IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things

David Hanes, CCIE No. 3491

Gonzalo Salgueiro, CCIE No. 4541

Patrick Grossetete

Robert Barton, CCIE No. 6660, CCDE No. 2013:6

Jerome Henry, CCIE No. 24750

Cisco Press

800 East 96th Street

Indianapolis, Indiana 46240 USA

Chapter 7

Data and Analytics for IoT

In one of the famous episodes of the classic American science fiction TV series *Star Trek*, a harmless furry alien creature known as a "tribble" is brought aboard the starship *Enterprise*. At first, the cute little tribble is treated like a pet, but then its unusual property shows up: It is able to multiply itself at an alarming rate, to the point that the ship soon becomes so filled with tribbles that they consume all supplies on board and begin interfering with the ship's systems.

The problems of data generated by IoT networks might well resemble "The Trouble with Tribbles." At first, IoT data is just a curiosity, and it's even useful if handled correctly. However, given time, as more and more devices are added to IoT networks, the data generated by these systems becomes overwhelming. Not only does this data begin to consume precious network bandwidth but server resources are increasingly taxed in their attempt to process, sort, and analyze the data.

Traditional data management systems are simply unprepared for the demands of what has come to be known as "big data." As discussed throughout this book, the real value of IoT is not just in connecting things but rather in the data produced by those things, the new services you can enable via those connected things, and the business insights that the data can reveal. However, to be useful, the data needs to be handled in a way that is organized and controlled. Thus, a new approach to data analytics is needed for the Internet of Things.

This chapter provides an overview of the field of data analytics from an IoT perspective, including the following sections:

- **An Introduction to Data Analytics for IoT:** This section introduces the subject of analytics for IoT and discusses the differences between structured and unstructured data. It also discusses how analytics relates to IoT data.
- **Machine Learning:** Once you have the data, what do you do with it, and how can you gain business insights from it? This section delves into the major types of machine learning that are used to gain business insights from IoT data.

- **Big Data Analytics Tools and Technology:** *Big data* is one of the most commonly used terms in the world of IoT. This section examines some of the most common technologies used in big data today, including Hadoop, NoSQL, MapReduce, and MPP.
- **Edge Streaming Analytics:** IoT requires that data be processed and analyzed as close to the endpoint as possible, in real-time. This section explores how streaming analytics can be used for such processing and analysis.
- **Network Analytics:** The final section of this chapter investigates the concept of network flow analytics using Flexible NetFlow in IoT systems. NetFlow can help you better understand the function of the overall system and heighten security in an IoT network.

An Introduction to Data Analytics for IoT

In the world of IoT, the creation of massive amounts of data from sensors is common and one of the biggest challenges—not only from a transport perspective but also from a data management standpoint. A great example of the deluge of data that can be generated by IoT is found in the commercial aviation industry and the sensors that are deployed throughout an aircraft.

Modern jet engines are fitted with thousands of sensors that generate a whopping 10GB of data per second.[1] For example, modern jet engines, similar to the one shown in Figure 7-1, may be equipped with around 5000 sensors. Therefore, a twin engine commercial aircraft with these engines operating on average 8 hours a day will generate over 500 TB of data daily, and this is just the data from the engines! Aircraft today have thousands of other sensors connected to the airframe and other systems. In fact, a single wing of a modern jumbo jet is equipped with 10,000 sensors.

Figure 7-1 *Commercial Jet Engine*

The potential for a petabyte (PB) of data per day per commercial airplane is not far-fetched—and this is just for *one* airplane. Across the world, there are approximately 100,000 commercial flights per day. The amount of IoT data coming just from the commercial airline business is overwhelming.

This example is but one of many that highlight the big data problem that is being exacerbated by IoT. Analyzing this amount of data in the most efficient manner possible falls under the umbrella of data analytics. Data analytics must be able to offer actionable insights and knowledge from data, no matter the amount or style, in a timely manner, or the full benefits of IoT cannot be realized.

Note Another example regarding the amount of data being generated by IoT, and thus the need for data analytics, is the utility industry. Even moderately sized smart meter networks can provide over 1 billion data points each day. For more details about this data challenge, refer to Chapter 2, "IoT Network Architecture and Design."

Before diving deeper into data analytics, it is important to define a few key concepts related to data. For one thing, not all data is the same; it can be categorized and thus analyzed in different ways. Depending on how data is categorized, various data analytics tools and processing methods can be applied. Two important categorizations from an IoT perspective are whether the data is structured or unstructured and whether it is in motion or at rest.

Structured Versus Unstructured Data

Structured data and unstructured data are important classifications as they typically require different toolsets from a data analytics perspective. Figure 7-2 provides a high-level comparison of structured data and unstructured data.

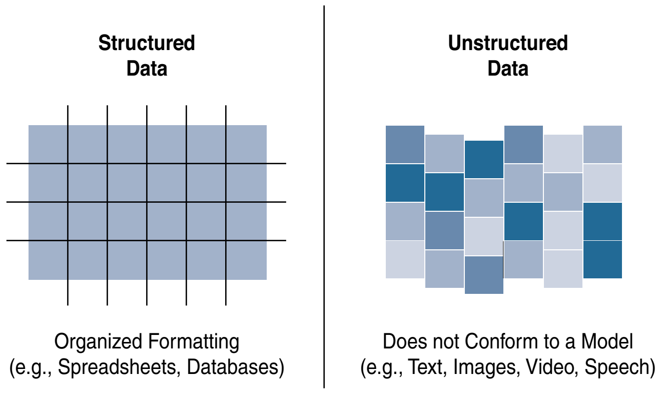

Structured Data
Organized Formatting (e.g., Spreadsheets, Databases)
Unstructured Data
Does not Conform to a Model (e.g., Text, Images, Video, Speech)

Figure 7-2 *Comparison Between Structured and Unstructured Data*

Structured data means that the data follows a model or schema that defines how the data is represented or organized, meaning it fits well with a traditional relational database management system (RDBMS). In many cases you will find structured data in a simple tabular form—for example, a spreadsheet where data occupies a specific cell and can be explicitly defined and referenced.

Structured data can be found in most computing systems and includes everything from banking transaction and invoices to computer log files and router configurations. IoT sensor data often uses structured values, such as temperature, pressure, humidity, and so on, which are all sent in a known format. Structured data is easily formatted, stored, queried, and processed; for these reasons, it has been the core type of data used for making business decisions.

Because of the highly organizational format of structured data, a wide array of data analytics tools are readily available for processing this type of data. From custom scripts to commercial software like Microsoft Excel and Tableau, most people are familiar and comfortable with working with structured data.

Unstructured data lacks a logical schema for understanding and decoding the data through traditional programming means. Examples of this data type include text, speech, images, and video. As a general rule, any data that does not fit neatly into a predefined data model is classified as unstructured data.

According to some estimates, around 80% of a business's data is unstructured.[2] Because of this fact, data analytics methods that can be applied to unstructured data, such as cognitive computing and machine learning, are deservedly garnering a lot of attention. With machine learning applications, such as natural language processing (NLP), you can decode speech. With image/facial recognition applications, you can extract critical information from still images and video. The handling of unstructured IoT data employing machine learning techniques is covered in more depth later in this chapter.

