Distributed Programming II
A.Y. 2019/20

Special Projects Proposals

This is the list of proposed special projects for DP2. If interested in applying, send an e-mail message to d1943@studenti.polito.it by 20/10/2019 with subject “DP2 Special Project Application”. The message must include the following information:

- The id of the project(s) you are interested in (in case you specify more than one project, please sort them in order of preference)
- Your CV including the transcript of your exams and the marks you got

If you apply for a project which is for more than one student you can indicate the other students you would like to work with. In this case they also have to apply. Otherwise, if you don’t indicate the other students you would like to work with, we will decide for you. The special projects will be assigned by 27/10/2019. Students who have been assigned a special project will have to confirm their decision of undertaking it and get the full documentation for their work from the teacher. When a special project is completed, the material produced for the project has to be submitted to the teacher and discussed with the teacher. Special projects must be completed by the deadline indicated in each project. Completing a special project exempts from the final test in the Lab (the exam terminates with the discussion of the special project). All special projects offer the possibility to continue the project work as a subsequent thesis on the same topic.

1) RESTful interface for TAMELESS (max 1 student)

TAMELESS is a tool developed in Prolog (specifically in XSB, a dialect of the Prolog developed by the Stony Brook University, http://xsb.sourceforge.net/index.html). TAMELESS receives as input the description of a system to be analyzed, which specifies: the system’s elements and threats, their relations (i.e., the relation among elements and between elements and threats), and the assumed security properties.

The aim of this project is to design and implement a RESTful web service based on TAMELESS. The assignment can be divided into the following parts:

a. Design data representations (described by means of an XML schema) respectively of the system elements, the expected security proprieties and the system threats.

b. Design a REST API that can receive the input using the previously defined data formats (XML or JSON) and that can perform the analysis.

A documentation of the data formats and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

Deadline: The end of the February exam session
2) An Optimization tool for Firewall anomaly resolution (max 1 students)

Firewalls have been widely used to protect not only small and local networks, but also large enterprise networks. The configuration of firewalls is mainly done manually by network administrators. Thus, it suffers from human errors. A typical example of error in firewall configurations is conflict among two firewall rules, which arises when the effect of one rule is influenced or altered by another one, e.g. the actions of two rules (that are both satisfied simultaneously) contradict each other. In literature, several solutions have been proposed for firewall conflict detection. However, a detected conflict has to be solved manually by administrators, and none of the proposed approaches even tries to minimize such necessary administration operations. In order to overcome this limitation, the Netgrop (Computer Networks research group at Politecnico di Torino) recently defined a new approach, which also minimizes the number of resolutions that have to be performed by administrators to fix the detected conflicts. The aim of this project is to design and implement a RESTful web service based on this new approach.

The assignment can be divided into the following parts:

a. Design data representations (described by means of an XML schema) respectively of the firewall rules and anomalies.

b. Design a REST API that can receive as input the rules using the previously defined data formats (XML or JSON) and that can perform the optimized analysis (the algorithm is provided).

A documentation of the data formats and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

Deadline: The end of the February exam session

3) Integration of Verefoo and Verigraph (max 3 students)

Verefoo (https://github.com/netgroup-polito/verifoo) is a Java framework that can automatically allocate and configure network security functions in a virtualized network with a policy-based approach called refinement. Instead, Verigraph (https://github.com/netgroup-polito/verigraph) performs a formal verification of reachability policies in a network service (e.g., it checks that certain flows of packets can or cannot reach a given destination from a given source). Both tools have been designed explicitly for new generation networks based on Network Functions Virtualization (NFV) and Software-Defined Networking (SDN). Originally, they were two separate tools, but our research group is currently combining them into a single one, in order to provide joint refinement and verification. Even though the XML schemas on which they are based present some similarities, each one of them is not sufficiently expressive to fulfill the needs of both frameworks.

This project consists of:

a. Designing a new data format, described by means of an XML schema, for the XML/JSON representation of a virtualized network service and of the security policies that must be fulfilled. This will require an analysis of the current XML
schemas of the two frameworks, identifying all the features that are needed in the joint schema.

b. Implement a REST API that can receive and store the input using the previously defined data formats (XML or JSON), and that can execute joint refinement/verification by calling the Verefoo/Verigraph engine.

c. Add a simple static single-page web application to the REST API, which uses the REST API by means of AJAX calls, and allows users to make simple experiments with Verigraph and Verefoo.

A documentation of the data formats, including some examples, a test client for the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

Deadline: The end of the February exam session

4) Policy-driven NSF selection (max 2 student)

Verefoo (verified refinement and optimization orchestrator) is an extension of Verigraph (see project 3 for a brief description), which is capable of performing joint refinement and verification.

Refinement is the process that determines the resources needed to satisfy security requirements. It translates high-level requirements into low-level configurations that may be enforced by the system, and it verifies that the set of lower level policies or configurations actually meets the requirements.

This project consists of implementing a specific task of the refinement process: the optimal selection of the best network security functions (NSFs) that correctly enforce the security policy (i.e. security requirements). The assignment can be divided into the following parts:

b) Design a REST API that can receive both a policy and a catalog using specific data formats (XML or JSON) and that can perform the selection.

c) Implement a library of Java classes for selecting (just selection, without ordering) from a given catalog the security functions necessary to enforce a given a policy, by using Z3 OPT (https://rise4fun.com/Z3/tutorial/optimization), a tool that can solve optimization problems under given hard constraints. Finally, the library has to be used to implement the RESTful web service designed at point b). The heuristics for selection (i.e. the hard constraints and cost functions for optimization) will be made available when the project work starts.

A documentation of the data formats and of the web service, a client for testing the service, and an ant script that automates compilation and running of the various programs and tests must also be provided as part of the solution.

Deadline: The end of the February exam session