CHAPTER 1

The Internet of Things and Data

The Internet has enabled developers to create solutions that produce data that can be viewed by anyone anywhere in the world. Adapting prototypes or smaller versions of a solution to incorporate the Internet can be a challenge. It is not as simple as taking a working solution on a local network or similar communication mechanism and adding Internet connectivity. For example, growing your sensor network from a few sensors monitoring data viewed by a few people to a sensor network incorporating hundreds of sensors with the data viewable by potentially everyone may require redesigning your communication methods, data collection, and data storage.

Not only do you have to figure out how to scale the communication among your sensors, data collectors, and data-hosting services or database servers, you also have to deal with an explosion of data. That is, capturing data from a dozen sensors is relatively easy and may not require much in the way of careful planning to save the data, but capturing data from hundreds or even thousands of sensors is much more difficult because the data accumulates exponentially.

Clearly, storing the data on removable media or in a file is out of the question—especially so if you consider how the data will be used. Furthermore, making sense of all that data is complicated by how the data is stored or more appropriately retrieved. For example, what would happen if every device in your home, car, office, and so on, were to produce data? Add in the rising interest in wearable sensors and similar devices and you've got the potential to generate more data than any human can manage or even decipher.

However, it isn't just sensor networks that face a similar data crisis. Indeed, the greatest concern of innovators in the emerging and developing world of the Internet of Things (IOT) is the potential for a data explosion as more and more devices generate, communicate, and present data. What we need is a way to explore and exploit this data, and one place to start is how the data is gathered and stored on a smaller scale like a sensor network.

WHAT IS THE INTERNET OF THINGS?

The essence of the IOT is simply interconnected devices that generate and exchange data from observations, facts, and other data, making it available to anyone.[1] While there seems to be some marketing efforts attempting to make anything connected to the Internet an IOT solution or device (not unlike the shameless labeling of everything as "cloud"), IOT solutions are designed to make our knowledge of the world around us more timely and relevant by making it possible to get data about anything from anywhere at any time. Regardless, it is clear there is potential for the number of IOT devices to exceed the human population of the planet.

[1] https://en.wikipedia.org/wiki/Internet_of_Things.

This book will be your guide to successfully plan and implement the data storage component for your sensor network and similar IOT solution. Let's begin with a short discussion about IOT solutions and then look at the key challenges for developing IOT solutions focusing on the data.

IOT Solutions

An IOT solution is simply a set of devices designed to produce, consume, or present data about some event or series of events or observations. This can include devices that generate data such as a sensor, devices that combine data to deduce something, devices or services designed to tabulate and store the data, and devices or systems designed to present the data. Any or all of these may be connected to the Internet.

IOT solutions may include one or all of these qualities whether they are combined in a single device such as a web camera, use a sensor package and monitoring unit such as a weather station, or use a complex system of dedicated sensors, aggregators, data storage, and presentation such as a complete home automation system. Figure 1-1 shows a futuristic picture of all devices everywhere connected to the Internet via either databases, data collectors or integrators, display services, and even other devices.

Figure 1-1. *The future of IOT—all devices, everywhere*[2]

[2] https://pixabay.com/en/network-iot-internet-of-things-782707/.

IOT Is More Than Just Connected to the Internet

So if a device is connected to the Internet, does that make it an IOT solution? That depends on whom you ask. Some believe the answer is yes. However, others (such as myself) contend that the answer is not unless there is some benefit from doing so.

For example, if you connected your toaster to the Internet, what could be the benefit of doing so? It would be pointless (or at least extremely eccentric) to get a text on your phone from your toaster stating your toast is ready. So in this case, the answer is no. However, if you have people such as irresponsible teenagers or perhaps older adults whom you would like to monitor, it may be helpful to be able to check to see how often they use their toaster and when. That is, you can use the data to help you make decisions about their care and safety.

Allow me to illustrate with another example. I was fortunate to participate in a design workshop held on the Microsoft campus in the late 1990s. During our tour of the campus, we were introduced to the world's first Internet-enabled refrigerator (also called a *smart refrigerator*).[3] There were sensors in the shelves to detect the weight of food. It was suggested that, with a little ingenuity, someone could use the sensors to notify their grocer when their milk supply ran low, which would enable people to have their grocery shopping done not only online but also automatically.[4] This would have been great if you lived in a location where your grocer delivers but not very helpful for those of us who live in rural areas. While it wasn't touted as an IOT device (the term was coined later), many felt the device illustrated what could be possible if devices were connected to the Internet.

Thus, being connected to the Internet isn't what IOT is about nor is it a new concept. Rather, IOT solutions must be those things that provide some meaning—however small that benefit is to someone or some other device or service. Figure 1-2 depicts this a bit more clearly than Figure 1-1.

[3] https://en.wikipedia.org/wiki/Internet_refrigerator.

[4] Many businesses have automated reordering features built into their software. Most are triggered by a software or database event (such as low quantity), while sensors in the storage units trigger others.

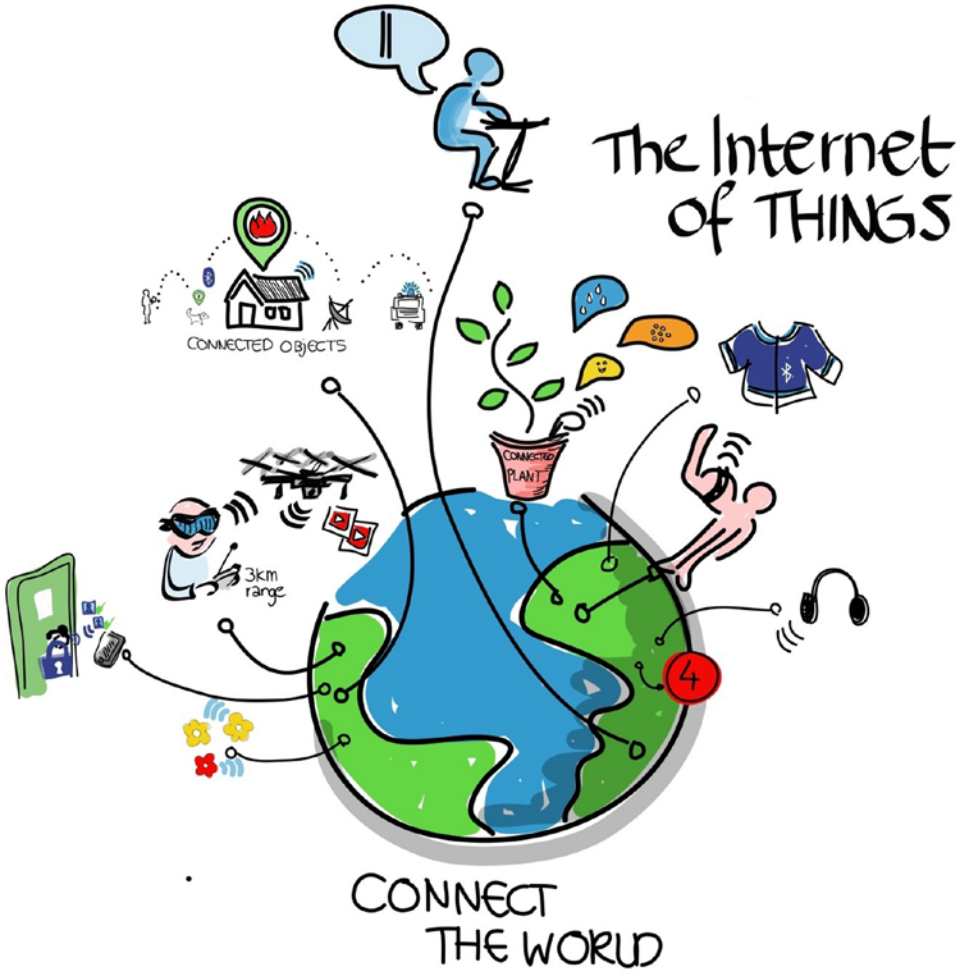

Figure 1-2. *"Connect the world" by Wilgengebroed on Flickr,[5] via Wikimedia Commons*

Notice here you see things connected to the Internet in logical groupings. Observe the connected plants. The drawing depicts several sensors, but each sensor isn't necessarily connected directly to the Internet. It is more likely and more practical to connect the sensors from one or more plant to an intermediate node that sends the data either to a service on the Internet or perhaps to another node in the network for later processing. Figure 1-3 shows how this would look in a logical form.

[5]CC BY 2.0 (http://creativecommons.org/licenses/by/2.0).

