This Ain’t Your Dose: Sensor Spoofing Attack on Medical Infusion Pump

Youngseok Park¹,², Yunmok Son², Hocheol Shin², Dohyun Kim², and Yongdae Kim²

¹ NAVER Labs
² System Security Laboratory, KAIST

10th USENIX Workshop on Offensive Technologies (WOOT ’16)
Aug.09.2016
Sensor

- Sensing changes in physical property and converting to electric signal
- Gyroscope, Accelerometer, Radar, Sonar, Infrared sensor, etc.
Sensing and Actuation System

Real World

Sensor
- Radar
- Gyroscope

System
- ADC
- Processor

Actuator
- Crash avoidance
- Flight control

ADC: Analog-to-Digital Converter
Sensing and Actuation System

Spoofing!

Real World

System

Sensor

Actuator

No Authentication

Vulnerable to sensor spoofing attack

ADC: Analog-to-Digital Converter
Sensor Spoofing Attack

- Manipulating sensors with a malicious signal
- Previous works
  - Attacking Circuit using EMI: Injecting EMI into a wire of a defibrillator (S&P’13)
  - Canceling and injecting Active Sensor Signal: magnetic signal on ABS sensor (CHES’13)
  - Generating Resonance (DoS): Injecting sound noise into a gyroscope of a drone (SEC’15)

EMI: Electromagnetic Interference
ABS: Anti-lock Braking System
This Work: Manipulating Sensing Values by Saturating Receiver
Target: Medical Infusion Pump

- Controlling infused volume of medicine to patients
- Sometimes using a drop sensor for accuracy
Infusion Pump Operation

Medicine

Drop sensor

IV Tube

To human's body

Actuator (Peristaltic Fingers)

Light Intensity

time
Sensor Saturation

- New type of sensor spoofing attack using **saturation**
  - Sensors have typical operating region
  - Output is saturated when exceeding a saturation point
  - Blinding sensors

![Diagram of sensor saturation and spoofing](image)

In case of the infusion pump:

- Saturated signal
- Drops under spoofing
- Normal operation
- Time

Operating region (Linear) vs. Saturation region (Nonlinear)
Medical Infusion Pump

- Two infusion pumps with drop sensors

<table>
<thead>
<tr>
<th>Infusion pump</th>
<th>Drop sensor</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSB-1200</td>
<td><img src="image1" alt="Drop sensor" /></td>
</tr>
<tr>
<td>(Pump1)</td>
<td><img src="image2" alt="Infusion pump" /></td>
</tr>
<tr>
<td></td>
<td><img src="image3" alt="Drop sensor" /></td>
</tr>
<tr>
<td></td>
<td><img src="image4" alt="Infusion pump" /></td>
</tr>
<tr>
<td>BYS-820</td>
<td><img src="image5" alt="Drop sensor" /></td>
</tr>
<tr>
<td>(Pump2)</td>
<td><img src="image6" alt="Infusion pump" /></td>
</tr>
</tbody>
</table>
Hardware Analysis

- **Pump1 (JSB-1200)**

- Tube
- LED
- Peristaltic fingers
- Infusion pump
- Drop sensor
- IR receiver
- IR emitter
- IR Filter
Measuring signal with oscilloscope
- Connector = 4 pins: VCC, GND, LED, and IN (signal)
Simple Test (Saturation, w/o filter)
Simple Test (Saturation, w/o filter)
Hardware Analysis

- Mainboard (2 MCUs)

- W78E516D (MCU2)
- AT89S52 (MCU1)

Internal structure

- SPI Port
- Drop sensor port
- SPI Port
- AT89S52 (MCU1)
- W78E516D (MCU2)
Hardware Analysis

- Sensor output is inserted to MCU1 after ADC
  - 8-bit ADC (0 to 255)
  - Digital signal indicates voltage level of the drop sensor
Firmware Extraction

- Extracting firmware of MCU1 via SPI port
- Reading Flash memory using USBISP and AVR Studio
- Data section -> 8051 assembly -> IDA Pro
Finding sensor output in **Timer interrupt function**

```assembly
; Code 432
nop
nop
mov RAM 41, R3
lcall code_3E5
mov RAH_30, RAH_42
lcall code_202
jnb RAM 20.3, code_451

; Code 3E1
lcall code_3E2
clr C
ori P1.4
mov A, RAH_42
rlc A
mov RAM 42, A
movz B, code_3E1

; Port 1
setb P1.5
ret

; End of function code_3E5
```

