High Intensity Deconstruction



Aaron Levy



- Security Researcher
- Reverse Engineer
- Leads Security Engineering at a Payments / POS company

Nick Draffen @tcprst



- Product Security Architect
- Hardware Hacking / Reverse Engineer
- RF Village Staff
- RFID Hacking by Iceman Discord Moderator

Babak Javadi



- Hardware Hacker
- Professional Red Teamer and Covert Entry Instructor
- Red Team Alliance, Co-Founder
  - Physical Security Training and Certification
- Founder and Director of Research of The CORE Group
  - Penetration Testing / Reverse Engineering
- TOOOL, Co-Founder
  - 501(c)(3) Non-Profit Educational Lockpicking Organization

Kate Gray @MistialD



- Reverse Engineer
- Security Architect
- PACS Integrator
- Open Source Access Control Developer

Topics of the Hour

Physical Access Control Systems

Obsession

A Heist

Aftermath: Preview



Obsession

- Project Has Been All-Consuming
- Much Work Done During COVID
- Long 12-14 Hour Days
- Hundreds of Shower Thoughts
- Thousands of Collective Hours
- Hundreds of Readers
- Many, Many, Thousands of Dollars on Hardware (it's better for if I don't calculate it…)
- Very Little Sleep


Maximum Warp

PACS: Physical Access Control System

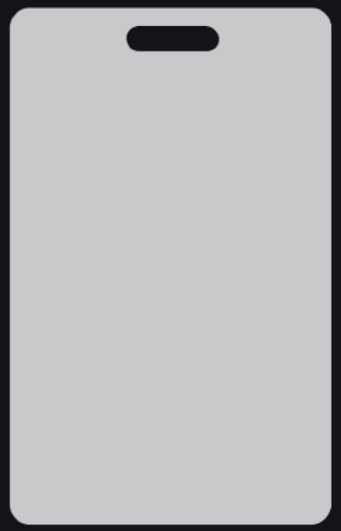
- Definition
  - A security framework designed to manage and regulate who can enter or exit specific physical spaces.
- Capabilities
  - Time-Based Access: Restrict Access by Date and Time
  - Audit Trails: Record and Monitor Access Events
  - Integration: Sync with Alarms, CCTV, and Other Security Systems
- Components
  - Credentials: Cards, Fobs, Biometrics, PINs
  - Readers: Devices that Read Credentials and Transmit Data to Controller
  - Controllers: Units that Process Data and Make Access Decisions
  - Locks: Electromechanical Mechanisms that Secure Doors
  - Software: Tools for Administering, Monitoring, and Auditing Access

PACS: Design Intent

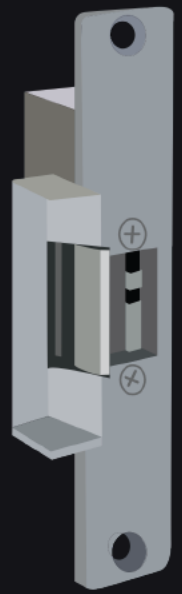
- Cost-Effective
- Simple to Use
- Simple to Deploy
- Secure [Enough]

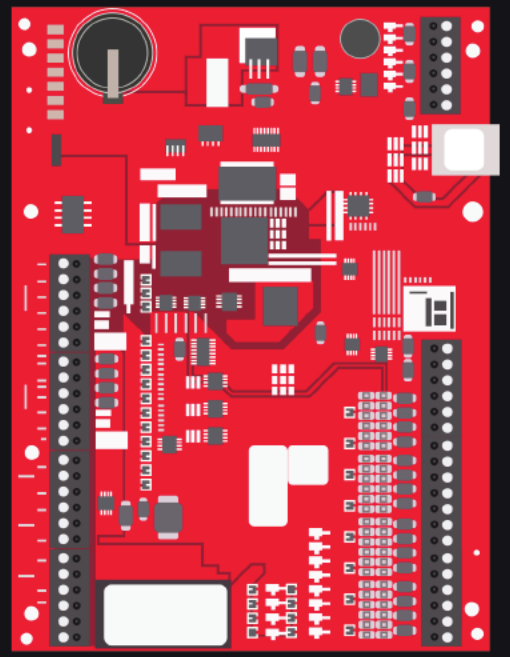
- Golden Rule of Access Control
  - Never get in the way of doing business!

PACS: System Overview











Card

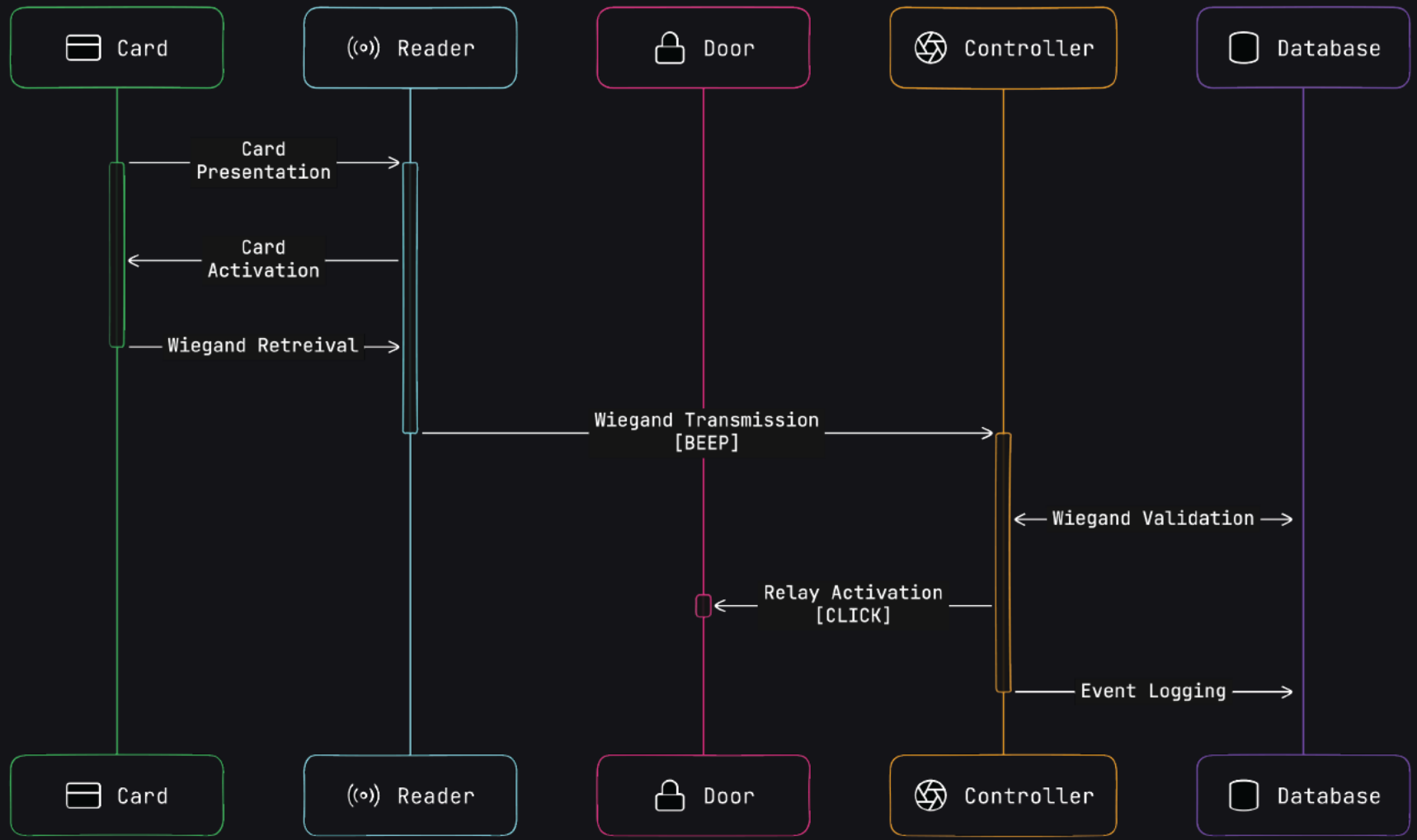
Reader

Door

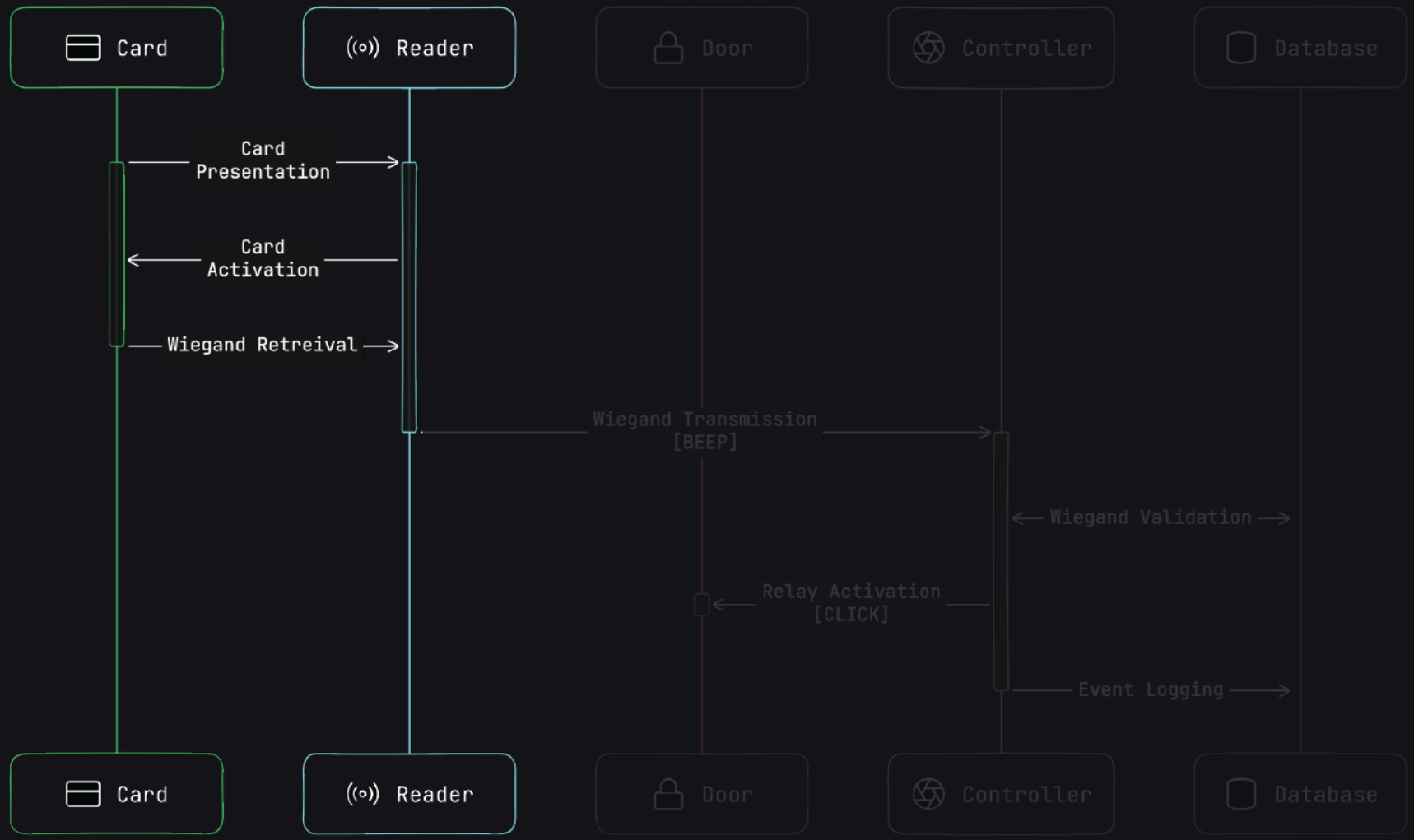
Controller

Database

PACS: Theory of Operation



PACS: Area of Focus



Credentials: What is Wiegand Data?

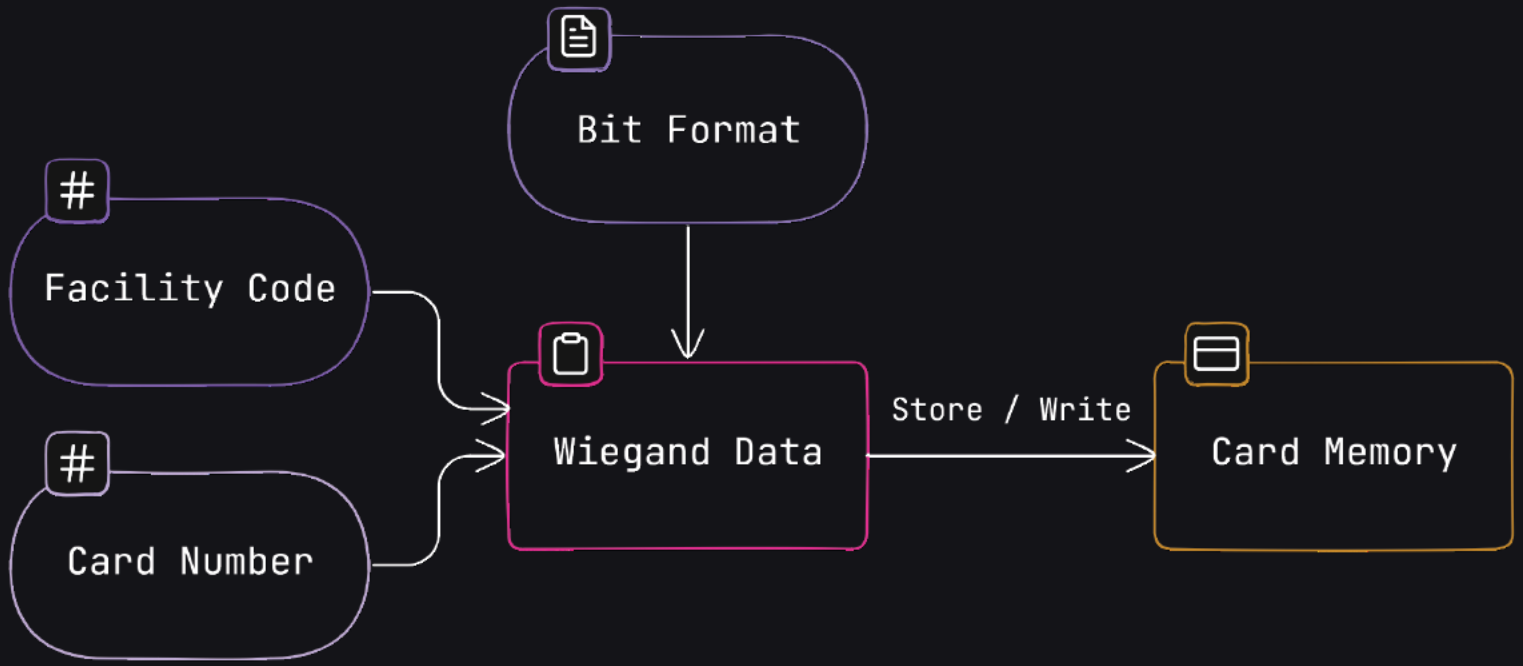
- Static Authentication String Contains Basic Data
  - Facility Code (FC) – Shared Value(s) for All Credentials within System
    - Usually Not Selected or Chosen by End Customer
  - Card Number (CN) – System-Unique Identifier for Credential
- Bit Format
  - Set of Rules for Combining FC and CN into Binary String
  - Contains Rudimentary Parity Checks
  - Completely Arbitrary and Made Up by Vendors
  - “Bit Length” Varies Wildly, but Does Not Meaningfully Impact Security

Credentials: Card vs. Credential

- Most PACS Cards Are Just Memory Storage
- Card Becomes “Credential” Once Wiegand Data is Written
- PACS Credential Interactions Can Be Broken into Two Operations
  - Data Storage (Encoding)
  - Data Retrieval (Decoding)



Credentials: Store (Encode)



Credentials: Store (Encode)

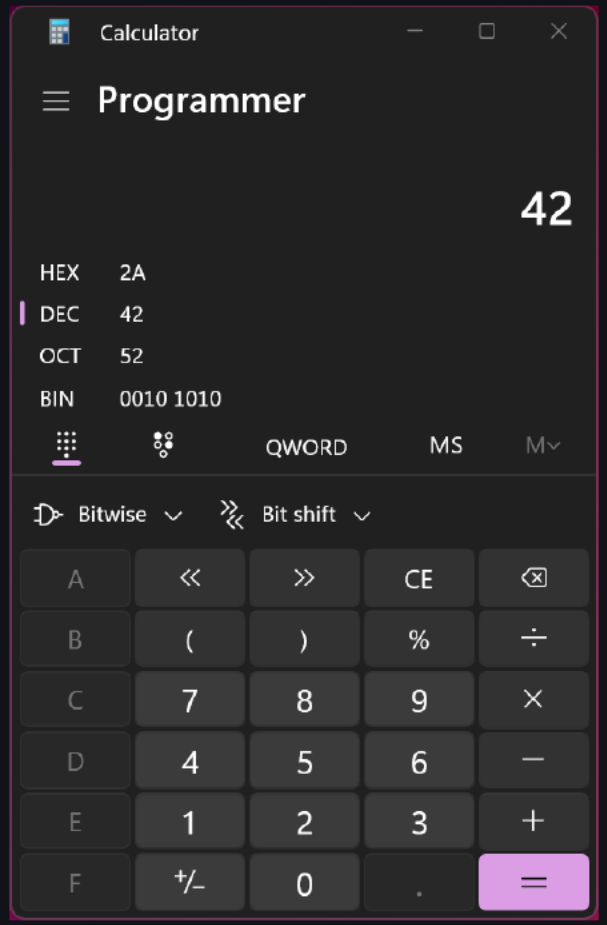
Example Credential

Facility Code: 42

Card Number:

26-Bit Open Format (H10301)

EFFFFFFFFCCCCCCCCCCCCCCCCO



Credentials: Store (Encode)

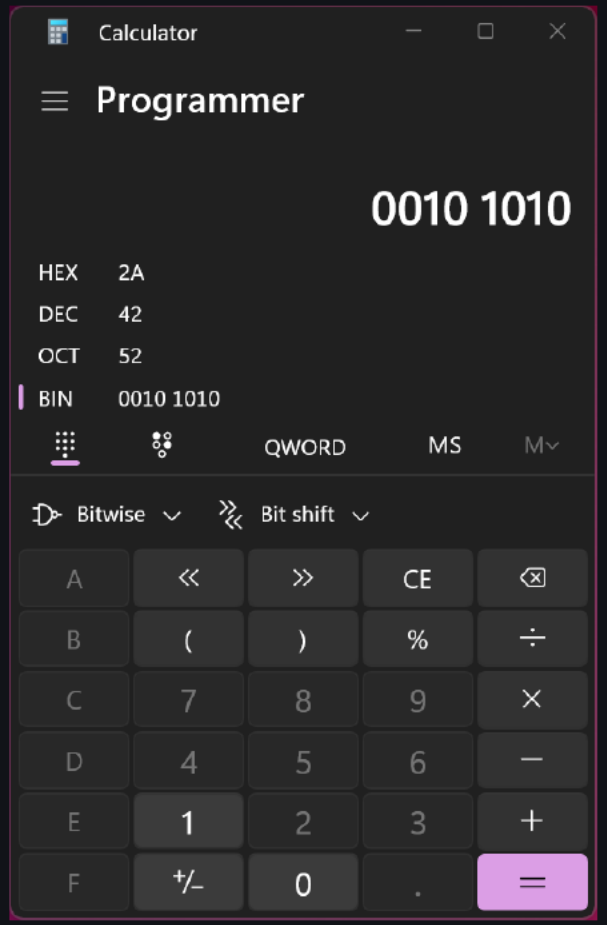
Convert Facility Code to Binary

Facility Code: 00101010

Card Number:

26-Bit Open Format (H10301)

EFFFFFFFFCCCCCCCCCCCCCCCCO



Credentials: Store (Encode)

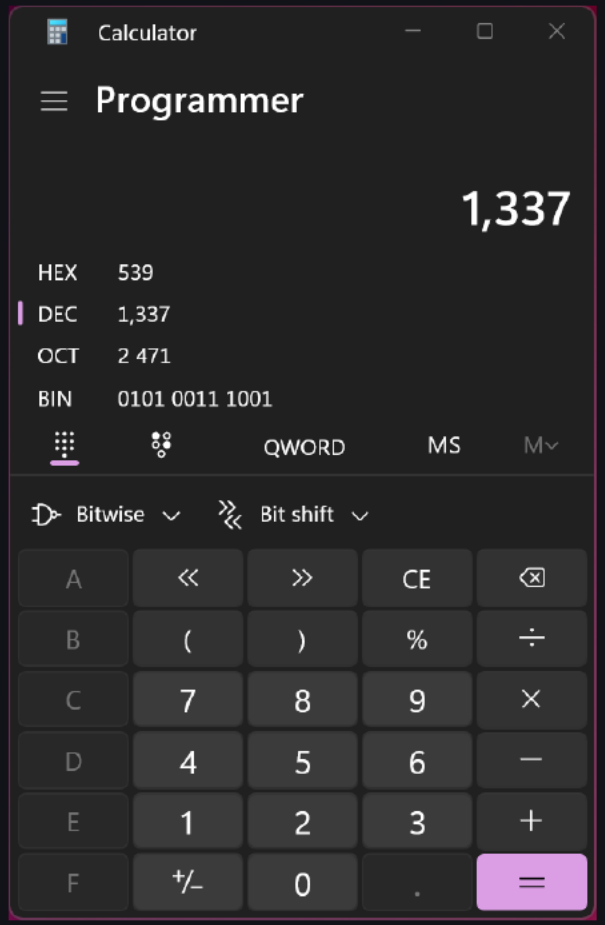
Convert Card Number to Binary

Facility Code: 00101010

Card Number: 1337

26-Bit Open Format (H10301)

EFFFFFFFFCCCCCCCCCCCCCCCCO



Credentials: Store (Encode)

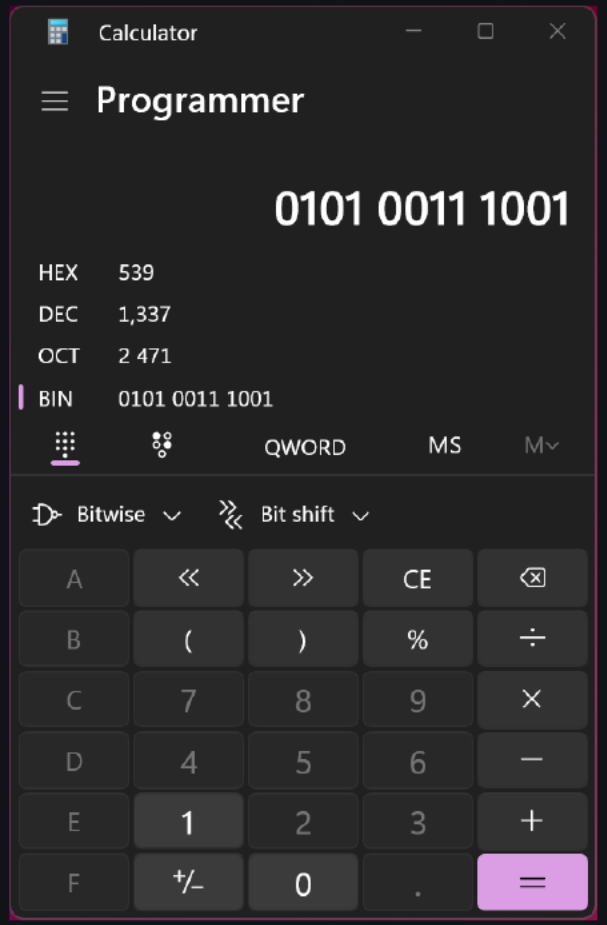
Convert Card Number to Binary

Facility Code: 00101010

Card Number: 0000010100111001

26-Bit Open Format (H10301)

EFFFFFFFFCCCCCCCCCCCCCCCCO



Credentials: Store (Encode)

Insert Into Bit Format

00101010 0000010100111001

EFFFFFFFFCCCCCCCCCCCCCCCCO

Credentials: Store (Encode)

Insert Into Bit Format

E00101010000001010011100 1O

Credentials: Store (Encode)

Calculate Parity Bits

E0010101000000101001110010

Credentials: Store (Encode)

Calculate Parity Bits

100101010000001010011100 11

Credentials: Store (Encode)

Static Wiegand Credential String is Complete

```
10010101000000101001110011
```

Credentials: Store (Encode)

Add Any Necessary Preamble or Start Bits

```
0010000000001 10010101000000101001110011
```



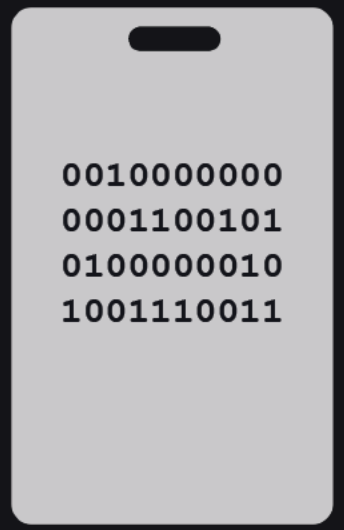
Credentials: Store (Encode)

Store Data to Credential Memory

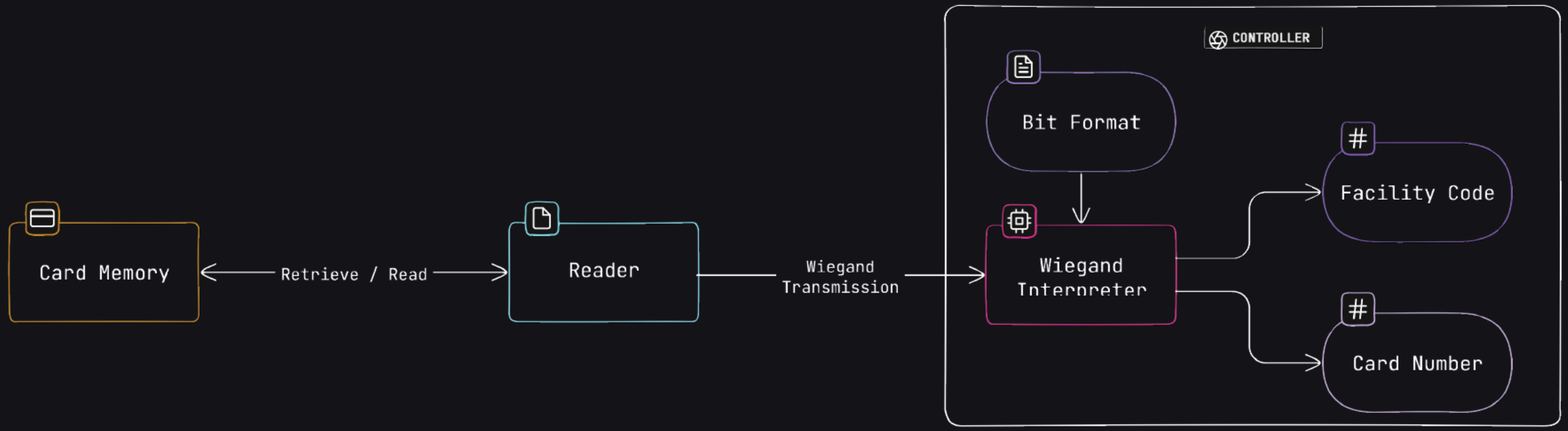


Credentials: Store (Encode)

