First we break your tag, then we break your System
Attacks to RFID Systems

Lukas Grunwald

Phreak.de

www.phreak.de

Agenda

- What is RFID?
- How to exploit and attack RFID systems
- Attacks against the middleware
- Reader-emulation, soft-tags
- Unexpected risk middleware
- New ways to exploit the system
- Encrypted RFID Tags (14443, MRTD)

What is RFID?

- Radio Frequency Identification (RFID)
  - Wireless transmission of information between transponder and reader without visibility
  - Bidirectional transfer (read and write)
  - Transponder (tag) can be attached, embedded or implanted
  - Automatic correlation between object and saved data

Generic Terms

- RFID is often used as generic term for complete infrastructures.
  - A transponder (aka RFID-chip, -tag, -label, wireless label or simple chip)
  - A reader (in fact most of them can write to the tag too)
  - Some middleware, which connects the reader to a server
  - Some communication infrastructure
  - Integration with server farms, data warehouses, services and supporting systems

Variants

Different types of RFID transponders

Short range	Mid range	Long range
<= 15 centimeter	<= 5meter	Up to 500 meter
ISO 14443 A+B	ISO 15693	ISO 18000-xx
13.56 MHz, 125-134.2kHz	13.56 MHz, 125-135kHz	860-956 MHz (UHF) 2.4 GHz (Microwave) 5.8 GHz (Microwave)
E-field, magnetic field	EM-field	EM-field

Transponders

- There are different kinds of transponders:
  - Only transmitting a unique ID (serial-number)
    - Only passive
    - Identification
    - Tracking (Fast-track)
    - Only clear text communication

Transponders

- There are different types of transponders:
  - Storage of Data / Metadata R/W WORM
    - Most passive, some active
    - EPC
    - Smart Labels
    - Most use clear text communication, some are with encrypted communication

Transponders

- There are different types of transponders:
  - Act as Smart Card Interface
    - Most active, some passive
    - Biometric Passport (ICAO - MRTD)
    - Access Control System (Mifare DESFire)
    - Encryption, authentication, encrypted communication

Generic Attacks

- Sniffing of the communication between transponder and reader
  - Counterfeiting of the communication
  - Obtain UID, user data and meta data
  - Basic attack on structures and tags
  - Replay attack to fool the access control systems

Generic Attacks

- Counterfeiting the identity of the reader and unauthorized writing to the tag
  - Change of UID via manipulation of the administrative block
  - Declare false identity
  - UID must be readable in clear text
  - Manipulation of product groups and prices

Generic Attacks

- Manipulation of data stored on the transponder
  - Manipulation of data
  - Manipulation of metadata
  - Swap of objects
  - Logical duplication of objects

Generic Attacks

- Deactivation of the transponder
  - Disable the traceability of objects
  - Disable the visibility of objects

Generic Attacks

- Attack the structures in the middleware and backends, manipulation of data structures.
  - Injection of malware into the backend and middleware systems
  - E.g. database worms
  - Manipulation of backend systems
  - Denial of Service attack against the infrastructure

Generic Attacks

- Jamming of the RFID frequencies
  - Use of “out-of-the-box” police jammer (broadband jamming transmitter)
  - Attack against anti-collision (RSA attack)
  - Prevent reading of the tag
  - Simple denial of service attack against the RFID System
  - Shut down production, sales or access

Encrypted RFID

- MIFARE are the most used RFID transponders featuring encryption
  - Technology is owned by Philips Austria GmbH
  - Technology is based on
    - ISO 14443
    - 13.56 MHz Frequency

MIFARE Tags

- MIFARE Standard
  - Proprietary high-level protocol
  - Philips proprietary security protocol for authentication and ciphering
  - MIFARE UltraLight: same tags without encryption

MIFARE Tags

- MIFARE Pro, ProX, and SmartMX
  - Fully comply to ISO 14443-4 standard
  - The different types of tags offer memory protected by two different keys (A and B)
  - Each sector could be protected with one of these keys.

