ArmoredSoftware Architecture

Perry Alexander   Andy Gill   Prasad Kuklarni
Leon Searl
Information and Telecommunication Technology Center
The University of Kansas
{palexand,andygill,prasadk,lsearl}@ku.edu

November 9, 2013

Contents

List of Figures

List of Tables

Abstract

This document describes the evolving ArmoredSoftware architecture.

1 Introduction

The objective of ARMOREDSOFTWARE is to *provide a portable trusted computing capsule for applications executing in the cloud.* This capsule, referred to as *armor*, provides three major functions:

Appraisal – Request and assess measurement information from the operational environment and other armored components.

Measurement – Gather run-time measurement information from its application

Attestation – Assemble and deliver evidence to appraisers in a manner that assures measurement integrity

Figure 1 graphically depicts the major architectural components of a protected application. The *application* is the application to be protected by the infrastructure. The *measurement* component performs measurement operations on the running application while the *attestation* component gathers measurements and delivers them with cryptographic assurance of integrity and confidentiality. The *appraisal* component requests information from the environment and other components to assess the overall operational environment. *Access control* governs access to all critical resources in the protected application to assure secrets are preserved and enforce information flow restrictions.

Figure 2 graphically represents the interaction among protected components while figure 3 shows the sequencing of interactions during appraisal. A component's appraisal module will request information from a second component's attestation module. The attestation model will select an attestation protocol that instructs the measurer what information to gather and in what sequence. The attestation module packages measurement results in a evidence packet that is return to the appraiser requesting information. The appraiser then assesses cryptographic signatures and encryption to determine the trustworthiness of the measurements, then assesses measurements to determine the trustworthiness of the component being appraised. Note that the same process occurs when appraising the component's operational environment.
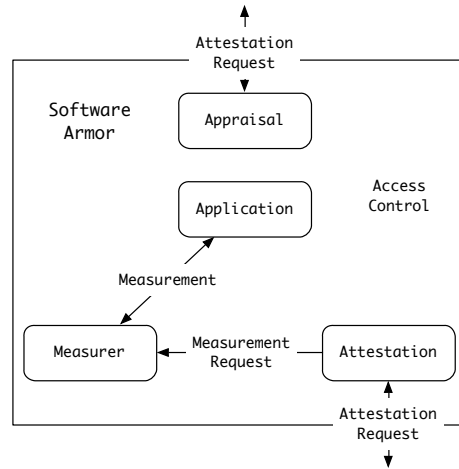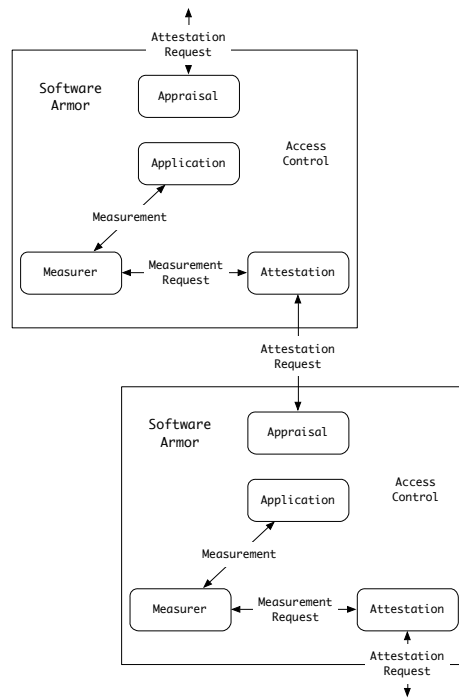
Figure 1: AS Architecture
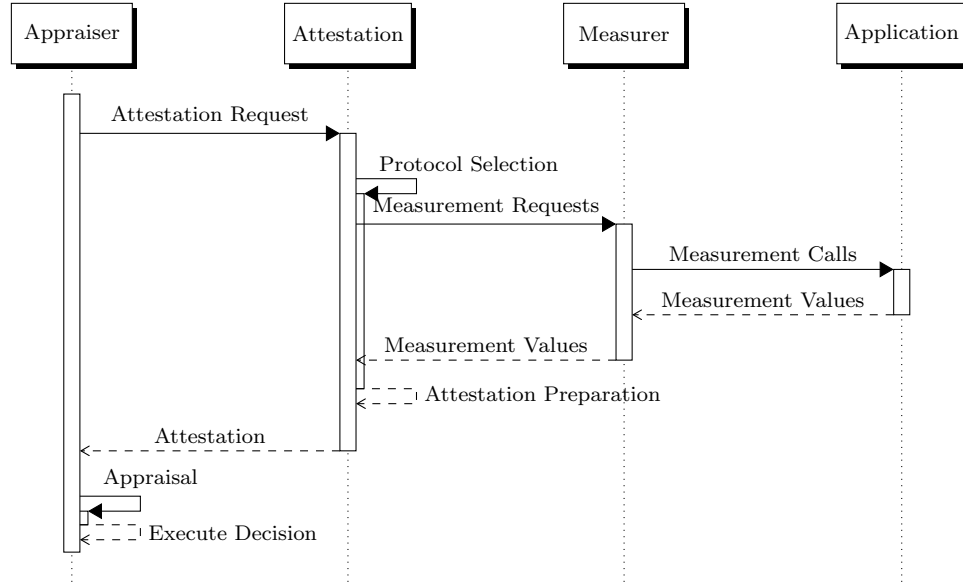