Store Data to Credential Memory

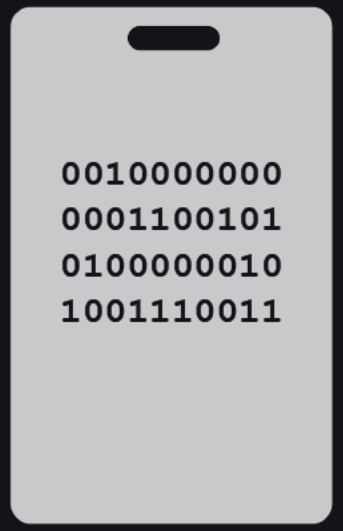


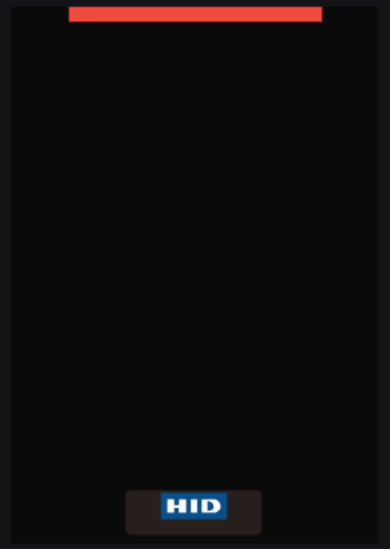
Credentials: Retrieve (Decode)

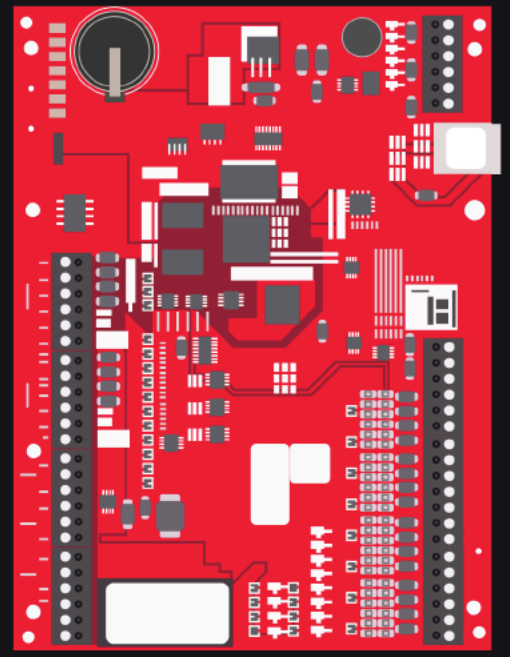


Credentials: Retrieve (Decode)

Retrieve Credential from Memory





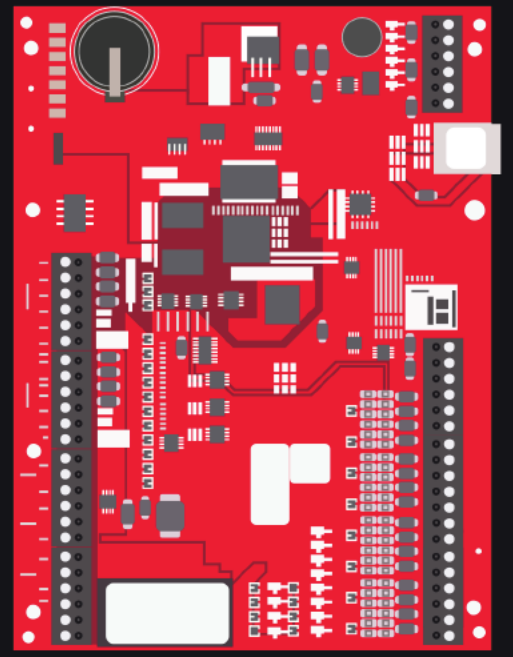


Credentials: Retrieve (Decode)

Reader Parses Wiegand Data



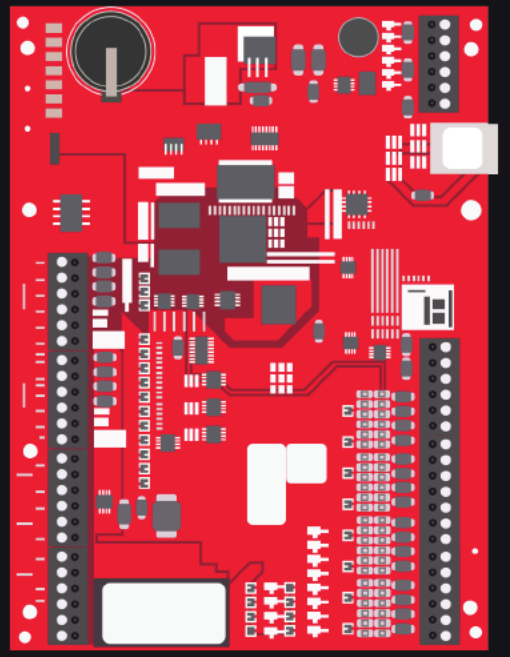




Credentials: Retrieve (Decode)

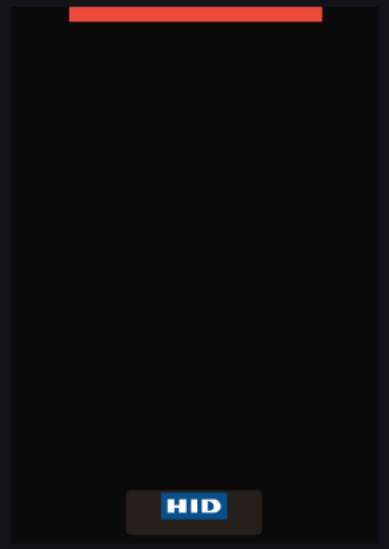
Reader Strips Preamble and Transmits Wiegand String





Credentials: Retrieve (Decode)

Controller Receives Wiegand Data





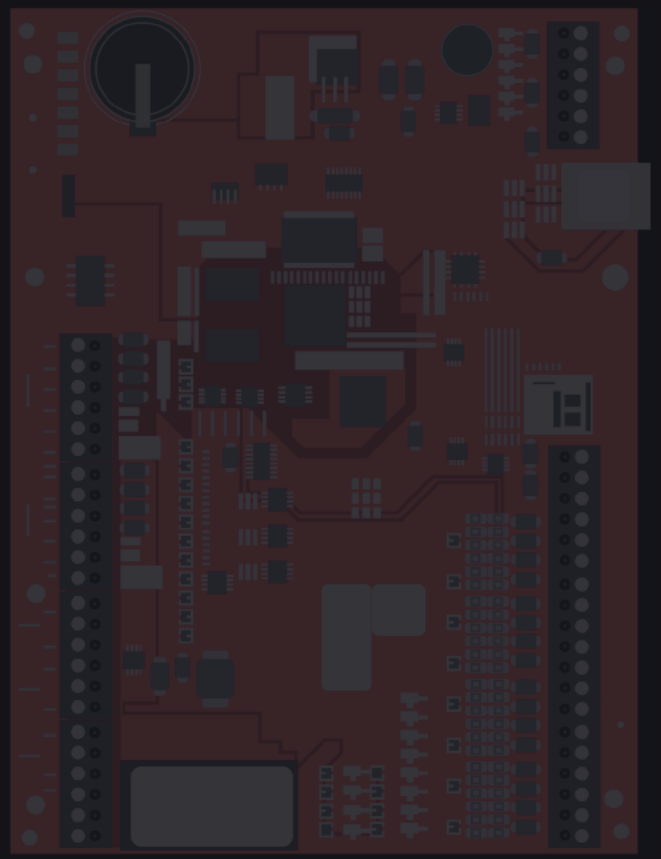
Credentials: Retrieve (Decode)

Controller Parses Wiegand Data



Credentials: Retrieve (Decode)

Controller Parses Wiegand Data

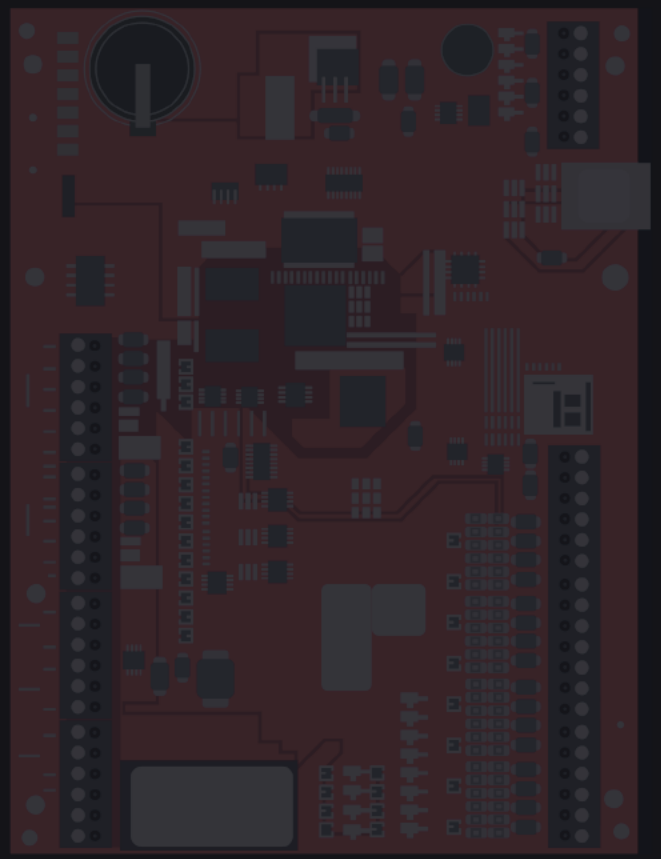


00101010

0000010100111001

Credentials: Retrieve (Decode)

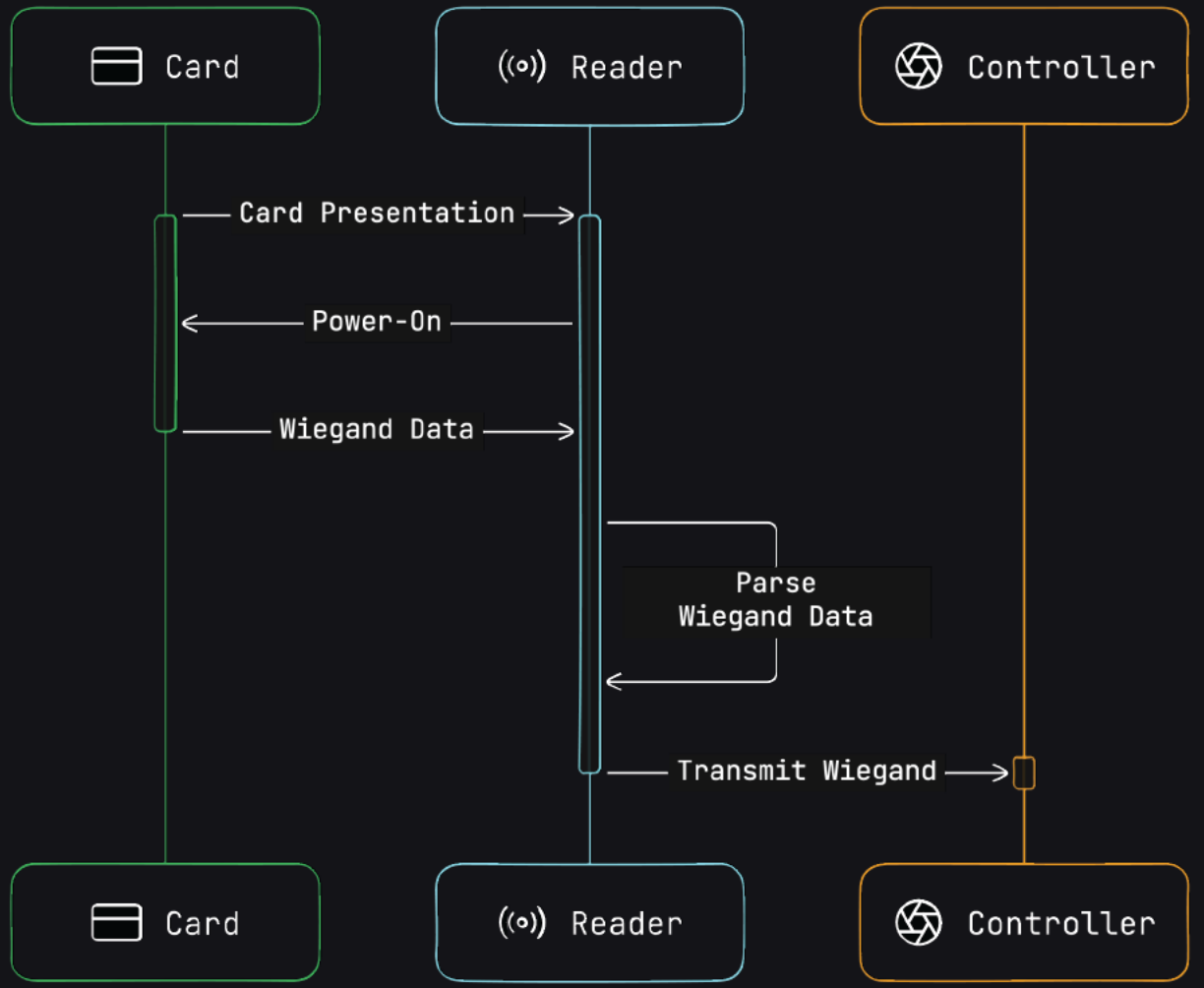
Controller Parses Wiegand Data



FC: 42

CN: 1337

Credentials: Unauthenticated Cards

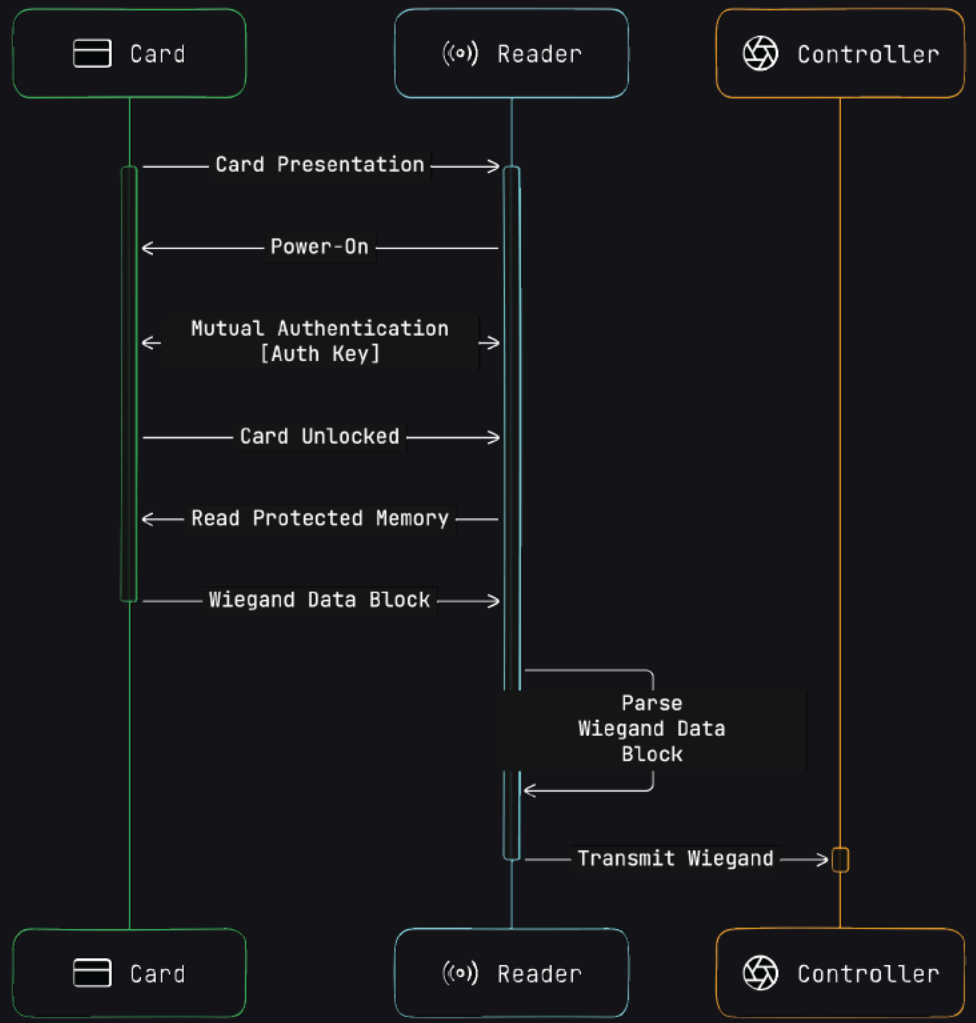


Credentials: Unauthenticated Problems

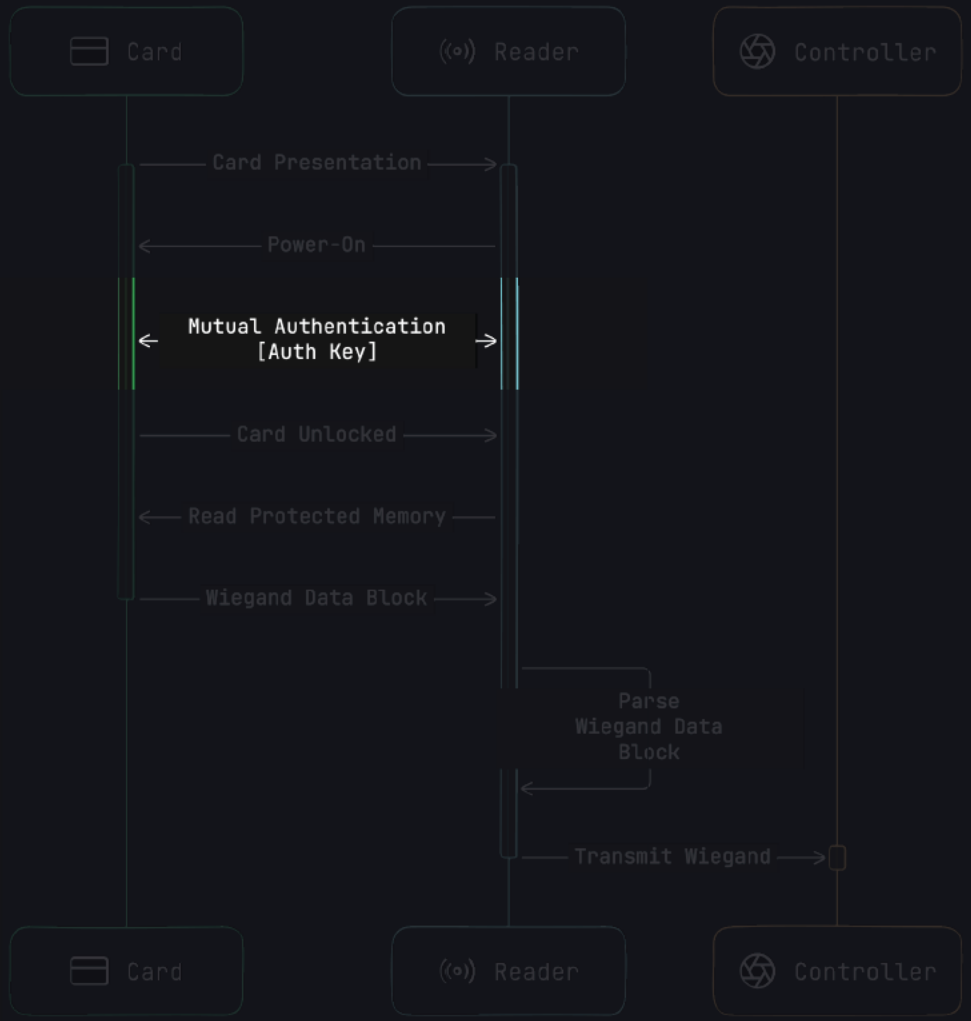
- Legacy 125kHz Credential Struggles
  - Data Generally Read without Authentication
  - No Data Protection at Rest
  - No Data Protection in Motion
- Solution?
  - Card Authentication



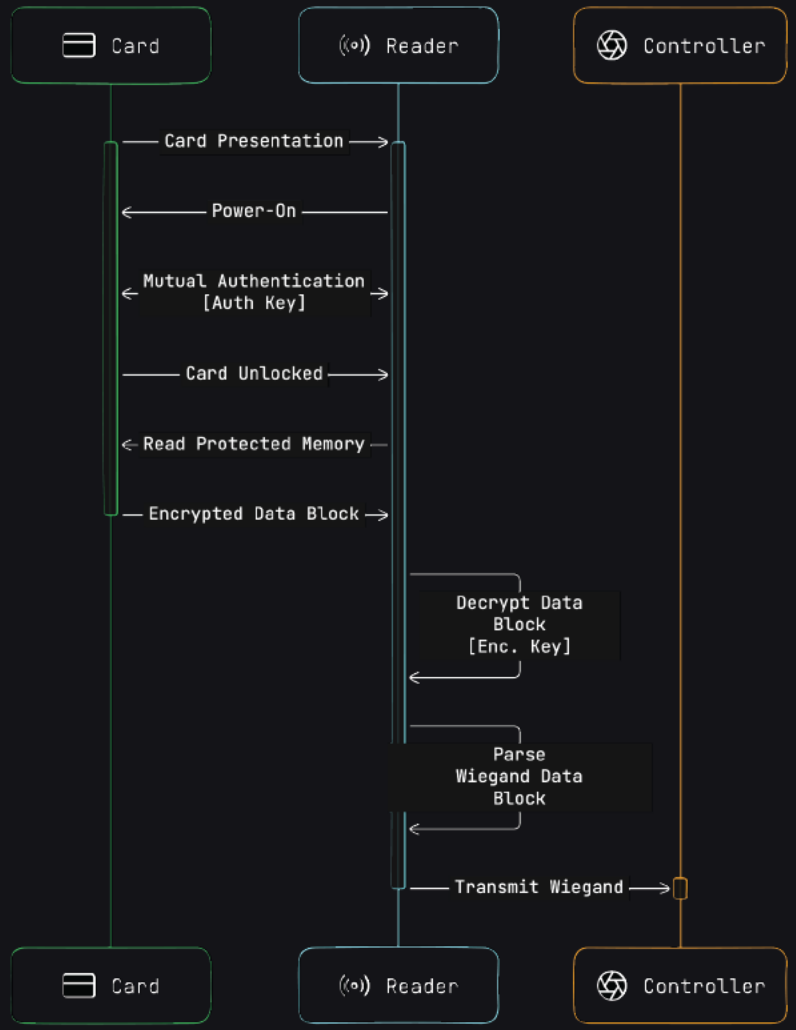
Credentials: Authenticated Cards



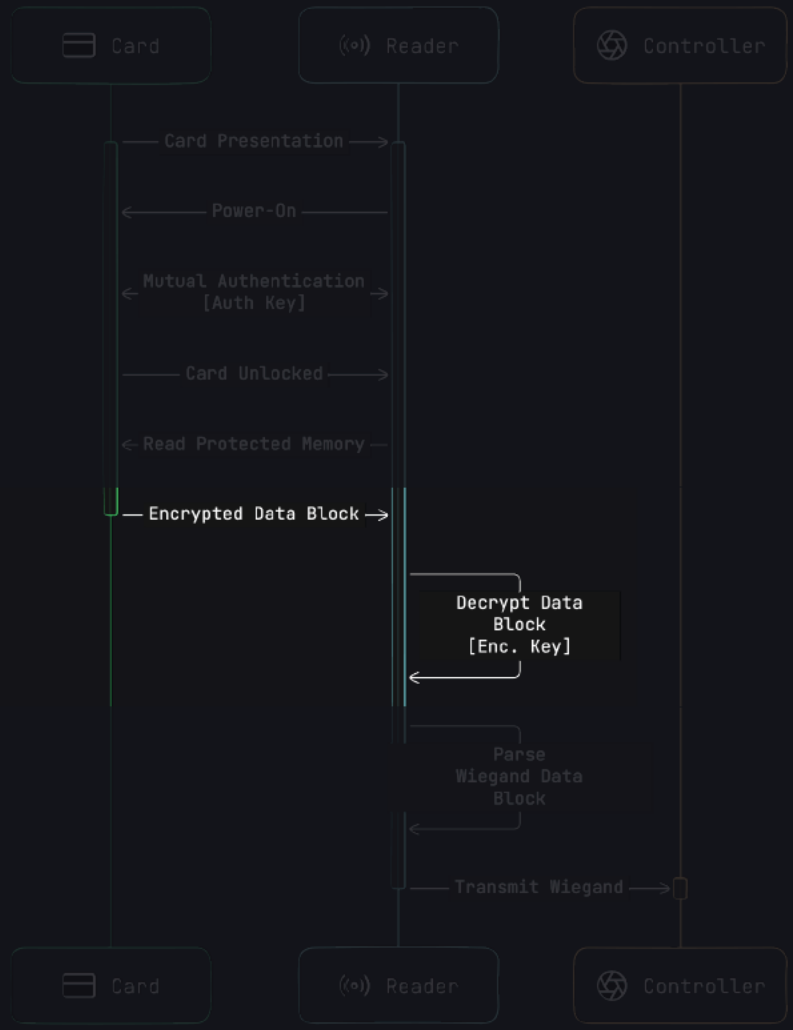
Credentials: Authenticated Cards



Credentials: “Transport” Encryption



Credentials: “Transport” Encryption

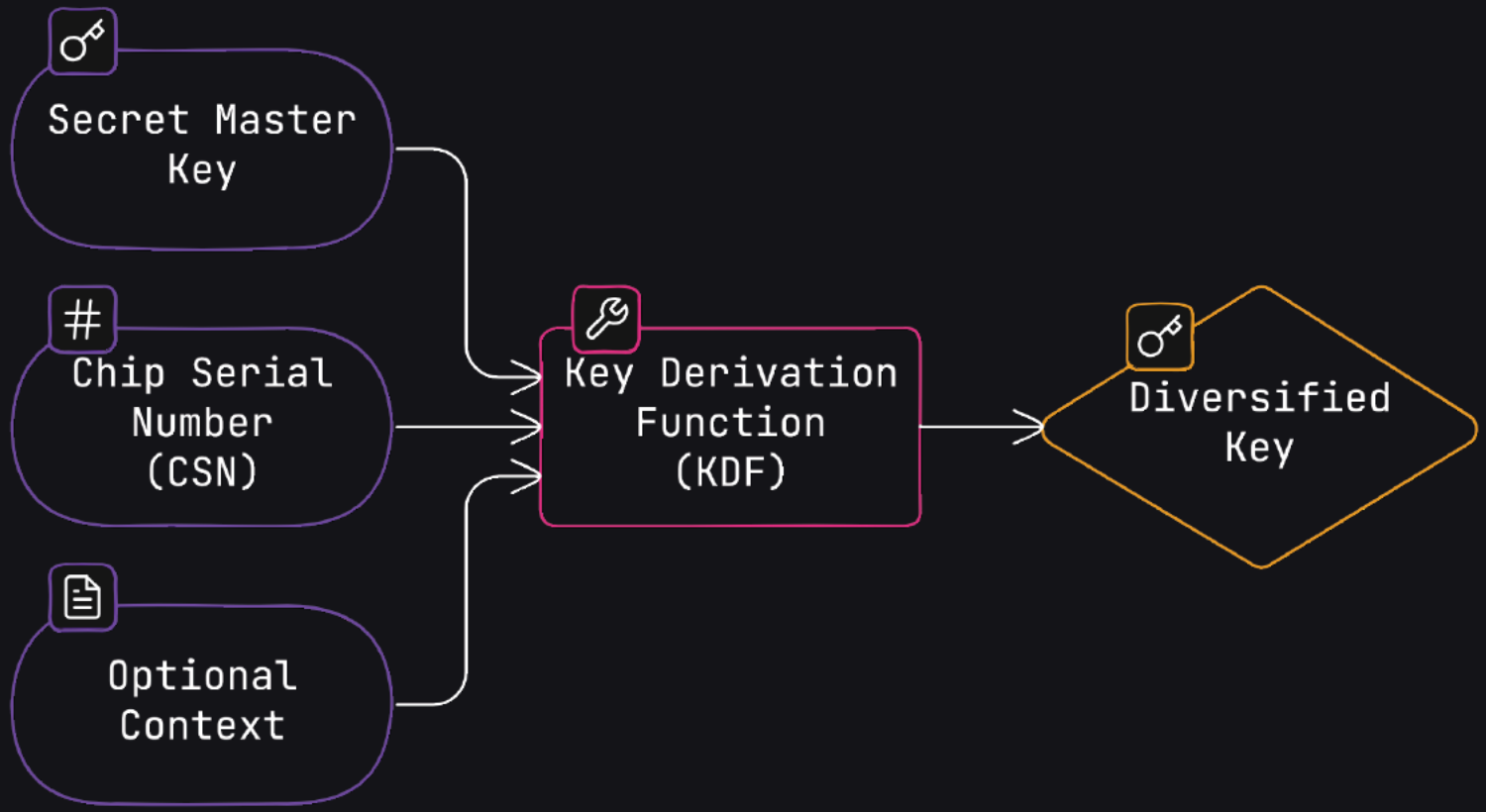


Credentials: Mo' Security, Mo' Problems

- Authentication Improves Security, but Creates a New Problem
  - Key Management
- Keys Must Be Managed Appropriately
  - Generation: Secure Key Generation with Random Seeds
  - Storage: Secure, Tamper-Resistant Key Storage
  - Usage: Use Keys without Exposing or Divulging Key Material



