Code Security

Phillip Hallam-Baker
Default Deny Security

In case you are looking for the ‘pitch’…

- Not a vendor…
  - Education (mostly free podcasts)
  - Protocol design (mostly open standards)
  - Professional speaking (often free)

When I did control engineering…

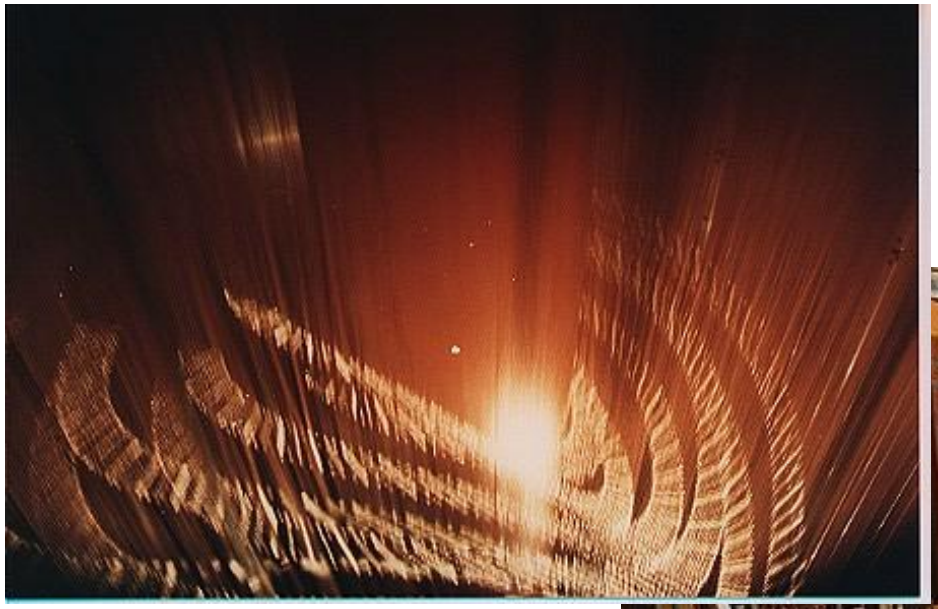
Control Engineering circa 1984

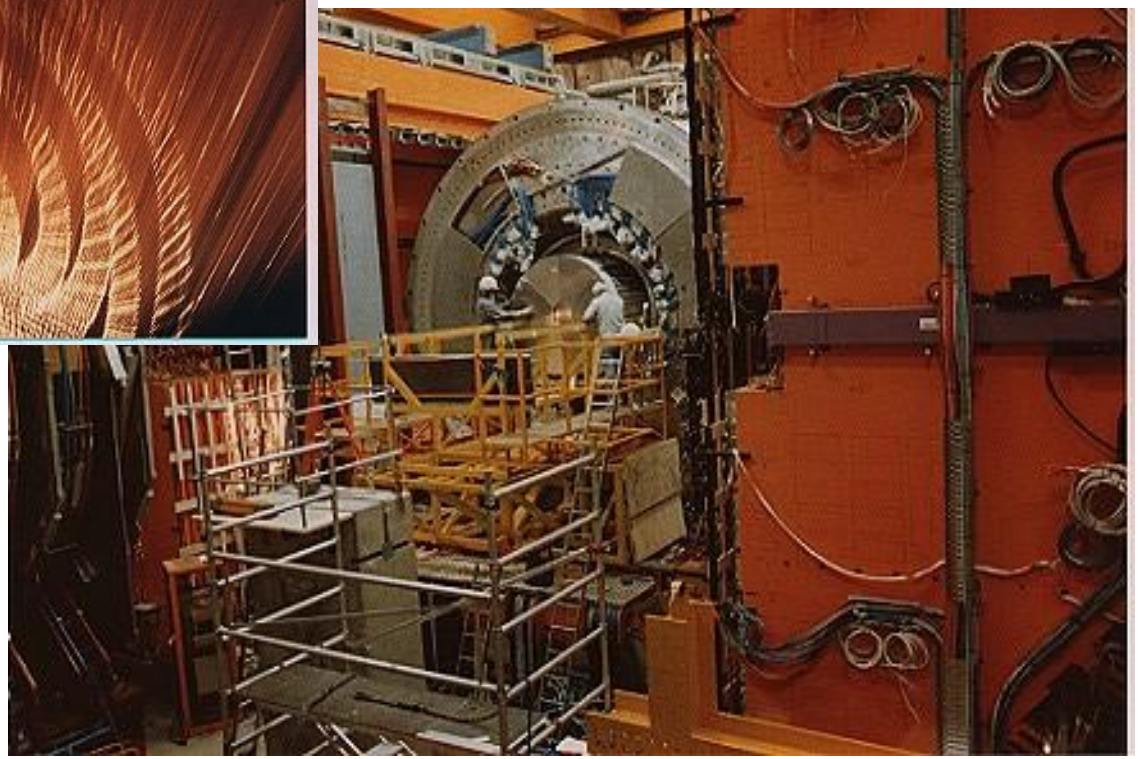
- The major market for micro-controllers
  - Micro-computers built using re-purposed chips designed for use in industrial control
- Pneumatic control still common
  - Digital controls regarded with suspicion
  - Digital controls had to prove themselves secure

