22nd IEEE Computer Security Foundations Symposium (CSF 2009)
Short Talks Session (Session Chair: Anupam Datta)
July 9, 2009 (16:00 - 17:15)

1) From Information-Flow Policies to Cryptographic Mechanisms: A Secure Compiler for Distributed Programs

Cedric Fournet, Microsoft Research
Gurvan le Guernic, MSR-INRIA
Tamara Rezk, INRIA

We enforce information flow policies in programs that run at multiple locations, with diverse levels of security. We build a compiler from a small imperative language with locality and security annotations down to distributed code linked with concrete cryptographic libraries. Our compiler splits source programs into local threads; inserts checks on auxiliary variables to enforce the source control flow; implements shared distributed variables using instead a series of local replicas with explicit updates; and finally selects cryptographic mechanisms for securing the communication of updates between locations. We establish computational soundness for our compiler: under standard cryptographic assumptions, all confidentiality and integrity properties of the source program also hold with its distributed code, despite the presence of active adversaries that control all communications and some of the program locations. We also present experimental results for the code obtained by compiling sample programs.

2) Intersection and Union Types for Secure Implementations

Michael Backes (Saarland University - MPI-SWS)
Catalin Hritcu (Saarland University)
Matteo Maffei (Saarland University)
Thorsten Tarrach (Saarland University)

We present a static analysis technique for verifying implementations of cryptographic protocols written in RCF, a core calculus of ML with support for concurrency. Security properties are formalized as authorization policies and statically enforced by a novel type system, which extends prior work on refinement types [Bengtson et al., CSF 2008] with intersection, union, and polymorphic types.

The expressiveness of our type system allows us to analyze implementations of important protocol classes that were out of the scope of existing static analysis techniques, such as protocols based on zero-knowledge proofs as well as protocols based on nested cryptography, signatures of private data, and public-key encryption of authenticated data.

Cryptographic primitives are considered as fully reliable building blocks and represented symbolically using a sealing mechanism. Zero-knowledge proofs, in particular, are specified in a high-level language and automatically compiled down to a symbolic implementation using seals.

We discuss ongoing work on applying our analysis technique to the Direct Anonymous Attestation protocol and to the recently proposed Civitas electronic voting system.

3) Modular Verification of Security Protocol Code by Typing

Karthikeyan Bhargavan, Microsoft Research
Cedric Fournet, Microsoft Research
Andrew D. Gordon, Microsoft Research

We propose a method for verifying the code of security protocols. Our method relies on invariants for cryptographic structures. We use predicates to characterize cryptographic structures, and we define pre- and post-conditions on cryptographic algorithms so as to maintain invariants. We present a theory to justify the soundness of code verification via our method. Our implementation uses F7, a type-checker for refinement types, that is, types including formulas to record invariants. As illustrated by a series of examples, our method can flexibly deal with a range of different cryptographic algorithms. We evaluate the method on a series of larger examples of protocol code, previously checked using a whole-program analysis performed by a tool chain relying on ProVerif, a leading domain-specific tool for cryptographic protocols. Our results indicate that compositional verification by typechecking with refinement types is more scalable than the best whole-program analysis currently available.

4) A generic security API for symmetric key management on cryptographic devices

Véronique Cortier
Graham Steel

We describe a new generic API for managing symmetric keys on a trusted cryptographic device. We state and prove security properties for the API. In this talk, we will in particular compare our API and its security properties to the one proposed by Cachin and Chandran, which will be presented in the main CSF programme the day before the 5 minute talk session.

5) Security Theorems via Model Theory

Joshua D. Guttman (MITRE and Worcester Polytechnic Institute)

A model-theoretic approach can establish security theorems for cryptographic protocols. Formulas expressing authentication and non-disclosure properties of protocols have a special form. They are quantified implications

for all xs . (phi implies for some ys . psi).

Models (interpretations) for these formulas are *skeletons*, partially ordered structures consisting of a number of local protocol behaviors. *Realized* skeletons contain enough local sessions to explain all the behavior, when combined with some possible adversary behaviors.

