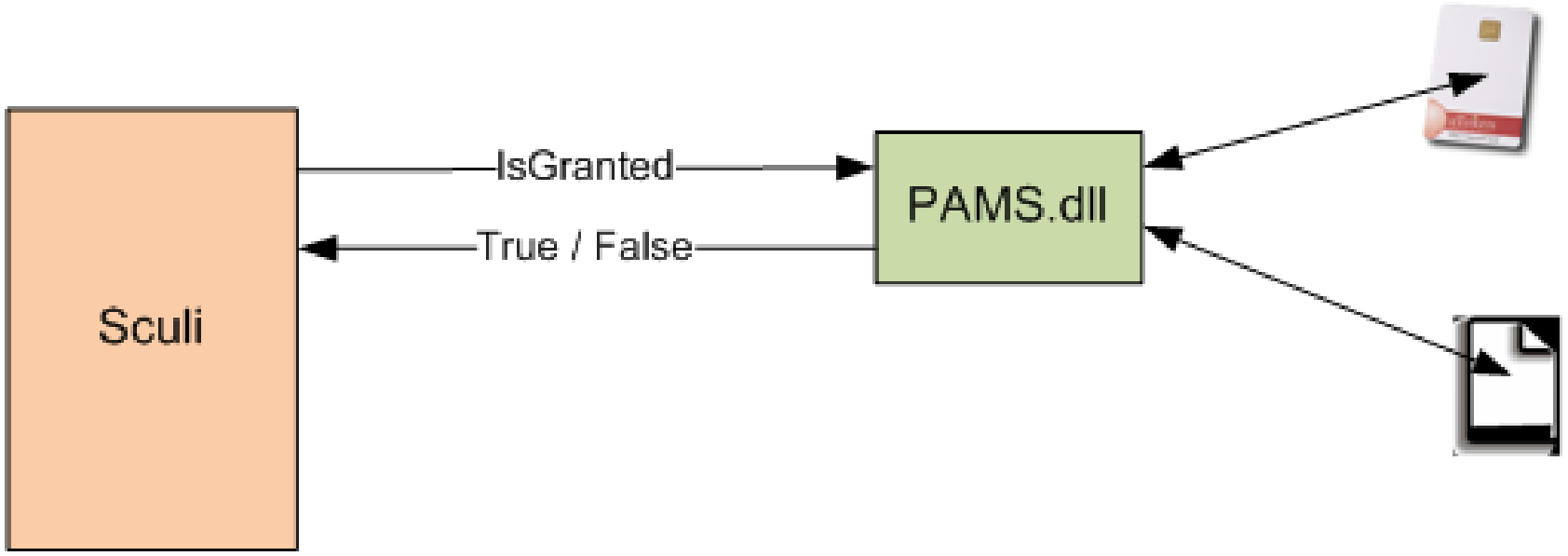
LIEBHERR

Definition of security

- It should not be possible to replace the security.dll by another one (signed with an other private key than the official one)
- It should not be possible to come-between Sculi.exe and the security.dll
- It should not be possible to mock the certificate security chain (same certificate names, created with Makecert for instance)
- It should not be possible to modify a license file
- It should not be possible to use a license file from another user (corresponding to the dongle)
- Trust any binary file which has been signed by the Sculi private key
- Trust the Sculi private key is kept safe and is not distributed to any unnecessary party.
- Do NOT trust the machine configuration, even if it takes admin rights to change it.
- We do not rely on secret only, to keep the application secure. i.e. Read-Only access to the source code is not enough to circumvent the security
- We do not protect ourselves from attempts at hooking Win32 APIs (Win32 native crypto API) and accessing uncyphered data kept in-memory by the dll or the program to protect from an application with the same level of trust as the program to protect.

