Medical Infrastructure Supply Chain (MISC)
System Design

MITRE eCTF 2024
Team **Cacti**
University at Buffalo

1 Introduction

The Medical Infrastructure Supply Chain (MISC) system simulates a supply chain security solution for microcontrollers on a medical device. Our design consists of three parts: host computer, Application Processor (AP), and Component (CP). The MISC system includes one AP and two CPs connected via an I2C bus. The host computer is a general-purpose computer communicating with the AP over a serial interface. The host tools in the host computer send commands to and receive messages from the AP.

The following summarizes the features of the MISC system:

- The AP can list provisioned and presented CP IDs.
- The whole system will only boot up when all the provisioned CPs are present and valid.
- The AP can retrieve attestation data from a CP with the correct PIN code.
- After replacing a CP, the AP needs to be notified of the change of a provisioned CP ID.
- The integrity and authenticity of communication messages between AP and CP are protected.

2 Security Requirements

This section defines the security requirements of our design.

The AP and CPs must be *valid* (built by the organizers) for the MISC system to work properly.

In the sections below, the term *nonce* refers to a random number.

The term *purpose label* is a 1-byte number that marks the purpose of the message, for example, booting, communicating, and attesting have different numbers.

2.1 SR1 & SR2

The Application Processor (AP) should only boot if all expected CPs are present and valid.

CPs should only boot after being commanded to by a valid AP that has confirmed the integrity of the device.

How we address it: The AP needs to verify all the present CPs have provisioned IDs and that they are valid CPs built by the organizers. We use the public-key cryptography algorithm and challenge-response authentication. There are two key pairs, the AP holds the first private key and the second public key, while the CP holds the first public key and the second private key.

When the host computer sends the boot command to the AP, the AP sends the boot command and a generated nonce (challenge) to the CP. The CP makes up a response by signing the booting purpose label, nonce, and CP ID with the second private key. Then, it generates another nonce and replies the response and the new nonce to the AP. The AP verifies the reply with the second public key. The booting purpose label, nonce, and CP ID should be correct. After receiving the replies from all the provisioned CPs, the AP makes up the responses for each CP using the same signing method mentioned before with the first private key. For a CP, it will verify the response from the AP using the first public key and boot up if verification passes.

The signing process includes the purpose label and CP ID to mitigate response reuse attacks (from other functionality or other CPs).

2.2 SR3

The Attestation PIN and Replacement Token should be kept confidential.

How we address it: The Attestation PIN and Replacement Token will not be stored in plain text or hard-coded in the program (e.g., use the macro generated at the build phase). The PIN and the Token will be hashed using a keyed-hash algorithm during the building process, and the PIN and Replacement Token macros will be replaced with the two hash strings. AP holds the hash key. After receiving a PIN or a token, the AP will apply SHA1 to the input, and compare the hash string with the saved one. The comparison time is constant regardless of the content of the PIN or Replacement Token to mitigate a side-channel attack. To mitigate a brute force attack, the AP will delay randomly up to 5 seconds after receiving a wrong PIN or token.

2.3 SR4

CP Attestation Data should be kept confidential.

How we address it: Use AES to encrypt the CP Attestation Data with a Python script to replace the plain text with the encrypted text in macros when building a CP. The AES key is generated based on a shared secret between AP and CP. A CP does not have the decryption process during runtime. When requesting the attestation data, a CP puts the encrypted text in the I2C buffer. The AP reads the encrypted text, decrypts it, and sends it to the host computer.

A CP needs to verify if the Attestation Data reading request message is from the valid AP. RSA algorithm and challenge-response mechanism are used for the verification process. The AP holds the private key, and the CP holds the public key. The AP sends a request message to the CP The CP generates a nonce and sends it to the AP as the challenge. The AP signs attesting purpose label, the challenge, and the CP ID using the private key and sends it back to the CP as the response. The CP verifies the response using the public key and checks if the purpose label, challenge, and CP ID are correct.

The signing process includes the purpose label and CP ID to mitigate response reuse attacks (from other functionality or other CPs).

