I. INTRODUCTION

Tire Pressure Monitoring Systems, to the best of our knowledge, represent the first in-car wireless network mandated for every new automobile sold or manufactured after 2008 in the United States [3]. A typical TPMS contains a few TPM sensors (mounted inside tires) periodically broadcasting tire pressure, and a TPM electric control unit (ECU) alerting drivers if the received measurement indicates significantly underinflated situation. Although TPMSs promise to increase overall road safety, the security and privacy implications of such in-car wireless sensor networks are not fully understood — in particular, it is unclear whether the metal car body sufficiently shields these low-power transceivers from eavesdropping or spoofing outside the vehicle.

To address these questions, this paper presents a privacy and security evaluation of two tire pressure monitoring systems using both laboratory experiments with isolated tire pressure sensor modules and experiments with a complete vehicle system. We found that the sensor messages can be sniffed and decoded up to 40m from a passing vehicle with a basic low-noise amplifier and the openly available GNU radio platform. Furthermore, current protocols do not employ authentication mechanisms and vehicle implementation do not appear to perform basic input validation or filtering of messages. This allows straightforward spoofing of sensor messages. Finally, we propose a set of recommendations and a key management protocol that could significantly improve privacy and security of TPMS and other forthcoming in-car wireless sensor networks.

II. REVERSE-ENGINEERING TPMS PROTOCOLS

As the first step to analyzing security and privacy risks, a thorough comprehension of the protocols for specific sensor systems is necessary. However, the specifics of the TPMS communication protocols are proprietary and we have to discover the details of the protocols via reverse-engineer efforts.

To assist reverse-engineering, we used the following hardware: (1) two representative tire pressure sensors (namely TPS-A and TPS-B) that are used in automobiles with high market share in the US; (2) a TPMS trigger tool [2], which is a handheld device used by car technicians for troubleshooting and which can activate and decode information from a variety of tire sensors; and (3) a GNU Radio in conjunction with the Universal Software Radio Peripheral (USRP) [1].

We began by collecting a few transmission bursts from each TPS-A or TPS-B sensor and performed signal analysis in Matlab to understand the modulation and encoding schemes. We found that amplitude shift keying (ASK) and frequency shift keying (FSK) are used for each type of sensors and both sensors were utilizing Manchester encoding. To understand the size and meaning of each bit field, we manipulated sensor transmissions by varying a single parameter and observed which bits changed in the message. Using this approach, we managed to determine the majority of message fields and their meanings for both TPS-A and TPS-B.

The results of reverse engineering reveal that encryption has not been used and each message contains a 28-bit or 32-bit sensor ID that does not change throughout the sensors’ lifetimes. This raises location privacy risks because vehicles could potentially be tracked through these identifiers and drivers do not have any option to disable the system. This also raises message spoofing risks because forged data can be injected into the vehicle ECU.

III. EAVESDROPPING VEHICULAR COMMUNICATION

A critical question for evaluating privacy implications of in-car wireless networks is whether the transmissions can be easily overheard from outside the vehicle body. Thus, we experimentally evaluate the range of TPMS communications.

To constantly monitor the channel and only record useful data using GNU radio together with the USRP, we created a live decoder/eavesdropper leveraging pipes. We used the GNU Radio standard python script usrp_rx_cfile.py to sample channels at a rate of 250 kHz, where the recorded data was then piped to a packet detector. Once the packet detector identifies high energy in the channel, it extracts the complete packet and passes the corresponding data to the decoder to extract the pressure, temperature, and the sensor ID.

We studied the reception range of stand-alone sensors in a hallway and observed that eavesdropping ranges are roughly
10.7 m. We notice that we were able to decode the packets when the received signal strength is larger than the ambient noise floor as depicted in Figure 1. Furthermore, we inserted a low noise amplifier (LNA) between the antenna and the USRP radio front end, and effectively improved the decoding range from 10.7 meters to 40 meters. This shows that with some inexpensive hardware a significant eavesdropping range can be achieved, a range that allows signals to be easily observed from the roadside.

In the outdoor experiment, we investigated the reception range of the sensors mounted in the front left wheel of one of the authors’ cars. We placed the USRP antenna at the origin (0, 0) and drove the car along trajectories parallel to the x-axis. These trajectories were 1.5 meters apart. Along each trajectory, we recorded the received signal strength (RSS) at the locations from where USRP could decode packets. The colored region in Figure 2, therefore, denotes the eavesdropping range, and the widest horizontal range is 9.1 meters at the parallel trajectory 3 meters away from the x-axis.

### IV. PACKET SPOOFING ATTACK

Our packet spoofing system is based on the ability to successfully eavesdrop on vehicular communication. This system takes custom input that includes the following sensor parameters: *sensor ID, temperature, pressure,* and *status flags,* and generates a properly formulated message. Since we found no USRP daughterboard available for purchase (at the time of this work) that can transmit in the 315MHz frequency band, we used a frequency mixing approach. To implement this, we leveraged two XCVR2450 daughterboards and a frequency mixer (mini-circuits ZLW11H). For 315 MHz, we used a tone at 5.0 GHz and the spoofed packet at 5.315 GHz.