We show two results. (1) If phi is the antecedent of a security goal, then there is a skeleton A_phi such that, for every skeleton B, phi is satisfied in B iff there is a homomorphism from A_phi to B. (2) A protocol enforces for all xs . (phi implies for some ys . psi) iff every realized homomorphic image of A_phi satisfies psi.

Hence, to verify a security goal, one can use the Cryptographic Protocol Shapes Analyzer CPSA (TACAS, 2007) to identify minimal realized skeletons, or "shapes," that are homomorphic images of A_phi. If psi holds in each of these shapes, then the goal holds.

6) Tracking information flow in dynamic tree structures

Andrey Chudnov (Stevens Institute of Technology)

I am going to talk about the problem of tracking information flow in the presence of dynamic tree structures. While investigating the problem of secure information flow in JavaScript we have found that contemporary solutions do not handle the operations with the Document Object Model (DOM) tree adequately. This fact allows carefully constructed exploits to circumvent the policy and leak secret information to a third-party.

The paper that covers the attacks as well as the proposed enforcement mechanism in detail will appear in the proceedings of ESORICS'09. Joint work with Alejandro Russo and Andrei Sabelfeld (Chalmers University of Technology).

7) Towards a provably secure design of anonymous webs of trust

Michael Backes (Saarland University, MPI-SWS)
Stefan Lorenz (Saarland University)
Matteo Maffei (Saarland University)
Kim Pecina (Saarland University)

Over the last years, the Web has evolved into the premium forum for freely and anonymously disseminating and collecting information and opinions. However, the ability to anonymously exchange information, and hence the inability of users to identify the information providers and to determine their credibility, raises serious concerns about the reliability of exchanged information.

We propose a method to exchange messages preserving the anonymity of the sender while guaranteeing to the receiver the existence of a certain trust relation between her and the sender. Our technique is compliant to the OpenPGP standard so it can be used on top of any already existing web of trust implementing this standard.

The trust relations among users consist of certificate chains involving digital signatures. The fundamental idea of our method is to deploy non-interactive zero-knowledge proofs: users assert their trust level by proving in zero-knowledge the existence of such certificate chains. Since the proofs are zero-knowledge, they provably do not reveal any information about the users except for their trust levels; in particular, the proofs hide their identity.

We verify the security properties of our protocol in a fully automated manner. The protocol is specified in the applied pi-calculus, trust policies are formalized as authorization policies, and anonymity properties are defined in terms of observational equivalence relations. The verification of these properties is then conducted using an extension of recently proposed static analysis techniques for reasoning about symbolic abstractions of zero-knowledge proofs.

We intend to implement our protocol and incorporate our approach into GnuPG, a free implementation of the OpenPGP standard, in a push-button manner: the zero-knowledge proof creation and verification utilizes already existing webs of trust without any effort on the user side. Additionally we plan to offer the possibility to build new webs of trust using state-of-the-art signature schemes and zero-knowledge proof schemes to reduce the computation as well as the communication complexity and to give tighter security guarantees.

8) The Continuous Authentication

Ines Brosso (College of Computation and Informatics, Mackenzie Presbyterian University, Sao Paulo - Brazil)
Graça Bressan and Wilson V Ruggiero (Laboratory of Computer Architecture and Networks, Department of Computer and Digital System Engineering, Polytechnic School of São Paulo University, Sao Paulo - Brazil)

The continuous authentication of the user in application software is a complement to the initial authentication and it is necessary to guarantee that the user who was identified and authenticated in the beginning of a session is the same. This work presents a mechanism of continuous authentication of user using behavioral analysis, based on the evidences of the behavior, to establish trust levels to authenticate continuously the user during application software.
In this context, this work developed a security system named KUCAS (Known User Continuous Authentication System) that captures the environment context information using the contextual dimensions (who, where, when, what, why) defined by the Context-aware computing and analyzes the user behavior.

9) Catch Me If You Can: Permissive Yet Secure Error Handling

Aslan Askarov (Cornell) and Andrei Sabelfeld (Chalmers)

Program errors are a source of information leaks. Tracking these leaks is hard because error propagation breaks out of program structure. Programming languages often feature exception constructs to provide some structure to error handling: for example, the try {} catch {} blocks in Java and OCaml. Mainstream information-flow security compilers such as Jif and FlowCaml enforce rigid rules for exceptions in order to prevent leaks via public side effects of computation whose reachability depends on exceptions.