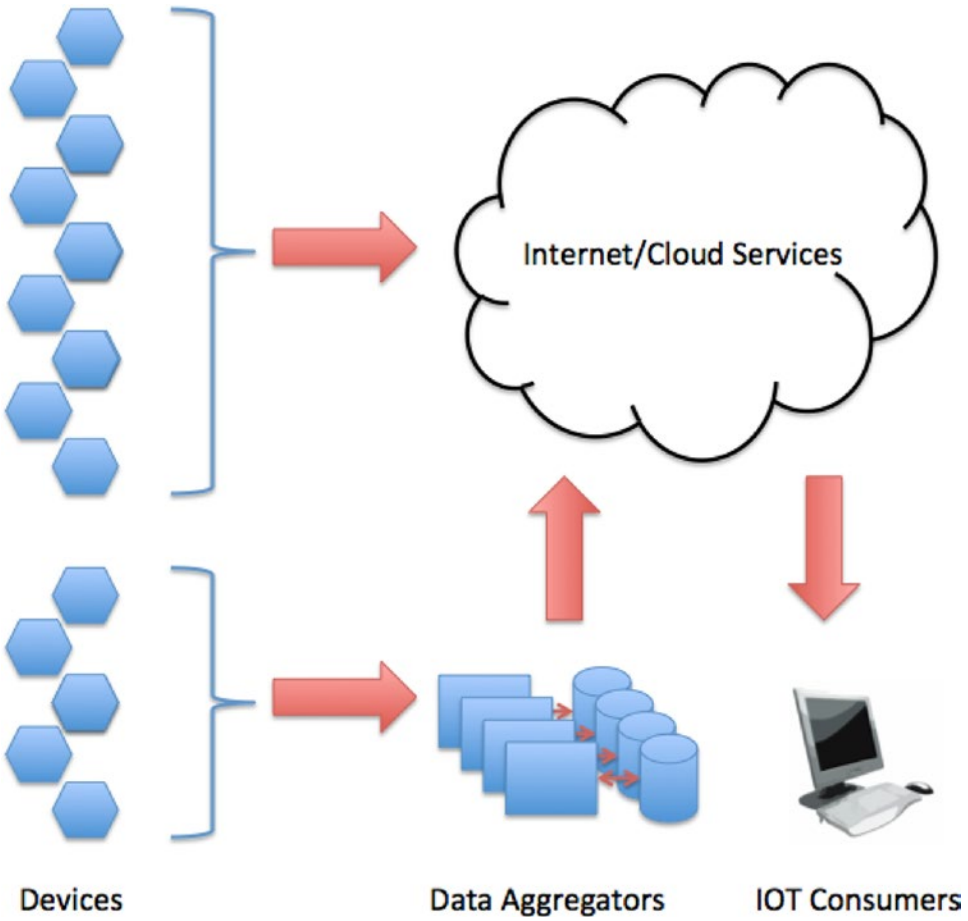

Figure 1-3. *How IOT devices connect to the Internet*

On the left side of Figure 1-3 are the IOT devices. These could be a simple sensor, an entire sensor network, a device with one or more sensors, an embedded microcontroller solution with sensors, a more sophisticated microprocessor-based solution, or even a device such as a microwave oven, alarm clock, or television. Some devices will connect directly to the Internet, as shown at the top of Figure 1-3. These devices are the more sophisticated devices with built-in networking capabilities. At the bottom are devices that will connect to intermediate nodes such as data aggregators, computers, and so on, that filter, augment, and store the data prior to either presenting it via an Internet connection or sending the data to cloud services. At the far right is a user accessing the data via the Internet or a cloud service. Naturally, this can be any type of device such as a laptop, desktop, tablet, phone, watch, or other smart device or appliance (including another IOT device).

IOT Services

Sadly, there are companies that tout having IOT products and services that are nothing more than marketing hype—much like what some companies have done by prepending "cloud" or appending "for the cloud" to the name. Fortunately, there are some really good products and services being built especially for IOT. These range from data storage and hosting to specialized hardware.

Indeed, businesses are adding IOT services to their product offerings faster than anyone can keep up. And it isn't the usual suspects such as the Internet giants. I have seen IOT solutions and services being offered by Cisco, AT&T, HP, and countless startups and smaller businesses. I use the term IOT *vendor* to describe those businesses that provide services for IOT solutions.

You may be wondering what these services and products are and why someone would consider using them. That is, what is an IOT service, and why would you decide to buy it? The biggest reason you may decide to buy a service concerns cost and time to market.

If your developers do not have the resources or expertise and obtaining them would require more than the cost of the service, it may be more economical to purchase the service. However, you should also consider any retooling necessary in the decision. I once encountered a well-meaning and well-documented contracted service that permitted a product to go to market sooner than projected at a massive savings. Sadly, while the champions of that contract won awards for technical achievement, they failed to consider that the systems had to be retooled to use the new service. More specifically, it took longer to adopt the new service than it would have to write one from scratch. So instead of saving money, the organization spent nearly triple and was late to market. Clearly, you must consider all factors.

Similarly, if your time is short or you have hard deadlines to make your solution production ready, it may be quicker to purchase an IOT service rather than create or adapt your own. This may require spending a bit more, but in this case, the motivation is time and not (necessarily) cost. Of course, in reality it is a mixture of both cost and time.

So, what are some of the IOT services available? The following are a few that have emerged in the past few years. It is likely more will be offered as IOT solutions and services mature.

- *Enterprise IOT data hosting and presentation*: Services that allow your users to develop enterprise IOT solutions such as connecting to, managing, and customizing data presentation in a friendly form such as graphs, charts, and so on. Example: Xively (`https://xively.com/`).
- *IOT data storage*: Services that permit you to store your IOT data and get simple reports. Example: Sparkfun's IOT Data service (`https://data.sparkfun.com/`).
- *Networking*: Services that provide networking and similar communication protocols or platforms for IOT. Most specialize in machine-to-machine (M2M) services. Example: AT&T's cellular global SIM service (`http://business.att.com/enterprise/Family/mobility-services/internet-of-things`).
- *IOT hardware platforms*: Vendors that permit you to rapidly develop and prototype IOT devices using a hardware platform and a host of supported modules and tools for building devices ranging from a simple component to a complete device. Example: Intel's IOT gateway development kits (`http://intel.com/content/www/us/en/embedded/solutions/iot-gateway/development-kits.html`).

HOW TO RAISE CAPITAL FOR DEVELOPMENT: KICKSTARTER

When developing new solutions, it can sometimes be a case of needing money to make money. However droll that sounds, most developers are not independently wealthy nor do they have the funds to sink into the tooling and production of mass production for their devices. Fortunately, the Internet provides a mechanism for developers to raise capital needed to take their ideas to market.

While there are several sites that offer similar services, Kickstarter (http://kickstarter.com) has provided a revolutionary way to raise funds. You can post your idea on the site and offer your product or services for a small donation. Indeed, most successful Kickstarter campaigns offer the contributor rewards commiserate with the donation. For example, for a small donation you may get a T-shirt or the chance to buy the product at a discount. For a larger donation, you may get one of the first production units, free upgrades to newer models, and even a stake in the profits.

Not only does Kickstarter allow developers to raise capital, it allows individuals to participate in funding a project at a much more modest monetary commitment than a typical venture capitalist. If you're interested, see the Kickstarter home page and browse the hundreds of available campaigns. You never know, you may find something you want to invest in.

To give you an idea of what is possible, take a look at the Kickstarter campaign for the Kossel Pro (http://kickstarter.com/projects/ttstam/openbeam-kossel-pro-a-new-type-of-3d-printer). The developers more than doubled their monetary goal, their product has become a reality, and orders are coming in fast and furious. I should know; I own one of the first production units!

Example IOT Solutions

Let's take a look at some example IOT solutions. The IOT solutions described in this section are a mix of solutions that should give you an idea of the ranges of sizes and complexities of IOT solutions. I also point out how some of these solutions leverage services from IOT vendors.

Sensor Networks

Sensor networks are one of the most common forms of IOT solution. Simply stated, sensor networks allow you to observe the world around you and make sense of it. Sensor networks could take the form of a pond-monitoring system that alerts you to water level, water purity (contamination), and water temperature; detects predators; or even turns on features automatically such as lighting or feeding the fish. If you or someone you know has spent any time in a medical facility, chances are a sensor network was employed to monitor body functions such as temperature, cardiac and respiratory, and even movement. Modern automobiles also contain sensor networks dedicated to monitoring the engine, climate, and even in some cars road conditions. For example, the lane-warning feature uses sensors (typically a camera and microprocessor and software) to detect when you drift too far toward lane or road demarcations.

Thus, sensor networks employ one or more sensors that take measurements (observations) about an event or state and communicate that data to another component or node in the network, which is then presented in some form or another for analysis. Let's take a look at an example of an intriguing alternative weather service.

Weather Underground (http://wunderground.com) is a community-driven site that permits amateur and professional weather enthusiasts around the world to connect their weather station to the Internet and share the data that their sensors provide. This means you can get the latest weather information for your region, city, and even locality simply by clicking the map and zooming in. You will see icons representing each local weather station (which displays local temperature) that you can click to see more information. Figure 1-4 shows Weather Underground's WunderMap that uses Google Maps to display the weather stations. As you can see, you can click any of the weather stations to see more information.

