Abstract

Direct measurement is the most accurate way to measure tire pressure. Short range radio is used to transmit the tire pressure data back to automobile’s on-board computer. However, malicious user can send invalid data that can lead to inaccurate readings. Thus the radio link should be secured. Furthermore, authenticity in communication is required so car A will not accept the tire pressure readings of car B parking or driving next to it. In this work, I will explore the security mechanisms necessary in the context of low power and low duty cycle hardware. The scheme should also be able to handle multi-hop communication since radio signals from tire sensors at far rear of a truck-trailer might not reach the car computer in one hop. Evaluation will include the power/security trade-off, both in terms of additional circuit and/or additional overhead in communicating packet size.

1. Introduction

On July 26, 2001, National Highway Traffic Safety Administration (NHTSA) issued a Notice of Proposed Rulemaking (Docket No. NHTSA-2000-8572[1]) that would require most automobiles weighing less than 10,000 pounds to be equipped with a tire pressure monitoring system (TPMS). The deadline for complying with the ruling in specified vehicles is November 1, 2006. There are two ways to measure tire pressure: an indirect method that infers pressure from wheel rotation speed using existing ABS sensors and a direct method that requires implanting sensing modules in the tire itself. Direct measurement is the most accurate way to measure tire pressure. It can determine the pressure of individual tire and also compensate for temperature. It requires short range radio to transmit the tire pressure data back to automobile’s on-board receiver and computer. However, malicious user can send invalid data that can lead to inaccurate readings. Thus secure and authentic communication is required so that car A will not accept the tire pressure readings of car B parking or driving next to it.

This mandate will require almost every car on the road to be equipped with TPMS. Thus it will provide a widespread and practical application of sensor network. There has been numerous work done on secure sensor networks [2, 3, 4, 5]. There are also some exploration of energy-efficient secure sensor network [7, 9]. TPMS has more relaxed security constraints than normal sensor network since (1) sensor nodes does not dynamically enter and leave the network once they are initialized into the system, (2) (at least in initial incarnation) the communication is uni-directional from sensor nodes to base station with finite (and possibly known number of intermediate hops) and (3) there is not peer-to-peer communication aside from routing packets from downstream nodes to base station.

In this work, I will explore the security mechanisms necessary in the context of low power and low duty cycle hardware. The scheme should also be able to handle multi-hop communication since radio signals from tire sensors at far rear of a truck-trailer might not reach the car computer in one hop. Evaluation will include the power/security trade-off, both in terms of additional circuit and/or additional overhead in communicating packet size.

2. Related Work

Perrig et al. proposed SPINS, which is a suite of security building blocks for resource-constrained wireless sensing environment, in [5]. They separated the suite into SNEP and µTESLA. SNEP provides basic security mechanisms for data confidentiality, two-party data authentication, and data freshness using RC5. µTESLA provides authenticated broadcast. For evaluation, the security suite was implemented for Berkeley’s TinyOS on sensor nodes that runs an 8-bit processor at 4MHz. Code size for SPINS is 2 kilobytes at most and the performance for RC5 was more than adequate for the platform they have chosen. Energy costs of SPINS was mentioned qualitatively. They claimed that most of the energy overhead was spent on communication
(the extra MAC for every message) whereas the actual encryption computation consumes negligible energy.

Law et al. compared two different cryptographic algorithms, TEA [8] and RC5 [6] for code size and energy usage for their EYES sensor platform [4]. Their target processor is a 16-bit Texas Instrument MSP430x149 operating at 3V and 1MHz with limited on-board flash memory and RAM. They measured energy consumption for encryption and decryption for TEA and RC5 to be less than 10 nJ with different optimizations. Given a 1MHz clock, this translates to about 10mW for cryptographic operations.

Venugopalan et al. surveyed computation requirements for a number of common cryptographic algorithms and embedded architectures in [7]. They also modeled the computational overhead for encryption algorithms in general. However, they did not provide energy comparison for these encryption algorithms and architectures.

3. Challenges

The lowest common denominator in prior works is an embedded general purpose microprocessor that will perform encryption, decryption and message authentication operations. However, based on the power measurement by Law et al., it takes about 10 mW to perform these cryptographic operations. Given that the lifetime desired by TPMS providers is 7–10 years and a typical lithium coin battery used in TPMS has about 250–300 mAh of energy, 10 µW per operation is a more realistic energy budget. Furthermore, radio packet will also increase in size due to message authentication code to ensure authentic messages. This further drains the limited energy available at each sensor node.

In order to meet the energy constraint, a custom hardware implementation of the cryptographic algorithm seems necessary. Challenges remain, though, for choosing the easiest and most cost-effective algorithm with “good enough” security for TPMS.

4. Proposed Solution

The first step in achieving the goal of 10µW for security operations would be designing dedicated hardware instead of relying on generic microprocessor unit. The constraint also necessitates symmetric key cryptography since public key algorithms requires more computations (thus, more energy) and much larger memory footprint. Among the plethora of symmetric key algorithms, two, TEA and RC5, emerges as promising candidates. These two algorithms share these important properties: (1) only XOR, addition and rotation operations, (2) avoid large tables and (3) most importantly, are simple to implement (therefore easier to get right).

To further reduce complexities, it is necessary to review the exact security primitive(s) needed in the TPMS application. Although data confidentiality (encryption) is a nice property to have in general (and fundamental in many applications), it is not needed in this particular application because tire pressure readings need not to be secret 1. Allowing an adversary to know your tire pressure readings does not enable him to do harm to your car. Data authenticity and data freshness properties, on the other hand, are extremely important. Data authenticity guarantees that the tire pressure readings come from sensors installed on your tires instead of your neighbor’s tires. It also prevent your on-board computer to accept any tire pressure reading. Data freshness prevents replay attacks 2. A common way to provide data authenticity is to attach keyed message authentication code (MAC) to the message being sent. To provide data freshness, a random number is padded with the sensor reading and the entire message is then MACed. LFSR can be used to generate the random number sequences both on sensor node and also on on-board computer.

To demonstrate feasibility of security on TPMS, I will evaluate the energy and area (gate count) of implementing MAC using either (or both) RC5 or TEA in hardware using latest technology.

References


---

1 One might ask then, why not used a keyed hash algorithm and dispense with these encryption algorithm altogether? The answer is that existing keyed message authentication code (MAC) algorithm is really heavy weight for our purpose.

2 A person can record the packet sent from the sensor when tire pressure is low and then re-send it at later time so the on-board computer will issue low pressure warning even when the tire is properly inflated.
