The VOMS Authentication Library

Vincenzo Ciaschini

EMI Security Workshop

25-26/5/09

Information Society and Media

What VOMS AuthN Library?

- **It is not a very well known fact**
  - But the VOMS APIs also provide a way to setup an SSL connection to use proxies with it.
  - #include <vomsssl.h>
  - The necessary code has always been there, it was just not published.
    - Made public on request from developers.
- **They allow to customize a not yet established SSL connections to accept proxies.**
  - No dependence on Globus to do this.
  - Both client- and server-side.

What does it do, exactly?

- **What it does, exactly ?**
  - Enables usage of GT2, GT3 and RFC proxies (hint: only use the latter) as long as EE certificates.
  - Checks .namespaces and .signing_policy files, if present.
  - Allows freely mixing the different kinds of proxies.
    - In the same chain.
    - Actually pretty common situation.
  - Expands the chain depth limit from OpenSSL.

Lessons learned 1

- **What to support?**
  - What the standards define?
- **Standards are often not respected!**
  - It does not matter what the RFCs say.
  - There are certificates and CAs out there which simply go straight against the RFC.
    - Even in the EUGridPMA distribution.
  - You cannot simply disable them for that.

- **Even if you read a MUST NOT in the RFC, you will encounter that very case sometime and will have to handle it.**

Lessons learned 2

- **There are no “optional” parts in the standard.**
  - They will be used somewhere.
  - And if you do not support them, you will fail.

- **The standard has “recently” changed in an backwards-incompatible way.**
  - Basically, certificates that did not qualify as self-signed CA certificates before, do qualify now.
  - Already encountered such a CA in the wild.
    - The available libraries do not handle them yet.

Lessons learned 3

- **Never assume that your libraries are reliable.**
  - They will break or change behaviour from version to version.
  - See past esperiences with OpenSSL and Bouncycastle.
- **Make sure to experiment with new versions of your libraries.**
- **You will find undocumented parts of the system you will have to use and support anyway.**
  - Best example: The .signing_policy files: their content is not defined anywhere.
    - And unfortunately, you cannot throw them away
- **Undocumented does not mean unimplemented. Study existing implementatios.**

Lessons learned 4

- **Okay, you have this very secure AuthN library**
  - It verifies everything correctly.
  - It handles all the edge cases
  - It handles aberrant behaviour
  - So you're good.
- **Stop right there!**
  - Who told you to verify everything?
- **I'm not kidding.**
  - There are legitimate use cases when you may not want to verify anything.
- **Make sure you allow deactivation of specific parts.**

Conclusions

- **Writing correct code is not enough.**
- **Writing standard-conforming code is not enough.**
- **Writing documented code is not enough.**

- **This is a clear case where pragmatism is necessary:**
  - If it is used, support it, regardless of other considerations