

Bypassing Smart-Card Authentication and Blocking Debiting:

*Vulnerabilities in Atmel Cryptomemory based
Stored-Value Systems*

Jonathan Lee

Neil Pahl

Primer

Who are we?

Jonathan Lee – UBC Computer Engineering student

Neil Pahl – UBC Electrical Engineering, recent graduate

Disclaimer !

- *We are not (yet) security industry professionals or 1337 code-breaking hackers*
- *We did not break Smart Card Security*
- *But, we can show that even with introductory security knowledge, we were able to find vulnerability in a 'secure' system's implementation*

Primer

What do we know about Security?

EECE 412: Intro to Computer Security

- Fundamental Security Principles
- Cryptography
- Authentication
- Access Control
- Secure Design

Primer

What Was Our Motivation For this Project?

Term Project to Perform a Security Analysis

➤ *(Opportunity to Legitimately try Cool Hacks)*

- Smart Cards are synonymous with Security
- We hoped to emulate known replay attacks on a nearby Smart Card laundry system

Primer

Why should anyone care about our Hack?

- Companies spend time and money to add more security functions to their products
- However, security functions cannot protect the system if they are not implemented thoughtfully

History?

The need for Authentication + Encryption

- Joe Grand (2009)– SF parking meters, vulnerable to replay attack
- Strom Carlson (2006) – Fedex Kinko's SLE4442, vulnerable to simple password replay

Early Efforts

Smart card laundry machine (stored value card)



Early Efforts

We were thinking of trying some of the usual attacks:

- Could we simply read and modify monetary values directly on the card?
- Could we sniff out read/write passwords?
- Could we perform replay attacks?

But, the success of these attacks depend on the security measures in place

Early Efforts

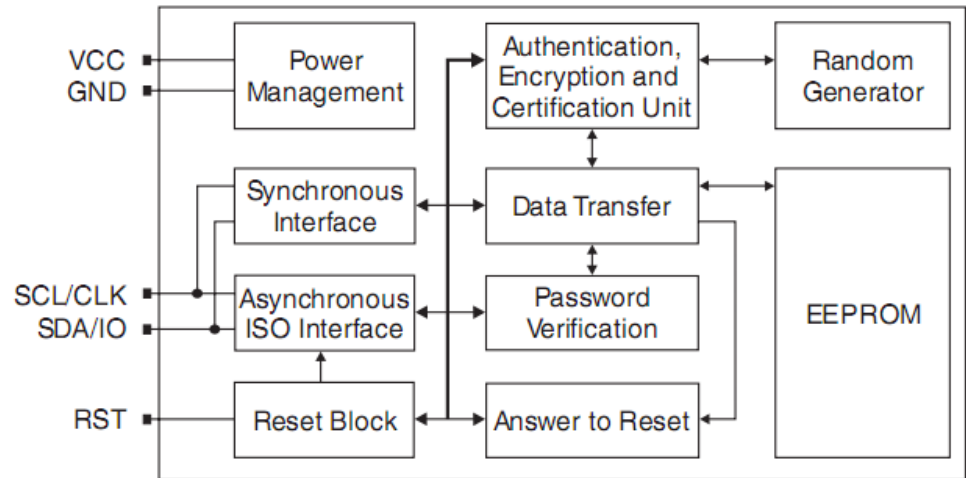
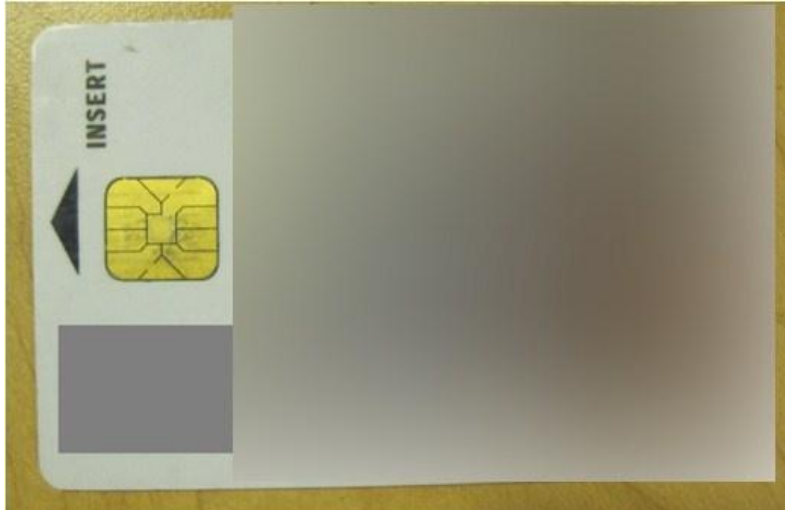
A few Google searches lead us to an interesting blog of someone attempting to hack a laundry system (<http://tinyurl.com/2csk6dr>)

- Their success was limited to reading the configuration settings from the card.
- We couldn't confirm it yet, but we decided to start with the assumption that our system used the same chip.

Early Efforts

Atmel CryptoMemory AT88SC0404 Smart-card

ATR: 3B B2 11 00 10 80 00 04



Early Efforts

High Security Features:

- **64-bit Mutual Authentication Protocol (under license of ELVA)**
- **Encrypted Checksum**
- **Stream Encryption**
- **Four Key Sets for Authentication and Encryption**
- **Eight Sets of Two 24-bit Passwords**
- **Anti-tearing Function**
- **Voltage and Frequency Monitor**

Early Efforts

Security Modes:

Mode	Configuration Data	User Data	Passwords	Data Integrity Check
Standard/Password	clear	clear	clear	n/a
Authentication	clear	clear	encrypted	MAC
Encryption	clear	encrypted	encrypted	MAC

- Standard/password mode is Default
- Command sequences are needed to enter other security modes

Atmel CryptoMemory AT88SC0404



Atmel CryptoMemory AT88SC0404



\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- Addresses

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404




Interpreting the Memory Dump:

- Addresses 
- Status - 'Access Granted' 

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- Addresses 
- Status - 'Access Granted' 
- Status - 'Access Denied' 

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- DCR – Data Config Reg 

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- DCR – Data Config Reg 
1011 1111

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- DCR – Data Config Reg 

1011 1111

└ 4 Authen. Attempts Allowed

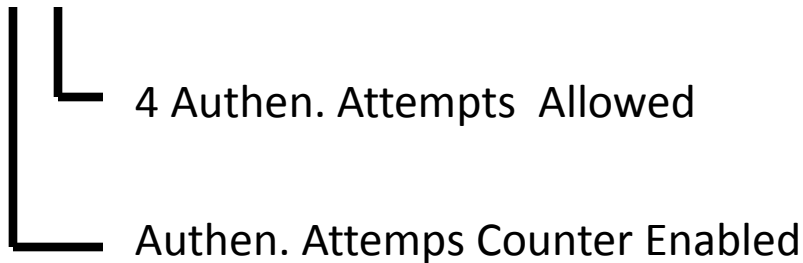
\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- DCR – Data Config Reg 

1011 1111



\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- ARn – Access Registers 

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- ARn – Access Registers 

1101 1111

\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- ARn – Access Registers 

1101 1111

└─ Encryption not necessary

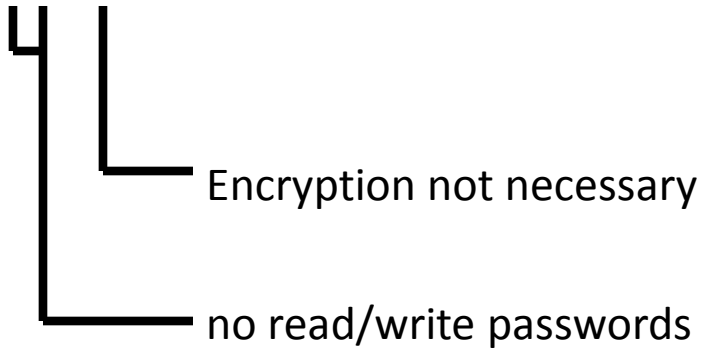
\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- ARn – Access Registers 

1101 1111



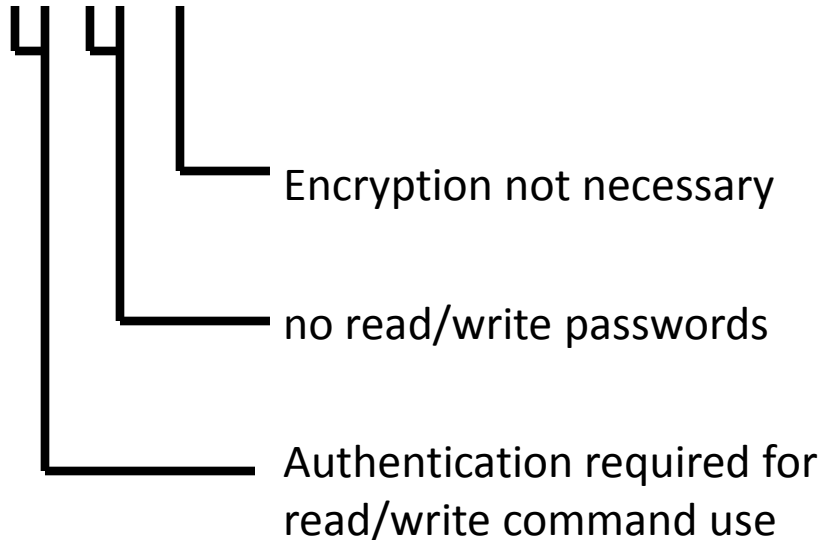
\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

Interpreting the Memory Dump:

- ARn – Access Registers 

1101 1111



\$0	3b	b2	11	00	10	80	00	04	90	00
\$8									90	00
\$10									90	00
\$18	bf								90	00
\$20	df	08	df	08	df	58	df	58	90	00
\$28	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$30	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$38	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$40	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$48	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$50	ff	e7	02	63	79	85	54	8d	90	00
\$58	69	00								
\$60	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$68	69	00								
\$70	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$78	69	00								
\$80	ff	ff	ff	ff	ff	ff	ff	ff	90	00
\$88	69	00								
\$90	69	00								
\$98	69	00								
\$a0	69	00								
\$a8	69	00								
\$b0	ff	20	20	20	ff	20	20	20	69	00
\$b8	ff	20	20	20	ff	20	20	20	69	00
\$c0	ff	20	20	20	ff	20	20	20	69	00
\$c8	ff	20	20	20	ff	20	20	20	69	00
\$d0	ff	20	20	20	ff	20	20	20	69	00
\$d8	ff	20	20	20	ff	20	20	20	69	00
\$e0	ff	20	20	20	ff	20	20	20	69	00
\$e8	ff	20	20	20	ff	20	20	20	69	00

Atmel CryptoMemory AT88SC0404

We Now Know:

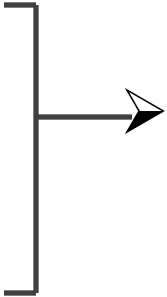
What it Tells Us:

Atmel CryptoMemory AT88SC0404

We Now Know:

- Authentication Attempts Counter Enabled
- 4 Authentication Attempts Allowed

What it Tells Us:



Brute force key cracking is impossible

Atmel CryptoMemory AT88SC0404

We Now Know:

- Authentication Attempts Counter Enabled
- 4 Authentication Attempts Allowed
- Encryption not necessary

What it Tells Us:

→ Brute force key cracking is impossible

→ Might be able to sniff read/write passwords

Atmel CryptoMemory AT88SC0404

We Now Know:

- Authentication Attempts Counter Enabled
- 4 Authentication Attempts Allowed
- Encryption not necessary
- No read/write passwords

What it Tells Us:

→ Brute force key cracking is impossible

→ Might be able to sniff read/write passwords

→ No Passwords to sniff

Atmel CryptoMemory AT88SC0404

We Now Know:

- Authentication Attempts Counter Enabled
- 4 Authentication Attempts Allowed
- Encryption not necessary
- No read/write passwords
- Authentication required for read/write command

What it Tells Us:

→ Brute force key cracking is impossible

→ Might be able to sniff read/write passwords

→ No Passwords to sniff

→ System will be run under Authentication Mode

Atmel CryptoMemory AT88SC0404

Recall the Security Modes:

Mode	Configuration Data	User Data	Passwords	Data Integrity Check
Standard/Password	clear	clear	clear	n/a
Authentication	clear	clear	encrypted	MAC
Encryption	clear	encrypted	encrypted	MAC

- MAC Value used for data integrity check is a checksum encrypted by a session key

Early Findings

A simple replay attack isn't possible...

And we have no expertise or time for cryptanalysis of authentication procedure either
(6 weeks!!)

What's left?

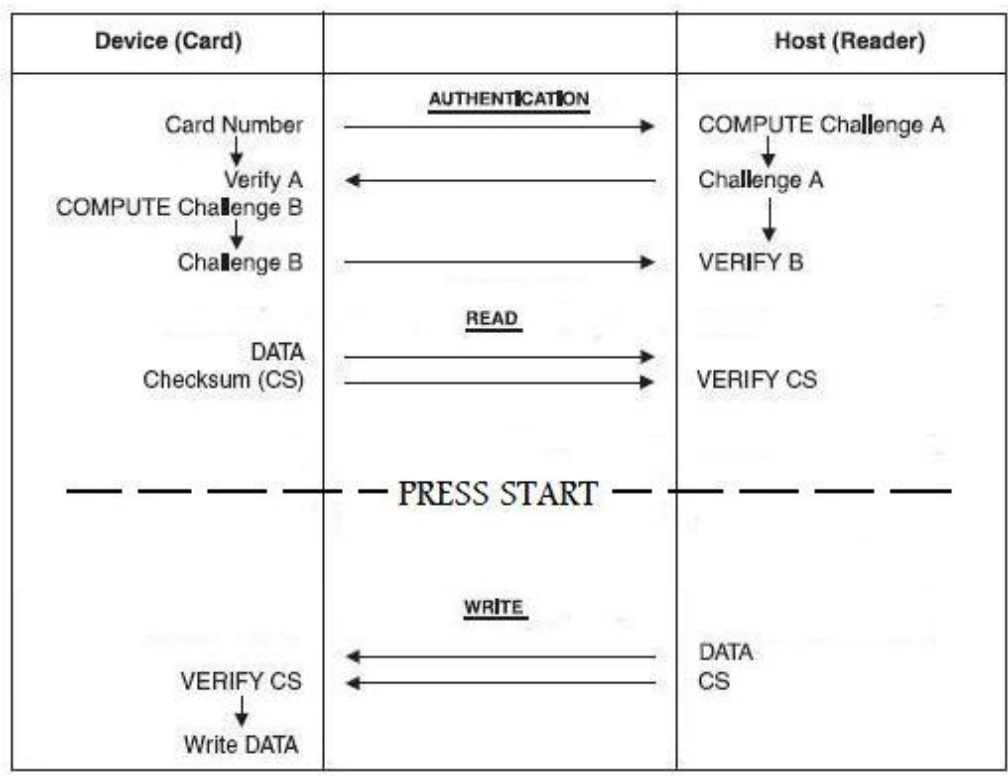
A Hypothesis...

What would be the most logical implementation of this type of system?



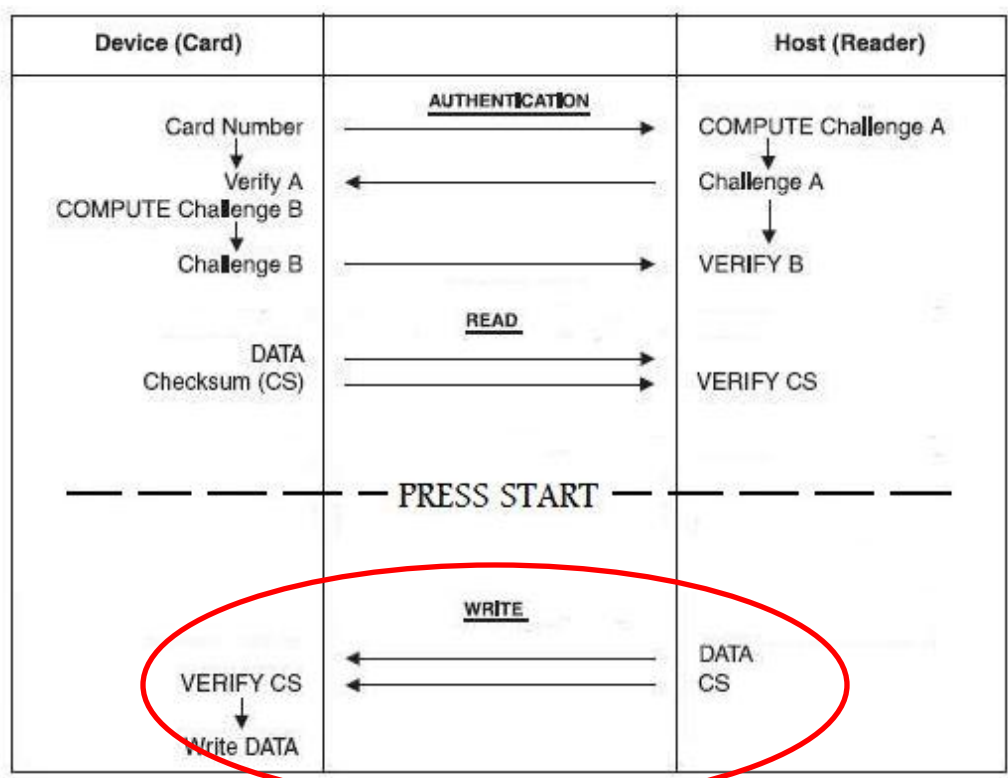
A Hypothesis...

What would be the most logical implementation of this type of system?



A Hypothesis...

What would be the most logical implementation of this type of system?



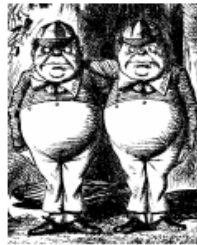
Would there be a MiM opportunity here?

Alice, Bob and Trudy...

Man in the middle concept



Alice, a



Trudy, t



Bob, b



Can we apply this in hardware?

