A Code Inspection Process for Security Reviews

Gabriele Garzoglio
(garzoglio@fnal.gov)

Computing Division, Fermilab, Batavia, IL

INTRODUCTION

In recent years, it has become more and more evident that software threat communities are taking an increasing interest in Grid infrastructures. To mitigate the security risk associated with the increased numbers of attacks, the Grid software development community needs to scale up effort to reduce software vulnerabilities. This can be achieved by introducing security review processes as a standard project management practice.

The Grid Facilities Department of the Fermilab Computing Division has developed a code inspection process tailored to reviewing security properties of software. The goal of the process is to identify technical risks associated with an application and their impact.

This is achieved by focusing on the business needs of the application (what it does and protects), on understanding threats and exploit communities (what an exploiter gains), and on uncovering potential vulnerabilities (what defects can be exploited). The desired outcome of the process is an improvement of the quality of the software artifact and an enhanced understanding of possible mitigation strategies for residual risks.

This poster describes the inspection process and lessons learned on applying it to Grid middleware.

SCOPE.
GOALS.
LESSONS LEARNED.

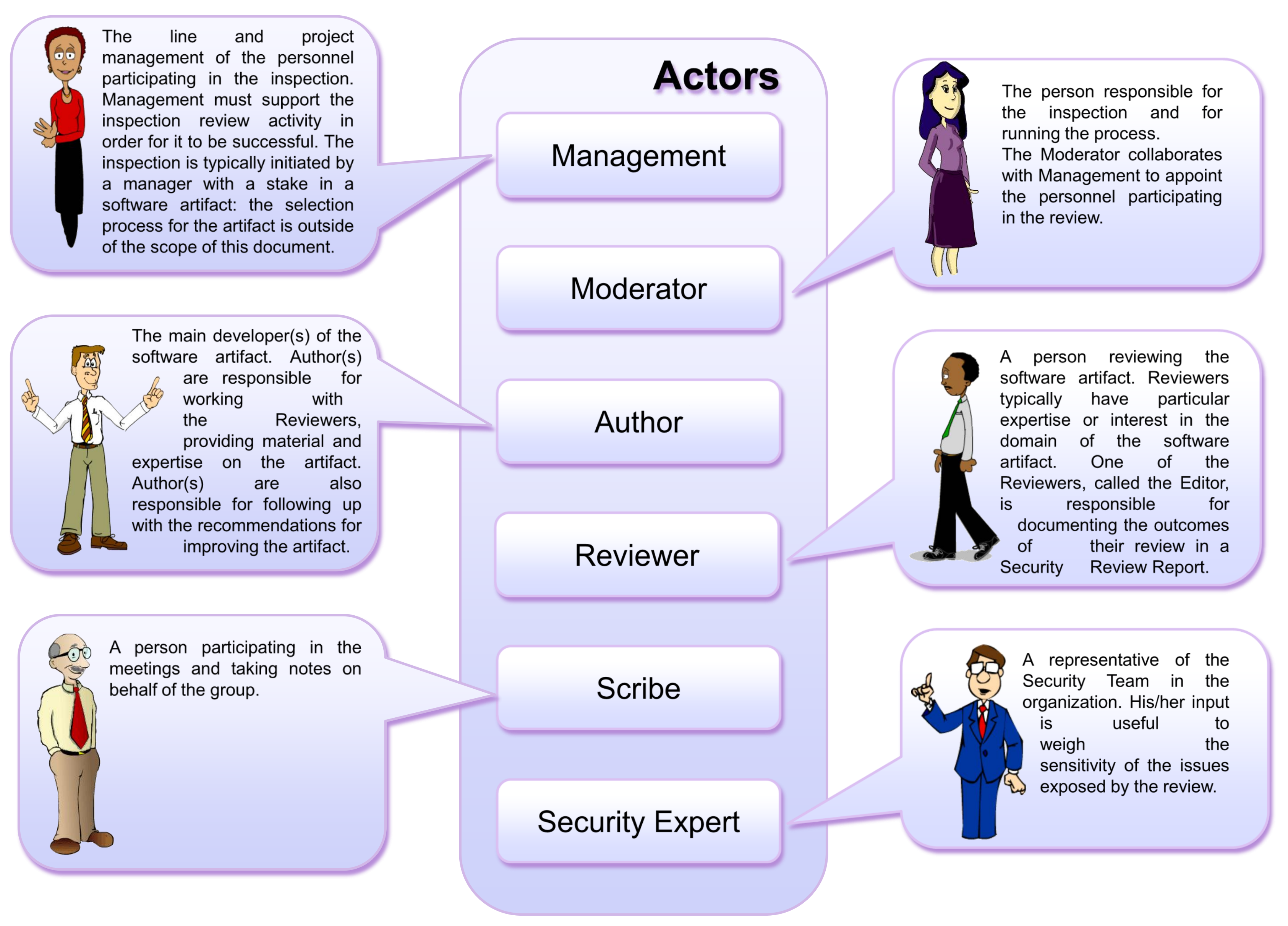
This work defines a process to assess the security issues of a software artifact. The goals of the process are to identify the technical risks associated with the application and the impact of these technical risks. The focus is on studying the security issues within the code itself, rather than with the operations of the software.

