Highway to Hell: Hacking Toll Systems

Nate Lawson
Blackhat USA
2008/8/6

root labs
My background

• Root Labs founder
  – Design and analyze security components
  – Focused on:
    • Embedded and kernel security
    • Software protection
    • Crypto

• IBM/ISS
  – Original developer of RealSecure IDS

• Cryptography Research
  – Co-designed Blu-ray disc content protection layer, aka BD+
How I got interested in toll passes

• I have never used FasTrak
  – Privacy concerns
    • Bridge
    • Freeways
  – Pay cash or take public transit

• How does it work?
  – Almost no analysis available online
  – Title 21 (protocol) is a standard though

• What’s really inside?
  – Buy transponder from Safeway without signing up ($26 cash)

• Perhaps privacy issues can be fixed?
What is electronic toll collection?

• Automatic debit of an account for use of a bridge or toll road
• Many possible implementations
  – RFID transponder
  – Image recognition of license plate
• Current systems
  – E-ZPass (East Coast)
  – TollTag, Sunpass, etc.
  – FasTrak (Bay Area + Southern California)
Electronic toll collection

Readers

Windshield Transponder

“Lane pwned”

Screen grab courtesy of KTVU News
Tracking and privacy

• Few realize all freeway travel is also tracked
  – Transponders are queried by readers on signs to generate realtime traffic statistics (511.org)
  – Separate agency (and thus servers) from toll collection, but same transponder
Tracking and privacy

- Toll transactions are logged
  - Indefinitely? No info in privacy policy
- Freeway travel is separately logged by 511.org
  - The transponder ID is “anonymous” and “only stored for 24 hours” (KTVU news report)
  - “Users remain anonymous through … encryption software that scrambles each FasTrak toll tag ID” (privacy policy)
- Lawyers know this info is available
  - “FasTrak gets about one subpoena a month for toll records.” (KTVU news report)
  - Wouldn’t they stop bothering if this info wasn’t useful?
Adding anonymity afterwards difficult

• Conventional approach (adding anonymity)
  – ID → Hash(ID)
    • Not anonymous, just exchanges one ID for another
    • Subject to correlation attacks
    • Ignores meta-information (timing, length, date)

• AOL anonymized search term scandal (2006)
  – Real names and addresses could be recovered by correlating info across multiple searches
Robust anonymity has to be built-in

- **Reduce collection**
  - Query based on a random timer, not all cars
  - Only one 1 out of 100 cars necessary to get average speed

- **Limit distribution**
  - Calculate speed and throw away original IDs after two readings
  - Limit the number of systems that touch it along the way

- **Expire aggressively**
  - Only statistic needed is sign-to-sign interval
  - Discard IDs after a few minutes

- **Cryptography**
  - “Untraceable RFID Tags via Insubvertible Encryption” (Ateniese, Camenisch, and de Medeiros)
  - "A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags" (Molnar, Soppera, and Wagner)
Title 21 system history

• California legislature passed a technical law
  – Title 21, Chapter 16 (1992)
  – Developed mostly by Texas Instruments

• FasTrak
  – All Bay Area bridges (BATA)
  – Orange County toll highways
  – Airport parking lots

• Over one million transponders purchased
Title 21 standard

• Layer 1: modulation and frequency
  – Reader downlink
  – Transponder uplink

• Layer 2: packet framing
  – Start sequence, checksum

• Layer 3: packet types
  – Poll messages
  – Responses

• Layer 7: allocation of IDs among agencies
Layer 1: modulation and frequency

- **Downlink from reader**
  - ~900 MHz carrier frequency
  - Square-wave AM
    - Unipolar ASK of the carrier using Manchester encoding
    - “1”: signal during first half, “0”: signal during second
- **Uplink from transponder**
  - Backscatter of carrier via antenna polarization
  - Dual-frequency AM
    - FSK of 1200 KHz/600 KHz (“1” and “0”, respectively)
- **300 Kbps data rate (both directions)**
Layer 1: reader modulation

- Downlink from reader

- 300 Kbps data rate
- 600 KHz square wave (ASK)
  - “1” = high in first half of period
  - “0” = high in second half of period
Layer 1: transponder modulation

- Uplink from transponder

- 300 Kbps data rate
- 1200/600 KHz square wave (FSK)
  - “1” = higher frequency
  - “0” = lower frequency
Layer 2: packet framing

- Wakeup signal before message
  - 33 µs burst of 1-bits
  - 100 µs no signal
- Packet start: 0xAAC
- Ends with 16-bit CRC
  - Standard says “CRC-CCITT”
  - Spec bug: initial value is 0, not 0xFFFF like CCITT says
    - If you actually implemented the Title 21 spec, you’d be incompatible
Layer 3: standard messages