Figure 2: System Architecture

Figure 3: Architecture component interaction

2 System Architecture

2.1 Measurement

2.2 Attestation

Figure 4 graphically shows the processing of an attestation request by an Ar-
moredSoftware attestation component. The attestation request specifies in-
formation requested by an appraiser. Upon receipt, the *Attestation Protocol
Selector* or *AP Selector* identifies one or more *Attestation Protocols* that could
satisfy the appraisers request. The protocol is passed to the *AP Instantiation*
process that selects specific mechanisms for achieving individual requests in the
AP. The resulting protocol instance is passed to *AP Execution* where it is ex-
ecuted by: (i) making requests to the component's vTPM; making requests to
another components attestation service; or (iii) invoking the measurer on the
component's associated application.

The results of request to the vTPM, another attestation component, and the
local measurer are returned as vTPM quotes, evidence packages, and measure-
ments respectively. The AP Execution component monitors execution and col-
lects various results for processing by the AP Instantiation component. AP In-
stantiation assembles individual measurement, vTPM and appraisal results into
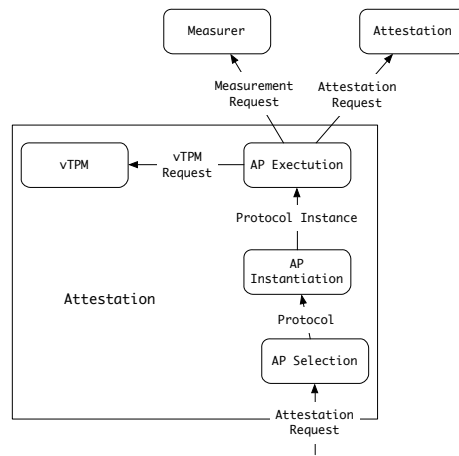a package representing information requested in the attestation request. The

Figure 4: Attestation request processing

AP Selection component uses cryptographic techniques to provide assurances to the appraiser requesting information that evidence can be trusted. Finally, the evidence package is returned to the requesting appraiser where it is evaluated.
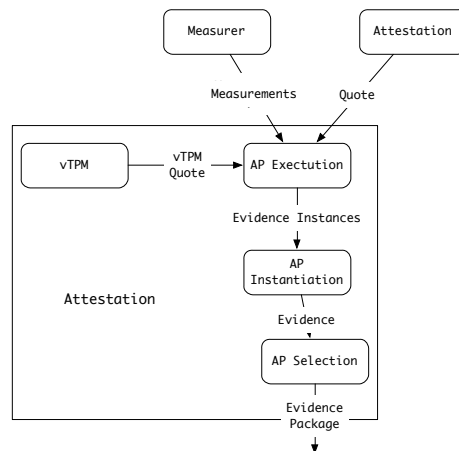

Figure 5: Attestation evidence processing

2.2.1 TPM and vTPM

2.3 Appraisal

2.3.1 Migration

2.4 Access Control

3 Component Interaction