Note A third data classification, semi-structured data, is sometimes included along with structured and unstructured data. As you can probably guess, semi-structured data is a hybrid of structured and unstructured data and shares characteristics of both. While not relational, semi-structured data contains a certain schema and consistency. Email is a good example of semi-structured data as the fields are well defined but the content contained in the body field and attachments is unstructured. Other examples include JavaScript Object Notation (JSON) and Extensible Markup Language (XML), which are common data interchange formats used on the web and in some IoT data exchanges.

Smart objects in IoT networks generate both structured and unstructured data. Structured data is more easily managed and processed due to its well-defined organization. On the other hand, unstructured data can be harder to deal with and typically requires very different analytics tools for processing the data. Being familiar with both of these data classifications is important because knowing which data classification you are working with makes integrating with the appropriate data analytics solution much easier.

Data in Motion Versus Data at Rest

As in most networks, data in IoT networks is either in transit ("data in motion") or being held or stored ("data at rest"). Examples of data in motion include traditional client/server exchanges, such as web browsing and file transfers, and email. Data saved to a hard drive, storage array, or USB drive is data at rest.

From an IoT perspective, the data from smart objects is considered data in motion as it passes through the network en route to its final destination. This is often processed at the edge, using fog computing. When data is processed at the edge, it may be filtered and deleted or forwarded on for further processing and possible storage at a fog node or in the data center. Data does not come to rest at the edge. (For more information on edge and fog computing, refer to Chapter 2.)

When data arrives at the data center, it is possible to process it in real-time, just like at the edge, while it is still in motion. Tools with this sort of capability, such as Spark, Storm, and Flink, are relatively nascent compared to the tools for analyzing stored data. Later sections of this chapter provide more information on these real-time streaming analysis tools that are part of the Hadoop ecosystem.

Data at rest in IoT networks can be typically found in IoT brokers or in some sort of storage array at the data center. Myriad tools, especially tools for structured data in relational databases, are available from a data analytics perspective. The best known of these tools is Hadoop. Hadoop not only helps with data processing but also data storage. It is discussed in more detail later in this chapter.

IoT Data Analytics Overview

The true importance of IoT data from smart objects is realized only when the analysis of the data leads to actionable business intelligence and insights. Data analysis is typically broken down by the types of results that are produced. As shown in Figure 7-3, there are four types of data analysis results:

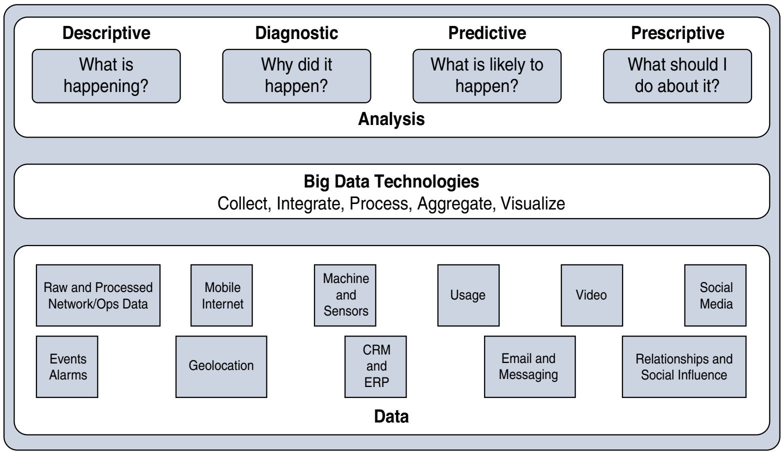


Figure 7-3 *Types of Data Analysis Results*

- **Descriptive:** Descriptive data analysis tells you what is happening, either now or in the past. For example, a thermometer in a truck engine reports temperature values every second. From a descriptive analysis perspective, you can pull this data at any moment to gain insight into the current operating condition of the truck engine. If the temperature value is too high, then there may be a cooling problem or the engine may be experiencing too much load.
- **Diagnostic:** When you are interested in the "why," diagnostic data analysis can provide the answer. Continuing with the example of the temperature sensor in the truck engine, you might wonder why the truck engine failed. Diagnostic analysis might show that the temperature of the engine was too high, and the engine overheated. Applying diagnostic analysis across the data generated by a wide range of smart objects can provide a clear picture of why a problem or an event occurred.
- **Predictive:** Predictive analysis aims to foretell problems or issues before they occur. For example, with historical values of temperatures for the truck engine, predictive analysis could provide an estimate on the remaining life of certain components in the engine. These components could then be proactively replaced before failure occurs. Or perhaps if temperature values of the truck engine start to rise slowly over time, this could indicate the need for an oil change or some other sort of engine cooling maintenance.
- **Prescriptive:** Prescriptive analysis goes a step beyond predictive and recommends solutions for upcoming problems. A prescriptive analysis of the temperature data from a truck engine might calculate various alternatives to cost-effectively maintain our truck. These calculations could range from the cost necessary for more frequent oil changes and cooling maintenance to installing new cooling equipment on the engine or upgrading to a lease on a model with a more powerful engine. Prescriptive analysis looks at a variety of factors and makes the appropriate recommendation.

Both predictive and prescriptive analyses are more resource intensive and increase complexity, but the value they provide is much greater than the value from descriptive and diagnostic analysis. Figure 7-4 illustrates the four data analysis types and how they rank as complexity and value increase. You can see that descriptive analysis is the least complex and at the same time offers the least value. On the other end, prescriptive analysis provides the most value but is the most complex to implement. Most data analysis in the IoT space relies on descriptive and diagnostic analysis, but a shift toward predictive and prescriptive analysis is understandably occurring for most businesses and organizations.

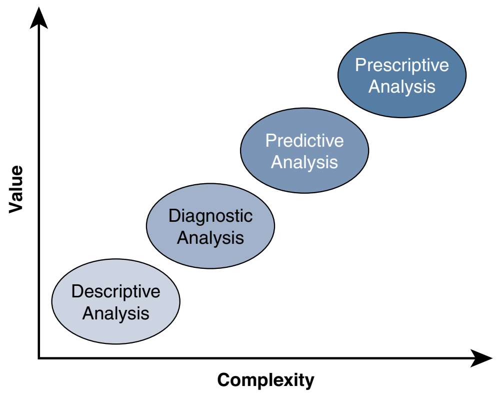


Figure 7-4 *Application of Value and Complexity Factors to the Types of Data Analysis*

IoT Data Analytics Challenges

As IoT has grown and evolved, it has become clear that traditional data analytics solutions were not always adequate. For example, traditional data analytics typically employs a standard RDBMS and corresponding tools, but the world of IoT is much more demanding. While relational databases are still used for certain data types and applications, they often struggle with the nature of IoT data. IoT data places two specific challenges on a relational database:

- **Scaling problems:** Due to the large number of smart objects in most IoT networks that continually send data, relational databases can grow incredibly large very quickly. This can result in performance issues that can be costly to resolve, often requiring more hardware and architecture changes.
- **Volatility of data:** With relational databases, it is critical that the schema be designed correctly from the beginning. Changing it later can slow or stop the database from operating. Due to the lack of flexibility, revisions to the schema must be kept at a minimum. IoT data, however, is volatile in the sense that the data model is likely to change and evolve over time. A dynamic schema is often required so that data model changes can be made daily or even hourly.

