The Internet of Things

What is the “Internet of Things”?

- Non-computing devices…
- …with CPUs
- …and connectivity
- (Without connectivity, it’s a simple *embedded system*)
- Examples: thermostats, fitness sensors, home appliances, TVs, cars, light switches, toys, medical devices (including implanted ones), etc.

Embedded Systems

- Dedicated CPU for particular purpose
- Used since the late 1970s
- A CPU and some one-time programming is cheaper than random logic
- Used when modestly complex decisions are needed

Security Issues

- Embedded and IoT systems run programs
- Such programs can be (and frequently are) buggy
- Buggy code is often insecure code
- But—does the attacker have *access*?

Back to Our Threat Model

- What is the attacker's access?
  - Break into your house?
  - Send you malicious TV programs?
  - Plant a malicious radio transmitter near your device?
  - Hack into your device via the Internet?
  - Hack into some cloud server?
- Different devices have different risks—I'm not worried about someone breaking into my house to attack my coffee maker (though there have been reports of malware-infected e-cigarettes)

Attacks

- Default passwords
- Fake firmware
- Ordinary hacking
- Data-driven attacks

Why is This Different?

- Many people never change (or don’t know of) the password
- Most people don’t think about the security of, say, their TV
- There’s no antivirus software for IoT computers
- Much IoT software is never patched

The Patch Problem

- People don't know about installing patches
- Some devices aren't easy to patch—you need an administrative interface to tell it to find and install the patch
- Old models—software versions—aren't patched
- The lifespan of most gadgets is longer than the software support lifetime from the manufacturer
- There are *always* new models
- (I bought a new printer 6 months ago. There's now a replacement model that's 25% cheaper.)

Famous Holes

- VoIP phones (hacked here at CU)
- HP printers (hacked here at CU)
- Routers with (deliberate?) back doors
- Baby monitor cameras
- Car tire pressure monitors
- Many more—and the Internet of Things is just getting started...

IoT Architecture

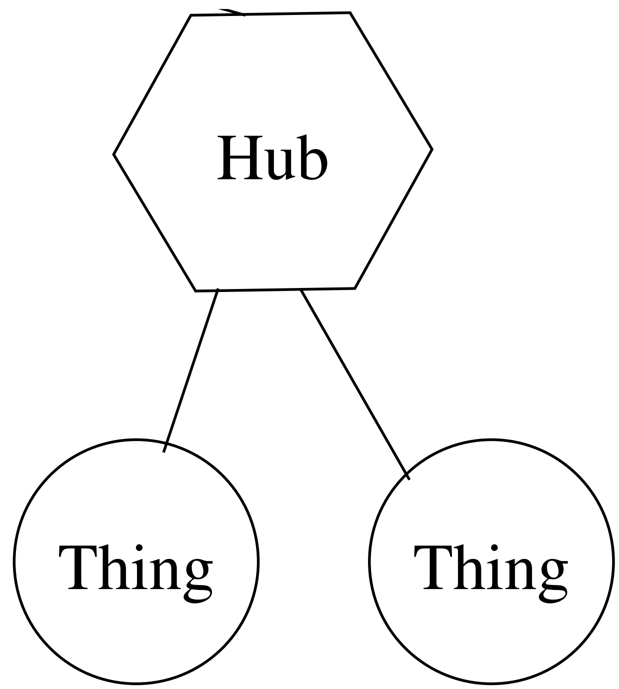
☞ Note: this is a plausible but *imaginary* architecture

☞ Most of these pieces exist, but may not be connected as I envision

- *Things* talk to *Hubs*
- Hubs talk to *Vendor Servers*
- *Managers* talk to Things via Hubs

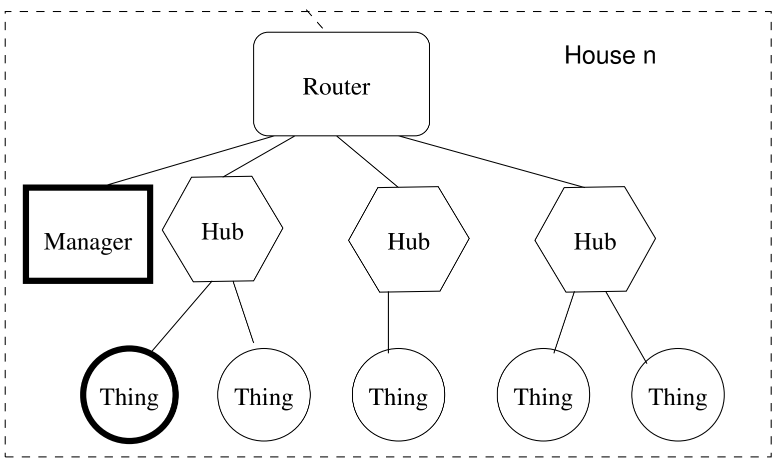
Hubs

- Many Things don't talk to the Internet directly
- Wrong interface, e.g., Bluetooth instead of WiFi
- Don't support full TCP/IP stacks (though they could)
- Don't implement security, user interface, etc.



