Learn the architecture - Introduction to security

Version 1.0

Non-Confidential

Issue 02

102406_0100_02_en

Learn the architecture - Introduction to security

Release information

Document history

Issue	Date	Confidentiality	Change
0100-02	8 January 2020	Non-Confidential	First release

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm® welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Contents

1. Overview

Arm provides secure compute platforms for a range of security-focused applications. This guide introduces some generic concepts about security.

This guide is a primer for some other guides in this series, and provides context for those guides that discuss specific security-related technologies.

This guide is a high-level introduction only. You can find third-party books and guides on this subject if you want to learn more.

At the end of this guide you will be able to:

- Define confidentiality, integrity, and authenticity, and give practical examples
- Explain the categories into which attacks are split
- Understand how different techniques are used to protect different aspects of a security system

2. What do we mean by security?

When we talk about security, we use the term asset to refer to a thing that we want to protect. For example, an encryption key is often classed as an asset. Let's consider the properties of an asset that we might want to protect.

When we consider security, we should ask ourselves some questions:

- What thing do I want to protect?
- What is it about that thing that I want to protect?
- Who or what am I protecting that thing from?
- What is the value of the thing that is being protected?

Confidentiality

Confidentiality refers to who can see, access, or use an asset. We want to control access to the asset and ensure that unauthorized access is prevented. The most common mechanisms to achieve confidentiality are cryptography and access control.

The contents of your smartphone provide an example of confidentiality protection. Many smartphones have an encrypted file system to protect the data on your phone if you lose it. The cryptographic key that is used to decrypt the file system is only available after a user has authenticated themselves. In this example, cryptography is used to protect the confidentiality of the data in the filesystem. Access control is used to protect the confidentiality of the key that is needed to read the filesystem.

Integrity

Integrity refers to protecting the accuracy and completeness of an asset. Integrity protects the asset from unauthorized modifications or detects whether modifications have occurred.

For example, consider a device that includes the public key of an update server. This public key is not a secret, so we might not care about ensuring its confidentiality. However, we do need to ensure that the key is not replaced or tampered with. Replacing the key would allow an attacker to make a malicious update that appeared to be a genuine update. Corrupting the key might prevent the system from installing genuine updates. This corruption would leave the system open to known attacks.

Authenticity

Authenticity refers to verification that an asset is what it claims to be.

Let's use again the example of a device that includes the public key of an update server. When a device downloads a software update from an update server, we need to ensure that the update is authentic. The software provider signs the update with its private key, and the device can verify the signature using its copy of the public key. By ensuring the integrity of the stored public key, the device can check the authenticity of the software update. If an attacker tries to substitute the update, we can detect this and prevent installation.

Availability

Availability means that authorized users can access and use the asset in the ways that it is intended to be used.

A denial-of-service attack is a common method of denying availability. Typically, a denial-of-service attack involves flooding a system with false requests or data. For example, a denial-of-service attack might send large numbers of requests to a website, hoping to make the website so busy that it cannot service its genuine users.

Availability is about ensuring that, even if attacked, the user can still access and use an asset.

3. Attacking a system

In this guide and related guides, we use the following terms:

- Vulnerability: An underlying bug or weakness in a system
- Exploit: A specific attack that exploits a vulnerability
- Attacker: Also called an adversary, someone trying to compromise an asset
- Threat: A combination of an attacker, an exploit, and one or more assets that are being attacked

Types of attack

We classify attacks into three broad categories:

- Software attacks: A software attack is an attack that does not require physical access to the device. A software attack could include, for example, a malicious download from an app store or an email with a malicious attachment. Software attacks can target large numbers of devices anywhere in the world. For example, because the cost of sending an email is very low, an attacker can send an email with a malicious attachment to millions of recipients. Even if only a small percentage of those attacks succeed, the attack is worthwhile. Software attacks are either unprivileged attacks or privileged attacks. The malicious email that we described in our example is an unprivileged attack. An unprivileged attack is launched from user space and tries to exploit a vulnerability in the operating system to access assets that it should not be able to access. In a privileged attack, a privileged entity attacks another entity with similar privilege. For example, software in a system that is running as privileged on one processor can try to compromise another part of the system.
- Basic hardware attacks: A basic hardware attack requires physical access to the device, sometimes only temporarily. This type of attack is called basic because it requires tools, for example, JTAG probes or logic analyzers, that are easy to acquire. A basic hardware attack can involve software elements. The important difference between a software attack and a basic hardware attack is that a basic hardware attack requires physical access to a device. If physical access, or proximity, to a device is required, the number of potential target devices is greatly reduced. Reducing the number of targets significantly changes the threat profile. Note This type of attack was sometimes referred to as a shack attack. The name suggests that, for this type of attack, you only need tools that you can buy at a RadioShack store, rather than specialist tools. An example of a basic hardware attack is an evil maid attack. Imagine that you have left your laptop in a hotel room. Someone posing as a member of the hotel housekeeping staff could use the opportunity of access to your room to plug in a USB device.
- Advanced hardware attacks: Like a basic hardware attack, an attacker in an advanced hardware attack requires physical access to the device that is being attacked. The difference between a basic hardware attack and an advanced hardware attack is the level of equipment, knowledge, and time that is necessary to carry out the attack. Examples of advanced hardware attacks include placing a device under an electron microscope, or side-channel analysis. Side-channel analysis includes techniques like using the precise timing or power consumption of an operation to infer something about what the device is doing. Advanced hardware attacks are costly, in terms of money and time. This means that advanced hardware attacks need to be more focused than other types of attacks, and an attacker cannot normally target as many devices. These categories are useful for discussions about attacks, but are not fixed categories.

Attackers use a combination of attack types, or an attack might include elements from more than one category.