To deal with challenges like scaling and data volatility, a different type of database, known as NoSQL, is being used. Structured Query Language (SQL) is the computer language used to communicate with an RDBMS. As the name implies, a NoSQL database is a database that does not use SQL. It is not set up in the traditional tabular form of a relational database. NoSQL databases do not enforce a strict schema, and they support a complex, evolving data model. These databases are also inherently much more scalable. (For more information on NoSQL, see the section "NoSQL Databases" later in the chapter.)

In addition to the relational database challenges that IoT imposes, with its high volume of smart object data that frequently changes, IoT also brings challenges with the live streaming nature of its data and with managing data at the network level. Streaming data, which is generated as smart objects transmit data, is challenging because it is usually of a very high volume, and it is valuable only if it is possible to analyze and respond to it in real-time. Real-time analysis of streaming data allows you to detect patterns or anomalies that could indicate a problem or a situation that needs some kind of immediate response. To have a chance of affecting the outcome of this problem, you naturally must be able to filter and analyze the data while it is occurring, as close to the edge as possible.

The market for analyzing streaming data in real-time is growing fast. Major cloud analytics providers, such as Google, Microsoft, and IBM, have streaming analytics offerings, and various other applications can be used in house. (Edge streaming analytics is discussed in depth later in this chapter.)

Another challenge that IoT brings to analytics is in the area of network data, which is referred to as *network analytics*. With the large numbers of smart objects in IoT networks that are communicating and streaming data, it can be challenging to ensure that these data flows are effectively managed, monitored, and secure. Network analytics tools such as Flexible NetFlow and IPFIX provide the capability to detect irregular patterns or other problems in the flow of IoT data through a network. Network analytics, including both Flexible NetFlow and IPFIX, is covered in more detail later in this chapter.

Machine Learning

One of the core subjects in IoT is how to makes sense of the data that is generated. Because much of this data can appear incomprehensible to the naked eye, specialized tools and algorithms are needed to find the data relationships that will lead to new business insights. This brings us to the subject of machine learning (ML).

Machine learning, *deep learning*, *neural networks*, and *convolutional networks* are words you have probably heard in relation to big data and IoT. ML is indeed central to IoT. Data collected by smart objects needs to be analyzed, and intelligent actions need to be taken based on these analyses. Performing this kind of operation manually is almost impossible (or very, very slow and inefficient). Machines are needed to process information fast and react instantly when thresholds are met. For example, every time a new advance is made in the field of self-driving vehicles, abnormal pattern recognition in a crowd, or any other automated intelligent and machine-assisted decision system, ML is named as the tool that made the advance possible. But ML is not new. It was invented in the middle of the twentieth century and actually fell out of fashion in the 1980s. So what has happened in ML that makes it the new tool of choice for IoT and data analytics?

Machine Learning Overview

Machine learning is, in fact, part of a larger set of technologies commonly grouped under the term *artificial intelligence* (*AI*). This term used to make science fiction amateurs dream of biped robots and conscious machines, or of a *Matrix*-like world where machines would enslave humankind. In fact, AI includes any technology that allows a computing system to mimic human intelligence using any technique, from very advanced logic to basic "if-then-else" decision loops. Any computer that uses rules to make decisions belongs to this realm. A simple example is an app that can help you find your parked car. A GPS reading of your position at regular intervals calculates your speed. A basic threshold system determines whether you are driving (for example, "if speed > 20 mph or 30 kmh, then start calculating speed"). When you park and disconnect from the car Bluetooth system, the app simply records the location when the disconnection happens. This is where your car is parked. Beyond the appearance of artificial intelligence (the computer knows that you are parked and where this happened), the ruleset is very simple.

In more complex cases, static rules cannot be simply inserted into the program because they require parameters that can change or that are imperfectly understood. A typical example is a dictation program that runs on a computer. The program is configured to recognize the audio pattern of each word in a dictionary, but it does not know your voice's specifics—your accent, tone, speed, and so on. You need to record a set of predetermined sentences to help the tool match well-known words to the sounds you make when you say the words. This process is called machine learning. ML is concerned with any process where the computer needs to receive a set of data that is processed to help perform a task with more efficiency. ML is a vast field but can be simply divided in two main categories: supervised and unsupervised learning.

Supervised Learning

In supervised learning, the machine is trained with input for which there is a known correct answer. For example, suppose that you are training a system to recognize when there is a human in a mine tunnel. A sensor equipped with a basic camera can capture shapes and return them to a computing system that is responsible for determining whether the shape is a human or something else (such as a vehicle, a pile of ore, a rock, a piece of wood, and so on.). With supervised learning techniques, hundreds or thousands of images are fed into the machine, and each image is labeled (human or nonhuman in this case). This is called the *training set*. An algorithm is used to determine common parameters and common differences between the images. The comparison is usually done at the scale of the entire image, or pixel by pixel. Images are resized to have the same characteristics (resolution, color depth, position of the central figure, and so on), and each point is analyzed. Human images have certain types of shapes and pixels in certain locations (which correspond to the position of the face, legs, mouth, and so on). Each new image is compared to the set of known "good images," and a deviation is calculated to determine how different the new image is from the average human image and, therefore, the probability that what is shown is a human figure. This process is called *classification*.

After training, the machine should be able to recognize human shapes. Before real field deployments, the machine is usually tested with unlabeled pictures—this is called the validation or the test set, depending on the ML system used—to verify that the recognition level is at acceptable thresholds. If the machine does not reach the level of success expected, more training is needed.

In other cases, the learning process is not about classifying in two or more categories but about finding a correct value. For example, the speed of the flow of oil in a pipe is a function of the size of the pipe, the viscosity of the oil, pressure, and a few other factors. When you train the machine with measured values, the machine can predict the speed of the flow for a new, and unmeasured, viscosity. This process is called *regression*; regression predicts numeric values, whereas classification predicts categories.

Unsupervised Learning

In some cases, supervised learning is not the best method for a machine to help with a human decision. Suppose that you are processing IoT data from a factory manufacturing small engines. You know that about 0.1% of the produced engines on average need adjustments to prevent later defects, and your task is to identify them before they get mounted into machines and shipped away from the factory. With hundreds of parts, it may be very difficult to detect the potential defects, and it is almost impossible to train a machine to recognize issues that may not be visible. However, you can test each engine and record multiple parameters, such as sound, pressure, temperature of key parts, and so on. Once data is recorded, you can graph these elements in relation to one another (for example, temperature as a function of pressure, sound versus rotating speed over time). You can then input this data into a computer and use mathematical functions to find groups. For example, you may decide to group the engines by the sound they make at a given temperature. A standard function to operate this grouping, *K-means clustering*, finds the mean values for a group of engines (for example, mean value for temperature, mean frequency for sound). Grouping the engines this way can quickly reveal several types of engines that all belong to the same category (for example, small engine of chainsaw type, medium engine of lawnmower type). All engines of the same type produce sounds and temperatures in the same range as the other members of the same group.

