# 2018 MITRE Collegiate Embedded Capture-the-Flag

## Agenda

- **Welcome!**
- **Competition Prizes**
- **Design Phase Flags**
- **Q&A w/ special guest: Brian Marquis**
- **Hardware Attacks**
- **Questions?**
  - During this presentation, we will be monitoring our Slack channel for questions

## Welcome to the 2018 eCTF

- **ATM w/ Chip-and-PIN**
  - Has everyone seen the rules?
  - Has every team received their hardware at this point?

- **14 registered schools!**
  - Please send team member email addresses to ectf@mitre.org for Slack registration

MITRE

## Competition Schedule

- **Kickoff – January 17th, 2018**
  - Competition officially kicks off
- **System Handoff – March 1st, 2018**
  - System design and implementation is due
  - After MITRE has verified a submitted design, the designing team will be given access to all other verified designs for attack
- **Scoreboard Closes – April 13th, 2018**
  - Flag submission is closed
  - Teams will be contacted for write-ups, which will be due April 18th
- **Award Ceremony – April 19th, 2018**
  - The top scoring teams will be invited to MITRE to present their work at an award ceremony, where MITRE will announce the results of write-up judging and present awards

MITRE

## Prizes and Competition Qualification Requirements

- **This year we are adding $5000 in prizes for the winning teams**
  - 1st Place: $2000
  - 2nd Place: $1000
  - 3rd Place: $500

  - Special Awards ($1500)

  (Special awards may be split among multiple teams)

- **Anyone can compete in the eCTF, but to receive prize money you must meet certain eligibility requirements – check website Terms of Service**

- **Processes have been put in place to keep the competition fair**
  - Current and former MITRE employees and interns will be competing
    - Competition organizers are firewalled from MITRE participants – no discussions or hardware exchange allowed outside of official channels
  - All questions and requests for help are taken on a first-come-first-serve basis
  - Write-ups are anonymized before judging

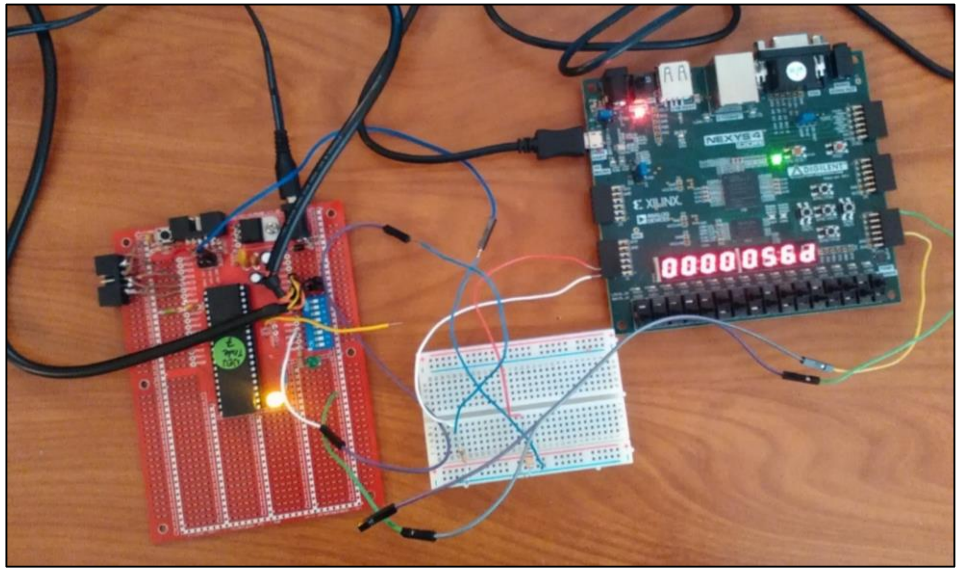
## Design Phase Flags

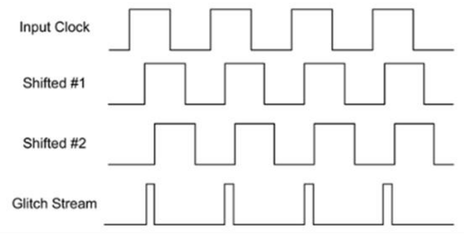
- **Designed to help keep teams on track during the Design Phase**
  - Additionally gives the organizers some insight into team's progress
  - Descriptions of each flag can be found in the challenge description document
- **PSoC firmware for flag capture will available soon (probably tomorrow)**
  - Only first three flags will be ready for capture
  - Check our website and Slack for links to the github page
- **Fourth (and final) flag challenge will be released shortly**
  - Please send us IP addresses of computers/networks that will need access to competition servers
    - Necessary for this flag AND the attack phase
  - (All school-owned IP blocks will be automatically added)

## Brian Marquis Q&A

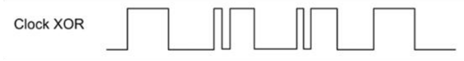
- **Member of 2017 UConn Team (“Firmware Dogs”)**
  - *First Place Overall for MITRE 2017 Collegiate eCTF*

- **Hired by MITRE as Embedded Security Engineer in June 2017**

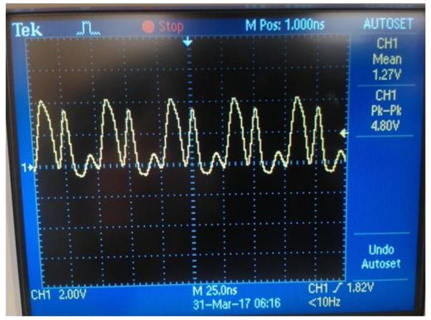




Example:



Source: www.newae.com



Worked on jmp and rjmp instructions. Not rcall or call though …

## Brian Marquis Q&A

- **What one thing do you wish you had known before the attack phase started?**

- **How did you approach the problem for defense and attack?**

- **What was your development process like?**
  - Did you set a schedule? (If so, were you able to stick to it?)