Home IoT

- Local managers (which may be apps on a phone or computer) talk to hubs
- Hubs talk to devices via private protocols
- Hubs talk to vendor servers
- Things may talk to each other via hubs, but probably use the vendor servers
- Vendor servers *may* talk to each other



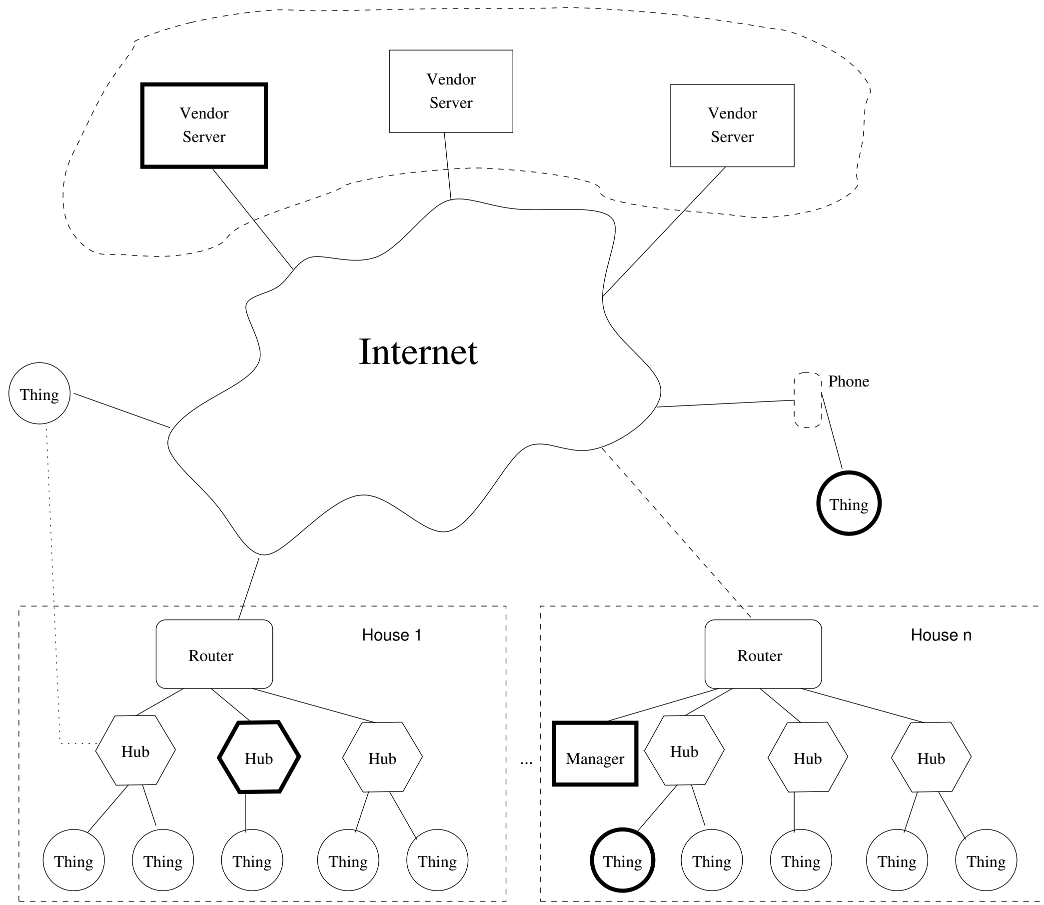
Why Have Vendor Servers?

- Many devices cannot be called directly (e.g., Nest thermostats), because of limited battery power: they're not always online
- NATs prevent direct inward calls from outside of the house
- Devices may not have enough CPU power for some tasks (e.g., voice recognition on Amazon Echo)
- Vendors like the service model—it forces users to keep coming back
- Vendors like to gather data (with all that implies for privacy)

(Privacy and the IoT)

- Some of the data captured is scary, e.g., on a Samsung smart TV
- Logs when, where, how long you use your TV
- Facial recognition camera (data nominally stays local)
- Voice command: “Please be aware that if your spoken words include personal or other sensitive information, that information will be among the data captured and transmitted to a third party”
- Hackers can retrieve all of that data from the TV
- (Source: `http://www.brennancenter.org/analysis/im-terrified-my-new-tv-why-im-scared-turn-thing`)

The Full Architecture



Any Component Can Be Attacked

- Things
- Hubs
- Servers
- Links?

