Using the coral System to Discover Attacks on Security Protocols

41
Using the CORAL System to Discover Attacks on Security Protocols

Graham Steel, Alan Bundy, Ewen Denney

Introduction

Inductive theorem provers are frequently employed in the verification of programs, algorithms, and protocols. Programs and algorithms often contain bugs, and protocols may be flawed, causing the proof attempt to fail. However, it can be hard to interpret a failed proof attempt: it may be that some additional lemmas need to be proved or a generalisation made. In this situation, a tool which can not only detect an incorrect conjecture, but also supply a counterexample in order to allow the user to identify the bug or flaw, is potentially very valuable. Here we describe such a tool, CORAL, based on a previously under-exploited feature of proof by consistency. Proof by consistency is a technique for automating inductive proofs in first-order logic. Originally developed to prove correct theorems, this technique has the property of being refutation complete, i.e., it is able to refute in finite time conjectures which are inconsistent with the set of hypotheses. Recently, Comon and Nieuwenhuis have drawn together and extended previous research to show how it may be more generally applied [4]. CORAL is the first full implementation of this method.

We have applied CORAL to the analysis of cryptographic security protocols. Paulson has shown how these can be modelled inductively in higher-order logic [16]. By devising a suitable first-order version of Paulson's formalism, we are able to automatically refute incorrect security conjectures and exhibit the corresponding attacks. The flexibility of the inductive formalism allows us to analyse group protocols, and we have discovered new attacks on such a protocol (the Asokan-Ginzboorg protocol for ad-hoc Bluetooth networks [2]) using CORAL.

In the rest of the paper, we first briefly look at the background to the problem of refuting incorrect conjectures and the formal analysis of security protocols. Then we outline the Comon-Nieuwenhuis method. We describe the operation of CORAL and then show how it can be applied to the problem of protocol analysis. Finally, we describe some possible further work, including some other possible applications for CORAL, and draw some conclusions.

Background

The refutation of incorrect inductive conjectures has been studied before, e.g., by Protzen [17], Reif [18], and Ahrendt [1]. Ahrendt's method works by constructing a set of clauses to send to a model generation prover and is restricted to free datatypes. Protzen's technique progressively instantiates terms in the formula to be checked, using the recursive definitions of the function symbols involved. It finds many small counterexamples. Rief's method instantiates the formula with constructor terms and uses simplifier rules in the prover KIV to evaluate truth or falsehood. His method is a marked improvement on Protzen's, but is too naïve for a situation like protocol checking, where it is not obvious what combination of constructor terms constitutes a possible exchange of messages.

Proof by consistency

Proof by consistency was originally conceived by Musser [14] as a method for proving inductive theorems by using a modified Knuth-Bendix completion procedure. It was developed by various authors, [8, 10, 6], for the next fifteen years (see [20] for the story), but interest waned, as it seemed too hard to scale the technique up to proving larger conjectures. However, later versions of the technique did have the property of being refutation complete, that is, able to spot false conjectures in finite time.

The Comon-Nieuwenhuis method

Comon and Nieuwenhuis [4] have shown that the previous techniques for proof by consistency can be generalised to the production of a first-order axiomatisation A of the minimal Herbrand model such that $A \cup E \cup C$ is consistent if and only if C is an inductive consequence of E. With A satisfying the properties they define as a Normal I-Axiomatisation, inductive proofs can be reduced to first-order consistency problems and so can be solved by any saturation based theorem prover. There is not room here to give a full formal account of the theory, but informally, a proof attempt involves two parts: in one, we pursue a fair induction derivation. This is a restricted kind of saturation, where we need only consider overlaps between axioms and conjectures. In the second part, every clause in the induction derivation is checked for consistency against the I-Axiomatisation. If any consistency check fails, then the conjecture is incorrect. If they all succeed, and the induction derivation procedure terminates, the theorem is proved. Comon and Nieuwenhuis have shown refutation completeness for this system, i.e., any incorrect conjecture will be refuted in finite time, even if the search for an induction derivation is non-terminating.

Cryptographic security protocols

Cryptographic protocols are used in distributed systems to allow agents to communicate securely. They were first proposed by Needham and Schroeder [15]. Assumed to be present in the system is a spy, who can see all the traffic in the network and may send malicious messages in order to try to impersonate users and gain access to secrets.