Different attackers

In addition to assets and attacks, we should think about who is attacking the system.

It is easy to imagine an attacker as someone in a dark room tapping at the keyboard of a laptop. But like we did with assets and attacks, we can categorize attackers.

Some attackers will be third parties. One example is the malicious email attachment example that we mentioned in Software attacks.

At other times, we protect the system from the user. For example, the purpose of Digital Rights Management (DRM) is to protect media from the owner of the device that includes the protected media.

Threat models and putting it all together

Threat models are a way of putting together all the ideas that we have discussed so far:

- What are the assets and what about them do we wish to protect?
- Who are the attackers and how can they attack the system?

Our threat model should also consider the different combinations of attack and attacker. This analysis is critical because it drives decisions on what defense we might employ.

4. Different security solutions

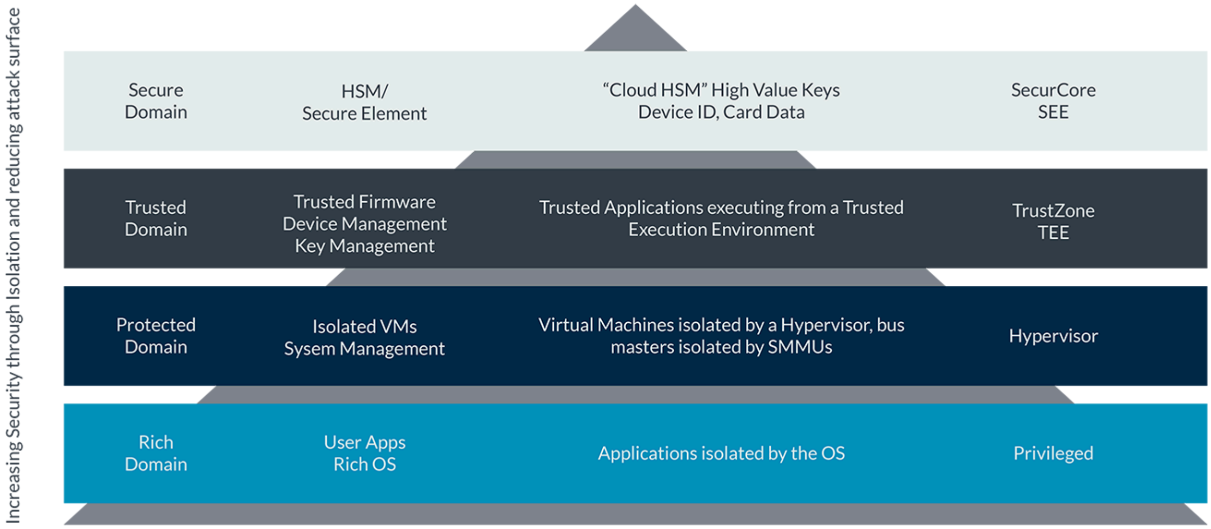
On the basis that any security can be broken if an attacker has enough time and money, we should not describe the security requirements for a design as impossible to bypass. Instead, we describe security requirements in value terms: Attack A on asset B should take at least Y days and Z dollars.If a set of countermeasures mean that a successful attack will take too long or will cost too much, then the defense is a success. In this situation, most attackers will move on to a different target.

This means that we need to balance factors including:

- Value of asset: What value does the asset have to us, and what value does it have to a potential attacker? Do we value different aspects of an asset differently? For example, we might want to protect the confidentiality and integrity of an asset, but we might value confidentiality more highly than integrity.
- Cost of defense: There are typically several ways that an asset can be defended. Which defense we choose will depend on the threat model. We consider the value of what is being protected, the likelihood of the attack, and the cost of the defense.
- Practicality and usability: Passwords are an example of the practicality and usability problem. Users are advised to use long and complex passwords and to use a unique password for each system. Although long, complex, and unique passwords are better for security, many users will not remember and be able to manage them. This means that solutions must balance security needs with user needs.

This diagram shows different security features in the Arm architecture. These features can be used to defend different assets in the system:

Figure 4-1: Different security features in the Arm architecture


Increasing Security through Isolation and reducing attack surface

Secure Domain	HSM/ Secure Element	"Cloud HSM" High Value Keys Device ID, Card Data	SecurCore SEE
Trusted Domain	Trusted Firmware Device Management Key Management	Trusted Applications executing from a Trusted Execution Environment	TrustZone TEE
Protected Domain	Isolated VMs Sysem Management	Virtual Machines isolated by a Hypervisor, bus masters isolated by SMMUs	Hypervisor
Rich Domain	User Apps Rich OS	Applications isolated by the OS	Privileged

The technologies that are highlighted in the diagram provide different levels of protection and at varying levels of cost. System designers identify which technology is most appropriate for protecting particular assets.

Other guides in this series will look at these technologies in more detail.

5. Related information

Here are some resources related to material in this guide:

- Arm architecture and reference manuals - Find technical manuals and documentation relating to this guide and other similar topics.
- Arm Community - Ask development
- Arm Security Manifesto - Find information on security threats and how our engineers are responding.
- Platform Security Architecture guides - Learn about particular use cases and show how a threat analysis is carried out. One example is this guide on smart locks.

6. Next steps

This guide provided a high-level introduction to security concepts. We have learned about security basics, and can now explore specific security-related technologies in the following guides:

- TrustZone for Armv8-A - This guide introduces and explains the Arm TrustZone technology. Arm TrustZone technology offers an efficient, system-wide approach to security with hardware-enforced isolation that is built into the CPU.
- Providing protection for complex software - This guide explores the features that are available in the Arm architecture to provide robust protection against common software attacks including pointer authentication, branch target identification and memory tagging. The guide introduces the features that are required to mitigate against Jump-Orientated Programming and Return-Orientated Programming attacks.