Brute Force the Tag

- 2^6^8 bit for the keyspace
- 25 ms per try with a brute force perl script using Linux and a self written driver
- Using one RFID reader

$$\frac{6^{\left(2^{8}\right)} \bullet 0.025s}{3600s} \approx 81445305\,\mathrm{Days} \approx 22623\,Years$$

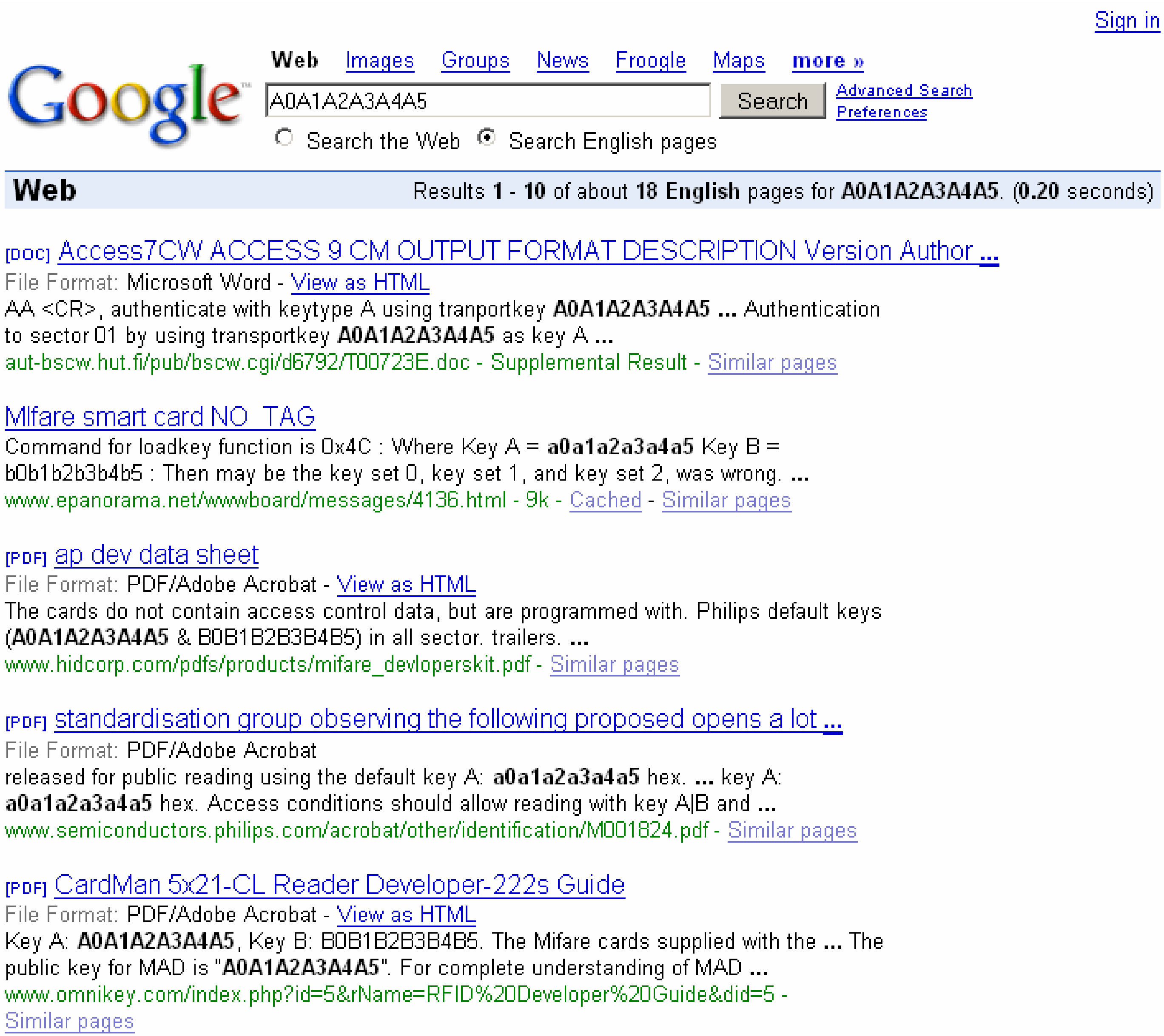
Brute Force the Tag

- 2^6^8 bit for the keyspace
- 25 ms per try with a brute force perl script using Linux and a self written driver
- Using 1.000 RFID readers