2.4 SR5

The integrity and authenticity of messages sent and received using the post-boot MISC secure communications functionality should be ensured.

How we address it: With the public-key cryptography algorithm, we use the challenge-response authentication for message authenticity and digital signature for message integrity. There are two key pairs, the AP holds the first private key and the second public key, the CP holds the first public key and the second private key.

To send a message to a CP, the AP sends the sending message command to the CP. The CP generates a nonce as the challenge for the AP, and the AP signs the reading purpose label, nonce, and CP ID using the first private key as the response. The AP also signs the message with the same key. The AP sends the response, message, and the message signature to the CP. The CP verifies the response and the message signature by using the first public key. It receives the message if validations pass.

To read a message from a CP, the AP sends the reading command and a nonce challenge to the CP. The CP signs the challenge with the second private key as the response, and puts the response, message, and the signature of the message (with the second private key) to the I2C buffer waiting for the AP to read. The AP reads the response, message, and the signature. It verifies the response and the message signature using the second public key.

The signing process includes the purpose label and CP ID to mitigate response reuse attacks (from other functionality or other CPs).

3 Security Implementations

3.1 Build MISC System

3.1.1 Build Deployment

Attackers will NEVER be given access to the Deployment Secrets generated in this step.

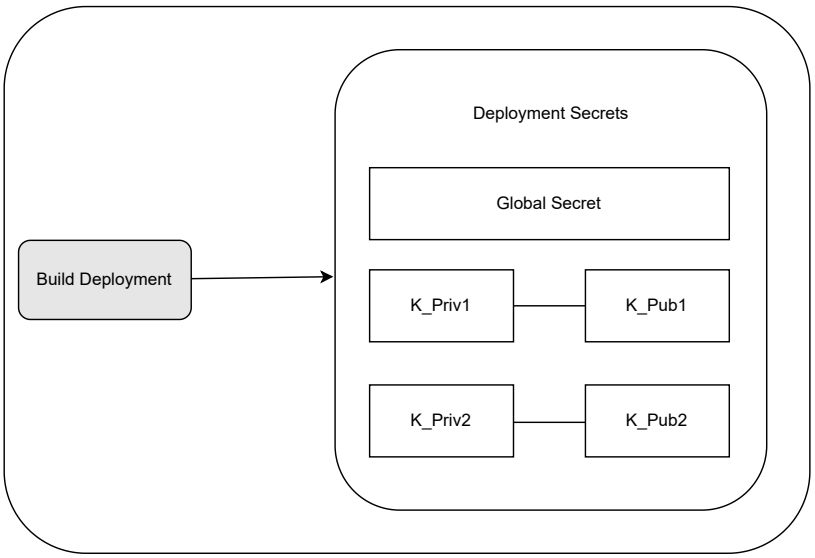

Figure 1: Deployment Secret

Figure 1 shows the Deployment Secrets. During the building deployment process, a Global Secret and two pairs of public and private keys will be generated. The Global Secret will be shared among the AP and all the CPs. AP will store the first private key of `K_Pri1` and the second public key of `K_Pub2`. CP will store the second private key of `K_Pri2` and the first public key of `K_Pub1`. We generate the Deployment Secrets randomly when building deployment to prevent the attacker from retrieving it through the source code.

3.1.2 Build AP and CP Firmware

To build the AP firmware, the number of provisioned CPs, provisioned CP IDs, attestation PIN code, and replace Token need to be provided.

To build the CP firmware, the CP ID and attestation data need to be provided. The attestation data includes location, date, and customer name.

3.2 Load Devices and Booting

3.2.1 Load AP and CP Firmware

The firmware will be loaded by the provided host tool, teams are not allowed to modify this step.

3.2.2 First Booting

Devices will be initialized at the first boot. The AP will store provisioned CP IDs in plain text, the private key of `RSA1`, and the public key of `RSA2` in the flash memory. A hash key is generated based on the Global Secret for the keyed-hash algorithm. The PIN and replace Token will be keyed-hashed, and the AP stores the hash values in the flash. The CP stores the private key of `RSA2` and the public key of `RSA1`. An AES-128 key will be generated based on the Global Secret to encrypt the attestation data. The encrypted data will be stored in the flash.

