Master / Bachelor

Security through Code Instrumentation / Sicherheit durch Codeinstrumentierung

This thesis can also be supervised and written in English. For the English version of the announcement see below.

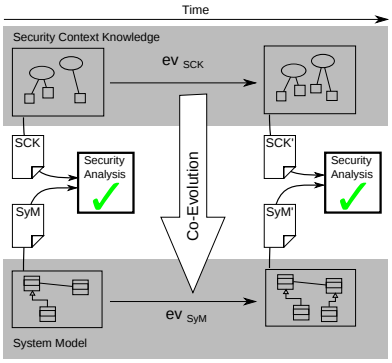

Abbildung 1: Verhältnis von Evolution zu Co-Evolution

Motivation

Model-driven software engineering especially is used for compliance to requirements of software systems. Requirements are mostly rather abstract and can come from various sources (e.g. laws, domain-specific regulations, company policies).

Nowadays, many systems are used over a long timespan (i.e. long-living systems). Many of these are used to provide critical infrastructure, so that livelong compliance to security requirements is of vital significance.

The project *SecVolution* captures and manages knowledge of a specific project and general security knowledge (Security Context Knowledge (SCK)) as well as essential security requirements (ESR) and to make them part of a software model, thus supporting compliance to them during development and even run-time.

If the knowledge evolves, it needs to be checked if the system model still is compliant to the evolved knowledge (s. Abb. 1).

During run-time, it is necessary to ensure that security assumptions made during design phase hold during the system execution. This can be done using code instrumentation.

—

Deutsche Version:

—

Modellgetriebene Softwareentwicklung eignet sich unter anderem dazu, die Einhaltung von Anforderungen an das System sicher zu stellen. Anforderungen (*requirements*) werden auf abstrakter Ebene im Entwicklungsprozess definiert oder sind bereits in Form von externen Quellen vorgegeben.

Software, die in kritischen Bereichen eingesetzt wird, wird oftmals über eine lange Laufzeit betrieben (*langlebiges System*). Langlebige Systeme sind insbesondere mit dem sich verändernden Wissen ihrer Umgebung konfrontiert, wie beispielsweise der Tatsache, dass bestimmte Verschlüsselungsalorithmen, die als sicher gelten, wenn ein System initial entwickelt wird, später unsicher werden können.

Das Projekt *SecVolution* beschäftigt sich damit, Wissen über ein Projekt und allgemeines Sicherheitswissen (Security Context Knowledge (SCK)) sowie generische Sicherheitsanforderungen (Essential Security Requirements (ESR)) zu sammeln und zu verwalten und als Annotationen an das Modell abzubilden, um die Einhaltung dieser Anforderungen bei der Modellierung / Implementierung sicherzustellen.

Ändert sich das Wissen (Evolution), muss überprüft werden ob das System-Modell die Sicherheitsanforderungen gegenüber dem veränderten Wissen noch erfüllt (s. Abb. 1).

Um die Einhaltung von Sicherheitsanforderungen zur Laufzeit eines Systems sicherzustellen, kann der Sourcecode auf Basis von Sicherheitsannotationen instrumentiert werden.

Aufgabenstellung/Ziele

Im Rahmen dieser Arbeit soll untersucht werden, unter welchen Voraussetzungen aus Modellannotationen Sourcecode instrumentiert werden kann, um Sicherheitsanforderungen zur Laufzeit sicherzustellen. Die praktische Umsetzung soll anhand von Fallbeispielen und der Implementierung / Erweiterung vorhandener Werkzeuge / Ansätze erfolgen. Mögliche Teilziele sind hierbei:

- Festlegen unterschiedlicher Security-Typen, die zur Laufzeit ggf. unterschiedlich überwacht werden müssen
- Untersuchung geeigneter Ansätze/Werkzeuge, um Modell und Implementierung zu verbinden (*trace links*)
- Untersuchung geeigneter Ansätze/Werkzeuge, um Code zur Laufzeit zu überwachen (weave-ins/aspektorientierte Programmierung, automatische Testgenerierung, etc.)
- Ggf. Erweiterung existierender Werkzeuge

Die Arbeit soll auf die bestehenden Vorarbeiten im Rahmen des Projekts SecVolution aufbauen.

Motivation

Model-driven software engineering especially is used for compliance to requirements of software systems. Requirements are mostly rather abstract and can come from various sources (e.g. laws, domain-specific regulations, company policies).

Nowadays, many systems are used over a long timespan (i.e. long-living systems). Many of these are used to provide critical infrastructure, so that livelong compliance to security requirements is of vital significance.

The project *SecVolution* captures and manages knowledge of a specific project and general security knowledge (Security Context Knowledge (SCK)) as well as essential security requirements (ESR) and to make them part of a software model, thus supporting compliance to them during development and even run-time.

If the knowledge evolves, it needs to be checked if the system model still is compliant to the evolved knowledge (s. Abb. 1).

During run-time, it is necessary to ensure that security assumptions made during design phase hold during the system execution. This can be done using code instrumentation.

Objectives

The goal of this thesis is to investigate, how security model annotations can be used to instrument sourcecode for checking security properties at run-time. The practial realization should be done using a case study and by implementing a new or extending existing tools / approaches. Conceivable sub-goals are:

- Define different security requirements types which need different kinds of run-time checking
- Evaluation of appropriate tools / approaches to link model and code (*trace links*)
- Evaluation of appropriate tools / approaches to weave in code for checking code at run-time (e.g. aspect oriented programming, automated test generation)
- Extension of existing tools if necessary

The thesis is in context of the SecVolution project and should be build by integrating with existing approaches and tools.

Organisatorisches

Kontakt:
Dipl.-Inf. Jens Bürger (buerger@uni-koblenz.de)