$$\frac{6^{\left(2^{8}\right)} \bullet 0.025s}{3600s \bullet 1000} \approx 81445\,\mathrm{Days} \approx 226\,Years$$

MIFARE Sector Keys

- Philips puts all information under NDA
- We are not interested to sign an NDA
- Extract information from RFID software via „UNIX strings“
- Google helps a lot, Google desktop search is very popular at smartcard developers´ PCs ;-)
- Look at the results



Results **1** - **10** of about **18** **English** pages for **A0A1A2A3A4A5**. (**0.20** seconds)

[DOC] Access7CW ACCESS 9 CM OUTPUT FORMAT DESCRIPTION Version Author ...
File Format: Microsoft Word - View as HTML
AA <CR>, authenticate with keytype A using tranportkey **A0A1A2A3A4A5** ... Authentication to sector 01 by using transportkey **A0A1A2A3A4A5** as key A ...
aut-bscw.hut.fi/pub/bscw.cgi/d6792/T00723E.doc - Supplemental Result - Similar pages

MIfare smart card NO_TAG
Command for loadkey function is 0x4C : Where Key A = **a0a1a2a3a4a5** Key B = b0b1b2b3b4b5 : Then may be the key set 0, key set 1, and key set 2, was wrong. ...
www.epanorama.net/wwwboard/messages/4136.html - 9k - Cached - Similar pages

[PDF] ap dev data sheet
File Format: PDF/Adobe Acrobat - View as HTML
The cards do not contain access control data, but are programmed with. Philips default keys (**A0A1A2A3A4A5** & B0B1B2B3B4B5) in all sector. trailers. ...
www.hidcorp.com/pdfs/products/mifare_devloperskit.pdf - Similar pages

[PDF] standardisation group observing the following proposed opens a lot ...
File Format: PDF/Adobe Acrobat
released for public reading using the default key A: **a0a1a2a3a4a5** hex. ... key A: **a0a1a2a3a4a5** hex. Access conditions should allow reading with key A|B and ...
www.semiconductors.philips.com/acrobat/other/identification/M001824.pdf - Similar pages

[PDF] CardMan 5x21-CL Reader Developer-222s Guide
File Format: PDF/Adobe Acrobat - View as HTML
Key A: **A0A1A2A3A4A5**, Key B: B0B1B2B3B4B5. The Mifare cards supplied with the ... The public key for MAD is "**A0A1A2A3A4A5**". For complete understanding of MAD ...
www.omnikey.com/index.php?id=5&rName=RFID%20Developer%20Guide&did=5 - Similar pages

Default Keys

- Found the following default keys:
  - Key A A0 A1 A2 A3 A4 A5
  - Key A FF FF FF FF FF FF
  - Key B B0 B1 B2 B3 B4 B5
  - Key B FF FF FF FF FF FF
  - About 60 keys from example applications
  - No protection 00 00 00 00 00 00

MAD

- Additional found the Mifare Application Directory, a PDF that shows how MIFARE are specifying the type of use of one of the transponders, each applications should have an entry to show the Type of Service

Example Layouts

- In the datasheets and „googled“ documentation are a lot of examples.
- These examples include different keys and tag / memory layout and data structure for:
  - Ticketing
  - Access Control
  - Online Payment

Software developers are lazy

- Checking a couple of cards shows that more than 75% use one of these default keys!
- It compiles let's ship it !
- The programmers use <u>not</u> only the example layouts, they also use the <u>example keys</u> !

