Hybrid Multi-module Security Policy Verification

Igor Kotenko, Olga Chervatuk, Ekaterina Sidelnikova and Artem Tishkov
St. Petersburg Institute for Informatics and Automation, 39, 14th Linija, St. Petersburg, Russia
{ivkote, ovch, sidelnikova}@comsec.spb.ru, avt@computer.edu.ru

To build a powerful and flexible security policy verification tool, it is very important to use the approach which allows covering all possible inconsistencies, has open (extendable) architecture and efficient verification implementation. We suggest using a family of different verification modules each of which can work with acceptable computational complexity for the particular types of conflicts, the system scale and the policy complication. The poster describes a common approach to security policy verification and presents a novel hybrid multi-module SEcurity Checker (SEC) software tool that can serve as a security policy debugger for various categories of security policy, including authentication, authorization, filtering, channel protection and operational rules.

Utilizing several verification modules (Figure 1), SEC combines general verification methods with specialized algorithms which better handle particular types of inconsistencies. The SEC architecture is open for addition of new modules and therefore offers a flexible and scalable solution for security policy verification problems. The proposed classification of inconsistencies by security rule category serves as a basis for multi-module SEC architecture. SEC takes input data from XML repository: the specification of the network in System Description language and the security policy in Security Policy Language [1]. Verification modules are controlled by manager determining which modules and in which sequence will be used for verification. System administrator is able to activate the needed modules for verification. Every module generates log with verification results and auxiliary information. The “Verification result” form allows browsing detected inconsistencies. Each inconsistency is described by inconsistency type, conflicting rules, possibility to resolve, and resolution strategies. User selects resolution strategy and resolves inconsistency by corresponding buttons.

In the poster we try to address the following issues: how the same properties are translated into multiple formalisms, how are they expressed and translated, what is the policy and system description in each one of the formalisms, what conflicts can be detected, etc.

To analyze efficiency of verification modules we have fulfilled a multitude of experiments for various security policies, system configurations and conflicts. The completeness of conflict detection and the computational complexity have been estimated for main verification modules. We have compared the efficiency of the verification modules implemented and possibility to use different formal approaches in practice for large-scaled computer network systems. Some results were very interesting, for example, the Event Calculus module works more effectively than some specialized modules if the number of conflicts is not very high.

The research is supported by grants of Russian Foundation of Basic Research, Department for Informational Technologies and Computation Systems of the RAS, Russian Science Support Foundation and by the EC as part of the POSITIF project.

References