Credentials: Key Diversification



Credentials: Shared vs. Custom Keys

- Shared Master Keys
  - Credentials and Readers Interoperable by Default
  - Easier for Integrators
    - No Configuration, Fast Deployment, Sell from Inventory
  - Easier for End Customers
    - Quick Availability
    - Cost-Effective
- Custom Master Keys
  - Unique Key Generated for Customer
  - Credentials and Readers Must Use Matching Keyset
    - Poor Integrator Competency
    - Custom Orders Result in Increased or Unexpected Lead Times
    - No Local On-Demand Inventory Available

Credentials: Still Wiegand

- Unauthenticated Credential w/ Plaintext Data
  - Reader Outputs Wiegand
- Authenticated Credential w/ Plaintext Data
  - Reader Outputs the same Wiegand
- Authenticated Credential w/ Encrypted Data
  - Reader Outputs the same Wiegand
- Authenticated Credential w/ Encrypted & Signed Data bound to CSN
  - Reader Outputs the same Wiegand

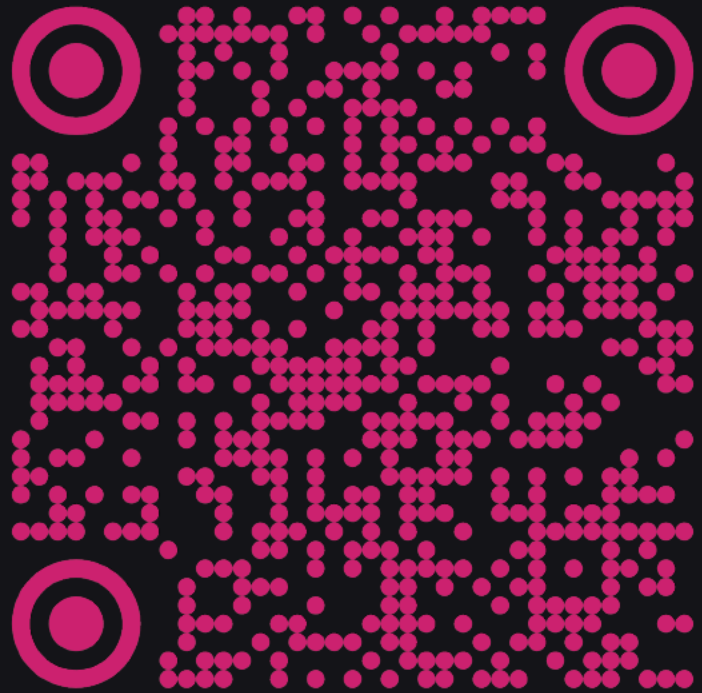
A Wee Bit of Context

While Standing on the Shoulders of Giants

Link to Paper

iCLASS: Heart of Darkness

- Heart of Darkness - exploring the uncharted backwaters of HID iCLASS security
  - Milosch Meriac
  - 27th CHAOS COMMUNICATION CONGRESS, DECEMBER 2010
- Demonstrated Fragility of Shared Key Architecture
- Highlighted Need for Proper Security Engineering



iCLASS: Overview

- Introduced in August 2002
- Primary Application
  - Physical Access Control
- Extended Applications
  - Stored Value Systems (Transit, Coin-Op, Prepaid)
  - Biometric Data
  - Logical Access Control (PC / Network Logon)
- Fast, Cost-effective, Relatively Secure at the Time (if implemented correctly)
- Credentials on INSIDE Contactless PicoPass Silicon (Datasheet)
- Very Popular in North America, Australia, and Singapore

iCLASS: Legacy Credential Application

[=] ------------------------------ Credential Memory Map ----------------------------------

[=] Block #	Encrypted Data	Unencrypted Data	Label
[=] 0/0x00	72 6C 9B 02 F8 FF 12 E0	72 6C 9B 02 F8 FF 12 E0	Card Serial Number (CSN)
[=] 1/0x01	12 FF FF FF 7F 1F FF 3C	12 FF FF FF 7F 1F FF 3C	Card Configuration Block
[=] 2/0x02	DE FE FF FF FF FF FF FF	DE FE FF FF FF FF FF FF	ePurse (Monotonic Counter)
[=] 3/0x03	7F DF 66 49 62 A3 6A E6	7F DF 66 49 62 A3 6A E6	AA1 Authentication Key
[=] 4/0x04	FF FF FF FF FF FF FF FF	FF FF FF FF FF FF FF FF	AA2 Authentication Key
[=] 5/0x05	FF FF FF FF FF FF FF FF	FF FF FF FF FF FF FF FF	Application Issuer Area
[=] 6/0x06	03 03 03 03 00 03 E0 17	03 03 03 03 00 03 E0 14	PACS Data Model Config
[=] 7/0x07	76 9C B4 A1 98 E0 DE C8	00 00 00 00 06 54 7E 68	PACS Data (Wiegand)
[=] 8/0x08	2A D4 C8 21 1F 99 68 71	00 00 00 00 00 00 00 00	PACS Data (Wiegand)
[=] 9/0x09	2A D4 C8 21 1F 99 68 71	00 00 00 00 00 00 00 00	PACS Data (Wiegand)
[=] 10/0x0A	FF FF FF FF FF FF FF FF	FF FF FF FF FF FF FF FF	Unused
Cont...	FF FF FF FF FF FF FF FF	FF FF FF FF FF FF FF FF	Unused
[=] 18/0x12	FF FF FF FF FF FF FF FF	FF FF FF FF FF FF FF FF	Unused

[=] ---------+------------------------+------------------------+------------------------------

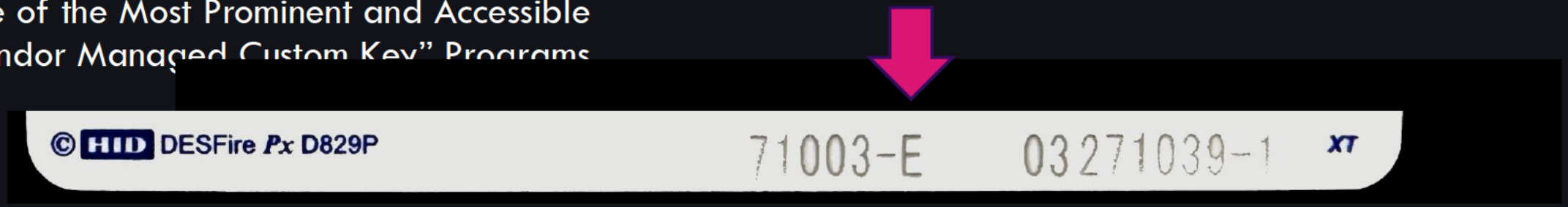
iCLASS: Reader Hardware

- Several Hardware Revisions
  - Rev. A (Original, Brief)
  - Rev. B (Intermediate, Extremely Brief Release Cycle)
  - Rev. C (Most Popular Version in Field)
- Rev. A Selected as Target
  - Oldest Reader for the Most Bugs
  - Used Commodity PIC18F452 PIC MCU
  - MCU Not Hardened Against Attacks (Basic Read Protection)



iCLASS: Key Management

- Standard Keys
  - Shared Master Keys for Readers and Cards
  - Default Configuration
  - By Far Most Common
- Elite Keys
  - Managed Custom Key Program
  - Cards and Readers Built to Order
  - Cards and Readers Arrive Pre-Programmed
  - Customer Doesn't Have Access to Key
  - Orders Must Be Placed by Authorized Individual
  - One of the Most Prominent and Accessible "Vendor Managed Custom Key" Programs



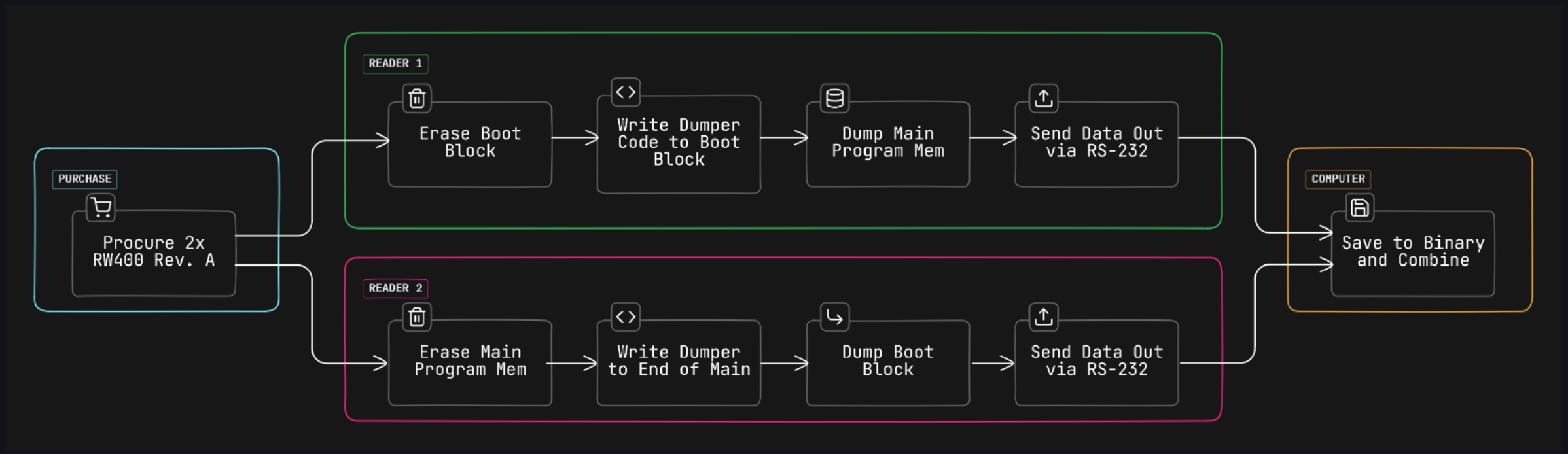
iCLASS: Encoding

- RWK400
  - Multiple Use-Cases
    - Stored-Value Applications
    - Vending
    - Logical Access
    - Other Custom Applications
  - RS-232 Interface Using “iCLASS Serial Protocol”
  - For a Long Time, Most Reliable Method to Write Cards





iCLASS: Heart of Darkness Hack

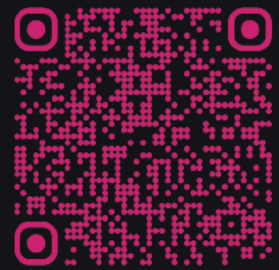


iCLASS: HoD Consequences

- Standard Authentication and Transport Keys Recovered
  - Broken Chain of Custody for Keys
- Firmware Recovered
  - Revealed Proprietary Diversification Algorithms
- Prompted Years of Cryptographic Analysis of Algorithms
  - Resulted in Contactless Covert Key Recover of Elite Keys (loclass)
  - Increased Risk for “High Security” Customers
- Contributed to Discovery of Legacy Elite Keygen Seed and Algo (DEF CON 30)
  - 0xFFFF
  - Iceman

iCLASS: Subsequent Research

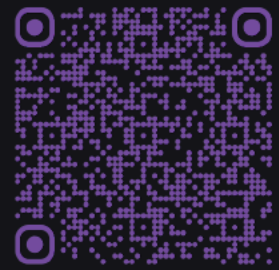
- 2011: Exposing iClass Key Diversification
  - Garcia, de Koning Gans, Verdult



- 2012: Dismantling iClass and iClass Elite
  - Garcia, de Koning Gans, Verdult, & Meriac



- 2014: Wirelessly Lockpicking a Smart Card Reader
  - Garcia, de Koning Gans, Verdult



iCLASS Legacy Failure Analysis

- Key Management
  - Poor Key Storage Mechanism
  - Flawed KDF for Elite
  - Poor Key Generation Methods
- Data Integrity
  - Transport Encryption was Optional, and Controlled by Credential
  - Symmetric Encryption without Signature Allowed for Tampering
- Upgradeability
  - In-Field Firmware Upgrades Not Possible
  - Difficult to Implement Patches or New Functionality
    - Couldn't Fix or Upgrade Key Algorithms
    - Unpatched Rev C Reboot Bug (Carl55)
    - Unpatched ePurse Implementation Bugs

iCLASS Reborn



iCLASS SE: Overview of Security Solution

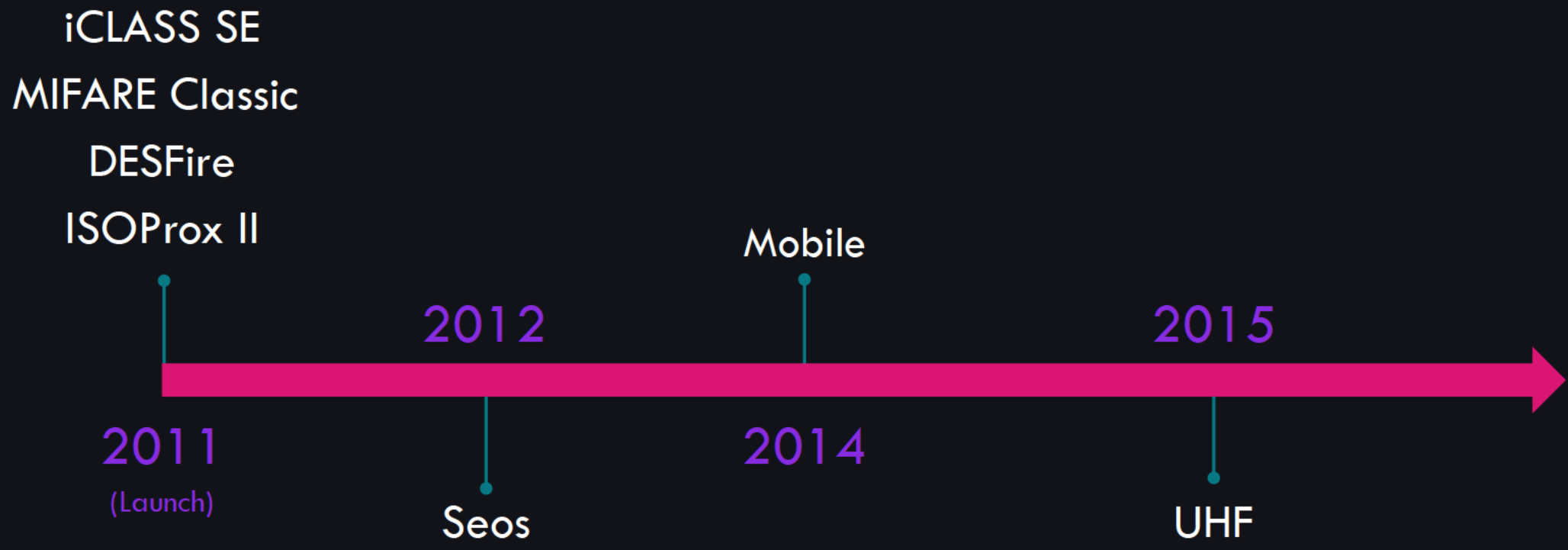
- Launched in 2011
- iCLASS SE = SIO-Enabled
  - SIO = Secure Identity Object
  - Yes, it's a nested acronym!
- *Extremely* Substantial Upgrade (Branding Doesn't Do it Justice)
  - iCLASS SE Cards:
    - New AES-based KDF (NIST SP800-108)
    - Enhanced 16-byte Master Key (vs. Previous 8-byte Key)
  - iCLASS SE Readers:
    - Cryptographic Key Material Stored in Discrete SAM
    - Support for Wide Range of Card Technologies (e.g. MIFARE Classic, DESFire, etc.)

iCLASS SE: Branding Confusion

- iCLASS SE is…
  - Name of a Platform
  - Name of a Reader
  - Name of a Credential Technology
- Re-use of the “iCLASS” Terminology has Created Confusion
  - Integrators
  - Customers
  - Industry as Whole
- In Fairness, the Entire Industry Seems Perpetually Confused



iCLASS SE: Supported Technologies



iCLASS SE Cards: Secure Identity Objects

- Contains Encrypted Wiegand and Authentication MAC
- Extensible
  - Algorithm Can Be Upgraded Over Time
  - Can Contain Non-Wiegand Data

```
30 35 81 06 01 85 ED 54  93 22 A5 02 05 00 A6 08
81 01 01 04 03 03 00 08  A7 19 85 17 09 4B 32 1E
5C 10 26 C0 EC FF 76 83  BE 2E 11 0A AA DA 6E 49
DA 76 69 A9 02 05 00
```

```
SEQUENCE  (5 elem)
    [1]  (6 byte) 0185ED549322
    [5]  (1 elem)
        NULL
    [6]  (2 elem)
        [1]  (1 byte) 01
        OCTET STRING  (3 byte) 030008
    [7]  (1 elem)
        [5]  (23 byte) 094B321E5C1026C0ECFF7683BE2E110AAADA6E49DA7669
    [9]  (1 elem)
        NULL
```

iCLASS SE Cards: SIO Advantages

- Cryptographically Bound to UID / CSN (using Key Derivation Function)
  - Does Not Protect Against "Magic" Cards with Changeable UID
- "Tamper Seal" Behavior
  - Data Alteration Corrupts the Signature, Invalidating the SIO
  - Data Encryption Protects the Contents of the SIO
- Data Security Independent of Card Technology
  - Chip Itself Does Not Know or Use SIO Keys
  - SIOs Can be Encoded with Customer-specific Keys



iCLASS SE Cards: PicoPass Redux

- PicoPass Was Still Cost-Effective
- Silicon Itself Was Not Broken at the Time, Just the Implementation
- Excellent Performance and Read Distance Compared to Other HF Cards
- New Data Model (SIO) and KDF (NIST SP 800-108)
- Differentiated by SE Marking on Card (SR for Transition)



CARD # SALES # / LINE # SIO

iCLASS SE Cards: PicoPass Limitations

- Limited Memory and Application Areas
- Replay Protection Requires Monotonic Counter (ePurse) to be Updated
- Some Versions of Reader Firmware Did Not Update ePurse
- PicoPass is EOL
  - HID Switched to Custom IC with PicoPass Emulation
- Reader Firmware Implementation Defects Allow MAC Authentication Bypass
  - Original Discovery by NVX, CVE-2024-41566
  - Check out NVX's Talk "iCLASS - Throwing Away the Keys" on Friday, August 9th RF Village at 13:30

iCLASS SE Cards: Seos

- Flagship Technology Launched in 2012
- Originally Branded "iCLASS Seos" Despite Being Completely Different Technology
- More Memory for Applications
- Enhanced Security (e.g. Random UID, Discrete Privacy and Access Keys)
- APDU-based Protocol Allows for Flexible Implementations
  - Physical Cards, BLE, Software, iOS, Android
- Original Variant Ran on Java Card
  - Was Expensive! — Approx $15 per Card
- Recent Cards Leverage Cost-Effective “Bare Metal” Implementation
  - Single-Application “Seos Essential” Available for $2.40-$3.75 Per Card

iCLASS SE Cards: Support for Other Technologies

- SIO design enabled card agnostic capabilities
  - Adds an additional layer of security to a cards built-in security mechanisms
- Future proof for when card technologies age and vulnerabilities are discovered
  - Can’t patch the silicon
- Card technologies (like barcodes) have regional dominance
- Many multi-application cards intend to be the one credential to rule them all, however successful implementations are rarely seen in the wild

SE Credential Encoding

Inspect

- Obtain Binding from Credential (CSN/UID)
- Calculate Diversified Write Keys

Encode

- Generate Unique ID and Calculate SIO Keys
- Encrypt and Sign Wiegand Data

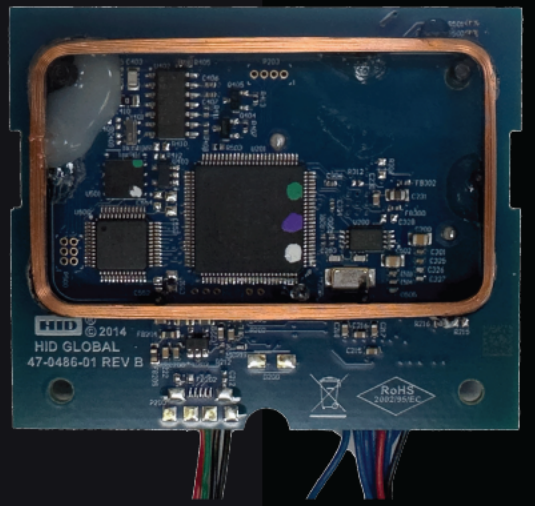
Write

- Authenticate to Card with Diversified Write Keys
- Write Encrypted SIO Data

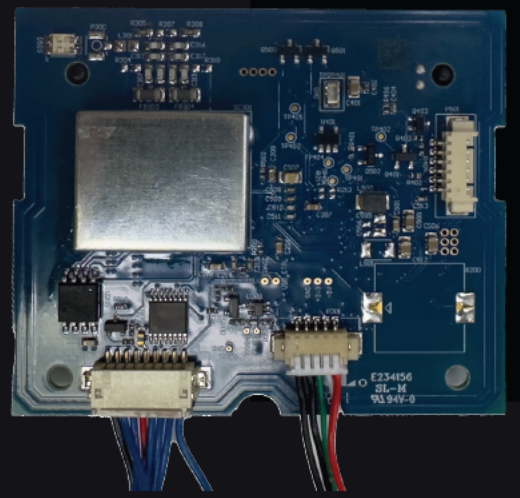
iCLASS SE Encoder: Secure Issuance

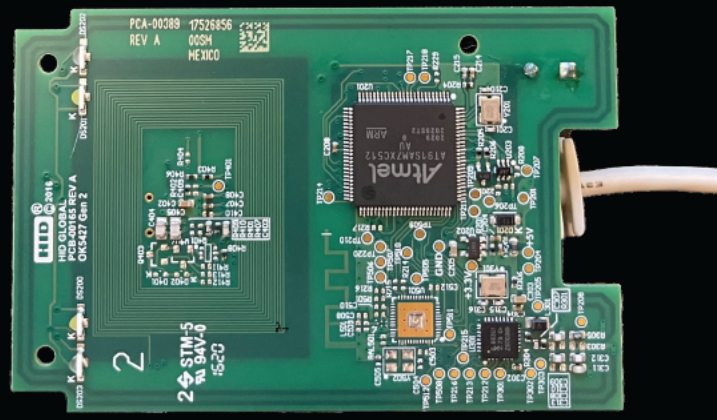
- CP1000D Desktop Encoder
  - Chassis Identical to OK5427
  - Different Internals

- Printer Encoder
  - Based on OK5127
  - Extra Connector for Printer Control



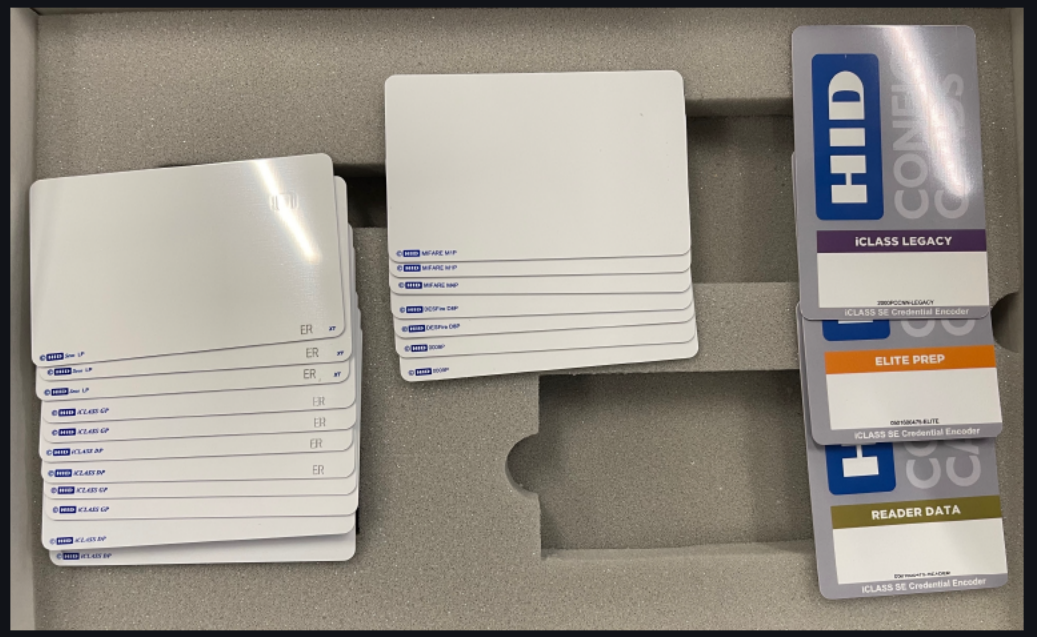






iCLASS SE Encoder: Capabilities

- Field-Encoding of Genuine HID and 3rd-Party Credentials
  - Blank “Encoder Ready” (ER) Cards
- Supports Reading/Writing Several Technologies:
  - Low Frequency (125KHz) ISOProx II
  - High Frequency (13.56 MHz)
    - Seos
    - NXP MIFARE DESFire EV1/EV2
    - iCLASS SE (PicoPass)
    - iCLASS Legacy (PicoPass)
    - NXP MIFARE Classic
- Supports Creating Custom Configuration Cards