Attack the Tag

- Directory attacks are possible with found default and example keys
  - Variations of the directory are always possible
- „Smart“ brute-force attack to the tag are possible
  - never seen a lockout or false login counter
  - a delay for a false key does not exist

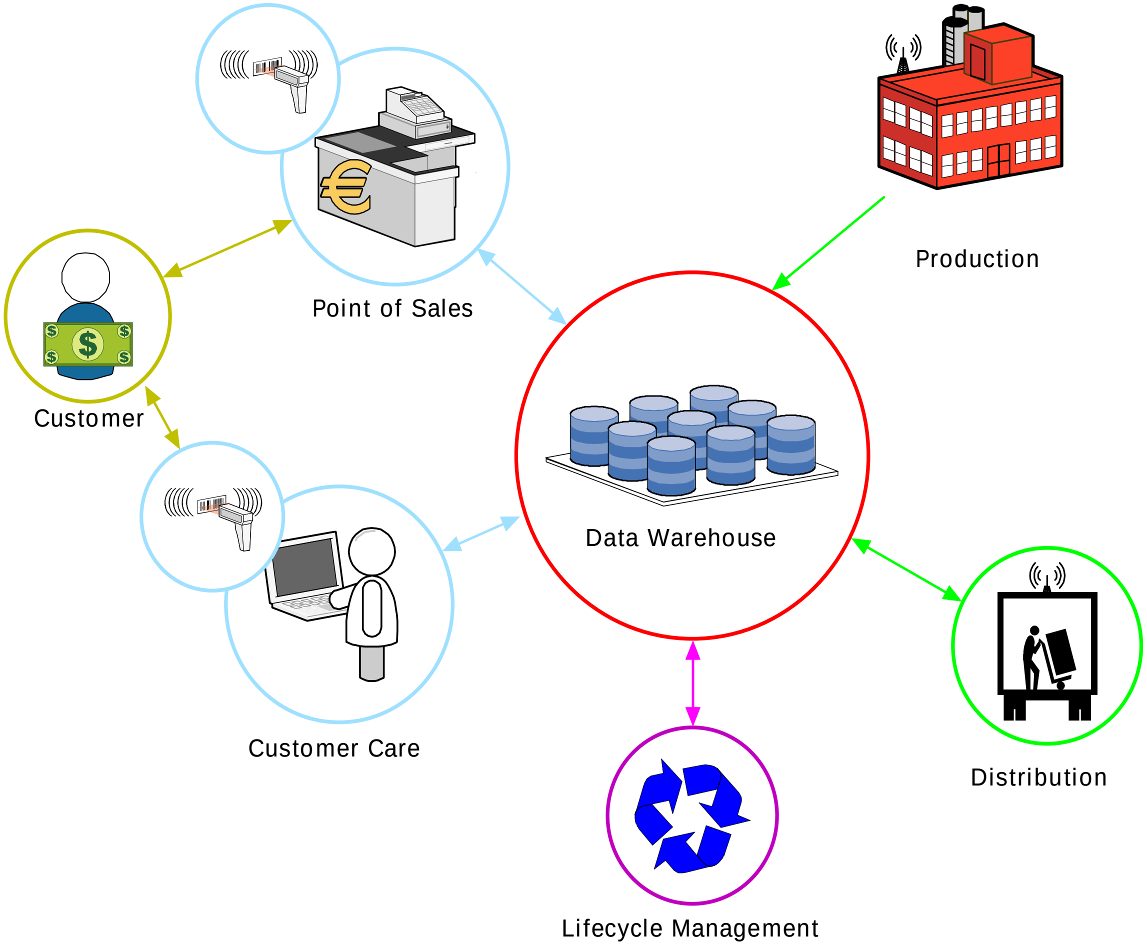
Attacks to the Backend

- The memory of a ISO 15693 tag acts like a normal storage
- RFDump (Black Hat 2004) could help to manipulate data like with a hex-editor
- SQL-Injection and other attacks are possible