3.3 Functionalities

The term *packet* refers to all the communication data either the AP sends to a CP, or the AP reads from a CP.

3.3.1 Listing

The Listing functionality checks the provisioned CP IDs of the APs and the IDs of all the presented CPs. As CP IDs are not secrets, there is nothing to protect during this process.

The process (shown in Figure 2) is described as follow:

1. The host computer sends the `list` command through USB to the AP.
2. The AP reads all the provisioned CP IDs from its flash memory and sends the IDs back to the host computer.
3. The AP sends the `scan` command to all the possible CP IDs one by one over the I2C bus.
4. For one specific ID, if the CP with this ID is presented, the CP puts the ID value in the I2C buffer for the AP to read.
5. The AP reads the ID value from this CP and sends the ID value to the host computer as one present CP ID.

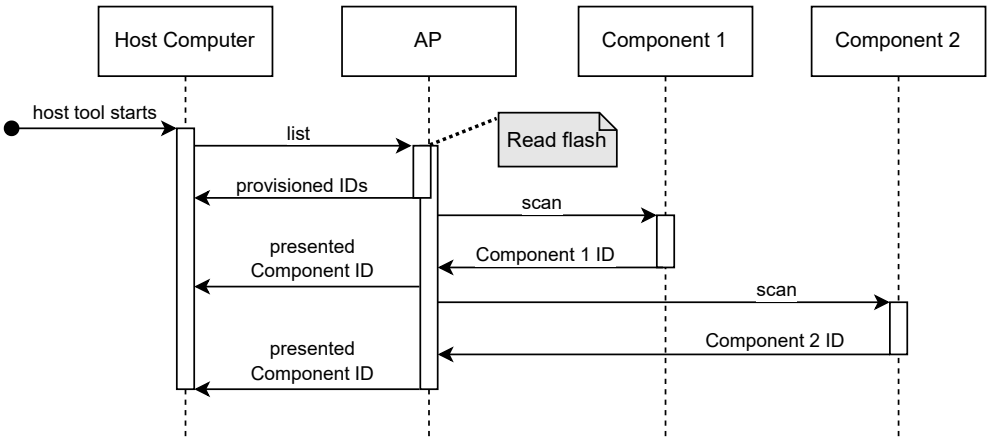

Figure 2: Listing Sequence

3.3.2 Boot

This functionality requires the AP to test if all the provisioned CPs are present and valid before the AP boots. The AP sends commands to all the CPs to let them boot, and the CPs must ensure that the boot command is from the valid AP before booting.

