

# Shoehorning Security into the EPC Standard

Daniel V. Bailey and Ari Juels

`dbailey@rsasecurity.com` & `ajuels@rsasecurity.com`

`http://www.rsalabs.com`

RSA Laboratories

# *What is RFID and why does it matter?*

- Soon to be the most numerous computational devices in the world

# *What is RFID and why does it matter?*

- Soon to be the most numerous computational devices in the world
- Tiny tags applied to consumer goods, pharmaceutical items
- Asset and animal tracking
- Roadway toll-payment systems

## *What is RFID and why does it matter?*

- Soon to be the most numerous computational devices in the world
- Tiny tags applied to consumer goods, pharmaceutical items
- Asset and animal tracking
- Roadway toll-payment systems
- Spectacular growth driven by improved supply-chain visibility for fast-moving consumer goods
- and low-cost tags (expected to be five U.S. cents in a couple years)

# *What is RFID and why does it matter?*

- Soon to be the most numerous computational devices in the world
- Tiny tags applied to consumer goods, pharmaceutical items
- Asset and animal tracking
- Roadway toll-payment systems
- Spectacular growth driven by improved supply-chain visibility for fast-moving consumer goods
- and low-cost tags (expected to be five U.S. cents in a couple years)
- Wal-Mart and their suppliers are investing billions
  - Buying thousands of readers
  - ...and billions of tags
- Mostly to track cases and pallets as they travel through the global supply chain

# *What is RFID and why does it matter?*

- Soon to be the most numerous computational devices in the world
- Tiny tags applied to consumer goods, pharmaceutical items
- Asset and animal tracking
- Roadway toll-payment systems
- Spectacular growth driven by improved supply-chain visibility for fast-moving consumer goods
- and low-cost tags (expected to be five U.S. cents in a couple years)
- Wal-Mart and their suppliers are investing billions
  - Buying thousands of readers
  - ...and billions of tags
- Mostly to track cases and pallets as they travel through the global supply chain
- They expect a quick return on their investment
  - Supply-chain inefficiencies cost Wal-Mart billions
  - Items out-of-stock and supply-chain theft cost suppliers billions

## *What's new in RFID?*

- RFID has been around for a long time
- Primitive versions used during World War II

# What's new in RFID?

- RFID has been around for a long time
- Primitive versions used during World War II
- EPCglobal imagines a hierarichy of tags:
  - **Class-1: Identity Tags** Passive-backscatter tags offering only basic features like a fixed EPC identifier, a tag identifier, kill function, and optional password-protected access control
  - **Class-2: Higher-Functionality Tags** Passive tags with all of Class-1's features and extended tag identifier and user memory, as well as *authenticated access control*
  - **Class-3: Semi-Passive Tags** with all of Class-2's features as well as sensors and on-tag power sources like batteries
  - **Class-4: Active Tags** with all of Class-3's features as well as tag-to-tag communications and ad-hoc networking

## *What is C1G2?*

- RFID is a broad term that can cover everything from cell phones to contactless smart cards to electronic article surveillance tags
- Here we focus on EPCglobal UHF Class-1 Generation-2 (C1G2) tags
  - Designed to be the “tag of choice” for fast-moving consumer goods
  - These have longer range (10s of meters) than ISO 14443 or 15693
  - More tags inventoried per second
  - Use the UHF radio band - makes global operation difficult
  - Allegedly lower cost than 15693
  - Now ratified as ISO 18000-6C, but we’ll still call it C1G2

## *What is C1G2?*

- RFID is a broad term that can cover everything from cell phones to contactless smart cards to electronic article surveillance tags
- Here we focus on EPCglobal UHF Class-1 Generation-2 (C1G2) tags
  - Designed to be the “tag of choice” for fast-moving consumer goods
  - These have longer range (10s of meters) than ISO 14443 or 15693
  - More tags inventoried per second
  - Use the UHF radio band - makes global operation difficult
  - Allegedly lower cost than 15693
  - Now ratified as ISO 18000-6C, but we’ll still call it C1G2

# *RFID Controversy*

- But RFID is not without controversy

# *RFID Controversy*

- But RFID is not without controversy
- Many concerns have been voiced about unauthorized tracking
  - If everything you have from clothes to gadgets is RFID-enabled, you could be tracked without your knowledge
  - This transparency is what bothers many people

# RFID Controversy

- But RFID is not without controversy
- Many concerns have been voiced about unauthorized tracking
  - If everything you have from clothes to gadgets is RFID-enabled, you could be tracked without your knowledge
  - This transparency is what bothers many people
- In essence, we can call this the *rogue reader problem* and it has gotten most of the attention