MACHINE-TO-CARD COMMUNICATION

Machine-to-Card Communication

Sniffing Tools:

- Modified Smart Card



Machine-to-Card Communication

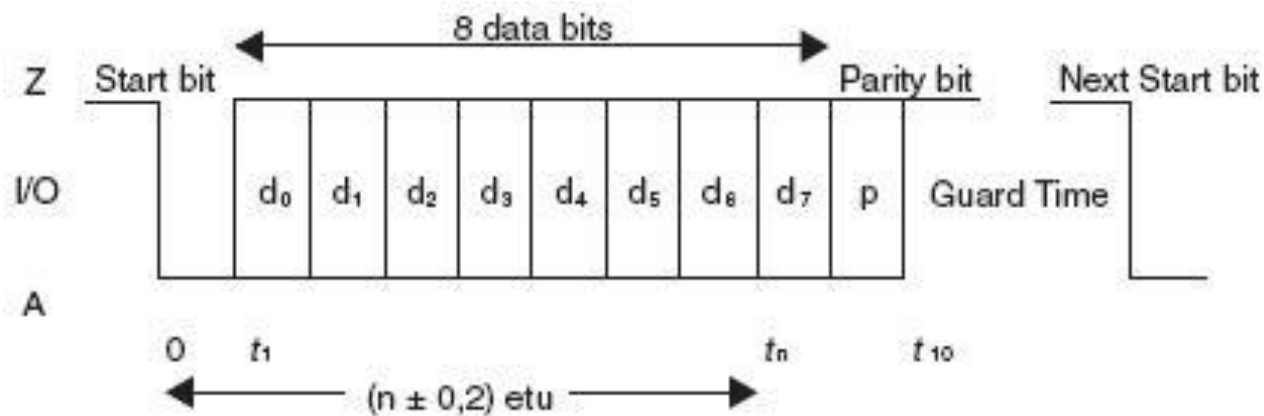
Sniffing Tools:

- Modified Smart Card
- Logic Analyzer



Machine-to-Card Communication

- 9600 Baud UART
- Byte-By-Byte data transmission in the form:



- Groups of Bytes make up Commands

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

- Card Sends Answer To Reset (ATR)

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

- Card Sends Answer To Reset (ATR)
- Machine Reads 8-Byte Cryptogram and 7-Byte ID
 - Machine uses Cryptogram to compute challenge value

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

- Card Sends Answer To Reset (ATR)
- Machine Reads 8-Byte Cryptogram and 7-Byte ID
 - Machine uses Cryptogram to compute challenge value
- Machine Sends Authentication Request Command
 - Machine sends 16-Byte challenge value and nonce
 - Card uses challenge to compute new cryptogram

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

- Card Sends Answer To Reset (ATR)
- Machine Reads 8-Byte Cryptogram and 7-Byte ID
 - Machine uses Cryptogram to compute challenge value
- Machine Sends Authentication Request Command
 - Machine sends 16-Byte challenge value and nonce
 - Card uses challenge to compute new cryptogram
- Machine Reads *new* 8-Byte Cryptogram
 - New cryptogram is verified

Machine-to-Card Communication

Communication Flow (Before Pressing Start):

- Card Sends Answer To Reset (ATR)
- Machine Reads 8-Byte Cryptogram and 7-Byte ID
 - Machine uses Cryptogram to compute challenge value
- Machine Sends Authentication Request Command
 - Machine sends 16-Byte challenge value and nonce
 - Card uses challenge to compute new cryptogram
- Machine Reads *new* 8-Byte Cryptogram
 - New cryptogram is verified
- Machine Reads from User Zones

Machine-to-Card Communication

Communication Flow (After Pressing Start):

Machine-to-Card Communication

Communication Flow (After Pressing Start):

- Machine Reads from User Zones

Machine-to-Card Communication

Communication Flow (After Pressing Start):

- Machine Reads from User Zones
- Machine Writes Newly Decremental Balance to User Zones (Using 3 Write Commands)

Machine-to-Card Communication

Communication Flow (After Pressing Start):

- Machine Reads from User Zones
- Machine Writes Newly Decremental Balance to User Zones (Using 3 Write Commands)

** Read and Write Commands Require Trailing CheckSum Cmds

Machine-to-Card Communication

Looking closely at the final 3 write commands:



Machine-to-Card Communication

Looking closely at the final 3 write commands:

5-Character Command



Machine-to-Card Communication

Looking closely at the final 3 write commands:

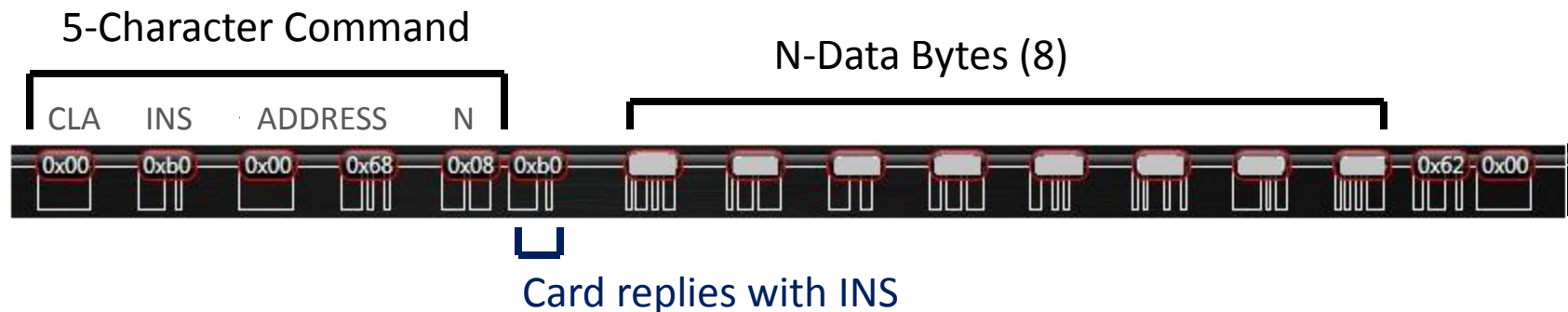
5-Character Command



Card replies with INS

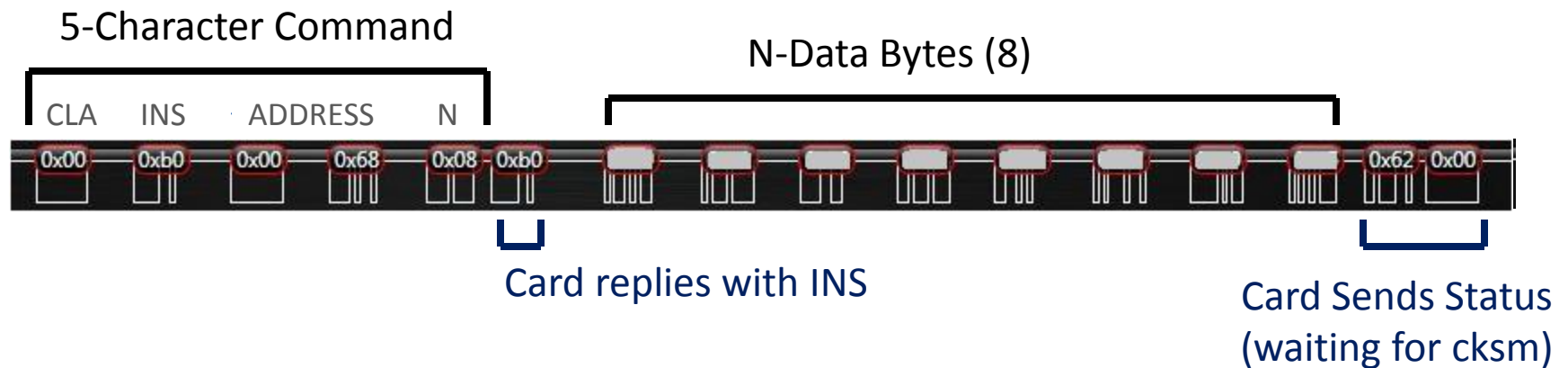
Machine-to-Card Communication

Looking closely at the final 3 write commands:



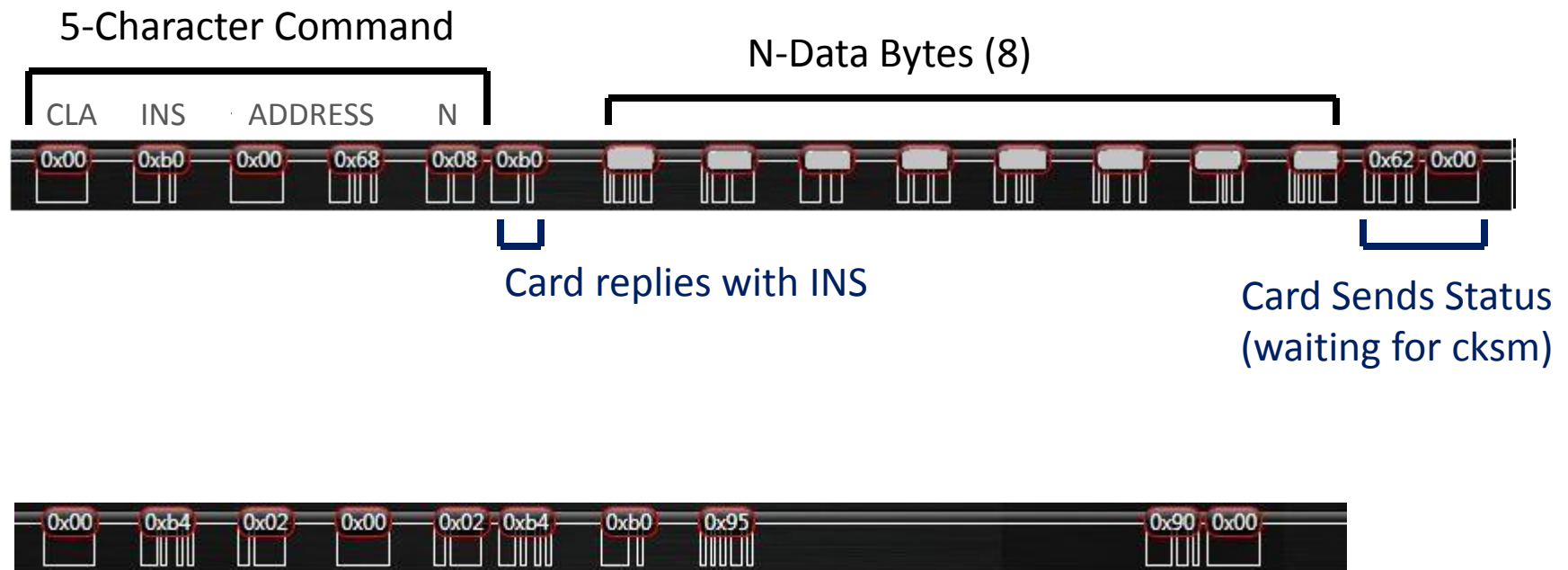
Machine-to-Card Communication

Looking closely at the final 3 write commands:



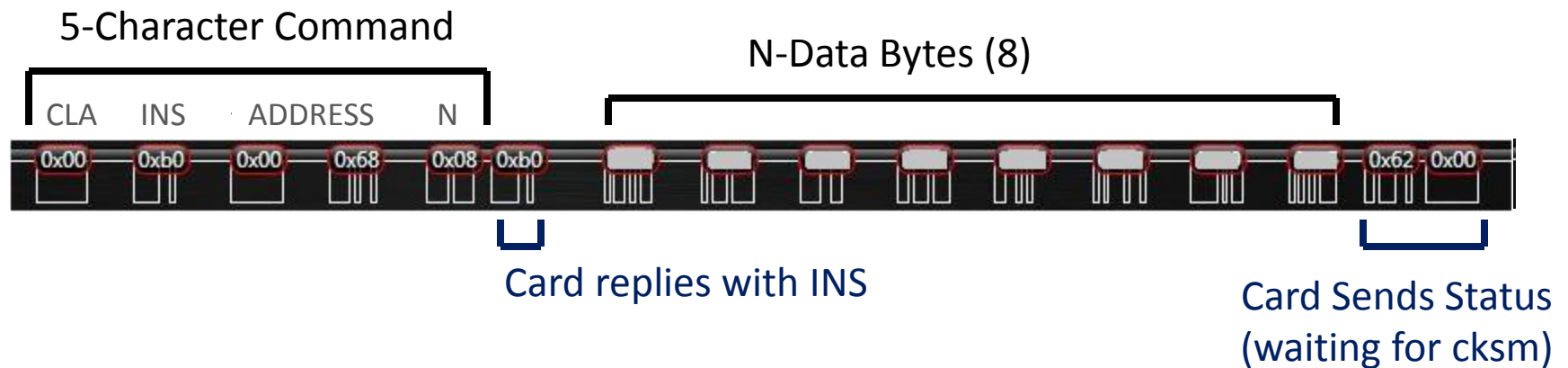
Machine-to-Card Communication

Looking closely at the final 3 write commands:



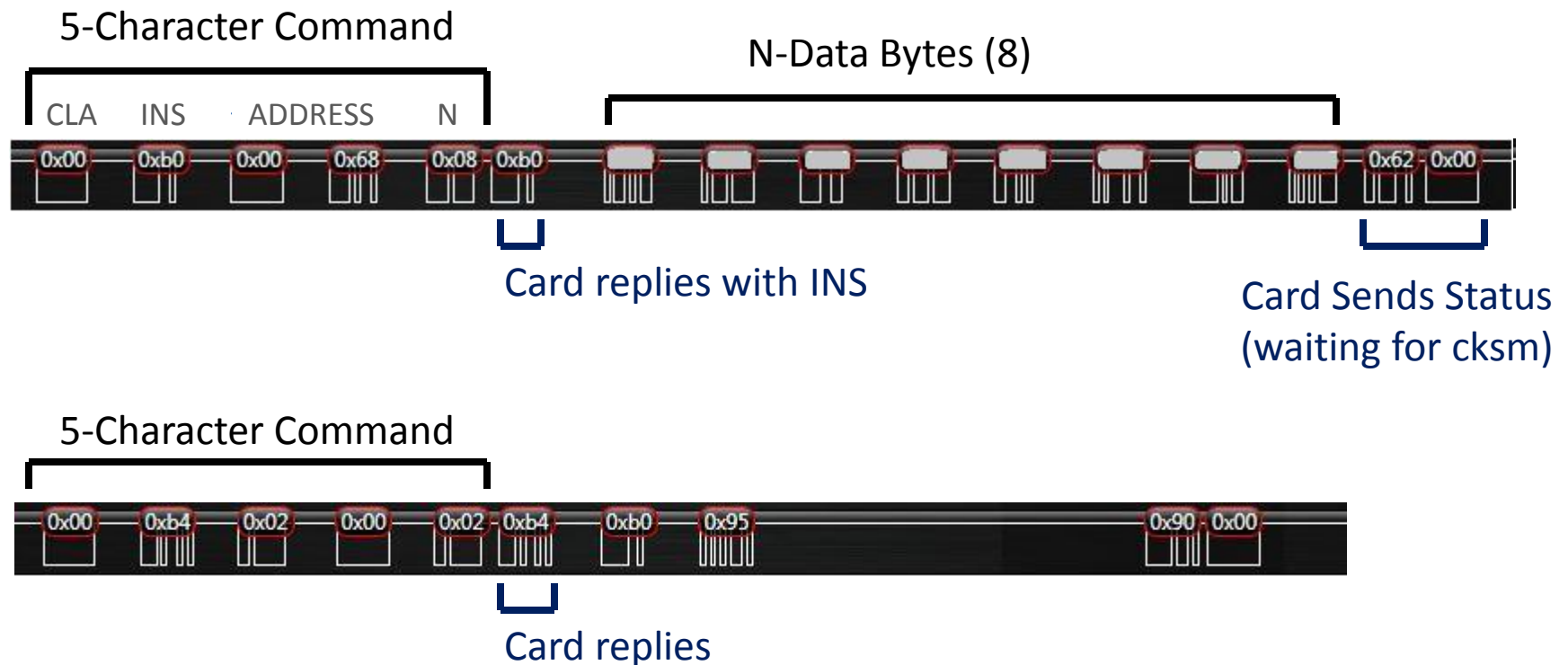
Machine-to-Card Communication

Looking closely at the final 3 write commands:



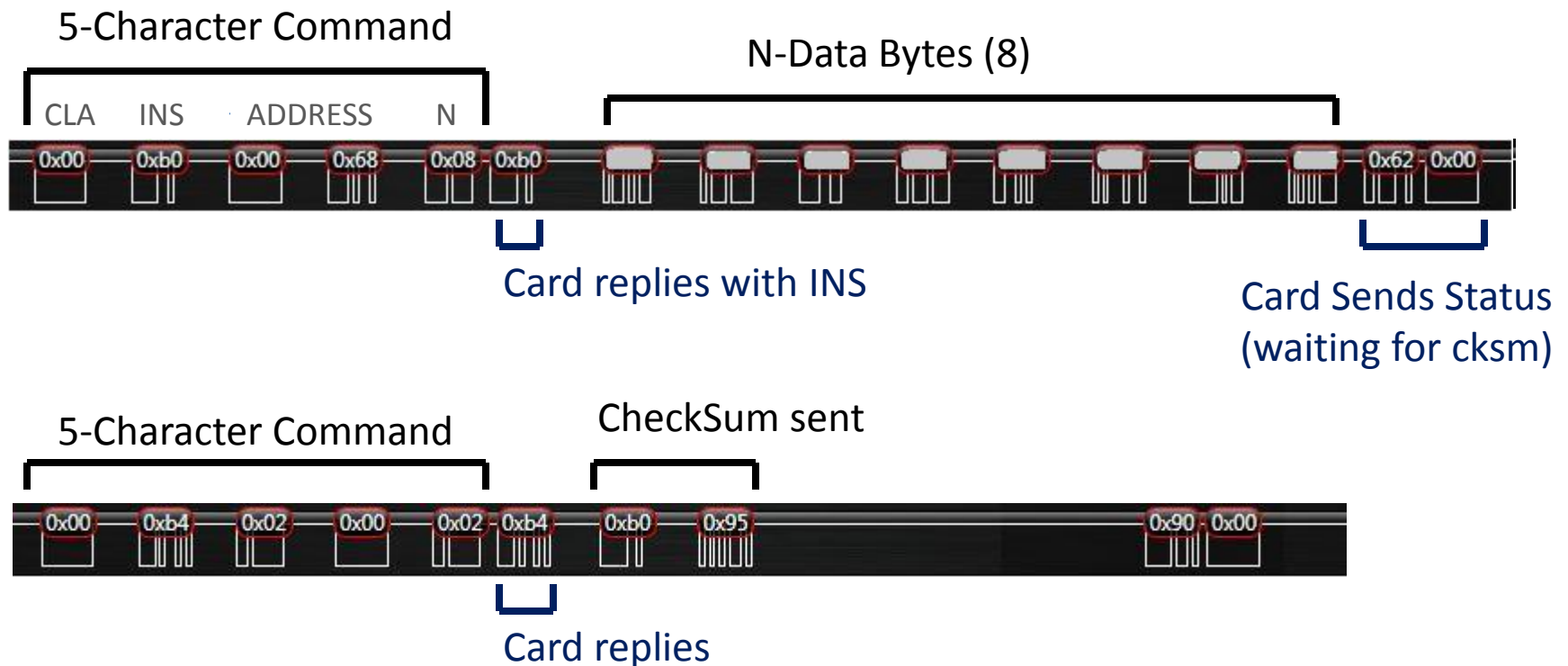
Machine-to-Card Communication

Looking closely at the final 3 write commands:



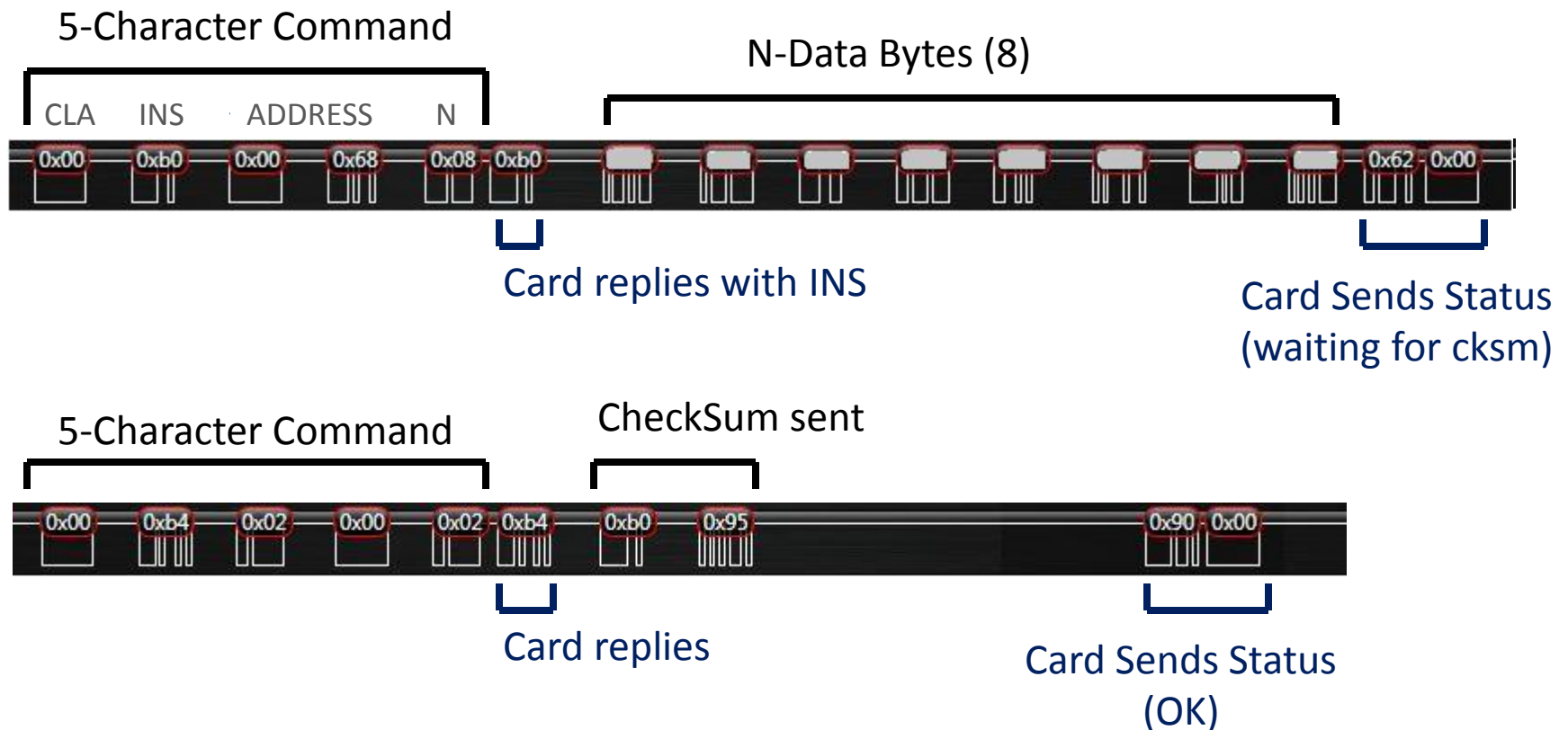
Machine-to-Card Communication

Looking closely at the final 3 write commands:



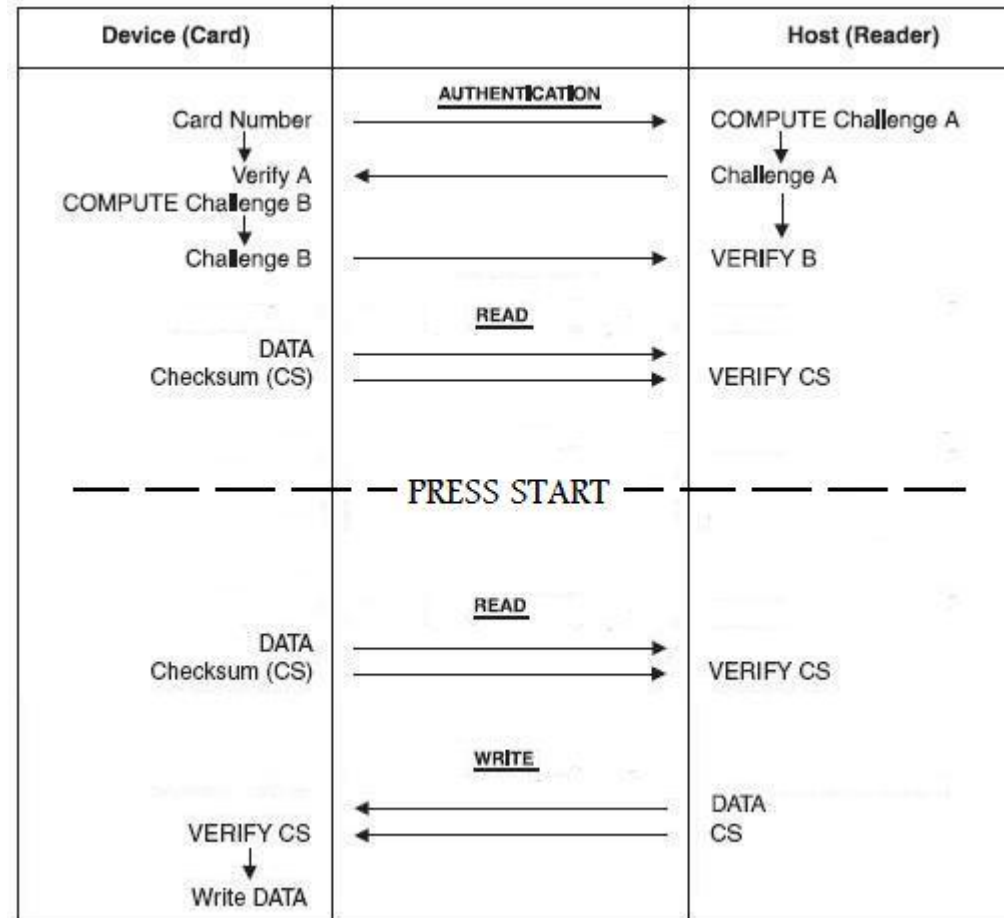
Machine-to-Card Communication

Looking closely at the final 3 write commands:



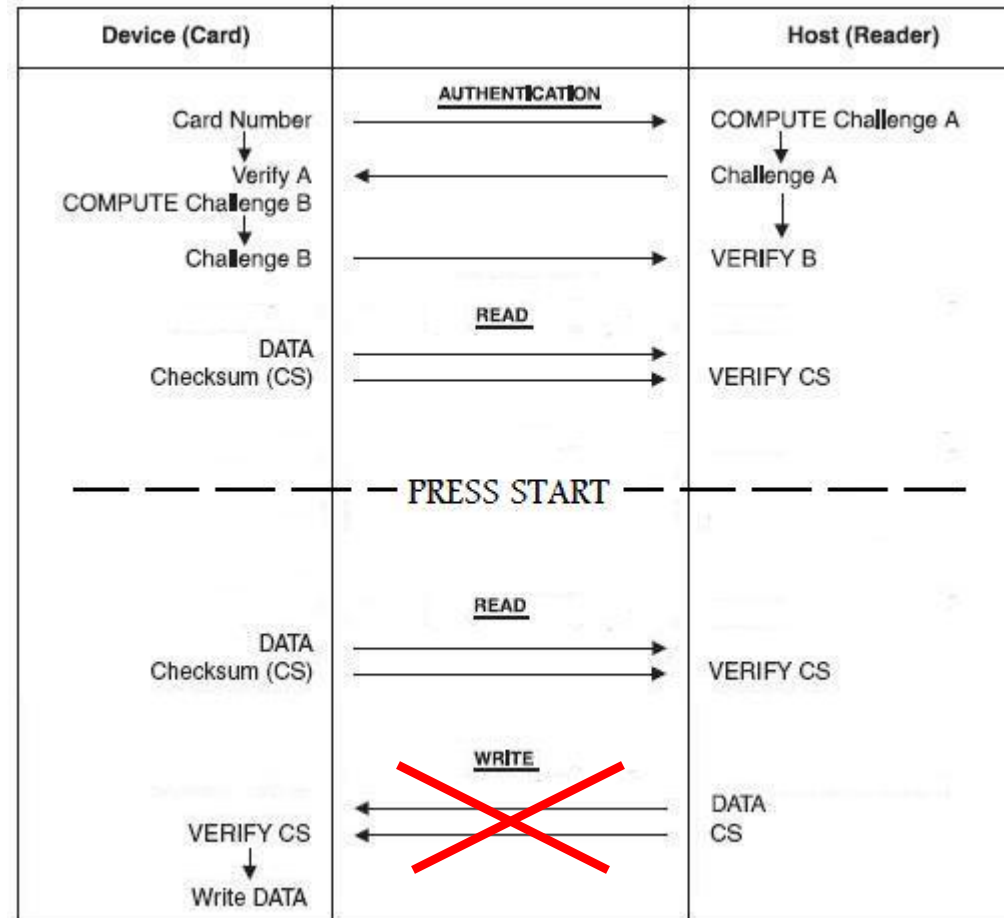
The Vulnerability

To Recap:



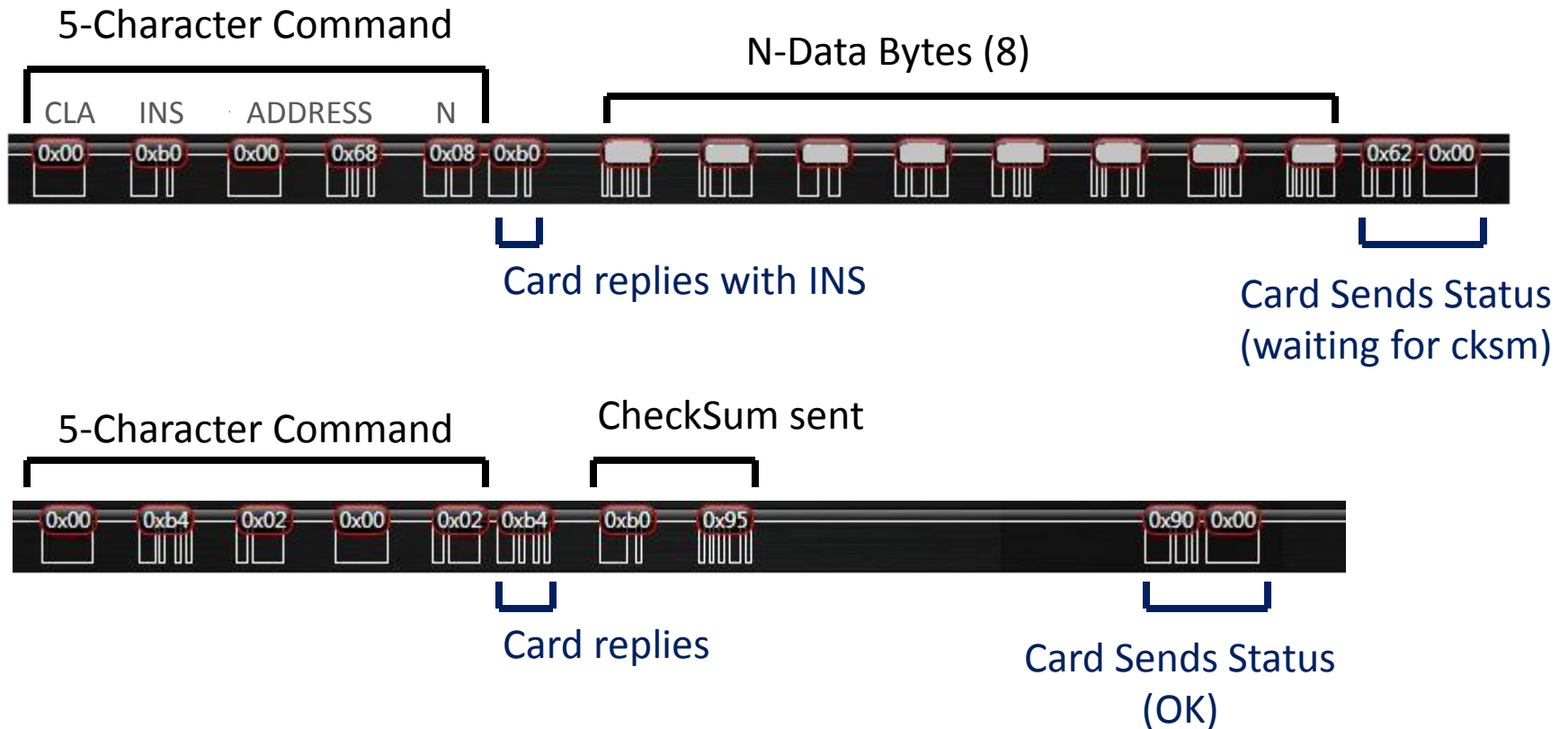
The Vulnerability

The Plan:



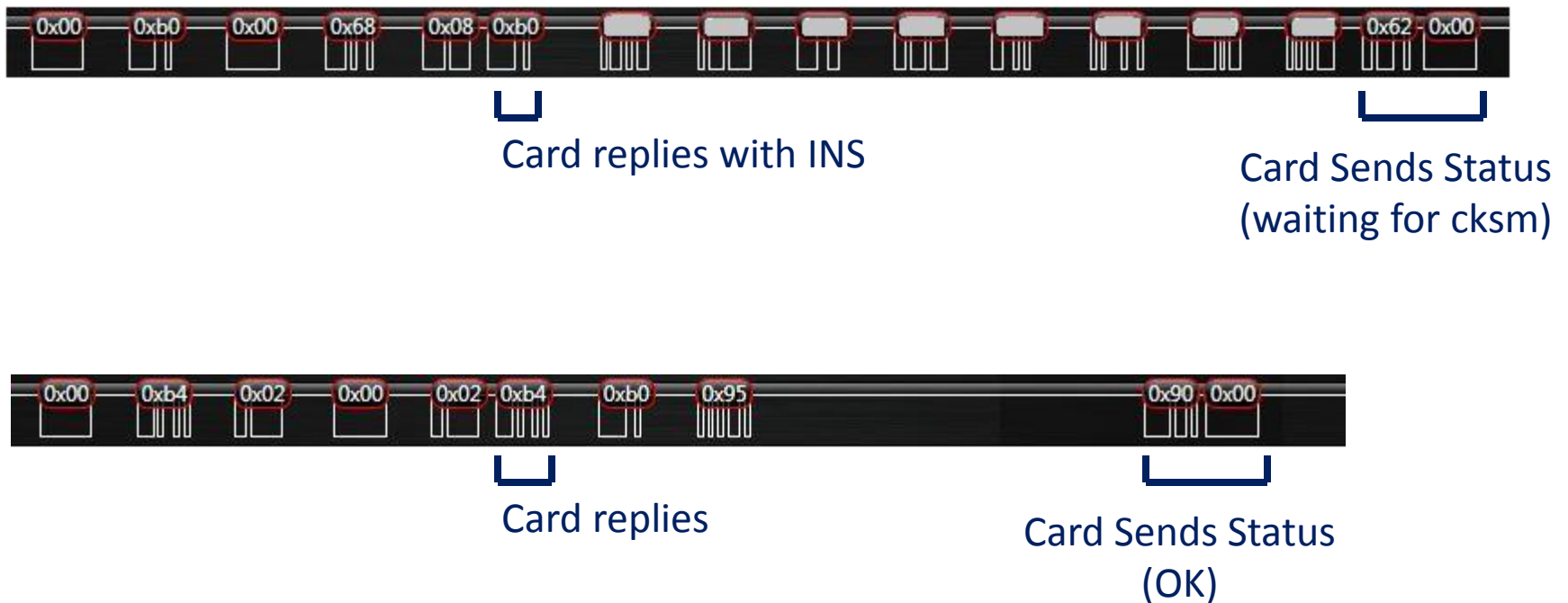
The Vulnerability

Recall the writing process...



The Vulnerability

... the Machine expects non-unique replies



The Vulnerability

THERE IS NO SECURE RECEIPT AFTER WRITING!