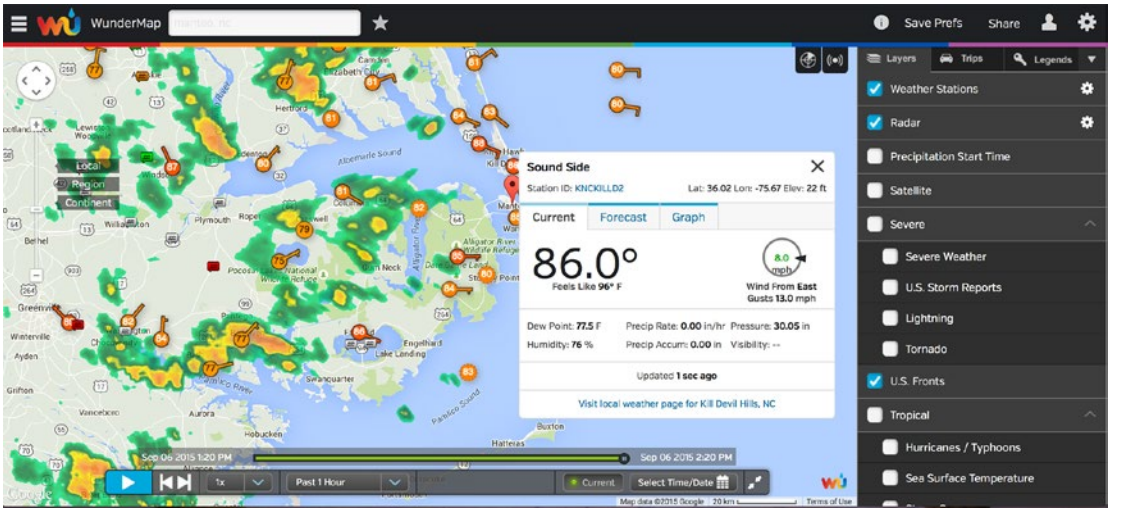

Figure 1-4. *Weather Station Network (courtesy of http://wunderground.com/wundermap)*

You can do far more than this with the WunderMap. As you can see, the map also shows radar data that you can use to see where precipitation may be occurring. You can also see the map in motion replaying data from previous updates from the radar and weather stations. Weather Underground is a really great site for those interested in weather—whether you have your own weather station or not, you can get a ton of information from the site.

This is perhaps one of the best examples I've found to illustrate the power of IOT and sensor networks in particular. Not only are you able to see the data generated by the sensors in your neighbor's weather station, you can also see the data from dozens more stations from around the area, the state, or even the country! This is the true power of IOT materialized.

Fleet Management

Another example of an IOT solution is a fleet management system (https://en.wikipedia.org/wiki/Fleet_management). While developed and deployed well before the coining of the phrase IOT, fleet management systems allow businesses to monitor their cars, trucks, ships—just about any mobile unit—not only to track their current whereabouts but also to use the location data (GPS coordinates taken over time) to plan more efficient routes, thereby reducing the cost of shipment.

Fleet management systems aren't just for routing. Indeed, fleet management systems allow businesses to monitor each unit to conduct diagnostics. For example, it is possible to know how much fuel is in each truck, when its last maintenance was performed (or more importantly when the next maintenance is due), and much more. The combination of vehicle geographic tracking and diagnostics is called *telematics*. Figure 1-5 illustrates a fleet management system.

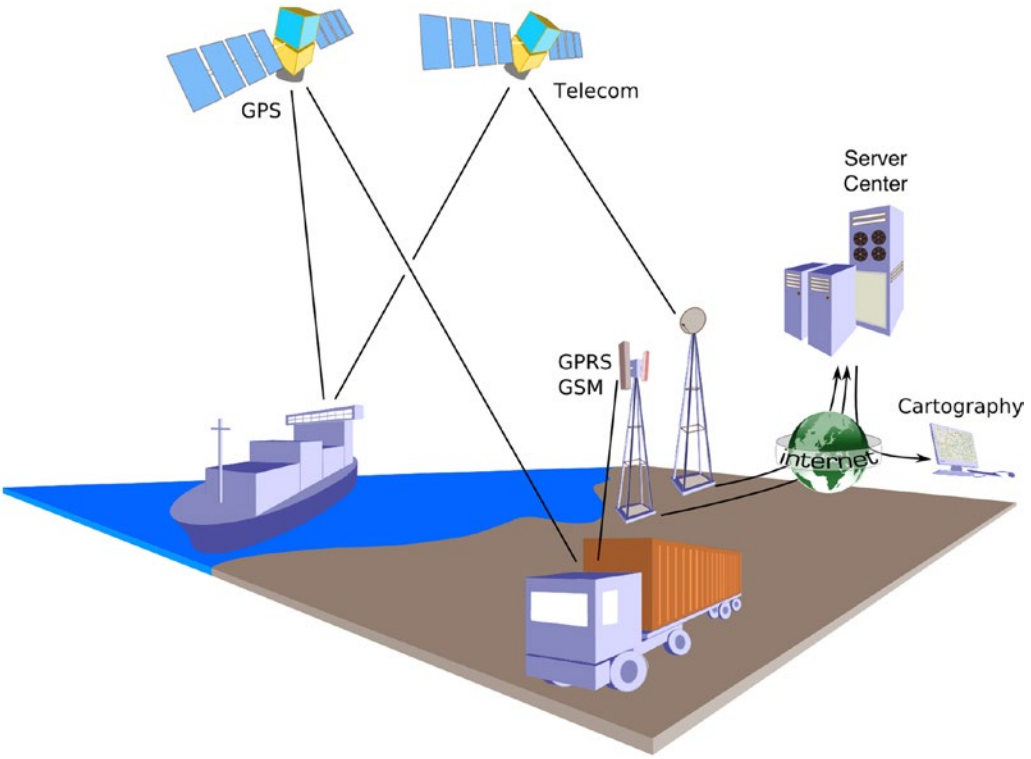

Figure 1-5. *Fleet management example*[6]

In this figure, you will see how GPS systems can track location as well as satellite communication to transmit additional data such as diagnostics, payload states, and more. All of these ultimately traverse the Internet, and the data becomes accessible by the business analysts.

You may think fleet management systems are only for large shipping companies, but with the proliferation of GPS modules and even the microcontroller market, anyone can create a fleet management system. That is, they don't cost millions of dollars to develop. For example, if you owned a bicycle delivery company, you could easily incorporate GPS modules with either cellular or wireless connectivity on each delivery person to track their location, average travel time, and more. More specifically, you can build a GPS tracking solution with an Arduino and a small supporting set of electronics.[7] In fact, I propose such a solution could be used to minimize delivery times by allowing packages to be handed off from one delivery person to another rather than having them return to the depot each time they complete a set of deliveries.

[6]CC BY 2.0 (http://creativecommons.org/licenses/by/2.0).
[7]See http://makezine.com/projects/make-37/gps-cat-tracker-2/ for information about how to build a small GPS tracker for pets. The technology would be the same or similar.

Home Automation

Another IOT solution that is becoming more prevalent (or at least more popular) is home automation[8] (also known as a *smart home*). While not new in any sense (home automation has been around for quite a while), home automation solutions have become more interesting now that many vendors are making them Internet ready. Indeed, most solutions provide either direct access from the Internet or access through cloud services.

A home automation solution is typically a set of sensors, actuators, and devices that allow you control things in your home. You can find sensors to detect movement, the state of windows and doors (open/close), temperature, humidity, and more. You can also find actuators such as locks that allow you to unlock or lock your doors remotely, open and close garage doors, and even turn lights on and off. Finally, you can find more sophisticated devices such as smart thermostats that can be programmed remotely, cameras that you can view and record images, robotic vacuum cleaners, and even phones that permit you to dial as if you were at home. While none of these is new, what is new is the packaging of these devices into an IOT solution.

For example, many home improvement stores such as Lowes and Home Depot carry their own line of home automation solution. You can find ready-to-install devices such as door locks, cameras, thermostats, and more that you can quickly and easily install and, with the aid of the included software, access remotely. There is even a device that allows you to turn your kitchen faucet on and off!

The Lowes solution, called Iris (`http://irisbylowes.com`), is a subscription service combined with a special network hub (called the Iris smart hub) to which all the devices connect. To access the data generated and features such as home security, pet monitoring, and more, you use an application (*app*) for either an Android or IOS device. The subscription has several levels ranging from a free basic service that allows you to connect to devices for basic services such as locking and unlocking doors, detecting movement, viewing short videos from cameras, and more. The paid service allows you much more access to devices and to control them remotely such as scheduling device power (turning lights on while you are away randomly to make it appear you are home) and more. Iris is sold as a starter kit with a few devices that you can expand as your budget or needs permit. See the Iris page for more information.

The Home Depot solution (`http://homedepot.com/c/Home_Automation_Basics`) also uses a hub (called Wink), but unlike the Lowe's solution, the devices you can add do not need to be from a single vendor. In fact, you can add devices that communicate via Wi-Fi, Bluetooth LE, Z-Wave, ZigBee, Lutron, ClearConnect, and Kidde. Thus, you can mix and match your devices with more freedom to grow your solution to your needs. Like the Lowe's solution, you can monitor all manner of things and view the data via an app called the Wink app. I should also add that some of the home automation devices available from Home Depot can operate without the hub, but the hub is required for remote access. Like Lowe's, you can purchase starter kits that are easy to set up and use. See the Home Depot Home Automation page for more details and links to the kits available.

The Lowe's and Home Depot solutions are just two of many available. A quick Google search will result in dozens more that you can choose—some that are proprietary like the Lowe's solution and others that are more open or can be expanded by devices from multiple vendors.

You can also build your own home automation solution using microcontrollers such as Arduino, Raspberry Pi, and BeagleBone. There are a number of books on the subject ranging from simple solutions to complex solutions. The following are just a few of the growing number of DIY home automation books.

- Steven Goodwin, *Smart Home Automation with Linux and Raspberry Pi* (Apress, 2013)
- Marco Schwartz, *Arduino Home Automation Projects* (Packt Publishing, 2014)
- Onur Dundar, *Home Automation with Intel Galileo* (Packt Publishing, 2015)

[8] `https://en.wikipedia.org/wiki/Home_automation`.

