Exploring Embedded System Vulnerabilities

John Cook
Herkimer College

What is an embedded system?

- An embedded device is an object that contains a special-purpose computing system. The system, which is completely enclosed by the object, may or may not be able to connect to the Internet.

- Embedded systems have extensive applications in consumer, commercial, automotive, industrial and healthcare markets. It's estimated that by 2015, over 15 billion embedded devices will be connected to the Internet, a phenomenon commonly referred to as the Internet of Things.

- Generally, an embedded device's operating system will only run a single application which helps the device to do its job. Examples of embedded devices include dishwashers, banking ATM machines, routers, point of sale terminals, game consoles, CATV boxes and cell phones. Devices that can connect to the Internet are called *smart* or *intelligent*. If an embedded device can not connect to the Internet, it is called *dumb*

- Source: http://whatis.techtarget.com/definition/embedded-device

Why explore embedded systems?

- Education
- Emulation
- Repair
- Repurposing devices
- Enhancing Capabilities
- Security/Patching

Can I do this?

Prerequisites:

- Fundamental understanding of:
  - Networking
  - Computer Hardware
  - Serial/USB Connectivity
  - Linux/Shells (Bash)
  - Scripting/Runtime: HTML/Python/JavaScript/PHP
  - Basic Electricity/Electronics
- Exploring executables =Low Level=Advanced!:
  - Disassembler/debugger (IDAPro, etc)
  - Assembler
  - Processor Architecture

Internal/External interfaces

- Explore/Exploit **existing interfaces**:
  - Fuzzing
  - Hidden menus
  - Non-standard input device
    - An Xbox controller is USB, can I hook up a USB keyboard, USB memory, keyboard?
  - Give it what you want it to have->TFTP/HTTP(s) configuration

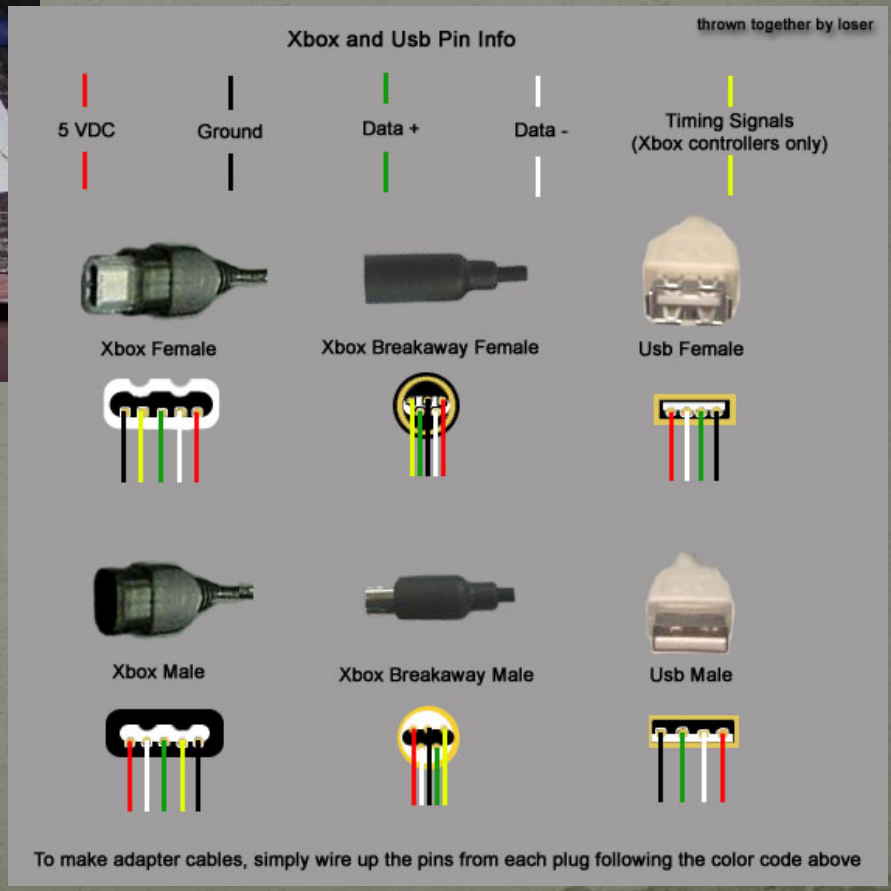
- Many devices have internal hardware interfaces:
    - Serial (RS232)/UART
    - JTAG (Joint Test Action Group)
    - SPI (Serial Peripheral Interface)

  - Interactive use: (serial, JTAG) Terminal access, often with no authentication
    - Cisco console ports (RS232)
    - Often 5v logic
    - Onboard direct UART Connection
      - Serial often done directly from microcontrollers via a UART (Universal Asynchronous Receiver/Transmitter)
      - Often 3.3v logic

  - Non-interactive: (JTAG,SPI)
    - Extract/Program directly from/to onboard NVRAM

External interfaces

Original Xbox controller was USB with a non-standard connector.