iCLASS SE Encoder: Target Markets

Custom Keys

Credential Re-Encoding

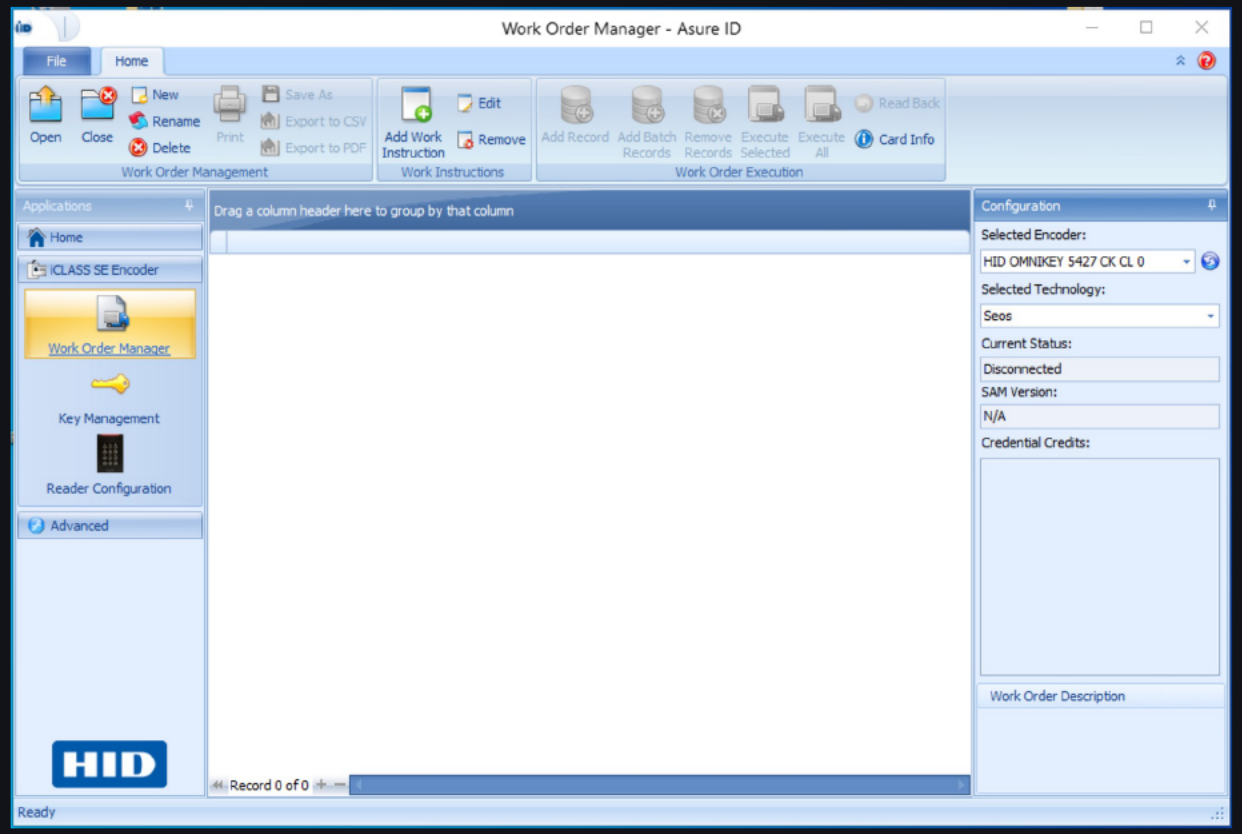
Customer-Generated Card Numbers

Custom Config Cards

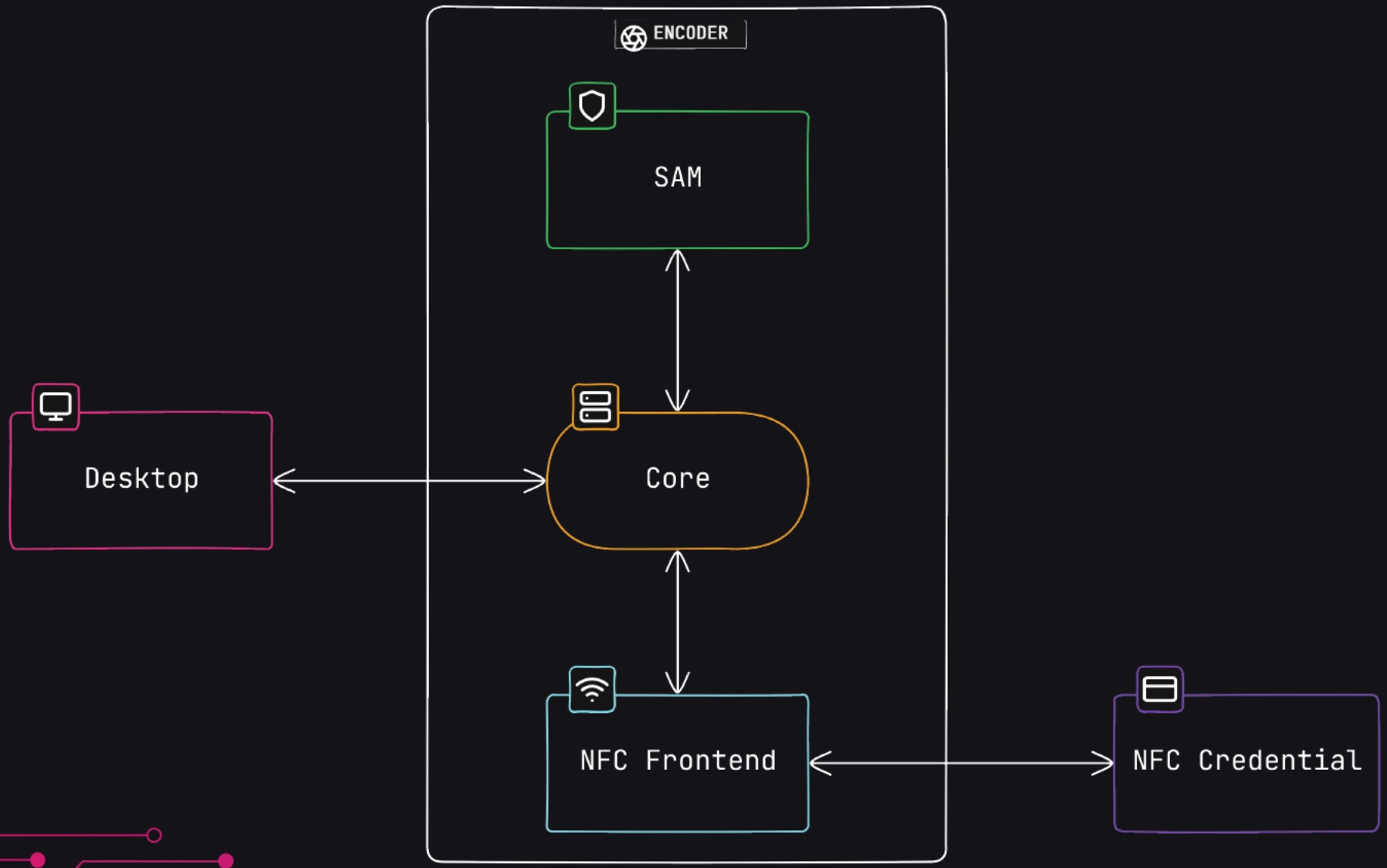
Integrator Lock-In

iCLASS SE Encoder: Asure ID

- Used Primarily for Badge Printing
- Supports Credential Encoding When Used with a CP1000D
- Asure ID SDK Can be Integrated into 3rd-Party Software



iCLASS SE Encoder: Internal Operation



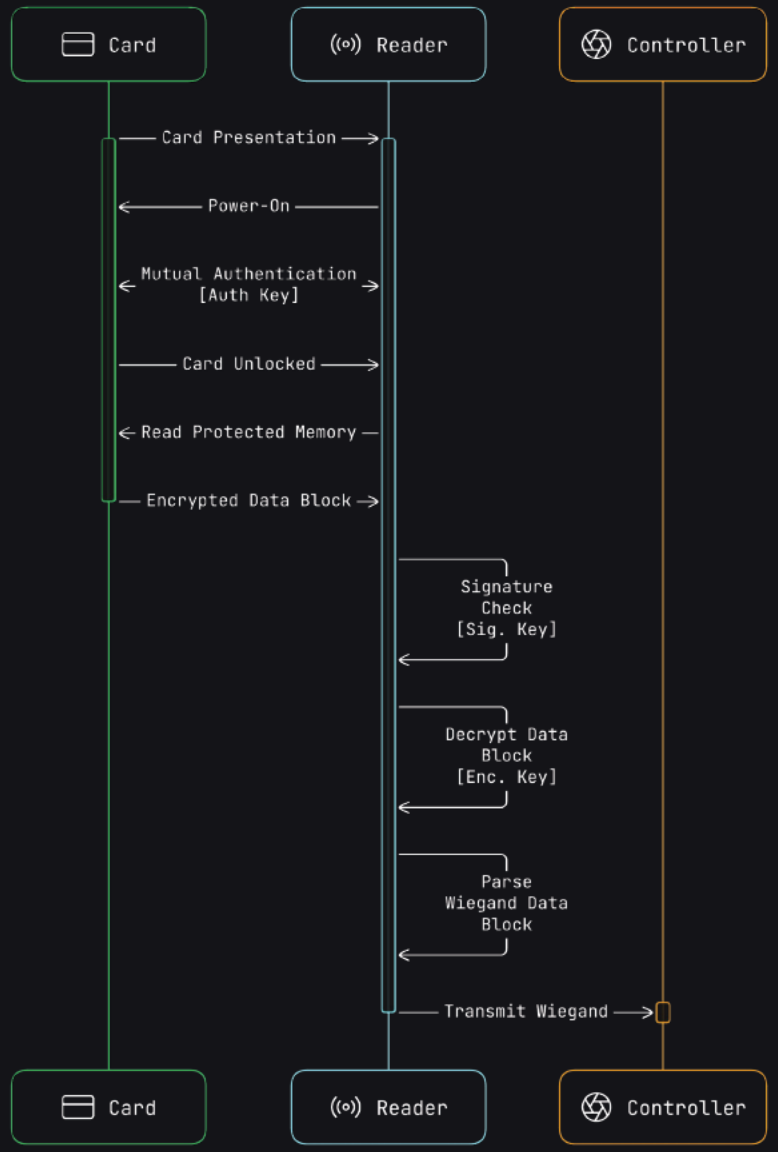
iCLASS SE Encoder: Extensibility

- CP1000D is Highly Customizable, Allows For
  - Standard Keys
  - Elite Keys
  - User-Generated Custom Keys
- Encoding Regulated by “Credit” System
  - Each Encoder Ships with a Limited Number of Credits
  - Credits are Uniquely Issued to an Encoder
  - Credits are Specific to a Particular Credential Technology
  - Credits Further Separated by Card OEM "Genuine HID” vs Third-party
  - Additional Credits Needed for Elite Variants

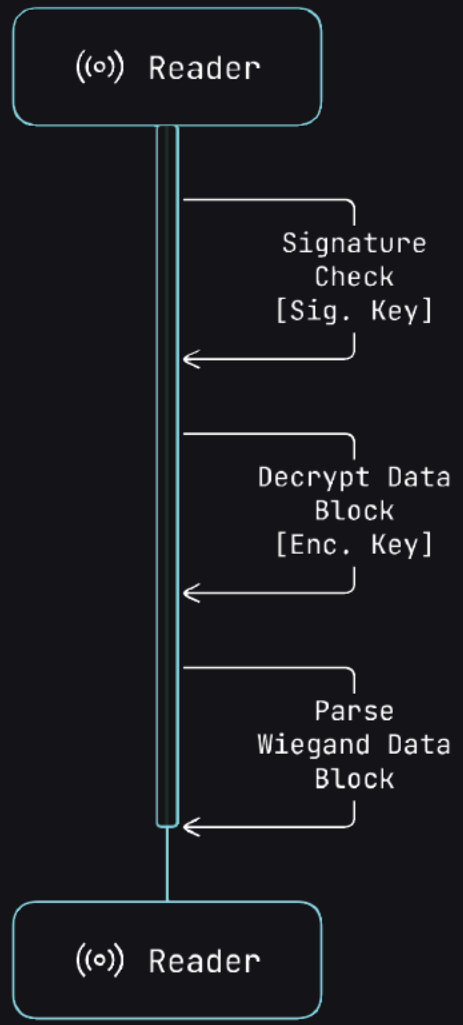
iCLASS SE Reader: Overview

- iCLASS SE Reader Platform is Generally “Rev. E” Hardware
- Readers May be Branded
  - iCLASS SE
  - multiCLASS SE (branding to indicate 125KHz hardware is present)
  - pivCLASS (branding to indicate support for PIV)
  - Omnikey
  - Third-Party
- Rev. E Ships in Different Sizes
  - Wall Switch (R40)
  - Keypad (RK40)
  - Mullion (R15)
  - Mini Mullion (R10)
  - Décor (R95)
  - PC-Attached (5427CK/5127CK/etc.)

iCLASS SE Reader: Decode Diagram



iCLASS SE SIO: Decode Diagram



- Has the data been modified?
- Is it bound to UID / CSN of card?
- Decrypt Wiegand Data.
- Prepare Wiegand Data for output to controller.

iCLASS SE Reader: Migration Modes

- “Migration Readers” Support Old and New Technologies
- Logistically Necessary for Successful Adoption by Customers
- Intended to Support Migration from Old to New Cards
- Most Common Example: multiCLASS SE Reader
  - Supports Prox, iCLASS Legacy, MIFARE Classic, DESFire, Seos
  - Swiss Army Knife of Card Readers!
  - Often Stocked and Installed by Default Because they “Always Work”

iCLASS SE Reader: Downgrade Attacks

- Card Reader Output is Credential-Agnostic
- Doesn't Matter What Credential Technology
  - Plaintext
  - Authenticated
  - Authenticated & Encrypted
  - Authenticated, Signed, & Encrypted
- It's Wiegand All the Way Down
- Allows for "Cloning" of Secure Credentials without "Fancy" Keys

iCLASS SE Reader: Embedded Modules

- Used by 3rd parties to embed a reader in their product
- Several interfaces and antenna options



iCLASS SE Reader: Config Methods

- It's all SNMPv3 over APDUs over NFC, BLE, OSDP, UART
- Near Field Communication (NFC)
  - Reader Manager
  - Configuration Cards
- Bluetooth Low Energy (BLE)
  - Reader Manager
- OSDP Manufacturer Specific Commands
- UART Debug Interface

iCLASS SE Reader: Config Use Cases

- Turning off legacy credential types
- Changing credential types
- Adjusting credential priority for multi-tech cards
- Change output format of keypad
- Switch from Wiegand to OSDP
- Reset OSDP key / config
- Adding new keys to reader
- Beep / LED configuration

iCLASS SE Reader: Configuration

- New method for re-configuring readers
  - Changing simple settings (LED Color, Beep profile)
  - Secure Loading of key material

- Uses iCLASS SE Configuration Card
  - Java Card (a.k.a. Artemis Updater Card)
  - One “credit” per reader configured
  - Retail / factory configuration cards have unlimited credits



iCLASS SE Reader: Configuration



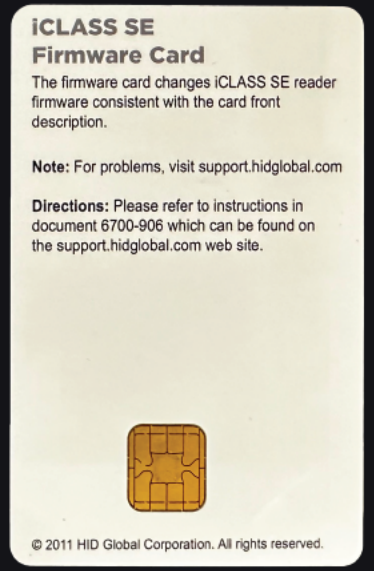
iCLASS SE Reader: Firmware Overview

- Core Firmware
- SAM Firmware
- Expansion Module / Plugin Firmware
- All are upgradeable

iCLASS SE Reader: Firmware Upgrades

- Same interfaces as Config updates
  - Debug / UART Interface
  - Firmware Update Cards (Artemis Updater applet)
  - Reader Manager (via BLE or NFC)







iCLASS SE Processor: Intro to SAM

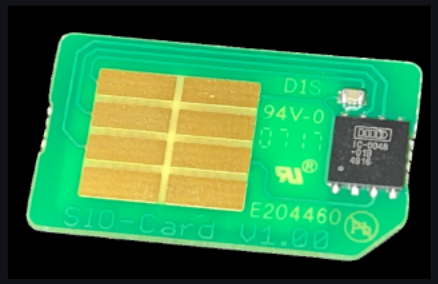
- Secure Application Module
  - Generally implemented on top of a secure microcontroller
- Black box to protect your secrets
  - Intellectual Property
  - Key material
- Resist advanced threats
  - Hardware protection in the silicon
  - Focused on protecting against fault injection techniques
- Common applications
  - EMV Credit/Debit Card Terminals
  - Other Smart Card Readers (e.g. public transit, physical access control, etc.)

iCLASS SE Processor: Hardware Capabilities

- HID IC-0048
  - Custom package containing a secure microcontroller
    - Infineon SLE88CFX4000P CC EAL 5+ Security Processor
  - How do you get to the SAM in a reader?
    - Depotting
  - What do you do once you've got it?
  - Make SAM take a bath and trip on acid
- ISO 7816 (Smart Card Interface)

iCLASS SE Processor: Package Types

- SAM on PCB (QFN)
- SAM on SIM (PVC)
- QFN SAM on a SIM adapter





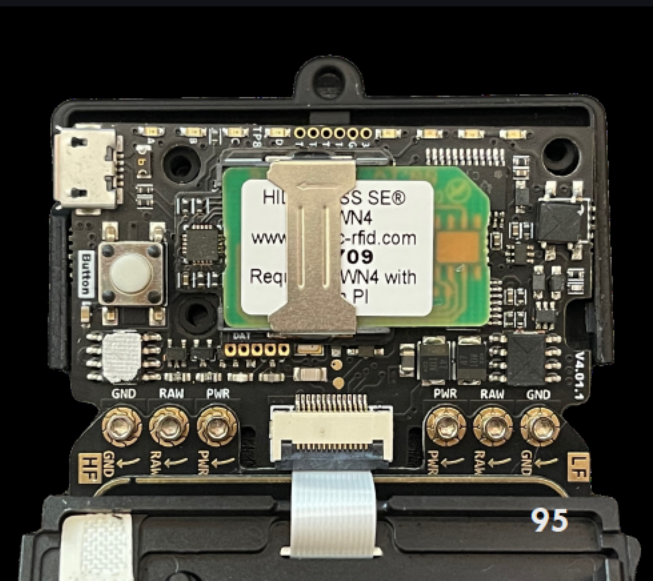
iCLASS SE Processor: Key Management

- All keys (excluding the SNMP keys) are stored as SNMP OIDs
  - Specifically are OIDs that are "write only"
  - Keys that are under HID's OID prefix can only be written by the HID Admin SNMP user

iCLASS SE Processor: Applications

- Official
  - Native readers
  - Third party readers
    - e.g. rf IDEAS WAVE ID, ELATEC, some biometric readers
  - Printers
    - Read-only
    - Encoder
- Unofficial
  - Flipper Zero (thanks Bettse)
  - Proxmark3 (thanks Iceman)





iCLASS SE Processor: Firmware

- Artemis SAM – Top Layer
  - Main SAM platform implementing interfaces for storage, etc…
  - Embedded .NET VM / Interpreter
    - Stripped down version of .NET runtime
    - Similar to the “Java Card” runtime / VM that powers many EMV cards
- SNMP Loader – Middle Layer
  - Alternate boot target for updating/replacing the Artemis SAM (top layer)
- Mini Loader – Base layer
  - Alternate boot target for installing/updating the SNMP Loader (middle layer)

iCLASS SE Processor: Chain of Trust

- Mini Loader – Base Layer
  - Loaded onto a virgin / vanilla Infineon SLE88
    - Allows for securing the SLE88 native firmware/debug mechanisms
  - Used in the factory to load SNMP Loader firmware
  - Could be used in the field to update SNMP Loader firmware
- SNMP Loader – Middle Layer
  - Used to securely load the Artemis SAM firmware via signed and encrypted SNMPv3 messages
- Artemis SAM – Top Layer
  - Verifies signatures on loaded Assemblies

Time to Hack!

Reconnaissance: Unravel the Unknowns

- Unknown: Where are the master keys?

- Well, what operations require master keys?
  - Encoders must calculate a diversified key to write a credential…
    - Therefore, **encoders** have master keys
  - Anything that reads must calculate a diversified key to read a credential…
    - Therefore, **readers** have master keys
- So, how do these keys get to encoders & readers?
  - Encoders are given keys using ISE files
    - A file specific to an individual encoder containing encrypted and signed SNMPv3 messages that load keys
  - Readers are given keys using Configuration Cards (a.k.a. Artemis Updater Cards)
    - A smartcard that can generate SNMPv3 messages to load keys in a target reader

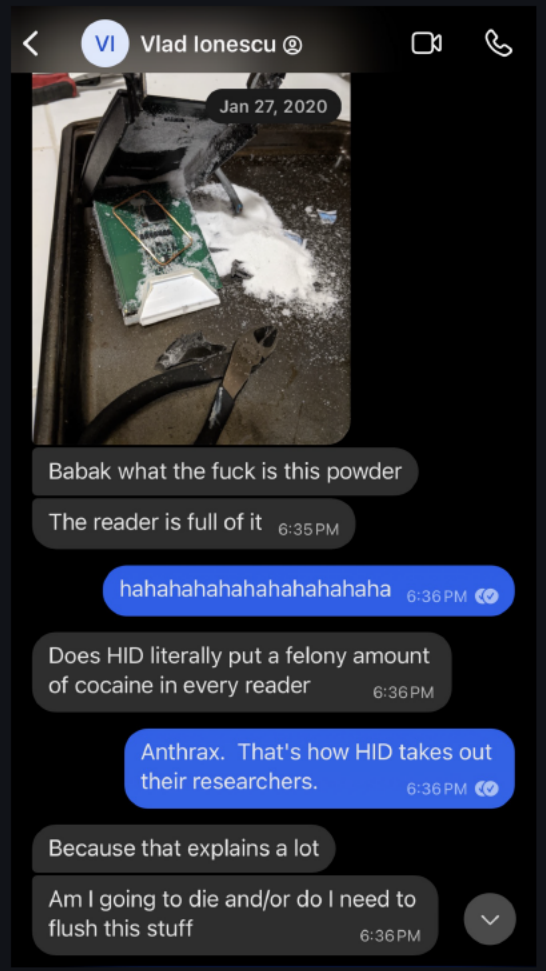
Reconnaissance: Identify Targets

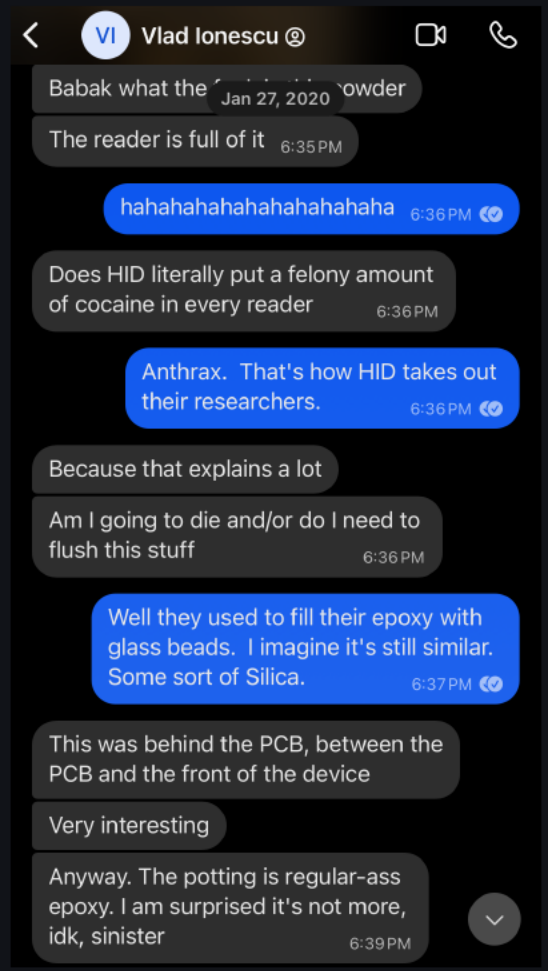
- Anything with Keys
  - Standard Readers
  - Desktop USB Readers
  - Embedded Readers
  - Desktop Encoders
  - Embedded Encoders
  - Configuration Cards
- Common Denominator: iCLASS SE Secure Processor (SAM)
  - One exception… Configuration Cards

Reconnaissance: Define Approach

- How do we turn unknowns into knowns?
- Hardware
  - Get inside/Take it apart, de-potting/conformal coating
  - Identify Components, decap custom packages
  - Enumerate interfaces between components
    - Sniff interfaces with Logic Analyzer
- Firmware
  - Acquire / recover unencrypted firmware
  - Unpack, disassemble, and decompile
- Software
  - Acquire, Unpack, Deobfuscate, Disassemble, Decompile

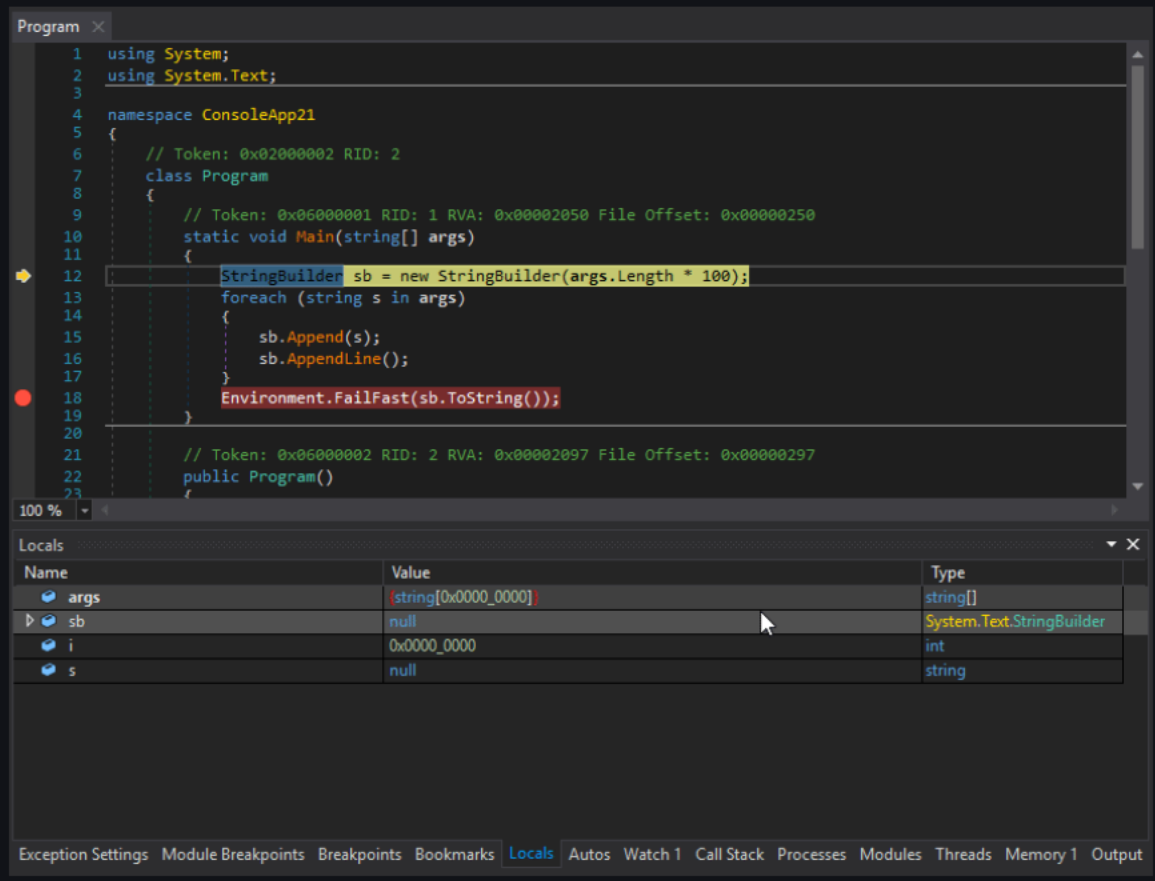
Reconnaissance: Define Approach





Reconnaissance: R/E Toolkit

- What tools did we use for this process?
  - Jetbrains DotPeek / dnSpy (.NET)
  - Jadx/Jeb (java)
  - Saleae Logic Analyzer
  - Ultrasonic Knife
  - Tiny wires
  - Rework Station
  - Sharp knife
  - Blood, Sweat, Tears
  - Patience
  - Neurodivergent Persistence



Reconnaissance: FCC Research

- For RFID readers (and other RF devices), companies must file with the FCC
- The FCC publishes these filings online
  - Treasure trove of info on both existing and new / unreleased devices
  - Information on new devices can be limited due to Short Term Confidentiality
- Official search (Bad UI, No Ads)
  - https://www.fcc.gov/ecfs/search/search-filings
- Unofficial Search (Better UI, ALL OF THE ADS)
  - https://fccid.io/ and https://fcc.report/ (Ad-Blockers Needed)

Reconnaissance: FCC “Leaks”

Apple Submits Mystery 'Wireless Device' With Bluetooth and NFC to FCC

Friday September 23, 2016 9:26 am PDT by Joe Rossignol

Earlier this week, we spotted an Apple filing for a nondescript "wireless device" pass through the FCC ID database. Apple characteristically requested permanent confidentiality for most of the documents in the filing, including photos, user manuals, and schematics, so the entry largely remains a mystery.