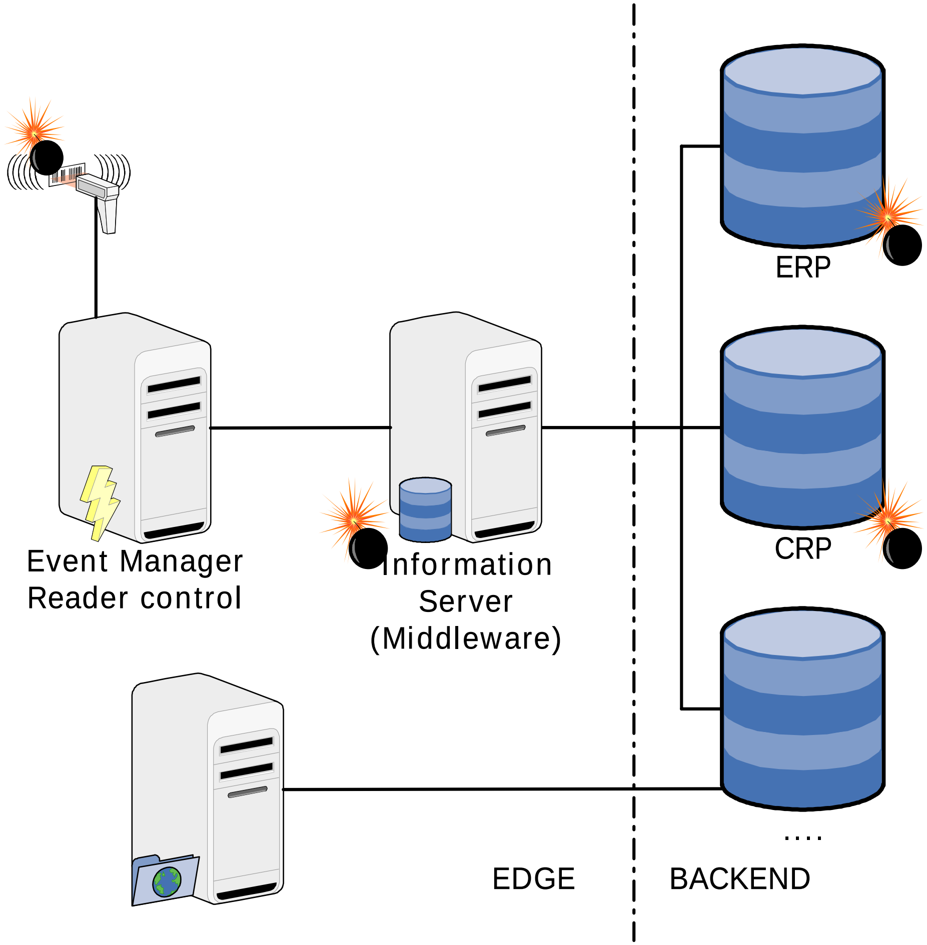
Preventing security functions

- If the tag is „read only“ read it with RFDump and write the manipulated data to an empty one
- Checksum, some implementations use the UID (Unique ID) as mirror block in the UD, both must be changed
- If the block is encrypted, the Sector Key must be broken