This work presents a general and permissive alternative to the rigid solution: the programmer is offered a choice for each type of error/exception whether to handle it or not. The security mechanism ensures that, in the former case, it is never handled and, in the latter case, it is always handled with the mainstream restrictions. This mechanism extends naturally to a language with procedures and output, where we show the soundness of the mechanism with respect to termination-insensitive noninterference.

10) From dynamic to static and back: Riding the roller coaster of information-flow control research

Alejandro Russo (Chalmers) Andrei Sabelfeld (Chalmers)

Historically, dynamic techniques are the pioneers of the area of information flow in the 70’s. In their seminal work, Denning and Denning suggest a static alternative for information-flow analysis. Following this work, the 90’s see the domination of static techniques for information flow. The common wisdom appears to be that dynamic approaches are not a good match for security since monitoring a single path misses public side effects that could have happened in other paths. Dynamic techniques for information flow are on the rise again, driven by the need for permissiveness in today’s dynamic applications. But they still involve nontrivial static checks for leaks related to control flow. This talk demonstrates that it is possible for a purely dynamic enforcement to be as secure as Denning-style static information-flow analysis, despite the common wisdom. We do have the trade-off that static techniques have benefits of reducing runtime overhead, and dynamic techniques have the benefits of permissiveness (this, for example, is of particular importance in dynamic applications, where freshly generated code is evaluated). But on the security

side, we show for a simple imperative language that both Denning-style analysis and dynamic enforcement have the same assurance: termination-insensitive noninterference.

11) Automatic analysis of distance bounding protocols
Sreekanth Malladiy (Dakota State University)
Bezawada Bruhadeshwar, Kishore Kothapalli (International Institute of Information Technology)

Distance bounding protocols are used by nodes in wireless networks to calculate upper bounds on their distances to other nodes. However, dishonest nodes in the network can turn the calculations both illegitimate and inaccurate when they participate in protocol executions. It is important to analyze protocols for the possibility of such violations.

In this presentation, we show how the constraint solver tool could be used to automatically analyze distance bounding protocols:

1. We first formulate a new trace property called Secure Distance Bounding (SDB) that protocol executions must satisfy; 2. We then classify the scenarios in which these protocols can operate considering the (dis)honesty of nodes and location of the attacker in the network; 3. Finally, we extend the constraint solver so that it can be used to test protocols for violations of SDB in these scenarios and illustrate our technique on some published protocols (on-line demo at http://homepages.dsu.edu/malladis/research/ConSolv/Webpage/).

12) A Logic of Secure Systems and its Application to Trusted Computing
Anupam Datta, Jason Franklin, Deepak Garg, Dilsun Kaynar (Carnegie Mellon University)

We present a logic for reasoning about properties of secure systems. The logic is built around a concurrent programming language with constructs for modeling machines with shared memory, a simple form of access control on memory, machine resets, cryptographic operations, network communication, and dynamically loading and executing unknown (and potentially untrusted) code. The adversary's capabilities are constrained by the system interface as defined in the programming model (leading to the name CSI-ADVERSARY). We develop a sound proof system for reasoning about programs without explicitly reasoning about adversary actions. We use the logic to characterize trusted computing primitives and prove code integrity and execution integrity properties of two remote attestation protocols. The proofs make precise assumptions needed for the security of these protocols and reveal an insecure interaction between the two protocols.

13) Computational Soundness for Key Exchange Protocols with Symmetric Encryption
Ralf Kuesters and Max Tuengerthal

In this talk, we present the first general computational soundness result for key exchange protocols with symmetric encryption. More specifically, we develop a symbolic, automatically checkable criterion, based on observational equivalence, and show that a key exchange protocol that satisfies this criterion realizes a key exchange functionality in the sense of universal composability. Our results hold under standard cryptographic assumptions.

14) An Epistemic Approach to Coercion-Resistance for Electronic Voting Protocols
Ralf Kuesters and Tomasz Truderung

Coercion resistance is an important and one of the most intricate security requirements of electronic voting protocols. Several definitions of coercion resistance have been proposed in the literature, including definitions based on symbolic models. However, existing definitions in such models are rather restricted in their scope and quite complex.