Images stolen from:
http://natt-pklegends.synthasite.com/modding-stuff.php
http://www.instructables.com/id/Original-XBox-Controller-Memory-Card-Slot-USB-Conv/

Internal/External interfaces

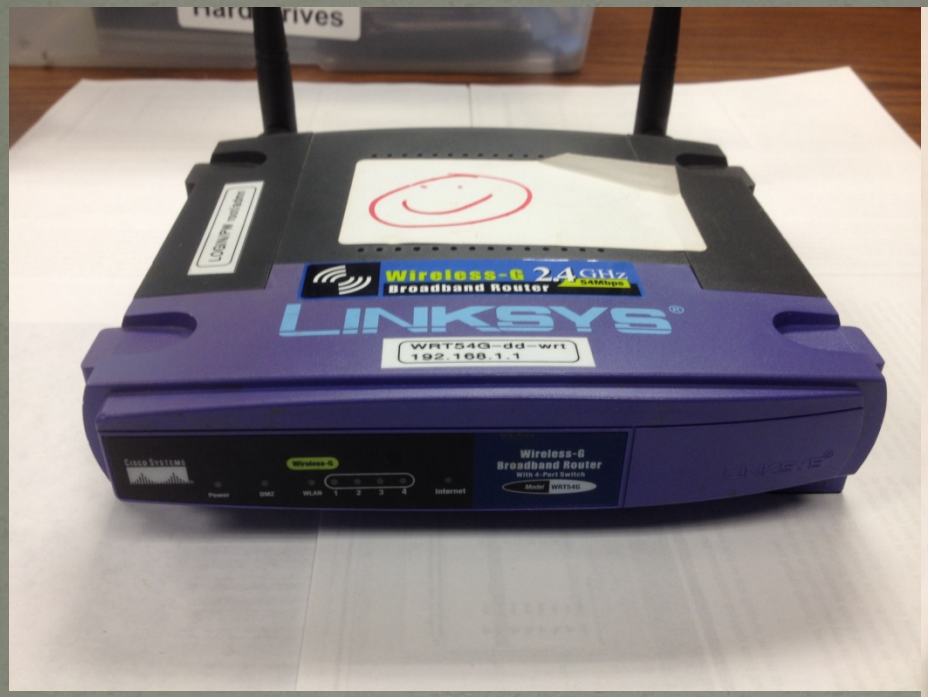
- Explore/Exploit existing interfaces:
  - Fuzzing
  - Hidden menus
  - Non-standard input device
    - An Xbox controller is USB, can I hook up a USB keyboard, USB memory, keyboard?
  - Give it what you want it to have->TFTP/HTTP(s) configuration