Reader

Polling Message Type 1 (8000)
- Requests the transponder to send its ID
- Agency code, 16-bit

Data Message Type 1 (0001)
- Transponder ID, 32-bit

Transponder

Acknowledge Message Type 2 (C000)
- Confirms reception of the transponder ID
- Transponder ID, 32-bit
- Reader ID, 32-bit
- Status, 16-bit
Enrollment process
Enrollment process

• Validation code is just toll tag serial number in hex
• Used as a checksum for typos

Toll Tag Number*: (This is the 10-digit number located in the upper left corner of your toll tag.)

Validation Code*: (This is the 8-digit alpha-numeric code located on the velcro strip.)
Diving into the transponder

“If a Toll Tag fails to operate for reasons other than abuse… we will replace it…”
Diving into the transponder
Transponder operation

- Receive side
  - Signal is received and amplified (analog)
  - Demodulated and presented to pin 2.5 as a square wave
- Transmit side
  - Carrier reflected back by swapping pins 1.6 and 1.7 quickly
- Buzzer
  - Timer interrupt + XOR (pins 2.0 and 2.1)

Thanks go to Adam O’Donnell for the RF help
MSP430 basics

• Low-power 16-bit microcontroller
  – 2, 4, and 6-byte instructions
    • Kinda strange: MOV @R14+, R15
  – Von Neumann address space (shared code/data)
    • Helpful for stack/integer overflows (Travis Goodspeed)
  – Self-programmable flash memory
    • Persist that exploit

• MSP430F1111A
  – Peripherals: timer, comparator, ports (address 0)
  – 128 bytes RAM (0x200)
  – 256 bytes data flash (0x1000)
  – 2 KB code flash (0xF800)
# FasTrak MSP430 memory map

<table>
<thead>
<tr>
<th>Memory Type</th>
<th>Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt vectors</td>
<td>FFE0 - FFFF</td>
</tr>
<tr>
<td>Code flash</td>
<td>F800 - FFFF</td>
</tr>
<tr>
<td>Data flash</td>
<td>1000 - 10FF</td>
</tr>
<tr>
<td>Boot ROM</td>
<td>0C00 - 0FFF</td>
</tr>
<tr>
<td>RAM</td>
<td>0200 - 027F</td>
</tr>
<tr>
<td>Peripherals</td>
<td>0000 - 01FF</td>
</tr>
</tbody>
</table>
Dumping the firmware

• Internal firmware is protected by JTAG fuse
  – Normal programming method is via JTAG

• Bootstrap loader (BSL)
  – 256-bit password allows access to flash
    • Probably checked with `memcmp()`
    • Go see Travis Goodspeed’s talk on timing attacks in the BSL

• Rule 1: always try the front door
Old transponders are not locked

• JTAG fuse is not set
  – Plug in microcontroller and read flash memory
  – Verified on a transponder from Southern California

• Newer transponders are locked
  – Need more magic to verify their contents
Bypassing the JTAG fuse

• Silicon magic courtesy of Chris Tarnovsky
  – Depackage chip
  – [Fuse magic happens here]
  – Rebond to DIP package
  – Read out flash
• Code was identical to unlocked transponder

If you make silicon, Fly Logic does amazing analysis work.
http://flylogic.net/
Monitoring transponder IO

- Add header and socket for DIP CPU
Monitoring transponder IO

- FPGA tap board and socket on breadboard
What’s inside?

• Load code with IDA MSP430 plugin
  – Full reply messages with checksum laid out in order
  – Main loop: switch (packetLen); dispatch handler
  – Timer interrupts, comparator trigger

• Build a modified msp430simu
  – Cycle-accurate simulator in python
  – Breakpoint/log support routines
    • Checksum
    • Memcpy
    • Receive (poll) for packet
    • Transmit packet
    • Beep
Reader request messages

- **Standard**
  - Request for ID (8000, 8 bytes)

- **Reserved by spec but not supported by firmware**
  - Encrypted ID request (80xx, 11 bytes)
    - Agency code (16 bits)
    - Proprietary TI encryption key (24 bits)
  - Encrypted unknown message (88xx, 13 bytes)
    - Transponder ID (32 bits)
    - Proprietary TI encryption key (24 bits)

Lengths include “AAC” header, rounded up to nearest byte.
Reader request messages

