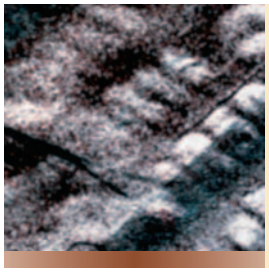
Embedded System Security

Philip Koopman, Carnegie Mellon University

From cars to cell phones, video equipment to MP3 players, and dishwashers to home thermostats—embedded computers increasingly permeate our lives. But security for these systems is an open question and could prove a more difficult long-term problem than security does today for desktop and enterprise computing.

Security issues are nothing new for embedded systems. In 2001, Peter Shipley and Simson L. Garfinkel reported finding an unprotected modem line to a system that controlled a high-voltage power transmission line ("An Analysis of Dial-Up Modems and Vulnerabilities," 2001; www.dis.org/filez/Wardial_ShipleyGarfinkel.pdf). However, as more embedded systems are connected to the Internet, the potential damages from such vulnerabilities scale up dramatically.

This issue is already upon us. Today you can buy Internet-enabled home appliances and security systems, and some hospitals use wireless IP networks for patient care equipment. Cars will inevitably have indirect Internet connections—via a firewall or two—to safety-critical control systems. There have already been proposals for using wireless roadside transmitters to send real-time speed limit changes to engine control computers. There is even a proposal for passenger jets to use IP for their primary flight controls, just a few firewalls away from passengers cruising the Web (Aeronautical Radio, Inc., "Draft 1 of Project Paper 664, 'Aircraft Data Networks,' Part 5, 'Network Interconnection Devices,'" AEEC Letter 01-112/SAI/742, 2 May 2001).



Security for embedded systems involves issues beyond those problems currently being addressed for enterprise and desktop computing.

WHAT'S DIFFERENT ABOUT EMBEDDED SECURITY?

Internet connections expose applications to intrusions and malicious attacks. Unfortunately, security techniques developed for enterprise and desktop computing might not satisfy embedded application requirements.

Cost sensitivity

Embedded systems are often highly cost sensitive—even five cents can make a big difference when building millions of units per year. For this reason, most CPUs manufactured worldwide use 4- and 8-bit processors, which have limited room for security overhead. Many 8-bit microcontrollers, for example, can't store a big cryptographic key. This can make best practices from the enterprise world too expensive to be practical in embedded applications.

Cutting corners on security to reduce hardware costs can give a competitor a market advantage for price-sensitive products. And if there is no quantitative measure of security before a product is deployed, who is to say how much to spend on it?

Interactive matters

Many embedded systems interact with the real world. A security breach thus can result in physical side effects, including property damage, personal injury, and even death. Backing out financial transactions can repair some enterprise security breaches, but reversing a car crash isn't possible.

Unlike transaction-oriented enterprise computing, embedded systems often perform periodic computations to run control loops with real-time deadlines. Speeds can easily reach 20 loops per second even for mundane tasks. When a delay of only a fraction of a second can cause a loss of control-loop stability, systems become vulnerable to attacks designed to disrupt system timing.

Embedded systems often have no real system administrator. Who's the sysadmin for an Internet-connected washing machine? Who will ensure that only strong passwords are used? How is a security update handled? What if an attacker takes over the washing machine and uses it as a platform to launch distributed denial-of-service (DoS) attacks against a government agency?

Energy constraints

Embedded systems often have significant energy constraints, and many are battery powered. Some embedded systems can get a fresh battery charge daily, but others must last months or years on a single battery.

By seeking to drain the battery, an attacker can cause system failure even when breaking into the system is impossible. This vulnerability is critical, for example, in battery-powered devices that use power-hungry wireless communication.

Development environment

Many embedded systems are created by small development teams or even lone engineers. Organizations that write only a few kilobytes of code per year usually can't afford a security specialist and often don't realize they need one.

However, even seemingly trivial programs may need to provide some level of security assurance. Until standard development practice includes rigorous security analysis, developers may overlook even the solutions already available.

EXAMPLE: INTERNET THERMOSTATS

Because embedded systems can effect changes in the physical world, the consequences of exploiting their security vulnerabilities can go beyond mere annoyance to significant societal disruption.

Let's dispense with the most obvious potential attack first. Attackers that break into a computer and get complete control of it can do anything they want with the attached sensors and actuators—send commands to traffic lights, shut down power stations, and so on.

We could argue that industrial-strength security approaches will take care of the really big systems, but smaller systems are less likely to receive lavish attention.

Consider, for example, the household thermostat, which controls heating and cooling. Many have an embedded computer that adjusts the set point a few times each day to keep the house comfortable when people are present and to save energy when they aren't.