We used this setup to send a spoofed packet of low tire pressure to a car with TPS-A sensors and successfully turned on the low-pressure warning light as shown in Figure 3. We also sent various forged packets and made the following observations that suggest the lack of input filtering or validation — (1) The vehicle ECU ignores packets with a sensor ID that does not match one of the known IDs of its tires, but appears to accept all other packets. (2) Neither the increased rate of 40 packets per second, nor the occasional different reports by the real tire pressure sensor seemed to raise any suspicion in the ECU or any alert that something is wrong. (3) The illumination of the low-pressure warning light depends only on the alert bit — the light turns on even if the rest of the message reports a normal tire pressure of 32 PSI.

Similar to the decoding range shown previously for eavesdropping, we did a range study of the packet spoofing attack. The maximum successful range was 16.8 meters, using simple low-cost antennas and radio devices. We deployed the attacks against willing participants on interstate highway I-26, and we observed that the attacker was able to trigger the low-pressure warning light on the victim’s car when traveling at 55 km/h and 110 km/h, respectively.

### Protecting TPMS Systems from Attacks

Some of the problems that we identified arise from poor system design, while other issues are tied to the lack of proper cryptographic mechanisms to secure the communications. Therefore, first, the TPMS software should follow basic reliable software design practices. For instance, the control unit could have employed some detection mechanism to, at least, raise an alarm when the incoming packet rate exceeds the expected one. Second, we must also maintain the keys shared between the TPMS control unit and the TPMS sensors to secure their communication. Key management schemes for TPMS should be efficient, and it should not be initiated easily in a public location.

More extensive experiment and the detail solutions have been examined and are not presented due to the space limitation.

**REFERENCES**


Security and Privacy Vulnerabilities of Automotive Tire Pressure Monitoring Systems (TPMS)

Ishtiaq Rouf\textsuperscript{*}, Rob Miller\textsuperscript{*}, Hossen Mustafa\textsuperscript{*}, Travis Taylor\textsuperscript{*}, Sangho Oh\textsuperscript{*}, Wenyuan Xu\textsuperscript{*}, Marco Gruteser\textsuperscript{†}, Wade Trappe\textsuperscript{†}, Ivan Seskar\textsuperscript{†}

\textsuperscript{*} University of South Carolina – Arena for Research on Emerging Networks and Applications (ARENA)  
\textsuperscript{†} Rutgers University – Wireless Information Network Laboratory (WINLAB)

### Security and Privacy Vulnerabilities of Automotive Tire Pressure Monitoring Systems (TPMS)

#### Background

- As of 2008, Tire Pressure Monitoring Systems (TPMS) are required in all new vehicles.
- Regulations mandate tire pressure reports at least once every 60 to 90 seconds.
- The majority of systems utilize unique identifiers to report measurements such as tire pressure and temperature.
- Protocols are proprietary and vary with manufacturer.

#### Reverse-Engineering Methodology

- We captured and analyzed packet transmission to reveal physical layer specifics.
- We discovered the demodulation and decoding schemes from the captured data to reveal the packet format.

#### Physical Layer Specifics

- LF waveform is used to stimulate the sensors.
  - Transmitted at 125 kHz.
  - CW or simple modulated waveform.
- Sensor measurements are reported at 315/433 MHz.
  - \textit{Cleartext} transmission of packets.
  - ASK/FSK modulation.
  - \textit{Differential Manchester} encoding.
- Each packet includes a unique 32-bit ID, which is enough to track all cars in the US.

#### Vulnerabilities

- **PRIVACY**: Adversary may discover the victim's location and use such information for profiling.
- **TRACKING**: Adversary may launch individual tracking and hostile targeting in real-time using timestamp.
- **SPOOFING**: Adversary may send spoofed packets with arbitrary information, disassociate a sensor, or drain battery.

#### Eavesdropping

- **Eavesdropping Range**
  - Sensor ID can be extracted if received signal strength is higher than ambient noise.
  - We boosted the range to 40 meters by using a low-noise amplifier.

- The body of the car affects the range, i.e., longer range is observed for approaching vehicles.
- One receiver every 9 meters can track all vehicles.

#### Equipments

- Universal Software Radio Peripheral (USRP) with GNU Radio.
- RFX-1800 or XCVR-2450 Daughterboard (2).
- Low-noise amplifier, frequency mixer, vector signal analyzer.
- TPM sensors, commercial TPMS reader.

#### Signal Spoofing

- **Commercial TPMS Reader Attack**
  - We transmitted arbitrary data with custom sensor ID to successfully attack a commercial TPMS reader.

- **Vehicular Attack**
  - We transmitted spoofed packets with arbitrary information to trigger the in-car alert system.
  - Our attack was validated by illumination of \textit{warning lights} and \textit{audible beep}.
  - Maximum range was 16.8 meters.
  - Successful at speed over 70 mph.

#### Recommendations

- Re-design protocol to prevent external activation when the vehicle is in motion.
- Introduce \textit{authentication} session and basic \textit{encryption} mechanisms.
- Improve message format by including \textit{sequence number}, \textit{timestamp}, and \textit{cryptographic checksum}.

---

**Arena for Research on Emerging Networks and Applications**

\textsuperscript{†} Rutgers University – Wireless Information Network Laboratory (WINLAB)