You can also find instructions for creating home automation solutions on the Internet. One example of a complex solution is the article (called an *instructable*) by Eric Tsai at http://instructables.com/id/Uber-Home-Automation-w-Arduino-Pi/. In his article, Eric describes a foundation for a home automation solution that you can build yourself from easy-to-obtain and easy-to-use components. Figure 1-6 shows an example of Eric's article that depicts what is possible with a little imagination.

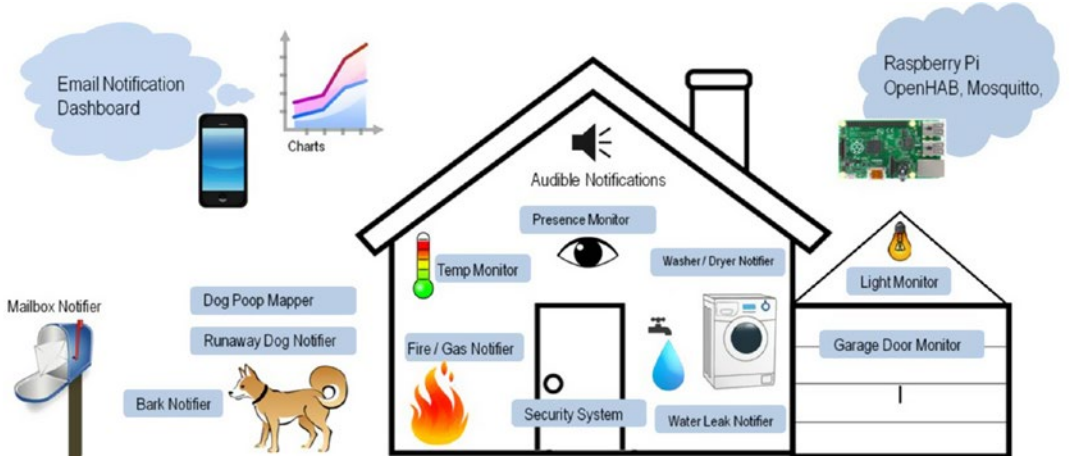

Figure 1-6. *DIY home automation (courtesy of Eric Tsai, http://etsai.net)*

As you can see, with a DIY home automation solution, you can build it however you like. You can add Arduino-controlled sensors that communicate via ZigBee modules, Raspberry Pi boards to monitor cameras, custom electronics to control garage doors, and more! In fact, if you aspire to be a maker or are an accomplished maker, chances are you can modify your existing devices to add remote capabilities. For example, garage door openers are easy to add a wireless module to in order to send signals over the local WiFi (or even the Internet). Even if you aren't an experienced software developer, you can use services such as Xively (http://xively.com/) to send your home automation data for monitoring. Of course, if you invest a little time in developing a simple web page, you can connect your home automation solution through your local network. While it may not be as fancy as the commercial units, you can make it do whatever you want.

Now that you know what IOT solutions are and have seen some examples, let's discuss what is arguably one of the most critical components of an IOT solution: data.

What Is IOT Data?

Whether your IOT solution is keeping watch on your breakfast burrito while you finish your shower or you're relying on instruments to pilot your boat to safety during a storm, the data produced and acted on is the most important artifact and indeed the lifeblood of the IOT solution.

Why? Because the solution is meaningless without the data. For example, if you had an IOT solution that monitored your body functions but never stored the data, the most you could achieve is an instantaneous reading. Without storing the data, you cannot perform any diagnostics from events in the past. Clearly, the data is important.

Perhaps equally important is how that data is stored and protected from exploitation or misuse. This is an especially important aspect of IOT solutions, which I will discuss in more detail later. For now, let's consider data for IOT solutions starting with a simple plant-monitoring system. Figure 1-7 shows a simple solution with a single plant. While a good solution would also include a light sensor to monitor how much light the plant is receiving, this one has been simplified so you can see the data produced.

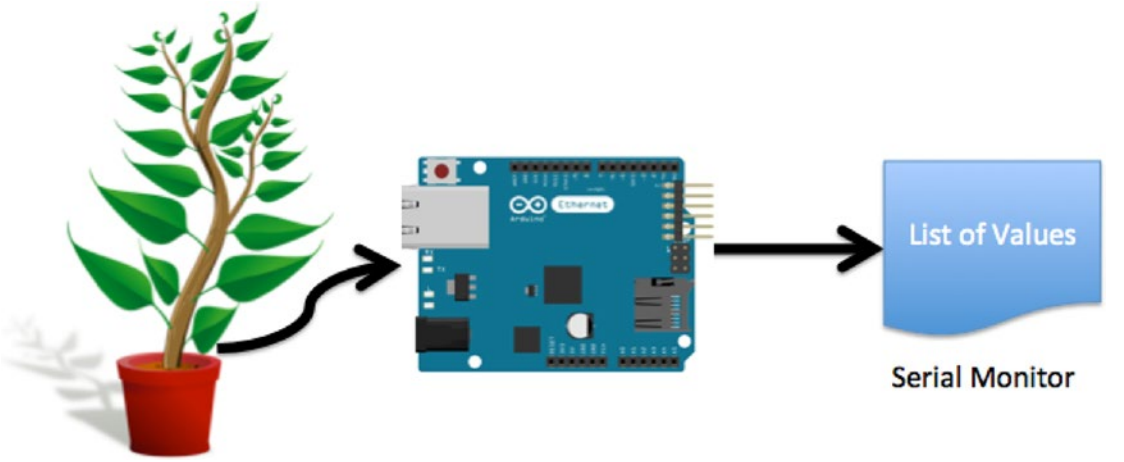

Figure 1-7. *Plant monitoring with Arduino*

In this example, you see an Arduino with a soil moisture sensor connected. The Arduino generates a list of readings from the soil moisture sensor that is placed in the soil of the plant. The code to do this is not complicated, as shown in Listing 1-1.

Listing 1-1. Plant Soil Monitoring

```
/*
 Simple Plant Monitor Example

 Display the value of the soil moisture sensor and threshold status.

*/

// Thresholds for sensor to detect wet/dry conditions. Adjust these
// to match your soil characteristics.
int upper = 400;
int lower = 250;

// Pin for reading the sensor
int sensorPin = A0;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(115200);
  while (!Serial);
```

```
  Serial.println("Simple plant monitor");
  Serial.println("raw value, moisture state");
}

void loop() {
  int value;

  // Read the sensor
  value = analogRead(sensorPin);

  Serial.print(value);
  Serial.print(",");

  // If value is less than lower threshold, soil is dry else if it
  // is greater than upper threshold, it is wet, else all is well.
  if (value <= lower) {
    Serial.print("dry");
  } else if (value >= upper) {
    Serial.print("wet");
  } else {
    Serial.print("ok");
  }
  Serial.println();
  delay(1000);
}
```

I kept this simple using the serial monitor as the output so that you can see the data generated. Listing 1-2 shows the output of a sample run where I watered the plant during the run. Notice in the listing of the output that the sensor correctly measured a change when I watered the plant (not a dramatic change in values). If you were building a more sophisticated (more useful) plant monitor, you would likely use some form of output such as an LCD panel or web server or you would store the data in a database.

Listing 1-2. Output of Soil-Monitoring Example

```
Simple plant monitor
raw value, moisture state
159,dry
217,dry
225,dry
224,dry
225,dry
225,dry
226,dry
248,dry
249,dry
256,ok
261,ok
279,ok
276,ok
254,ok
266,ok
295,ok
```

```
291,ok
302,ok
394,ok
467,wet
506,wet
419,wet
```

■ **Note** You can download the source code examples for this book from the Apress web site.

For purposes of this illustration, the text output is sufficient. This is because I want to call your attention to the rows of data. The first column is the raw value as read from the sensor. Clearly, this data has little human-readable information. After all, it's just a numeric value for the analog signal. That is why I used threshold values to determine or qualify the data. You can see this as the values change and the moisture level rises.

As you can see, the data that makes most sense to us is the "dry," "wet," and "ok" values. The raw values are not really that interesting. This is an excellent albeit simplistic illustration of how the raw data from a sensor needs additional augmentation to make it useful. However, I must also point out that if you stored only the derived values, if you need to adjust your thresholds, you cannot reevaluate the derived values. For example, if you determine the upper threshold needs to change and you want to do some analysis on how the change would have affected readings in the past, since you have only the "dry," "wet," and "ok" values, you cannot perform this analysis. Thus, saving the raw data is always a good practice.

■ **Tip** Always save the raw data as well as the calculated or derived values. You never know when you will need it.

If you are interested in building this example or perhaps embellishing it with an LCD or LED to warn when the plant needs water or better still to automatically water the plant with a servo or stepper motor and a water source, I encourage you to do so. It is a fun project. You can even modify the code to support a web server that you can use to remotely check the status of your plants.[9] The parts are easy to find and available from most online electronics stores such as SparkFun (http://sparkfun.com), Maker Shed (http://makershed.com), and AdaFruit (http://adafruit.com). I've included a couple of links to articles ordered from simple to complex that explain how to build this project and similar projects. Figure 1-8 shows how the sensor is wired to the Arduino.