What I learned from Control Engineering

- Rules of robustness
  - Keep it simple
    - Eliminate unnecessary features
  - Code for reliability, not speed
    - Validate inputs and outputs
    - Check array bounds

Where I went Next

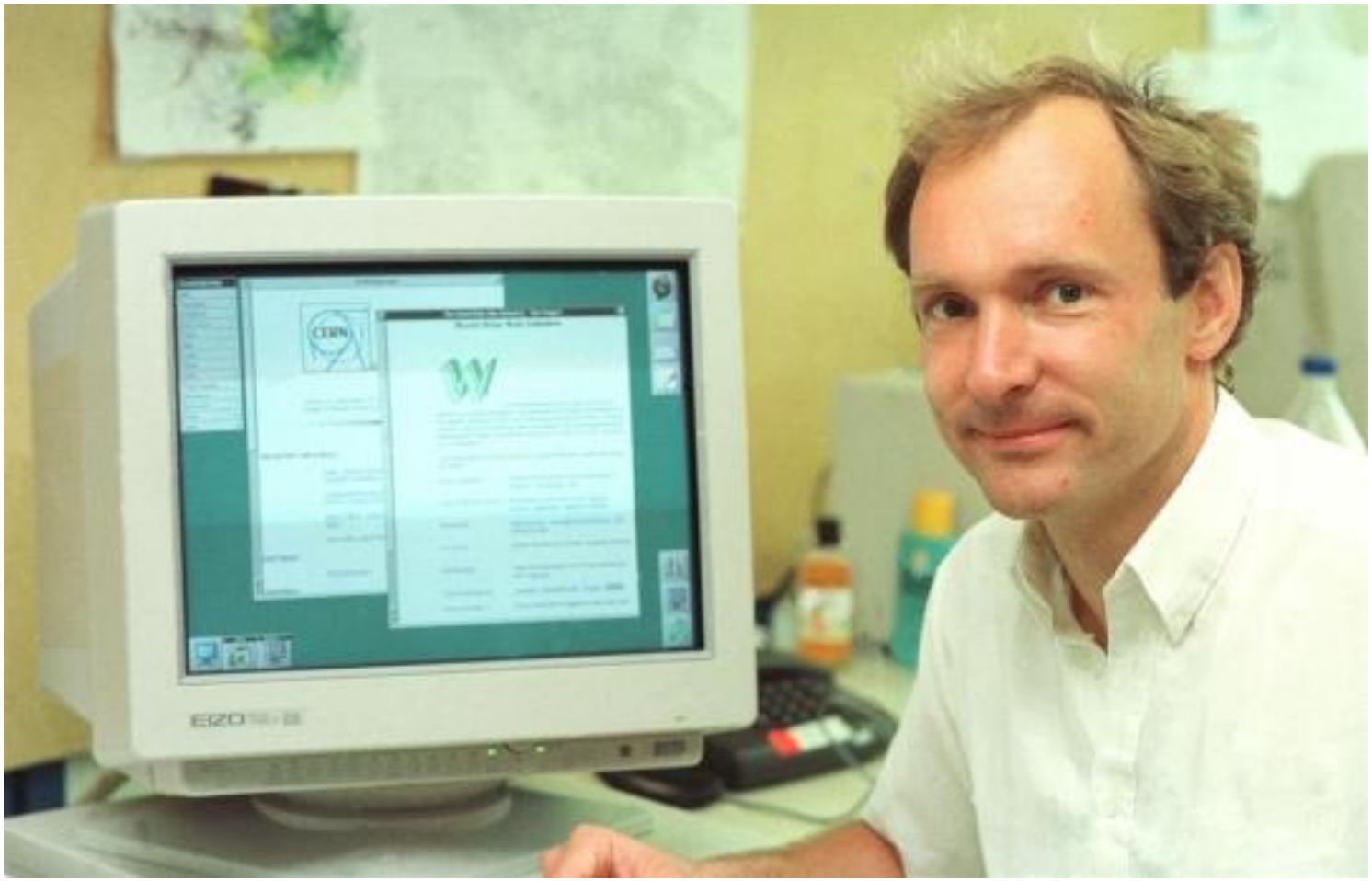




1990 ZEUS Data Acquisition: 6 TB/sec

- One great big SCADA System
- Formal proof of correctness
  - On > 5000 lines of code

Who I met next



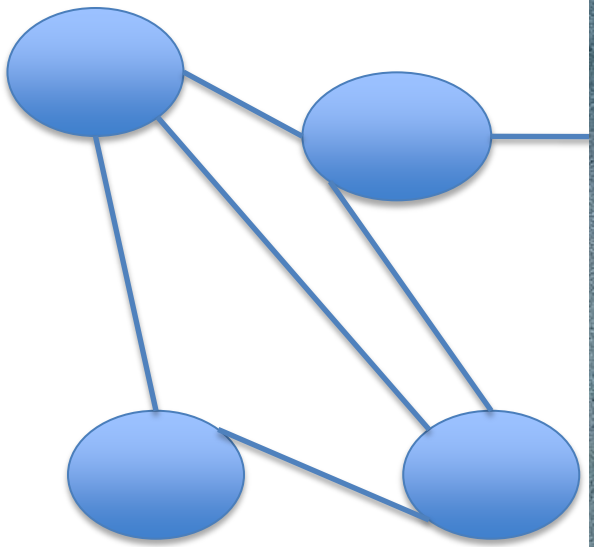
18 years later…

- Serious concern for ‘SCADA security’
- Based on consequences, not risk
- The issue is Security of Industrial Controls
  - The SCADA subsystem is only one concern

What has changed?

- Industrial Control no longer a killer application
  - Network protocols have stagnated
    - MODBUS still lives
  - Systems increasingly built on consumer designs
    - Many consumer devices have better security
  - Consumer device security is generally poor
    - WEP, DEC6.0, Bluetooth, Zigbee, all deficient

What has changed?



Internet

What has changed?

- Capabilities of attackers
  - Money buys anything: Botnets, Exploits
- Level of violence
  - PIRA / PLO / RAF limited murder of civilians
    - Had capacity to kill far more
    - Bologna bombing was a false flag operation
  - McVeigh, Al Qaeda changed the tactics
    - Intentional murder of 100s, 1000s

What has changed?

- ROM is now Flash ROM
  - Early PLCs and PIDs used ROM or EPROM
  - Today they use Flash ROM
  - Code can be modified via the network