# Counterfeit Tags

- This problem's opposite, or *counterfeit tags* is perhaps even more important
  - Item-level RFID will inevitably be used to discourage counterfeiting
  - The world-wide counterfeit goods trade reaches into the billions every year
  - Counterfeit pharmaceuticals are responsible for many deaths and injuries

# Counterfeit Tags

- This problem's opposite, or *counterfeit tags* is perhaps even more important
  - Item-level RFID will inevitably be used to discourage counterfeiting
  - The world-wide counterfeit goods trade reaches into the billions every year
  - Counterfeit pharmaceuticals are responsible for many deaths and injuries
- So we clearly have an authentication problem

# Counterfeit Tags

- This problem's opposite, or *counterfeit tags* is perhaps even more important
  - Item-level RFID will inevitably be used to discourage counterfeiting
  - The world-wide counterfeit goods trade reaches into the billions every year
  - Counterfeit pharmaceuticals are responsible for many deaths and injuries
- So we clearly have an authentication problem
- What shall we do?

# *EPCglobal Class-1 Gen-2 (C1G2) Logical Layer Protocol*

- C1G2 standard includes physical and logical layers

# *EPCglobal Class-1 Gen-2 (C1G2) Logical Layer Protocol*

- C1G2 standard includes physical and logical layers
- Both are brutally optimized for simple tags...and low cost
- In this talk, we focus on the logical layer
  - Simple reader-talks-first protocol allows the reader to obtain 128-bit Electronic Product Codes (EPCs) from multiple tags in rapid succession after issuing a single command to a population of tags

# *EPCglobal Class-1 Gen-2 (C1G2) Logical Layer Protocol*

- C1G2 standard includes physical and logical layers
- Both are brutally optimized for simple tags...and low cost
- In this talk, we focus on the logical layer
  - Simple reader-talks-first protocol allows the reader to obtain 128-bit Electronic Product Codes (EPCs) from multiple tags in rapid succession after issuing a single command to a population of tags
- Most tags perform this Query-Response pair and little else, delivering their 128-bit payload and going back to sleep
- A reader can additionally single-out a particular tag for a one-to-one conversation using simple commands like Read and Write
- Tags contain three memory spaces: EPC, TID, User (optional)
- Memory may be writable, but this again is optional

# Security in C1G2

- All tags must support one privileged operation: Kill
- Upon presentation of a 32-bit password from the reader, the tag must self-destruct
- Given the need for password validation, the standard's authors extended password-control to a couple other operations
  - Write/BlockWrite - A tag may require a password before updating on-tag data
  - Lock - A tag may selectively lock/unlock/permanently lock the ability to write data, including passwords themselves
  - Note that Read is NOT password-protected, making all tag data world-readable

# Security in C1G2

- All tags must support one privileged operation: Kill
- Upon presentation of a 32-bit password from the reader, the tag must self-destruct
- Given the need for password validation, the standard's authors extended password-control to a couple other operations
  - Write/BlockWrite - A tag may require a password before updating on-tag data
  - Lock - A tag may selectively lock/unlock/permanently lock the ability to write data, including passwords themselves
  - Note that Read is NOT password-protected, making all tag data world-readable
- There's no provision for cryptographic operations of any kind
- From the reader's point of view, the tag is just read/write storage
- So security protocols seem out of reach

## *Extending the Standard*

- As much as possible, we'd like to minimize changes to the protocol
- Ideally, we'd like to design a standards-compliant tag which can also do crypto
- Backward-compatibility (at least for non-privileged commands) is required!
- At the same time, we need to continue to focus on brutal optimization to reduce costs and design complexity

# *Extending the Standard*

- As much as possible, we'd like to minimize changes to the protocol
- Ideally, we'd like to design a standards-compliant tag which can also do crypto
- Backward-compatibility (at least for non-privileged commands) is required!
- At the same time, we need to continue to focus on brutal optimization to reduce costs and design complexity
- We face another fundamental problem: choosing primitives and protocols
- The cryptographic literature represents an embarrassment of riches. What to choose?