There will occasionally be an engine in the group that displays unusual characteristics (slightly out of expected temperature or sound range). This is the engine that you send for manual evaluation. The computing process associated with this determination is called *unsupervised learning*. This type of learning is unsupervised because there is not a "good" or "bad" answer known in advance. It is the variation from a group behavior that allows the computer to learn that something is different. The example of engines is, of course, very simple. In most cases, parameters are multidimensional. In other words, hundreds or thousands of parameters are computed, and small cumulated deviations in multiple dimensions are used to identify the exception. Figure 7-5 shows an example of such grouping and deviation identification logic. Three parameters are graphed (components 1, 2, and 3), and four distinct groups (clusters) are found. You can see some points that are far from the respective groups. Individual devices that display such "out of cluster" characteristics should be examined more closely individually.

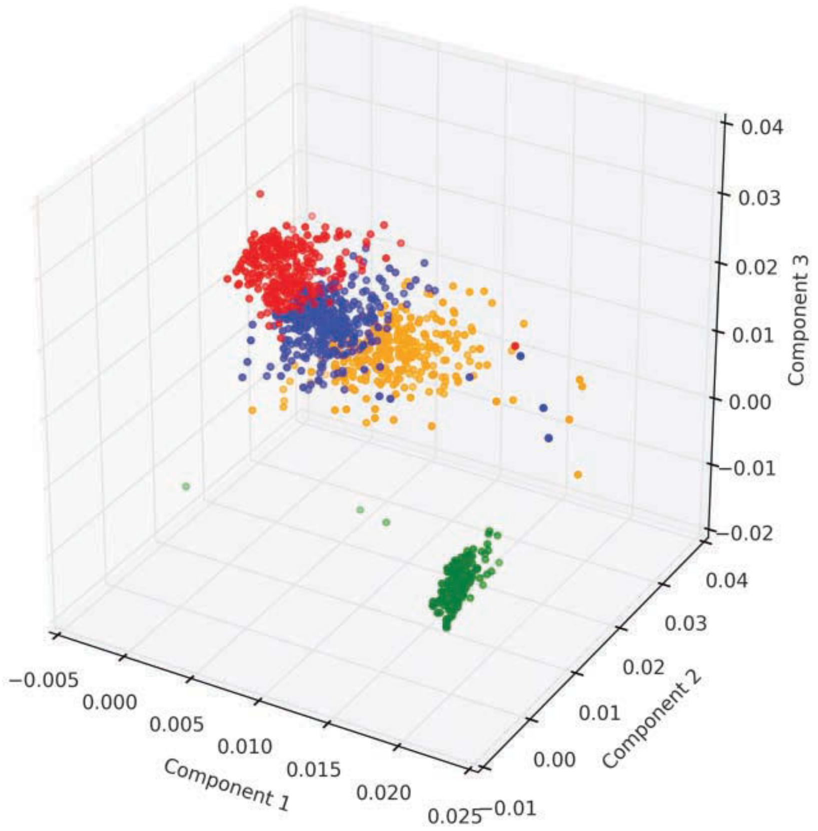


Figure 7-5 *Clustering and Deviation Detection Example*

Neural Networks

Processing multiple dimensions requires a lot of computing power. It is also difficult to determine what parameters to input and what combined variations should raise red flags. Similarly, supervised learning is efficient only with a large training set; larger training sets usually lead to higher accuracy in the prediction. This requirement is partly what made ML fade away somewhat in the 1980s and 1990s. Training the machines was often deemed too expensive and complicated.

Since the 2000s, cheaper computing power along with access to very large data sets (shared over the Internet) rejuvenated the possibilities of ML. At the same time, immense progress has been made in the efficiency of the algorithms used. Take the case of the human shape recognition for mining operations. Distinguishing between a human and a car is easy. The computer can recognize that humans have distinct shapes (such as legs or arms) and that vehicles do not. Distinguishing a human from another mammal is much more difficult (although nonhuman mammals are not common occurrences in mines). The same goes for telling the difference between a pickup truck and a van. You can tell when you see one, but training a machine to differentiate them requires more than basic shape recognition.

This is where neural networks come into the picture. Neural networks are ML methods that mimic the way the human brain works. When you look at a human figure, multiple zones of your brain are activated to recognize colors, movements, facial expressions, and so on. Your brain combines these elements to conclude that the shape you are seeing is human. Neural networks mimic the same logic. The information goes through different algorithms (called *units*), each of which is in charge of processing an aspect of the information. The resulting value of one unit computation can be used directly or fed into another unit for further processing to occur. In this case, the neural network is said to have several layers. For example, a neural network processing human image recognition may have two units in a first layer that determines whether the image has straight lines and sharp angles—because vehicles commonly have straight lines and sharp angles, and human figures do not. If the image passes the first layer successfully (because there are no or only a small percentage of sharp angles and straight lines), a second layer may look for different features (presence of face, arms, and so on), and then a third layer might compare the image to images of various animals and conclude that the shape is a human (or not). The great efficiency of neural networks is that each unit processes a simple test, and therefore computation is quite fast. This model is demonstrated in Figure 7-6.

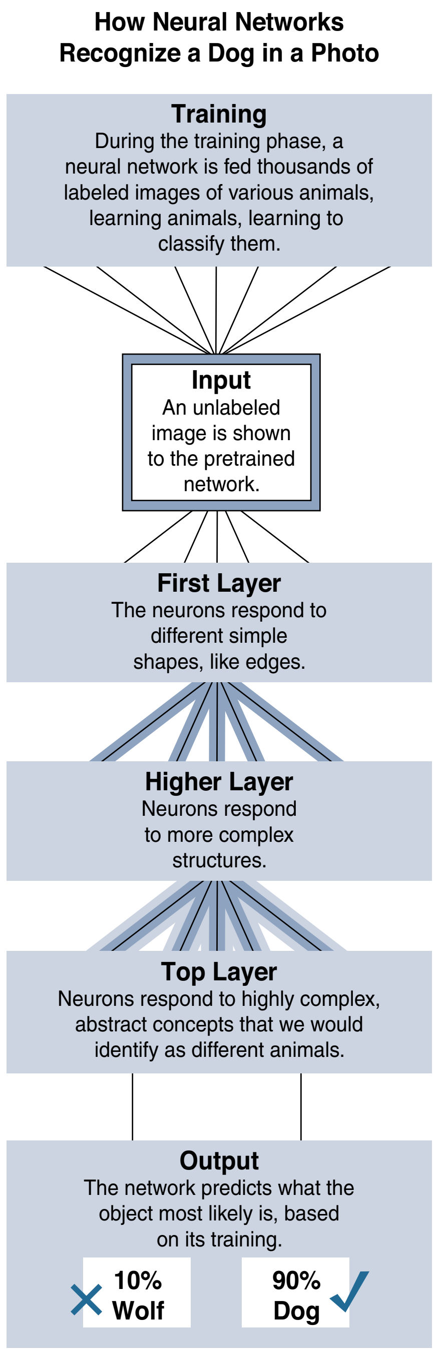

How Neural Networks
Recognize a Dog in a Photo
Training
During the training phase, a neural network is fed thousands of labeled images of various animals, learning animals, learning to classify them.
Input
An unlabeled image is shown to the pretrained network.
First Layer
The neurons respond to different simple shapes, like edges.
Higher Layer
Neurons respond to more complex structures.
Top Layer
Neurons respond to highly complex, abstract concepts that we would identify as different animals.
Output
The network predicts what the object most likely is, based on its training.
10% Wolf
90% Dog