Link Security

- It’s relatively easy to encrypt links
- It’s probably not that important, *if* people have secure home WiFi networks
- But—primary exposure from a compromised link is the devices at either end, including sending corrupted firmware
- Also: is there authentication data, such as passwords?

Ownership

- Who can control a Thing? Obviously, only the owner should be allowed to
- How does the Thing know who the owner is?
- What if the Thing is sold? What if the house is sold?
- Remember that Things have little or no user interface
- Solution is Thing-dependent, but will frequently rely on a physical “reset everything” request

Authentication

- Do *not* use passwords for Thing-to-Hub or Hub-to-Server authentication
- How do Things connected to different Hubs and different Servers authenticate each other?
- More precisely, how do they establish that they have the same owner?
- Complex cryptographic protocols may be necessary

Corrupted Things

- Consequences are Thing-dependent
- Obvious risk: private data collected by the Thing
- Possible risk: physical devices controlled by the Thing (this is a *cyber-physical system*)
- Possible risk: the human reaction to corrupted data
- Can attack Hubs, and possibly Servers via the Hubs

Corrupted Hubs

- Can attack Things or Servers
- Can attack other Hubs, Manager, or other home computers
- (The computers are already at considerable risk. . .)
- Can change access control rules

Access Control

- Users may have complex access control rules
- Who can change the thermostat? Within what limits?
- What about read access? (The Supreme Court has worried that "at what hour each night the lady of the house takes her daily sauna and bath" is private information.)
- Note: we know that setting access control rules properly is very hard

Corrupted Servers

- Biggest risk: corrupted firmware—but can be digitally signed
- Can send evil commands to Things
- Private data can be stolen from them
- Authentication data can be stolen

Defenses

- Request filters
- Cryptography
- Authorization
- Intrusion detection

Request Filters

- Things should incorporate sanity filters
- Example: Nest thermostats reject preposterous settings and will *always* activate if temperatures go outside certain ranges
- But—it isn't always possible to detect bad commands

Cryptography

- All messages should be cryptographically authenticated
- Encrypt messages to the extent possible—but don't cripple the IDS
- Avoid passwords—but Servers probably have to accept passwords, to allow direct logins from web browsers

Authorization

- Don't do authorization on the Servers—they're more exposed to attack
- Authorization is intimately tied to ownership—must be possible to buy and sell Things

Intrusion Detection

- There are many avenues for attack, and many components
- Use intrusion detection to detect ongoing attacks or other compromised components
- Example: alert the owner (via the Manager) if a sanity filter is used
- It might be a real, physical failure, e.g., the furnace has failed, so the house is too cold, or it might be an attack—but either way, the owner needs to know

How Are We Doing?

- The industry isn't doing a very good job
- Elementary cryptography isn't being used, or isn't being used properly
- Too much software is never updated
- Insufficient attention to threat model

Security Analysis: Internet Thermostats

- I recently decided to investigate Internet thermostats
- Control and monitor my house temperature remotely
- Are there security risks?