To achieve these goals, the reviewers should study the software artifact with the following in mind:
- What are the business context and risk? What does the software do and protect?
- What does an exploiter gain? What is the threat and exploit community?
- What defects can be exploited? What are the potential vulnerabilities?
- Loosely defining risks as *vulnerabilities x threats*, what are the risks?

These are a few of the lessons learned from our experiences.
- Not all reviews require the same level of formalism. For small software components, it may be sufficient to focus on the code review only.
- The running of application tests is the most useful technique to show software problems to Authors

References

[1] The Fagan inspection process at Fermilab: http://ods.fnal.gov/ods/www/process/faganInsp.html
[2] Hoglund, G. and McGraw, G. “Exploiting Software: How to break the code”, Addison-Wesley
[3] McGraw, G. “Software Security: Building Security in”, Addison-Wesley
[4] Viega, J. and McGraw, G. “Building Secure Software”, Addison-Wesley
[5] Code review tools: http://www.softpanorama.org/SE/code_reviews_and_inspections.shtml
[8] OWASP testing portal: http://www.owasp.org/index.php/OWASP_Testing_Guide_v2_Table_of_Contents
[9] Resources on Fuzzing tools: http://www.dragoslungu.com/2007/05/12/my-favorite-10-web-application-security-fuzzing-tools/



Management: The line and project management of the personnel participating in the inspection. Management must support the inspection review activity in order for it to be successful. The inspection is typically initiated by a manager with a stake in a software artifact: the selection process for the artifact is outside of the scope of this document.

Moderator: The person responsible for the inspection and for running the process. The Moderator collaborates with Management to appoint the personnel participating in the review.

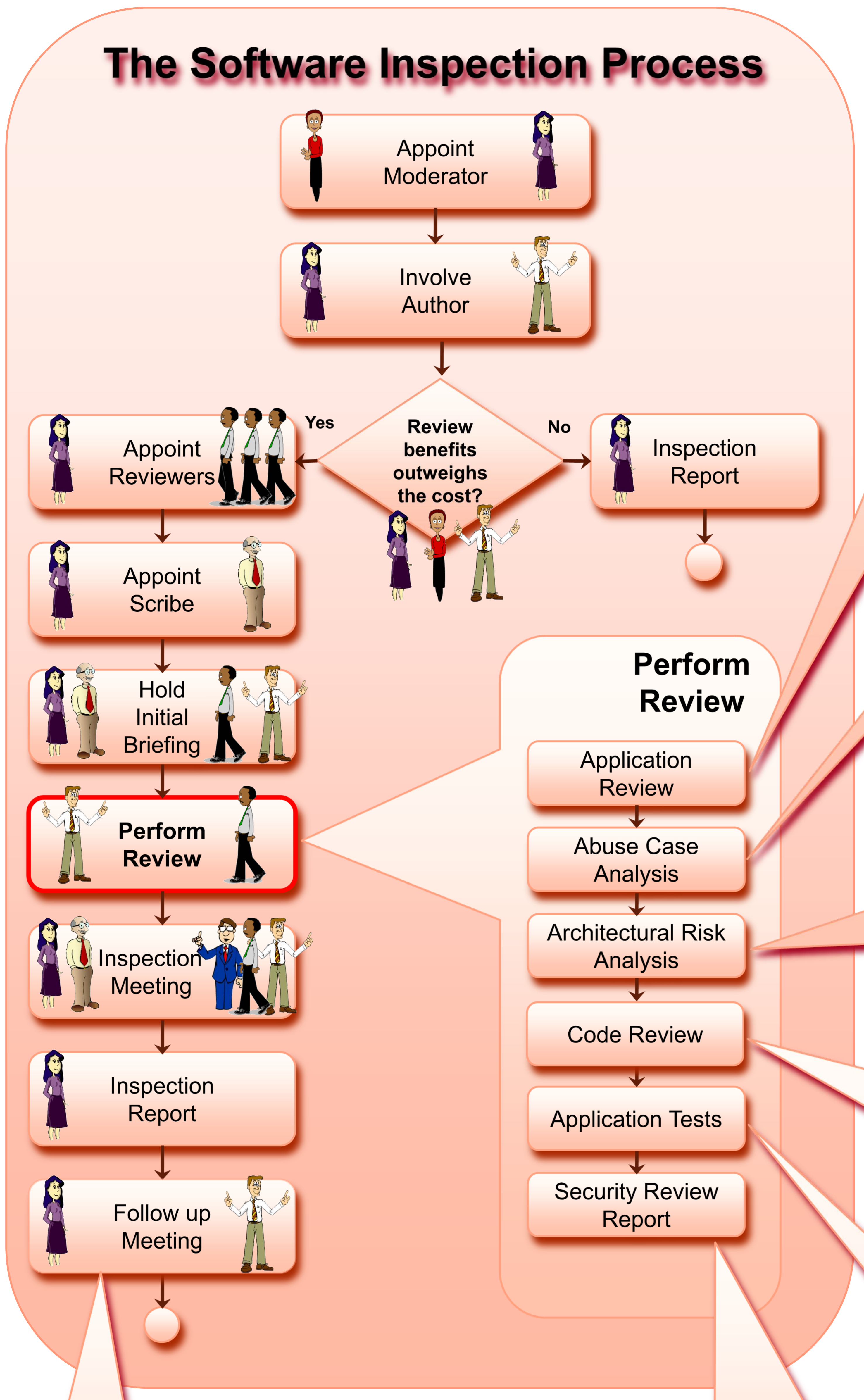
Author: The main developer(s) of the software artifact. Author(s) are responsible for working with the Reviewers, providing material and expertise on the artifact. Author(s) are also responsible for following up with the recommendations for improving the artifact.