Figure 7-6 *Neural Network Example*

By contrast, old supervised ML techniques would compare the human figure to potentially hundreds of thousands of images during the training phase, pixel by pixel, making them difficult and expensive to implement (with a lot of training needed) and slow to operate. Neural networks have been the subject of much research work. Multiple research and optimization efforts have examined the number of units and layers, the type of data processed at each layer, and the type and combination of algorithms used to process the data to make processing more efficient for specific applications. Image processing can be optimized with certain types of algorithms that may not be optimal for crowd movement classification. Another algorithm may be found in this case that would revolutionize the way these movements are processed and analyzed. Possibilities are as numerous as the applications where they can be used.

In a sense, neural networks rely on the idea that information is divided into key components, and each component is assigned a weight. The weights compared together decide the classification of this information (no straight lines + face + smile = human).

When the result of a layer is fed into another layer, the process is called deep learning ("deep" because the learning process has more than a single layer). One advantage of deep learning is that having more layers allows for richer intermediate processing and representation of the data. At each layer, the data can be formatted to be better utilized by the next layer. This process increases the efficiency of the overall result.

Machine Learning and Getting Intelligence from Big Data

When the principles of machine learning are clear, the application to IoT becomes obvious. The difficulty resides in determining the right algorithm and the right learning model for each use case. Such an analysis goes beyond the scope of this chapter, but it can be useful to organize ML operations into two broad subgroups:

- **Local learning:** In this group, data is collected and processed locally, either in the sensor itself (the edge node) or in the gateway (the fog node).
- **Remote learning:** In this group, data is collected and sent to a central computing unit (typically the data center in a specific location or in the cloud), where it is processed.

Note Associated with these two subgroups, you will encounter the term *inherited learning*. This term refers to results of learning that the local unit received from elsewhere. For example, a processing computer may collect data from multiple sensors and gateways, perform ML on this data, and send the resulting behavioral change request or conclusion back to the gateway and the sensor. This new received knowledge optimizes local operations and is inherited learning (as opposed to simple local learning).

Regardless of the location where (and, therefore, the scale at which) data is processed, common applications of ML for IoT revolve around four major domains:

- **Monitoring:** Smart objects monitor the environment where they operate. Data is processed to better understand the conditions of operations. These conditions can refer to external factors, such as air temperature, humidity, or presence of carbon dioxide in a mine, or to operational internal factors, such as the pressure of a pump, the viscosity of oil flowing in a pipe, and so on. ML can be used with monitoring to detect early failure conditions (for example, K-means deviations showing out-of-range behavior) or to better evaluate the environment (such as shape recognition for a robot automatically sorting material or picking goods in a warehouse or a supply chain).
- **Behavior control:** Monitoring commonly works in conjunction with behavior control. When a given set of parameters reach a target threshold—defined in advance (that is, supervised) or learned dynamically through deviation from mean values (that is, unsupervised)—monitoring functions generate an alarm. This alarm can be relayed to a human, but a more efficient and more advanced system would trigger a corrective action, such as increasing the flow of fresh air in the mine tunnel, turning the robot arm, or reducing the oil pressure in the pipe.
- **Operations optimization:** Behavior control typically aims at taking corrective actions based on thresholds. However, analyzing data can also lead to changes that improve the overall process. For example, a water purification plant in a smart city can implement a system to monitor the efficiency of the purification process based on which chemical (from company A or company B) is used, at what temperature, and associated to what stirring mechanism (stirring speed and depth). Neural networks can combine multiples of such units, in one or several layers, to estimate the best chemical and stirring mix for a target air temperature. This intelligence can help the plant reduce its consumption of chemicals while still operating at the same purification efficiency level. As a result of the learning, behavior control results in different machine actions. The objective is not merely to pilot the operations but to improve the efficiency and the result of these operations.
- **Self-healing, self-optimizing:** A fast-developing aspect of deep learning is the closed loop. ML-based monitoring triggers changes in machine behavior (the change is monitored by humans), and operations optimizations. In turn, the ML engine can be programmed to dynamically monitor and combine new parameters (randomly or semi-randomly) and automatically deduce and implement new optimizations when the results demonstrate a possible gain. The system becomes self-learning and self-optimizing. It also detects new K-means deviations that result in predetection of new potential defects, allowing the system to self-heal. The healing is not literal, as external factors (typically human operators) have to intervene, but the diagnosis is automated. In many cases, the system can also automatically order a piece of equipment that is detected as being close to failure or automatically take corrective actions to avoid the failure (for example, slow down operations, modify a machine's movement to avoid fatigue on a weak link).

For all these operations, a specific aspect of ML for IoT is the scale. A weather sensor mounted on a light pole in a street can provide information about the local pollution level. At the scale of the entire city, the authorities can monitor moving pollution clouds, and the global and local effects of mist or humidity, pressure, and terrain. All this information can be combined with traffic data to globally regulate traffic light patterns, reduce emissions from industrial pollution sources, or increase the density of mass transit vehicles along the more affected axes. Meanwhile, at the local level, the LED on the light pole can increase or reduce its luminosity and change its color to adapt to local conditions. This change can be driven by either local condition processing (local learning) or inherited learning.

The ability to combine fog computing on specific and specialized systems with cloud computing on data coming from multiple sources and derive global or local corrective actions is what makes ML so powerful for IoT. With open systems and the explosion of smart objects, the possibilities of correlations and cross-optimizations are very wide.

Predictive Analytics

Machine learning and big data processing for IoT fit very well into the digitization described in Chapter 1, "What Is IoT?" The advanced stages of this model see the network self-diagnose and self-optimize. In the IoT world, this behavior is what the previous section describes. When data from multiple systems is combined and analyzed together, predictions can be made about the state of the system. For example, Chapter 13, "Transportation," examines the case of sensors deployed on locomotives. Multiple smart objects measure the pull between carriages, the weight on each wheel, and multiple other parameters to offer a form of cruise control optimization for the driver. At the same time, cameras observe the state of the tracks ahead, audio sensors analyze the sound of each wheel on the tracks, and multiple engine parameters are measured and analyzed. All this data can be returned to a data processing center in the cloud that can re-create a virtual twin of each locomotive. Modeling the state of each locomotive and combining this knowledge with anticipated travel and with the states (and detected failures) of all other locomotives of the same type circulating on the tracks of the entire city, province, state, or country allows the analytics platform to make very accurate predictions on what issue is likely to affect each train and each locomotive. Such predictive analysis allows preemptive maintenance and increases the safety and efficiency of operations.