    - Often no security controls at all
  - Forget buffer run exploits, I'll just change the code

Why is that bad?

Leave Behinds

- Purpose
  - Exploit code
  - Re-infection code
  - Backdoors
- Media
  - In memory process
  - Attached storage (hard drive, flash memory)
  - Removable media (USB key)

Cost of an incident?

Direct costs

- At least comparable to incidents with natural cause.
- Not a threat to civilization.

Indirect costs

- Do Something!
- Audit every SCADA control system in the country
  - Instant Y2K bug
- Deploy upgrades
  - Whether needed or not
- Punish innocent along with the guilty!

How do we start fixing it?

Enterprise Security

- Is in a far worse mess
  - Security policies designed to minimize nuisance
    - Stop spam interrupting work
    - Stop viruses interrupting work
  - Default-Deny security policies are very rare
    - Few companies would tolerate them
- Is the wrong model
  - Systems are very complex
  - Systems change frequently
  - Nobody understands the system

Back to the Future

- Lets get back to 1980s
  - Keep the simplicity
  - Close the firmware hole
  - Add network security behind the firewall

Closing the firmware hole

- Not a technical problem
  - It is a marketecture problem
- How do we get manufacturers to create the products with the security characteristics we need?

RFP

Does the device contain?

- CPU element(s)
- Firmware
- Operating System
- Application Software
- Configuration Code

How is code/configuration managed?