Although security protocols are usually quite short, typically 2–5 messages, they often have subtle flaws in them that may not be discovered for many years. Researchers have applied various formal techniques to the problem to try to find attacks on faulty protocols and to prove correct protocols secure. These approaches include belief logics such as the so-called BAN logic [3], state machines [5, 11], model checking [12], and inductive theorem proving [16]. Each approach has its advantages and disadvantages. For example, the BAN logic is attractively simple and has found some protocol flaws, though in other cases found flawed protocols correct. The model-checking approach can find flaws very quickly, but can only be applied to finite (and typically very small) instances of the protocol. This means that if no attack is found, there may still be an attack upon a larger instance. Modern state-machine approaches [13, 19] can also find and exhibit attacks quickly, but require the user to choose and prove lemmas in order to reduce the problem to a tractable finite search space. The inductive method deals directly with the infinite-state problem and assumes an arbitrary number of protocol participants, but proofs are tricky and require days or weeks of expert effort. If a proof breaks down, there have previously been no automated facilities for the detection of an attack.

Implementation

Figure 1 illustrates the operation of CORAL, built on the SPASS theorem prover [23]. The induction derivation, using the Comon-Nieuwenhuis method as described above, is pursued by the modified SPASS prover on the right of the diagram. As each clause is derived, it is passed to the refutation control script on the left, which launches a standard SPASS prover to do the check against the I-Axiomatisation. The parallel architecture allows us to obtain a refutation in cases where the induction derivation does not terminate, as well as allowing us to split the process across multiple machines in the case of a large problem. Experiments with the system show good performance on a variety of incorrect conjectures from the literature and our on own examples [21].

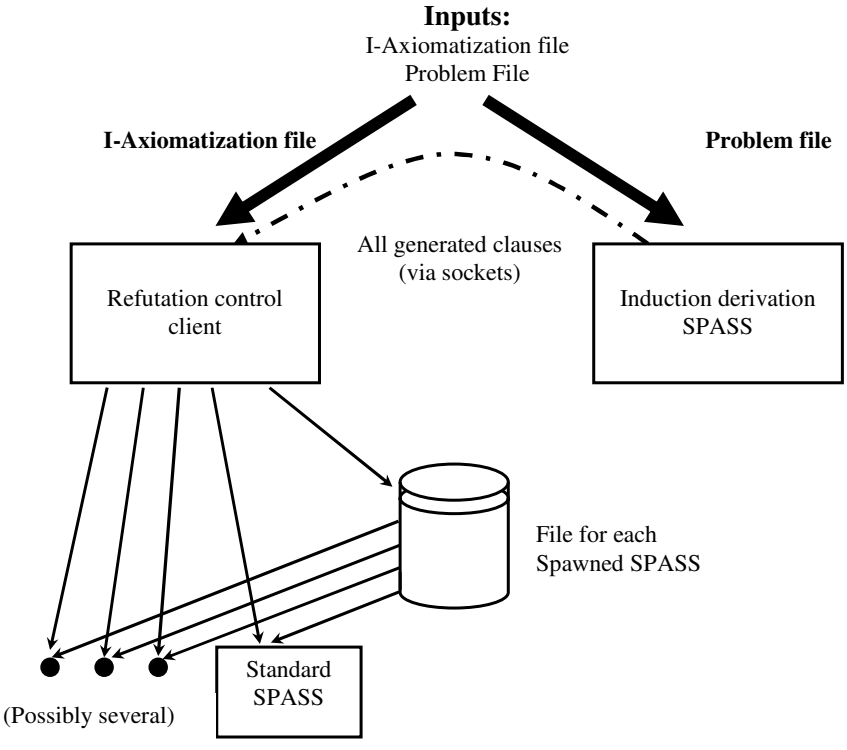

Figure 1: CORAL system operation

Application to cryptographic security protocols

Paulson's inductive approach has been used to verify properties of several protocols [16]. Protocols are formalised in typed higher-order logic as the set of all possible traces. Properties of the security protocol can be proved by induction on traces. However, as Paulson observed, a failed proof state can be difficult to interpret. Even an expert user will be unsure as to whether it is the proof attempt or the conjecture that is at fault. By applying our counterexample finder to these problems, we can automatically detect and present attacks when they exist. The use of an inductive model also allows us to consider protocols involving an arbitrary number of participants in a single round, e.g., conference-key protocols. Paulson's formalism is in higher-order logic. However, no 'fundamentally' higher-order concepts are used—in particular, there is no unification of functional objects. Objects have types, and sets and lists are used. All this can be modelled in first-order logic. The security protocol problem has been modelled in first-order logic before, e.g., by Weidenbach [24]. He used a two-agent model,

with fixed roles for participants, and just one available session key and nonce (a nonce is a unique identifying number), and so could not detect certain kinds of parallel session attacks described above. Like Paulson's, our model allows an indeterminate and unbounded number of agents to participate, playing either role and using an arbitrary number of fresh nonces and keys. Details of the model are in our earlier paper [21], but we will highlight now some recent developments.

