Important From Last Time

- A system is safety critical when its failure may result in injuries or deaths
- Verification and validation can dominate overall development effort

Today

- Embedded system security
  - General principles
  - Examples

Computer Security

- This is a huge area
  - Prof Kasera teaches a good course on it
- Today we are not talking about
  - Protocol design (another huge area)
  - Password issues
  - Access control
  - Cryptography (huge area)
  - Multilevel security
  - Network security

Old Joke

- Q: What does a secure computer system look like?
- A: It's buried in concrete, with the power turned off and the network cable cut

Embedded Security

- Main difference with respect to network security:
  - Attacker has access to the hardware

Trusted Computing Base

- Any secure system has a trusted computing base (TCB)
  - If the TCB operates properly, the system is secure
  - By definition, attacks do not originate from the TCB
- Obviously a smaller TCB is better
  - But almost always the compiler and OS are in the TCB
- Difficult to maintain integrity of TCB when attacker has access to the hardware
- Schneier: "A 'trusted' computer does not mean a computer that is trustworthy."
- U.S. DoD: "…*a system that you are forced to trust because you have no choice."*

TCB Example

- From Ken Thompson's Turing Award lecture "Reflections on Trusting Trust"
- What if the compiler recognized that it was compiling the OS and inserted a trapdoor?
  - Vulnerability not found anywhere in OS source code
- Compiler also has to recognize that it's compiling itself and add the attack code
  - Problem not found in the compiler code either
- Not a theoretical attack – this was implemented!
- Defenses against this?

Diverse Double Compilation

- The question is: Does the source code for the compiler correspond to its executable?
  - Here we assume that any attack code in the source would be detected through auditing
- Start with a compiler C1 that may be bad, and its source code CS
- Compile CS using C1 to generate C2
- Compile CS using a totally different compiler to create C3
- Compile CS using C3 to generate C4
- If C2 and C4 are bit-for-bit identical then C1 cannot be inserting attack code

Threat Models

- Makes no sense to discuss security without a threat model
- Components of a threat model:
  - Who is the attacker?
  - What are the attacker's goals?
  - What are the attacker's capabilities?
- Example classification:
  - Class 1 – Casual user
  - Class 2 – Clever, motivated outsider
  - Class 3 – Knowledgeable insider
  - Class 4 – Funded organization or government

System Questions

- How long must the system remain secure while under attack?
- Does the system need to be usable during the attack?
- Does the system need to be usable after the attack?
- Does the system require human intervention to remain secure?
- What sort of …
  - increase in cost
  - decrease in performance
  - decrease in usability
- is acceptable to achieve security?

Threat Model Examples

- What are some potential threat models for:
  - The door locks on your house?
    - Most everyday physical security systems are like this
  - Your laptop?
  - Your home computer?
  - A voting machine?
  - Your bank's ATM?
  - The GPS system?
  - A military mobile communications system?

Pacemaker Hacks

- Pacemakers have a magnetic switch: Under a magnetic field, they turn on a radio receiver
- When the radio is on, pacemaker can be queried and reprogrammed
- Researchers at UMass used a software radio to reverse-engineer the radio protocols
- It was possible to change device settings, change or disable therapies, and deliver shocks
  - Attacks were just replays of known signals

ATM Security

- **ATMs are a good case study**
  - In wide use for several decades
  - Substantial rewards for successful attacks
- **Fact: ATMs were the "killer app" for modern cryptography**
  - Earlier, crypto was a niche technology used by governments and militaries
- **First: What are the threat models?**

Review: Private Key Crypto

- **Given a private key and a block of data, a private-key algorithm encrypts the data so that it cannot be decrypted without the key**
  - Also called "symmetric key cryptography"
- **This technology is simple and efficient to implement**
- **DES and AES (Rijndahl) are popular examples**
- **Of course attackers are free to try to:**
  - Guess the key
  - Steal the key
  - Gain access to the unencrypted data
  - Etc.

