The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols

Eirik Klevstad

February 5, 2016

1 Introduction

Difficult for humans to analyse security protocols that are becoming more and more complex.

Research in formal analysis of security protocols.

Most effective approach so far: Automated falsification or verification of such protocols with state-of-the-art tools such as ProVerif and Avispa.

Based on pattern refinement algorithm, providing concise representations of infinite sets of traces.

Assist in analysis of attacks and possible protocol behaviours.

2 The Scyther Tool

Provides a GUI and command-line interface.

Combines a number of novel features with state-of-the-art performance.

1) Guaranteed to terminate whilst allowing to prove correctness of protocols for an unbounded number of sessions.

Provides useful results even when there are no attacks.

No attacks existing within a certain bound.

2) Assists in protocol analysis by providing different attack classes (or protocol behaviours) instead on only one attack trace as most other tools.

3) Multi-protocol analysis. Analyses the parallel composition of two sub-protocols. Has been infeasible before due to the enormous state-space, but Scyther enables such analysis to be performed.

spdl language

Three ways:

1) Verify whether security claims in the protocol description holds

2) Automatically generate appropriate security claims for the protocol and verify these.

3) Analyse the protocol by performing complete characterization.

2.1 Verification of claims

Specification of properties in terms of claim events. Inside roles we can specify that a certain value is, say, secret, and the Scyther will verify whether or not our claim holds.

2.2 Automatic claims

If the protocol contains no such claims, Scyther can generate claims on its own. Claims added to claim that the parties of the protocol have communicated in the expected way.

Added for all locally generated values (nonces) and variables.

2.3 Complete characterization

Each role can be characterized = Scyther analyses the protocol and provides a finite representation of all traces that contain an execution of the protocol role.

Most cases = 1-5 possible executable patterns.

Manually inspection: Quickly gain insight in potential problems with the protocol and modify these.

Scyther addresses the undecidability of the the security problem by

1) Significantly improving and extending class pruning theorems

2) Introducing a parameter that limits the pattern size, ensuring termination

3 Performance and applications