PROCEEDINGS OF SCIENCE

The EGI Software Vulnerability Group and EMI

L. A. Cornwall[1]
STFC, The Rutherford Appleton Laboratory
Harwell Oxford, Didcot, OX11 OQX, United Kingdom
E-mail: Linda.Cornwall@stfc.ac.uk

E. Heymann
Universitat Autonoma de Barcelona
Computer Architecture and Operating System Department, Campus de Bellaterra, Bellaterra 08193 (Barcelona), Spain
E-mail: Elisa.Heymann@uab.cat

This provides an overview of the activities of the European Grid Infrastructure (EGI) Software Vulnerability Group (SVG) and progress made in addressing vulnerabilities in collaboration with the European Middleware Initiative (EMI). This includes a summary of the formally agreed process for handling Software Vulnerabilities reported to the EGI SVG. It also describes the pro-active searching for vulnerabilities via 'Vulnerability Assessment' of software packages. Steps taken to prevent new vulnerabilities entering the infrastructure are described. The emphasis is on Grid Middleware as vulnerabilities in this software are generally not handled elsewhere. The collaboration with the Grid middleware software providers, in particular EMI, is described.

EGI Community Forum 2012 / EMI Second Technical Conference,
Munich, Germany
26-30 March, 2012

[1] Speaker

1. Introduction

The purpose of the EGI Software Vulnerability Group (SVG) is to eliminate existing vulnerabilities from the deployed infrastructure, primarily from Grid Middleware, prevent the introduction of new ones and prevent Security incidents. This describes the various strategies for keeping the infrastructure as free from software vulnerabilities as possible.

1.1 Definition of a Vulnerability

A vulnerability may be defined as a weakness allowing a principal (such as a user) to gain access to or influence a system beyond their intended rights. Gary McGraw's definition is a vulnerability is a defect or weakness in system security procedures, design, implementation, or internal controls that can be exercised and result in a security breach or violation of security policy. A vulnerability may result in an unauthorized user gaining access to a system, or an authorized user gaining unintended privileges (such as root or admin access), damaging a system, gaining unintended access to, deleting or modifying data or information or impersonating another user. In a Grid system where a large number of resources are distributed across many countries, the consequence of a malicious user exploiting a vulnerability can be devastating.

Generally, system administrators are trusted and any ability to carry out any actions on the site or sites which they administrate (such as view or delete information), are not considered to be vulnerabilities. The exception is access to bulk encrypted data and the encryption keys which allow such data to be decrypted. System administrators are not prevented from having access to data decrypted on the machines for which they are responsible, for example during processing. In addition, information which may be useful to an attacker (such as lists of port numbers) is not considered to be a vulnerability, nor are concerns such as 'These instructions are not clear, this could be installed in an insecure manner'.

Vulnerabilities due to weaknesses in software are the focus of this paper.

1.2 Strategies for reducing Software Vulnerabilities in the EGI environment

There are 3 main strategies for reducing software vulnerabilities in the EGI infrastructure. The first (and largest activity of the EGI SVG) is the handling of vulnerabilities reported, including the investigation, assessment and timely resolution of vulnerabilities found to be valid in the EGI environment. The second is 'Vulnerability Assessment', which is the pro-active investigation of software widely used in the EGI environment to see if there are any vulnerabilities present. The third is vulnerability prevention, which consists of developer education and the consideration of whether new middleware and functionality should be allowed onto the EGI infrastructure.

2. The Handling of Software Vulnerabilities reported

This is the largest activity of the EGI software vulnerability group, where issues reported are handled and resolved in a timely manner.

2.1 Background

In 2005 various people were discussing potential vulnerabilities in Middleware used in the Grid Infrastructure on open mailing lists, and comments such as 'someone should list these and we should get them fixed'. Hence the EGEE Grid Security Vulnerability Group (GSVG) began and a process for handling vulnerabilities was agreed and approved by the management at the time. Between 2005 and 2010 (prior to the start of EGI) 193 potential vulnerabilities were reported and handled by the GSVG resulting in approximately 100 bug fixes concerning these vulnerabilities. At the start of EGI the EGI Software Vulnerability Group (SVG) was formed.

2.2 Scope of EGI SVG issue handling

The main focus is to handle vulnerabilities found in the EGI Unified Middleware Distribution (UMD) repository [1], where EGI has a Service Level Agreement with the providers of this software stipulating response times and agreeing that issues should be handled according to the EGI SVG issue handling process. The largest provider of software in the EGI UMD is the European Middleware Initiative (EMI) [2]. In general, Grid middleware does not have any other group or activity handling vulnerabilities. The EGI SVG also handles vulnerabilities in other software deployed on the EGI infrastructure, and this is carried out jointly with EGI Computer Security Incident Response Team (CSIRT). CSIRT handles incidents and operational security to provide a consistent risk assessment of all vulnerabilities in software deployed on the EGI infrastructure. Software vulnerabilities in software other than that distributed as part of the EGI UMD is fixed by their providers and EGI has no service level agreement with such providers.