Similarly, sensors combined with big data can anticipate defects or issues in vehicles operating in mines, in manufacturing machines, or any system that can be monitored, along with other similar systems.

Big Data Analytics Tools and Technology

It is a common mistake for individuals new to the world of data management to use the terms *big data* and *Hadoop* interchangeably. Though it's true that Hadoop is at the core of many of today's big data implementations, it's not the only piece of the puzzle. Big data analytics can consist of many different software pieces that together collect, store, manipulate, and analyze all different data types. It helps to better understand the landscape by defining what big data is and what it is not. Generally, the industry looks to the "three *V*s" to categorize big data:

- **Velocity:** *Velocity* refers to how quickly data is being collected and analyzed. Hadoop Distributed File System is designed to ingest and process data very quickly. Smart objects can generate machine and sensor data at a very fast rate and require database or file systems capable of equally fast ingest functions.
- **Variety:** *Variety* refers to different types of data. Often you see data categorized as structured, semi-structured, or unstructured. Different database technologies may only be capable of accepting one of these types. Hadoop is able to collect and store all three types. This can be beneficial when combining machine data from IoT devices that is very structured in nature with data from other sources, such as social media or multimedia, that is unstructured.
- **Volume:** *Volume* refers to the scale of the data. Typically, this is measured from gigabytes on the very low end to petabytes or even exabytes of data on the other extreme. Generally, big data implementations scale beyond what is available on locally attached storage disks on a single node. It is common to see clusters of servers that consist of dozens, hundreds, or even thousands of nodes for some large deployments.

The characteristics of big data can be defined by the sources and types of data. First is machine data, which is generated by IoT devices and is typically unstructured data. Second is transactional data, which is from sources that produce data from transactions on these systems, and, have high volume and structured. Third is social data sources, which are typically high volume and structured. Fourth is enterprise data, which is data that is lower in volume and very structured. Hence big data consists of data from all these separate sources.

An additional point to consider while reviewing data sources is the amount of data ingested from each source, which determines the data storage layer design. You should also consider the mechanism to get the data from the ingest systems—namely push or pull. The type of data source—database, file, web service, stream—also needs to be considered as it also determines the structure of data.

Data ingest is the layer that connects data sources to storage. It's the layer that preprocesses, validates, extracts, and stores data temporarily for further processing. There are several patterns to consider for data ingest. First is multisource ingestion, which connects multiple data sources to ingest systems. In this pattern, ingest nodes receive streams of data from multiple sources and do processing before passing the data to intermediate nodes and to final store nodes. This pattern is typically implemented in batch systems and (less often, due to the delay of data availability) in real-time systems.

Data collection and analysis are not new concepts in the industries that helped define IoT. Industrial verticals have long depended on the ability to get, collect, and record data from various processes in order to record trends and track performance and quality.

For example, many industrial automation and control systems feed data into two distinct database types, relational databases and historians. Relational databases, such as Oracle and Microsoft SQL, are good for transactional, or process, data. Their benefit is being able to analyze complex data relationships on data that arrives over a period of time. On the other hand, historians are optimized for time-series data from systems and processes. They are built with speed of storage and retrieval of data at their core, recording each data point in a series with the pertinent information about the system being logged. This data may consist of a sensor reading, the quantity of a material, a temperature reading, or flow data.

Relational databases and historians are mature technologies that have been with us for many years, but new technologies and techniques in the data management market have opened up new possibilities for sensor and machine data. These database technologies broadly fit into a few categories that each have strengths and potential drawbacks when used in an IoT context. The three most popular of these categories are massively parallel processing systems, NoSQL, and Hadoop.

Massively Parallel Processing Databases

Enterprises have used relational databases for storing structured, row and column style data types for decades. Relational databases are often grouped into a broad data storage category called data warehouses. Though they are the centerpiece of most data architectures, they are often used for longer-term archiving and data queries that can often take minutes or hours. An example of this would be asking for all the items produced in the past year that had a particular specification. Depending on the number of items in the database and the complexity of the question being asked, the response could be slow to return.

Massively parallel processing (MPP) databases were built on the concept of the relational data warehouses but are designed to be much faster, to be efficient, and to support reduced query times. To accomplish this, MPP databases take advantage of multiple nodes (computers) designed in a scale-out architecture such that both data and processing are distributed across multiple systems.

MPPs are sometimes referred to as *analytic databases* because they are designed to allow for fast query processing and often have built-in analytic functions. As the name implies, these database types process massive data sets in parallel across many processors and nodes. An MPP architecture (see Figure 7-7) typically contains a single master node that is responsible for the coordination of all the data storage and processing across the cluster. It operates in a "shared-nothing" fashion, with each node containing local processing, memory, and storage and operating independently. Data storage is optimized across the nodes in a structured SQL-like format that allows data analysts to work with the data using common SQL tools and applications. The earlier example of a complex SQL query could be distributed and optimized, resulting in a significantly faster response. Because data stored on MPPs must still conform to this relational structure, it may not be the only database type used in an IoT implementation. The sources and types of data may vary, requiring a database that is more flexible than relational databases allow.

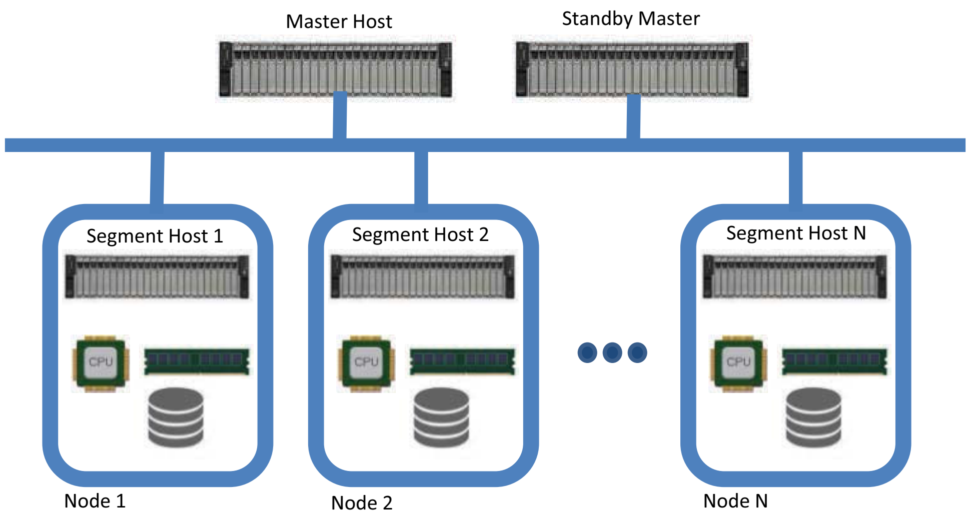


Figure 7-7 *MPP Shared-Nothing Architecture*

NoSQL Databases

NoSQL ("not only SQL") is a class of databases that support semi-structured and unstructured data, in addition to the structured data handled by data warehouses and MPPs. NoSQL is not a specific database technology; rather, it is an umbrella term that encompasses several different types of databases, including the following:

- **Document stores:** This type of database stores semi-structured data, such as XML or JSON. Document stores generally have query engines and indexing features that allow for many optimized queries.
- **Key-value stores:** This type of database stores associative arrays where a key is paired with an associated value. These databases are easy to build and easy to scale.
- **Wide-column stores:** This type of database stores similar to a key-value store, but the formatting of the values can vary from row to row, even in the same table.
- **Graph stores:** This type of database is organized based on the relationships between elements. Graph stores are commonly used for social media or natural language processing, where the connections between data are very relevant.