# *Extending the Standard*

- As much as possible, we'd like to minimize changes to the protocol
- Ideally, we'd like to design a standards-compliant tag which can also do crypto
- Backward-compatibility (at least for non-privileged commands) is required!
- At the same time, we need to continue to focus on brutal optimization to reduce costs and design complexity
- We face another fundamental problem: choosing primitives and protocols
- The cryptographic literature represents an embarrassment of riches. What to choose?
- We could spend a lot of time debating the relative merits of various schemes with an aim to select a single protocol
- Many standards efforts have gotten bogged down in the “algorithm wars” to noone's benefit

# *Extending the Standard*

- As much as possible, we'd like to minimize changes to the protocol
- Ideally, we'd like to design a standards-compliant tag which can also do crypto
- Backward-compatibility (at least for non-privileged commands) is required!
- At the same time, we need to continue to focus on brutal optimization to reduce costs and design complexity
- We face another fundamental problem: choosing primitives and protocols
- The cryptographic literature represents an embarrassment of riches. What to choose?
- We could spend a lot of time debating the relative merits of various schemes with an aim to select a single protocol
- Many standards efforts have gotten bogged down in the “algorithm wars” to noone's benefit
- Ultimately, supporting a range of applications with disparate security needs will require a diversity of primitives and protocols
- So we'll need some way for tag and reader to decide which security features they'll need

# ISO 7816

- We could develop a new sub-protocol to negotiate security features
- Many protocols like SSL do this today
- But we should re-use existing tools wherever possible

# ISO 7816

- We could develop a new sub-protocol to negotiate security features
- Many protocols like SSL do this today
- But we should re-use existing tools wherever possible
- ISO 7816 provides security services for smartcards in GSM phones, pay TV, and other applications
- It defines Application Protocol Data Units (APDUs) that tell the tag and reader to which protocol and primitive(s) a particular data payload corresponds
- It's also reader-talks-first, just like C1G2
- It offers a well-analyzed, robust set of features, but is still a bit heavyweight for our needs

# ISO 7816

- We could develop a new sub-protocol to negotiate security features
- Many protocols like SSL do this today
- But we should re-use existing tools wherever possible
- ISO 7816 provides security services for smartcards in GSM phones, pay TV, and other applications
- It defines Application Protocol Data Units (APDUs) that tell the tag and reader to which protocol and primitive(s) a particular data payload corresponds
- It's also reader-talks-first, just like C1G2
- It offers a well-analyzed, robust set of features, but is still a bit heavyweight for our needs
- But how would we implement ISO 7816 using only the tools available in C1G2?

# *Protocol Convergence*

- All we can do with the C1G2 protocol is read and write memory
- But the standard doesn't say that data written to memory can't be changed by the tag!

# *Protocol Convergence*

- All we can do with the C1G2 protocol is read and write memory
- But the standard doesn't say that data written to memory can't be changed by the tag!
- So we designate two areas of memory and implement a crude form of shared-memory interprocess communication
- The reader uses the BlockWrite command to write an ISO 7816 APDU to the tag
- The tag receives the APDU, processes it, and writes a response to a different area of memory
- The reader reads this area to retrieve its reply

# Protocol Convergence

- All we can do with the C1G2 protocol is read and write memory
- But the standard doesn't say that data written to memory can't be changed by the tag!
- So we designate two areas of memory and implement a crude form of shared-memory interprocess communication
- The reader uses the BlockWrite command to write an ISO 7816 APDU to the tag
- The tag receives the APDU, processes it, and writes a response to a different area of memory
- The reader reads this area to retrieve its reply

	Command	MemBank	WordPtr	WordCount	Data	Handle	CRC-16
Number of bits	8	2	EBV	8	Variable	16	16
Description	11000111	11	0000000	Number of words to write	APDU	handle	

## *Reading a Response APDU*

- If we're designating fixed memory locations for the APDU transfers, we can instantiate these as SRAM instead of EEPROM
  - This approach saves cost and power, as well as time
  - Time that otherwise is reserved (20 msec) in the protocol to complete a EEPROM operation can be used to compute an APDU
  - The standard calls for the reader to simply transmit a Continuous Wave (CW) during this time to power the tag

## Reading a Response APDU

- If we're designating fixed memory locations for the APDU transfers, we can instantiate these as SRAM instead of EEPROM
  - This approach saves cost and power, as well as time
  - Time that otherwise is reserved (20 msec) in the protocol to complete a EEPROM operation can be used to compute an APDU
  - The standard calls for the reader to simply transmit a Continuous Wave (CW) during this time to power the tag

	Command	MemBank	WordPtr	WordCount	Handle	CRC-16
Number of bits	8	2	EBV	8	16	16
Description	11000010	11	0000000	Number of words to read	handle	

# *Implementing ISO 7816 Features*

- ISO 7816-4 defines general command and response frames
- It further specifies instantiations of these to perform tasks like entity authentication of tag, reader, or both
- ...as well as transfer of encrypted or integrity-protected data.