If any vulnerability is reported to the EGI SVG the EGI SVG will look at it and take any action necessary. For example, if it is not relevant to EGI yet it is not certain that the provider of the software is aware of the problem then EGI SVG will forward the information to the software provider. Table 1 summarises which actions are carried out for reports of possible vulnerabilities in various types of software.

2.3 Summary of the EGI SVG Issue handling process

The full details of the EGI issue handling process is defined in detail in The EGI Software Vulnerability Issue Handling Procedure [3] and is carried out by the EGI SVG Risk Assessment Team (RAT)[4] who are have full access details of vulnerabilities reported.

- Anyone may report an issue by e-mail to report-vulnerability@egi.eu
- The issue is investigated by a collaboration between the reporter, RAT, and the developers of the software.
- If the issue is found to be valid, a risk assessment is carried out, whereby the issue is placed in one of four risk categories which are Critical, High, Moderate, or Low.
- A 'Target Date' for resolution is set according to a fixed value for each category. Critical = 3 working days, High = 6 weeks, Moderate = 4 months, Low = 1 year.
- An advisory is issued when the problem is resolved, and a new version of the software released, or on the target date.

Software Source	S/W provider aware/announced vulnerability	S/W provider not clearly aware of vulnerability	Risk Assessment	Other comment
EGI UMD – e.g. EMI/IGE software for which EGI has SLA	Problem fully handled according to process in this document by SVG		SVG	
Operational Tools developed by the EGI InSPIRE project	Problem fully handled according to the process in this document by SVG, except distribution of tools not in UMD		SVG	
Linux Operating system software on which the EGI infrastructure is based	CSIRT sub-group /SVG investigates relevance to EGI	Inform software provider	SVG/CSIRT subgroup jointly	Usually CSIRT member will contact provider if necessary
EPEL software (Extra Packages for Linux Enterprise)	CSIRT sub-group /SVG investigates relevance to EGI	Inform software provider	SVG/CSIRT subgroup jointly	SVG or CSIRT member will contact provider depending on knowledge
Other Software widely installed on the EGI Infrastructure	CSIRT sub-group /SVG investigates relevance to EGI	Inform software provider	SVG/CSIRT subgroup jointly	SVG or CSIRT member will contact provider depending on knowledge
Software not installed on the EGI infrastructure	Do nothing	Inform software provider	None	Only action is to forward information.

Table 1. Guide to how vulnerabilities in various types of software is handled.

The Risk Assessment is carried out by the RAT because mitigating or aggravating factors may exist in the EGI infrastructure, and these need to be taken into account. The procedure states that the RAT will vote on the Risk category, but in most cases a consensus is reached.
The EGI aims to complete this within 4 working days, or 1 working day for critical vulnerabilities.
The software providers and the EGI Deployed Middleware Support Unit (DSMU) then need to ensure that the issue is fixed and available in the EGI UMD for installation by the various Resource Providers or sites by the target date.
If a vulnerability is assessed as Critical, then SVG and CSIRT jointly decide on what action to take. Possibilities include setting a longer target date or carrying out operational mitigation. Usually it is apparent if a vulnerability is likely to be considered ‘High’ or ‘Critical’ risk as soon as it is reported and members of the RAT and the developers focus their attention on it.

2.4 Advisories

Advisories are released when the software vulnerability is fixed, or on the Target date, whichever is the sooner. This is known as 'responsible disclosure' which allows software providers to fix the issue but does not keep the information secret indefinitely. Advisories state what the problem is, including the effect of an exploit, but should not give enough information to allow the vulnerability to be easily exploited unless it is essential for the resolution of a vulnerability in the infrastructure.

Advisories are normally placed on the EGI SVG wiki [5] as well as being set to National Grid Infrastructure security contacts, Site security contacts, and CSIRT by e-mail. The original reporter also receives a copy of the advisory.

2.5 Progress during EGI

Since the start of EGI (in May 2010) 52 potential vulnerabilities have been reported. This has resulted in 14 advisories released publically by SVG as well as many of the alerts released by CSIRT for more non-middleware issues [6]. At the time of writing 12 Middleware issues are awaiting the release of a software fix (all assessed as 'Low' risk.)

2.6 Problems encountered and improvements planned.

Sometimes the co-ordination between the various parties has not quite worked to plan, and confusion has happened when trying to ensure the advisory is issued at the same time as the patch is available in the UMD. Co-ordination has improved with time, but may be improved further by making available a flow chart of who does what and when.

