A Formal Data-Model of the CORBA Security Service

David Basin Frank Rittinger Luca Viganò
Institut für Informatik, Albert-Ludwigs-Universität Freiburg
Georges-Köhler-Allee 52, D-79110 Freiburg, Germany
{basin,rittinge,luca}@informatik.uni-freiburg.de
http://www.informatik.uni-freiburg.de/~{basin,rittinge,luca}

ABSTRACT

We use the formal language Z to specify and analyze the security service of CORBA. In doing so, we tackle the problem of how one can apply lightweight formal methods to improve the precision and aid the analysis of a substantial, informal specification. Our approach is scenario-driven: we use representative scenarios to determine which parts of the informal specification should be formalized and then verify the formal specification against the requirements of these scenarios.

Keywords: CORBA Security, Formal Methods, Specification, Z.

1. THE CORBA SECURITY SERVICE

The CORBA security service specification of the OMG [3] consists of more than 350 pages, defining interfaces between standard security technologies (such as Kerberos or SSL) and the ORB (a 'broker' connecting the different system components and managing communication between them) or the business applications. The security architecture is based on a *Trusted Computing Base* (TCB), which encapsulates all security-related functionality in a small trusted kernel. Given this encapsulation, the business components can then be implemented independently of concrete security mechanisms. This has the advantage that the code implementing the security functionality is minimized, security policies are easier to keep consistent, and the trust between different components is limited to the trust between the TCBs of different domains.

The overall specification is quite complex, and this complexity stems from several factors. First, the problem itself is inherently complex. Any specification-conform implementation must achieve multilateral security guarantees for distributed object systems where the objects execute on different platforms in different environments. Second, the solution must be *loose* (or *open*). The OMG, a consortium whose members come from over 800 companies and organizations, does not define new technologies, but rather develops standards that encompass the technologies of its members. Hence, any specification of a standard by the OMG must be loose in the sense that it admits as models all implementations given by OMG members. This looseness is achieved, for example, by simply giving signatures for interfaces, where the use of general types does not constrain the values of possible implementations. More specifically, to achieve platform independence, all components in a CORBA system are known only through their interfaces, which are defined in a programming language independent way using the CORBA-specific *Interface Definition Language* (IDL). The OMG-specification, however, does not always have a clear semantics of what functionality implementations of the interfaces must provide; what semantics there is, is given only by informal prose (i.e. natural language) or ad-hoc diagrams. This allows future developers and implementors of the security service to interpret the specification in unintended ways that may lead to security-holes in applications built using the service. Finally, the specification has the complexity inherent in any document designed by a political process.

Our research is motivated by the question of whether, despite this complexity and the informal nature of the OMG-specification, it is possible to reason about the security of systems built using this service. Moreover, whether one can do this in a *lightweight* way without plunging into the black hole of formalizing both CORBA and its security service in their entireties. A formalization of the specification helps solve this problem, as it provides a rigorous semantics that can serve as a basis for communication between different developers, implementors and users of the system. Even a partial formalization is of assistance here, e.g. clarifying the data-model, although leaving control semi-formal. Moreover, even with a partial formalization, it is still possible, and becomes interesting, to formally establish security properties and thus demonstrate that the specification is "tight" enough.

2. SPECIFICATION APPROACH

To carry out a formal analysis in a pragmatic way, we have reduced the complexity of the problem along two dimensions. First, we took a data-oriented view of the system and formalized the service's data-model in the Z language[1].

[1] Z [5] is a formal language based on typed set theory and first-order logic with equality. We have chosen Z for our work as the CORBA security service specification is heavily

Second, we took a scenario-driven approach to determine which parts of the OMG-specification should be formalized, and then verified our formal specification against the requirements of these scenarios. This reduction allowed us to give a partial, but meaningful, formal analysis, without formalizing both CORBA and its security service in their entireties, or managing the additional complexity of employing multilanguage formalizations.

As examples, we consider here two requirements that, as postulated in the OMG-specification, we can impose on the specification, namely: (i) that upon authentication, no principal can gain more attributes (i.e. more credentials) than the system allows, and (ii) that a valid security association must be established prior to a secure object invocation. (Other scenarios and requirements are analyzed in [1, 4].)