- **In your opinion, what is the most important defensive measure?**

## Hardware Attacks

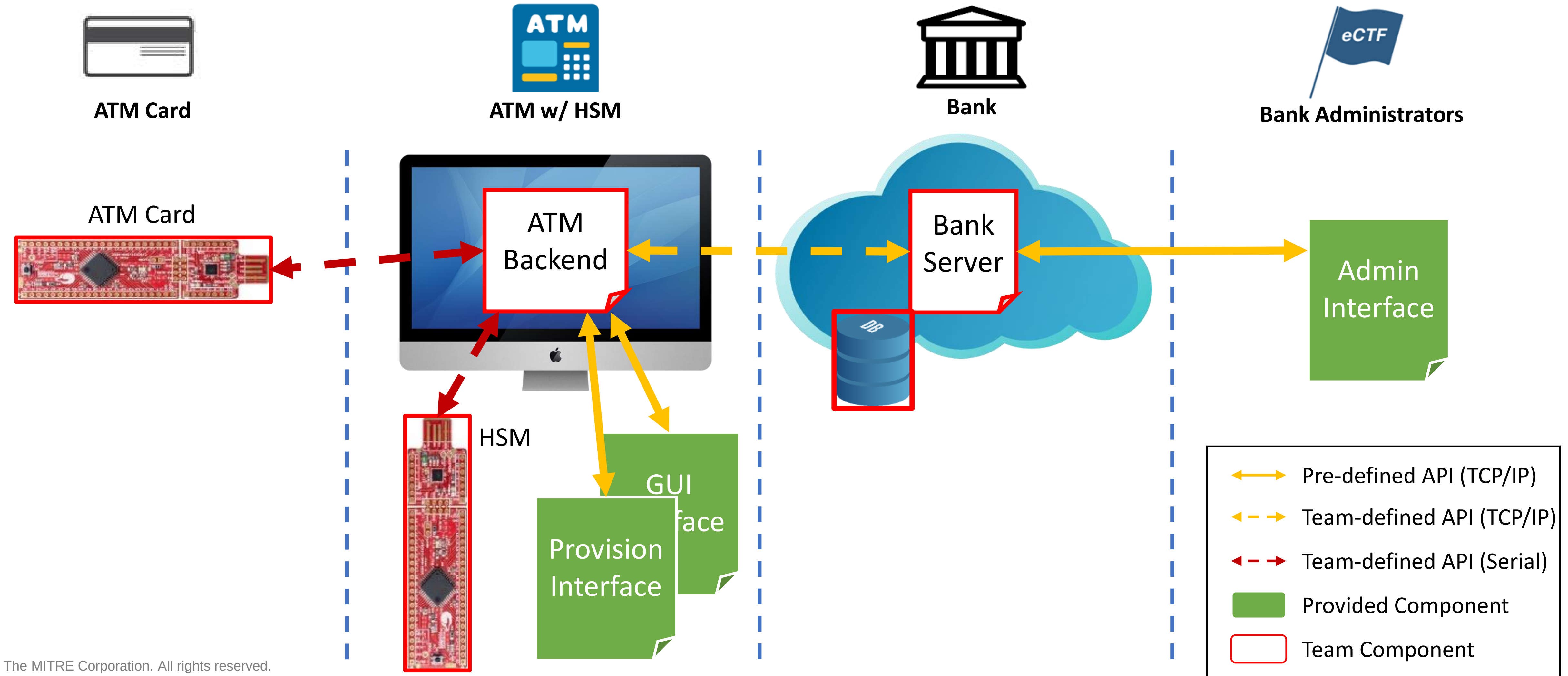
- **We recommend that you look into various embedded hardware attack vectors during the design phase**
  - Understand the possible attacks that need to be defended against
  - Test your design for security before handoff
  - Quicker flag capture during the attack phase

- **Classes of hardware attacks**
  - Code exploitation
  - Protocol Attacks
  - Fault Injection Attacks
  - Side-channel Attacks

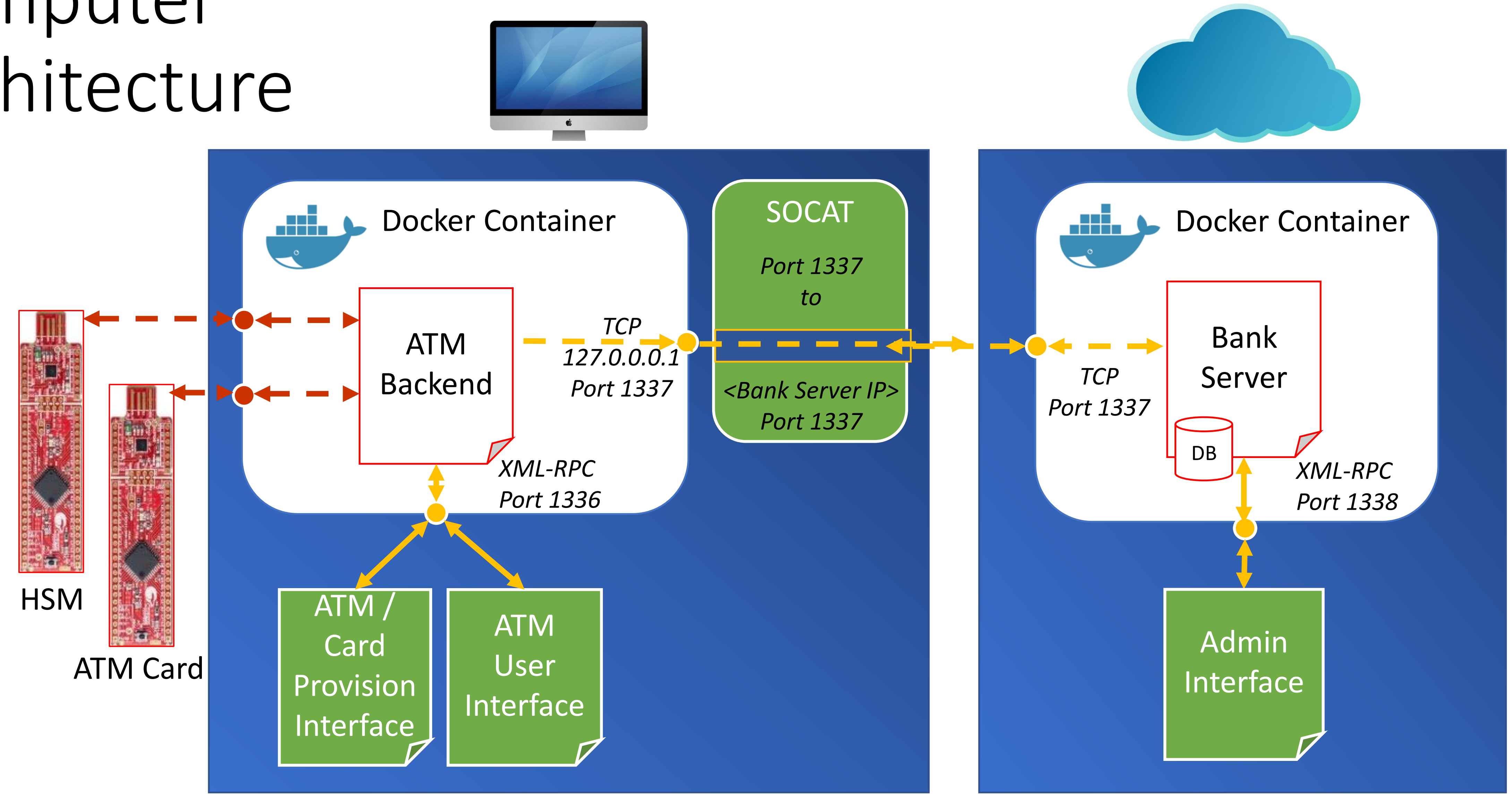
## Questions?

