1 Modern Car Ownership

On February 22nd, Dan Ellsworth presented “Modern Car Ownership: Automotive Attack Surfaces in Newer Model Vehicles.” This presentation focused largely on the paper “Comprehensive Experimental Analyses of Automotive Attack Surfaces.”

1.1 Background

Ellsworth suggests that the mental model most of us have of a car is largely the model with which we were presented in the early 20th century, i.e. wheels and an engine. However, a modern audience expects other features such as fuel injection, anti-lock brakes, traction control, keyless entry, and hands-free calling, among others. The fact that these systems exist, though, adds complexity and requires some kind of computation, e.g. anti-lock brakes require some kind of electronic control unit to determine whether to allow braking or not. Consequently, brakes are no longer mechanical levers, but electric devices.

Cars have a connector—an OBD-II port— which is used to perform quick diagnostics. Originally, such connectors were specific to manufacturers (to create lock-in on who could maintain a vehicle.) Today, we have a standardized connector due to certain legislation so that people could get information off the OBD (and later OBD-II) connector about air quality. The OBD-II port has about five or six different protocols which it can use, depending on the pins used. According to regulations, the OBD-II connector is within about two feet of your steering wheel (excepting manufacturer exception.)

2 OBD-II and the CAN-bus Protocol

Earlier work by the authors looked at compromises of the OBD-II port. Here, they use the CAN-bus protocol, which works using two voltage levels over two pins. A bit is transmitted by increasing the difference between the high and low voltage line. There is no clock shared by communicating nodes.

A single bit on CAN-bus is divided into four segments, each of which has some time associated with it. Collision detection is performed by seeing whether the information on the bus is different from what is being sent in the arbitration field. (n.b. distinct from Ethernet, as one backs off immediately.)

Dan then showed examples of messages in response to various events as received by the CAN-bus, e.g. door opening, key being places in ignition, &c &c. We then moved on to the paper proper.

3 The Findings

Firstly, physical access is boring. What we want to deal with are “indirect physical” attacks and short-range or long-range wireless. The ultimate goal is to show that, by compromising a single component of the car, we can take over all other components and gain full access to all parts of the car.
One indirect physical example is that a mechanic could be the point of weakness. The PassThru system is a system used by mechanics to communicate with the OBD-II system; these systems are basic computers that communicate over unauthenticated TCP/IP using a wireless network, so all manner of attacks against conventional wireless networks are able to be used to access this device. The PassThru box runs a basic Linux variant; if you are able to access a shell, you consequently gain full control of the PassThru system and therefore the OBD-II system in the car.

The stereo system contains the capability for automatic firmware update, so you could put a carefully crafted CD into the vehicle that tells the stereo to automatically update its firmware in a malicious way so that it communicates via OBD to other things in your car. In addition, you can communicate this same update via FM broadcast, or secretly via a downloaded MP3.

A phone with a trojan could silently pair with a car; once a car is paired, the car’s Bluetooth stack is vulnerable to attack. In the absence of a trojan (or in the presence of a smart user), a car will accept unsolicited Bluetooth requests if they have the correct PIN, and brute forcing that takes time, but with enough time (around 12 hours), one can still pair with a car. In addition, with a large number of cars (such as in a parking lot), it would be much easier to find a car willing to pair. Once the Bluetooth is accessed, it (is?) possible to access the Telematics unit.

The Telematics unit is a device that solves problems like locking a baby in a car, being unaware of where someone is, or crashing too much. Because of the amount of control it has, the telematics unit is an amazing attack vector, because it can control or access important car systems and be accessed via both Bluetooth and 3G and voice cellular.

3G is not consistently available, so the focus instead is on traditional cellular systems; because those systems were not designed for data transfer, a system called aqLink is used to transmit data over the voice channel. This is tied to the telematics unit, which allows for 100 byte packets; it also has a buffer overflow vulnerability that takes about 14 seconds to take advantage of at 300 bytes; however, if a handshake doesn’t happen within 12 seconds, it will close the connection.

The handshake consists of a random challenge that it sends to the Telematics Control Center; it only accepts the response once in this state. TCC hashes the challenge with a 64-bit shared key, and sends it back. If this fails, the car sends an alert packet and doesn’t accept new packets until an ACK is received. Luckily, it is easy to determine what the challenge is because cars seed with a specific random number; in practice, 1 in every 128 attempts was successful, but there was also a state in which the unit would apparently accept any request due to a software bug.

Once one has control of the car, it is possible to, among other things, steal the car simply by unlocking the doors and turning over the engine, to listen in on the conversations of the driver and keep track of where the car is being driven.

What can we do to assuage these threats? One option is to revert to purely mechanical cars without computer components. (This was greeted with derision from people who have actually driven cars without power steering such as Joe.) This does not seem to be a viable solution to the problem.
Many of the paper’s suggestions are that the software needs to be better, and that properly written software would alleviate concerns. In addition, the business model of the automotive industry does not help this system, as the industry engages in a lot of security-through-obscurity practices and (as Adam pointed out) outsourcing of other components, including (presumably) software.

Adam suggested that someone could take over a cell tower and stop traffic for a day by calling up cars via aqLink. Joe pointed out that while it is easy to take over an individual car, any kind of large-scale takeover of large numbers of car would be difficult; Reza and Adam disagreed, suggesting that even small-scale takeovers would still be lucrative and could also be used for terrorism.