Sometimes information on vulnerabilities has been leaked before the fixed version is available in the EGI UMD. This may be due, for example, to world readable open source software management tools making it possible for a person who is following changes to gain information on where vulnerabilities may be found. Software providers have been asked to avoid making changes publicly detectable for 'High' or 'Critical' risk vulnerabilities, but this problem is not yet fully resolved.

At times it has been difficult to track which version of the middleware component contains a vulnerability and which version does not. This has been resolved by tracking the fixed version of the middleware component, the first EMI version which contains the fix, and the first UMD version which provides that fix.

3. First Principles Vulnerability Assessment

3.1 Description

Researchers from the University of Wisconsin Madison and the Universitat Autonoma de Barcelone developed a methodology for manual (analyst centric) Vulnerability Assessments, called First Principles Vulnerability Assessment (FPVA) [7]. FPVA allows the in depth valuation of the security of a system. While FPVA is a labour intensive approach to Vulnerability assessment, it has been shown to be effective in several real systems, such as Condor, Wireshark, and VOMS, finding several serious vulnerabilities.

FPVA consists of 5 stages, architectural, resource, privilege, and component analysis followed by result dissemination.

Architectural Analysis: This step identifies the major structural components of the system, including modules, threads, processes and hosts. For each of these components, FPVA identifies the way they interact, both with each other and with users. Interactions are particularly important as they provide a basis for understanding how the trust is delegated through the system. The artefact produced at this stage is a document that diagrams the structure of the system and the interactions amongst the different components, and with the end users.

Resource Analysis: This second step identifies the key resources accessed by each component, and the operations supported on those resources. Resources include hosts, files, databases, logs, and devices. These resources are often the target of an exploit. For each resource, FPVA describes its value as an end target or as an intermediate target. The artefact produced at this stage is an annotation of the architectural diagrams with resources.

Privilege Analysis: This third step identifies the trust assumptions about each component, answering such questions as how are they protected and who can access them? The privilege level controls the extent of access for each component and, in the case of exploitation, the extent of damage that it can accomplish directly. A complex but crucial part of trust and privilege analysis is evaluating trust delegation. By combining the information from the first two steps, we determine what operations a component will execute on behalf of another component. The artefact produced at this stage is a further labelling of the basic diagrams with trust levels and labelling of interactions with delegation information.

Component Analysis: This forth step examines each component in depth. For large systems, a line-by-line manual examination of the code is unworkable. In this step, FPVA is guided by information obtained in the first three steps, helping to prioritize the work so that the code relating to high value assets is evaluated first. The work in this step can be accelerated by automated scanning tools. While these tools can provide valuable information, they are subject to false positives, and even when they indicate real flaws, they often cannot tell whether the flaw is exploitable and, even if it is exploitable, the tools cannot tell if it will allow serious damage. The artefacts produced by this step are vulnerability reports, which are provided to the software developers.

Result dissemination: Once vulnerabilities are reported, the obvious next step is for the developers to fix them. Vulnerabilities in EMI software are handled in the same way as other vulnerabilities reported to EGI SVG, a risk assessment is carried out and target date for

resolution set. Information on the full details of the vulnerability is not usually released publicly until the vulnerability has been fixed and sites have had plenty of time to ensure that the vulnerability is no longer present in the deployed infrastructure.

3.2 Application to EMI software

Various pieces of software from EMI have been assessed using the FPVA methodology. The Virtual Organisation Membership Service (VOMS) Admin 2.0.18 (which allows a Virtual Organisation Administrator to give appropriate membership and roles to users) has been assessed, and the vulnerabilities found fixed about 1 year ago. Argus 1.2 [8] (which facilitates authorization decisions in a distributed environment according to user's rights) has been assessed, and no vulnerabilities were found. gLexec 0.8 (which acts as a lightweight gatekeeper, taking the local site policy into account to authenticate and authorize a user according to their grid credentials, and allows identity switching) [9] was assessed, and some 'Low' risk vulnerabilities were found. These have been resolved in the latest version which is expected to be released at the end of April 2012. VOMS Core 2.0.2 has been assessed and one 'Low' risk vulnerability has been found.

At present members of the Universitat Autonoma de Barcelona are working on assessing the 'Workload Management System' WMS. Plans are also in place to assess the Computing Resource Execution And Management (CREAM).

If time and manpower permits, plans are also in place to assess two UNICORE components: The Target System Interface (TSI) which provides and interface between UNICORE and the individual resource management/batch system and operating system of the Grid resources and Gateway which is an authenticating web proxy service for web service requests (SOAP messages) and normal HTTP traffic of the UNICORE Grid Middleware.