## Backup

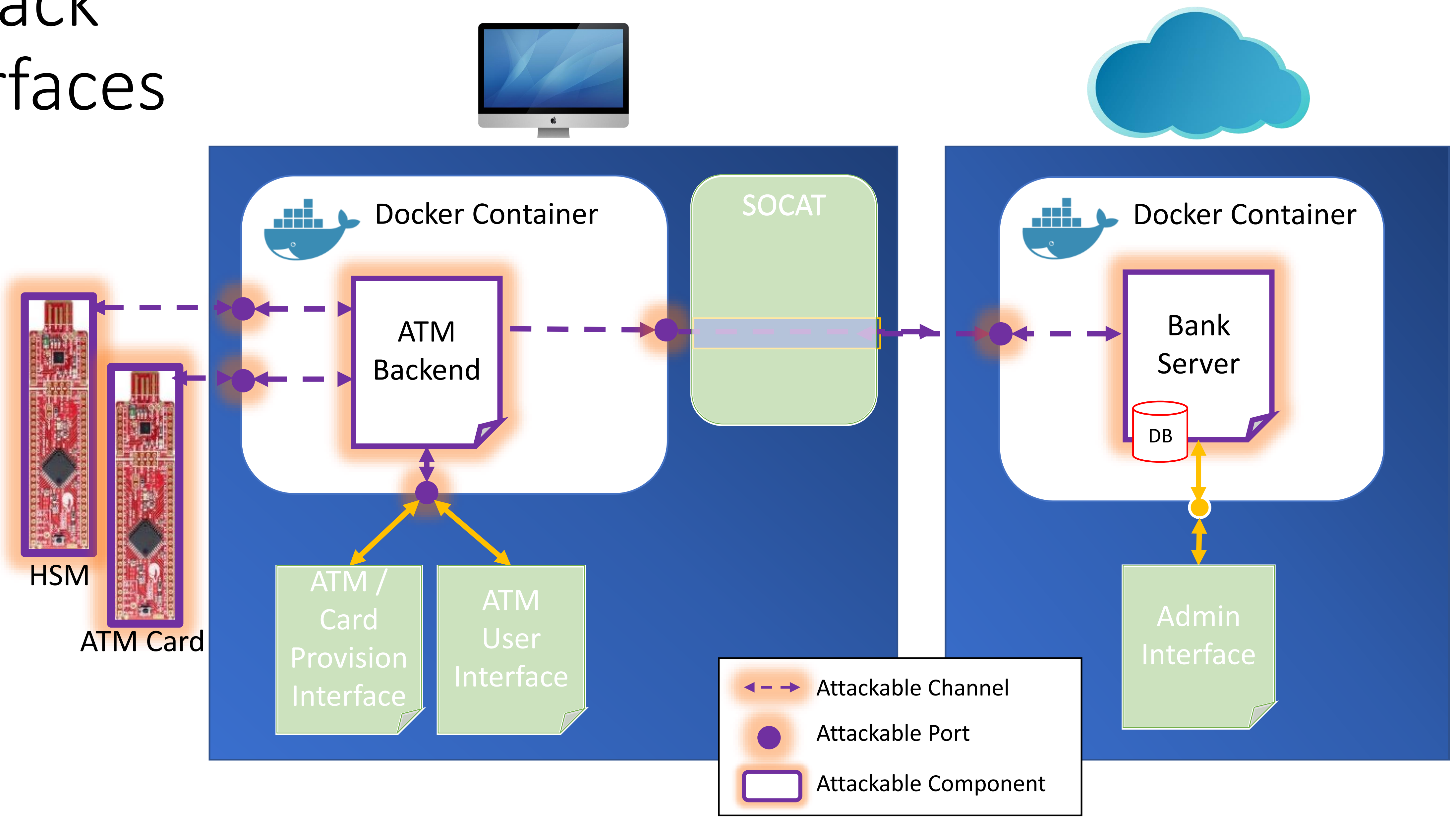
# System Architecture



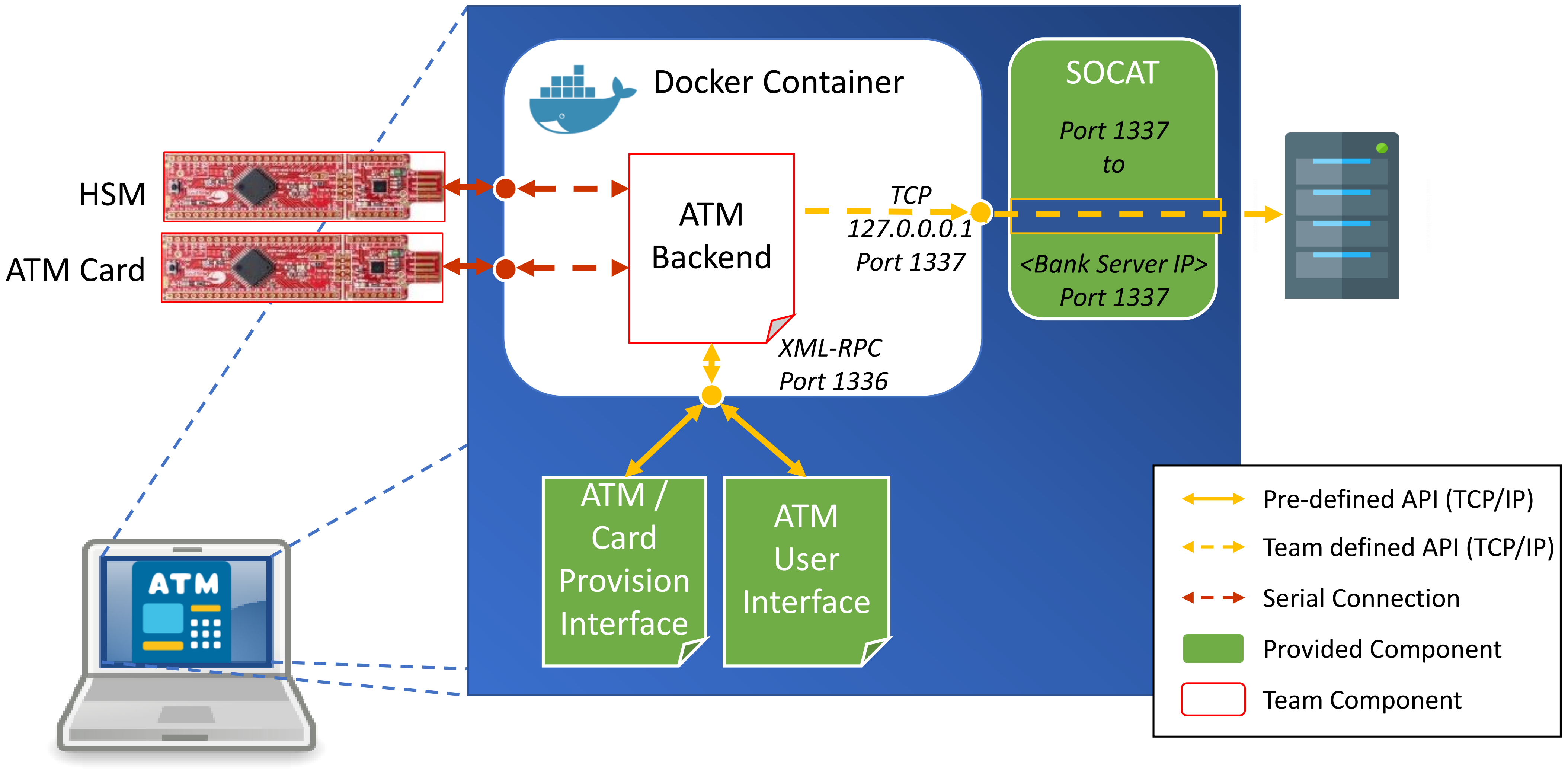
# Computer Architecture



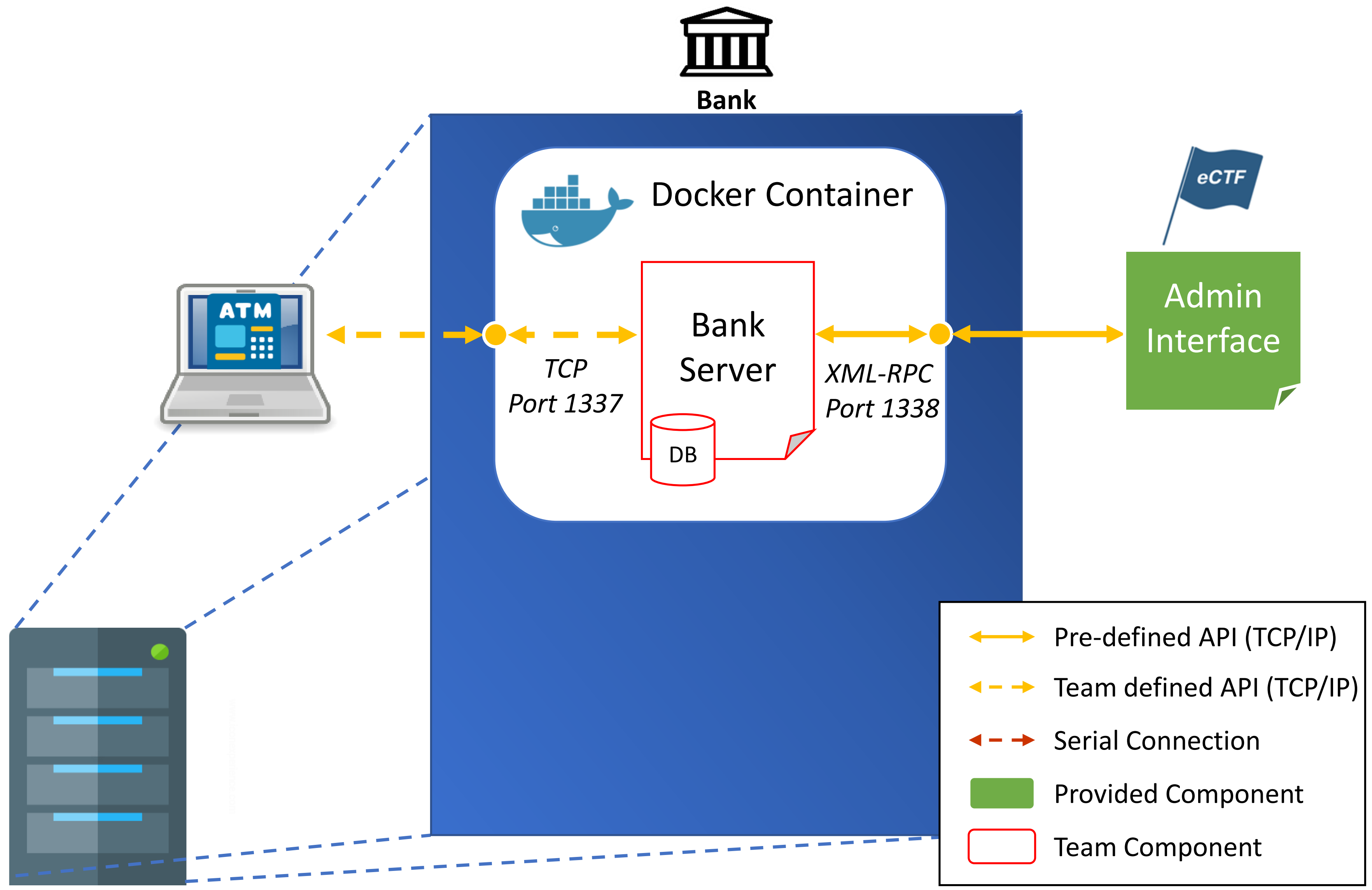
# Attack Surfaces



# ATM Design – PC Platform



# Bank Design – Server Platform



## System Provisioning

Interfaces that administrators will use to initially setup the bank, ATMs, accounts, and cards