ATM Security Overview

- **Each ATM has its own secret key**
  - Entered into ATM keyboard in two parts by two bank officials
- **When you use the ATM**
  - Account number is read from the magnetic stripe on your card
  - It's encrypted using the ATM's secret key
  - Resulting encrypted value is checked against your PIN
- **ATM has a "security module"**
  - Piece of trusted, tamper-proof hardware
  - Unencrypted data never leaves this module

What Goes Wrong in ATMs

- **Processing errors on the bank mainframe side cause lots of problems**
  - Error rate between 1/10,000 and 1/100,000
- **Mail fraud gives attackers cards and PINs**
- **Fraud by bank staff in poorly-run banks**
  - E.g. what could happen if both parts of an ATM key are given to a single worker?
- **Encryption is single-DES**
  - Can be brute forced

More ATM Problems

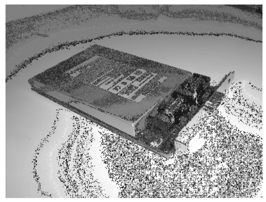
- **Repairman installs computer inside an ATM that sniffs and records card and PIN data**
- **Criminal finds PINs by looking over people's shoulders, then account numbers from receipts**
- **One kind of ATM would give out 10 bills when a specific 14-digit number was entered**
- **False terminals are used to collect lots of PINs**

Physical Tamper Resistance

- **Physical security is important**
  - Historically, naval code books were weighted so they could be thrown overboard in event of capture
  - Russian one-time pads were printed on cellulose nitrate
  - Bank servers are in a guarded computer room
  - ATM is basically a PC in a safe with some fancy peripherals

Secure HW: IBM 4758

- History: As computers got cheaper, location-based physical security became impractical
- PINs etc. cannot be trusted to standard HW/SW
- “The IBM 4758 is a secure crypto-processor implemented on a high-security, programmable PCI board.”



Cryptoprocessor Goals

- Critical data (keys) never leaves the device
  - Resist sniffing attacks
  - Resist physical attacks – attacker has a logic analyzer
  - Resist software attacks

Cryptoprocessor Features

- Robust metal enclosure
- Tamper-sensing mesh
- Key memory: Static RAM designed to be zeroed when the enclosure is opened
  - Data is kept moving to avoid burn-in
  - Freezing and radiation attacks difficult to foil
  - Military systems have used self-destruct
- Trusted core is “potted” in epoxy
  - Crypto processor
  - Key memory
  - Tamper sensors
  - Alarm circuitry
  - Forces attacks to involve cutting, drilling, etc.

Smartcards

- Smartcard:
  - Microcontroller
  - Serial interface
  - Packaged in a plastic card or a key-shaped device
- Tiny secure processors cannot use many features of the IBM 4758
  - However, bar is lower – these aren’t guarding an entire bank’s resources
- Single most widespread use: GSM phones
- Why are smartcards attractive?
  - Can validate that someone paid for something without contacting a central server

Smartcard Attacks

- Protocol attacks – sometimes it enough to listen to communication between the card and world
  - Defense: Avoid stupid protocols
- Stop the card from programming EEPROM
  - Vpp is higher than Vcc, requiring a voltage multiplier or external programming power
- Slow down the processor, then read voltages from the surface of the chip
  - Defense: Detect low clock rates
- Probe wires on the chip – probing the processor bus gives both code and data values
  - Defense: Surface mesh
- At present: Probably not feasible to build a smartcard that is secure when attacked by an equipped expert

Trusted Computing

- You need to trust Windows and Linux with any data on your computer
- However: Content providers cannot trust Windows and Linux
  - Consider the distribution of encrypted movies with software decryption in the OS kernel
- Trusted computing: Create PCs that content providers can trust
  - Said a different way: It’s not really your PC
  - Fundamentally tough problem: Give consumers the bits without giving them the bits

Trusted Computing Elements

- Endorsement key – a key unique to your machine that you must not get
- Protected I/O paths – data channels between processor and peripherals that cannot be altered or read
- Memory curtaining – areas of RAM for trusted computing that even the OS does not have access to
- Remote attestation – your computer can attest that it is your machine and has not been tampered with

More TC

- Digital rights management
- Preventing cheating in online games
- Protection from identity theft

- So… is it good?

Conclusions

- Embedded security is hard because the hardware is out in the world
- Only security experts should connect embedded systems to networks
  - Take a good security course if you're going to do this stuff
- Non-networked systems at least have a chance