A Conversation, Circa 1981

**Me:** You get a lot more performance if you buffer disk I/O

**Hobbyist:** But then I can’t just eject the floppy

**Me:** You also need memory protection

**Hobbyist:** Why? I’m the only one using the machine

**Me:** (Argghhh!)


**A Talk, Circa 1982**

Me: Writing code in a high-level language will improve productivity and reduce bugs

Audience: You don’t understand how small these machines are!

Me: They’ll get bigger

Audience: But today they’re small
“Programs written specifically for IBM compatibles could run faster by bypassing slow MS-DOS functions, e.g. by writing video information directly to the area of memory assigned to it.” —Wikipedia entry on DOS

That meant that Windows 95 had to permit such behavior, and hence couldn’t really run protected.

Windows 98 couldn’t, either; on Windows XP, most users run as Administrator because many applications require it.

*We are paying today for decisions made 25 years ago.*
A History Lesson

Mainframes, 1960  Single application at a time, no memory protection, limited address space

The Root Cause

There is a Threat
Cell Phone/PDA
Viruses
More Problems
Bellovin’s Laws of Networking
Interconnections
We Have to Start Somewhere
The Square Wheel
Parts of a Solution
Securing New Systems
Principles
Solution
Characteristics
Retrofits
It May be Easier
A History Lesson

Mainframes, 1960  Single application at a time, no memory protection, limited address space
Minis, 1970  Single application at a time, no memory protection, limited address space
A History Lesson

Mainframes, 1960  Single application at a time, no memory protection, limited address space
Minis, 1970  Single application at a time, no memory protection, limited address space
Micros, 1980  Single application at a time, no memory protection, limited address space
Mainframes, 1960  Single application at a time, no memory protection, limited address space
Minis, 1970   Single application at a time, no memory protection, limited address space
Micros, 1980  Single application at a time, no memory protection, limited address space
PCs, 1990    Single application at a time, no memory protection, limited address space
A History Lesson

Mainframes, 1960  Single application at a time, no memory protection, limited address space

Minis, 1970   Single application at a time, no memory protection, limited address space

Micros, 1980   Single application at a time, no memory protection, limited address space

PCs, 1990   Single application at a time, no memory protection, limited address space

Embedded systems, now . . .
Those who cannot remember the past are condemned to repeat it.

—George Santayana, 1906
The Root Cause

- Vendors shipped as soon as the hardware was capable of handling base functionality
- A year later, the better hardware is used for more functionality
- By the time people think about security, there’s an installed base problem
- Besides, no one believed there was a problem
- We have two challenges:
  - To ensure that new systems are designed properly
  - To figure out how to retrofit legacy systems
A Conversation, Circa 1981
A Talk, Circa 1982
The Sins of the Fathers...
A History Lesson

The Root Cause

There is a Threat
Cell Phone/PDA
Viruses
More Problems
Bellovin’s Laws of Networking
Interconnections
We Have to Start Somewhere
The Square Wheel
Parts of a Solution
Securing New Systems
Principles
Solution
Characteristics
Retrofits
It May be Easier

“Software longa, hardware brevis”
—Melinda Shore
There is a Threat

- 34 security incidents targeted at process plants were identified between 1995 and 2003.
- 29% of the incidents led to companies losing the ability to monitor or control the plant.
- 36% of external attacks came through the Internet.
- The number of incidents has been increasing sharply since 2000.

“Prepare for the likelihood of an increasing number of threats as time goes on.” (Microsoft.com)

“Cardtrap.A, a Trojan that attacks Symbian mobile phone operating systems, attempts to infect users’ PCs if they insert the phone’s memory card into their computers.” (news.com)

“What if a virus drained your cell’s battery and suddenly you couldn’t be reached?” … “Once initiated, it sends the attacker an email containing the IP address of your PDA.” (Symantec.com)
More Problems

- Systems are not designed for the threat model
- Note the third bullet on an earlier slide: SCADA systems are being attacked through the Internet
- Why are SCADA systems even connected to the Internet?
1. Networks interconnect
Bellovin’s Laws of Networking

1. Networks interconnect
2. Networks always interconnect
Bellovin’s Laws of Networking

1. Networks interconnect
2. Networks always interconnect
3. Networks interconnect at the edges, not the center
Interconnections

- No one deliberately connects an unprotected SCADA system to the Internet
Interconnections

- No one deliberately connects an unprotected SCADA system to the Internet
- On the other hand, it’s perfectly reasonable to connect a SCADA network to the corporate net
No one deliberately connects an unprotected SCADA system to the Internet

On the other hand, it’s perfectly reasonable to connect a SCADA network to the corporate net

Of course, the corporate net is (and should be) connected to the Internet...
We need to start on a solution *now*

- We need to learn what hasn’t worked
- In that category I place doing nothing, relying on obscurity, and assuming that a corporate net is secure
- I also assert that general-purpose subsets of corporate nets, even if firewalled, are likely to be insecure
The Square Wheel

“‘I’ve invented the triangular wheel. It’s a great improvement over the square wheel.’”

“Why is that?”

“One less bump!”
Parts of a Solution

- We need an architecture for secure new systems
- We need a way to layer a solution onto old systems
Securing New Systems

- The solution must be based on sound cryptographic and software engineering principles
- We can’t afford to cut corners again
- We can’t be hobbled by performance myths (see David Wagner’s talk on myths about sensor nets)
- You can do a remarkable amount of crypto in a very small system these days
- Measure before you say it can’t be done — and if it can’t be done today, it will probably be possible before your code is finished
“Moderate loss of local system efficiency due to judicious application of the principles often results in a gain in effectiveness under reasonable global cost metrics.”

–Peter Neumann, 1969
Solution Characteristics

- Universality — *all* requests must pass a security check
- Authentication
- Authorization
- Auditability — use (limited) local memory for short-term audits; keep larger, long-term logs at a border controller
- Updatability — security upgrades *will* be needed
Solution Characteristics

- Universality — *all* requests must pass a security check
- Authentication
- Authorization
- Auditability — use (limited) local memory for short-term audits; keep larger, long-term logs at a border controller
- Updatability — security upgrades *will* be needed
- Note that this list is identical to that for a conventional operating system
Retrofits

- Don’t rely on corporate firewalls
- Implement security principles via front ends
- We need application-specific firewalls, for the protocols used on embedded systems
Security for embedded systems is probably not harder than for general-purpose systems

It may be easier — they do fewer things

We need to understand what the hardware limits actually are
Mostly, though, we need the willpower to get around to it