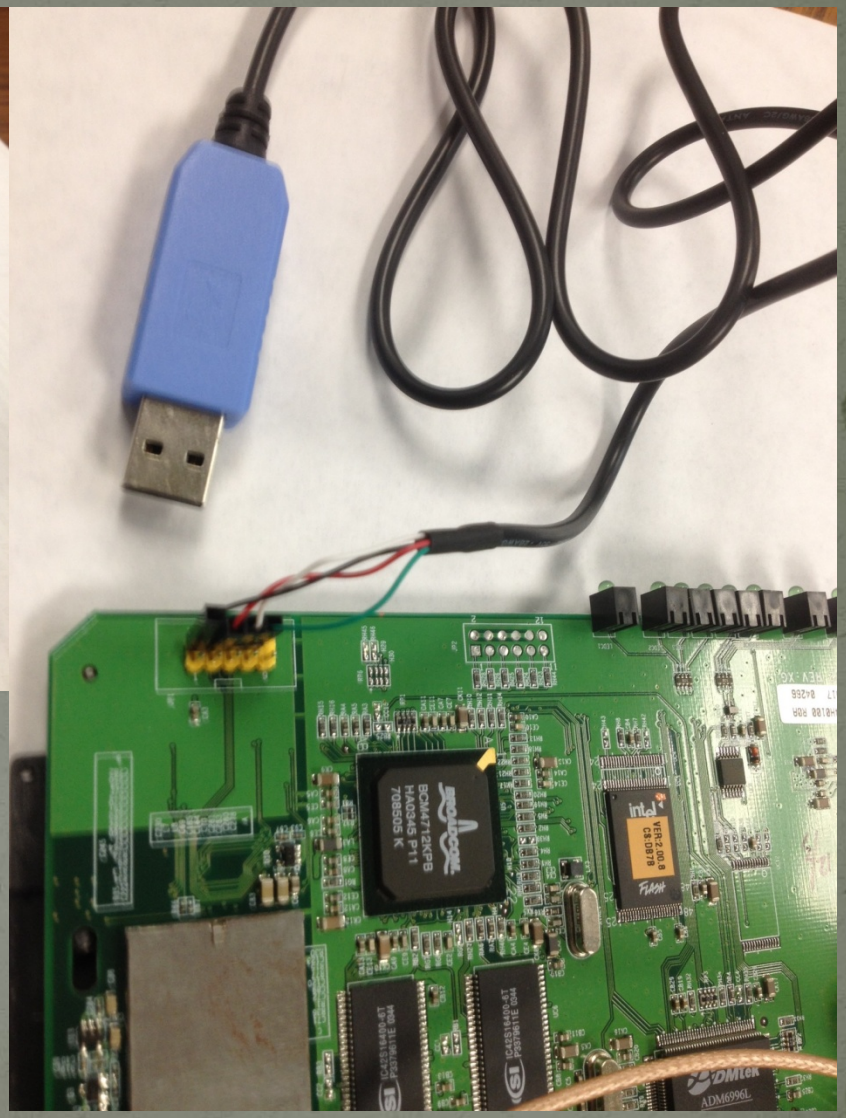
- Many devices have internal hardware interfaces:
    - Serial (RS232)/UART
    - JTAG (Joint Test Action Group)
    - SPI (Serial Peripheral Interface)

  - Interactive use: (serial, JTAG) Terminal access, often with no authentication
    - Cisco console ports (RS232)
    - Often 5v logic
    - Onboard direct UART Connection
      - Serial often done directly from microcontrollers via a UART (Universal Asynchronous Receiver/Transmitter)
      - Often 3.3v logic

  - Non-interactive: (JTAG,SPI)
    - Extract/Program directly from/to onboard NVRAM

Internal Hardware Interfaces





This is a Linksys WRT54G; It had failed.

Inside, a number of unpopulated pin connections existed, including a serial connection (un-authenticated) with access to a pre-boot interface, which permitted firmware upload!

Internal/External interfaces

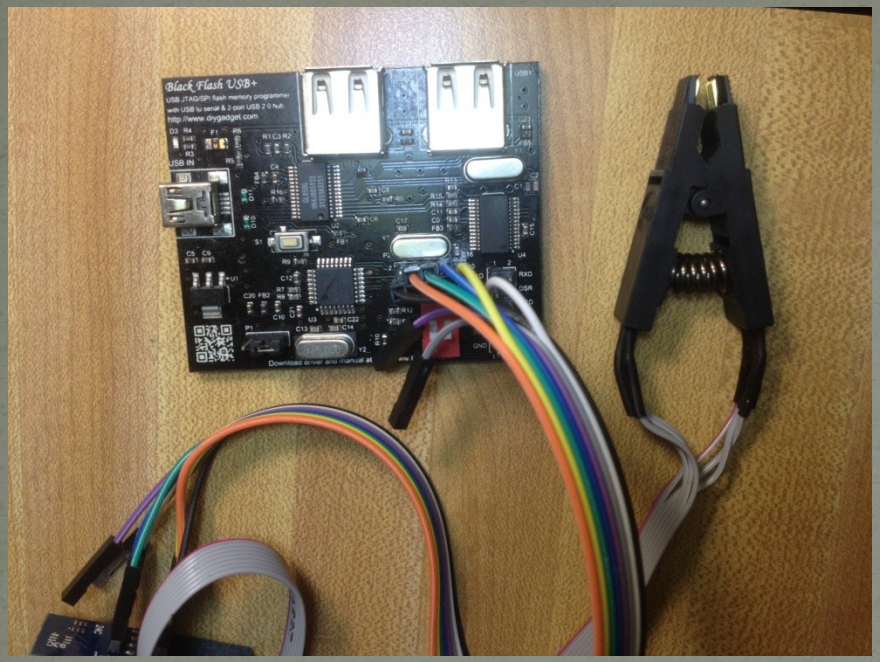
- Explore/Exploit existing interfaces:
  - Fuzzing
  - Hidden menus
  - Non-standard input device
    - An Xbox controller is USB, can I hook up a USB keyboard, USB memory, keyboard?
  - Give it what you want it to have->TFTP/HTTP(s) configuration