## Summary of Provisioning Process

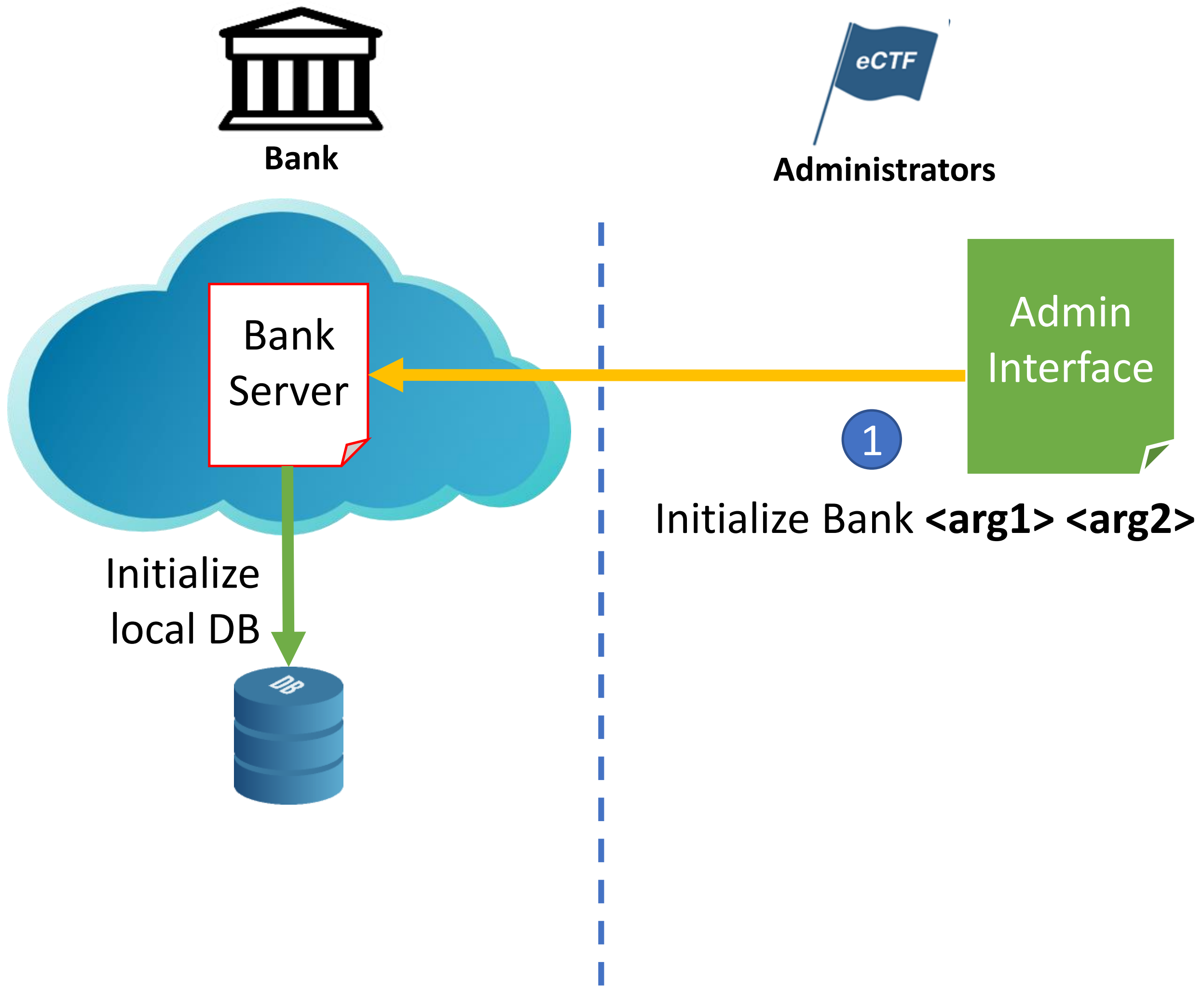
*(Performed by Admins)*

1. Initialize bank server
2. Create a new ATM record on the bank
3. Provision ATM/HSM
4. Create a new bank account
5. Provision ATM card for the new account

## 1. Initial bank setup initiated by Admins

Admins initiate bank setup by sending an "Initialize Bank" command and supplying **<the argument(s)>.**

*Note: If we don't actually supply any arguments during initial setup then we don't need this step. We should just specify that the server performs initialization on first startup. If possible, this would actually be my preferred way of doing it because it's simpler. - Dan*