NoSQL was developed to support the high-velocity, urgent data requirements of modern web applications that typically do not require much repeated use. The original intent was to quickly ingest rapidly changing server logs and clickstream data generated by web-scale applications that did not neatly fit into the rows and columns required by relational databases. Similar to other data stores, like MPPs and Hadoop (discussed later), NoSQL is built to scale horizontally, allowing the database to span multiple hosts, and can even be distributed geographically.

Expanding NoSQL databases to other nodes is similar to expansion in other distributed data systems, where additional hosts are managed by a master node or process. This expansion can be automated by some NoSQL implementations or can be provisioned manually. This level of flexibility makes NoSQL a good candidate for holding machine and sensor data associated with smart objects.

Of the database types that fit under the NoSQL category, key-value stores and document stores tend to be the best fit for what is considered "IoT data." Key-value store is the technology that provides the foundation for many of today's RDBMSs, such as MS SQL, Oracle, and DB2.[3] However, unlike traditional RDBMSs, key-value stores on NoSQL are not limited to a single monolithic system. NoSQL key-value stores are capable of handling indexing and persistence simultaneously at a high rate. This makes it a great choice for time-series data sets, which record a value at a given interval of time, such as a temperature or pressure reading from a sensor.

By allowing the database schema to change quickly, NoSQL document databases tend to be more flexible than key-value store databases. Semi-structured or unstructured data that does not neatly fit into rows and columns can share the same database with organized time-series data. Unstructured data can take many forms; two examples are a photograph of a finished good on a manufacturing line used for QA and a maintenance report from a piece of equipment.

Many NoSQL databases provide additional capabilities, such as being able to query and analyze data within the database itself, eliminating the need to move and process it elsewhere. They also provide a variety of ways to query the database through an API, making it easy to integrate them with other data management applications.

Hadoop

Hadoop is the most recent entrant into the data management market, but it is arguably the most popular choice as a data repository and processing engine. Hadoop was originally developed as a result of projects at Google and Yahoo!, and the original intent for Hadoop was to index millions of websites and quickly return search results for open source search engines. Initially, the project had two key elements:

- **Hadoop Distributed File System (HDFS):** A system for storing data across multiple nodes
- **MapReduce:** A distributed processing engine that splits a large task into smaller ones that can be run in parallel

Both of these elements are still present in current Hadoop distributions and provide the foundation for other projects that are discussed later in this chapter.

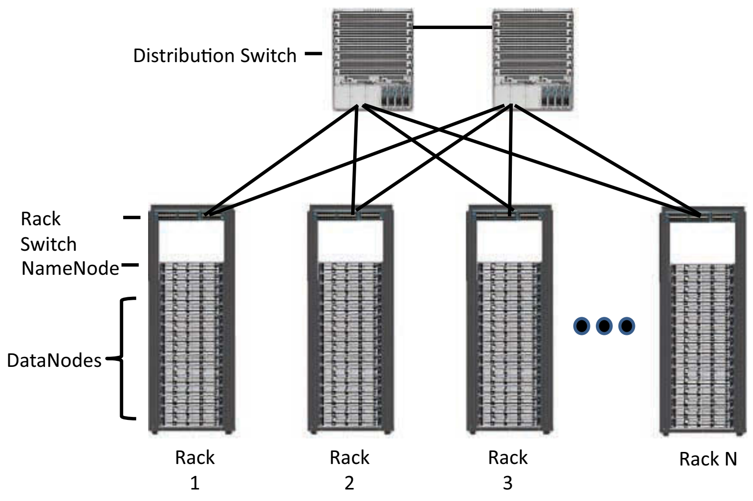


Figure 7-8 *Distributed Hadoop Cluster*

Much like the MPP and NoSQL systems discussed earlier, Hadoop relies on a scale-out architecture that leverages local processing, memory, and storage to distribute tasks and provide a scalable storage system for data. Both MapReduce and HDFS take advantage of this distributed architecture to store and process massive amounts of data and are thus able to leverage resources from all nodes in the cluster. For HDFS, this capability is handled by specialized nodes in the cluster, including NameNodes and DataNodes (see Figure 7-8):

- **NameNodes:** These are a critical piece in data adds, moves, deletes, and reads on HDFS. They coordinate where the data is stored, and maintain a map of where each block of data is stored and where it is replicated. All interaction with HDFS is coordinated through the primary (active) NameNode, with a secondary (standby) NameNode notified of the changes in the event of a failure of the primary. The NameNode takes write requests from clients and distributes those files across the available nodes in configurable block sizes, usually 64 MB or 128 MB blocks. The NameNode is also responsible for instructing the DataNodes where replication should occur.

- **DataNodes:** These are the servers where the data is stored at the direction of the NameNode. It is common to have many DataNodes in a Hadoop cluster to store the data. Data blocks are distributed across several nodes and often are replicated three, four, or more times across nodes for redundancy. Once data is written to one of the DataNodes, the DataNode selects two (or more) additional nodes, based on replication policies, to ensure data redundancy across the cluster. Disk redundancy techniques such as Redundant Array of Independent Disks (RAID) are generally not used for HDFS because the NameNodes and DataNodes coordinate block-level redundancy with this replication technique.

Figure 7-9 shows the relationship between NameNodes and DataNodes and how data blocks are distributed across the cluster.