To analyze these two requirements, we consider the following two scenarios: *principal authentication* and *security association*. The first describes that whenever a principal enters a CORBA system for the first time, the principal must authenticate itself to the system and is assigned identity and privilege attributes; the second describes that prior to object invocation, the CORBA environment establishes a security association, which is used to establish mutual trust between the client and the target and to negotiate the details of the communication.

From these scenarios we extract two kinds of specifications. First, we identify what parts of the OMG-specification must be formalized to describe those parts of the security service relevant for each scenario. Second, we determine the scenarios' security requirements, which constitute a (partial) specification of properties of the security service. Both of these specifications are then formalized in Z and provide the basis for formally establishing that the security service specification has the desired properties.

3. FORMAL ANALYSIS

Given the two kinds of formal specifications discussed above, we carry out our formal analysis by proving that the Z-specification of the security service satisfies the Z-specification of the requirements associated with each scenario.

In general, our analysis yields three different kinds of results: (i) the specification adheres to its requirements, which follows by proving a formula that represents the requirements; (ii) the specification does not adhere to its requirements, but may do so after the specification is refined; and (iii) the requirements cannot be expressed using our formalization in the first place.

During our analysis we have obtained results of all three kinds: we have been able to show that some of the security requirements are satisfied, while for others a refinement of our (and thus of the OMG's) specification is needed. Hence our formal analysis suggests how the OMG-specification can be improved both by removing subtle errors (results of the second kind) and omissions (results of the third kind).

data-oriented in that approximately half of it is focused on data, interfaces, and their functional requirements (while the remainder focuses on inter-ORB communication).

To aid both specification and analysis, and increase the confidence in the results, we have employed tool support. In particular, we wrote and type-checked our Z-specification using the ZETA-tool [6], which provided substantial assistance not only in finding mistakes in preliminary versions of our Z-specification but also in uncovering a number of errors and omissions in the original OMG-specification. Our proofs of results of the first two kinds were carried out by hand and also machine-checked using HOL-Z, an embedding of Z in higher-order logic within the generic theorem prover Isabelle [2].

4. CONCLUSIONS

Our work contributes to the understanding of how formal languages like Z can be used to model and analyze complex, informal standards. A common misconception is that formal methods are too heavy in such cases, as they require a complete formalization of the standard, which is of overwhelming complexity due to its size and the need to integrate different views of the system, e.g. static data-models with dynamic event or process-oriented models. We show here that it is possible to proceed with a pragmatic "light touch" in the formalization: One can proceed in a scenario-driven way, and even restrict attention to a particular view of the system (namely the data-view), and still arrive at general, useful results, with limited resources.

A formal specification like ours provides the basis for a formal verification of many of the multilateral security properties of the security service and of applications built upon it. Moreover, as Z supports high-level, readable specifications, we believe that our formal specification can also be of use to different vendors who wish to implement the CORBA security service or compare (or even formally verify) their implementation against the specification.

5. REFERENCES

[1] D. Basin, F. Rittinger, and L. Viganò. A formal analysis of the CORBA security service. Technical Report 154, Institut für Informatik, Albert-Ludwigs-Universität Freiburg, 2001.

[2] Kolyang, T. Santen, and B. Wolff. A structure preserving encoding of Z in Isabelle/HOL. In J. von Wright, J. Grundy, and J. Harrison, eds., *Proceedings of TPHOLs'96*, LNCS 1125, pp. 283–298. Springer, 1996. See also `http://www.informatik.uni-freiburg.de/~wolff/doc/isa_doc/HOL-Z/home.html`

[3] OMG Security Working Group. CORBAservices Security Service Specification v1.2. Technical report, Document: formal/98-12-17, OMG, 1998.

[4] F. Rittinger. Formale Analyse des CORBA Sicherheitsdienstes. Diplomarbeit, Institut für Informatik, Albert-Ludwigs-Universität Freiburg, Germany, 2000. In German.

[5] J. Spivey. *The Z Notation*. Prentice Hall, 1998.

[6] ZETA. `http://uebb.cs.tu-berlin.de/zeta/`.