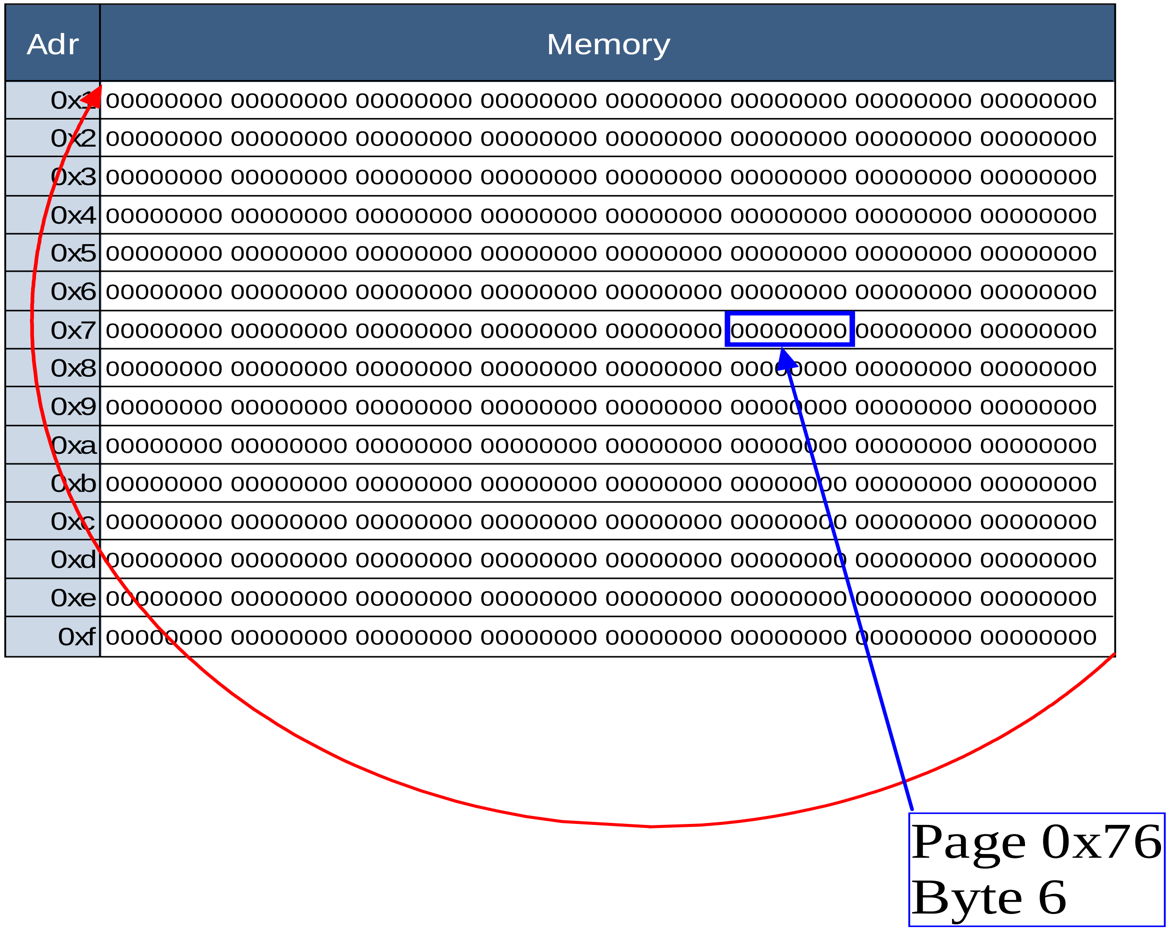
The RFID Supply chain



Break into the Systems



Problem Memory Size



Adr	Memory
0x1	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x2	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x3	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x4	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x5	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x6	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x7	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x8	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0x9	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xa	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xb	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xc	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xd	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xe	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
0xf	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Page 0x76
Byte 6

