CS5232 Project (40%)

Please submit your project report (hard and soft copy) and PAT code (soft copy) to Fernando Dileepa (fdileepa@comp.nus.edu.sg) at software engineering lab (COM2-01-09).

Final Presentation: During class time on 11 April (10%).

Final Report Deadline: 22 April (30%)

Project report format:

1) Introduction: give an executive summary
2) Problem Description: a brief introduction of the problem
3) System Modeling: present your model with descriptions
4) Investigated Properties: examples like deadlock, invariant, LTL properties etc. Think of your own, do not be restricted by the references.
5) Experiments: present the verification results (number of states/transitions and execution time) for each property on different settings (e.g., different number of processes)
6) Discussions: discuss your approach: limitation (of your model or PAT modeling language), and any possible extensions.
7) Feedback and Suggestion for PAT: If you find any bugs in PAT, please report here. Extra mark will be given for each bug found. Write your experience of using PAT and provide feedback and suggestions in terms of GUI, usability, features, modeling language and so on.
8) Conclusions
9) References, may include
10) Jun Sun, Yang Liu, Jin Song Dong and Jun Pang. PAT: Towards Flexible Verification under Fairness. The 21th International Conference on Computer Aided Verification (CAV 2009), pages 709-714, Grenoble, France, June, 2009.

Own Topics:

The PAT modeling techniques are very general, expressive and applicable to many research and application fields. For example, by applying those techniques, you can add rigorous, precise and mathematical foundations to your Honours/Master/PhD thesis or the design of your industrial projects. In the project, you are encouraged to choose a topic which relates to your Honours/Master/PhD main research area or your current industry project (for course master students). If you think that your own research/work is not suitable for formal modeling techniques in PAT, then you can choose the default topics.

Default Topics:

Topic 1: Security Protocol Verification (including 3 projects)

Brief Description:

A security protocol (a.k.a. cryptographic protocol) is a short algorithm describing how participants should behave in order to achieve security goals, using cryptography (For details, see https://en.wikipedia.org/wiki/Cryptogrphic_protocol), for example, the Needham-Schroeder Authentication Protocol modelled in PAT examples. Design of security protocols is well-known to be error-prone. Formal verification, due to its preciseness, has shown its strength in proving or disproving correctness of security protocols. In order to formally verify a protocol, a verifier needs to provide a formal model describing the behavior of the participants as well as the attacker, and a property to verify, for example, whether a state is reachable in which the attacker learns sensitive information. Using modeling checking tool (for example PAT), the verifier is able to automatically verify whether a protocol satisfies a certain property.

We are building a security protocol module based on PAT. Currently, we have a prototype, including a library and a model refiner (written in C#). The library defines data types which can be used to model cryptographic primitives, such as encryption and decryption, as well as attacker knowledge reasoning. The model refiner automatically adds attacker behavior. With the security protocol module, a verifier only needs to define the protocol behavior, leaving defining cryptographic primitives and attacker behavior to the security protocol module.

Project 1: Security Protocol Module Refinement

This project aims to build a model of a security protocol using the security module prototype and verify the protocol using PAT, find where the security module need to be improved and improve it.

Detailed tasks:

1) Model one of the security protocols:
   - The oauth 1.0 single-sign-on protocol,

     Reference: http://oauth.net/core/1.0a/
   - The digital envelop protocol,

     Reference: http://www.cs.bham.ac.uk/~mdr/research/papers/pdf/09-ables-summer.pdf
2) Improve the security protocol module, for example:
   - To support more variations of protocols, or
   - To support more cryptographic primitives, or
   - Simplify the modeling language

Project 2: Security Protocol Module Integration

The aim of this project is to integrate the library and the model refiner together, and provide a user interface (ideally using the PAT interface) for the security module prototype.

Detailed tasks:

1) Integrate the library and the model refiner

   Currently the library and the model refiner are independent. However, the model refiner may need to use data e.g., the attacker knowledge, from the library. Thus, this task is to extend the model refiner to call methods in the library to fetch data.

2) Integrate the library and the model refiner into the PAT user interface

   Alternatively provide a user interface which allows users to type in a simplified model which uses the library, and click a button to verify the model. The verification step includes automatically calling the model refiner and calling PAT to verify the refined model.

3) Model an example protocol to validate the implementation

Project 3: Language Translation

Since the security protocol module is based on PAT, it uses CSP# as modelling language and uses model checking to verify a model. There are other ways, for example using formal logic to model a protocol and reason about attack knowledge.