- Sparkfun's soil moisture tutorial (https://learn.sparkfun.com/tutorials/soil-moisture-sensor-hookup-guide?_ga=1.98811421.2053037341.1391972341)
- Soil moisture with Grove components (http://seeedstudio.com/wiki/Grove_-_Moisture_Sensor)
- Make's potted plant protector (http://makezine.com/projects/potted-plant-protector/)

[9]Hint: See the web server example under the Ethernet category in the Arduino IDE.

- Self-watering plant (http://instructables.com/id/Self-Watering-Plant/)
- Adafruit's wireless garden tutorial (https://learn.adafruit.com/wireless-gardening-arduino-cc3000-wifi-modules)

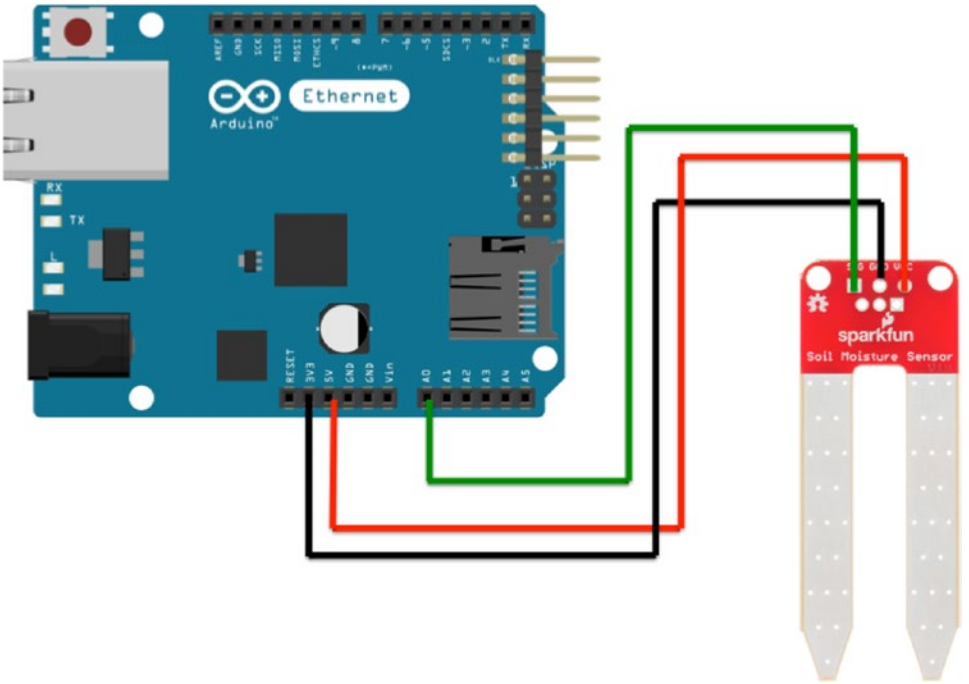

Figure 1-8. *Wiring the simple plant monitor*

There is another reason I wanted to show you this data. IOT solutions often employ intermediate nodes in the network. More specifically, it is often the case that a sensor is installed or connected to a much smaller set of electronics such as a microcontroller or even a simple integrated circuit. This node would then take the sensor value(s) and send them to another node elsewhere on the network. It is important to note that this level can use a networking protocol other than Ethernet or WiFi to simplify and reduce the need to have a unique address for each device. These devices typically do not have enough resources to support the more complicated networking protocols and thus require something more lightweight that can be achieved with limited resources.

When data is exchanged between nodes like this, it is called *machine-to-machine* data exchange and often transmitted in the raw form. There are several reasons for this. Most notable is to save memory and help speed communication, especially for small microcontrollers and similar embedded processors. That is, it is far faster to send a single integer or even a floating-point value than a formatted text string. This could be critical if the nodes at the sensor level use a communication protocol that operates with lower resources (memory) such as XBee modules.

With all this talk of the data in IOT solutions, one must wonder where it all goes and how this data will accumulate.

IOT Predictions: Data Overload?

You don't have to look far or long on the Internet to discover some dire predictions about the IOT.[10] Most of this stems from the proliferation of new devices and new ways to connect devices to the Internet. This presents two problems: the need for more address space (IP addresses) and addressability and the need for better ways to manage extremely large amounts of data (big data). Less popular but equally important is the security of IOT data, devices, and services. I address each of these in the following sections.

Addressing IOT Devices

Just because you can tack on an address to a random spot on the planet, it isn't necessarily easy to find. That is, if you mark a rock somewhere in the desert with a bright red "X" or simply write a number on it, no one is going to find it. Even if people spend the time to comb the desert, they will need a hint as to its general area to find it. Of course, you could add metadata such as latitude and longitude or even a GPS homing beacon, but the "X" you placed on the rock is clearly not enough information. This illustrates two issues for addressing IOT devices: having a way to uniquely identify the device and finding or addressing the device.

Are There Enough Addresses Available?

There is a legitimate concern that shortly the number of IOT devices will quickly exceed the maximum capacity of IP addresses available. Currently, the IPv6 protocol[11] allows for approximately 3.4×10^{38} addresses. That is 340 undecillion numbers! While that is a massive number, it is likely the usable range of public addresses will be considerably less but still in the undecillion level. This is great because some predict the number of IOT devices in the future to number in the billions or perhaps even hundreds, thousands, or millions of billions.

Even if the public addressable IPv6 addresses number half or even a fourth of the available IPv6 addresses, we won't run out any time soon. Indeed, it is possible (and some make claims that assure us) that IPv6 provides ample addressability for all conceivable IOT devices in the future, but having all of those devices addressed does not mean they will be easily found.

How Can You Find an IOT Device?

Assigning an IP address to an IOT device doesn't necessarily make it easy to find. Indeed, if every IOT device had its own IP address, all would be connected to the Internet and potentially each other, but how would you find the device you're looking for? If they are your devices or someone you know who is willing to share their address, you can find them by knowing the right information. But what if you wanted to know whether anyone else in your neighborhood had an outdoor camera? Suppose you needed to access their imagery to help solve a crime or identify the wanderings of a stray animal? Short of knocking on your neighbors' doors, there is no easy way to find these IOT cameras.

[10] I saw a bumper sticker once that read, "The Internet is full. Go play outside." While whimsical, the slogan contains a small grain of truth and a dollop of advice for the younger generations.
[11] https://en.wikipedia.org/wiki/IPv6.

A SIMPLE SEARCH IS NOT ENOUGH

It is not enough to merely search the available IPv6 addresses looking for an IOT device. Some have estimated, even with a modestly fast search engine, that it could take many years and perhaps even thousands of years to search and identify all IPv6 devices connected to the Internet. And that doesn't include the ones being added or removed daily. Clearly, we cannot simply add IPv6 capability to every electronic device or thing that moves (and some that don't) in our lives and expect to be able to access them without more knowledge of what they are, where they are, and what they do.

Like with the rock in the desert, you need more information. You need to know how that device is connected, what it does, how it is used, and what data it produces. Thus, you need some form of broker or service that tracks the device. Perhaps a more poignant question is, why do you need to know or access the device in the first place? Isn't it more likely that you would treat your neighbors' homes as services that you can request data? For example, if you could use Google Maps to find your neighborhood and click the homes around you to see what IOT data is publicly available, would that not be much more useful than trying to find a specific IOT device (camera)? Doesn't this sound familiar? It should. This is exactly how the WunderMap works in the Weather Underground site.

In this case, each home would be an IOT service provider generating data. Some data could be made public such as externally facing cameras, while other data may be private and require secure logins in order to access the devices in the home. Recall the home automation use case. Consider how convenient it would be to be able to check on your home remotely or perhaps grant permission to a babysitter to watch TV or use a device in the kitchen.

No matter what the IOT service is, the fact remains that it is unlikely that you would need to access an IOT device directly. The service can provide all the functionality needed (or permitted). Not only does this dramatically reduce the search problem, it also helps limit the number of publically addressable IOT devices. That is, devices "behind" or "inside" the IOT service do not need to be made public. Even more importantly, this allows you to firewall or secure your IOT devices in a more robust manner. Think of what it would mean to discover that your IOT camera could be accessed by anyone anywhere with a simple hack.

By placing the IOT devices behind an IOT service (or broker, application, and so on), you also permit the use of lower-resource communication protocols. That is, the IOT devices can be built from cheaper and more limited hardware. For example, you do not need a laptop to monitor a plant sensor. Furthermore, if you had a home automation solution that permitted you to connect a plant monitor that uses XBee protocols, you could build that plant sensor using much less hardware and therefore more cheaply.

Smaller or lower-resource hardware and communication protocols solve another issue with IOT devices and sensors in particular. Sensors are not generating data at a preset interval. While some sensors contain timer circuits or can generate a value only periodically, sensors are generally polled at certain intervals by another device (such as the Arduino, Raspberry Pi, and so on).

Furthermore, communication protocols such as XBee are not lossless mechanisms. Indeed, however seldom, data loss is likely and the protocols support complete loss of a packet. If your sensor is generating values 30 times a minute, does it really matter if you receive only 29 values instead of 30? Perhaps some solutions may require greater precision, but for sensors that monitor events, this is acceptable, and it fits the model for sensors quite well. It allows for the possibility that the value is not ready; the sensor has not changed its state, and so on, permitting a much broader range of sensor behaviors that you can support.