In this talk, we report about a new definition of coercion resistance in a symbolic setting, based on an epistemic approach, that we have proposed recently. Our definition is relatively simple and intuitive. It allows for a fine-grained formulation of coercion resistance and can be stated independently of a specific, symbolic protocol and adversary model. As a proof of concept, we apply our definition to three voting protocols. In particular, we carry out the first rigorous analysis of the recently proposed Civitas system. We precisely identify those conditions under which this system guarantees coercion resistance or fails to be coercion resistant. We also analyze protocols proposed by Lee et al. and Okamoto.

15) Implicit flows in nonmalicious code

Alejandro Russo (Chalmers)
Andrei Sabelfeld (Chalmers)
Keqin Li (SAP)

Information-flow technology is a promising approach for ensuring security by design and construction. However, recent experiments indicate that information-flow technology has run into a practical obstacle: false alarms. These alarms result from implicit flows, i.e., flows through control flow when computation branches on secret data and performs publicly observed side effects depending on which branch is taken.

The large body of literature exercises two extreme views on implicit flows: either track them (striving to show that there are no leaks, and often running into the problem of false alarms), or not track them (which reduces false alarms, but all bets are often off in terms of security guarantees).

This short talk explores a middle ground between the two extremes. We observe that implicit flows are often harmless in nonmalicious code: they cannot be exploited to efficiently leak secrets. To this end, we are able to guarantee strong information-flow properties with a combination of an explicit-flow and a graph-pattern analyses. Case studies on secure logging and cross-site scripting prevention suggest that our approach offers a desired combination of a lightweight analysis, strong security guarantees, and no excessive false alarms.

16) A Proof-Carrying File System

Deepak Garg (CMU)
Frank Pfenning (CMU)

There is a significant mismatch in the complexity of file access policies prevalent in large organizations like intelligence and military establishments, and the sophistication of mechanisms currently available for their enforcement. Policies often rely on high level motifs like delegation of rights, time-based expiration of credentials, and attributes of individuals and files, whereas the only available mechanism for enforcing these policies in file systems today is access control lists. Translating the intent of complex policy rules to these low level lists, and keeping the latter up-to-date with respect to changing credentials requires substantial and continuous manual effort and is a source of many policy enforcement errors.

Based on these considerations, we have recently designed and implemented a file system, PCFS, that uses a combination of proof-carrying authorization (PCA) and conditional cryptographic capabilities to rigorously enforce policies written in a logic, where high-level motifs can be represented directly. Our design makes two technically challenging, foundational contributions that may have practical implications even beyond PCFS. First, we introduce a new authorization logic, BL, which allows representation of policies depending on both explicit time (e.g., time-bound credentials), and untracked system state (e.g., file meta data). The proof theory of BL delicately balances expressiveness and the possibility of practical implementation in tools like the automatic theorem prover that we have built for PCFS.

Second, we develop an efficient mechanism for end-to-end enforcement of BL's rich policies. As opposed to PCA, which requires that proofs authorizing access be verified at each system call, our architecture delegates proof verification to a trusted program that is outside the system interface, and invoked in advance of file access. This trusted program returns cryptographic capabilities which authorize access to files. Capabilities can be checked 100-1000 times faster than proofs, thus allowing a very high throughput in the file system backend. The foundationally interesting idea here is that, in order to prevent access through capabilities derived from stale proofs, all time and state dependencies of a proof must be carried into capabilities generated from it. Capabilities are therefore conditional on such dependencies, and we prove formally that access due to stale proofs is impossible in our architecture.

In ongoing work we are considering capabilities with new types of conditions to enforce certificate revocation as well as credentials that can be used once only. At the policy level, the latter may be represented using ideas from linear logic.

17) Improving JavaScript Isolation using Rewriting Techniques

Sergio Maffeis (Imperial College)
Ankur Taly (Stanford University)

In this talk we complement the presentation of the paper "Language-Based Isolation of Untrusted JavaScript" with a survey of further work by the same authors on rewriting techniques for JavaScript. We present our rewriting techniques, discuss the expressiveness of the resulting JavaScript subset and its relation to Facebook FBJS.