Computer security in the Mobile Ambients calculus

René Rydhof Hansen and Jacob Grydholt Jensen
{rrh|grydholt}@daimi.aau.dk

With the advent, and ever increasing popularity, of large scale computer networking, e.g. the Internet and the World Wide Web, notions such as *mobile computing* and *mobile computation* (collectively referred to as *mobility*) have emerged. In this context, mobile computing refers to "ordinary" computing carried out on a mobile computer, e.g. notebooks, electronic calendars, handheld computers; while mobile computation refers to programs that are *themselves* mobile and move from location to location while carrying out computations, e.g. applets, agents. Clearly, allowing arbitrary agents to execute on a machine running critical software, e.g. the company database, introduces security risks. One way of dealing with the problem would be not to allow arbitrary agents to run on critical machines. A less Draconian approach to the problem would be to scan all agents before allowing them to execute on critical machines. The talk will focus on the use of control flow analyses to ensure certain security properties. This is in contrast to recent work in computer security, which employs type systems.

In order to study and ultimately understand mobility, the first step is to develop formalisms that are suitable for modelling this phenomenon. We will present such a formalism called "Mobile Ambients", designed by Cardelli and Gordon. What makes this calculus particularly relevant for our purpose is the notion of an *ambient*. Ambients provide a bounded place for computation to happen, thus making them obvious candidates for modelling computers and locations. Furthermore ambients are well suited for modelling mobile agents as they can be nested arbitrarily and moved in and out of other ambients.

Given an expression in the calculus one can ask the question: which ambients can enter which other ambients? A control flow analysis, to be presented in the talk, provides an answer to this question. This analysis is specified through a *flow logic*. The flow logic framework allows for concise specification of control flow analyses and facilitates rapid prototyping of analyses. Furthermore as our analysis is specified compositionally and using relatively simple primitives it is straightforward, for a given expression, to generate set constraints from the specification. These can then be solved using a stand-alone constraint solver to obtain the analysis result. A prototype constraint generator for the calculus has been implemented.

The analysis may be used to solve some of the computer security problems mentioned above. In order to do so we first identify the operational property of *allowed interaction*, i.e. which agents may execute alongside the critical software. To ensure that the dynamic property hold we define a static property, *respects*, in terms of an analysis result (an agent has this property if the analysis of the agent is on a certain form). It will be shown that agents that respect each other will only perform allowed interaction. This corresponds to using the analysis as a scanner.