Representation to the Backend

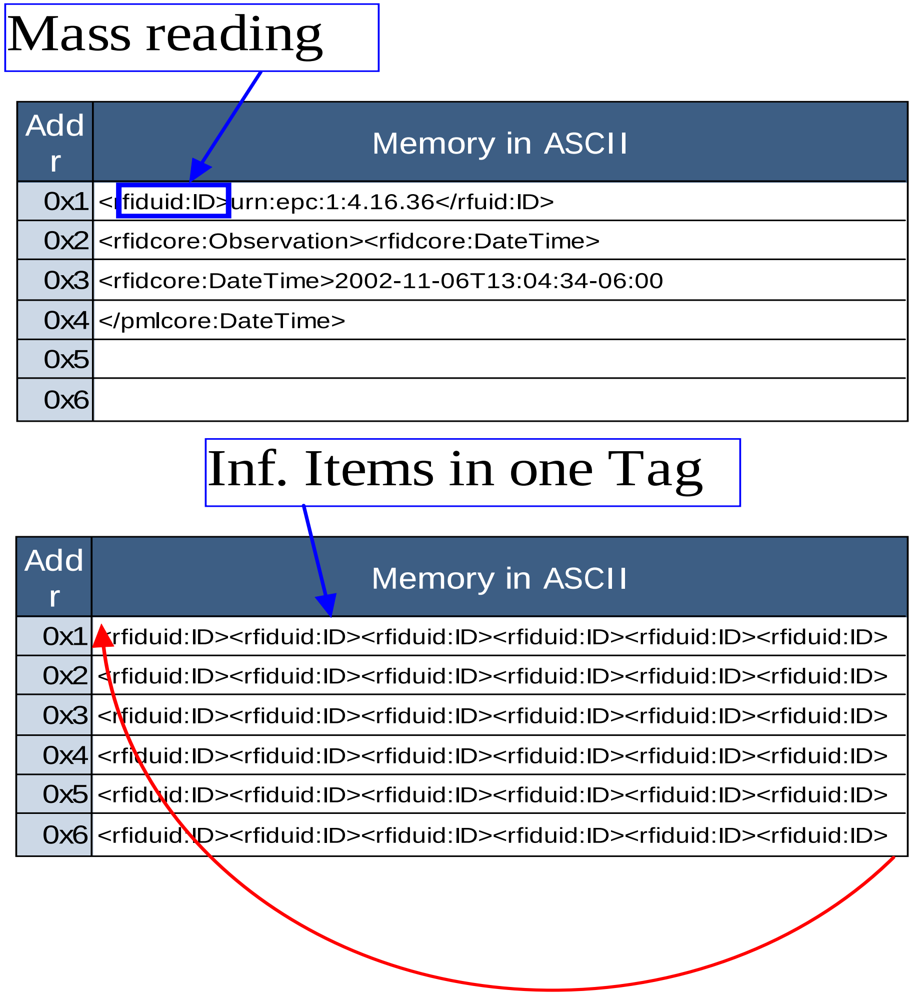
- Looks like unlimited space on the tag
  - E.g. RFDump uses a tag database to avoid reading over the boundary
- Normally reading is event-driven
  - Reading up to the EOF
  - Input is unchecked in all implementations we have seen

Tag DoS with C-Strings

End of String

Adr	Memory
0x1	68547369 69202073 6e616520 6178706d 656c6f20 20662061 616d696e 75706100
0x2	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x3	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x4	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x5	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x6	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x7	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x8	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0x9	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xa	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xb	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xc	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xd	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xe	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF
0xf	FFFFFFF FFFFFFF FFFFFFF FFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFFF

Tag DoS with XML



Mass reading

Addr	Memory in ASCII
0x1	<rfiduid:ID>urn:epc:1:4.16.36</rfuid:ID>
0x2	<rfidcore:Observation><rfidcore:DateTime>
0x3	<rfidcore:DateTime>2002-11-06T13:04:34-06:00
0x4	</pmlcore:DateTime>
0x5	
0x6	

Inf. Items in one Tag

Addr	Memory in ASCII
0x1	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>
0x2	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>
0x3	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>
0x4	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>
0x5	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>
0x6	<rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID><rfiduid:ID>

Soft-Tags

- Emulation of RFID-Tag and/or reader
- Serial-Emulation of any ISO 15693 tag
- Useful for testing backend and middleware
- Reads „backup“ from real tags
- Manipulation of any UID, User Data or administrative block.

Thank You

Questions ?