- The card confirms that data from the Host is legitimate
- However, the Host is not assured that the writing actually took place

The Vulnerability

What a hacker needs to do:

5-Character Command



The Vulnerability

What a hacker needs to do:

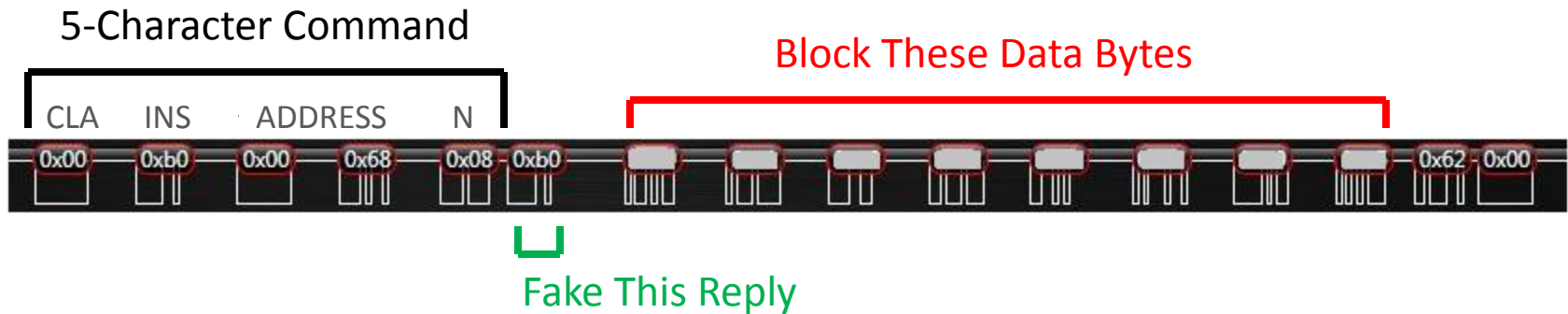
5-Character Command



Fake This Reply

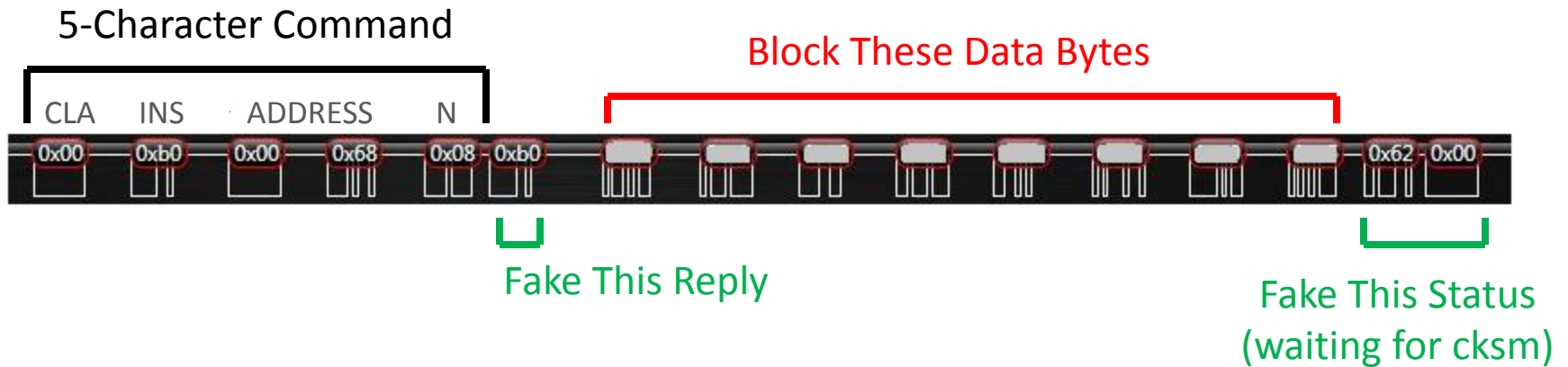
The Vulnerability

What a hacker needs to do:



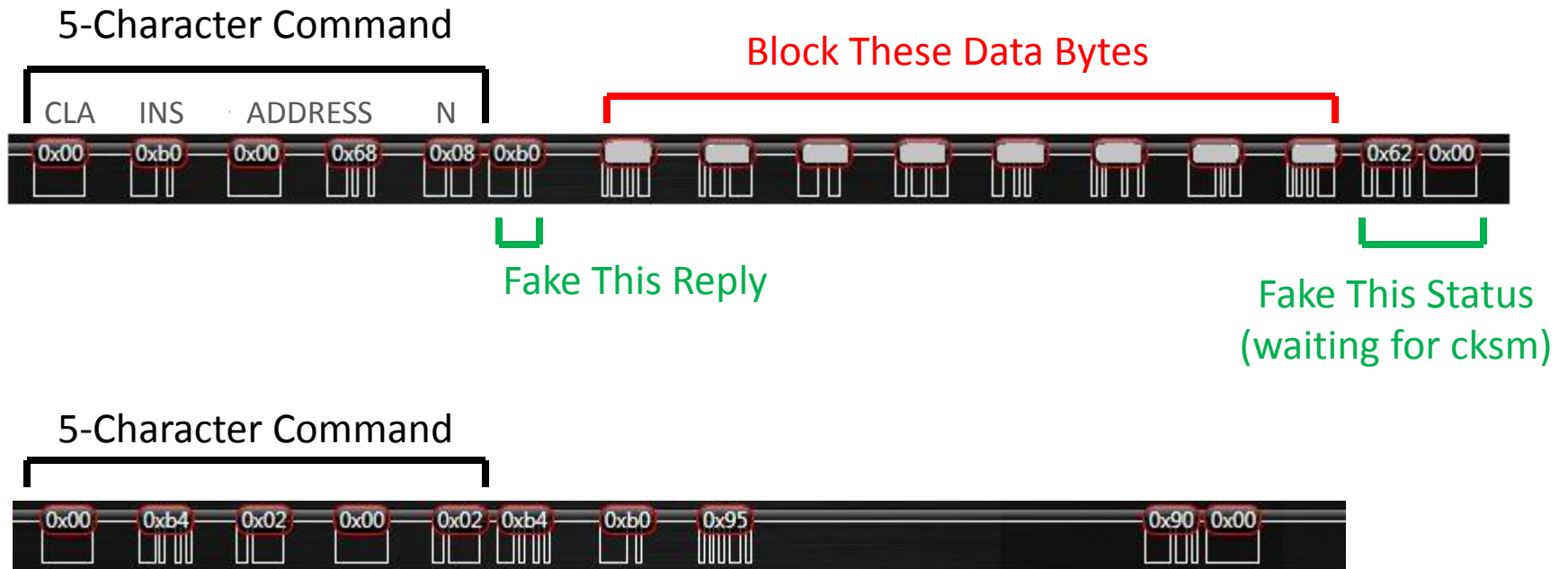
The Vulnerability

What a hacker needs to do:



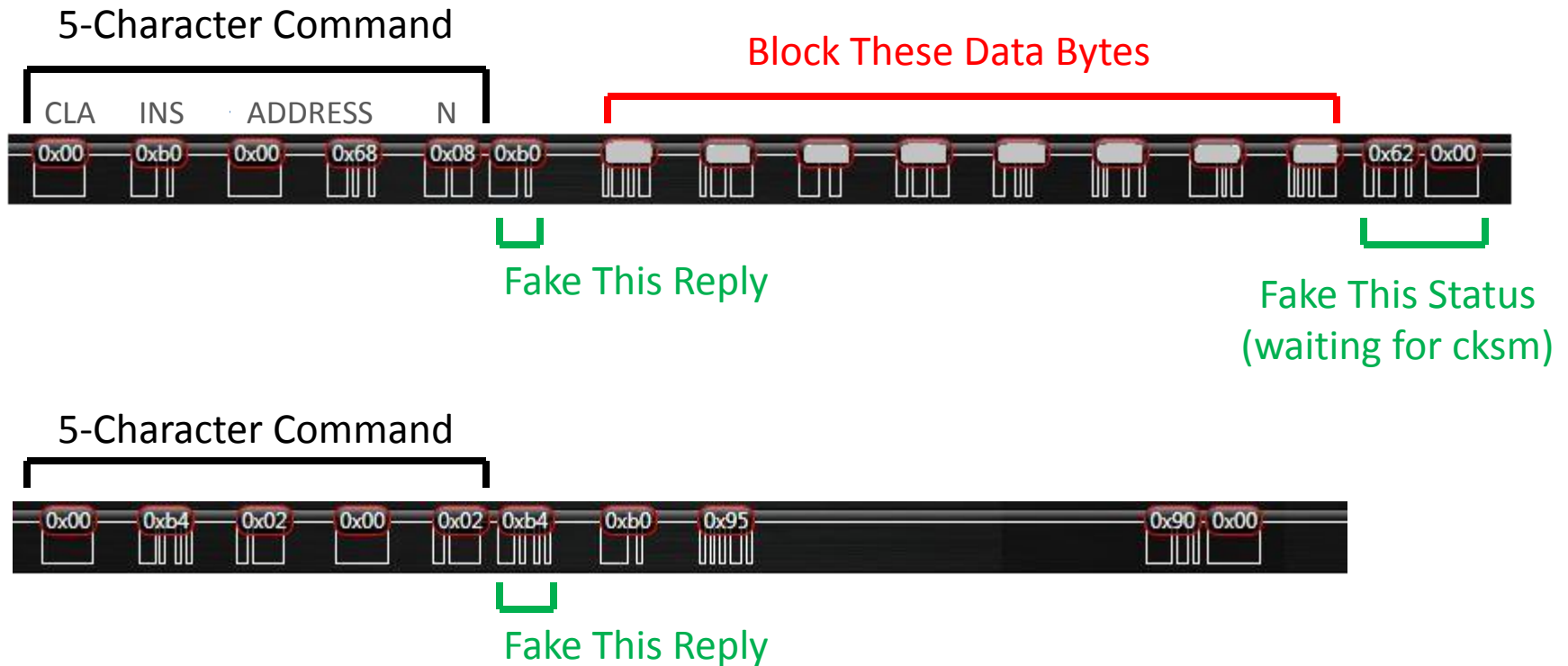
The Vulnerability

What a hacker needs to do:



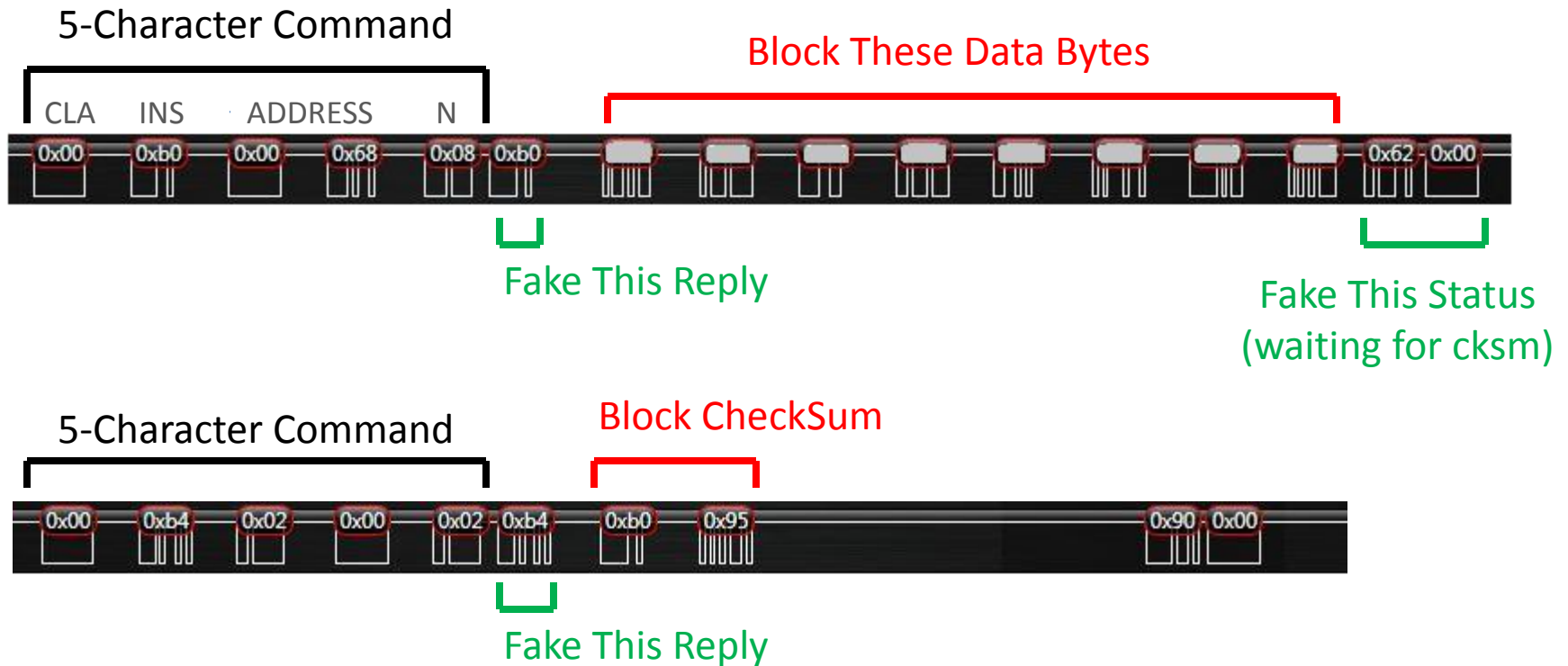
The Vulnerability

What a hacker needs to do:



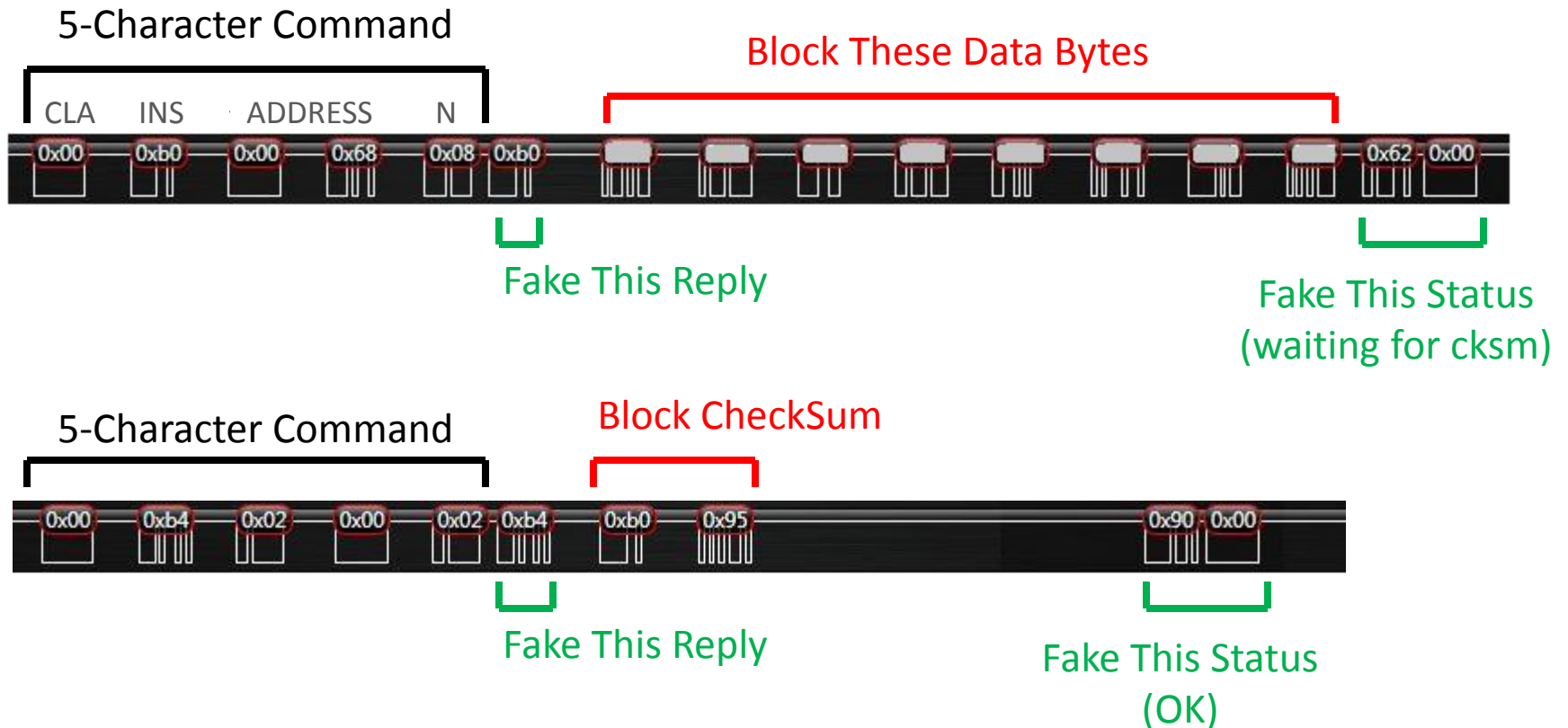
The Vulnerability

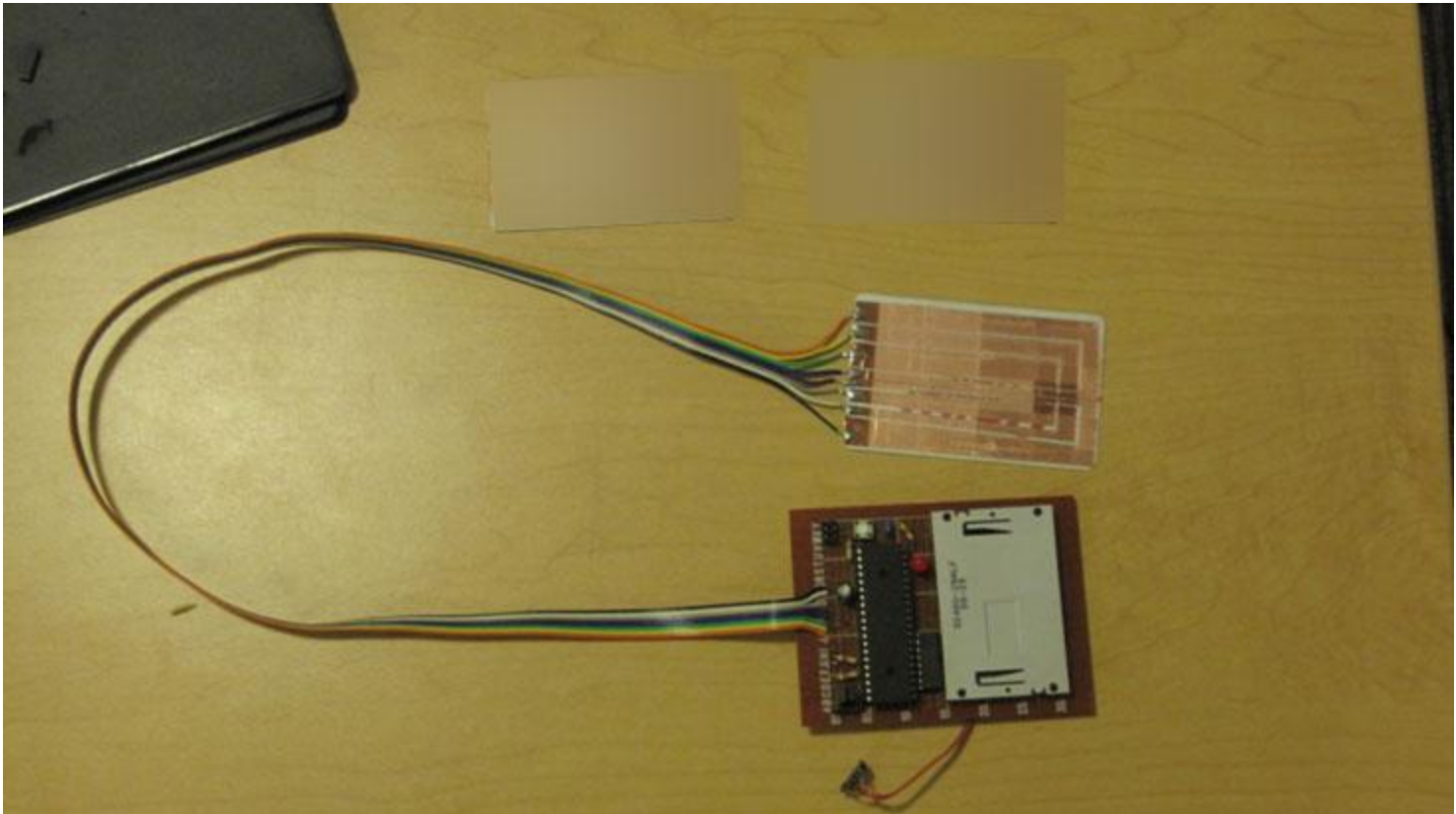
What a hacker needs to do:



The Vulnerability

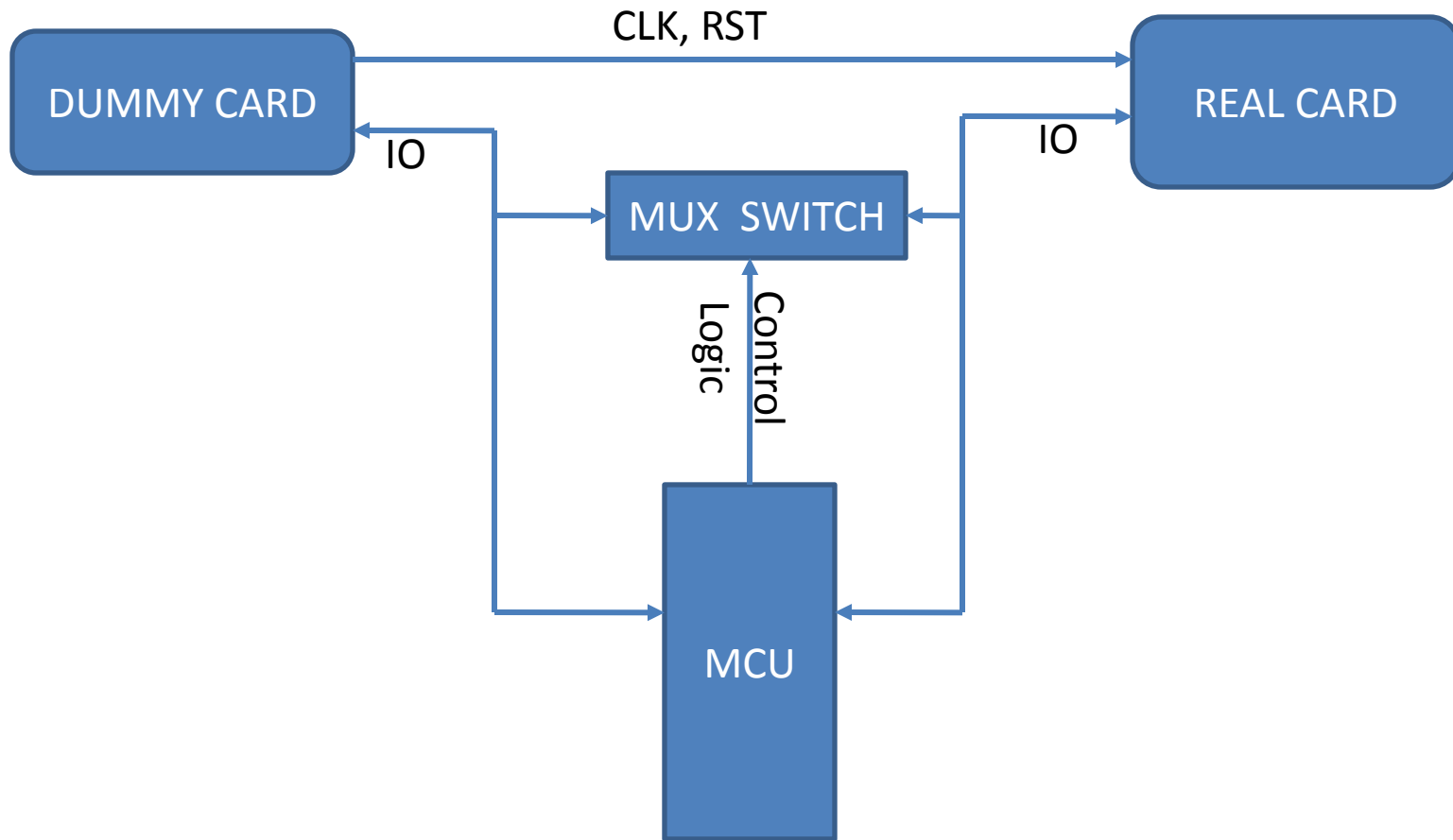
What a hacker needs to do:



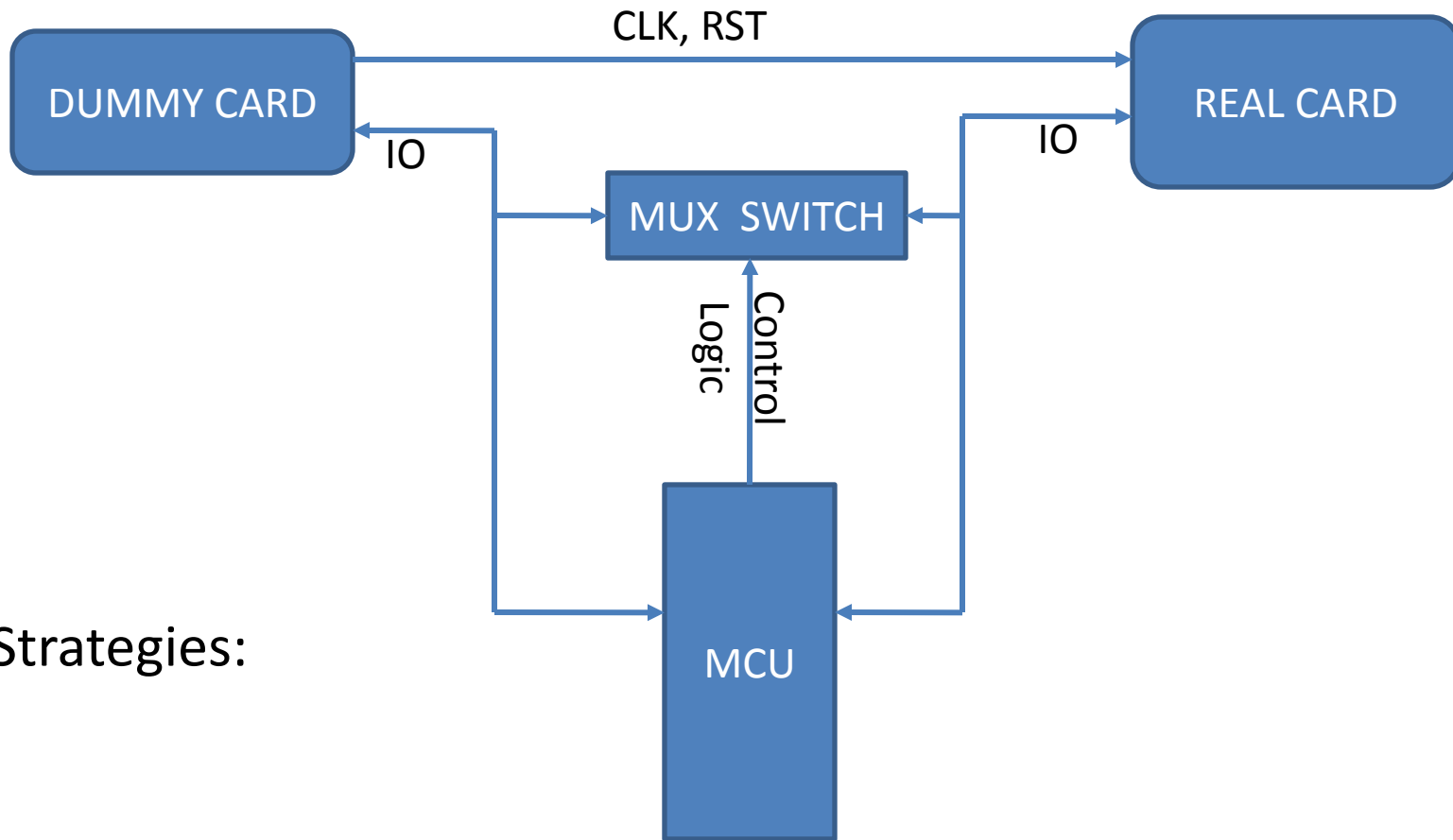


VULNERABILITY EXPLOITATION

Vulnerability Exploitation

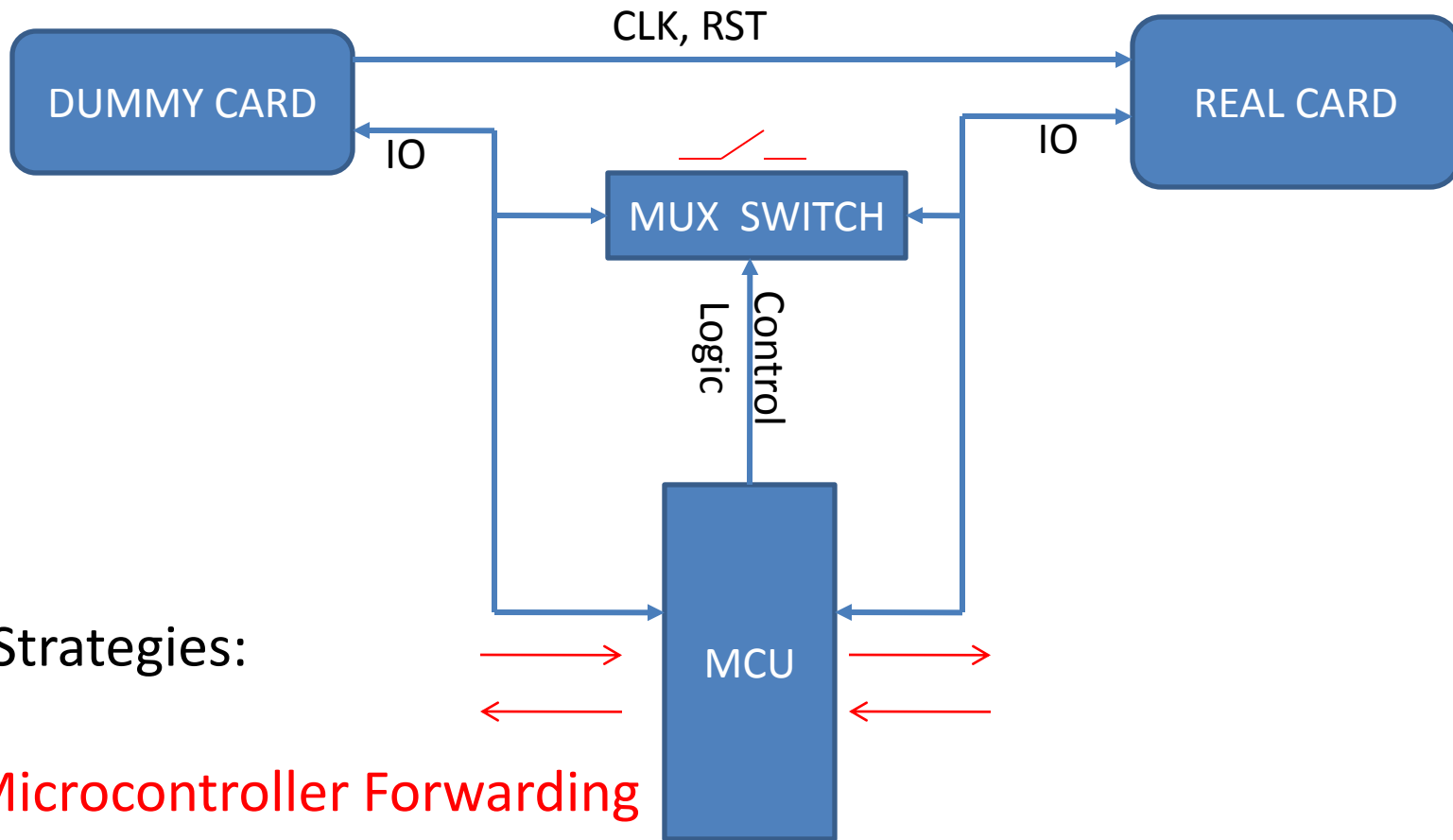


Vulnerability Exploitation

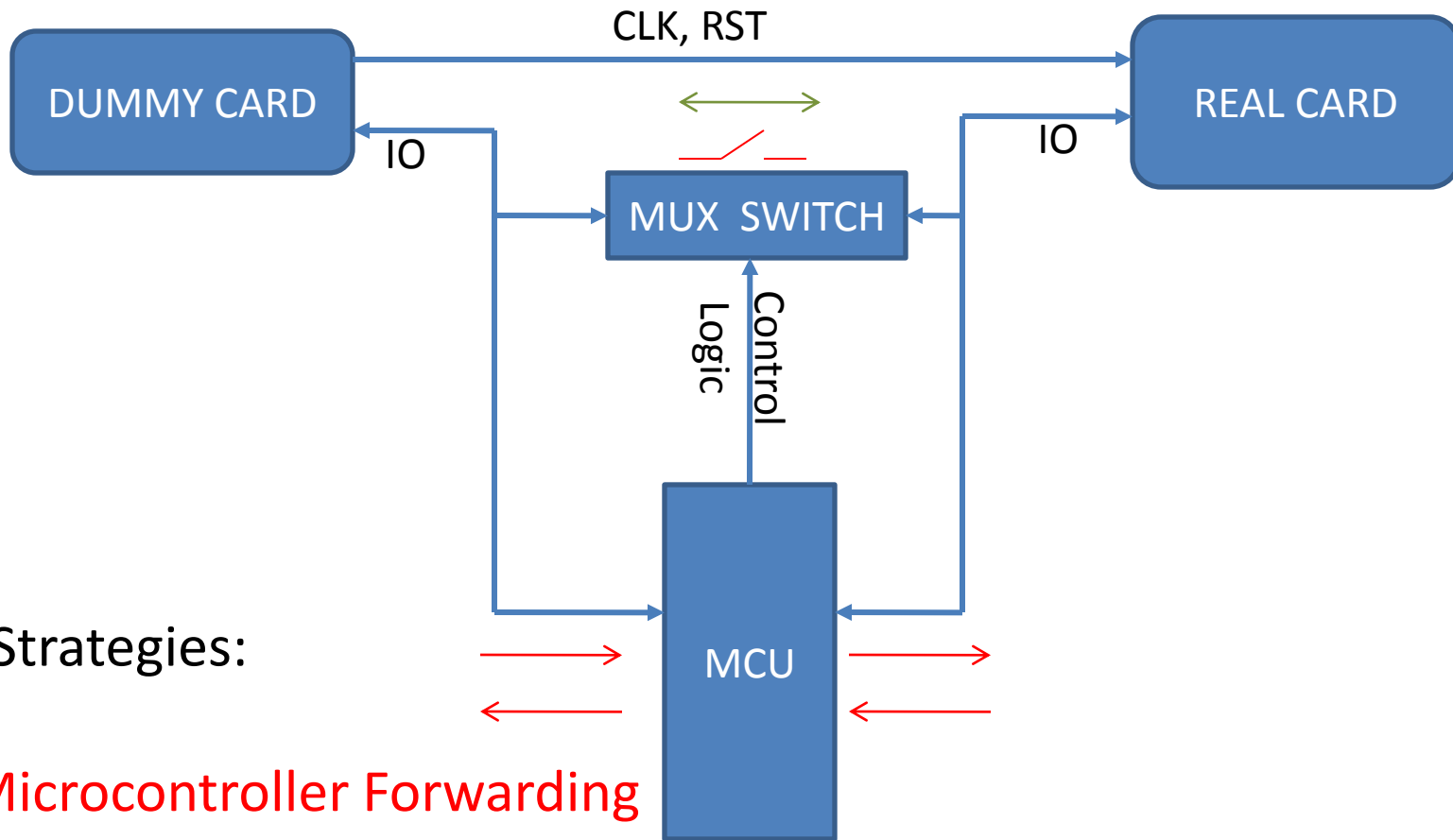


2 Strategies:

Vulnerability Exploitation



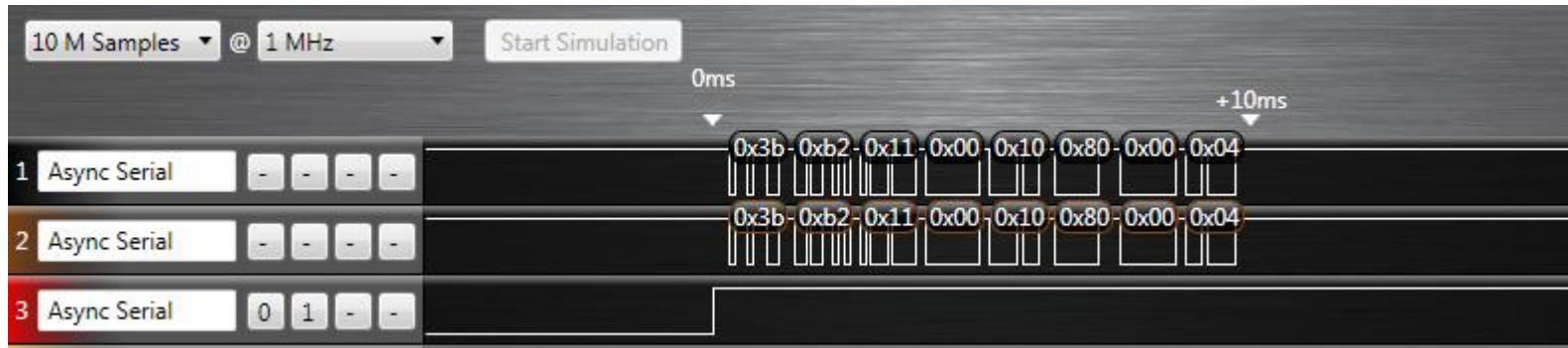
Vulnerability Exploitation



2 Strategies:

- Microcontroller Forwarding
- MUX Forwarding

Vulnerability Exploitation

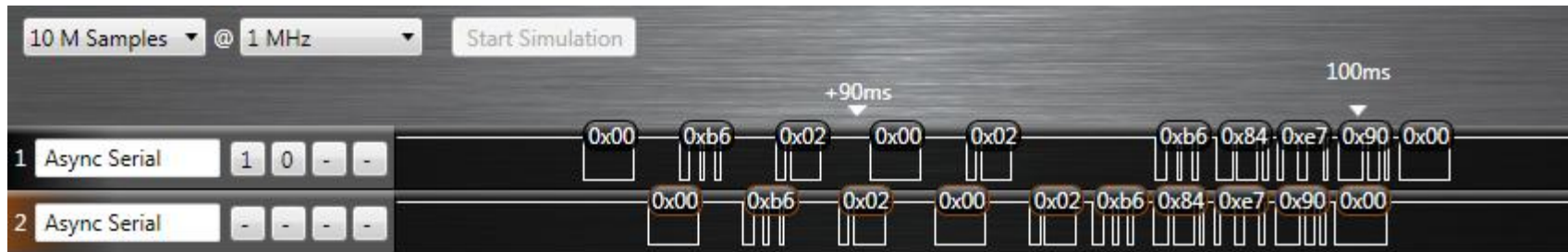
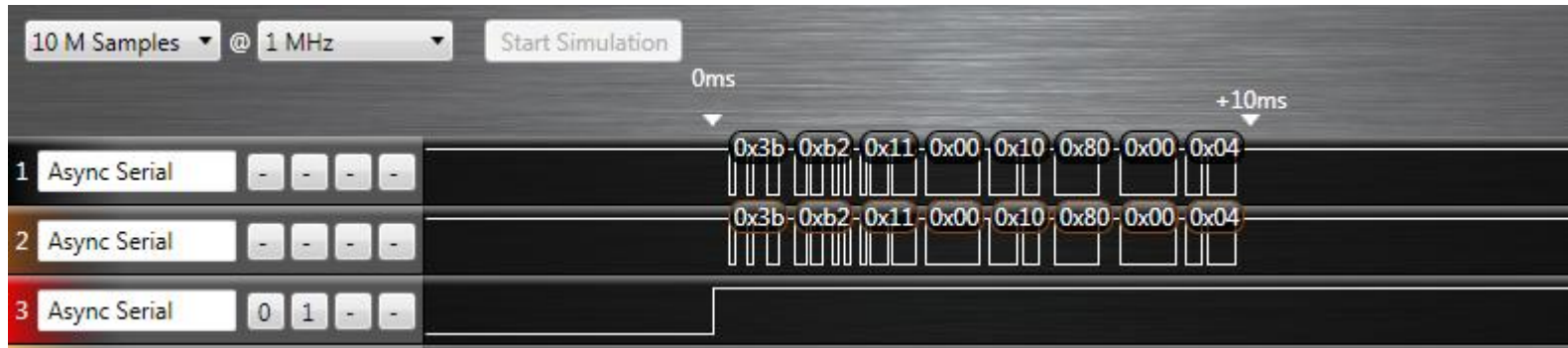


LINE 1 : Machine to Microcontroller Node

LINE 2 : Microcontroller to Card Node

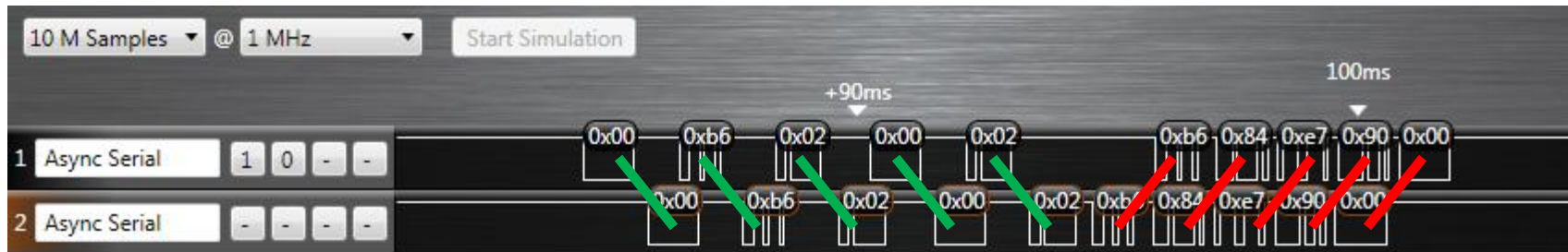
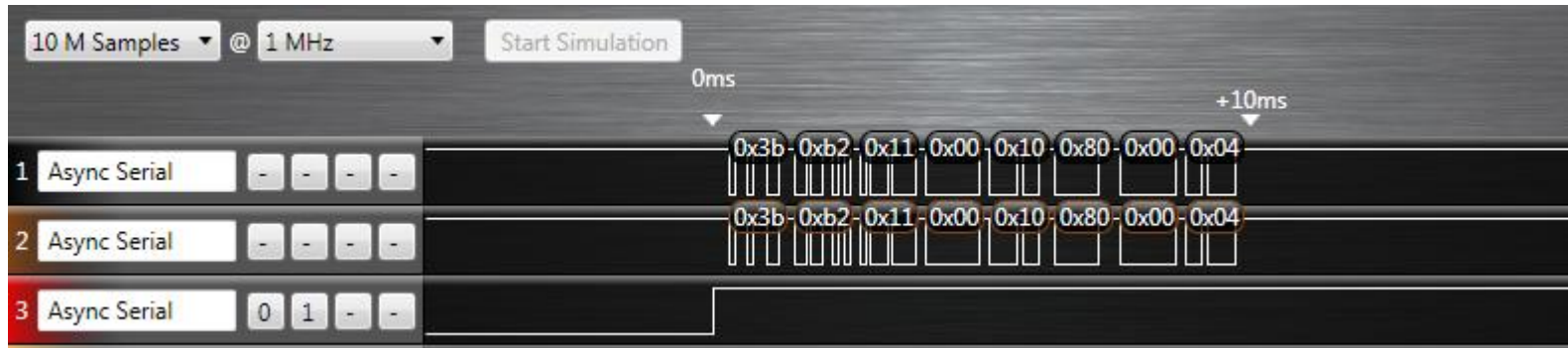
Answer To Reset is being forwarded through the MUX switch (No Delay)

Vulnerability Exploitation



Subsequent commands are being forwarded through the microcontroller

Vulnerability Exploitation




Subsequent commands are being forwarded through the microcontroller (1 Character Delay)

Vulnerability Exploitation



Vulnerability Exploitation



 Recognized Write Command

Vulnerability Exploitation



Recognized Write Command



Blocked all further communication to the card (Open Mux Switch)

Vulnerability Exploitation



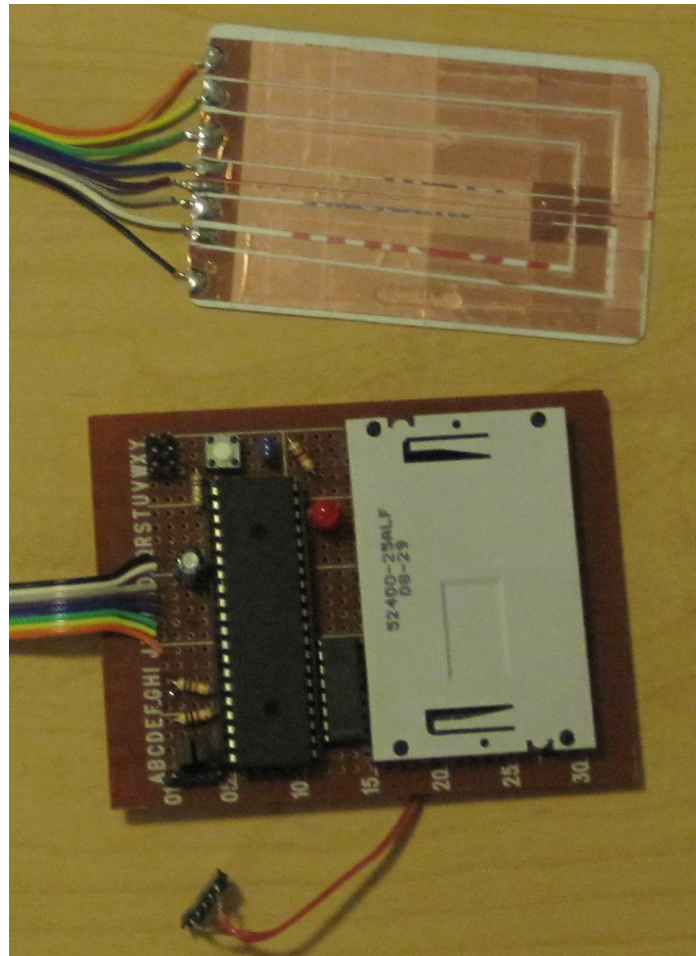
- Recognized Write Command
- Blocked all further communication to the card (Open Mux Switch)
- Forged Confirmation of Write Back to Machine

Vulnerability Exploitation

The Final Product:

Value of Components:

➤ ***Less than \$15***



Any More Vulnerabilities?

- Session Hijacking:
 - Allow system to reach the authenticated state
 - MIM communicates exclusively with smart card
 - Dump the protected user zones

Any More Vulnerabilities?

- Session Hijacking:
 - Allow system to reach the authenticated state
 - MIM communicates exclusively with smart card
 - Dump the protected user zones



Our Injected command to
read 0x7F bytes starting at
address 0x00

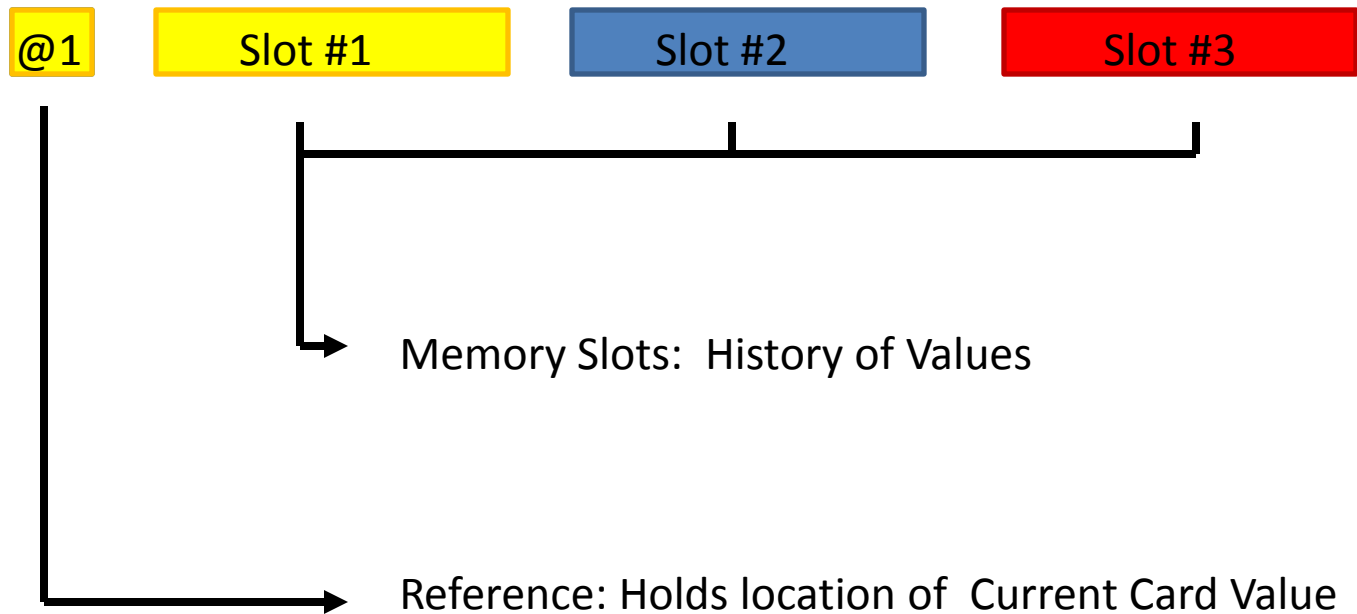
Smart Card begins spewing its
guts out

Any More Vulnerabilities?

- Analysing the Protected User Memory:

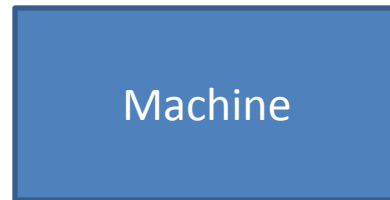
Any More Vulnerabilities?

- Analysing the Protected User Memory:
 - System uses History Buffer



Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



Times Used: **0**

@1

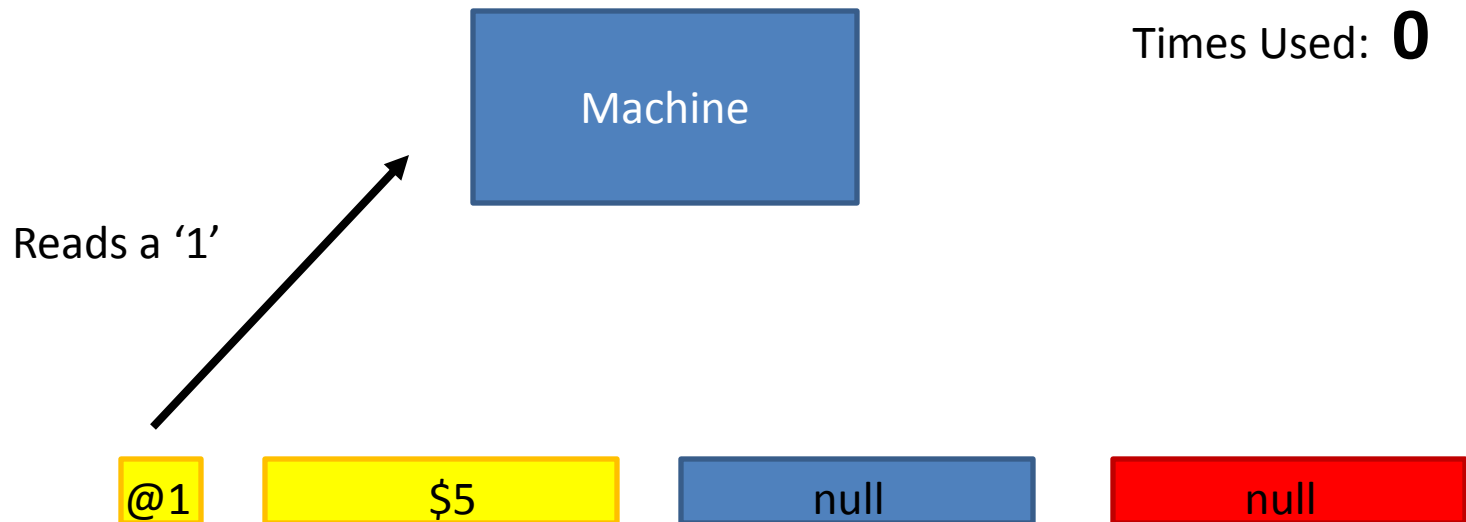
\$5

null

null

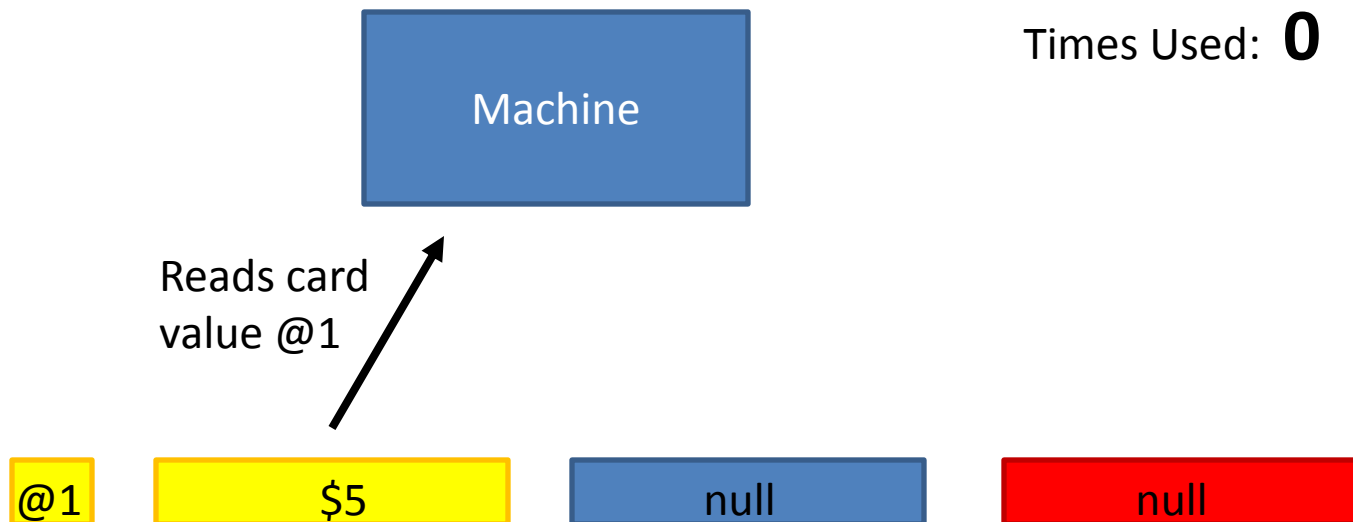
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