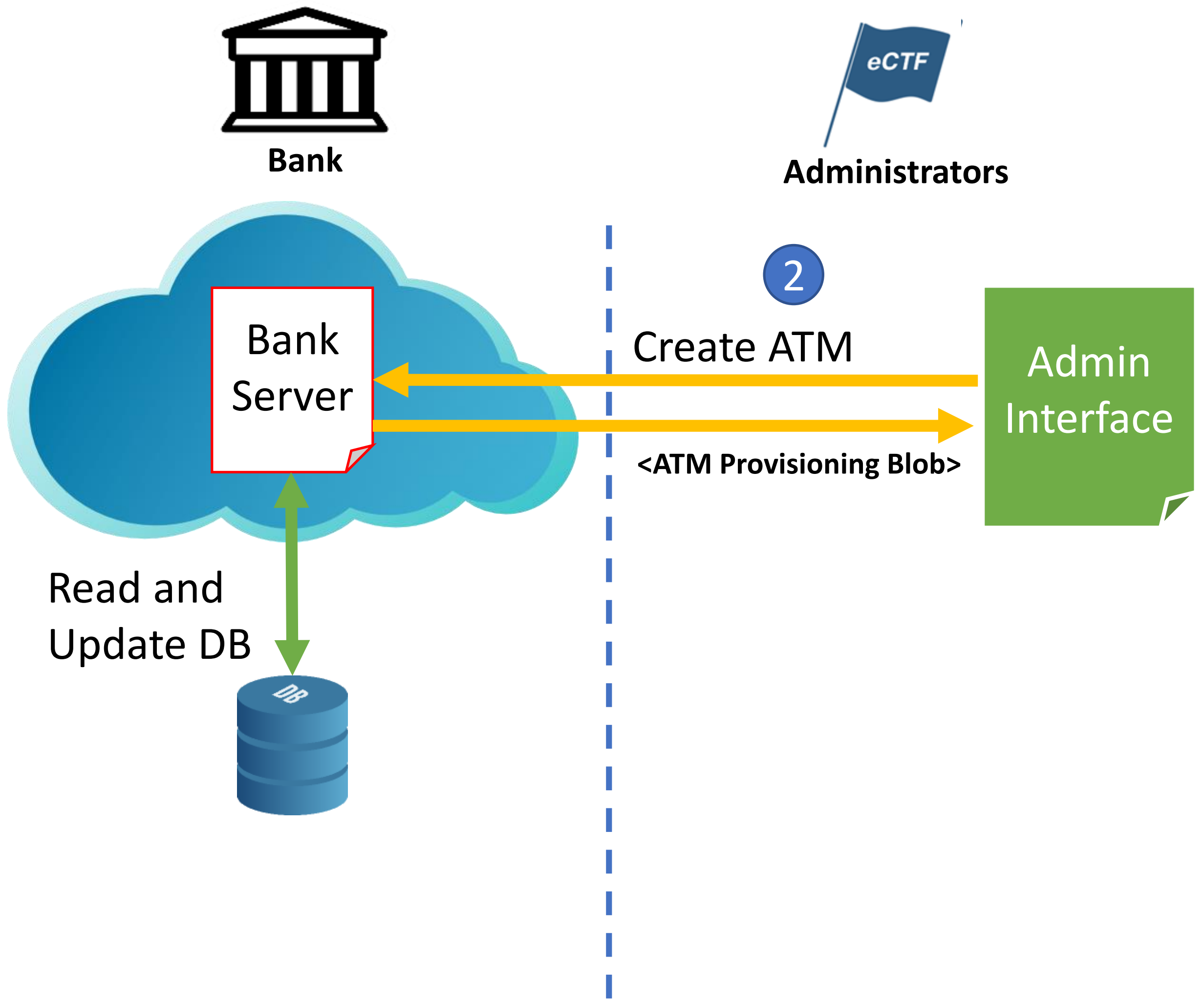


| | |
|---|---|
| ⟷ | Pre-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (Serial) |
| 🟩 | Provided Component |
| ▭ | Team Component |

## 2. Admins create an ATM record on the bank

Admins send a “Create New ATM” command to the bank to tell the server that it must support a new ATM. The bank may generate keys and/or update the local database to create a record to support the new ATM.

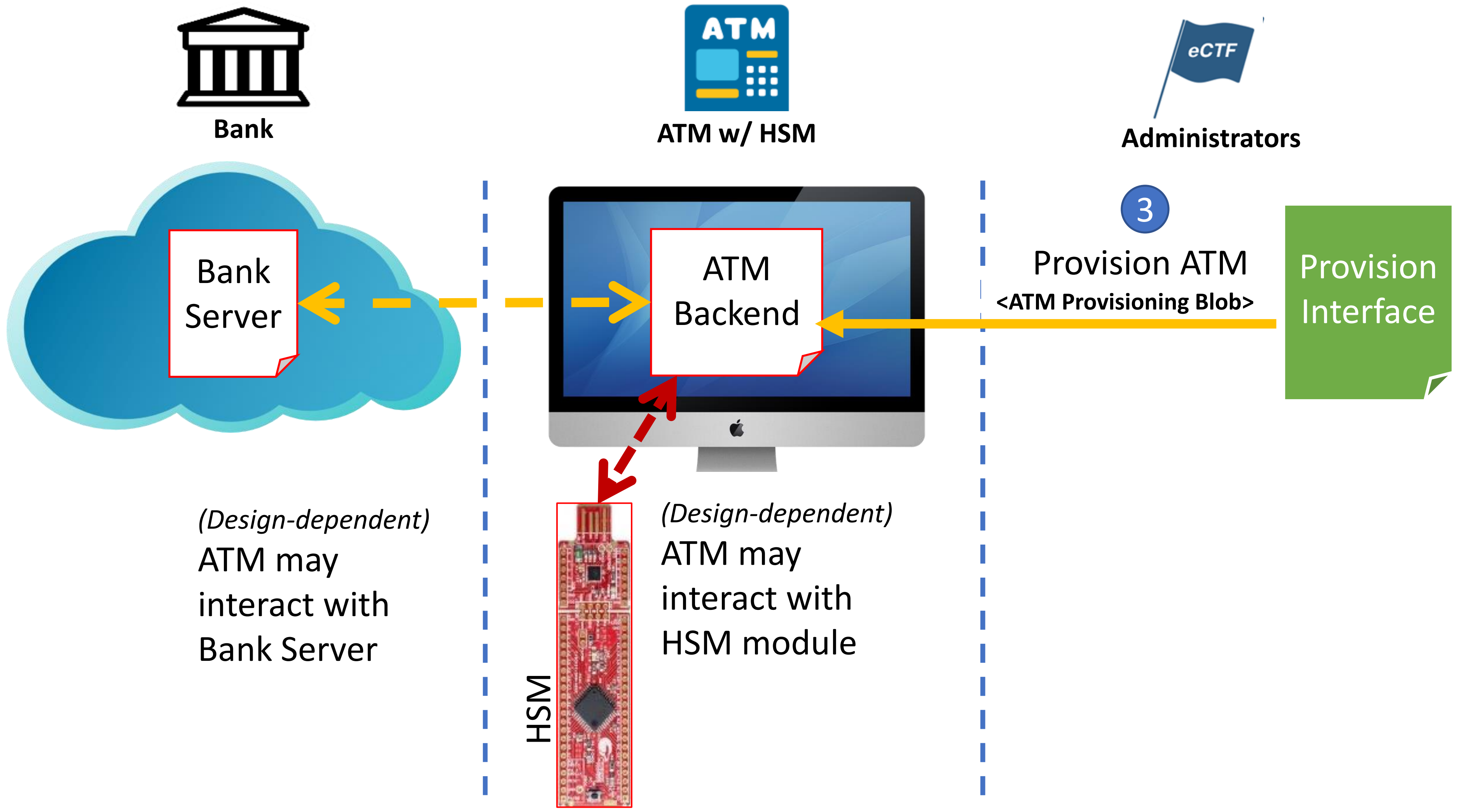
The bank returns an **<ATM Provisioning Blob>** which is used to provision the actual ATM in the next step.