This project aims to bridge CSP# and the horn clause (a popular formal logic to represent a protocol and the attacker).

Detailed tasks:

1) Learn the existing translation algorithm which translates a formal language (similar to CSP# with the library) to horn clause

   Reference: Section 7.1.2 in paper http://prosecco.gforge.inria.fr/personal/bblanche/publications/AbadiBlanchetJACM7037.pdf

2) Implement a prototype to translate a security protocol model using CSP# with the library to the horn clause representation

3) Use an example protocol model to validate the implementation

Topic 2: Model Checking in Android Applications (including 1 project)

Project Title: Model checking management of High Privileges in Android Applications

Project Description:

It becomes common for the mobile users to root their Android devices---it is reported that over 27.44% users do this. Accordingly, some applications also request root or adb privilege to implement some functionality that cannot be implemented through Android's official APIs. These privileges are extremely security sensitive and deserve a high-profile protection.

In this project, we will model how the applications manage the high-privilege module and analyze whether they have properly protected the granted high privileges. We will reverse engineer (with help of a TA) one of such applications (in particular, Helium

https://play.google.com/store/apps/details?id=com.koushikdutta.backup) and analyze how the high-privileged functions are invoked and managed in it. Afterwards, we will model the behavior of the app in CSP# and use PAT to check whether there exists privilege leakage.

Basic tasks (compulsory):
1) Learn how Helium, the app to analyze, uses and manages the adb privilege. Students may read paper http://www.comp.nus.edu.sg/~a0091939/Publications/Poaching.pdf, with focus on section IV.A; alternatively, students are encouraged to reverse-engineer by themselves the working process in managing the adb privilege.

2) Model the working process in CSP# and check it in PAT.

Advance tasks (optional, with extra points):
1) Automatically generate the CSP# models from Android apps
Students need to design an approach (with a TA) to extracting CSP# models from the apps' byte code.

2) Design and implement a secure working process of adb privilege management and verify it in PAT.

TAs:

Fernando Dileepa: fdileepa@comp.nus.edu.sg at software engineering lab (COM2-01-09).

Bai Guangdong baiguangdong@gmail.com at software engineering lab (COM2-01-09).

Dong Naipeng dongnaipeng@gmail.com at system & networking lab 7 (AS06-04-11)

Consultation for default topics:

Time: TBA

Venue: TBA

Guidelines to the Project

1. Summary

This is a guideline to the project. You should read the guideline carefully to make sure you understand all the requirements. In case you have any questions, please contact the TAs for help.

2. Project Description

The PAT modeling techniques are very general, expressive and applicable to many research and application fields. You can use PAT as a formal verification, plan making or more generally, problem solving tool. In the project, you are encouraged to propose a topic which relates to your Honors/Master/PhD main research background, your current industry project (for course master students) or simply a topic which you are interested in. Any computer science related problem, such as embedded system, network/security protocol, system design, web services etc. fits for the project. If you think that your own research/work is not suitable for formal modeling techniques PAT, you can choose the default projects.

3. Project Proposal Requirement

If you choose your own topic, you need to make the following aspects very clear:

a) What is the problem you are trying to model and solve and why is your problem interesting/important.

b) What interesting properties/goals you want to verify with PAT.

a) What is the scope of the problem, i.e., what you want to consider and what not to consider.

You are strongly encouraged to download PAT and try out the build-in examples to get a flavor of the PAT modeling languages. The build-in examples can also give you some inspirations on your own projects.

4. Build up your own project team

You can form up project teams (up to 5 person each team) or complete the project by your own. For those who had not informed your team members to TAs, please finalize your team members and inform the TAs before XXX.

5. Markings

We encourage you to propose your own projects. Extra bonus (up to 2 points) may be added to the final marks of your projects to reward your creativity. We consider all the members in a team to contribute equally to the project.

6. Delivery

There is no delivery requirements for the presentation.

7. Time and Venue for the presentation

   The presentation will be announced later. Each presentation is about 15mins and 5 mins Q&A. Print one set of slides (with names) and pass to the TAs before the presentation.

Guidelines to PAT code of final report

Each group submits a zipped file containing a top folder named after the Matric Number of the sender. The folder shall contain a source code folder, a report in word or pdf format, a .txt file which has name and Matric Number of the group members.

Example:

Zip: AXXXXXXX (Matric No.)

- Src
  - Src1.csp
  - Src2.csp
  - …
- report.pdf (or report.doc)
- readme.txt (group members)