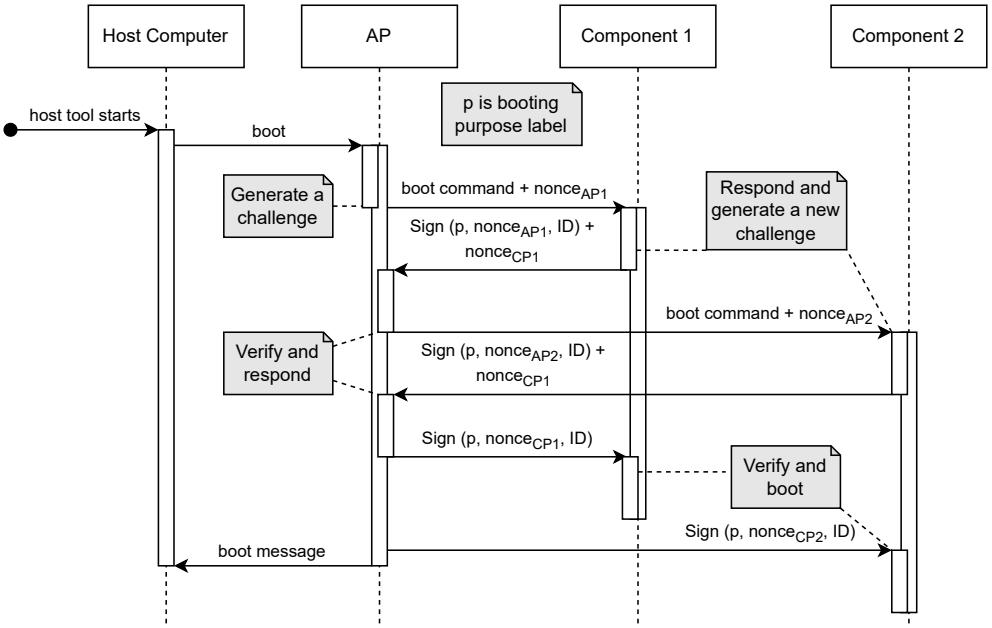

Figure 3: Boot Sequence

The process (shown in Figure 3) is described as follow:

1. The host computer sends the `boot` command to the AP.
2. The AP checks all the provisioned CPs one by one to make sure they are present and valid.
3. For each provisioned CP, the AP generates a nonce as the challenge. It then sends the `boot` command and the challenge to the corresponding CP.

4. After receiving the boot command and the challenge, the CP makes up the response by signing a boot purpose label (a 1-byte number marks the purpose is booting), the challenge, along with the CP ID using the `K_Pri2` key. It then creates a new challenge by generating a nonce and puts the response and new challenge to the I2C buffer waiting for the AP to read.

5. The AP reads the response from the CP and verifies the response using the `K_Pub2`. It checks if the purpose label is for booting, the nonce is the same as the originally generated one, and the CP ID are correct.

6. For a validation failure, the AP will randomly delay for up to 5 seconds to mitigate a brute force attack and terminate the booting process.

7. If all the CPs are present, the AP then tries to boot them all.

8. To boot a CP, the AP finds the nonce in the challenge sent by this CP before. Then sign the booting purpose label, nonce, and CP ID with the `K_Pri1` key as the response. The AP sends the response to this CP.

9. After receiving the response from the AP, the CP verifies the response by using the `K_Pub1`. The CP will boot up if verification passes.

3.3.3 Replace

The Replace functionality is for replacing a CP on the medical device. The technician in a repair station needs to tell the AP which CP ID are no longer provisioned and what the new CP ID with supplying the correct replacement Token. The replace Token will be hashed before comparing with the saved Token hash value.

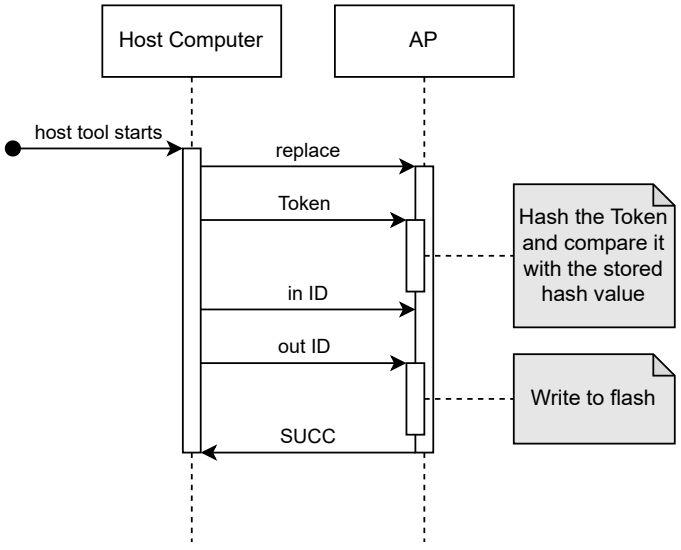

Figure 4: Replace Sequence

The process (shown in Figure 4) is described as follow:

1. The host computer sends the `replace` command to the AP.

2. The AP enters the `replace` mode and waits for the host computer to send the replace Token.

3. After receiving the Token, the AP applies the keyed-hash algorithm to the Token and compares it with the stored hash value. All the characters in the hash value are compared one by one. The comparison time is always constant no matter if the received Token is correct or not. If a wrong Token is received, the AP will randomly delay for up to 5 seconds to mitigate brute force and then terminate the replace mode. For a correct Token received, the AP will wait for the host computer to send the old (currently provisioned) and new IDs.