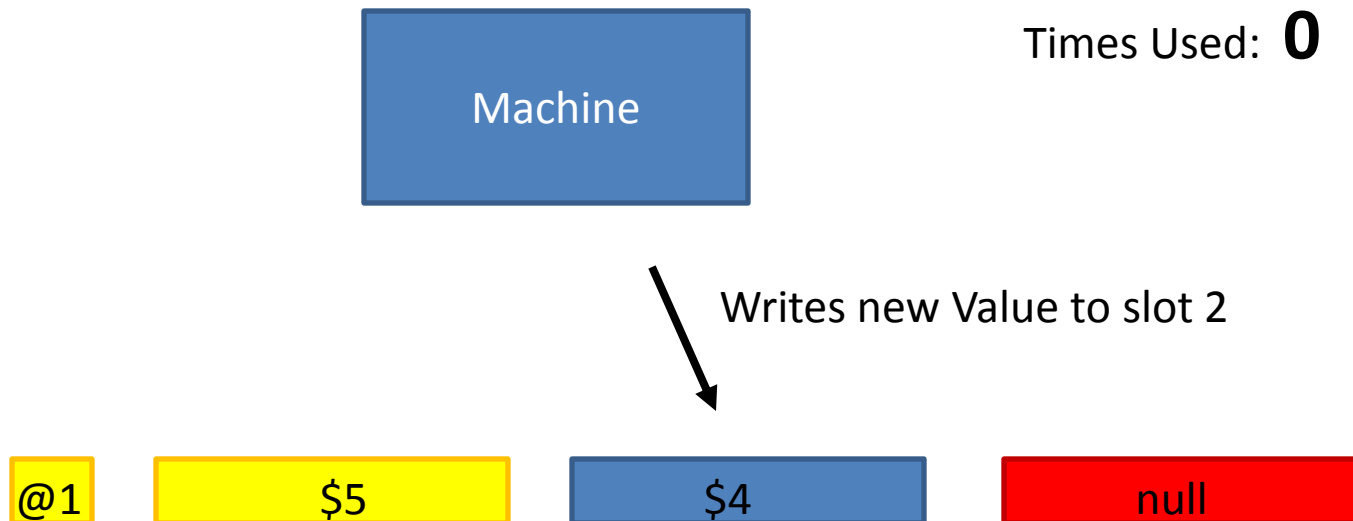
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



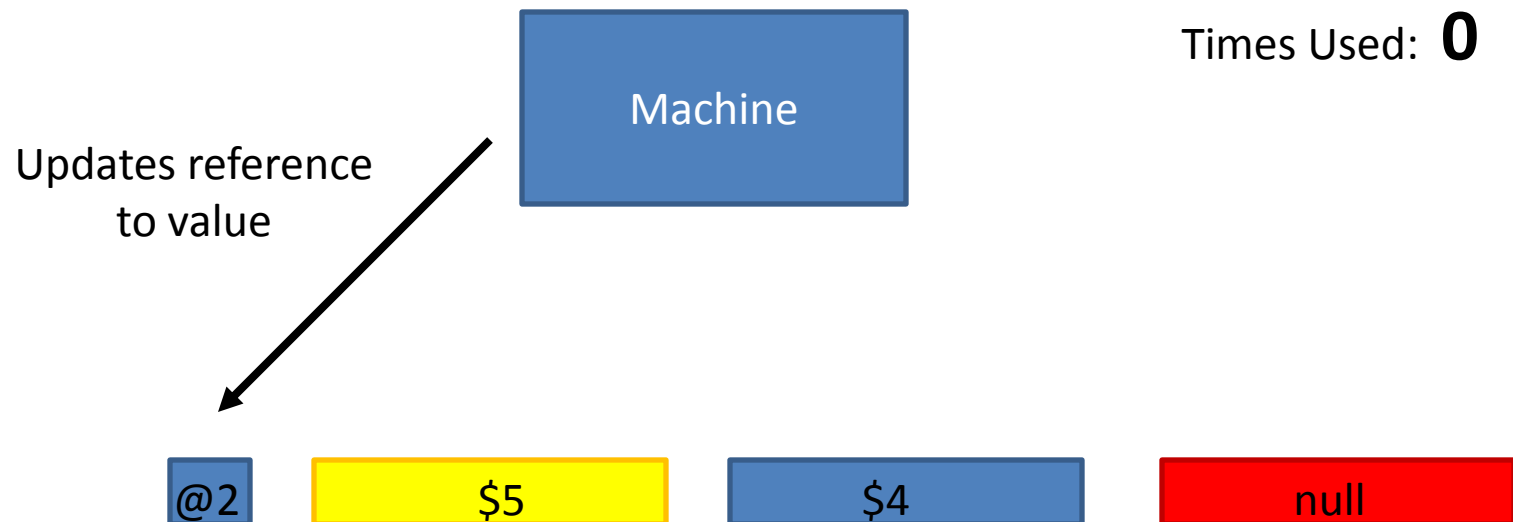
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



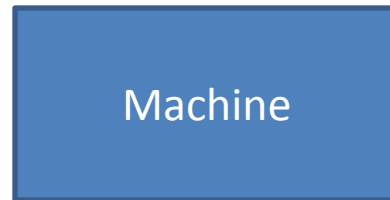
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



Times Used: **1**

@2

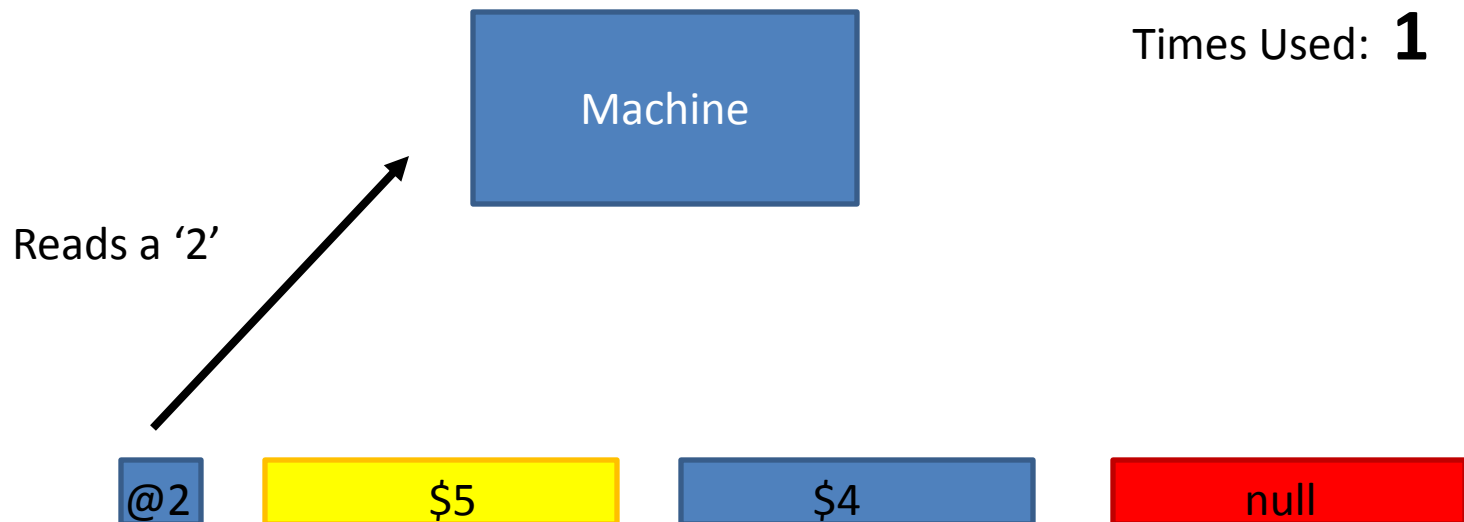
\$5

\$4

null

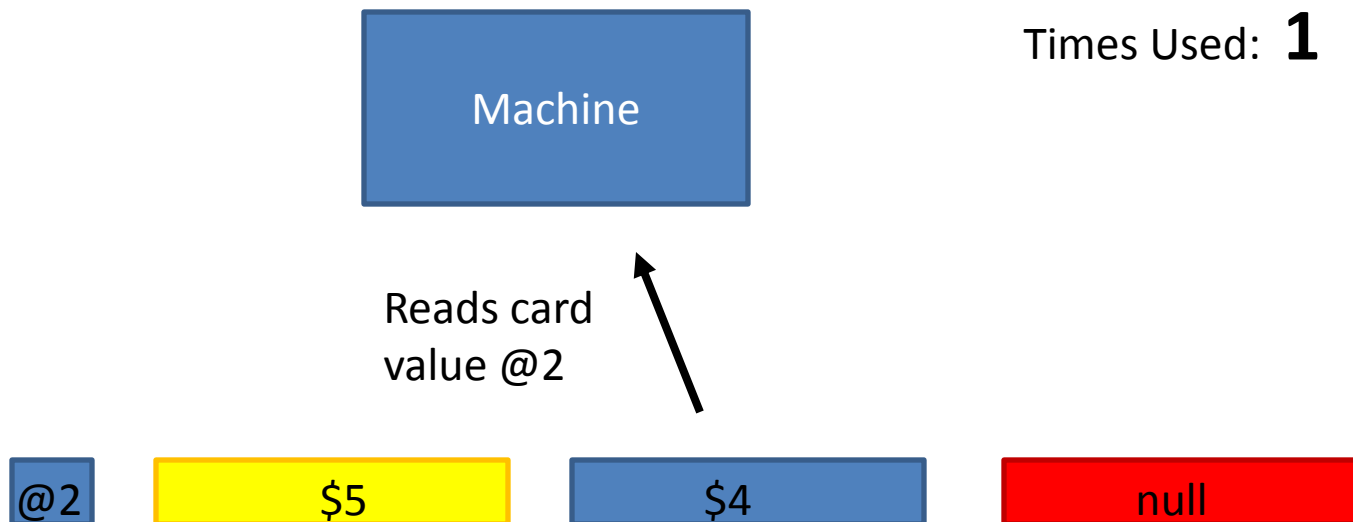
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



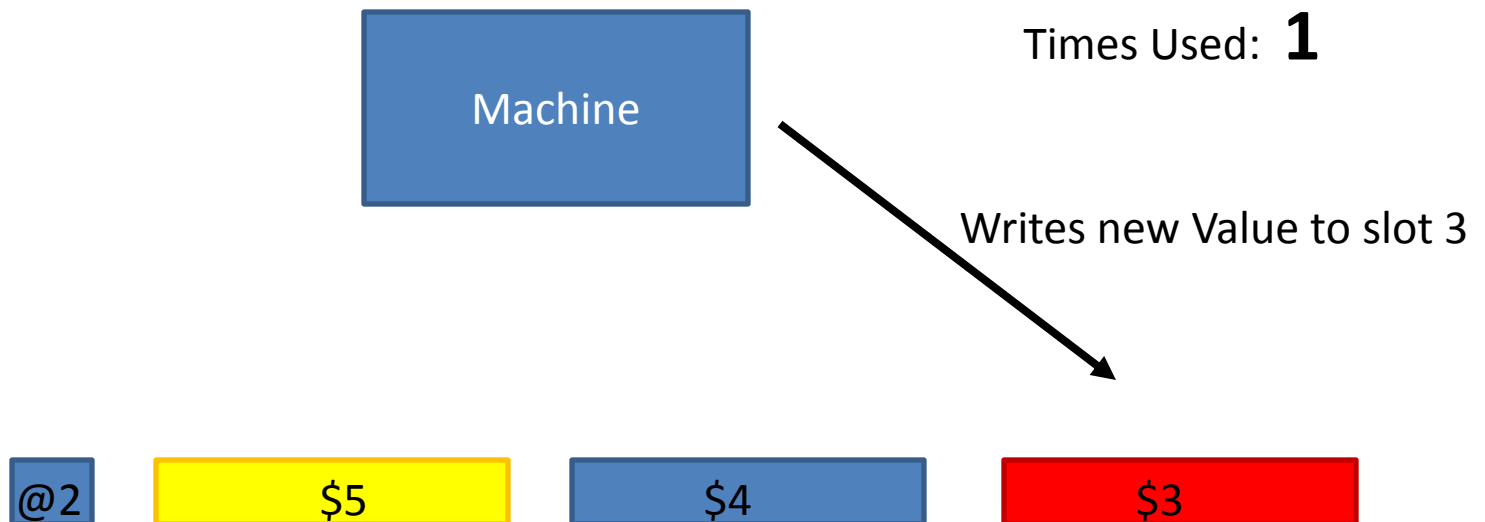
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



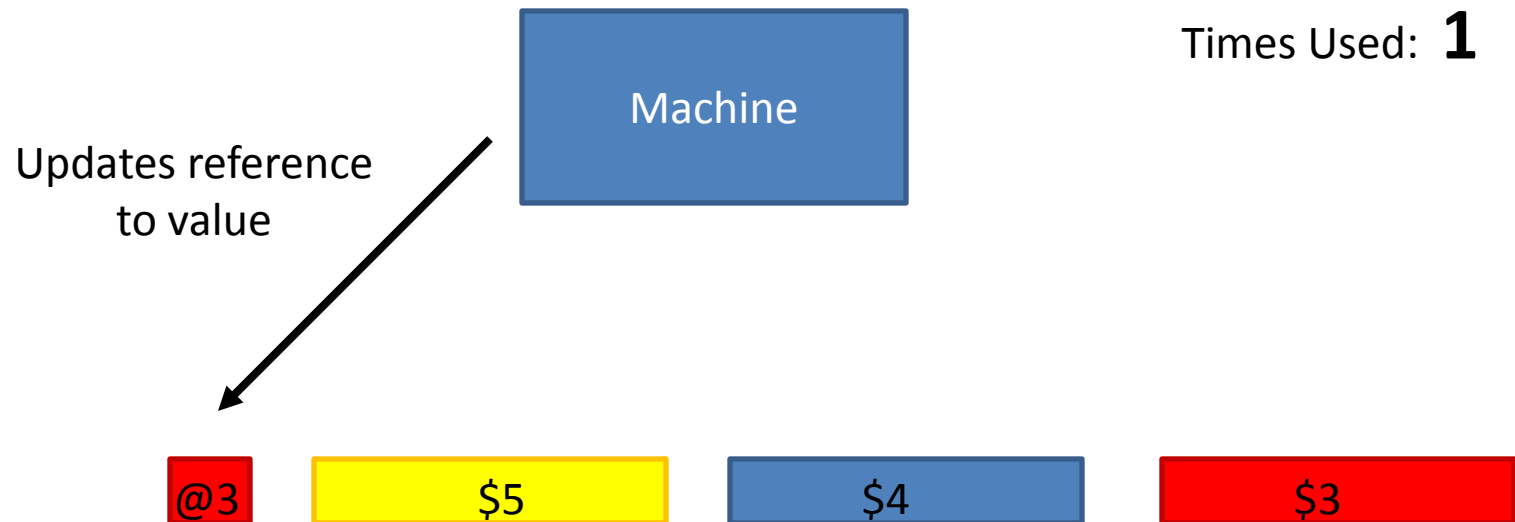
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



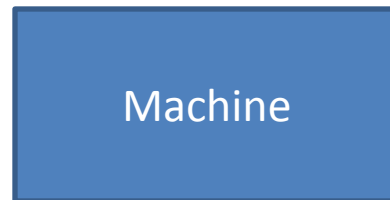
Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



Any More Vulnerabilities?

- Analysing the user memory:
 - System uses History Buffer
 - For example,



Times Used: **2**

@3

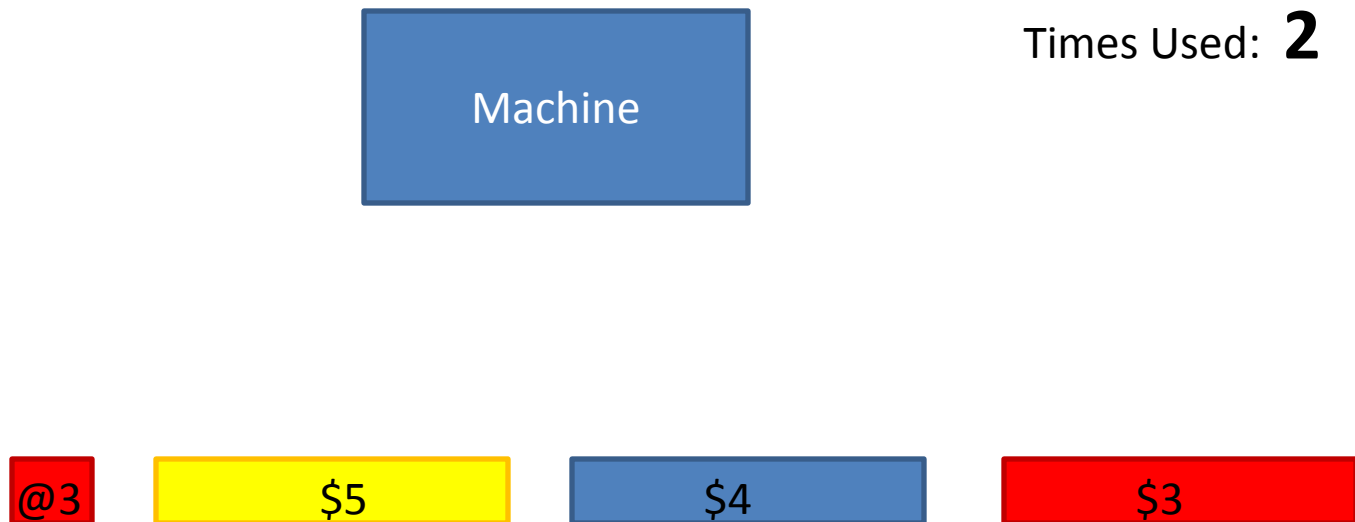
\$5

\$4

\$3

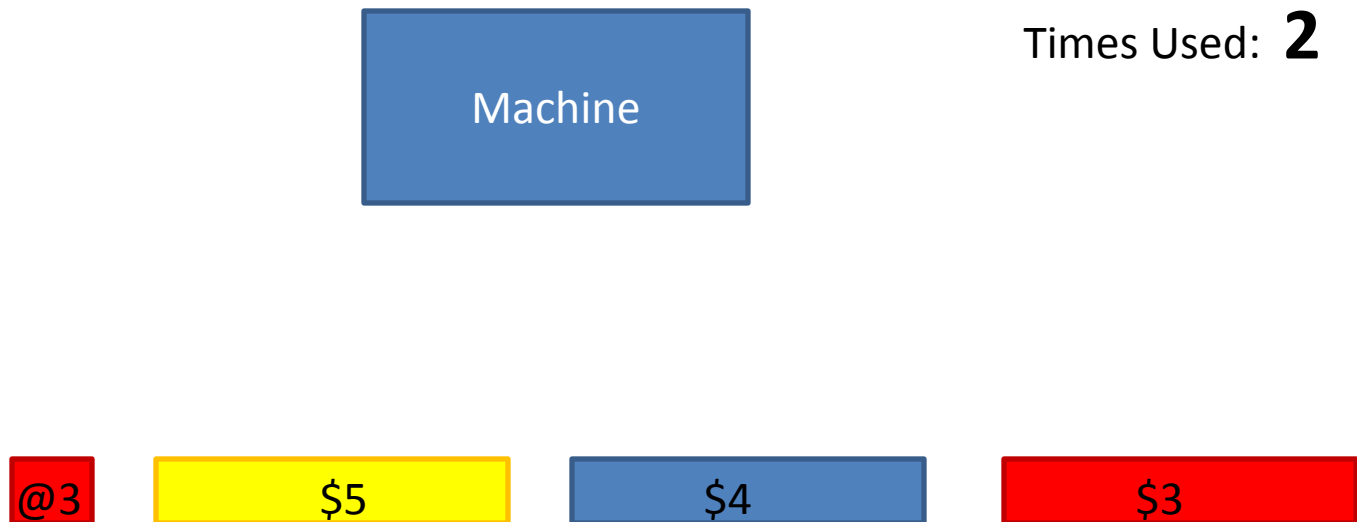
Any More Vulnerabilities?