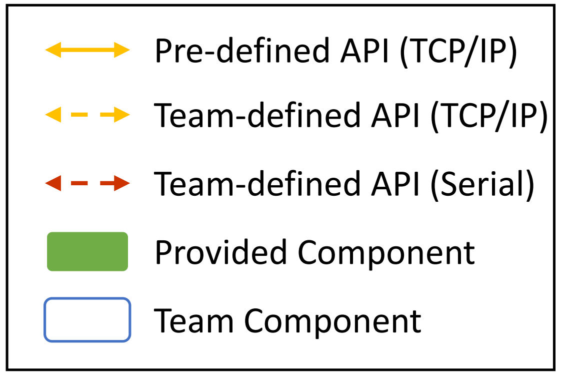


| | |
|---|---|
| ⟷ | Pre-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (Serial) |
| 🟩 | Provided Component |
| ▢ | Team Component |

# 3. Admins provision an ATM

Admins send an “Initialize ATM” command and supply the **<ATM Provisioning Blob>**.

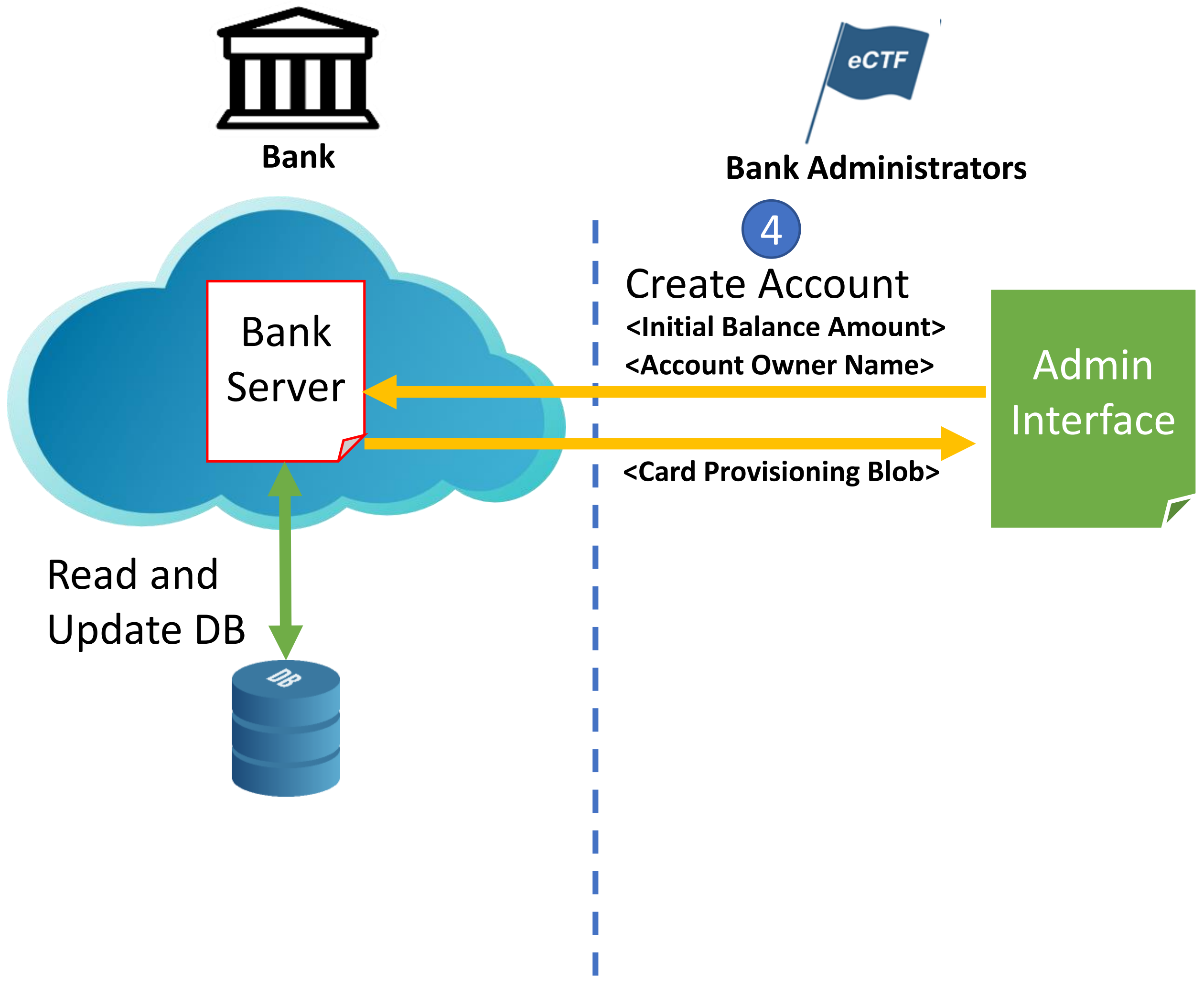




# 4. Admins create a new account

Admins send a “Create Account” command to the bank and supply an account number and owner name. The bank may generate keys and/or update the local database to create a record to support the new account.

