4 Can We Use Platform Information Remotely?

Section 2 described mechanisms for accumulating measurements of software state. In this section, we treat the issue of conveying these measurement chains to an external entity in an authentic manner. We refer to this process as *attestation*, though some works use the phrase *outbound authentication*. We also discuss privacy concerns and mitigation strategies that arise when sharing this information with third parties.

4.1 Prerequisites

The *secure boot* model (Section 3.1) does not capture enough information to securely inform a remote party about the current state of a computer, since it (at best), informs the remote party that the platform booted into some “authorized” state, but does not capture which state that happens to be, nor which values were considered during the authorization boot process.

Instead, a remote party would like to learn about the measurement of the currently executing code, as well as any code that could have affected the security of this code. Section 2 describes how a *trusted boot* process securely records this information in measurement chains (using either certificates or hashes).

4.2 Conveying Code Measurement Chains

The high-level goal is to convince a remote party (hereafter: verifier) that a particular measurement chain represents the software state of a remote device (hereafter: attestation). Only with an authentic measurement chain can the verifier make a trust decision regarding the attestation. A verifier’s trust in an attestation’s measurement chain builds from a hardware root of trust (Section 6). Thus, prerequisites for attestation are that the verifier (1) understands the attestation’s hardware configuration and (2) is in possession of an authentic public key bound to the hardware root of trust.

The attestation’s hardware configuration is likely represented by a certificate from its manufacturer, e.g., the IBM 4758’s factory Layer 1 certificate [182], or the TPM’s Endorsement, Platform, and Conformance Credentials [213]. Attestation-specific mechanisms for conveying public keys in an authentic way are treated with respect to privacy issues in Section 4.3. Otherwise, standard mechanisms (such as a Public Key Infrastructure) for distributing authentic public keys apply.

The process of actually conveying an authenticated measurement chain varies depending on the hardware root of trust. We first discuss a more general and more powerful approach to attestation used on general-purpose secure coprocessors such as the IBM 4758 family of devices. Then, given the prevalence of TPM-equipped platforms today, we discuss attestation as it applies to the TPM.
4.2.1 General Purpose Coprocessor-Based Attestation

Smith discusses the need for coprocessor applications to be able to authenticate themselves to remote parties [182]. This is to be distinguished from merely configuring the coprocessor as desired prior to deployment, or including a signed statement about the configuration. Rather, the code entity itself should be able to generate and maintain authenticated key pairs and communicate securely with any party on the Internet. Smith details the decision to keep a private key in tamper-protected memory and have some authority generate certificates about the corresponding public key. As these coprocessors are expensive devices intended for use in high assurance applications, considerably less attention has been given to the device identity’s impact on privacy.

Naming code entities on a coprocessor is itself an interesting challenge. For example, an entity may go through one or more upgrades, and it may depend on lower layer software that may also be subject to upgrades. Thus, preserving desired security properties for code and data (e.g., integrity, authenticity, and secrecy) may depend not only on the versions of software currently running on the coprocessor, but also on past and even future versions. The IBM 4758 exposes these notions as configurations and epochs, where configuration changes are secret-preserving and epoch changes wipe all secrets from the device.

During a configuration change, certificate chains incorporating historical data are maintained. For example, the chain may contain a certificate stating the version of the lowest layer software that originally shipped on the device, along with a certificate for each incremental upgrade. Thus, when a remote party interacts with one of these devices, all information is available about the software and data contained within.

This model is a relative strength of general-purpose cryptographic coprocessors. TPM-based attestations (discussed in the next section) are based on hash chains accumulated for no longer than the most recent boot cycle. The history of software that has handled a given piece of sensitive data is not automatically maintained.

Smith examines in detail the design space for attestation, some of which is specific to the IBM 4758, but much of which is more generally applicable [182]. A noteworthy contribution not discussed here is a logic-based analysis of attestation. See Section 8 for additional work on validating the bootstrapping of trust.

4.2.2 TPM-Based Attestation

TPM-based attestation affords less flexibility than general coprocessor-based attestation, since the TPM is not capable of general-purpose computation. During the attestation protocol (shown in Figure 4), software on the attestor’s computer is responsible for relaying information between the remote verifier and the TPM [213]. The protocol assumes that the attestor’s TPM has generated an Attestation Identity Keypair (AIK), which is an asymmetric keypair whose public component must be