We have modified our formalism slightly to make attacks easier to find. The idea is to prune out branches of the search space that cannot lead to an attack, or branches which represent a less succinct expression of a state already reached. For example, we merged together the formulae allowing the spy to send a fake message with those for the standard protocol, so that the spy can only send messages which look like a part of the real protocol. Sending anything else cannot fool any honest participants, since they only respond to correctly formed messages. We also have a reduction rule which prunes out clauses which represent states where the spy has sent two messages in a row. The spy can't gain anything from doing this, so by chopping off these branches we make the search problem more tractable.

With these improvements CORAL has rediscovered a number of known attacks, including the well known ones on the Needham-Schroeder public-key and Neuman-Stubblebine shared-key protocols. It can also find the attack on the simplified Otway-Rees protocol, an attack which requires an honest agent to generate two fresh nonces and to play the role of both the initiator and the responder. Recently, CORAL found two new attacks on the Asokan-Ginzboorg protocol for establishing a secure session key in an ad-hoc Bluetooth network [2]. Details of the attacks and a description of how we modelled this group protocol in a general way without restricting to a small fixed instance are in a forthcoming paper [22].

Further work

Future work will include testing the CORAL system on more group-key protocols. As CORAL is built on SPASS, a theorem prover capable of equational reasoning, we should be able to reason about some simple algebraic properties of the cryptosystems underlying protocols, such as Diffie-Helman type operations. In particular, Asokan and Ginzboorg have proposed a second version of their protocol that uses these kinds of operations, which would be an ideal candidate for future investigation.

There has been a proliferation of protocol analysis tools in recent years, and in the longer term we don't intend to try and compete with others for speed of attack finding or by analysing an enormous corpus of protocols. Rather, we intend to try to exploit the flexibility of our system as a general tool for inductive counterexample finding and apply it to some other security problems. One idea is to use the system to model security problems at a higher level. We could

model a company's computer network as a system of local networks and servers, firewalls, etc., all with formally defined behaviour, and examine how interactions in the presence of intruders might lead to exploitable vulnerabilities. To deal with larger problems like this, we might need to enhance SPASS to exploit domain knowledge a little more. Two possible ideas we intend to explore are a user-defined strategy that can vary as the proof proceeds and a critics mechanism [9] to suggest pruning lemmas. In theory, CORAL can also show security properties of protocols to be correct when there are no attacks to be found. However, to make this work in practice would require some considerable work. The formulae to be proved are significantly larger than the kinds of examples that have been proved by proof by consistency in the past. The critics mechanism for suggesting lemmas could help with this.

Conclusions

We have presented CORAL, our system for refuting incorrect inductive conjectures, and have shown how it can be applied to the problem of finding attacks on faulty security protocols. Our formalism is similar to Paulson's, which allows us to deal directly with protocols involving an arbitrary number of participants and nonces, and with principals playing multiple roles. CORAL has discovered a number of known attacks, and some new attacks on a group-key protocol. In the longer term, we hope to apply the system to other, related security problems and exploit its ability to do equational reasoning in order to analyse some crytpoanalytic properties of protocols. (This paper is a shortened and updated version of [21].)

