PSNC – status presentation

Rafał Lichwała, Gerard Frankowski

SA3 All Hands Meeting – UCY, Cyprus, 7-8.05.09

www.eu-egee.org

e-infrastructure

Porting

Rafał Lichwała,

PSNC Supercomputing Department

What has been done

- **gLite packages porting to x86_64 Debian platform is our main task**
- **Porting gLite WN packages on Debian 4 x84_64 platform - done!**
  - torquemaui - Success rate: 100% (3/3)#
  - VDT - Success rate: 100% (14/14)#
  - deps - Success rate: 100% (19/19)#
  - Basic - Success rate: 100% (12/12)#
  - VOMS - Success rate: 100% (13/13)#
  - DM - Success rate: 100% (19/19)#
  - gfal - Success rate: 100% (25/25)#
  - WN-3.2.0 - Success rate: 100% (59/59)
- **Detailed reports available on: http://grid.ie/autobuild/**

Current work and future plans

- **Currently we are focusing on porting UI packages for Debian 4 (current success rate: 93% [114/122])**

- **We also plan to prepare porting WN for Debian 5 – first steps already done:**
  - a proper virtual machine with Debian 5 has been prepared and ETICS installed
  - package external dependencies for deb5 x86_64 have been prepared
  - a new platform debian5 x86_64 has been registered in ETICS
  - building first packages on Debian 5
    - some issues still in progress...
    - .deb packages temporary disabled (.tar only) - VDT bulit in 100%

Daily porting issues

- **building lcmaps**
  - problem with two flex libraries: libfl.a and libfl_pic.a
  - problem fixed by changing symbolic link to the library with -fpic flag
- **building amga-serwer**
  - cannot convert 'SQLINTEGER*' to 'SQLLEN*'
  - problem fixed - details reported in savannah
- **building WN-lastest-VDT (VDT1.10.1-3)**
  - "globus.location" problem in configuration
  - fixed - details reported in savannah
  - missing libregexp-java installed
- **building org.glite.lb.client**
  - problem with missing libexpat.la
  - fixed by preparing new expat with missing libraries

Savannah and GGUS reporting

- **Savannah bug items:**
  - https://savannah.cern.ch/bugs/?func=detailitem&item_id=42319
  - https://savannah.cern.ch/bugs/?func=detailitem&item_id=44502
  - https://savannah.cern.ch/bugs/?func=detailitem&item_id=46462
  - https://savannah.cern.ch/bugs/?48512

- **GGUS problems raised:**
  - https://gus.fzk.de/ws/ticket_info.php?ticket=47891

Security

(Hydra and SCAS source code analysis)

Gerard Frankowski,

PSNC Security Team

Hydra – introduction

- **Hydra**
  - encrypted file storage solution
  - encryption file key is split and distributed in Hydra Keystores
  - the key may be reconstructed from a subset of key pieces
- **The structure and the versions investigated**
  - the server
    - written in Java
    - version 1.3.4.1 from the CVS
    - manual source code review supported with findbugs and pmd
  - the client
    - written in C
    - version 3.1.2.1 from the CVS
    - manual source code review supported with cppcheck and RATS

Hydra – the service

- **General**
  - Quite a good work, but SQL Injections possible
  - Java protects from buffer overflows etc.
  - Some minor resource leaks
  - A number of non-security related remarks
- **Report written**
- **Todo**
  - penetration tests with access to the service
    - will additionally verify several issues pointed in the report
    - a „fresh” person takes care of the pentests

Hydra – the service

- **SQL Injection**
  - The methods (e.g. MySQLSchemaHelper.java class) do not check the values of parameters passed to the prepareStatement method of a connection object
  - Even if the external environment assures that the values are ok, (which will be checked during the pentests), this class may be reused...
- **Not good**

```
// Initiate the table creation sql string
String sqlNewTable = "CREATE TABLE " + schemaName + "
(entry_id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, ";
```

- **Better**

```
String sqlNewTable = "CREATE TABLE " +
  validateSQL(schemaName) + "
(entry_id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, ";
```

Hydra – the service

- **Selected recommendations**
  - Please filter internally every parameter that builds an SQL query
  - Please remember to treat every single external source of data as potentially malicious, including e.g. the local database output
  - Remove any excess functionality and not used variables
  - Every output path from a method should assure that each allocated resource is released before the control is returned to the calling method
  - Please review the way of using exceptions. Avoid the situations where a method unnecessarily catches RuntimeExceptions