# Implementing ISO 7816 Features (2)

- What does an ISO 7816 APDU look like?

Field	Description	Number of bytes
Command header	Class byte denoted CLA	1
Command header	Instruction byte denoted INS	1
Command header	Parameter bytes denoted P1-P2	2
Command data-length $L_c$	Absent if $N_c = 0$ , otherwise equal to $N_c$	0, 1, or 3
Command data	Absent if $N_c = 0$ , otherwise a string of $N_c$ bytes	$N_c$
Maximum response length	Absent if $N_e = 0$ , otherwise equal to $N_e$	0, 1, or 3

Field	Description	Number of bytes
Response data	Absent if $N_r = 0$ , otherwise a string of $N_r$ bytes	$N_r$
Response trailer	Status bytes SW1 and SW2	2

## *Implementing ISO 7816 Features (3)*

- With this set of headers, data lengths, and trailers, the reader can unambiguously specify precisely which command is desired
- along with details on algorithms, protocols, parameters, key identifiers, and of course, command data.
- The tag can reply with status bytes indicating success, reasons for failure, or the fact that processing has not yet completed.

## *Implementing ISO 7816 Features (3)*

- With this set of headers, data lengths, and trailers, the reader can unambiguously specify precisely which command is desired
- along with details on algorithms, protocols, parameters, key identifiers, and of course, command data.
- The tag can reply with status bytes indicating success, reasons for failure, or the fact that processing has not yet completed.
- To address our anti-counterfeiting needs, let's focus on one 7816 command called Internal Authenticate
- In our examples, we'll suppose a simple unidirectional challenge-response protocol for authentication
- The reader sends a 64-bit challenge and gets a 64-bit response
- A typical command sees six bytes overhead; a response two.
- Can we reduce this cost?

# Internal Authenticate

Field	Description	Number of bytes
Command Class Byte	0h	1
Command Instruction Byte	88h	1
Command Parameters	0h	2
Command data-length	8h	1
Command data	$C_R$	8
Maximum response length	8h	1

Field	Description	Number of bytes
Response data	$R_T$	8
Status bytes	6100h	2

# *Internal Authenticate Sizes*

<b>Frame Type</b>	<b>Bytes</b>	<b>Bits</b>
BlockWrite Carrying Internal Authenticate with Challenge	22	169
BlockWrite Carrying Internal Authenticate with Implicit Challenge	13	97
Read Carrying Response	18	137

# Optimizations

- Most tags will continue to be low-cost single-application devices
- So they'll have a single security protocol they perform
- Be it tag- or reader-authentication, encryption or some combination of these
- So we optimize a single protocol while still allowing the full flexibility of ISO 7816 as a fallback
- We introduce a security sublayer to signal which of the Command Class Byte, Command Instruction Byte, and Command Parameters will be suppressed, saving up to 21 bits

	Command Class Byte	Command Instruction Byte	Command Parameters
Number of bits	1	1	1

## Optimizations (2)

- To save even more bits over the air, we can turn to new commands.
- In our use of the Read and BlockWrite commands, quite a few bits are devoted to specifying a memory location and data length.
- A new or custom command's identifier would directly imply the memory location
- In addition, the ISO 7816 APDU either specifies its own length explicitly in the DataLen field, or – like other parameters – it is previously known by both parties, allowing us to optionally dispense with the WordCount field.
- By defining a new command we can save 17 bits by using an unreserved 8-bit identifier, of which there are 22 currently available.

# New EPC-layer Command/Response Frames

	Command	Header	Compressed 7816 APDU	Handle	CRC-16
Number of bits	8	3	Variable	16	16
Description	11001001		$C_R$	handle	

	Header	Handle	CRC-16
Number of bits	1	16	16
Description	0	handle	

# Results

- We show how to implement ISO 7816 over C1G2
- The resulting C1G2 tag is fully compliant with existing EPCglobal standards
- Could serve as the basis for a C2G1 standard

Frame Type	Bytes	Bits
New EPC-layer Command for ISO 7816 APDU Command with Challenge	15	123
New EPC-layer Command for ISO 7816 APDU Command with Implicit Challenge	6	51
Custom EPC-layer Command for ISO 7816 APDU Command with Challenge	16	131
Custom EPC-layer Command for ISO 7816 APDU Command with Implicit Challenge	8	59
EPC-layer Tag Reply to Response APDU	15	113