4. The AP modifies the flash memory after receiving the two IDs and sends a success message to the host computer. And the new ID becomes the provisioned ID.

3.3.4 Attestation

The functionality gets the attestation data from one specific CP by supplying the correct PIN code and the CP ID. The attestation data is sensitive and encrypted when storing, so decrypting is necessary when retrieving the data. Also, the PIN code will be hashed before comparing with the saved PIN code hash value.

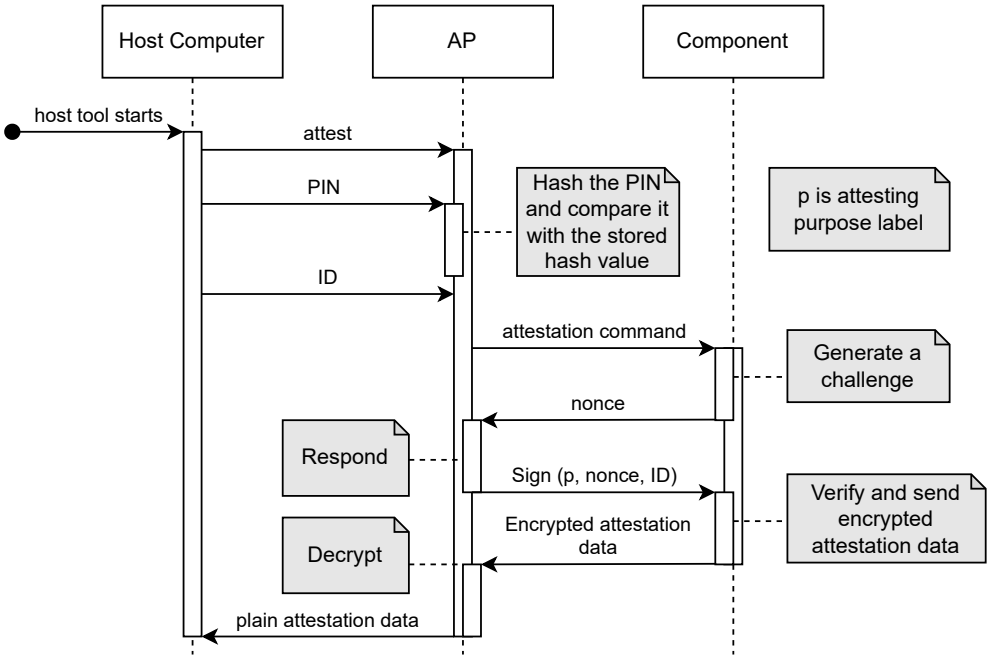

Figure 5: Attestation Sequence

The process (shown in Figure 5) is described as follow:

1. The host computer sends the `attest` command to the AP

2. The AP enters the `attestation` mode and waits for the host computer to send the PIN code

3. After receiving the PIN code, the AP applies the keyed-hash algorithm to the PIN and compares it with the stored one. The comparison time is always constant no matter the PIN correctness. The AP will randomly delay for up to 5 seconds to mitigate brute force and then terminate the attestation mode for a wrong PIN. When given a correct PIN, the AP tries to acquire the attestation data from the specific CP with the ID given by the host computer.

4. The AP sends the `attestation` command to the CP.

5. The CP generates a nonce and sends it as a challenge to the AP.

6. The AP signs the attesting purpose label, nonce, and the CP ID using the `K_Pri1` key and sending it as the response back to the CP.

7. The CP verifies the response with the `K_Pub1` key to check if the attesting purpose label, nonce, and the ID is correct. If validation fails, the CP sends the verification failure message to the AP, and the AP randomly delays up to 5 seconds to prevent brute force and terminate the attestation mode. Otherwise, the CP puts the encrypted attestation data in the I2C buffer waiting for the AP to read.

8. The AP decrypts the data using the AES algorithm and sends the plain attestation data to the host computer.

3.3.5 POST-BOOT Communication

After the AP and CPs boot up, the communication between the AP and a CP is through the I2C bus. The AP is the master device, while the CPs are slave devices. Thus, the AP can send messages to or read messages from a CP. However, the CP cannot send messages directly to the AP in normal situations. The integrity and authenticity of the communication are necessary.

