Credentials-Based Authorization: Evaluation and Implementation

Abstract of Plenary Lecture

Fred B. Schneider*

Department of Computer Science
Cornell University
Ithaca, New York 14558
U.S.A
fbs@cs.cornell.edu

Nexus is a new operating system that runs on computers equipped with tamper-proof secure co-processors; it is designed to support the construction of *trustworthy applications*—applications where actions can be attributed with some measure of confidence and where trust assumptions are explicit. To realize these goals, Nexus implements

- a novel architecture, which enables the trusted computing base to be relatively small,
- an expressive and flexible framework for making and using attestations about current and future states of a computation, and
- a high-performance cryptographically-implemented memory-isolation abstraction.

This talk focuses on how authorization policies are specified and implemented in Nexus.

We posit a guard for each resource, as has become standard. That guard receives client requests for access to the resource; it either grants or denies each request. Nexus innovates in how guards make authorization decisions and in what information is used. Our approach builds on logics having "says" and "speaks for" operators, and some surprising technical issues arise (some of which present opportunities for future research).

In Nexus, authorization decisions can involve a set of credentials that either accompany the request or that the guard obtains from elsewhere. Each *credential* is a statement whose validity can be checked by any principal. An authorization policy defines a set of requirements satisfied by suitable credentials. Given, for example, the policy that some file is accessible only to students, a guard would authorize a read request from Alice only if that guard obtains credentials attesting to Alice being a student: an attribute certificate about Alice's student status this semester, signed by a key purporting to speak for a dean; a delegation certificate asserting that the dean's key speaks for the dean; a delegation certificate

* Supported in part by AFOSR grant F9550-06-0019, National Science Foundation Grants 0430161 and CCF-0424422 (TRUST), and a gift from Microsoft Corporation.

L. Arge et al. (Eds.): ICALP 2007, LNCS 4596, pp. 12–14, 2007.

signed by the university president's key asserting that the dean is indeed the holder of that office; and so on.

The advantage of such *credentials-based authorization* is that it can decentralize authorization decisions in a way that mirrors an actual authority structure. Different principals are trusted on different aspects of the over-all authorization decision, according to their expertise or access to appropriate information. Moreover, the existence of suitable credentials at a guard documents the role participating principals played in determining each authorization decision outcome and, therefore, provides an auditable justification for that decision.

In Nexus, as in prior work, every authorization policy is encoded as a formula, herein called the *authorization goal formula*, in a logic. Credentials are checked for validity, and the (sub)set of valid credentials are treated as axioms of a logic. In the prior work, however, access requests are allowed if and only if the guard can establish that the authorization goal formula for that request is valid in all models satisfying the axioms (viz, associated credentials). The guard thus must implement a theorem prover, a decision procedure, or—if requests must be accompanied by a proof of the authorization goal formula—a proof checker.

By requiring that an authorization goal formula be valid, guards in the prior work are limited to implementing monotonic authorization policies. This is a significant restriction. It rules out many authorization policies that depend on the system state (which is always changing and might over time change in ways that invalidate a conjunct). Authorization policies that limit the number of times a particular resource may be read are an important class of authorization policies that depend on state. Also, policies that admit revocation become difficult to specify, hence enforce.

Authorization policies in Nexus need not be monotonic and may depend on the state. Implementation realities do force us to prohibit certain non-monotonic authorization polices—in particular, those that include conjuncts asserting the absence of credentials. First, it is difficult to ascertain whether such a conjunct is true; a denial of service attack might block the guard from receiving a credential even though that credential exists. Second, it is difficult to ensure that such a conjunct remains true short of freezing activity elsewhere in the system.

Nexus guards can support non-monotonic policies because the guard merely determines whether an authorization goal formula is satisfied.[1] That is, guards are evaluators and not validity checkers. Theorem proving is still necessary for determining the truth-value of certain clauses, because some conjuncts (e.g., "A speaks for B") cannot always be evaluated directly by inspecting a single credential or the system state. Nexus guards thus do have access to a proof checker and, when necessary, expect requests to be accompanied by proofs.

The implementation of guards in Nexus involves:

- a means to check whether formulas in the logic are satisfied, which leads to a (new) operational interpretation of "says" and "speaks for" operators,

[1] This raises some interesting but not insurmountable issues in connection with the usual Kripke interpretations of "says" and "speaks for" logics.

- trusted ways to make system state available to guards, including a kernel-supported introspection service that provides a window not only into each process's memory but also into various other aspects (e.g., the presence of channels and the identities of their endpoints) of executing processes,
- ways of representing credentials and conjuncts of authorization goal formulas as executables rather than only as static certificates, and
- various protocols to freeze aspects of system operation so that an authorization goal formula that is found to be true can be forced to remain true as long as needed by the semantics of the operation whose authorization was being regulated.

These innovations, supported by examples, form the core of the talk.

Acknowledgment. Nexus is a collaboration involving Cornell Ph.D. students Oliver Kennedy, Alan Shieh, Kevin Walsh, and Dan Williams; postdoctoral associate Patrick Reynolds; and faculty E. Gün Sirer and Fred B. Schneider. The work on credentials-based authorization reported herein is being done jointly with Kevin Walsh and E. Gün Sirer.