The solution I am describing here has been labeled in a number of ways, but the most correct and indeed the most profound proposed architecture for the IOT is called Chirp. Francis daCosta describes this architecture in his book *Rethinking the Internet of Thing*s (Apress, 2013).

Chirp is simply the name of a small packet of data that contains only the minimal information needed to convey the data to its destination. This fits the small overhead requirement for lighter protocols and also permits for occasional loss of a packet. At the lowest point in the IOT solution, devices use protocols that support chirps to be sent to nodes that can monitor the data and take appropriate action. For example, the node may be built to take 30 samples generated in a minute and average them over time (say every five minutes) to generate a smoother data sample. I've seen this technique employed for solutions that used less accurate sensors where spikes (high or low) beyond a threshold were not used, thus allowing for a smoother reading from the sensor at a slower rate. It is less data but potentially more accurate.

Thus, the IOT solution would be built with several layers. At the lowest layers, IOT devices are networked to nodes that collect or process the data, which are then connected to devices that provide or enable certain services. For example, an IOT camera could be connected to an intermediate node that takes commands from a node at a higher layer. Similarly, a sensor network could be employed and monitored using several data aggregation and data-processing nodes that send the data to a database, which is then accessed by nodes in the higher layers. Figure 1-9 shows an example of this concept.

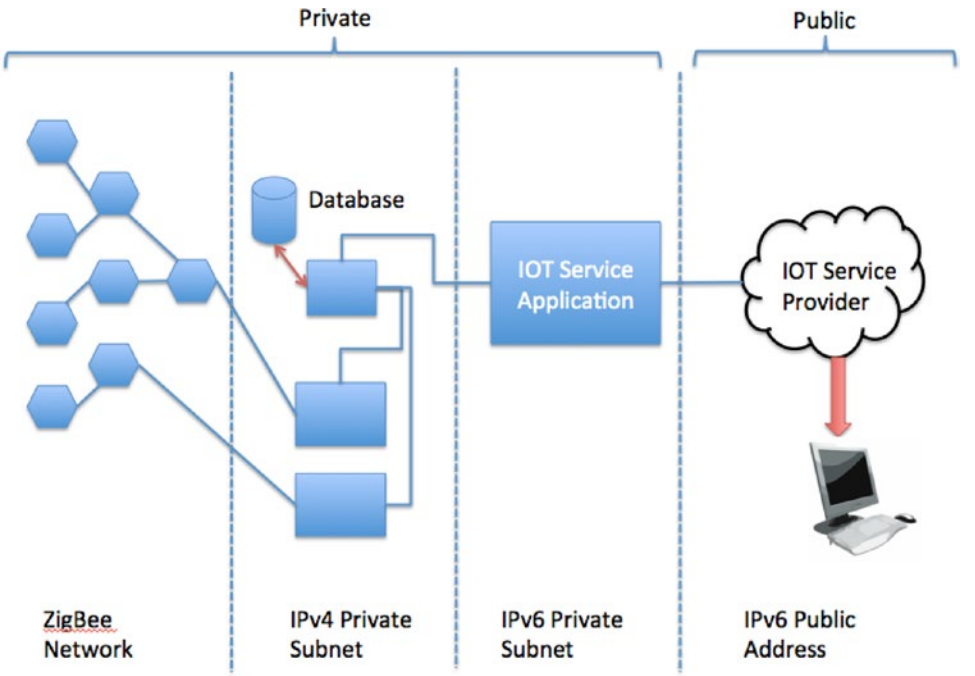

Figure 1-9. *Concept of layered IOT solution*

Notice at the top I've marked the layers behind the IOT service as private. This is to indicate that these layers and the devices within them are not directly accessible from the Internet. That doesn't cease to make them IOT devices; on the contrary, it helps define an architecture for layering the solution to permit optimization by using only the necessary protocols (and hardware needed for each layer). For instance, the sophistication of the devices, security practices, and programming increases from left to right in the drawing.

For example, on the left side of Figure 1-9 is a set of sensors or lightweight IOT devices connected using a ZigBee network. This could take the form of small devices employing XBee modules for sending data to the next layer, which contains the data aggregation and database nodes. At this layer, you could employ a private IPv4 Ethernet network. Why IPv4? Because most small devices like Arduino and Raspberry Pi support it

natively and by default. Thus, you can begin to connect these nodes with an application node that provides a gateway to the cloud-based services. At this point, you could continue to use IPv4 or switch to IPv6 since this is likely to be a more sophisticated server. On the right side of Figure 1-9 is a depiction of the IOT service (or services) you would then employ to grant access to your IOT solution. This could be anything from a front-facing web server to a complete IOT solution provided by commercial IOT vendors.

Clearly, an architecture like this permits the lower layers (those on the left) to communicate with lower-resource protocols and even deal with the occasional loss of data, while the higher layers (those on the right) communicate with more complex protocols. This also allows you to place the more complex parts of the solution on the layers or devices that are most appropriate for the task. For example, while you can connect your Arduino to a MySQL database server,[12] it is unlikely you would host it on the same platform (but you could on a Raspberry Pi) as other services. Finally, it permits you to design the appropriate level of security into each layer.

Now that you understand the addressing issue with IOT devices, let's talk for a moment about the data. With billions of IOT devices generating data, where is it all going, and how will you access it?

IOT and Big Data

Another concern IOT experts have is how quickly and how much the size of the data generated by IOT solutions will grow. That is, as more and more IOT devices are added and the data is archived, the size of that data will grow exponentially. As the data becomes sufficiently large, it ceases to be feasible to access it using traditional database access mechanisms. For example, accessing the sensor data for all the thermostats located in the United States could eventually become an absurdly large number (rows, bytes, and so on). Even if you had a reason to see this data, the amount of data would be overwhelming. If you narrowed it down by state, the data could still be more than is possible to search or for that matter retrieve.

WHAT IS BIG DATA?

Big data[13] refers to the relative size of data that will be processed, analyzed, viewed, or otherwise manipulated to draw conclusions (for example, data analysis, data warehousing, and so on). The relative size refers to the characteristic that big data exceeds the capacity of most computing platforms to contain or otherwise process the data in a reasonable amount of time. That is, it is more than a single system or even a complex system can handle.[14]

There are many approaches to handling big data, but most solutions use dozens to thousands of computing platforms to divide and conquer the problem. Two notable examples are Oracle's Big Data offerings (`http://oracle.com/big-data/index.html`) and MySQL + Hadoop (`http://mysql.com/why-mysql/white-papers/mysql-and-hadoop-guide-to-big-data-integration/`). While these solutions are differentiated based on customer or use cases, they both solve the problem (at a high level) of leveraging distributed database and execution to process the data.

[12] `https://github.com/ChuckBell/MySQL_Connector_Arduino`.
[13] `https://en.wikipedia.org/wiki/Big_data`.
[14] For example, it should not take 7.5 million years to get an answer of 42.

Thus, some pundits propose the IOT will produce data that is large and can be used only with big data solutions. For example, if you wanted to analyze patterns of use for a certain class of IOT devices across a geographical region (state, country), even though each IOT device may have its data hosted separately, there is a need to aggregate the data for compilation and processing. Thus, even if each device had only a small amount of data, aggregating the data over millions or even billions of IOT devices could generate a big data crisis.

Furthermore, while I and others propose a layered approach to IOT solutions and most will likely be built to host its own data, it is still probable at some point we will want to search and mine data across similar IOT solutions. While the thermostat example is a bit fictitious, it may be more likely that you need to research data from these solutions to develop patterns such as temperature fluctuations, fuel usage during peak weather months, or even evaporation rates of ponds and reservoirs to help predict water consumption and conservation rates. In this case, it is likely you would have to mine data from several repositories.

Thus, even if the data were compartmentalized so that there isn't a single database that holds everything,[15] the data you've mined will likely require temporary storage for analysis, making the data analysis a case of working with large (big) data. However it comes to pass, it is true that at some point acquiring, aggregating, and processing data from the IOT will require specialized big data solutions.

NOT EVERYONE AGREES

You may find it interesting that there are several schools of thought concerning big data. Some criticize its existence, others criticize the mechanisms we use to work with big data, and others insist the real solution is yet to be realized. Whatever the case, be advised that the landscape of big data is still evolving.

Fortunately, there are many vendors working on this problem, and the proliferation of cloud service providers ensures we will not have to create big data solutions ourselves. However, we will still have to deal with storing and making accessible our IOT solutions data—which is what this book is all about.

■ **Tip** If you'd like to know more about big data and how it relates to IOT, see Stackowiak, Licht, Mantha, and Nagode's book, *Big Data and the Internet of Things* (Apress, 2015).

Now that you understand that IOT data is no small matter and indeed will likely become a massive archive ripe for harvesting even more knowledge from the world around us, let's discuss the number-one concern beyond how to store the data—how to protect it and the IOT solutions from exploitation.

IOT Security

IOT developers also need to consider securing their devices, data, and services. Indeed, all solutions that use the Internet must develop better security practices. The unique aspects of an IOT solution make it especially difficult to plan and implement stringent security practices because of the multiple points of vulnerability. More specifically, each component may have different types of vulnerability, from physical access to sensors and IOT devices to remote attacks against the IOT services.

[15] Now that is absurd.