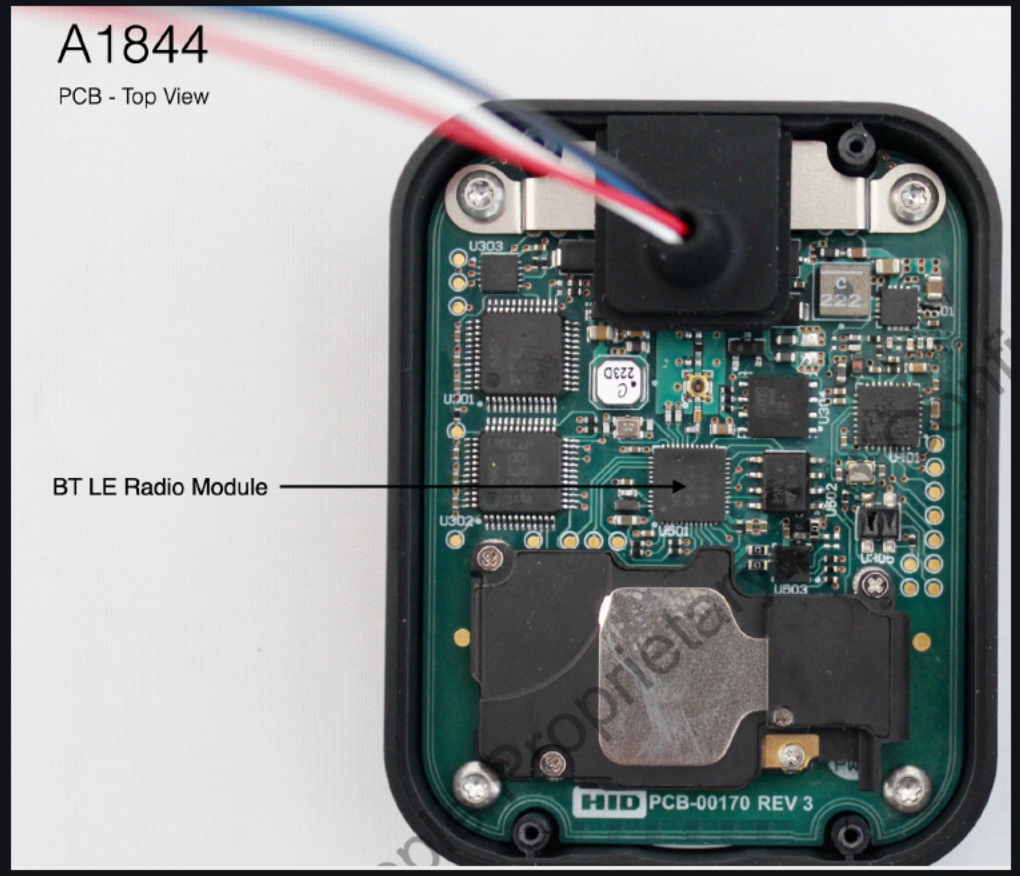
What we do know is the device has a model number of A1844, which does not line up with any existing Apple products. A regulatory label shows the device has two Torx screws on the back plate of the device, which appears to have at least two slightly curved edges. The device has an electrical rating of 5.5V to 13.2V.



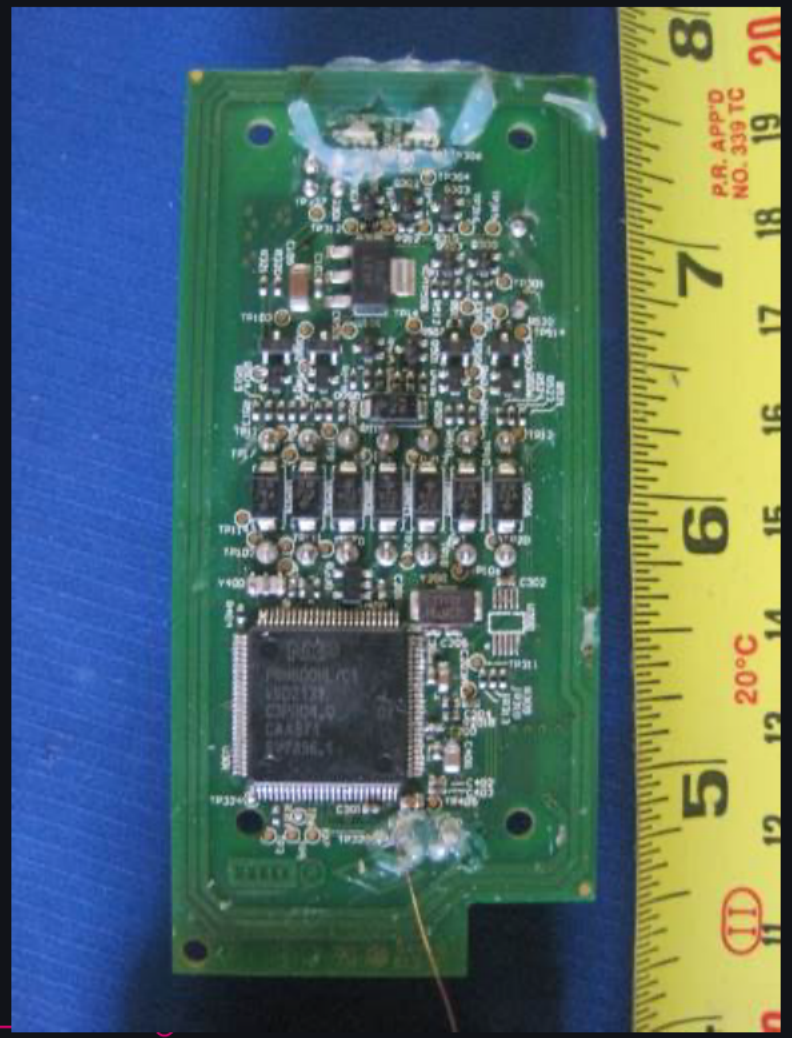
Test reports completed by UL Verification Services also reveal the wireless device has Bluetooth and NFC, although Wi-Fi is not mentioned.

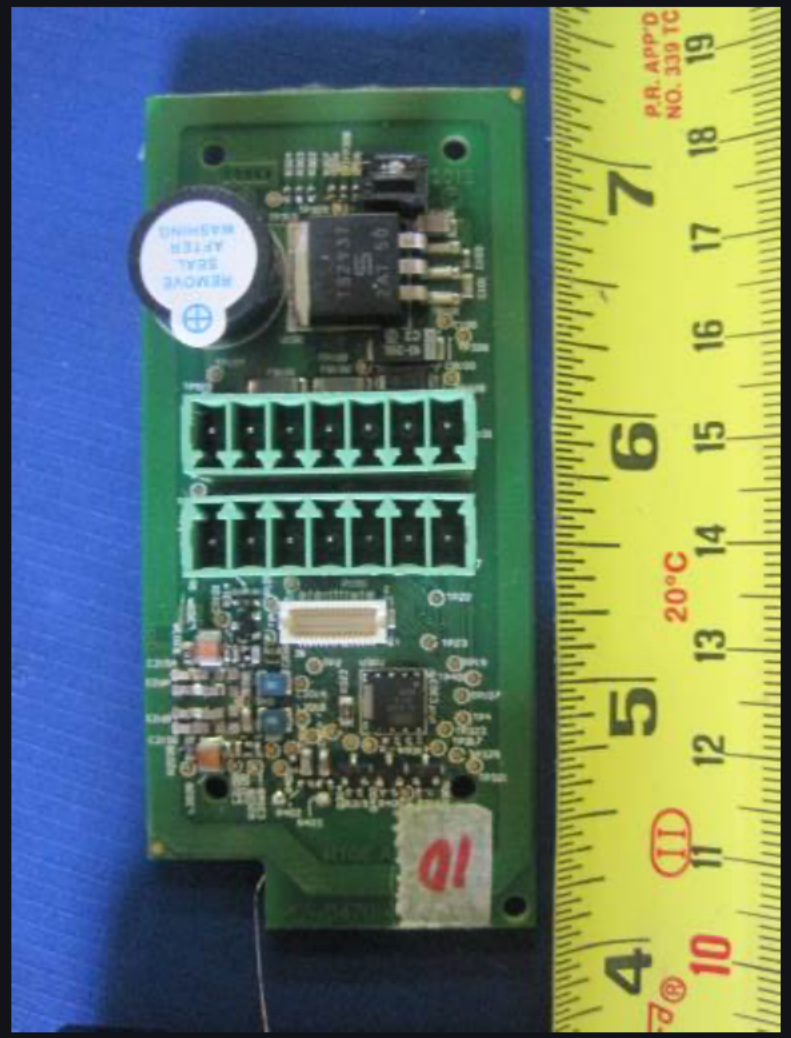
Some websites have speculated the filing could represent a new Apple TV, but the device appears to be smaller based on the artwork -- although there are no exact measurements for scale. The device also has oddly specific regulatory text etched directly on the back of the device, including a wiring guide, which would be uncharacteristic of Apple to include on the exterior of a consumer-facing product.

Perhaps, then, the wireless device is for internal use. Back in 2014, an FCC filing revealed Apple's first-party iBeacon hardware, for example, which the company uses in its retail stores. Originally introduced at WWDC 2013, iBeacon technology enables iOS devices to communicate with transmitters via Bluetooth LE in order to deliver relevant information to apps and services when a user is nearby.



Reconnaissance: FCC Research

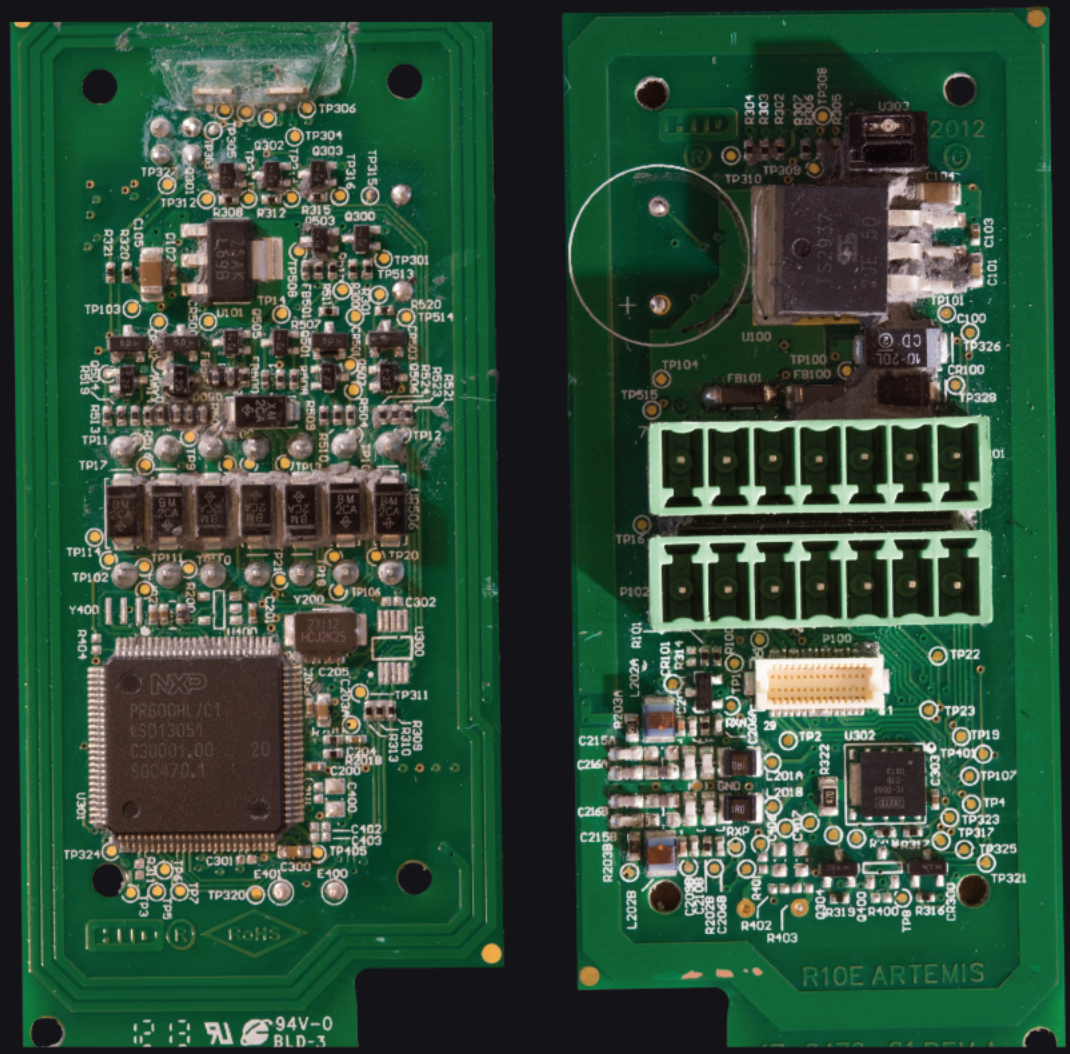




Reconnaissance: Reader Depotting

- Things We Tried
  - Cutting
  - Acetone
  - Boiling
  - Hot Air
- What “Worked”
  - Hot Air Embrittled the Epoxy
  - Heat Up Epoxy Until Almost Smoking
  - Hack Away with Mechanical Tool

Reconnaissance: Reader Depotting

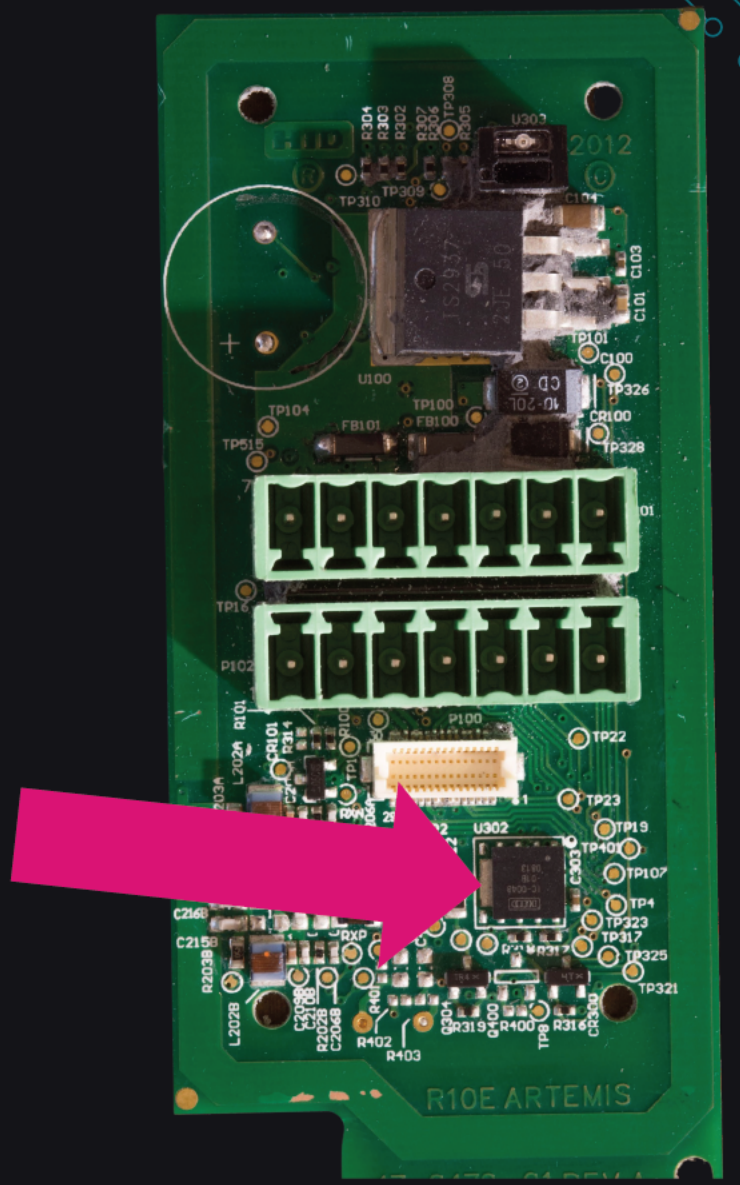


(Photo Credit: 0xFFFF)

Reconnaissance: SAM

- Protects Secrets Things!





(Photo Credit: 0xFFFF)

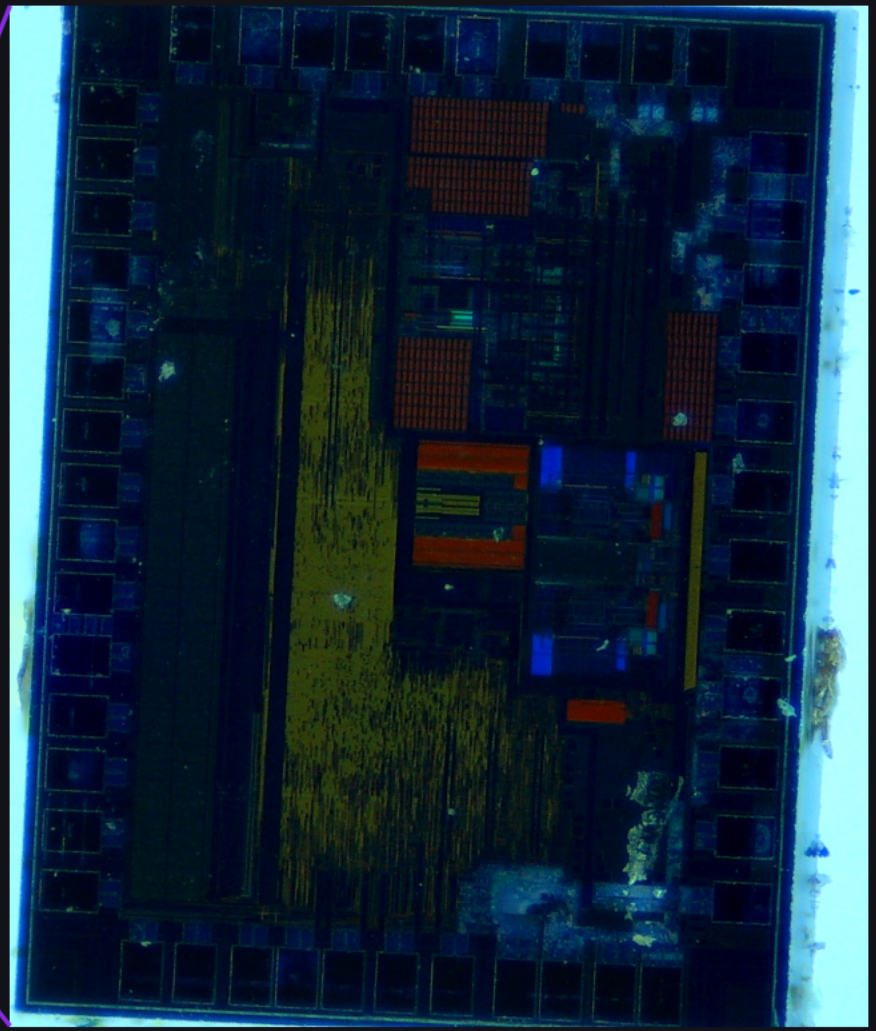
iCLASS SE Reader: HF Frontend



PR600

LPC1227

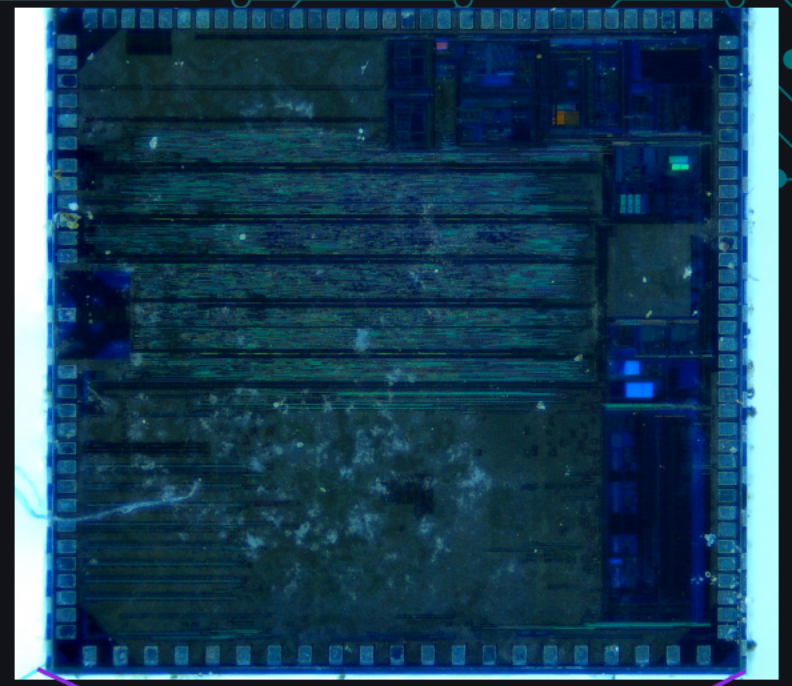
CLRC663



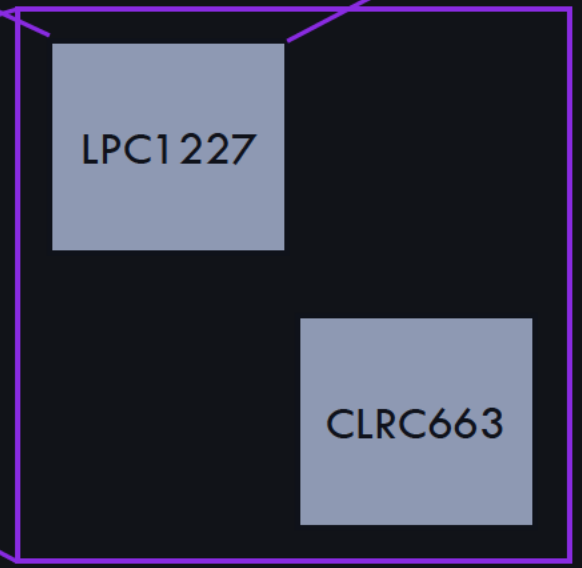
(Left & Right Photo Credit: 0xFFFF)

iCLASS SE Reader: Core

- Routes messages between components
  - SAM (ISO7816)
  - HF Frontend
  - Interfaces
    - Wiegand
    - UART
    - I2C
  - Peripherals
    - LEDs
    - Piezo (Beeper)
    - Keypad







(Top Right & Bottom Left Photo Credit: 0xFFFF)