- Can this system be manipulated?



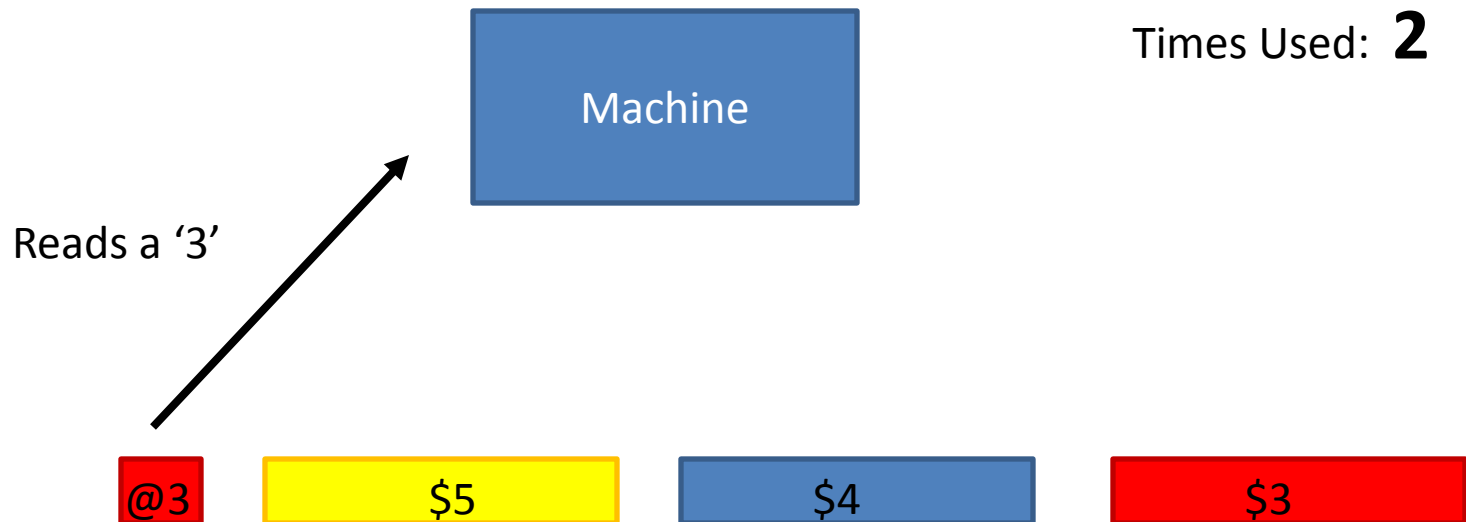
Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...



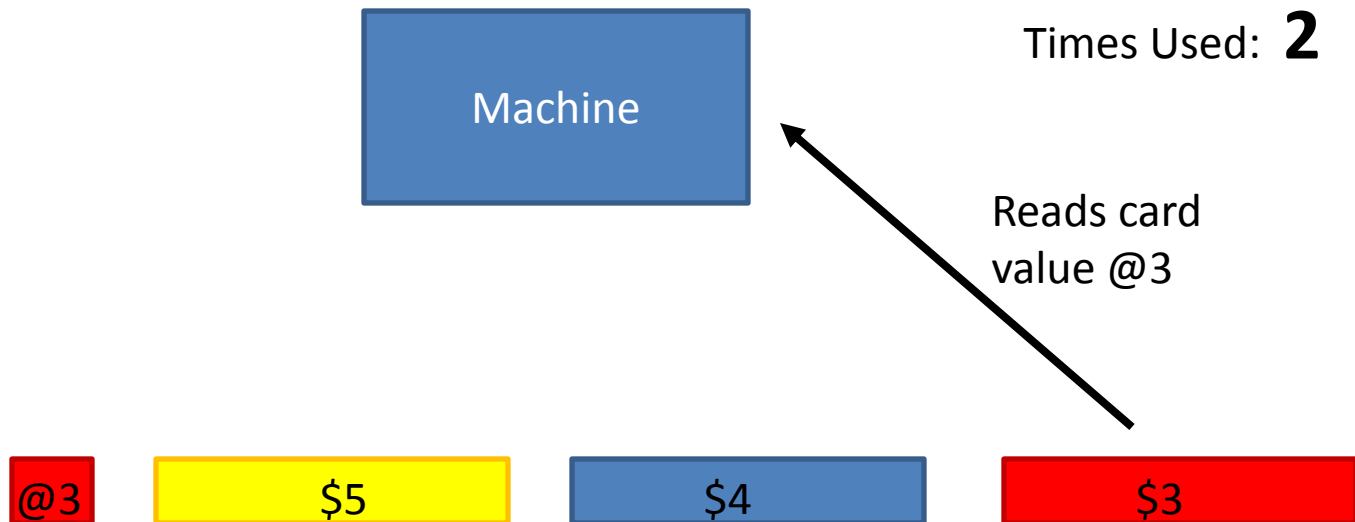
Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...



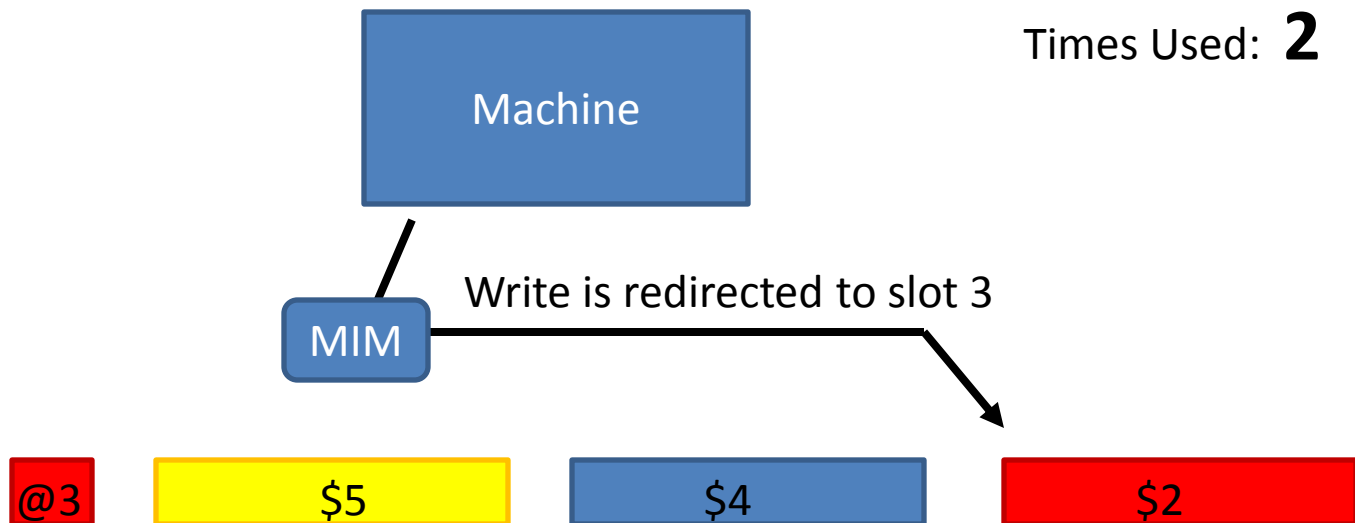
Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...



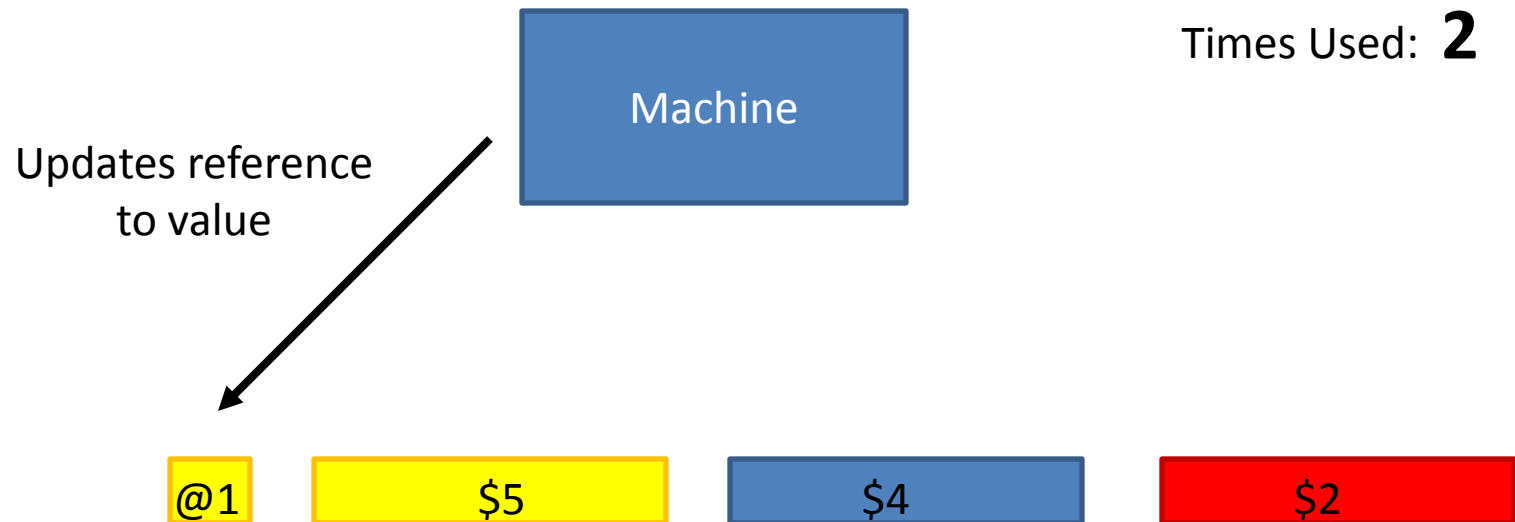
Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...



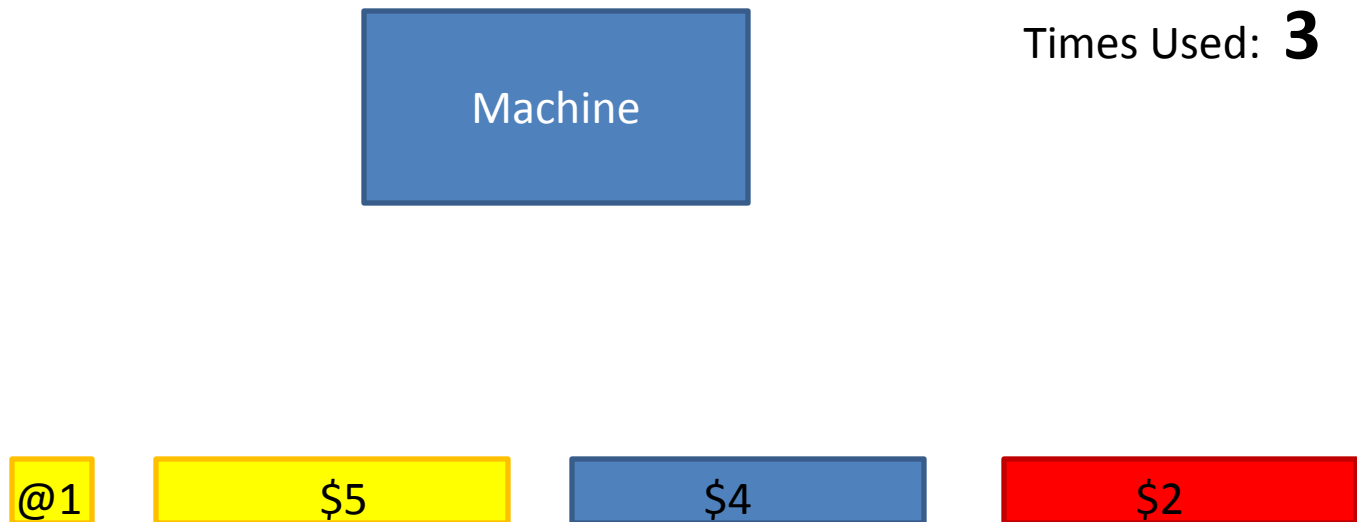
Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...



Any More Vulnerabilities?

- Can this system be manipulated?
 - When the final memory slot is reached, what if...

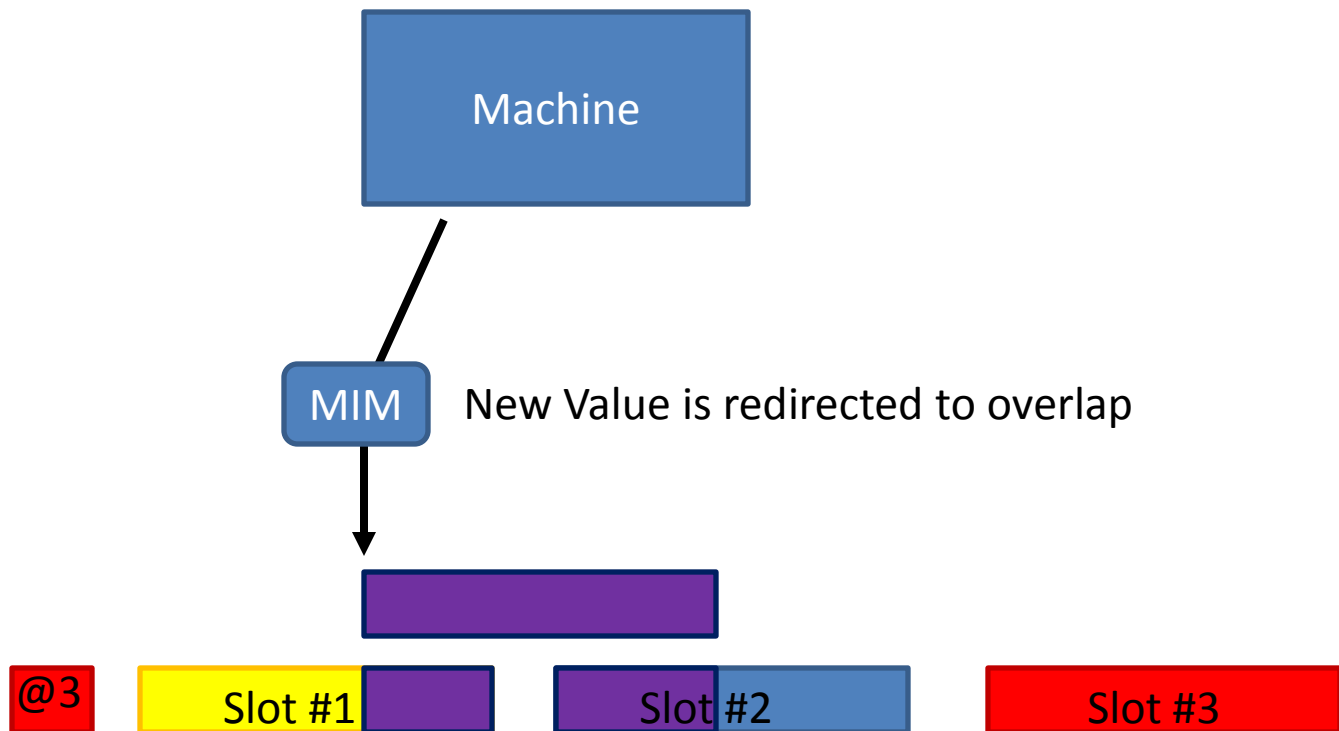


The card will now hold the original \$5 value!

Any More Vulnerabilities?

- This attack is possible if we are able to redirect the write command
- What else is possible with a redirected write?
 - Change the value maybe...?

Any More Vulnerabilities?

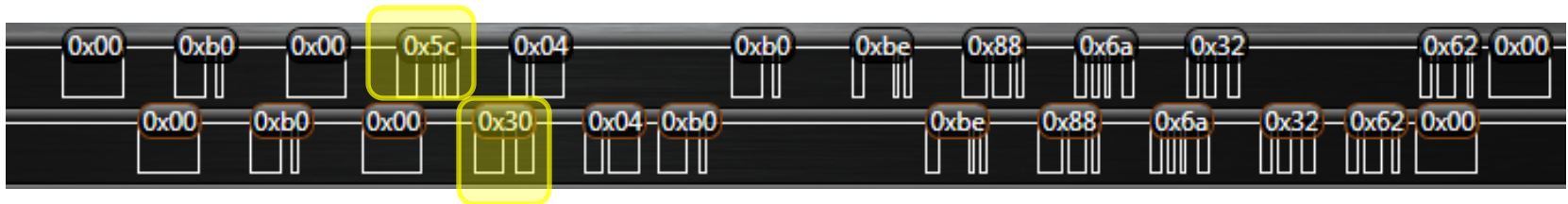


What value will slot #1 hold?

Any More Vulnerabilities?

- So we tried to redirect the write...

But, were denied. ☹️



Summary

- Communications that end in write command are not secure!
 - Even though Atmel App note recommends this exact behaviour



- Fix:
 - End Communication with a read command to verify the new card value

Summary

- Dump User Zone memory using MiM and session Hijacking
 - Fix:
 - Use encryption mode
 - Host side tamper detection(voltage/current monitor)

MONTAGE VIDEO TIME!!!