The recent rash of massive data breaches proves that security simply wasn't good enough. We've seen everything from outright theft to exploitation of the data stolen from well-known businesses like Target (more than 40 million credit card numbers may have been compromised) and government agencies like the United States Office of Personnel Management (more than 20 million Social Security numbers compromised). Interestingly, the source of the breach was traced back to third-party contractors and services. Clearly, no one is safe. We need a revolutionary step rather than refining the tried-and-true mechanisms.

Sadly, there is a limit to how far we can go in securing our solutions. As any information technology (IT) professional will tell you, applying the best, stringent password policies and tight security practices can force users to jeopardize the very strategy designed to protect them and their data. For example, consider password policies that require passwords to be 16 or more characters with at least four capital letters, six numerals, three special characters, and no English dictionary words; to expire every 60 days; and to not have more than seven characters in common with previous passwords.[16] In this situation, some users will be forced to write down their passwords because they cannot remember a random mixture of letters, capitals, numerals, and special characters.

However, making passwords harder to guess or crack is only one strategy. Indeed, there are various philosophies about how to secure systems properly. While an in-depth discussion of all techniques is beyond the scope of this book, it is important to consider one of the more popular philosophies we can use for our IOT solutions. It is the philosophy of using multiple components to identify individuals.

For example, a user may need to know a key phrase (password), possess an authorized tangible talisman (an RFID badge), and have on file a biometric signature (fingerprint). To gain access to one of the systems, the user would have to know the password, present a valid RFID badge, and have their fingerprint read and verified.

This may sound a little like science fiction or even super-secret spymaster work, but it ensures the access will be granted only to someone who meets all three components. That is, it may be possible to guess, crack, or simply steal a user's password, and it may be possible to acquire or even spoof an RFID badge; it even may be possible (however far-fetched) to copy someone's fingerprint. It is far more difficult to acquire all three components without compromising the identity of the user. However, the downside is the user cannot gain access unless she has all three components. While it is unlikely the user would lose their fingerprint (but injuries and skin conditions can make the reader fail), it is possible the user could lose or misplace their badge or forget their password. Thus, once again, the security practice is diminishing the user's experience and making it more difficult for the user to access the system.

So what do we do? Do we implement good practices to ensure the systems are not easily compromised, or do we risk lower security for ease of use? The bottom line is you must choose the security solution that best meets the need to protect the data and services without forcing the user to endure onerous practices and without making their lives difficult.

You may be wondering what this has to do with your Arduino-based sensor platform. After all, there isn't much someone can do to exploit an Arduino, is there? That depends on the Arduino and how it is connected. For example, a newer Arduino that supports common lightweight operating systems or that is connected to your network can be exploited. I won't expand on that, but suffice to say it is possible. The risk of exploitation increases with the sophistication of the IOT device. For example, in general, a Raspberry Pi may have more risk than a bare-bones sensor and XBee module. This is because the Raspberry Pi is capable of running Linux and therefore supporting all manner of hacking tools and utilities.

[16]No, really, this does happen. It is a perfect example of how good security practices can go wrong however well intended. That is, if the policy makes the users' lives so difficult that they must violate best security practices to cope, the policy has gone too far.

■ **Caution** Whichever security philosophy or strategy you employ for your user-accessible devices, you still must consider securing the rest of the nodes in the network.

But it isn't just software that can be exploited. For example, placing an IOT device in an enclosure outside your home that is connected via Ethernet is vulnerable to hackers who gain access to the Ethernet cable. Granted, someone would have to know the IOT device exists, but the risk of exploitation is real. To combat this, you can employ lighter-weight hardware and more simplistic communication protocols[17] that cannot be easily hacked.

But is security really a concern for well-designed IOT solutions? Let's look at a recent example. One of the biggest automotive brands in the United States (and the world), Jeep, has recently come under fire for vulnerability in its infotainment[18] solution. A group of highly skilled hackers was able to remotely access the system and hack into the other electronics in the car. The group was able to sound the horn, turn on the wipers, and even affect the handling and brakes. Worse, this all happened while Andy Greenberg, the author of the article "Hackers Remotely Kill a Jeep on the Highway: With Me in It,"[19] was at the wheel! No, this is not a myth. It actually happened, and Jeep has issued not one but two recalls for security patches to its systems. So what does this say about the future of IOT-enabled cars? You had better be certain security is not only built in but done very well. Clearly, Jeep has some more work to do.

WHY SECURITY?

You may be wondering why we are discussing security in a book dedicated to databases in IOT solutions. You may have heard "charity begins at home," which means we must teach our children the morals and ethics of taking care of others through generosity. For IOT solutions, there is analogy that applies to security. We must build security into our IOT solutions from the beginning. That is, we must design with the overarching goal of protecting the data and access to it from exploitation or theft. Every component must have security design goals, from simple sensors connected to innocuous, discrete communication electronics to sophisticated embedded microprocessors with full access to the Internet. For the purposes of this book, we will focus on security from the data collection point (for example, sensors and devices) to the database and all nodes in between. As you will see, a little security prevention can go a long way to safe guarding your data.

You may also consider security something that needs to be stronger for solutions that are higher risk for humans such as a nuclear power plant or a medical facility. While those are indeed solutions for which we would expect very good security, consider the case of home automation. What would happen if someone were able to hack into your smart home and be able to lock and unlock the doors? In fact, a recent popular baby monitor was found to be easy to hack, allowing hackers to view the images, listen in on conversations, and even manipulate the camera.

You may wonder how someone could use mundane data for nefarious activities. Consider a case where a family who owns a smart home decides to go on vacation. Let's also consider the family is security

[17]Not a true fix, but it certainly lowers risk.
[18]I utterly loathe portmanteau (https://en.wikipedia.org/wiki/Portmanteau). Why can't we just say "information and entertainment"?.
[19]http://wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

conscience and has not broadcasted their vacation plans via social media.[20] Let's also assume the only vulnerability hackers can find is a dump of the data from the smart meter on the home. So what? Well, consider that when the family goes on vacation, they use less electricity. Air-conditioning units may be set to higher (lower) temperatures to conserve power, no televisions will be on, no hot water will be used, no cooking is being done, and so on. Thus, the sudden drop in kilowatts used can tell thieves that the family isn't at home. Thus, even innocuous data can be exploited.

■ **Tip** There is no golden rule or silver bullet for security. Security practices must be constantly adjusted, new mechanisms need to be invented, and you must be generally proactive to keep ahead of those who would circumvent security. That said, you must take security seriously and develop your solution around solid best practices.

Now that I've scared you, let's talk a bit about security for IOT solutions starting with an overview of the most common security threats and how you can handle them. Again, we are examining these so we can prepare to build in security from the ground up in our IOT solutions.

Common Security Threats

Almost every aspect of an IOT solution is at risk for security. You've already seen how easy it would be for someone to exploit IOT devices. Even IOT devices that have security built in may not be sufficient. For example, a recent study from HP[21] showed 8 out of 10 devices failed to implement strong password requirements for access. Indeed, most used something as simple as "1234." As we've discussed, password security is just one area where security needs to be improved.

The report also concluded that, of the devices tested, 60 percent of those that had some form of user interface were vulnerable to attack, 70 percent used unencrypted network services, 80 percent failed to require passwords at all (even their cloud and mobile components), and 90 percent collected some form of personal information or data. With this in mind, the following sections discuss a few key areas we IOT developers need to consider when planning our IOT solutions.

Communication Protocols

The network or communication protocols used can be intercepted, especially if the data is transmitted using well-defined, formatted, clear-text chunks of data (called a *packet* in some protocols). It isn't all that difficult to sense the electrical current on a network cable or intercept a WiFi signal to determine what data is being exchanged. One way to combat this is to use encryption.

Data encryption, while somewhat ubiquitous, is a good way to protect your data. This is especially true if you use encryption that uses 128-bit algorithms and keys that are difficult to guess. Fortunately, encryption has been built into several forms of integrated circuits, making it possible to add it to small electronics. Indeed, you can buy a shield for an Arduino that has encryption functions (`http://sparkfun.com/products/13183`).

---

[20] You don't do this, do you? If you do, stop it! Post those photos after you get back, not while you're neck deep in sand 3,000 miles away.

[21] `www8.hp.com/h20195/V2/GetPDF.aspx/4AA5-4759ENW.pdf`.

Whether you use encryption or not, securing your communication protocols from direct access is another area where some solutions fail. That is, don't run your Ethernet cable outside of your home where it can easily be reached. If you must run Ethernet or similar cables, be sure to place them in conduit buried so that no one can accidentally discover them. If you cannot hide or secure the cabling, paint any exposed cabling the same color as the surrounding area to make it harder to find. A white cable against a white fence post is hard to see if you don't know it is there.

Privacy Policies

One security aspect that is often overlooked is the privacy policy for data collected and retained (sometimes called a *data retention policy*). If you are developing an IOT solution for yourself and storing data on your own database server, there may not be an issue. However, if you are using IOT cloud services, you may want to consider the privacy policy of the service. For example, if you use a service to store data for later access or analysis and decide to cancel the service, what does the company do with the data? Do they leave it where it is for anyone to stumble upon, or does the company delete it after your account expires?

The privacy policy may not be an issue for data that has little or no value and cannot be used against you. But for data such as your address, name, medical history, and so on, the privacy policy could be an issue. Thus, you should always check the privacy and data retention policies of all the services you plan to use.