- Many devices have internal hardware interfaces:
    - Serial (RS232)/UART
    - JTAG (Joint Test Action Group)
    - SPI (Serial Peripheral Interface)

  - Interactive use: (serial, JTAG) Terminal access, often with no authentication
    - Cisco console ports (RS232)
    - Often 5v logic
    - Onboard direct UART Connection
      - Serial often done directly from microcontrollers via a UART (Universal Asynchronous Receiver/Transmitter)
      - Often 3.3v logic

  - Non-interactive: (JTAG,SPI)
    - Extract/Program directly from/to onboard NVRAM

Interface interfaces? ;)



<- BlackFlash USB+ USB, SPI, Jtag

USB to Serial

Internal/External interfaces

- Explore/Exploit existing interfaces:
  - Fuzzing
  - Hidden menus
  - Non-standard input device
    - An Xbox controller is USB, can I hook up a USB keyboard, USB memory, keyboard?
  - Give it what you want it to have->TFTP/HTTP(s) configuration

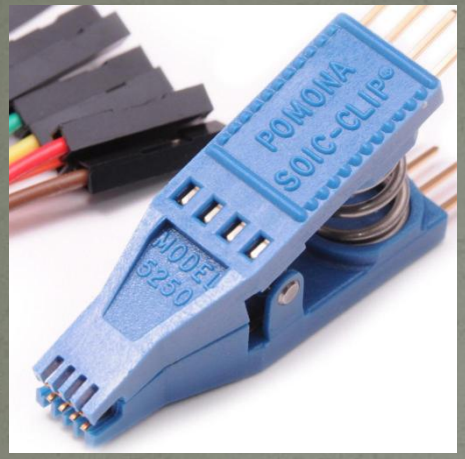
- Many devices have internal hardware interfaces:
    - Serial (RS232)/UART
    - JTAG (Joint Test Action Group)
    - SPI (Serial Peripheral Interface)

  - Interactive use: (serial, JTAG) Terminal access, often with no authentication
    - Cisco console ports (RS232)
    - Often 5v logic
    - Onboard direct UART Connection
      - Serial often done directly from microcontrollers via a UART (Universal Asynchronous Receiver/Transmitter)
      - Often 3.3v logic

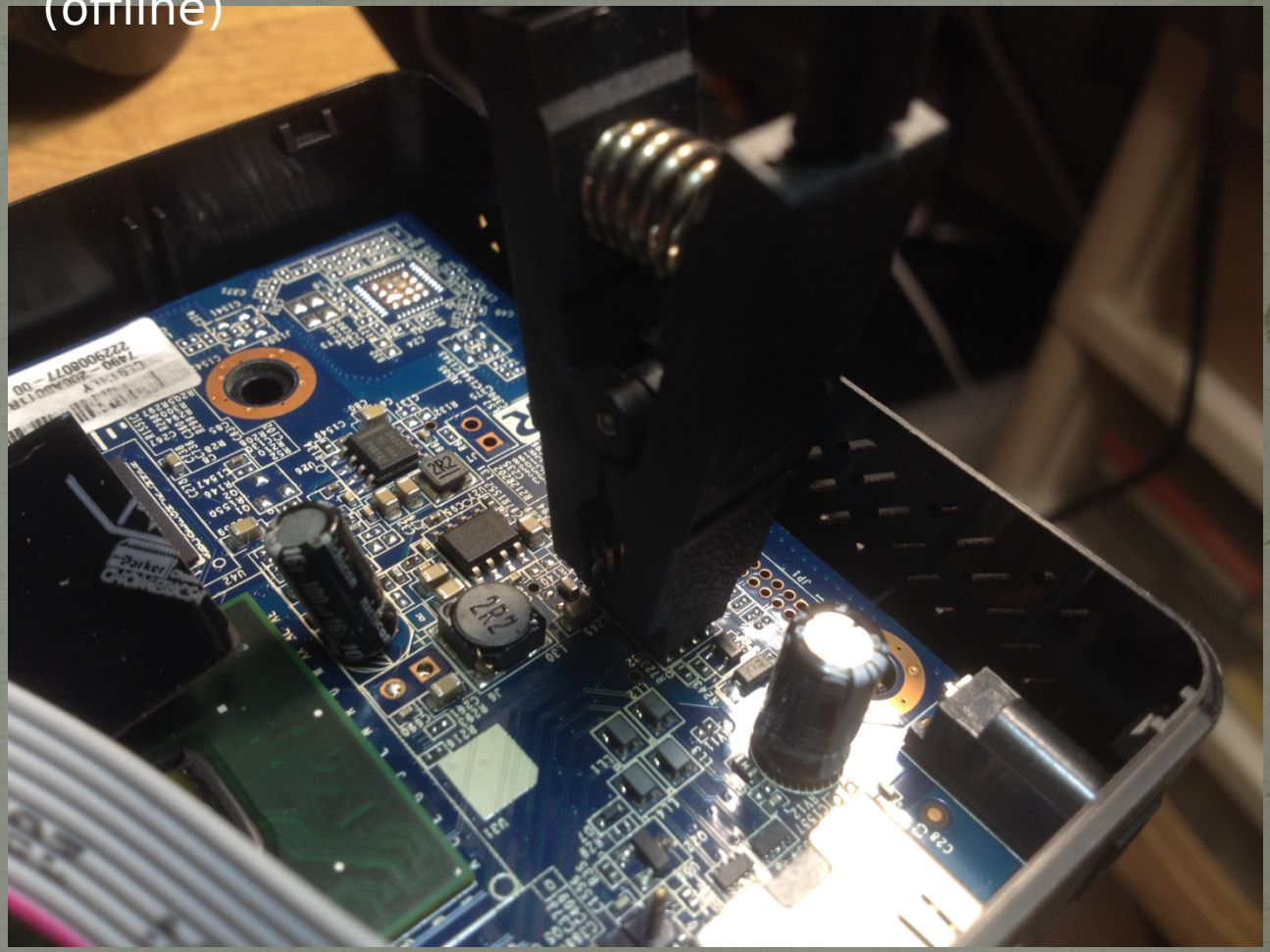
  - Non-interactive: (JTAG,SPI)
    - Extract/Program directly from/to onboard NVRAM