Hydra – the client

- **General**
  - The analyzed code does not contain direct big security holes...
  - ...but uses some practices that may easily cause them!
    - using of unrecommended C functions like strcpy()
    - sizes of local buffers defined locally as numbers (but reasonably)
  - Too little data filtering of the input parameters of cmd line tools
    - library calls could not be checked
  - A single case of not verifying malloc() return value
  - Race conditions
- **Report written**
- **Todo**
  - Take a look at the security of the key pieces in memory

Hydra – the client

- **An example of potentially insufficient data filtlering**
  - eds.decrypt.c, function main()

```
87: id = argv[optind++];
```

- **Where does *id* go then?**
  - glite_eds_decrypt_init(**id**, &error) [eds-simple.c: 731]
  - glite_eds_init(**id**, &key, &iv, &type, error); [eds-simple.c: 552]
  - glite_eds_get_metadata(**id**, &hex_key, &hex_iv, &cipher_name, &keyinfo, error) [eds-simple.c: 448]
  - glite_eds_get_metadata_single(endpoints[i], **id**, &data, &err) [definition: eds-simple.c: 238]
  - glite_metadata_getAttributes(ctx, **id**, attrs_count, attrs, &result_cnt) [metadata-simple-api.c:711]
  - soap_strdup(ctx->soap, **item**) [?], or
  - soap_call_metadata__getAttributes(ctx->soap, ctx->endpoint, NULL, **sitem**, &req, &resp) [?]

Hydra – the client

- **An example of a potential race condition**
  - metadata-simple-api.c, function is_ctx_ok()

```
205: if (getenv(GLITE_METADATA_SD_ENV))
206: ret = _glite_catalog_init_endpoint(ctx,
  metadata_namespaces,
207:          getenv(GLITE_METADATA_SD_ENV));
```

- **Issues**
  - What happens if between the getenv() calls the processor is switched to another task and before it is returned, someone changes or deletes the GLITE_METADATA_SD_ENV variable?
  - There is no filtering of all, environment variables are potentially malicious
  - Is it optimal to call getenv() twice? Will the compiler optimize the source? Always?

Hydra – the client

- **Selected recommendations**
  - Please implement filtering the input parameters, addressing literally every parameter that is not a pure option (e.g. filenames) – please check its size and verify also the format
  - Please consider migrating from C functions considered as dangerous (like strcpy or sprintf to their “n” versions)
  - Please consider applying *defensive programming* rules, e.g.:
    - Verify if every pointer passed to every function is not NULL (unless it is acceptable)
    - Verify the values of all parameters that the users have control over (i.e. environment variables and contents of the files)
    - Do not use numeric values for the definition of local buffers sizes, but rather defined constants like MAX_BUF_LEN or MAX_ERROR_LEN, that are consistently applied throughout the code.

SCAS – introduction

- **SCAS**
  - Site Central Authorization Service
  - a Web Service that allows client programs to query for an authorization decision based upon user credentials
- **The structure and the packages investigated**
  - the service
    - written in C
    - org.glite.security.scas from the CVS
    - manual source code review supported with cppcheck, RATS and flawfinder
  - the client & plugins
    - written in C
    - org.glite.security.lcmaps-plugins-scas-client from the CVS
    - manual source code review supported with cppcheck and RATS

SCAS – test methodology

- **A test plan**
  - 2 persons
  - person A investigates the service thoroughly (manual code review and scanners) while the person B analyzes the client & plugins in the same manner
  - then they switch to the other module and make only a short review
- **Two approaches to tests with tools:**
  - scan-first:
    - scan the code, look at the potential issues and eliminate false positives
    - read the rest of the code
  - read-first:
    - read the whole code and select potential threats
    - scan the whole code and look if you were able to notice all threats

SCAS – current status

- **Source code tests**
  - client and plugins: identified scanner output, the code is being reviewed if they are not false positives
  - service: the code is being read
  - a brief analysis on security of XACML was prepared
- **Report: in the background, currently a sketch**
- **Necessary effort: ca. 80 hours**
- **In June we can start another module**
  - June (very busy): learning the module
  - July: tests

Questions?

Thank you!

Rafał Lichwała

syriusz@man.poznan.pl

Gerard Frankowski

gerard@man.poznan.pl