A. Device has no CPU, no code to change
B. Code is permanently fixed at factory
C. Code updates controlled by cryptography
D. Code updates require physical access
E. Code updates are not secure

- For B, C, D explain how this is achieved
- Give answers for each layer in stack

Do we want upgrade capability?

- What is the device function?
  - Temperature probe – simple, unlikely to change
  - PLC – simple but might change
  - SCADA hub – expect frequent change
- How much does the device cost?
  - $50 device is probably not worth upgrading
  - $1000 device is probably unacceptable bricked

Consequences

	A N/A	B Fixed	C Crypto	D Physical	F None
Manufacturer can issue updates?	No	No	?	Yes	Yes
Owner can issue updates?	No	No	?	Yes	Yes
Protection against:					
Random Target Attack (e.g. Virus)	High	High	High	High	Low
Targeted Remote Attack	High	High	High	High	Low
Local attack (e.g. insider)	High	High	High	Low	Low
Security	High	High	High	Med	Low

Code Signing Technology

- Many examples in use
  - Authenticode signatures of code from net
  - Windows, Microsoft Office
  - .NET Framework
  - PGP
- Core technology is unencumbered
  - Algorithms, mechanisms are > 20 years old

Basic Principles

- Digital Signature
  - A Cryptographic System with 2 keys:
  - Private key is used to create a signature
    - Ideally protected using hardware token
  - Public key is used to verify a signature
    - Knowledge of public key or signatures does not allow
      - Creation of a signature
      - Determination of a private key

Simple Code Signing

- Frequently used for
  - Patch updates
  - Source and Binary Distribution
- Code publisher publishes key
  - My key is 19293020……
  - Fingerprint of my key is A2ED …

Code Signing with PKI

- Simple system authenticates code
  - But how do we distribute keys?
  - What if the attacker modified the key as well?
- Digital Certificate
  - Use keys to sign keys
    - Publisher signs code
    - Certificate Authority signs key to create certificate
- Certificate Chains
  - Keys signing keys that sign keys that …
  - Ultimately Signed by a Root

PKI Roots

- Private
  - CA serves a closed group
    - A Company
    - A Government Agency
- Public
  - CA signs keys for anyone who meets criteria
    - Domain Validated SSL Certificates
    - Extended Validation SSL Certificates

Trustworthy Hardware

- Only runs code signed by a trusted party
  - Directly signed by key
  - Requires ability to verify signature in device

Who owns the root?

- Manufacturer Locked-In
  - Root is embedded, cannot be changed
  - E.g. Game Console, iPhone
- Owner Locker-In
  - Owner can change root, so can attacker
  - e.g. Nothing intentional (jailbroken iPhone)
- Attestation
  - Owner can change root, so can attacker
  - Remote party can verify the trust root
  - Trustworthy Computing Group devices

Choices, choices

Manufacturer Locked-In

- Simple to implement
  - Unencumbered
- Only manufacturer can:
  - Introduce malicious code
  - Implement extensions
  - Fix bugs
- Appropriate for
  - Most appliance firmware

Attestation

- Requires some complexity
  - May be encumbered
  - Requires a locked-in subsystem
- Anyone can change root
  - But changes are detectable
  - Can prevent modified code accessing the network
- Appropriate for :
  - Supervisory computers

Current Status

- Trustworthy hardware is available
  - Targeted at enterprise use
    - We don't want a complete O/S here!
  - Starting to see interest in embedded O/S
- Perceived as an enabling technology for DRM
  - Lock-in vs. Attestation is not well understood
  - Has harmed adoption by open source groups
    - GPL3 has anti-Trustworthy computing clauses

RFP Detail on C

- Who can sign code?
- How is code validated?
  - Not / Before Installation / Before Execution
- What roots are supported?
  - Manufacturer / Public / Site-Specific
- What trustworthy computing models are used?
  - Locked-in / Attestation

[Where multiple configurations are supported check all and explain how choice is secured]

Actions

- Write standard questionnaire for use in RFPs
- Develop standards based support for option C
  - How to manage code/configuration updates

Next Targets

- Cryptographic protocols for restricted devices
  - Designed to reduce administration costs
  - Open, Free Specifications
    - Aim to co-opt maker-space
- Verifiable hardware
  - Alarm system must have read access
  - Prove that it has no write access
  - e.g. use opto-isolation

For more information

- http://hallambaker.com/
  - Links to podcast security series (coming soon)
  - Links to papers
- The dotCrime Manifesto
  - Sets out the default-deny architecture