- Supported by firmware but not specified
  - 11-byte requests
    - 00DE, 01DE, 02DE, 03DE, 0480, 04DE
  - 36-byte requests
    - 01DF, 05DF
  - 37-byte requests
    - 05DE
Transponder reply messages

- **Standard**
  - ID response (0001, 10 bytes)

- **Reserved and supported by firmware**
  - ID and serial response (0007, 22 bytes)
    - “Block A data” (128 bits) which is actually:
      - Unknown (16 bits)
      - Transponder ID (32 bits)
      - Unknown (16 bits)
      - Transponder serial number (BCD, 48 bits)
      - Padding (08FF)

- **Reserved by spec but unsupported**
  - “Block A and B, C, or D data” (000x, 38 bytes)
Transponder reply messages

- Supported by firmware but not specified
  - Misc ID+serial messages
    - 0002, 38 bytes
      - 16 bytes empty
    - 0005, 38 bytes
      - Bits checked when processing other msgs
  - Empty messages (for future?)
    - 5F07, 30 bytes
    - 0003, 38 bytes
    - 0004, 38 bytes
    - 5F06, 38 bytes
Reader response messages

• Standard
  – Status reply (C000, 16 bytes)
    • Transponder ID (32 bits)
    • Reader ID (32 bits)
    • Status code (16 bits)

• Reserved but not supported by firmware
  – Unknown response 1 (C00x, 20 bytes)
  – Unknown response 2 (C00x, 36 bytes)
    • All the above + 128 bits “data”
What’s not inside?
CRYPTO.
Cloning attacks

• Passive cloning
  – Set up a receiver near a freeway
  – Record IDs as they are transmitted to reader

• Active cloning
  – Drive past parking lots, shopping centers, etc.
  – Use portable reader to scan and log IDs of parked cars

• Missing cryptographic property: replay resistance
  – Reader proves itself to transponder
  – Transponder proves itself to reader
Monetizing cloning attacks

• Create a subscription service
  – Users get customized transponders or hack existing ones
  – Device downloads new IDs from PC over the air
  – Each ID is used only once, preventing pattern analysis

• Low risk
  – Failure to read transponder = $29 fine
  – Service can pay penalty for subscribers

• Potential customers
  – Trucking companies
  – Drug couriers
Digging still deeper...

• Does FasTrak write data to your transponder?
  – “FasTrak is a read-only device. There's no memory to write anything to.” (John Goodwin, BATA)
  – Best interpretation: “We only use it in a read-only manner and are not aware our vendor used a flash device”

• But there is memory and it is writeable
  – MSP430F1111A is flash-based, only the BSL is in ROM
  – Supports in-system erase/rewrite
Firmware has ID update routine

- Flash write subroutine is present in firmware

```assembly
mov   #0A550h, &FCTL2
mov   #0A500h, &FCTL3
mov.b @R14+, 0(R12)
```

- Called from multiple places in packet processing function
- Appears to be used to update the IDs of various message responses stored at 0x1000
IDs can be wiped/overwritten from remote

• Flash update can be triggered with a couple messages
  – Packet 1: prepare to flash
  – Packet 2: data to write

• Update routine
  – Calculates checksum of data from packet
  – Writes it to various locations of IDs within pre-computed response messages stored in flash

• Caveat: update routine only tested in simulator so far
Alibi attack

• Establish presence elsewhere during crime
  – Read and save neighbor’s FasTrak ID from parked car
  – Send message to update his transponder with your ID
  – He goes to work at 9 am, you commit crime
  – Subpoena records: you were on the bridge at 9 am!

• Questions
  – Is FasTrak data really considered so indisputable?
  – Will this alibi hold up in court?
Contacting the vendors

• FasTrak is:
  – A technical standard ratified as law by the California legislature
  – Administered by CalTrans
  – Locally run by Metropolitan Transportation Commission (MTC), Bay Area Toll Authority (BATA), Orange County Transportation Corridor Agencies (TCA), etc.
  – With devices potentially provided by multiple vendors, but in practice, mostly Sirit

• No response after …
  – Email contact form on bayareafastrak.org
  – Sending business card with reporter who later talked to BATA
  – Speaking to CalTrans consultant
Conclusions

- Electronic toll collection needs improvement
  - Excessive loss of privacy in current usage
    - Please fix this before we move to license plate recognition
  - Clonable if no encryption
  - Untrustworthy for legal evidence
    - Transponder IDs can be overwritten over-the-air
- Found many surprises when opening the box, even with an established system
  - I’m happy to explain the details for free to any FasTrak authorities who contact me

Contact: nate@rootlabs.com     Info/blog: rootlabs.com