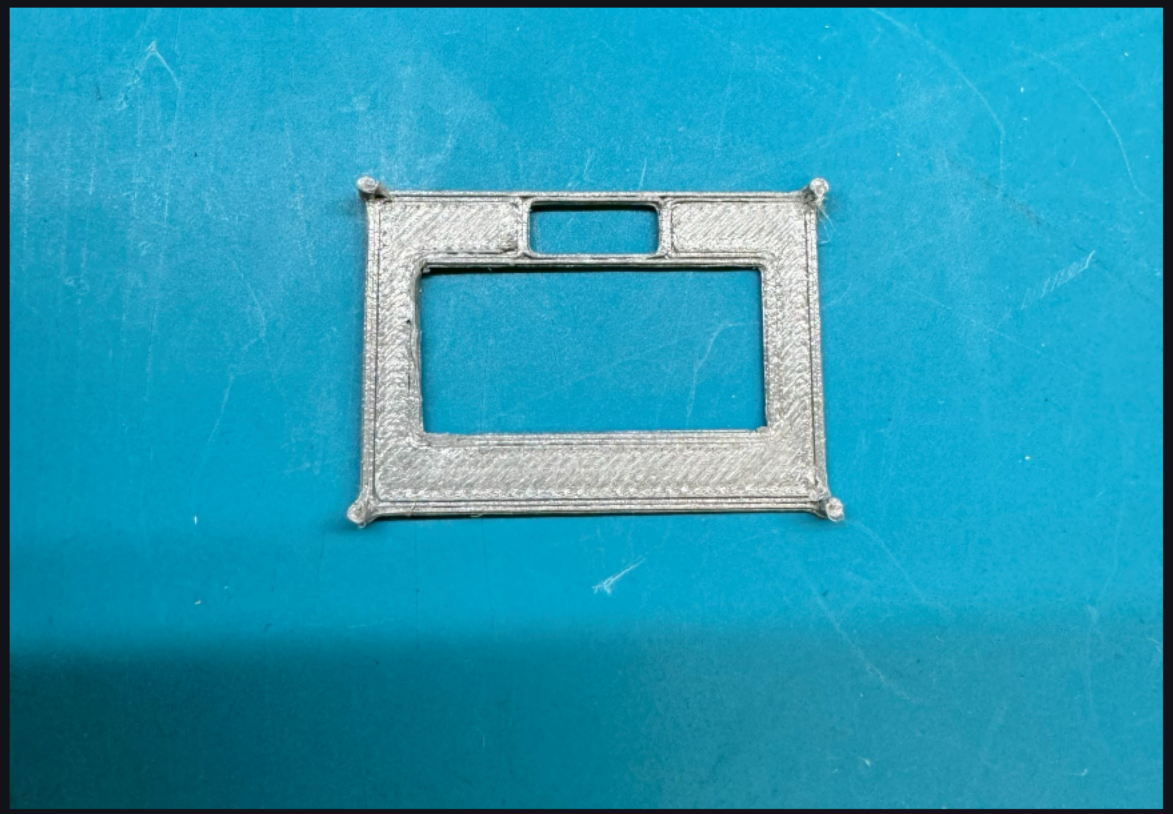
Reconnaissance: SAM Harvesting





Official SAM Extraction Kit v1.0

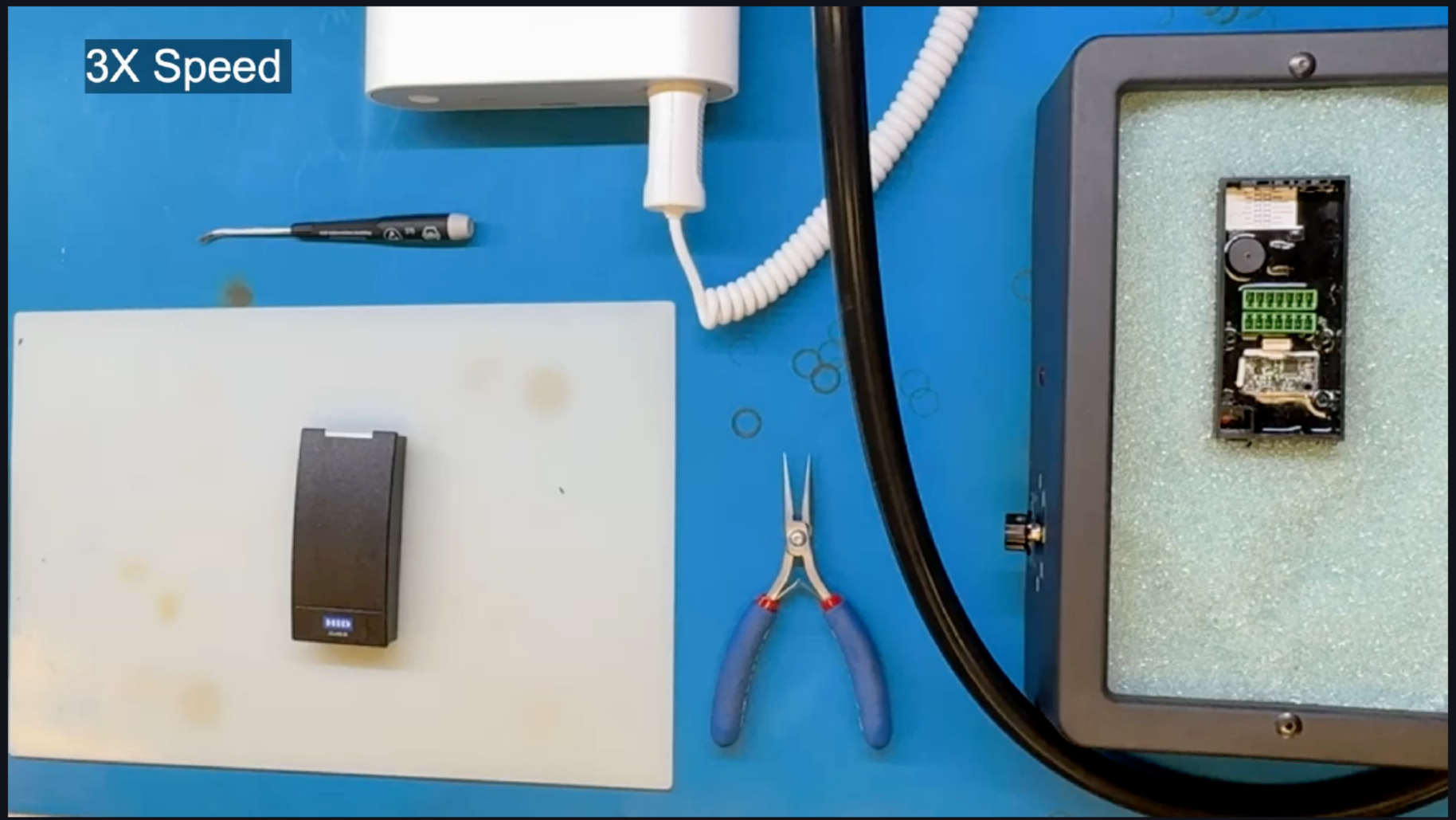
Reconnaissance: SAM Harvesting





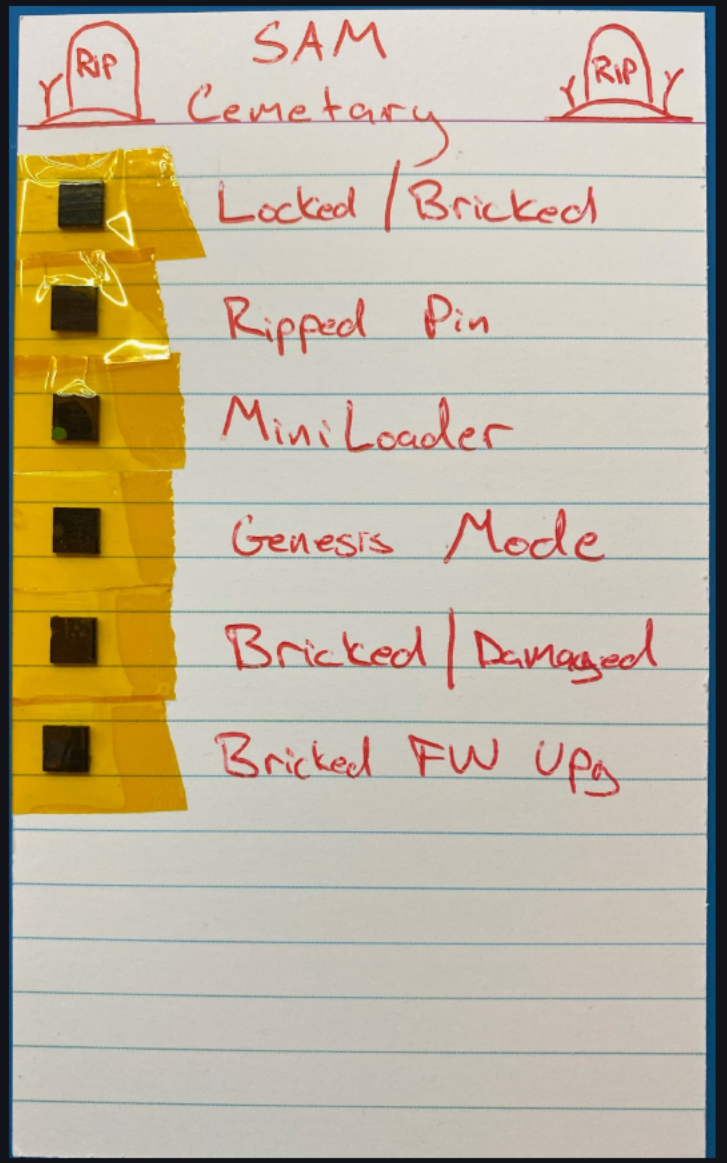
Official SAM Extraction Kit v1.1

Reconnaissance: SAM Harvesting



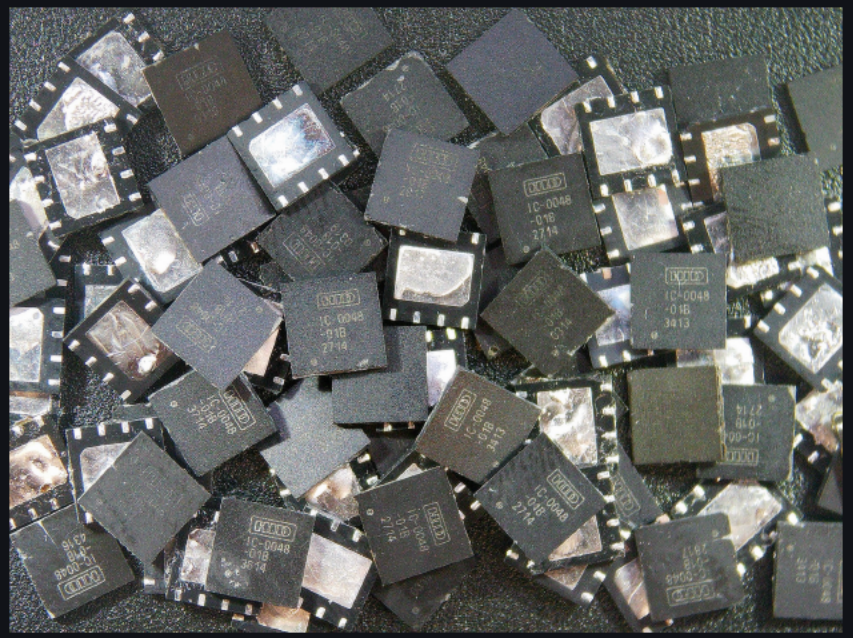
Reconnaissance: Mistakes

- Cooked SAMs
  - Epoxy Sinks Heat REALLY Well
  - SAMs Easy to Overheat
- Physically Damaged SAMs
  - Insufficient Removal of Surrounding Epoxy
  - Insufficient Heating
- Ripped Pads
  - Insufficient Heating
  - Excessive Force

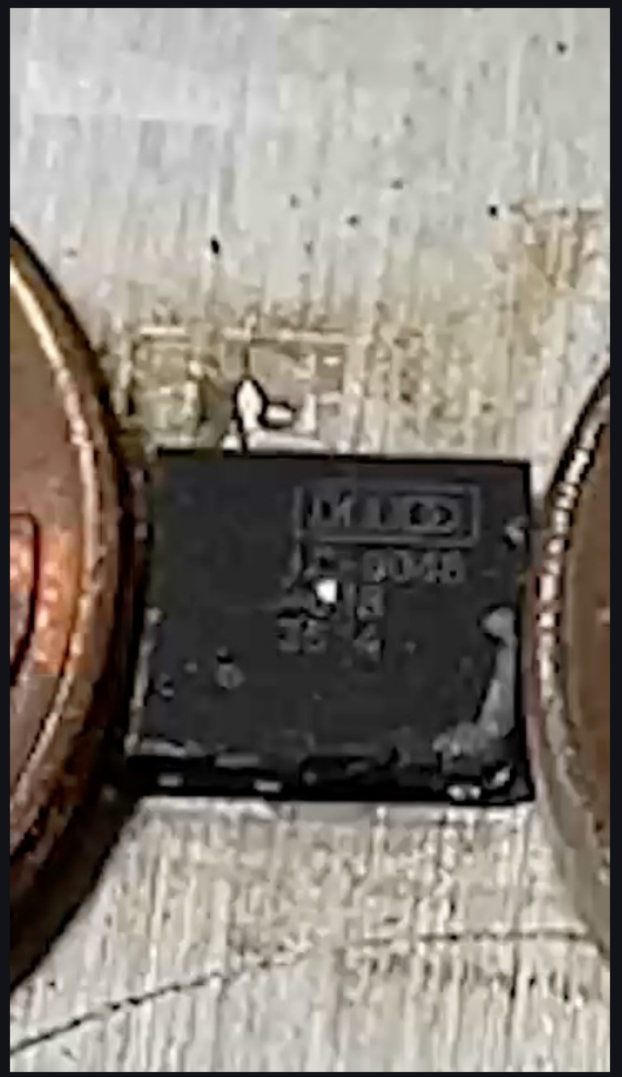


Reconnaissance: What Worked

- Choosing the Right Reader
  - Find SAMs with Minimal Epoxy, Potting, or Coating
- Debug / Expansion Port Positioning Found to be Critical
  - Alignment Frame for Expansion Module Created Air Pocket in Epoxy
  - FCC Photos Showed Different SAM Position for KP vs. Non-KP Reader
  - SAM Located in Air Pocket in Non-KP Readers!
- Cardboard “Jig” + Ultrasonic Knife = GREAT SUCCESS



Reconnaissance: Lasers!



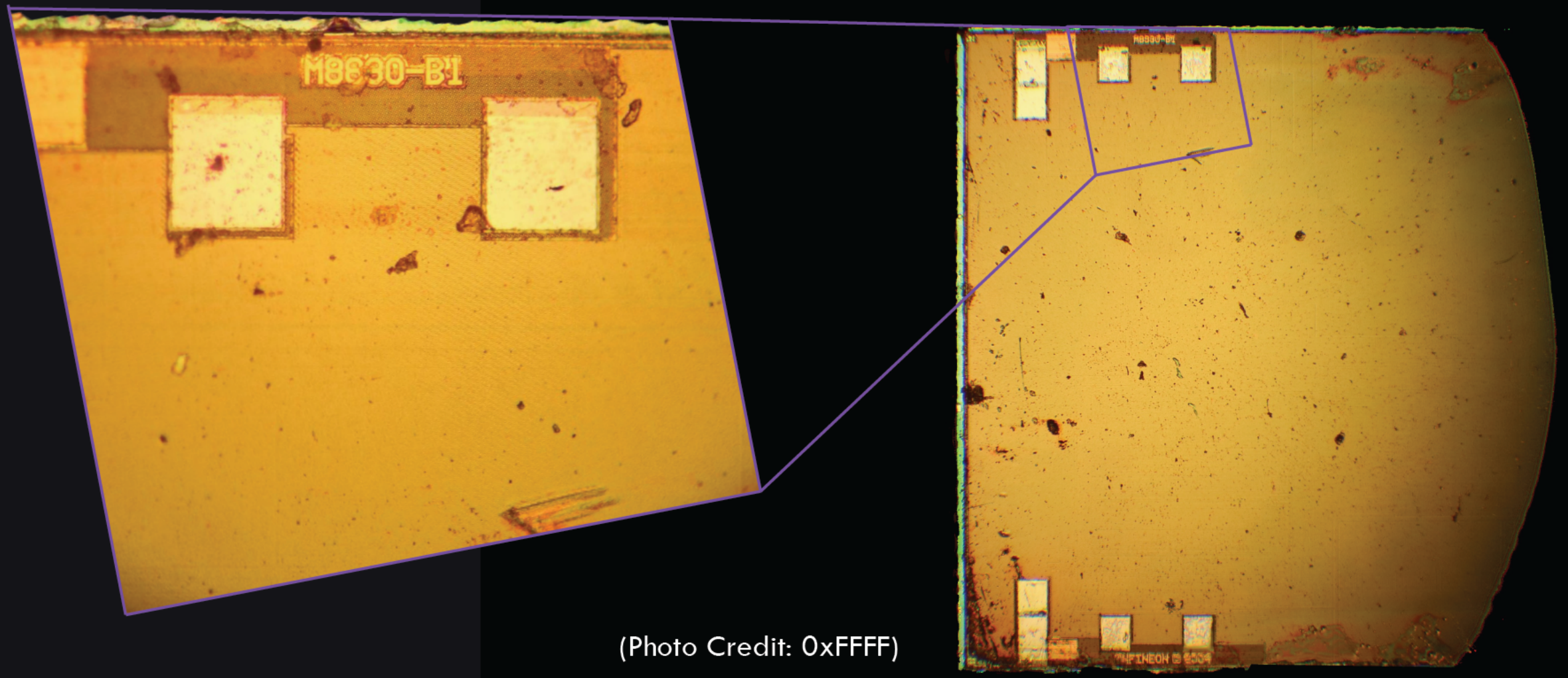
Reconnaissance: Die Shots





(Photo Credit: 0xFFFF DIY microscope)

Reconnaissance: Die Shots



(Photo Credit: 0xFFFF)

Reconnaissance: High Level R/E Overview

- Areas We Focused Reverse Engineering Efforts
  - Inter-Chip Protocols
    - Debug Ports
    - Tapped “Backpacks”
    - Tapped Core and SAM Communication
  - Firmware Packages
    - Reader Firmware Upgrades
  - Software Packages
    - Public-Facing Software
    - Publicly Available Documentation

Reconnaissance: Architecture Overview

- Core
  - MCU that acts as a "router" relaying messages from one component to another
- SAM
  - Secure microcontroller that handles all key material and cryptographic operations
  - Also "runs the show" for standard PACS wall reader operations, in what is called "Autonomous Mode"
- HF Frontend
  - Used for talking to High Frequency RFID credentials
- LF Frontend
  - Used to talking to Low Frequency RFID credentials on readers with LF hardware
- Sidecar
  - Used for expansion modules to support BLE, OSDP, Magnetic Stripe reader, etc.

Reconnaissance: Artemis

- Artemis is the codename for the iCLASS SE Reader Platform
- Mentioned on PCBs, in HID binaries / DLLs, etc.
- Includes:
  - Artemis SAM (SLE88)
  - Artemis Reader (SLE88 SAM + LPC1227 Core)
  - Artemis Protocol (serial protocol used for communication between reader components)
- Does not include:
  - OMNIKEY readers that contain an Artemis SAM
    - The SAM is Artemis, but the OMNIKEY reader platform is separate



(Photo Credit: 0xFFFF)

Reconnaissance: SNMPv3

- Simple Network Management Protocol (SNMP)
- Managed Device is the thing that can be inspected/configured by SNMP
- Management Information Base (MIB) defines objects and their ACLs
- Object Identifier (OID) is a hierarchical identifier assigned to a variable within the MIB
- Request Type (PDU)
  - GetRequest - retrieve the value associated with an OID
  - SetRequest - update the value associated with an OID
- Version 3 (the secure one)
  - Engine ID (a unique identifier assigned to each Managed Device)
  - User Security Model (USM) defines Authentication and Privacy mechanisms

Reconnaissance: SNMPv3

- Efficient, Stateless, ASN.1 based binary protocol (important considering the transport layer)
- Don't reinvent the wheel, it's a well-designed interface to a Key/Value store:
  - where the Key is an OID associated with a configuration item, key, or key reference
  - and the Value is the actual configuration data, key material, or key reference OID
- Meets Confidentiality, Integrity, and Authentication requirements
- Managed Devices (SAMs) each have a unique Engine ID
  - Ability to generate messages that can only be used on a specific SAM
  - SAMs receive diversified administration keys and don't need the master
- USM Users used as Roles
  - OEMAdmin - Intended for use by the end user or integrator
    - Can view configuration and change some settings (LED Color, etc…)
  - HIDAdmin - Higher level user with access to more things

Reconnaissance: Roles

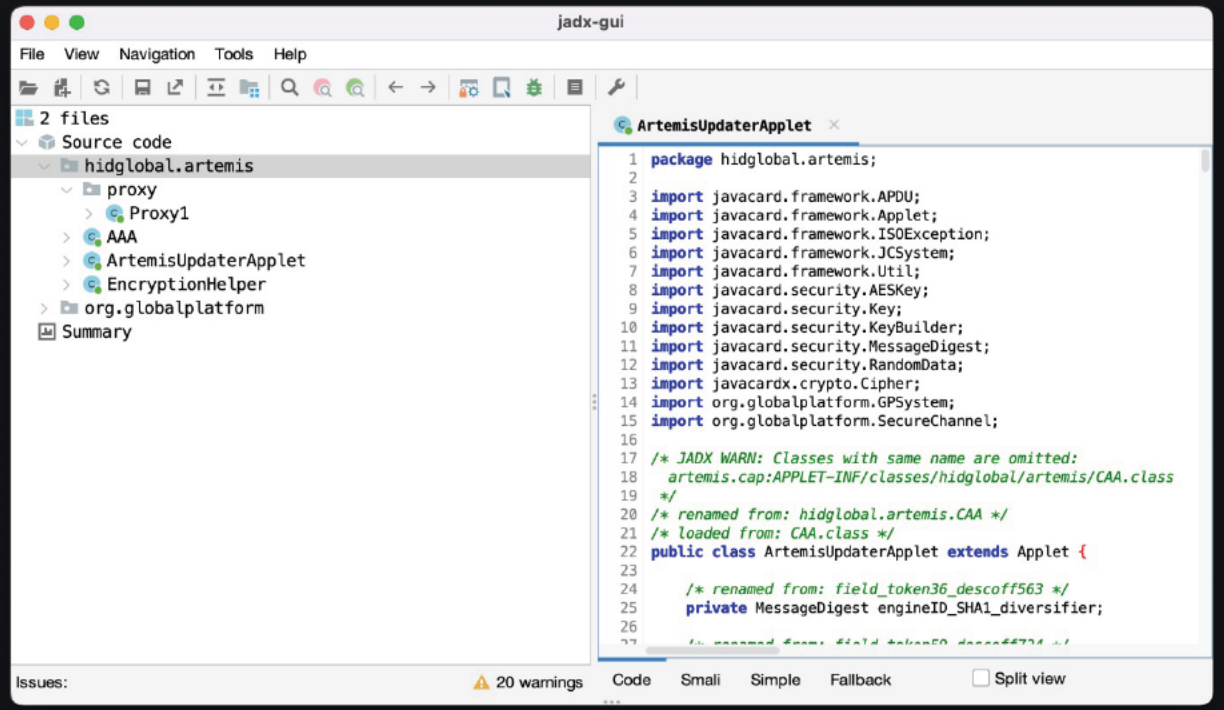
- SNMPv3 User Security Model (USM) used for privilege separation
  - OEM Admin
    - Can retrieve (GetRequest) configuration and change (SetRequest) some settings:
      - LED Color
      - Wiegand/OSDP
      - Datamodels (Enabled credential technologies)
    - Cannot retrieve/GET key values
  - HID Admin
    - Can retrieve and change all configuration settings
    - Cannot retrieve/GET key values

Reconnaissance: Java Cards

- Java Card Versions: JCOP2 vs. JCOP3 vs JCOP4
- Global Platform for Card Lifecycle Management
  - Security protocols and mechanisms (Secure Channel, encryption, authentication)
- Subset of Java language for operating within resource constraints and security applications
- Java Cards run “Applets”
- Example Use Cases of Java Card
  - EMV Cards (Credit/Debit bank cards)
  - Subscriber Identity Module (SIM)
  - PIV Applications
  - Virtual Credentials
  - Firmware Updates
  - Secure Memory Storage

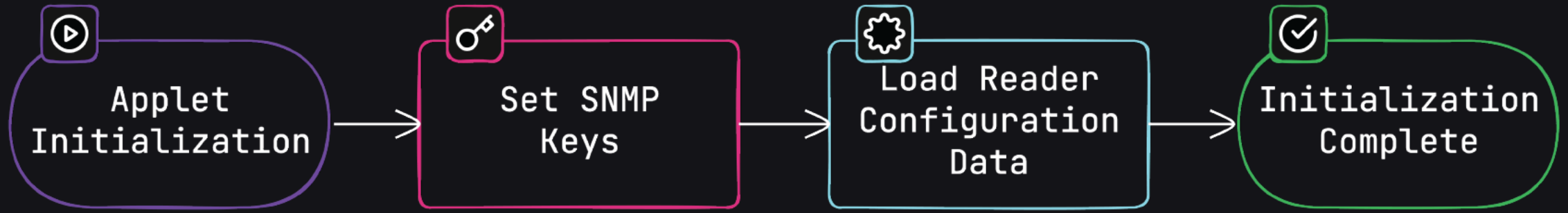
Reconnaissance: Config Cards

- Reverse engineered applet to understand operations and identify vulnerabilities
- Same applet can be initialized as:
  - Configuration Updater
  - Firmware Updater
  - SNMP Keyroller
  - Assembly Loader
- Applet enforces use of transport security
  - Global Platform Secure Channel
  - AES Key wrapping



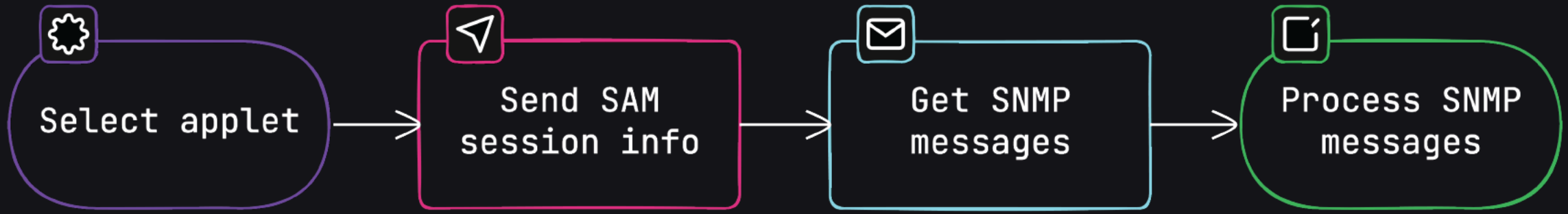
Reconnaissance: Config Card Operations

Encoding:



Reconnaissance: Config Card Operations

Interaction with a reader:

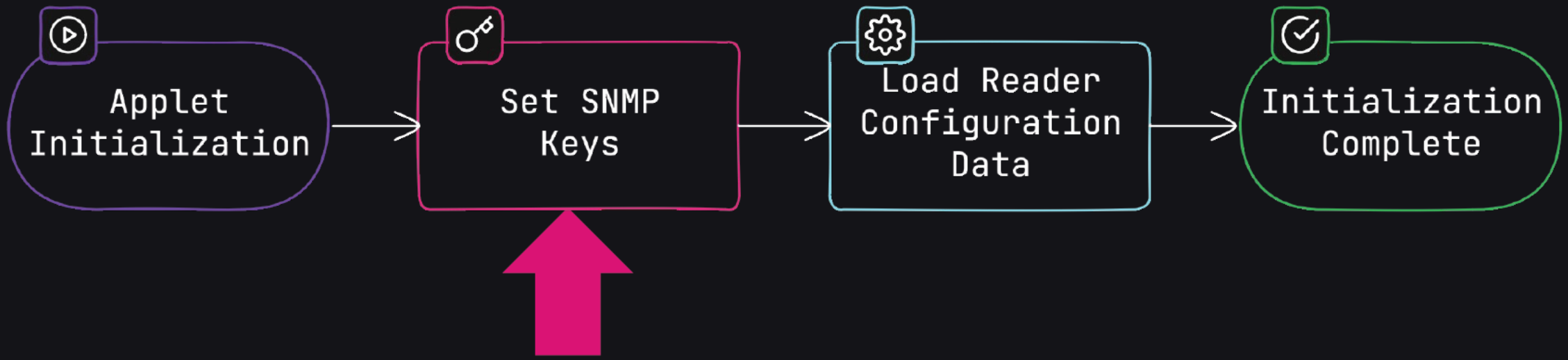


The Heist: The Good Part!

