Summary of In Class Discussion


Written by Michelle Pirtle

Time and effort to use the presented approach:
- Man-month, is this reasonable? Not including the modifications to the process yes
- Possibly work on with other projects too, making the tracking of time an issue

Is the approach low-cost, practical, provide high assurance?
- Chad – Depends on definition of practical
  - Do you have proper training (mathematical background)? In industry, probably not since background may not be there
- What is Required Background to use this approach?
  - Heitmeyer feels this is easier than UML (Stated by Dr. Cheng)
  - Dr. Cheng – There is no explanation of tables given with SCR
    - RSML may be a middle ground between SCR and UML
  - Jesse – Tabular notation is easier than UML, minus proof techniques,
    - States a combination of UML and tabular notation would be best solution so that diagrams can be included for industry
  - Sascha – Tables are not user friendly, diagrams are needed
  - Brian – Argued for SCR since he works with bright people
    - Feels that SCR is feasible to learn
    - There may be some need for UML to help address problems
  - Jack – What do practitioners feel?
    - SCR could be feasible since it represents the real world in a natural way,
    - SCR uses black-boxes which is practical
  - Michelle – Only one tool is fully automated, authors are working on making the other tools automated to various extents.

Difference between SCR and SCR*
- Jesse – To use SCR*, theorem proving background knowledge is needed for PVS
- Kyle – PVS states that one only need 6 months to understand PVS
- Dr. Cheng – PVS is harder to learn then PVS states

Downside of theorem proving (Dr. Cheng)
- Trying to establish what can be proved
- Lots of “Aha” steps
  - One must know which axioms, etc they want to apply in order to use them
- Method of determining theorem proving patterns
  - Coming up with theorem proving patterns may result in failure
  - Failure could mean the need to add more invariants
- Jesse – Mentioned the other side to practicality in relation to tools
  - Use of both model checker and theorem prover to achieve goal is beneficial
  - Model checker needs to be more automatic (also mentioned in paper)
PVS discussion
  o Kyle – PVS is automated once you give it a strategy
  o Chad – As an invariant prover, useful
  o Dr. Cheng – The invariant prover is weak
  o Sascha – The properties in the paper are disappointing for proving security
    ▪ Only 8 properties are even mentioned

Complaints on paper
  o Dr Cheng – No security properties are mentioned, all properties relate to safety
    ▪ Unsure on some of the security aspects of how this is related
  o Does SCR support security properties?
    ▪ Jesse – Definition of security properties is needed
      • Some of the tools that do support security properties are too specific since they define the intruder
      • Murph-Ø doesn’t explicitly represent the intruder for this reason
    ▪ Sascha – Murph-Ø converts more into safety properties

Hardware security is different from software security, but this method had been used on both hardware and software

• Kyle – **Combination** needed
  o Formalization of requirements needed
  o From formal requirements, deduce the security properties
  o Formal requirements to security properties will result in new properties, find and will find different errors.

• Ali –
  o **Is there a difference for success in verifying and validating properties when properties are a mixture of security properties with other properties, such as safety?**
  o **What kind of logic is sufficient for representing security properties?**

• Sascha – SCR is more general, and could allow for verifying and validating various properties