**Put 8-bit sensor output to RAM**
Drop Detection Algorithm

Algorithm 1: Simplified algorithm of sensing drops

Input: Input
Output: Sensing drop, Alarm
Data: Maximum, Minimum, Average

1 Loop

2 Save sensor value to Input:

3 If Input < Average - 15 then

4 // Voltage drop by 0.32V
5 Sensing drop. // A drop is sensed
6 Send command through serial port
7 return;
8 end

9 If Input < Minimum then
10 Minimum = Input;
11 end

12 If Input > Maximum then
13 Maximum = Input;
14 end

15 If Maximum - Minimum >= 5 then

16 // Correct value after sensing drop
17 Maximum = Input;
18 Minimum = Input;
19 return;
20 else

21 Average = (Maximum + Minimum) / 2
22 if Average < 8 then
23 // Average is lower than 0.16V
24 Alarm; // Error occurred
25 end
26 return;
27 end

28 End Loop
Pump1 Structure

1. Drop sensor output enters into AT89S52 (MCU1)
2. MCU1 sends data to W78E516D (MCU2) via serial comm.
3. MCU2 actuates peripherals with this data
   - Pins of MCU2 are directly connected to motor, display and alarm
Vulnerability

- **Drop sensor**
  - Saturated with an external source
  - Cannot sense drops in saturation

- **Drop detection algorithm**
  - Counting drops based on a relative change in voltage
  - Making a voltage drop to sensor output
Experimental Setting

- IR Laser (905nm, 30mW)
- Drop sensor
- Arduino
- Measuring cylinder
- Infusion pump
Experiment

- Performed on both infusion pumps (Pump1, Pump2)
- **Saturation (failed in Pump2)**
  - Sensor is saturated when injecting IR laser to receiver
  - Drop sensor cannot sense real drops -> **Over-infusion**
- **Fake drops**
  - Sensor is deceived by fake drops with external IR
  - Pump perceives that there are drops already -> **Under-infusion**
- Both cases cause an **alarm**
Spoofing Pattern

- **Over-infusion**
  - Alarm: "**No drop is detected**"
  - Inject some period and compensate insufficient drops

- **Under-infusion**
  - Alarm: "**Too many drops are detected**"
  - Find properly interval of fake drops experimentally

- **Example (60mL/h setting)**
  - 1 drop per 3 seconds

![Diagram showing normal operation, continuous saturation, over-infusion, under-infusion, real drop interval, fake drop interval, saturation time, and alarm.]

Real drop interval (3s) ✕ fake drop interval  ✕ drop  ✕ saturation time (13s)  ✕ alarm  ✕ 2s
Demo (Over-infusion)
Demo (Under-infusion)
Spoofing Pattern

- Over-infusion
  - Alarm: “No drop is detected”
  - Inject some period and compensate insufficient drops

- Under-infusion
  - Alarm: “Too many drops are detected”
  - Find properly interval of fake drops experimentally
Results

- Controlling infused volume is possible
  - By adjusting saturation time or fake drops
  - Measured in 10 minutes and 5 times each (No alarm rings over 30 minutes)
  - Over-infusion fails on Pump2
Discussion

- **Attack distance**
  - Related to power of source
  - Possible in the range of 12m with 30mW IR laser

- **Mitigation**
  - Authentication between emitter and receiver
    - PyCRA (CCS ‘15)
    - Generate random zero signal in an emitter
  - Voltage level detection
    - Checking boundary of legitimate signal
  - Physical isolation

![Diagram of PyCRA concept](image)

![Voltage level detection](image)
Discussion

- Attack distance
  - Related to power of source
  - Possible in the range of 12m with 30mW IR laser

- Mitigation
  - Authentication between emitter and receiver
    - PyCRA (CCS ’15)
    - Generate random zero signal in an emitter
  - Voltage level detection
    - Checking boundary of legitimate signal
  - Physical isolation
Conclusion

- Presenting a new type of sensor spoofing attack
  - Deceiving a sensor by saturation
- Analysis on medical infusion pumps
  - Finding vulnerability in drop detection algorithm
- Controlling infused fluid from 65% to 330%

- Note
  - Infusion pump was not communicating at all.
  - IR lay is invisible to human eyes.
  - FDA approved US devices?

- Sensor security
  - Most sensors are exposed to receive signal
  - Must be considered for safety
Thank You!

E-mail: ys.park@navercorp.com