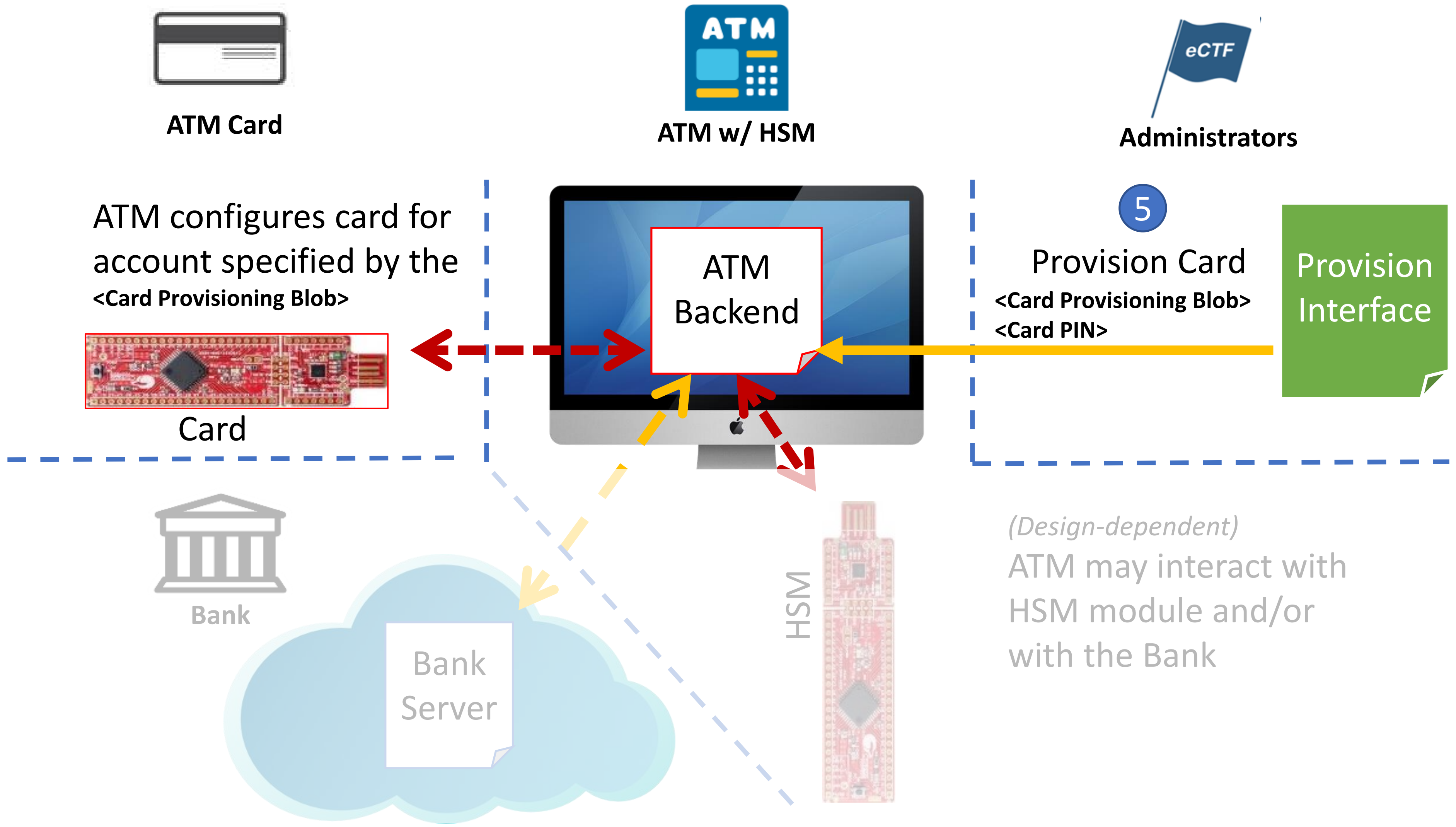
The bank returns a **<Card Provisioning Blob>** which is used to provision the ATM card for this account.



| | |
|---|---|
| ⟷ | Pre-defined API (TCP/IP) |
| ⇠⇢ | Team-defined API (TCP/IP) |
| ⇠⇢ | Team-defined API (Serial) |
| 🟩 | Provided Component |
| ▭ | Team Component |

# 5. Admins provision an ATM card

Admins send a "Provision Card" command and supply the **<Card Provisioning Blob>**.



| | |
|---|---|
| ⟷ | Pre-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (TCP/IP) |
| ⇠ ⇢ | Team-defined API (Serial) |
| 🟩 | Provided Component |
| ▭ | Team Component |

## Flags

| Flag Name | Capturing this flag proves that you can… | Security Goal |
|---|---|---|
| **Cloning** Read-Access | …determine the account balance of the "Borrowed Card" account. | Two-Factor Authentication Pt. 1 |
| **Cloning** Write-Access | …withdraw money from the "Borrowed Card" account. | Two-Factor Authentication Pt. 1 |
| **PIN Bypass** Read-Access | … determine the account balance of the "Stolen Card" account. | Two-Factor Authentication Pt. 2 |
| **PIN Bypass** Write-Access | …withdraw money from the "Stolen Card" account. | Two-Factor Authentication Pt. 2 |
| **Skimming** Read-Access | … determine the account balance of the "Skimmed Card" account. | Secrecy and Authenticity |
| **Skimming** Write-Access | …withdraw money from the "Skimmed Card" account. | Secrecy and Authenticity |
| **ATM Exploit** | …withdraw more cash from the ATM than you were given or withdraw money without a card. | Software Assurance |
| **PIN Extraction** | …extract PIN from the "Stolen Card" (may not be possible for systems that do not store PINs on the card) | PIN Secrecy (Card) |
| **Data Breach** | …determine all PINs from the provided "Hacked Server" container (may not be possible for systems that do not store PINs in the Bank Server) | PIN Secrecy (Bank) |

MITRE

## Attack Phase Delivered Items

- **The IP address and port of the Bank Server**
  - This server and IP address/port will be unique for the requesting team
- **A provisioned ATM Docker container and associated HSM (PSoC dev board)**
- **Provisioned ATM Cards (PSoC dev boards)**
  - **“Own Card”** (for the Known PIN flag)
  - **“Stolen Card”** (for the Unknown PIN flag)
- **The full Docker container representing the “Hacked Server” (for the “Data Breach” flag)**
- **Recordings of transactions involving the “Skimmed Card” (for the “Skimming” flag)**
  - Includes all serial and network traffic from the following legitimate operations:
    - Withdrawal
    - Balance check
  - The PIN used in the recordings for the **“Skimmed Card”**
- **Instructions for how to return the “Own Card” to MITRE, allowing it to become the “Borrowed Card”**

MITRE

# Design Phase Flags

| Milestone | Proof |
|---|---|
| **Read the rules** | Read the rules |
| **Build + Program + Serial** | Build and program the “Milestones Demo” project to the PSoC and read the flag from the serial interface. |
| **Debug** | Build and program the “Milestones Demo” project to the PSoC and read the in-memory flag using the debugger. |
| **Network Plumbing (socat)** | Build and program the “Milestones Demo” project to the PSoC and connect it to the server (network address of the server provided with the milestones demo code) using socat. |

MITRE