Extraction

Extraction



Retrieving firmware from an 8M SPI device (offline)



Data Sheets for almost device can be found online, describing electrical specifications and pinouts

Internal/External interfaces

- Explore/Exploit existing interfaces:
  - Fuzzing
  - Hidden menus
  - Non-standard input device
    - An Xbox controller is USB, can I hook up a USB keyboard, USB memory, keyboard?
  - Give it what you want it to have->TFTP/HTTP(s) configuration

- Many devices have internal hardware interfaces:
    - Serial (RS232)/UART
    - JTAG (Joint Test Action Group)
    - SPI (Serial Peripheral Interface)

  - Interactive use: (serial, JTAG) Terminal access, often with no authentication
    - Cisco console ports (RS232)
    - Often 5v logic
    - Onboard direct UART Connection
      - Serial often done directly from microcontrollers via a UART (Universal Asynchronous Receiver/Transmitter)
      - Often 3.3v logic

  - Non-interactive: (JTAG,SPI)
    - Extract/Program directly from/to onboard NVRAM

Firmware exploration

- What does firmware look like?
  - Some devices are ASICs
    - FAST!
    - Binary Low-level Firmware
    - Very specific/special purpose – done in hardware
    - Not dynamic in capabilities
      - Layer 2 switch
  - Many modern devices are SBCs
    - Much like a Desktop PC or Laptop
    - Various implementations of common operating systems:
      - Windows CE
      - Linux
- What does this have to do with firmware?
  - BIOS = firmware?
  - Embedded device OS is often a single file
  - Not just the “settings” for the OS, often IS the OS!

Firmware exploration

- Single File: *firmware.bin-> OS?*
  - *What is in it?*
    - Bootloader
    - Kernel
    - OS hierachy/files
  - Can be obfuscated (encrypted or packed with bytes/blocks swapped)
- Very often: CramFS, Squashfs, etc
  - Similar to tar, often with compression
  - Used to build a file of a filesystem
  - Very extensible (bz, bz2, lzma, etc)

Extraction

- Common methods:
  - Binwalk:
    - Utility to “unsquash” firmware
    - Generally extracts “known” objects
    - One way, but can be logged/verbose
  - Firmware Modification Kit (FMK)
    - Uses binwalk
    - Adds automated “re-squash”
    - Adds custom scripts for known obfuscation
    - Useful when attempting to change and reflash

Extraction Hints

- Binwalk or FMK didn’t work?
  - Not perfect
  - Sometimes requires “massaging”
    - File
    - Strings
    - Hexdump
    - dd
- Obfuscated
  - Using above utilities is often helpful!
- Encrypted
  - Sometimes keys exist “on device”
  - Usually not done