Some thermostats let a homeowner use the Internet, perhaps via cell phone, to communicate imminent arrival home after a vacation or a day at work. This gives the thermostat time to reach a comfortable temperature before the owner actually arrives.

However, allowing Internet control of a thermostat gives rise to several potential attacks.

Centralized control

If the system permits transition only between a pair of "comfort" and "saver" set points, an attacker could send false "I'm coming home" messages to change set points and waste energy. If it permits arbitrarily changing set points, the attacker could subject the house to extremes of heat and cold or even turn off the system, causing pipes to freeze in the winter and pets to die of heat in the summer.

Allowing Internet control of a thermostat gives rise to several potential attacks.

Of course, a properly designed system with safety interlocks and a well-administered password policy could prevent this from happening. But the potential for it to occur makes it a threat that must be countered.

Internet thermostats also offer utility companies the possibility of suggesting or demanding changes in thermostat operation during periods of peak demand. Some US electric power companies already use radio commands to disable or reduce the duty cycle of air conditioning units during peak loads. Utility customers volunteer to do this and are compensated for the inconvenience.

Internet thermostats can make this process more sophisticated. The utility could instruct each thermostat to change its set point a few degrees to ease power requirements during peak loads. Because air conditioners form a significant part of peak electricity demand during hot summer days, this mechanism could make the difference between a blackout and continued operation of the power grid.

But such a system creates potential vulnerabilities. For example, someone might trick a number of thermostats into thinking that it is not a peak day, thereby increasing demand. If done on a broad enough scale, this could cause power-grid failure, especially if the electricity provider has factored the ability to change set points into its plan for sizing its generating capacity.

Centralized control for a power-saving scheme can create even more serious problems. Attacks that successfully break into the central control computer for set point commands, or even just spoof commands, can attempt to coordinate power consumption among many homes.

What if someone wrote a virus that took over computers for the purpose of launching attacks on all Internet-connected thermostats? The approach could be subtle, such as bumping thermostat temperature a bit hotter in the winter or cooler in the summer to increase energy consumption and inflate utility bills.

Or it could be not so subtle: If all the thermostats in a city suddenly activated their air conditioners simultaneously during a peak load period, the power surge could cause a significant problem.

Then there are pranks. What if some kid on the other side of the world decided to change your thermostat setting by 20 degrees while you were asleep every night? (Actually, using your home control system to flash your lights on and off might be more entertaining, but we're talking about thermostats.)

Battery attacks

Many thermostats, including at least one Internet brand, are battery powered. This is partly because line voltage isn't available and partly because safely converting line voltage to a thermostat's needs takes a large converter that costs money, requires extra wiring, and complicates electrical safety.

Some thermostats use wireless networking to avoid wiring costs, but too

many networking conversations can run the battery down quickly. If the thermostat is connected to the Internet, an attacker could run the battery down simply by repeatedly querying the thermostat's status.

A low-voltage detection circuit could disable the wireless connection before the battery died, but the developer needs to design this capability into the system.

Privacy

A person who can monitor your thermostat setting could also determine whether you're likely to be asleep, at home, or out of the house. Even if an attacker can't query the thermostat directly, simply monitoring traffic for inbound packets talking to the thermostat can indicate whether the house is vacant—and a potential burglary target.

An Internet-enabled thermostat can also let Big Brother monitor whether you're setting it properly to do your part for an energy crisis—and set it for you if you're not. Some gas, water, and electric meters already report to utility companies via modem, so the infrastructure for automated utility monitoring is already in place.

A thermostat controls only a limited amount of energy release, and people are often around to notice it's misbehaving. Other application areas are more challenging—for example, connecting vehicles to the Internet.

In many ways, we aren't ready to deal with the security challenges we are sure to face. Some involve simply ensuring that design teams acquire the right skills as they start making products that are exposed to security risks, but others involve significant research before we can hope to address them.

For example, how do we create impenetrable firewalls to keep attackers from manipulating safety-critical sensors and actuators? How can we ensure that real-time deadlines will be met, even in the face of DoS attacks or compromised system components? Can we create intrusion-detection systems that can respond fast enough to restore a system to correct operation before a 50-millisecond control loop loses stability?

Can we securely upgrade unattended embedded systems without being vulnerable to attacks on the upgrading mechanism? Can we detect and avoid attacks designed to drain batteries? Can we do all this on a $1 microcontroller?

We have made progress on some of these problems, of course. But some important areas aren't on enterprise-focused research agendas, and the need for embedded security is already upon us. ■

Philip Koopman *is an associate professor in the Department of Electrical and Computer Engineering at Carnegie Mellon University, where he is also a member of the Institute for Complex Engineered Systems and the Institute for Software Research International. Contact him at koopman@cmu.edu.*