Remote Maintenance

Companies that provide IOT devices and solutions that have embedded software often provide a means to update the firmware or software periodically. Indeed, it is important to consider solutions that provide this so that you can get the latest fixes and improvements. Not only do you get new features, but more importantly, you can get the latest security updates. For example, Jeep has patched its infotainment systems and provides patches (dealer installation required) to improve security.

However, the mechanism for how the patch or fix is transmitted and applied should be secure. For example, if the patch requires a special administration account, be sure the account is secured with a password that you set. In other words, don't use the factor default—ever. Furthermore, how the patch gets to your system is another concern. If you have to expose your solution to the Internet either to a machine or to a human, you may want to reconsider. Only use patch transmission mechanisms that are secure. That is, downloading the patch to a USB drive and then transferring it to the IOT system and applying it deliberately is more secure than allowing the IOT vendor to automatically update your system.

Password Policies

I discussed password policies previously. In review, be sure to use passwords on all accounts wherever possible and choose your passwords so that they are sufficiently complex enough (not 1234 or the name of your dog, street, or spouse) yet not so much you cannot remember them. Don't ever use default passwords, user accounts without passwords, or the same password for multiple accounts.

Physical Security

I mentioned this topic earlier too. For IOT solutions, this applies to all devices in the system. For those that must be physically outside of a secure area, be sure to make them as secure as possible by minimizing the hardware exposed and locking the enclosure. For example, a camera that sends video to the IOT server can be split so that only the camera portion is external to the building (for instance) and the communication electronics are inside the building. While it is possible for someone to hack the camera (or destroy it), it is less likely they can do anything more than intercept the signal.

Similarly, locking the IOT device in an enclosure can reduce the risk, but the risk is balanced against how sturdy the lock and enclosure is. That is, if the enclosure is made from a material that can be cut or the lock can be easily removed, the security measure only slows down the perpetrator. The determined will still prevail.[22]

Software and Firmware

Another area of concern is the firmware and software (operating systems) used in the IOT solution. We need to ensure the base operating system and other software is secure. More specifically, the software uses secured accounts, cannot be compromised remotely, uses encryption, and can be made hardened from attack. For example, a secure Linux operating system is preferred over an open access system. This also applies to any IOT services outside of the firewall (publically accessible) including web servers, IOT cloud services, and so on.

Now that you have seen some of the more common and more serious security risks, let's discuss security from the aspect of an IOT solution.

Securing IOT Solutions

Let's turn our attention to how we can employ security practices for IOT solutions. While this section is not complete in the sense it covers all possible security practices, it is intended to get you thinking about how your own IOT solution should be protected. But before we get to that, let's review a general architecture and nomenclature for IOT solutions. The following describes several types of nodes that you may use to build your IOT solution starting from the lowest layer (the IOT devices) to the highest layer (IOT cloud services). Keep in mind the layers are also ordered from simple to complex regarding capabilities. This happens to also correspond roughly with security at each level. That is, the lower layers are easier to secure (with some exceptions like physical access of external sensors) than the higher layers. Figure 1-10 shows how the nodes could be arranged in an IOT architecture.

[22]Which is kind of like windows. Sure, we all lock our windows, but a brick or reasonably sized stone will make short work of the glass.

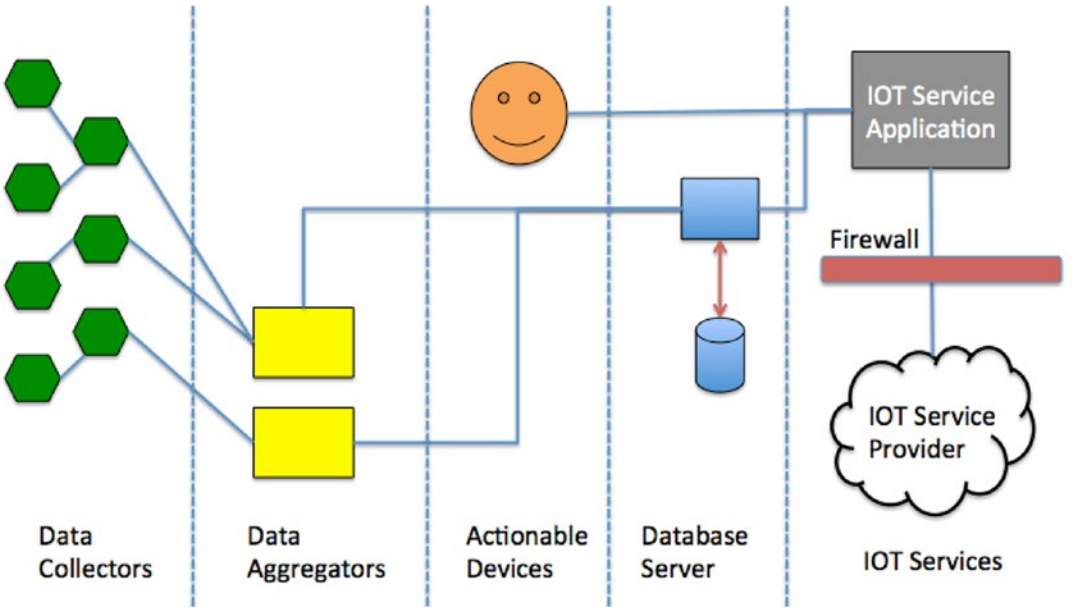

Figure 1-10. *IOT device architecture*

■ **Tip** I use these terms throughout the book.

- *Data collector*: A sensor, IOT device, and so on, that produces data from some event or observation.
- *Data aggregator*: A node (embedded controller, microcontroller, small computer, and so on) that receives information from one or more data collectors. Its purpose is to aggregate and augment the data for storage at the next layer.
- *Actionable device*: An IOT device that provides some user-controllable feature such as moving a sensor, operating locks, and so on.
- *Database server*: A node, typically a server that stores the data collected for later retrieval and analysis.
- *IOT services*: A computer system that provides an access layer to the database server and actionable devices. It may be systems located inside or outside the solution firewall. Systems are typically Internet servers or cloud services that allow users to view the data and manipulate the actionable devices.

Now let's discuss how to secure each of these layers.

Securing Data Collectors

The data collectors (or IOT devices) are those devices that have one or more sensors that produce data. These are often built with low-cost electronics that provide only the bare minimal capabilities for transmitting the data. I've already suggested using XBee modules to transmit the data via a simplistic protocol. I have also mentioned physically securing the device from tampering. In addition to these principles, you should consider making the data collectors from low-cost electronics, avoiding the temptation to use a more sophisticated small computer when a microcontroller or XBee module (for example) will do. If planned well, devices at this level do not need accounts or other login features.

Securing Data Aggregators

The data aggregator represents the next step in sophistication. Here we need more powerful electronics to manipulate the data. For example, we may want to qualify the data like we did in the plant-monitoring example earlier. Also, data aggregators are the first layer to start using more sophisticated communication protocols such as Ethernet or WiFi. Thus, you need to consider account access security (passwords) as well as remote access capabilities. For example, your data aggregators may support network access via remote logins. These need to be secured in the same manner as any other computing system on the network.

Securing Actionable Devices

Actionable devices can be a bit more challenging to secure. This is because unless you built it yourself, chances are the device is a commercially available device that has more features than you need. You should consider disabling any feature you do not need, ensuring any remote access is secured using as much as possible. Furthermore, I recommend placing the device behind the firewall or an IOT service such as a computer application or system that has a more secure access mechanism. For example, use an application that can be secured using encryption and highly secure remote access to send commands to the device. If you do this, not only do you make it more secure, you can also limit the features or actions available to the outside (Internet).

Securing the Database Server

The database server (if employed) should be secured from access like any other computer system. There are many texts (books, blogs, wikis, and so on) that cover this topic in great detail. I will present some of the best practices in a later chapter. In the interim, consider making your database server a single node in the solution secured from access outside the firewall, all accounts secured with passwords, and the data secured from local file access. In other words, harden your database server.

■ **Tip** You can find MySQL security best practices at `http://mysql.com/why-mysql/presentations/mysql-security-best-practices/`.

Securing IOT Services

This layer is the hardest to secure. If you purchase (lease) IOT services from a third party, security is built for you. It is up to you to ensure you use all possible, reasonable practices to secure your data in the service. For example, use good password policies. On the other hand, if you build your own Internet-facing services, you should treat the system with the most secure practices possible. That is, your server should not be accessible remotely without encryption, secure logins, and so on. If you plan to do this, you should consider becoming proficient at securing web services.

Summary

The Internet of Things is an exciting new world for everyone. Those of us young in heart but old enough to remember *The Jetsons* TV series recall seeing a taste of what is possible in make-believe land. Talking toasters, flying cars that spring from briefcases, and robots with attitude notwithstanding, television fantasy of decades ago is coming true. We have wristwatches that double as phones and video players, we can unlock our cars from around the world, we can find out whether our dog has gone outside, and we can even answer the door from across the city. All of this is possible and working today with the advent of the IOT.

In this chapter, you discovered what the IOT is and how IOT solutions are constructed, were introduced to some terminology to describe the architecture of IOT solutions, and saw some examples of well-known IOT solutions. We also discussed the two most critical issues of IOT solutions: big data and security through practical examples and discussion of the high points of the issues. You even saw a bit of source code along the way!

In the next chapter, you will see a number of hardware you can use to build IOT solutions. You will see devices for hosting or reading sensors and devices for collecting, augmenting, storing, and presenting data.