The Heist

Config Card Attack

The Heist: Config Card Encode

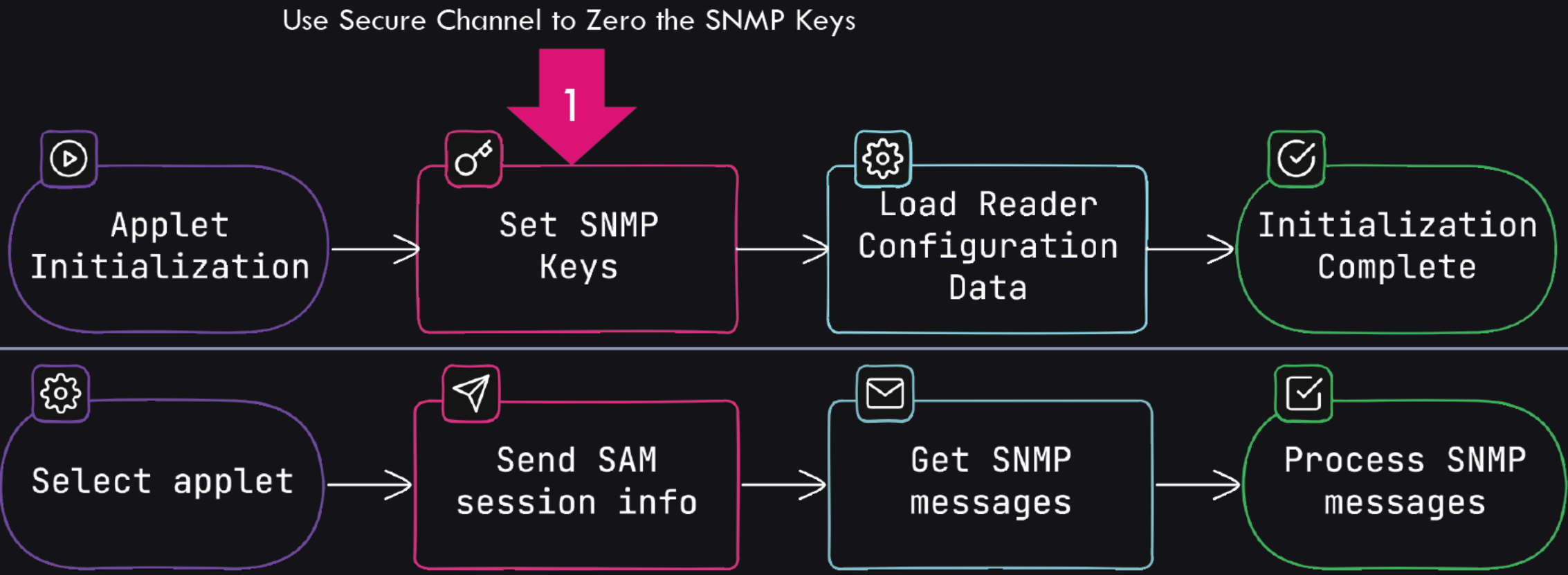


The Heist: Config Attack Feasibility

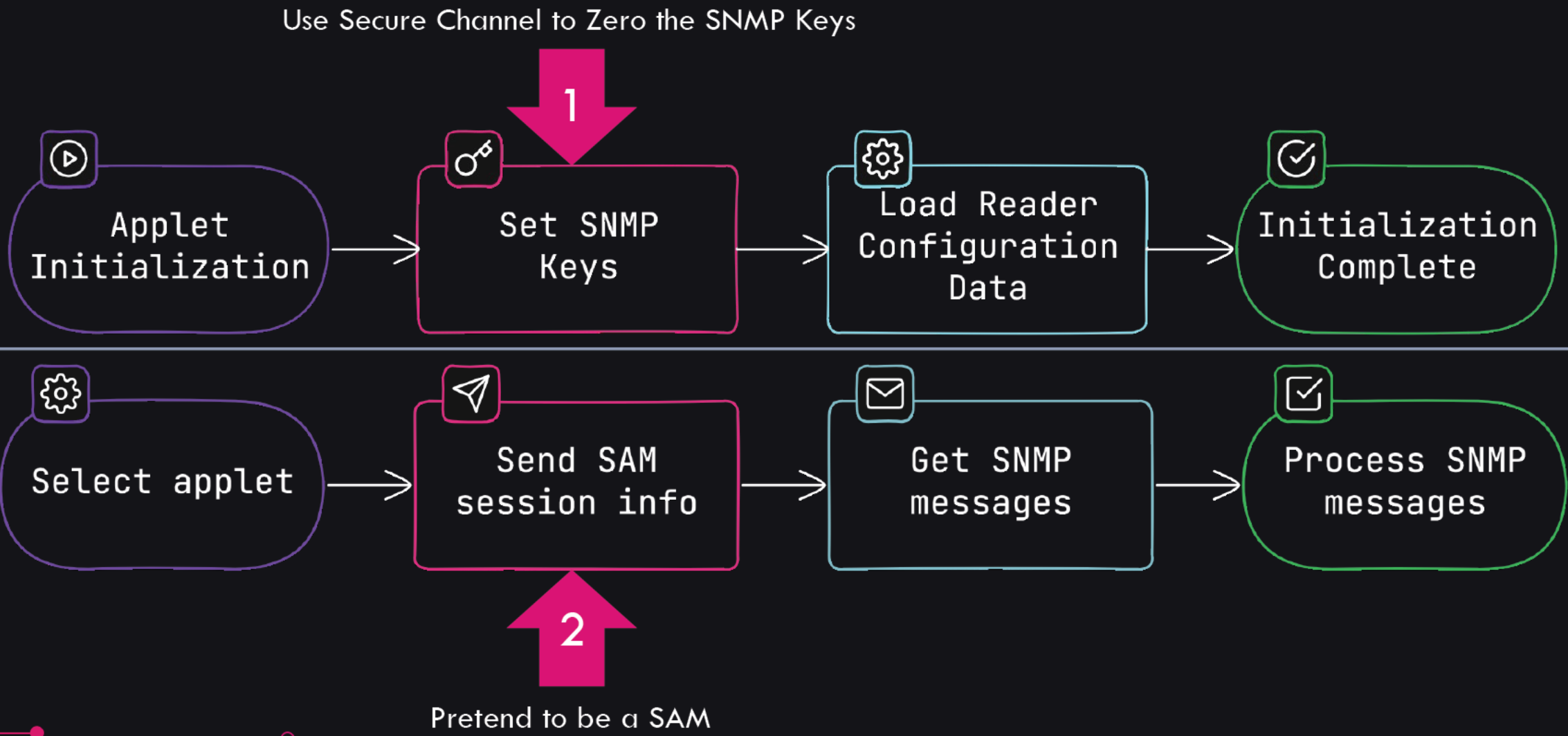
- GlobalPlatformPro tool for interacting with Java Cards (thanks Martin Paljak)
- Establish a Secure Channel with the card



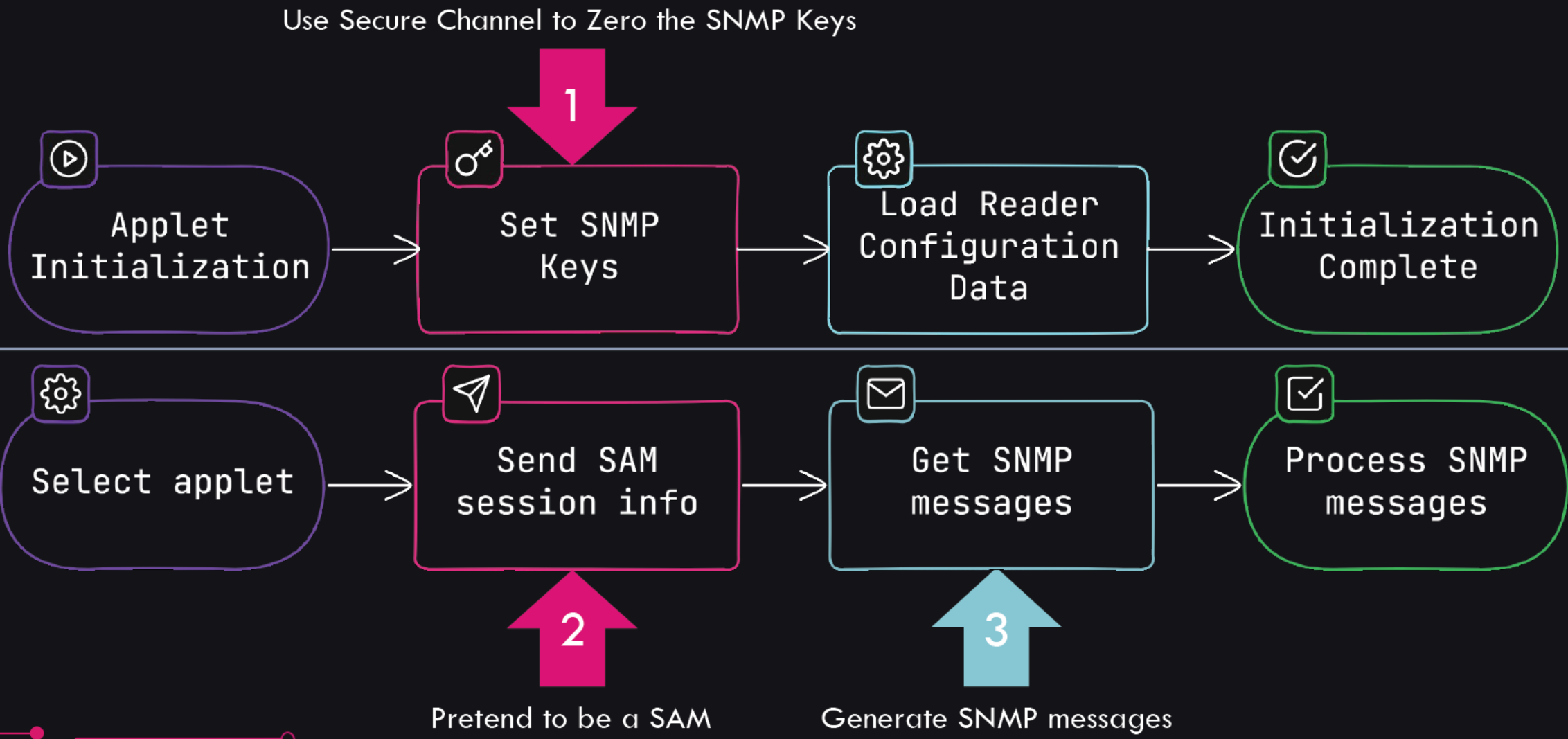
The Heist: Config Card Attack Chain



The Heist: Config Card Attack Chain



The Heist: Config Card Attack Chain



The Heist: Execute Config Card Attack



The Heist: Config Card Impact

- Retrieval of data stored on Config Cards
  - Configuration Items
  - Firmware
  - Keys

- Other Issues Identified
  - Memory
  - Debug

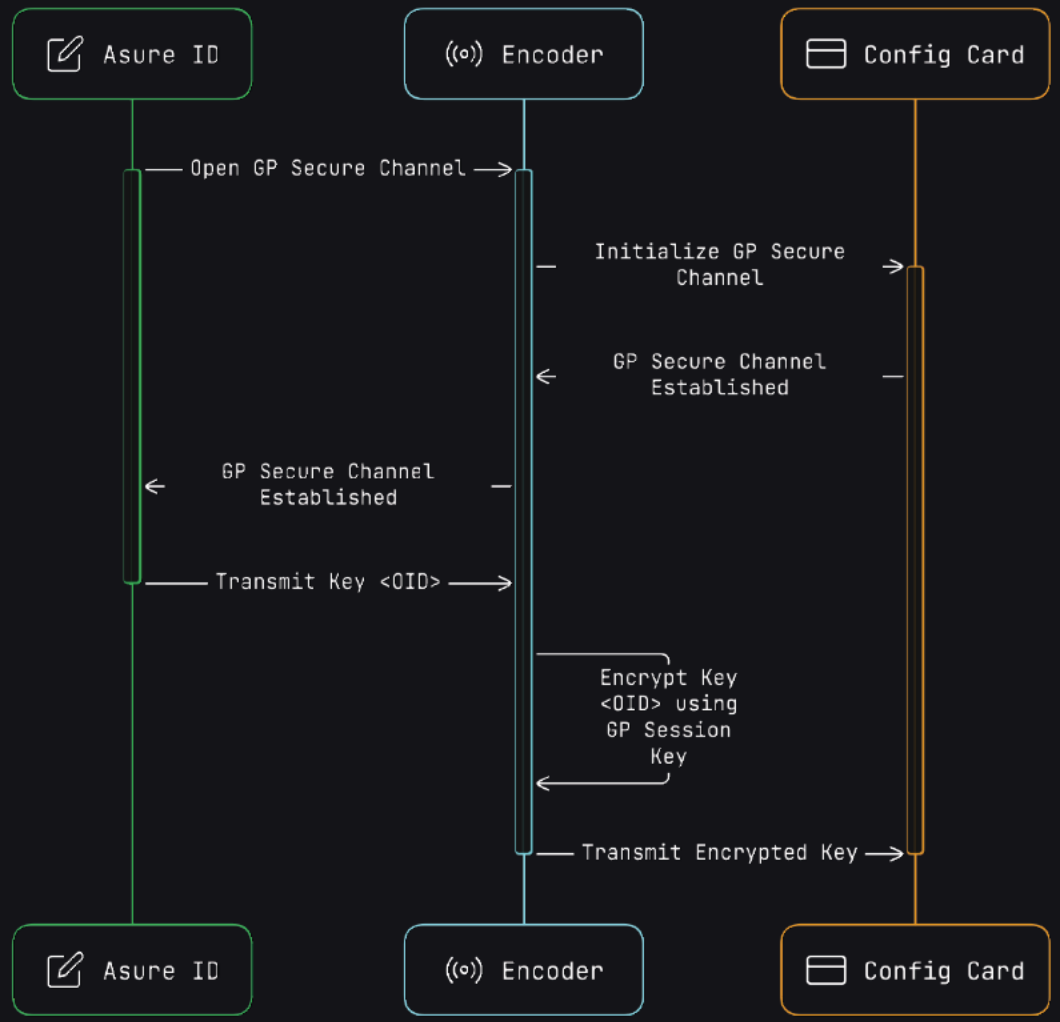
The Heist

Encoder Attack

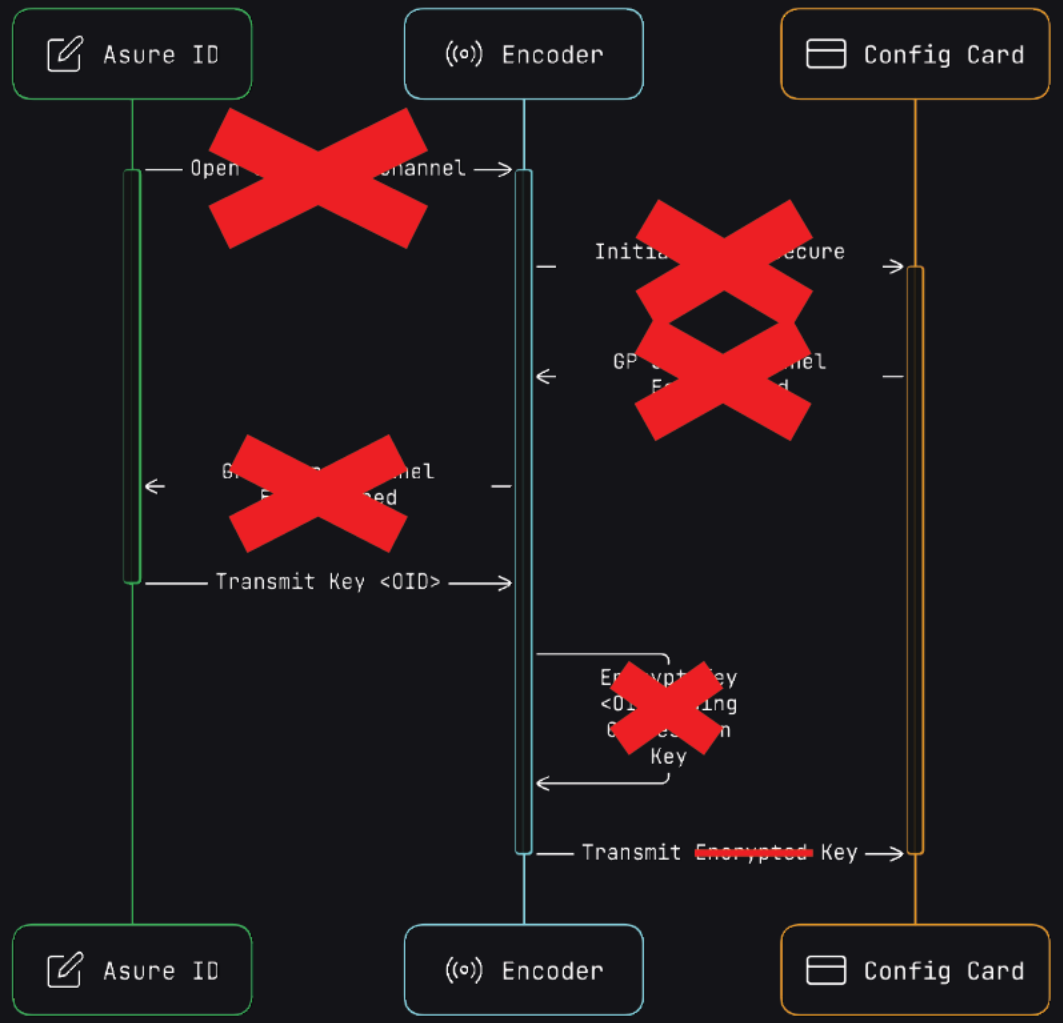
The Heist: Encoder Attack

- The encoder contains Artemis .NET assemblies used for encoding operations
- The assembly intended to program config cards accepts several commands:
  - Open Global Platform Secure Channel
  - Determine Value Size
  - Initialize Applet
  - Transfer Credits
  - Transmit Key
- Typically, the command to establish a secure channel is used prior to transmitting key material to ensure that the cleartext value cannot be sniffed, however there is no logic within the encoder assembly Transmit Key command to check and require a secure channel
- This allows you to issue the Transmit Key command and sniff the cleartext key value being transmitted from the encoder to the card

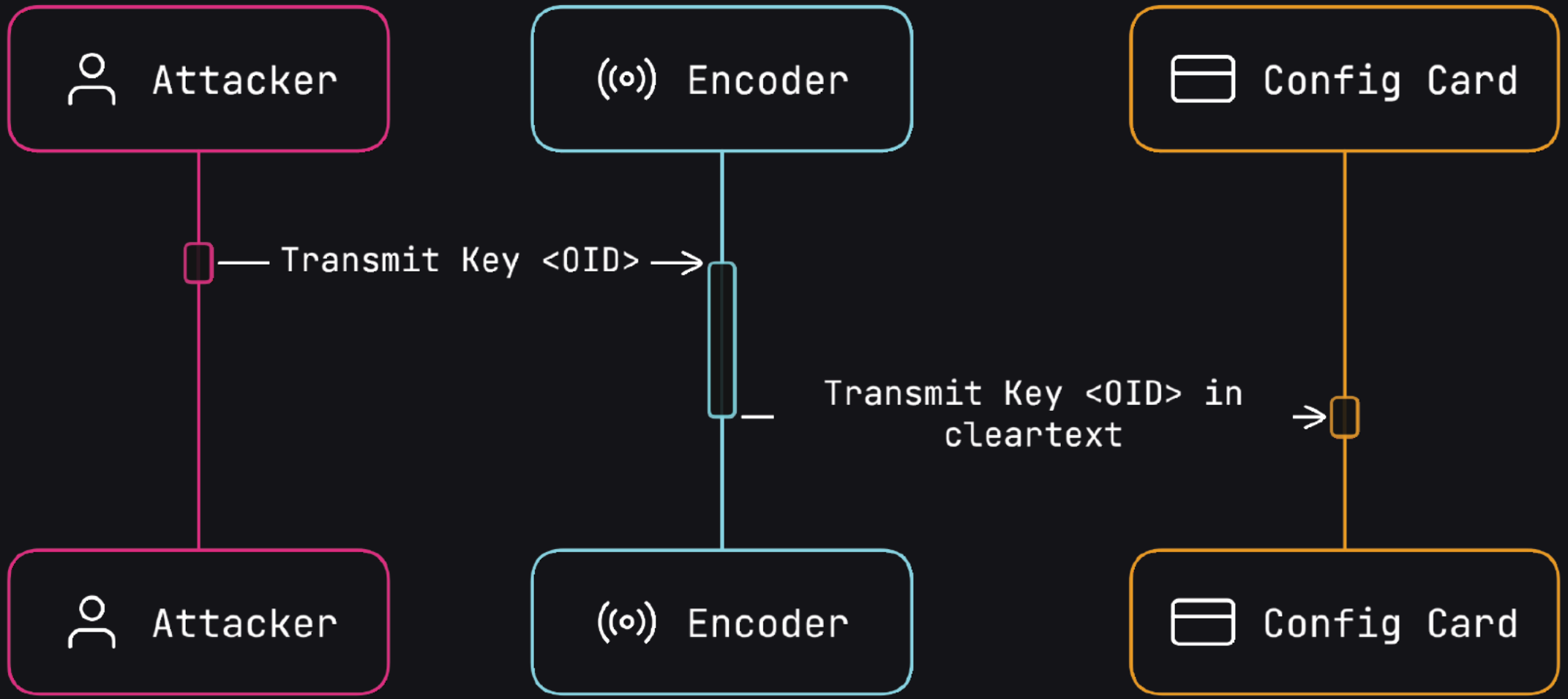
The Heist: Encoder Diagram Highlights



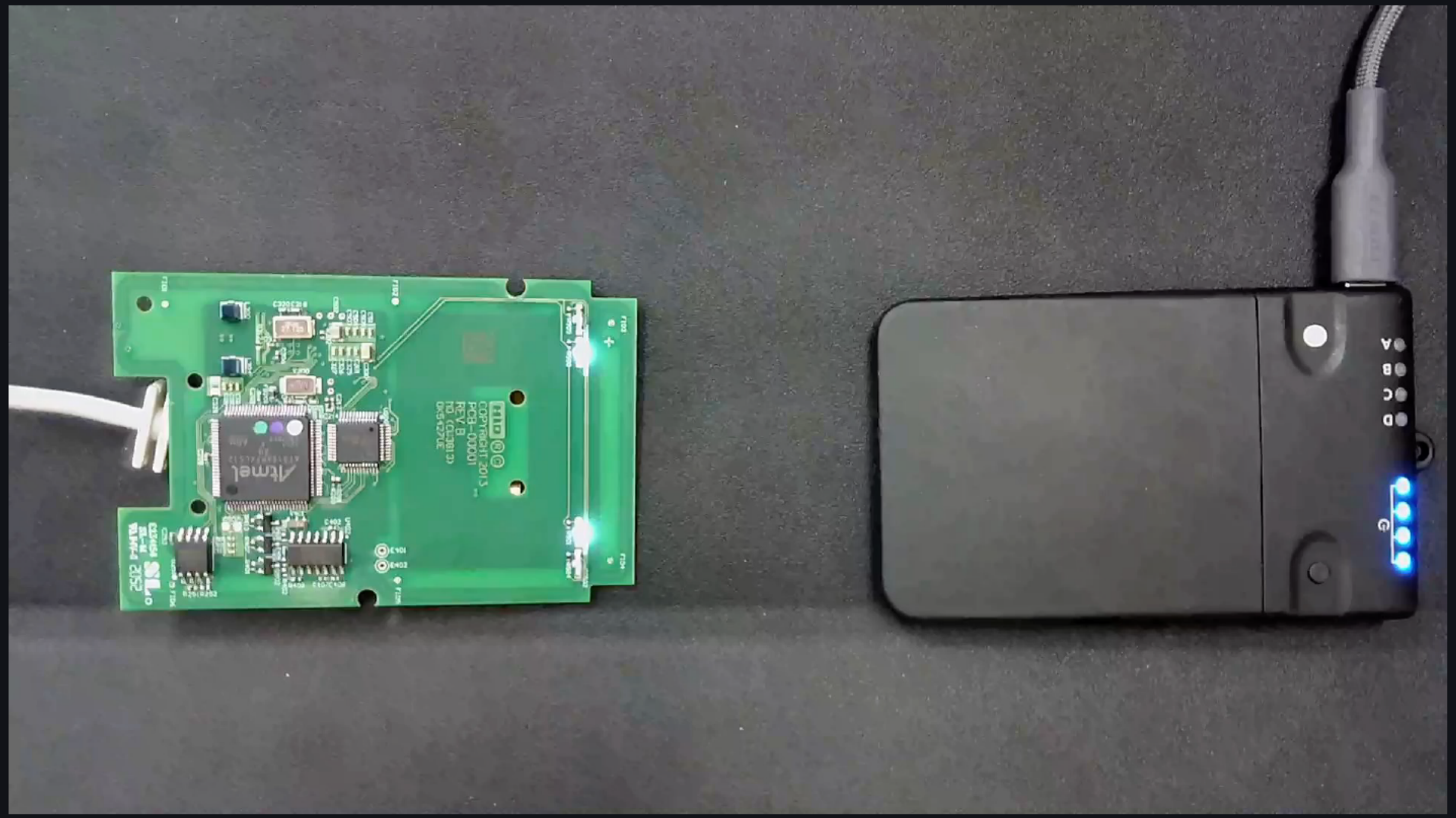
The Heist: Define the Encoder Attack Chain



The Heist: Define the Encoder Attack Chain



The Heist: Execute Encoder Attack



The Heist: Encoder Impact

- Recovery of Shared Key Material
  - Platform Auth / Admin Keys
  - Media Keys
  - Data Object Signing Keys
- Recovery of Other Key Material
  - Elite Keys
  - Custom Keys
  - Third-Party Keys
- Broken Chain of Custody
  - Reduced Confidence in Card Data Integrity
  - Lower Total Cost of Compromise of a Given Target

Aftermath

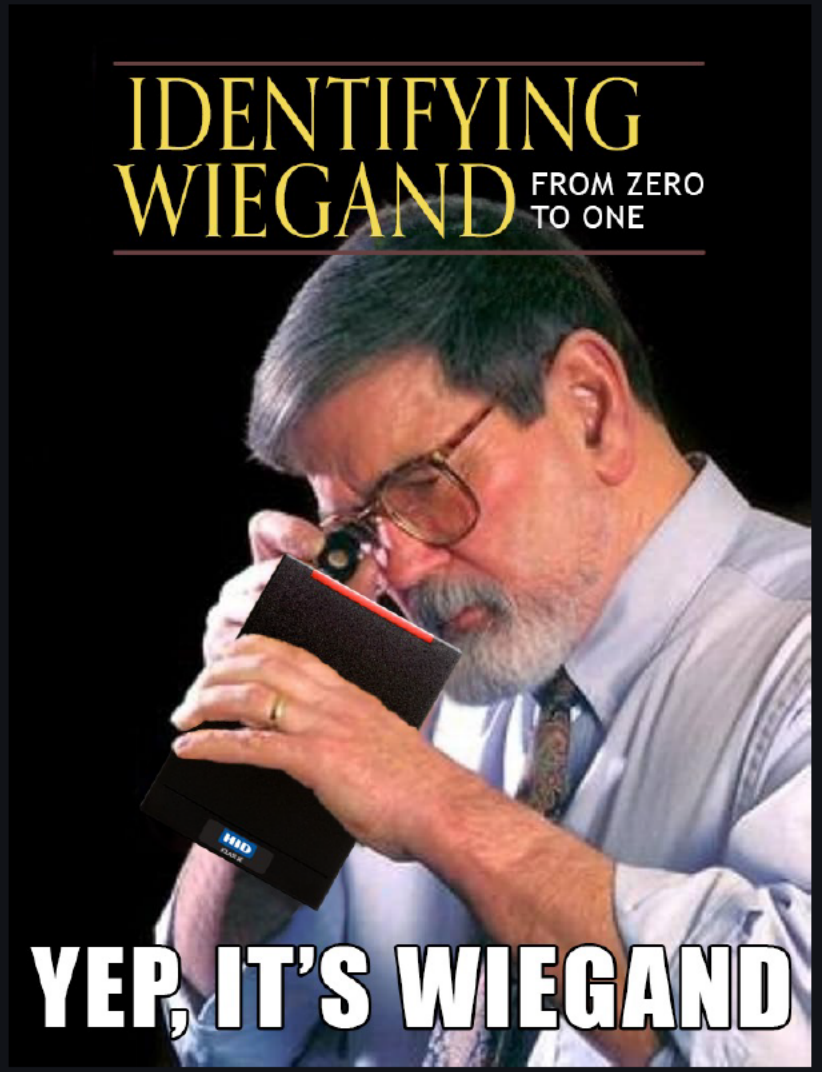
Aftermath: What Went Wrong?

- Config Cards
  - SNMP Key Change vuln is arguably a result of overreliance on the trustworthiness of user input. Any change of SNMP keys should also wipe the data payload of the card
  - Plaintext key transmission vuln is likely a result of debugging code left in place by mistake
  - Buffer reuse vuln is arguably a result of trying to support too many different use cases within a single applet, leading to a much broader attack surface
- Encoder
  - Secure Channel Downgrade vuln is arguably a result of overreliance on the trustworthiness of the client / user input

Aftermath: Internal / External Pressures

- Vendors Often Struggle with Balance
  - Security
  - Convenience
  - Profit
- Internal Pressures (Business Drivers)
  - Sales-Driven Decisions
  - Innovation at Expense of Maturity
  - Pressure to Ship “Functional Enough” Products
- External Pressures (Customer or Market Drivers)
  - Maximum Flexibility and Functionality
  - Minimum Cost
  - Low Maintenance

Aftermath: Industry Hooked on Wiegand



Aftermath: More Flex-Tape for Wiegand

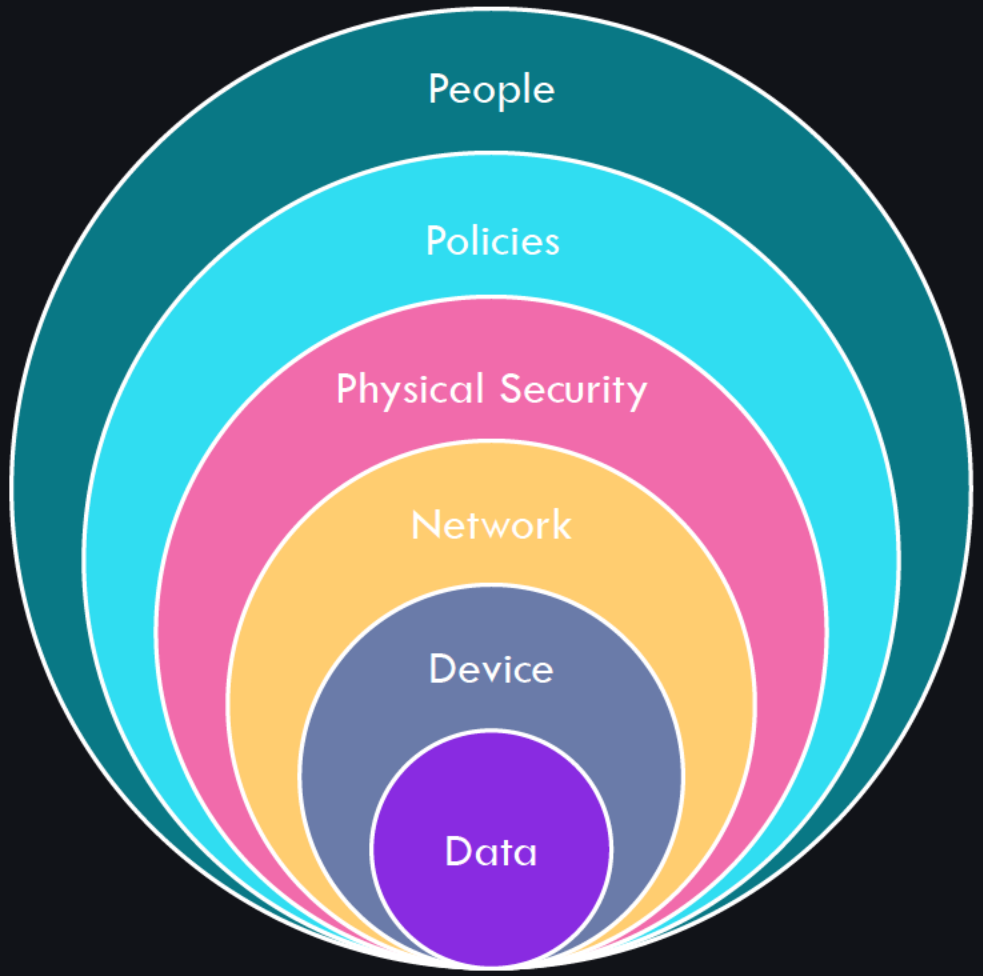


Aftermath: No Wiegand Vendor is Safe!