LIEBHERR

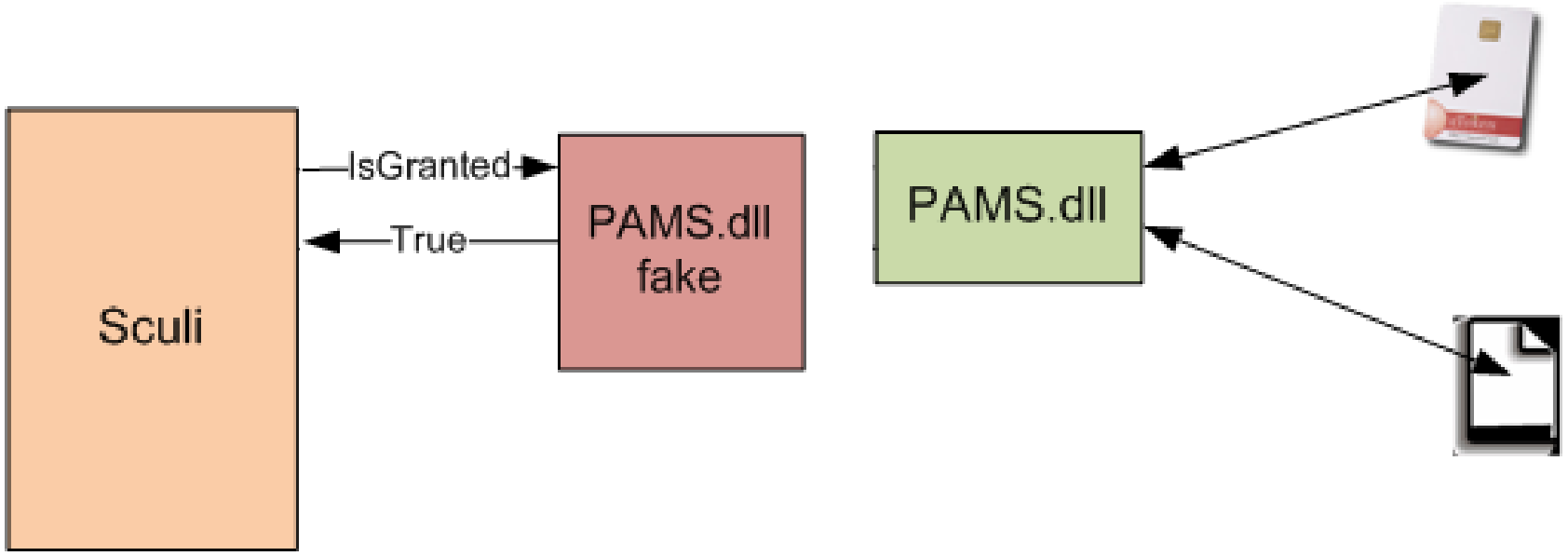
Usage of PAMS library



- Sculi ask the library if an action/feature is granted
- The library answers with true of false

LIEBHERR

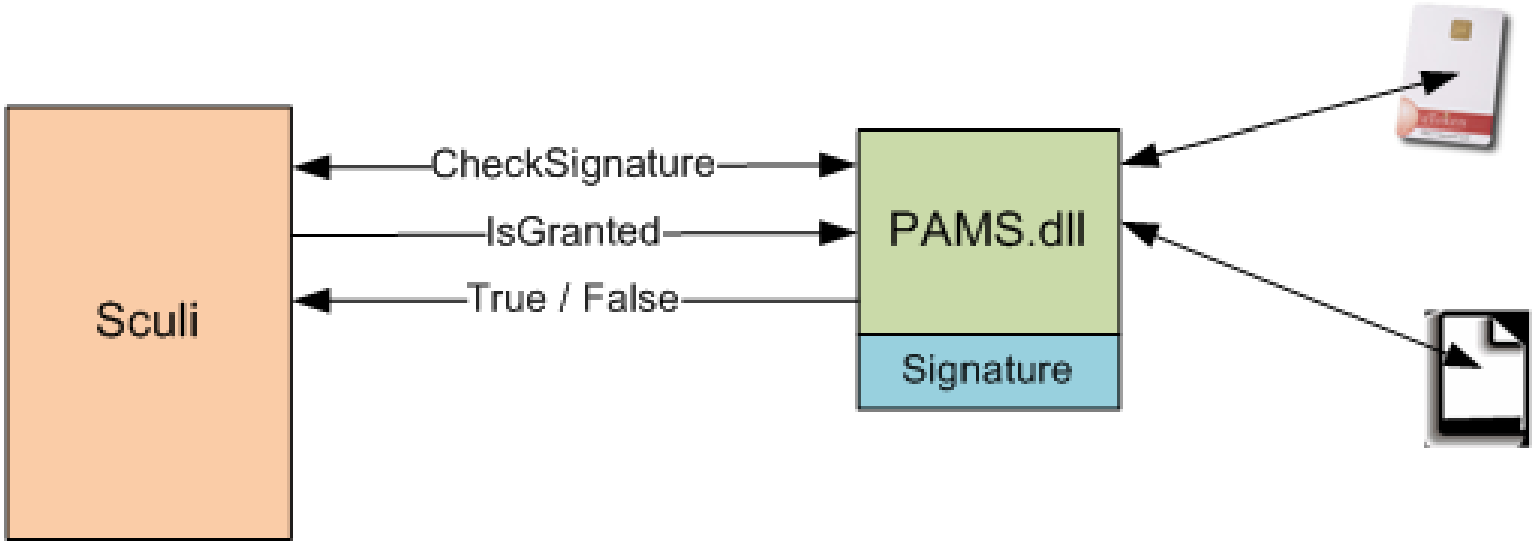
“Man in the middle” attack



- Risk
  - Another library can be placed in between

LIEBHERR

How to prevent the “man in the middle” attack



- The PAMS.dll is signed
- The application must check the signature of the library before using it

LIEBHERR

Obfuscation discussion with Mr. Karl Zerlauth & Mr. Andreas Jung

- 1) What can the attacker achieve if he learns from the code of PAMS.dll
  - if he can attack PAMS license files or other systems behind, there is a need to obfuscate
- 2) What can the attacker achieve if he alters the code of PAMS.dll
  - e.g. if he can create custom PAMS files, which appear valid to the systems relying on PAMS.dll,
  - there is a need of true verification of integrity (cannot be changed easily, -> roundtrip validation)
- 3) What can the attacker achieve if he creates his own PAMS.dll only containing method stubs returning the desired values/objects
  - e.g. if the attacker can create a custom pams.dll with a method which grants him the highest admin level,
  - it is necessary to "hard link" PAMS.dll to each application using it (SCULI etc.).

LIEBHERR

Disadvantage of obfuscation

- When the library uses *reflection*, the business logic will be broken.
- When the library is obfuscated, the *Test Project* must use obfuscated methods/members
  - Test readability is broken
  - In BDD, tests are used to define specifications

LIEBHERR