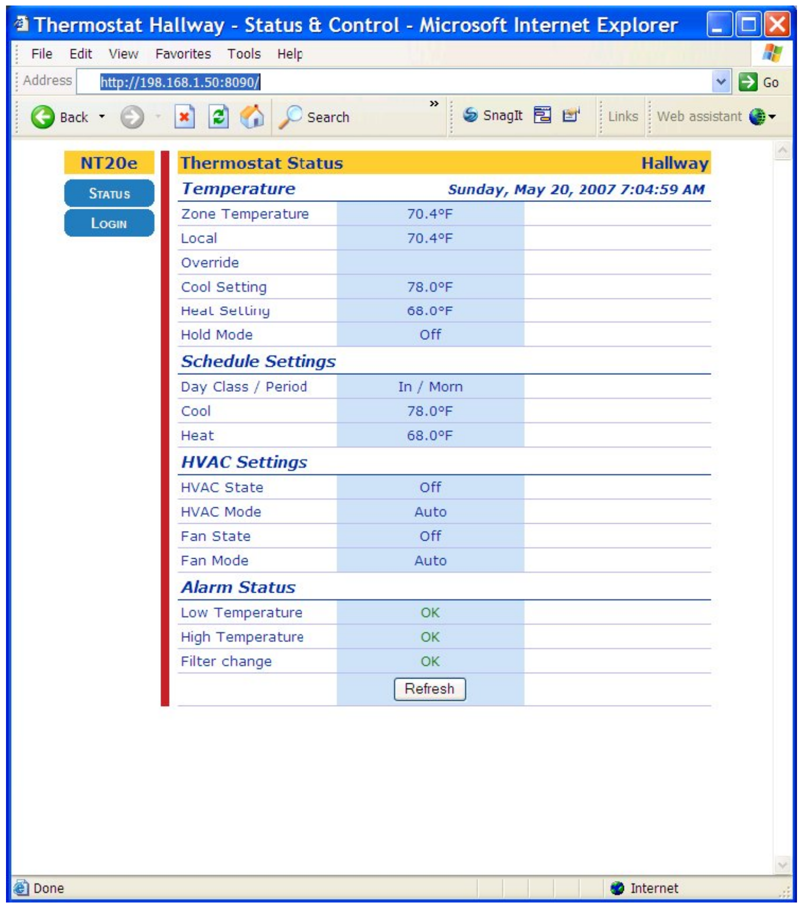
One Popular Brand

- Some thermostats have built-in web servers
- Simplest mode: direct connection to thermostat
- Alternate mode: thermostat and user connect to company's web site; company can generate alert emails
- Note: no hub for this brand

What's at Risk?

- Turning off someone's heat in the middle of winter?
- Turning on the heat in the summer?
- Run heat and air conditioning simultaneously?

Local Management



Local Problems

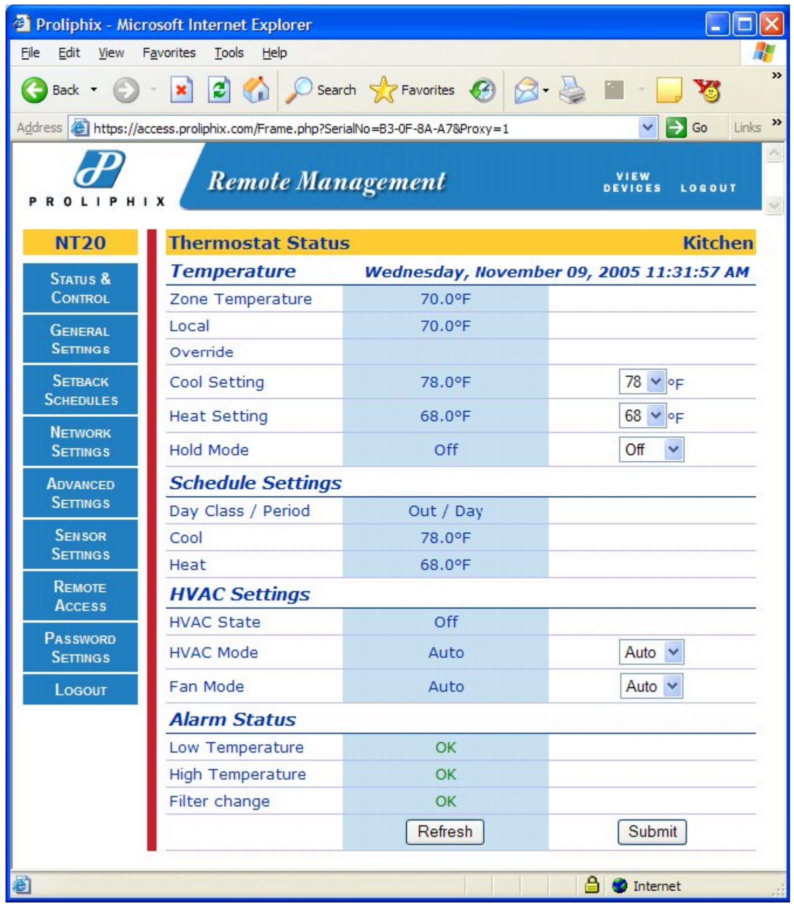
- No https — people can eavesdrop
- Uses "Basic Authentication":

  "The most serious flaw in Basic authentication is that it results in the essentially cleartext transmission of the user's password over the physical network.. . .

  "Because Basic authentication involves the cleartext transmission of passwords it SHOULD NOT be used (without enhancements) to protect sensitive or valuable information."

- No read-only mode

Remote Management



Remote Problems

- Https — but only to the server
- Unencrytped traffic from the server to the thermostats
- (The words “security” and “encyption” are not mentioned in the API manual. . .)
- Passwords are sent in the clear across the Internet
- Passwords are stored in bulk on the server

Privacy Issues

- Energy consumption patterns
- Al Gore's thermostat setting? Japanese office thermostat settings?
- Vacation schedules (burglary risk?)

Defenses

- Can’t touch thermostat software
- Add layering — access controls on top of built-in controls
- Use crypto tunnels
- Filter setting change requests

Last-Ditch Defenses

- Add a low-limit heat switch in parallel
- Add a high-limit heat switch in series
- These are hardware devices, not software
- Protect against bugs
- What if they fail?
- Independent failure modes; protect against each other

How to Analyze This?

- Hard to *know* all the threats
- Approach: see what is made available, and ask who might want it
- Reason by analogy and effect
- Check the "gold standard" (Au): **Au**thentication, **Au**thorization, **Au**dit