4.Vulnerability Prevention

4.1 Developer Training

One way to help prevent new vulnerabilities from occurring is to train developers to write secure code. In particular not to trust user input, including that from a client supplied by the developers if it is possible for a user to modify that client. File permissions should also be checked, for example if a file has world write permission which is executable as there is the potential for a root exploit if a user can execute code on that system. Programmers should also be aware of buffer overflows; different kind of injection attack, including command injection and SQL injection; attacks through strings or integers; directory traversal; and web attacks such as Cross-site scripting amongst many others.

Developer training has largely been carried out by EMI, for example at the EGI Technical Forum in Lyon in September 2011[10] . EMI offers tutorials on Security Risks, Vulnerability Assessment and Secure Programming for users, software developers, system administrators and managers.

Currently the above mentioned tutorial is being expanded to a University course consisting of 80 hours of lectures and exercises in C, C++, Java and scripting languages.

4.2 Certification testing

It is possible to test for some vulnerabilities at Certification of a software package. As part of the EMI certification the software is checked for world writeable files, and since this has been in place no vulnerabilities have been found due to the presence of world writeable files. Further tests may be considered if they are identified in the future.

4.3 Assessing new software for vulnerabilities

At times EGI forms partnerships with new software providers. Even though a lot of effort has gone into reducing the vulnerabilities in software provided by existing providers the addition of new providers opens the possibility of new software vulnerabilities in the EGI infrastructure. Ideally such software would be assessed in detail for vulnerabilities, for example using the FPVA techniques referred to above. However due to lack of manpower this is not possible. The possibility of checklists for important criteria for the acceptance of new software has been suggested but no progress has been made so far.

4.4 Assessing new requirements

New software requirements may in principle introduce a vulnerability. For example, if a requirement was accepted from a user group which provides information which is useful to them, it may also expose information which should not be exposed, and lead to a vulnerability. Consideration of the security implications of new requirements should be considered but no progress has been made so far.

5. Conclusions and future work

The EGI SVG in conjunction with EMI have worked together to reduce the number of vulnerabilities in the EGI infrastructure, and prior to that in the EGEE infrastructure.

Less new vulnerabilities which are ‘High’ risk or ‘Critical’ risk are being reported in the EMI middleware, which suggests that the vulnerability prevention measures are having affect.

Since EMI is coming to an end in 2013 plans need to be put in place to ensure that security support for deployed middleware components is available and procedures for handling vulnerabilities in this new situation defined.

New technology including Cloud technology and virtualization will inevitably introduce new security issues, which need to be investigated and addressed. One aspect is the software which facilitates the federation of cloud infrastructure for research communities, vulnerabilities of various types may be found in this software which will need to be addressed.

When new technology appears there is a tendency for security to be ignored or thought about later. Security (including software security) needs to be addressed and have an appropriate level of investment from the beginning.

FPVA has proven to be an effective methodology for finding vulnerabilities. Nevertheless it is a manual methodology, therefore is expensive and time consuming. At present research is being carried out to automate parts of it. In particular, techniques like self-propelled instrumentation [11] will allow us to automatically obtain the architectural diagram of a

distributed application. In addition tools are being worked on we are for guiding the analyst *where* in the code to search for *what* problems.

Overall, the 3 main activities for reducing software vulnerabilities in the EGI infrastructure (handling reported vulnerabilities, assessing code for vulnerabilities, and vulnerability prevention) are progressing well.

References

[1] The EGI UMD repository http://repository.egi.eu/

[2] EMI http://www.eu-emi.eu/

[3] The EGI Software Vulnerability Issue handling Procedure https://documents.egi.eu/secure/ShowDocument?docid=717

[4] The EGI Risk Assessment Team Members https://wiki.egi.eu/wiki/SVG:RAT_Members

[5] EGI SVG advisories https://wiki.egi.eu/wiki/SVG:Advisories

[6] EGI CSIRT Alerts https://wiki.egi.eu/wiki/EGI_CSIRT:Alerts

[7] James A. Kupsch, Barton P. Miller, Eduardo César, and Elisa Heymann, "First Principles Vulnerability Assessment", 2010 ACM Cloud Computing Security Workshop (CCSW), Chicago, IL, October 2010

[8] Argus http://www.switch.ch/grid/argus/index.html

[9] gLexec https://wiki.nikhef.nl/grid/GLExec

[10] EGI Technical Forum 2011 Training Tutorial on secure programming https://www.egi.eu/indico/contributionDisplay.py?sessionId=57&contribId=75&confId=452

[11] Alexandre V. Mirgorodskiy and Barton P. Miller, "Diagnosing Distributed Systems with Self-Propelled Instrumentation", *ACM/IFIP/USENIX 9th International Middleware*, Leuven, Belgium, December 2008