- We've Inspected Many Reader Vendors
- Most Reader Vendors Don't Even Use SAM's
  - Some Use On-Chip Secure Enclave, but Can Be Risky
  - On-Chip Trust Zones Can Blur Boundaries During Development
- No Reader Vendor Appears to be "Impenetrable"
- iCLASS SE Solution Has Been Most Comprehensive Thus Far

Risk Mitigation: The Sky Isn't Falling…

- …but it is a little bit closer.
- RFID PACS is Just One Element of One Layer



Risk Mitigation: Redefining Risk

- Not Every Vulnerability Represents a Material Threat Today
- Different Customers Protect Different Assets from Different Actors
- Appropriate Threat Modeling is Key

Risk Mitigation: Threat Matrix

- Assessing Your Risk
  - Asset Value: What are you protecting? (e.g., sensitive data, intellectual property, physical assets)
  - Geographical Location: Are you in a high-risk area?
  - Industry-Specific Threats: Are there known threats to your industry?

Criteria	Standard Threat Model	Advanced Threat Model
Value of Protected Assets	Low to Moderate	High
Likelihood of Attack	Low to Moderate	High
Industry-Specific Threats	Few or None	Significant
Adversary's Resources	Limited	Plentiful

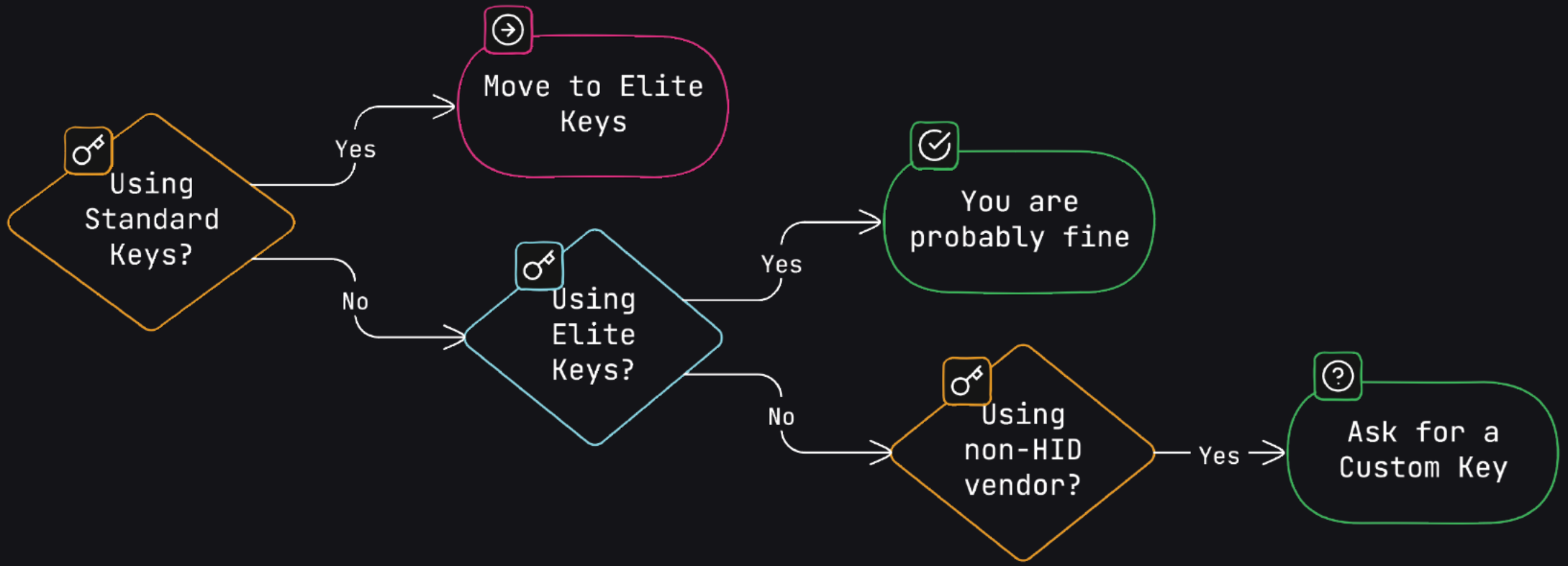
Risk Mitigation: Threat Actors

- Standard Threat Actors
  - Typical Attackers: Opportunistic Criminals, Low-Skill Hackers
  - Motivations: Financial Gain, Mischief, Boredom
  - Examples: Simple Break-ins, Basic Malware Attacks, Opportunistic Probing
- Advanced Threats Actors
  - Typical Attackers: State-sponsored Actors, Organized Crime, Seasoned Hackers
  - Motivations: Espionage, Targeted Attacks, Significant Financial Gain
  - Examples: Sophisticated Phishing Attacks, Zero-Day Exploits, Targeted Espionage

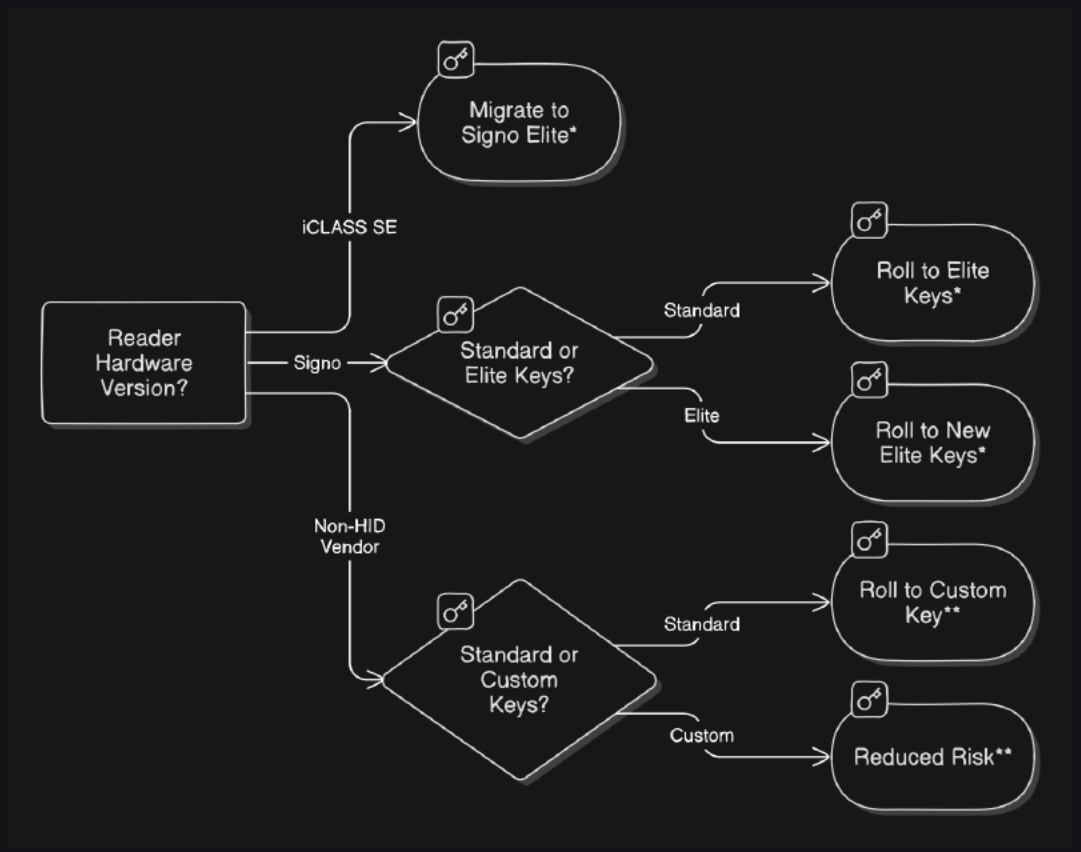




Risk Mitigation: Standard Risk



Risk Mitigation: Advanced, Short Term



* Ensure All Readers Fully Up-to-Date

** Readers without SAMs Offer Reduced Protection

Risk Mitigation: Advanced Risk

- Secure Your Supply Chain
- Secure Key Destruction
  - Physical Destruction May Be Necessary
- Signo Hardware Steps Things Up Considerably
  - Significant Hardware Upgrade with Long-Term Support
  - Incorporation of Many Additional Best-Practices
  - Caveat: All Wiegand-Based Technologies Still Carry Risk
- New Tools and Capabilities Inbound!
  - Key Management
  - Key Rolling

Care to share?

- No.
- Keys Are Not Public (At This Time)
- The Risk is Now Appropriately Proven
- Impacted Parties Need Time to Adapt
- Responsible Research Responsibilities
  - Sharing Keys Today Does Harm
  - Please Don’t Share Keys or Exploits
  - Digital Actions Have Analog Consequences



Doing it Right

PACS + PKI (PIV): Intro and History

- 2000: US Department of Defense (DoD) creates “Common Access Card” office, developing standards for cross-agency trusted credentials. Cards use digital signatures instead of static key, with the private keys never leaving the card.
- 2001: Cards begin to be issued by DoD.
- 2002: DoD issues it’s millionth PKI card
- 2004: US President George Bush issues executive order HSPD-12 in response to the September 11th terrorist attacks, requiring the government to adopt high security credentials.
- 2005: PIV (Personal Identity Verification) standards are released, somewhat interoperable with CAC and intended for government employees and contractors.

PACS + PKI: PIV Sequence Diagram

Inspect Card

- Card is Presented, Reader Selects PIV Application
- Reader Requests Unique Identifier from Card

Challenge Card

- Controller Generates Random Challenge
- Reader Sends Random Challenge to Card

Verify Response

- Card Signs Challenge, Reader Returns Response
- Controller Verifies Signature against Public Key

PACS + PKI: PIV Pro's and Con's

- Pros:
  - Private Key Generated On-Chip (or Imported at Manufacture)
  - Private Key Never Leaves Card
  - **Challenge Response Always Changes**
  - Allows for Secure Interoperability

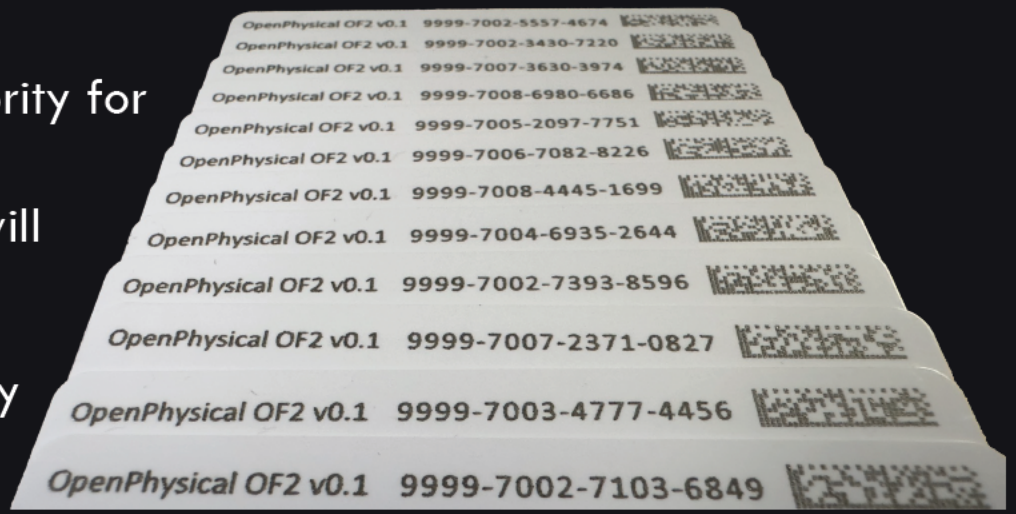
- Cons
  - PKI Historically Slower Due to Heavy Complexity
  - Reduced "Read Range" Due to Increased Power Requirements
  - Key Management is More Complex
  - Certificate Issuance Can Be Cumbersome

PACS + PKI: Open-Source Solutions

- PivApplet (https://github.com/arekinath/PivApplet)
  - PIV-Compatible JavaCard Applet
  - Designed to Interoperate with YubiKey Tools
  - Simplified Key / Certificate Loading, Requires Existing or New Certificate Authority
- OpenFIPS201 (https://github.com/makinako/OpenFIPS201)
  - Production-Ready, Alternative Applet Targeting FIPS 201 (PIV) Certification
  - Increased Administrative Complexity
  - Requires Existing or New Certificate Authority
- OpenPhysical
  - Project Focused on Fully Open-Source PIV Architecture
  - “Plug and Play” Deployment, No Certificate Authority Required
  - Releasing “Soon”™
  - Interested? JOIN US! https://www.openphysical.org

PACS + PKI: OpenPhysical Status

- Card provisioning finalized
  - Cards come pre-configured with *card* certificates, **not user certificates**
  - Cards will come ready ”out-of-the-box”
  - Cards have a unique ID that can be used similarly to RFID Printed Card Numbers, allowing automatic enrollment by PACS systems
- Certificate authority [in-progress]
  - OpenPhysical will operate a free community Certificate Authority for email-verified user certificates and on-line validity checking
  - End users will be able to replace all keys and certificates at will
- Manufacturing Partners [tentative]
  - Any organization can make OpenPhysical cards, provided they meet auditability, security, and compatibility requirements
- Advocacy [ongoing]
  - Work with standards bodies and industry to promote PIV adoption

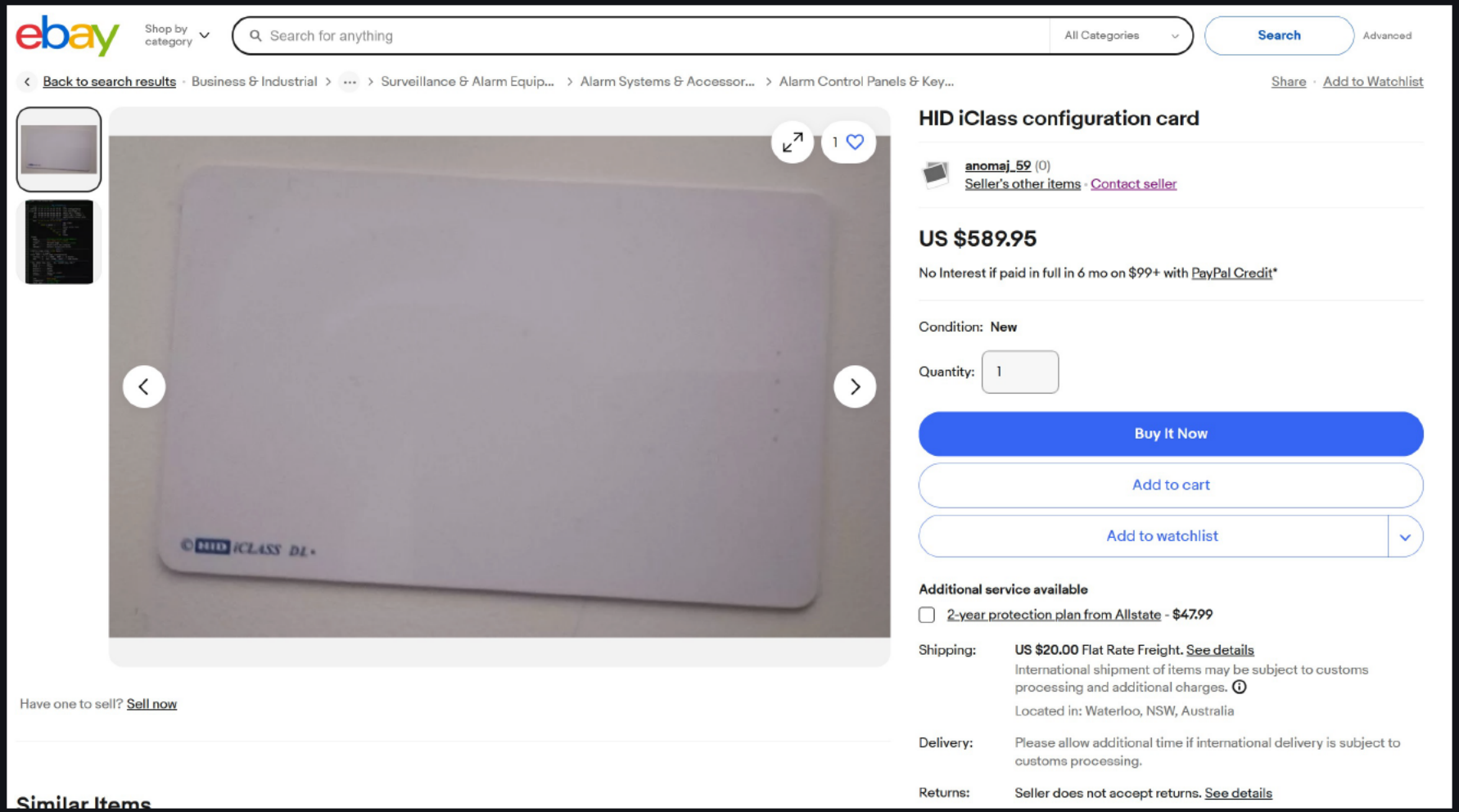


PACS + PKI: Resources to Learn More

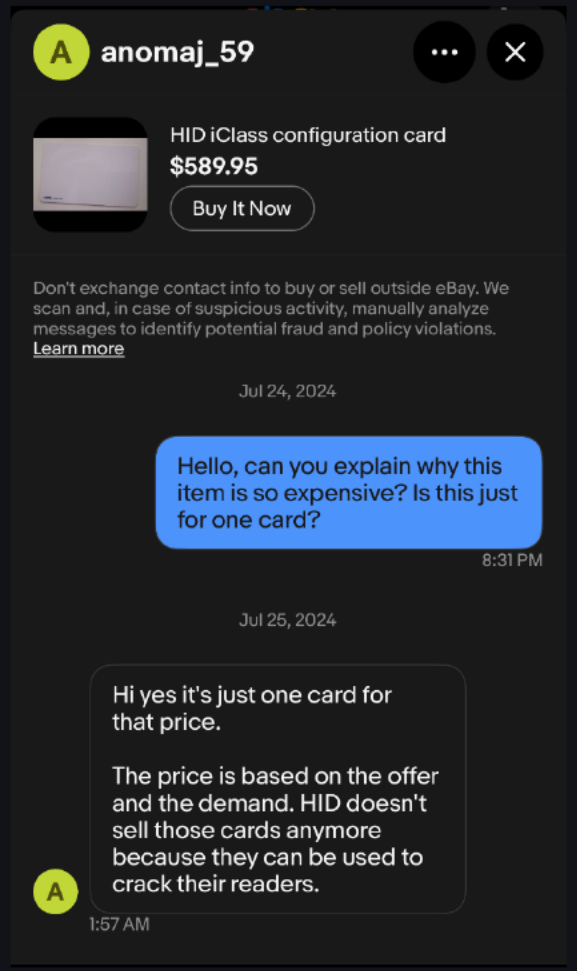
- A comparison of PIV, PIV-I (interoperable), and CIV (commercial PIV equivalent) credentials
  - https://web.archive.org/web/20170830074250/www.securetechalliance.org/resources/pdf/PIV_PIV-I_CIV_brief_022212.pdf
  - The Commercial Identity Verification (CIV) Credential – Leveraging FIPS 201 and the PIV Specifications: Is the CIV Credential Right for You?
    - https://iqdevices.com/pdfFiles/CIV_WP_101611.pdf
  - CAC implemention guide
    - https://www.cac.mil/Portals/53/Documents/ref1.c.i-CAC_End-Point_Implementation_Guide_v1-22.pdf
  - PIV Standard (FIPS 201-3)
    - https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.201-3.pdf
  - SIA OSDP (OSDP PIV)
    - https://www.securityindustry.org/industry-standards/open-supervised-device-protocol/

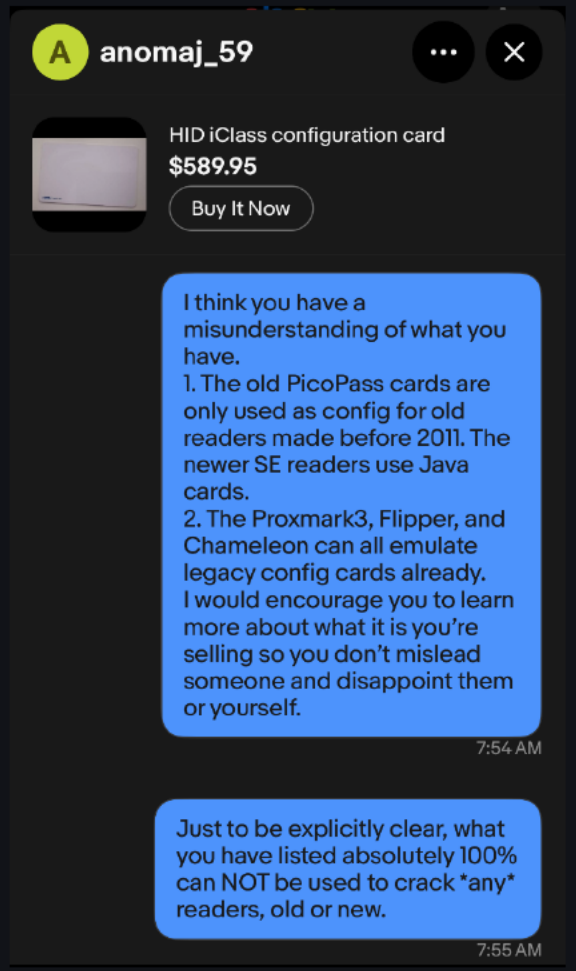
Story Time!

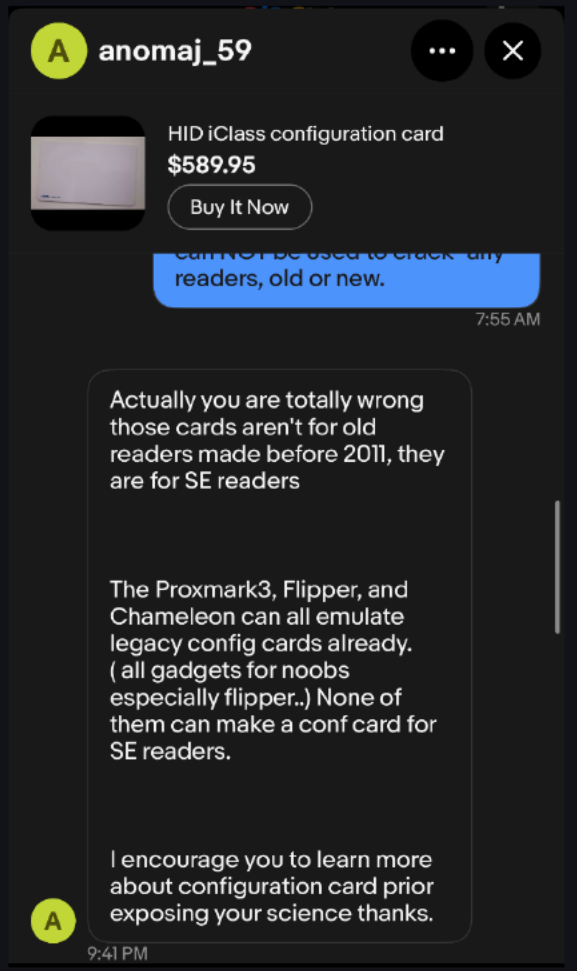
Story Time: Ridiculous eBay Sellers



Story Time: Ridiculous eBay Sellers







Outro: Resource Archive

- https://web.archive.org/web/20200218193359/http://www.proxmark.org/forum/viewtopic.php?id=1994

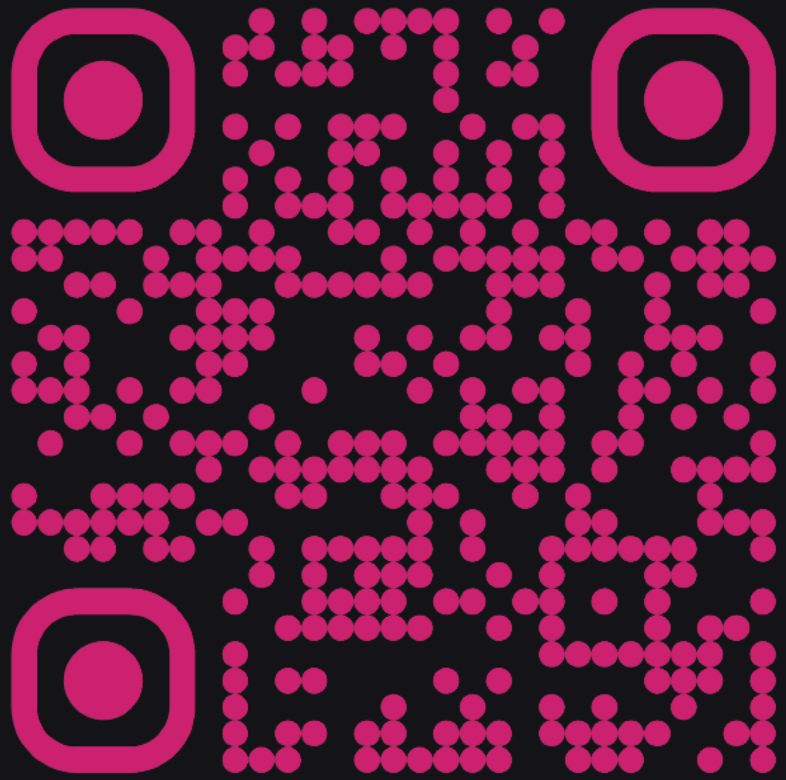
Outro: PSA's and Relevant CVE's

- HID-PSA-2024-001: Sensitive Data Extraction from Reader Configuration Cards
  - CVE-2024-23806
- HID-PSA-2024-002: Secure Channel Downgrade in Encoders / Readers
  - CVE-2024-22388

Q&A

Outro: On-Demand Q&A

Need More Detail or Guidance? Join our Discord and Ask!



https://discord.gg/KWcD4CWrfC

Outro: Vendor Appreciation

- Huge THANK YOU to the HID Security Response Team
- HID Has Been a Great Sport (Considering)
- https://www.hidglobal.com/security-center



Outro: Vendor Appreciation

- Thank You to Full Spectrum Laser for the Donated Laser Time!
- Phenomenal UV Lasers for IC Decapsulation and More
- These Guys Make Amazing Gear
- Support Them Because They're Awesome
- Use Code "RTA" for Discount



Full Spectrum
LASER

Outro: Special Thank You

- Kate Gray a.k.a. Mistial Developer
- Core Contributor to Research Team
- Unable to Formally Participate Due to Logistical Constraints



Outro: Individual Appreciation

- Our Partners Bruce, Sarah, and Sam, for their Seemingly Limitless Patience
  - (They have advised it is not, in fact, limitless…)
- 0xFFFF
- Iceman
- Carl55
- Milosch Meriac
- carrot_c4k3

Outro: Community Appreciation

- Proxmark.org Forums
- Discord:
  - RFID Hacking by Iceman
  - Red Team Alliance
- RF Village

- NVX
- Eric Betts
- Craftbyte
- Micsen
- topy
- thesle3p

Outro: Audience Appreciation

