Trust4All

Patrik Floréen, Michael Przybilski, Taneli Vähäkangas,
Topi Musto

HIIT

Research Colloquium
16 March 2007

Some history

- Trust4All (2005) is a continuation project for Robocop (2001) and Space4U (2003).
- Previous projects produced a component based middleware software architecture for embedded devices. (component $\sim$ code that can be executed)
- The Roboarchitecture allows adding and changing components during runtime.
- Written in C for Linux.

Roboarch has some problems

- All memory is shared between components.
- If single component crashes, the whole system crashes.
- It is possible to add (third-party) components.

Components can contain bugs or simply be malicious, what can we do?

Enter Trust4All

Idea was to extend the Roboarch in a such way that the system's overall security would not be compromised in any case.

This leads to three questions:

1. How to detect potentially untrustworthy components?
2. What to do when we detect them?
3. If all fails, how to minimize the damage?

Containment

- The basic operating system protection for memory is multiple processes.
  - Roboarch was single process, lots of work implementing.
- For filesystem protection we can use the Linux *chroot()* system call.
- For network protection we can use FreeBSD *jail()*-like functionality.
- Protecting each kind of resource requires a different kind of container.

For monitoring the behaviour of the components we already have the Resource Management Framework from previous project.

Trust Evaluation Framework

- Each component can specify what kind of needs it has (security and resource related).
- The Trust Evaluation Framework reads the specifications and monitors the components.
- Utilising subjective logic calculates the *believed trustworthiness* of a component.
- Decides in which containers new components should be run.

Trust Evaluation Function

- Calculates the trustworthiness from a set of *quality attributes*.
- Trustor can weight different QA's in any way.
- Trustor chooses which QA's metric values are considered positive.
- Can use outside recommendations.
- Can use observations.

Decisions can made by comparing the output to some threshold.

Work in progress

- Finishing implementation.
- Standardization.
- Demonstrators.