Reviewer: A person reviewing the software artifact. Reviewers typically have particular expertise or interest in the domain of the software artifact. One of the Reviewers, called the Editor, is responsible for documenting the outcomes of their review in a Security Review Report.

Scribe: A person participating in the meetings and taking notes on behalf of the group.

Security Expert: A representative of the Security Team in the organization. His/her input is useful to weigh the sensitivity of the issues exposed by the review.

The Software Inspection Process



Application Review

This step is useful to familiarize the Reviewers with the software artifact. The principal investigation mechanisms consist of documentation review and Author interview.
Some elements of interest are:
General Functionality, Application Environment, Use Cases, Specific Domain Features, Architecture, Relevant Code Portions, Project management practices, Risk Analysis / Security Requirements / Security-related Operations (if available).

Abuse Case Analysis

Abuse case analysis is discussed in [2] and [3]. "Abuse cases" are malicious ways of misusing the software. Studying abuse cases helps prepare for abnormal or exceptional application behavior. Reviewers can loosely follow the process:

1. **Identify applicable attack patterns** [2]. Attack patterns include cross-site scripting attacks, SQL injections, buffer overflows, etc.
2. **Build an attack model** using the applicable attack patterns. [2]
3. **Determine misuses** of the software **and abuse cases**

Remember to talk to the Authors, as they might be already aware of potential abuses of the software.

Architectural Risk Analysis

Architectural risk analysis is discussed in [3] and [4]. Reviewers can loosely follow the following process.

1. **Build a one page architectural overview** of the software system, if not available.
2. **Analyze the architecture**, focusing on the following properties.
   - **Attack resistance**: how resilient is the application in the face of an attack?
   - **Presence of ambiguity**: are there ambiguities in the architecture, in terms of functionality, responsibilities, etc.?
   - **Presence of weaknesses**: are there weaknesses built into the architecture, such as single points of failures, communication inefficiencies, etc.?

3.Identify and rank architectural risks. Focus on what the application protects. Understand what would an exploiter gain with a successful attack (threat) and what defects can be exploited (vulnerability). Risks are often loosely defined as the product of threats x vulnerabilities.
4.Define possible mitigation strategies for each risk.

Code Review

See [3] and [4] for recommendations. In general, it is best to use automated tools to guide the reviewers toward portions of the code more likely to have problems. Some of these tools, have advanced analysis features but are expensive to use. There are various resources on the web that discuss free tools [5].

If the code review is done by reading the code, the Reviewer should especially look for the following properties: Input validation and representation, API abuse, Security-related features, Time and state, Error handling, Code quality, Encapsulation, Environment.

Potential concerns are often one of the following:
1. An area of the artifact that the Author is uncertain about
2. An area that in the past has shown to be problematic
3. A special quality that the deliverable must contain (i.e. be memory efficient, incorporate all previous requirements etc.)

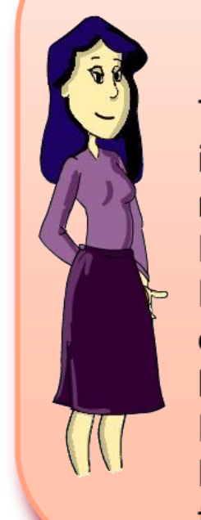
Application tests

There are many types of security testing techniques [3]. The following are a couple of pointers:
- OWASP [8]: for web-based application, offers step by step guides to uncover security vulnerabilities, using similar attack patterns as in [2].
- Tools for testing the response of software to a large set of input parameters are called "fuzzing" tools [9].

Write Security Review Report

The Security Review report is written by the Editor Reviewer and should include:
1. The findings of the review for each of the steps.
2. A discussion on: what the application does and protects, the threats and exploit community, potential vulnerabilities, and risks.
3. The impact of each risk, by linking risks with the business needs of the application.
4. A mitigation analysis for the highest impact risks.

Deliverables of the Inspection



The deliverable of the process is an Inspection Report. This report consists of the Security Review Report from the Reviewers and documentation of relevant discussions between Authors and Reviewers, as compiled by the Moderator using the notes from the Scribe.

CONCLUSIONS

We have developed a software inspection process for security reviews. It defines personnel roles, recommended meetings, and six steps to review the security properties of the investigated software.

This security review process was extended from software inspection guidelines that have been successfully used for years at Fermilab. The process has been successfully used on production software at Fermilab and is continuously being refined.