References

1. AHRENDT, W., 'Deductive search for errors in free data type specifications using model generation,' in *CADE-18, 18th International Conference on Automated Deduction*, 2002.
2. ASOKAN, N., AND GINZBOORG, P., 'Key agreement in ad-hoc networks,' *Computer Communications*, vol. 23, no. 17, 2000, pp. 1627–1637.
3. BURROWS, M., ABADI, M., AND NEEDHAM, R., 'A logic of authentication,' *ACM Trans. on Computer Systems*, vol. 8, no. 1, February 1990, pp. 18–36.
4. COMON, H., AND NIEUWENHUIS, R., 'Induction = I-Axiomatization + First-Order Consistency,' *Information and Computation*, vol. 159, no. 1–2, May/June 2000, pp. 151–186.
5. DOLEV, D., AND YAO, A., 'On the security of public key protocols,' *IEEE Trans. in Information Theory*, vol. 2, no. 29, March 1983, pp. 198–208.
6. GANZINGER, H., AND STUBER, J., 'Inductive theorem proving by consistency for first-order clauses,' in Rusinowitch, M. and Rémy, J.J., (eds.), *Proc. 3rd International*

Workshop on Conditional Term Rewriting Systems, Pont-à-Mousson, France, Springer, LNCS vol. 656, pp. 226–241.

7. GANZINGER, H., (ED.), *Automated deduction*, *CADE-16: 16th International Conference on Automated Deduction*, Trento, Italy, July 1999, Lecture Notes in Artificial Intelligence 1632, Springer-Verlag.
8. HUET, G., AND HULLOT, J., 'Proofs by induction in equational theories with constructors,' *J. of Computer Systems and System Sciences*, vol. 25, no. 2, 1982, pp. 239–266.
9. IRELAND, A., 'Productive use of failure in inductive proof,' *J. of Automated Reasoning*, vol. 16, no. 1–2, 1996, pp. 79–11.
10. JOUANNAUD, J.-P., AND KOUNALIS, E., 'Proof by induction in equational theories without constructors,' *Information and Computation*, vol. 82, no. 1, 1989, pp. 1–33.
11. KEMMERER, R., MEADOWS, C., AND MILLEN, J., 'Three systems for cryptographic protocol analysis,' *J. of Cryptology*, vol. 7, 1994, pp. 79–130.
12. LOWE, G., 'Breaking and fixing the Needham Schroeder public-key protocol using FDR,' in *Proc. TACAS*, LNCS 1055, Springer Verlag, 1996, pp. 147–166.
13. MEADOWS, C., 'The NRL protocol analyzer: An overview,' *J. of Logic Programming*, vol. 26, no. 2, pp. 113–131, 1996.
14. MUSSER, D., 'On proving inductive properties of abstract data types,' *Proc. 7th ACM Symp. on Principles of Programming Languages*, 1980, pp. 154–162.
15. NEEDHAM, R.M., AND SCHROEDER, M.D., 'Using encryption for authentication in large networks of computers,' *Comm. ACM*, vol. 21, no. 12, December 1978, pp. 993–999.
16. PAULSON, L.C., 'The inductive approach to verifying cryptographic protocols,' *J. of Computer Security*, vol. 6, 1998, pp. 85–128.
17. PROTZEN, M., 'Disproving conjectures,' in Kapur, D., ed., *CADE-11: 11th Conf. on Automated Deduction*, Saratoga Springs, NY, June 1992. Springer, Lecture Notes in Artificial Intelligence 607, pp. 340–354.
18. REIF, W., SCHELLHORN, G., AND THUMS, A., 'Flaw detection in formal specifications,' in *IJCAR'01*, 2001, pp. 642–657.
19. SONG, D., 'Athena: A new efficient automatic checker for security protocol analysis,' *Proc. 12th IEEE Computer Security Foundations Workshop*, 1999.
20. STEEL, G., 'Proof by consistency: A literature survey,' March 1999. http://homepages.inf.ed.ac.uk/s9808756/papers/lit-survey.ps.gz
21. STEEL, G., BUNDY, A., AND DENNEY, E., 'Finding counterexamples to inductive conjectures and discovering security protocol attacks,' *Proc. of the Foundations of Computer Security Workshop*, 2002, pp. 49–58; also in *Proc. of the Verify '02 Workshop*. Also available as Informatics Research Report EDI-INF-RR-0141.
22. STEEL, G., BUNDY, A., AND MAIDL, M., 'Attacking the Asokan-Ginzboorg protocol for key distribution in an ad-hoc Bluetooth network using CORAL,' to appear in *Proceedings of FORTE 2003* (work in progress papers).
23. WEIDENBACH, C., ET AL., 'System description: SPASS version 1.0.0,' in Ganzinger [7], pp. 378–382.
24. WEIDENBACH, C., 'Towards an automatic analysis of security protocols in first-order logic,' in Ganzinger [7], pp. 314–328.