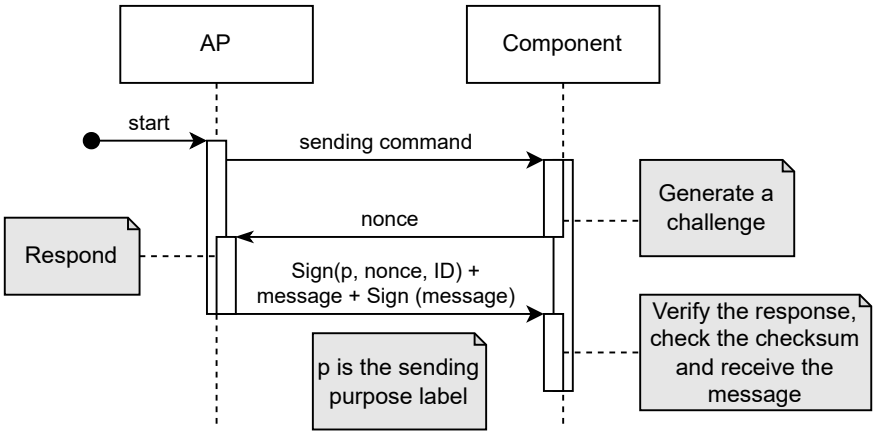

Figure 6: Sending Message Protocol Sequence

The sending message process is shown in Figure 6.

1. The AP sends the `sending` command to one specific CP.

2. The CP generates a nonce as the challenge for the AP.

3. The AP signs the sending purpose label, nonce, and the CP ID using the `K_Pri1` key as the response for the CP. The AP sends the response, message, and a signature of the message (signed using the `K_Pri1` key) to th CP.

4. The CP verifies using the `K_Pub1` key the response by checking if the reading purpose label, nonce, and ID is correct to ensure authenticity.

5. The CP then verifies the message signature by using the `K_Pub1` key to ensure the message integrity.

6. If the two verifications pass, the CP receives the message.

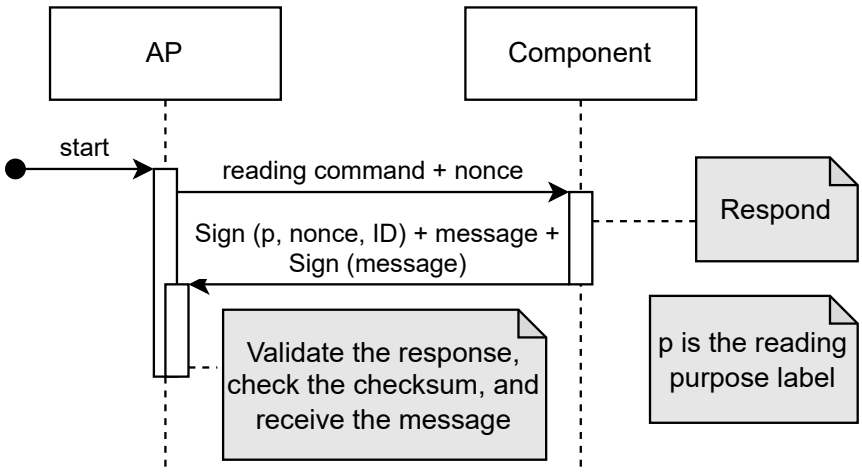

Figure 7: Reading Message Protocol Sequence

The reading message process is shown in Figure 7.

1. The AP sends the `reading` command and a generated nonce to one specific CP.
2. The CP signs the reading purpose label, nonce, and the ID with the `K_Pri2` key as the response. It puts the response, message, and a signature of the message signed using the `K_Pri2` key in the I2C buffer, waiting for the AP to read.
3. The AP reads the packet from the CP.
4. The AP verifies the response using the `K_Pub2` key to and check if the reading purpose label, nonce, and the CP ID are correct. It then verifies the message signature by using the `K_Pub2` to ensure the message integrity. If verifications pass, the AP accepts the message.