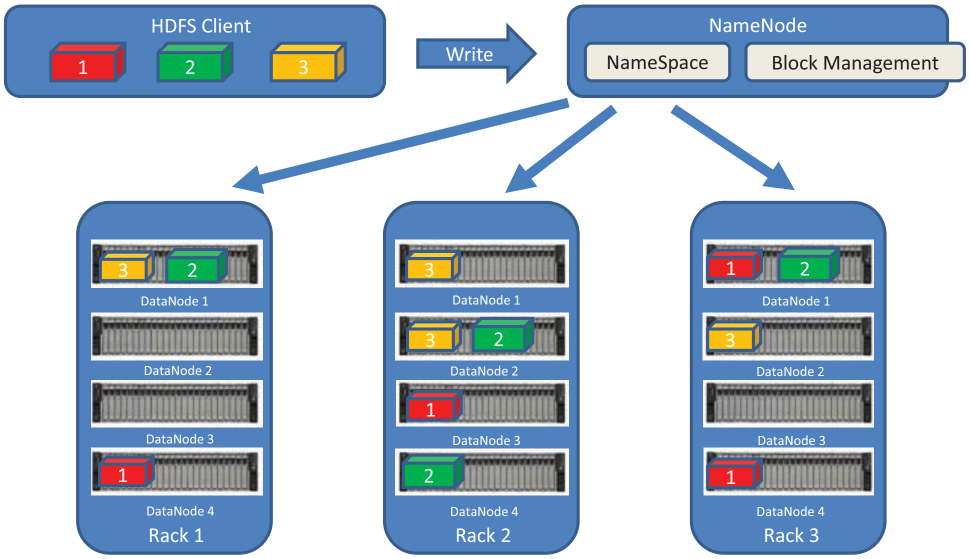


Figure 7-9 *Writing a File to HDFS*

MapReduce leverages a similar model to batch process the data stored on the cluster nodes. Batch processing is the process of running a scheduled or ad hoc query across historical data stored in the HDFS. A query is broken down into smaller tasks and distributed across all the nodes running MapReduce in a cluster. While this is useful for understanding patterns and trending in historical sensor or machine data, it has one significant drawback: time. Depending on how much data is being queried and the complexity of the query, the result could take seconds or minutes to return. If you have a real-time process running where you need a result at a moment's notice, MapReduce is not the right data processing engine for that. (Real-time streaming analytics is discussed later in this chapter.)

YARN

Introduced with version 2.0 of Hadoop, YARN (Yet Another Resource Negotiator) was designed to enhance the functionality of MapReduce. With the initial release, MapReduce was responsible for batch data processing and job tracking and resource management across the cluster. YARN was developed to take over the resource negotiation and job/task tracking, allowing MapReduce to be responsible only for data processing.

With the development of a dedicated cluster resource scheduler, Hadoop was able to add additional data processing modules to its core feature set, including interactive SQL and real-time processing, in addition to batch processing using MapReduce.

The Hadoop Ecosystem

As mentioned earlier, Hadoop plays an increasingly big role in the collection, storage, and processing of IoT data due to its highly scalable nature and its ability to work with large volumes of data. Many organizations have adopted Hadoop clusters for storage and processing of data and have looked for complimentary software packages to add additional functionality to their distributed Hadoop clusters. Since the initial release of Hadoop in 2011, many projects have been developed to add incremental functionality to Hadoop and have collectively become known as the *Hadoop ecosystem*.

Hadoop may have had meager beginnings as a system for distributed storage and processing, but it has since grown into a robust collection of projects that, combined, create a very complete data management and analytics framework. Hadoop now comprises more than 100 software projects under the Hadoop umbrella, capable of nearly every element in the data lifecycle, from collection, to storage, to processing, to analysis and visualization. Each of these individual projects is a unique piece of the overall data management solution. The following sections describe several of these packages and discuss how they are used to collect or process data.

Apache Kafka

Part of processing real-time events, such as those commonly generated by smart objects, is having them ingested into a processing engine. The process of collecting data from a sensor or log file and preparing it to be processed and analyzed is typically handled by messaging systems. Messaging systems are designed to accept data, or messages, from where the data is generated and deliver the data to stream-processing engines such as Spark Streaming or Storm. Apache Kafka is a distributed publisher-subscriber messaging system that is built to be scalable and fast. It is composed of topics, or message brokers, where producers write data and consumers read data from these topics. Figure 7-10 shows the data flow from the smart objects (producers), through a topic in Kafka, to the real-time processing engine. Due to the distributed nature of Kafka, it can run in a clustered configuration that can handle many producers and consumers simultaneously and exchanges information between nodes, allowing topics to be distributed over multiple nodes. The goal of Kafka is to provide a simple way to connect to data sources and allow consumers to connect to that data in the way they would like. The following sections describe several of these packages and discusses how they are used to collect or process data.

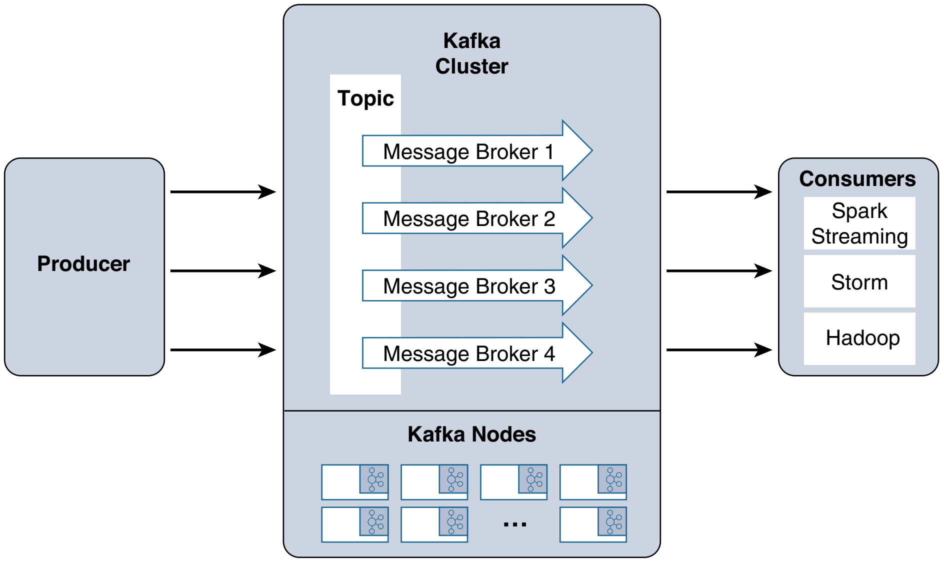


Figure 7-10 *Apache Kafka Data Flow*

Apache Spark

Apache Spark is an in-memory distributed data analytics platform designed to accelerate processes in the Hadoop ecosystem. The "in-memory" characteristic of Spark is what enables it to run jobs very quickly. At each stage of a MapReduce operation, the data is read and written back to the disk, which means latency is introduced through each disk operation. However, with Spark, the processing of this data is moved into high-speed memory, which has significantly lower latency. This speeds the batch processing jobs and also allows for near-real-time processing of events.

Real-time processing is done by a component of the Apache Spark project called Spark Streaming. Spark Streaming is an extension of Spark Core that is responsible for taking live streamed data from a messaging system, like Kafka, and dividing it into smaller microbatches. These microbatches are called discretized streams, or DStreams. The Spark processing engine is able to operate on these smaller pieces of data, allowing rapid insights into the data and subsequent actions. Due to this "instant feedback" capability, Spark is becoming an important component in many IoT deployments. Systems that control safety and security of personnel, time-sensitive processes in the manufacturing space, and infrastructure control in traffic management all benefit from these real-time streaming capabilities.

Apache Storm and Apache Flink

As you work with the Hadoop ecosystem, you will inevitably notice that different projects are very similar and often have significant overlap with other projects. This is the case with data streaming capabilities. For example, Apache Spark is often used for both distributed streaming analytics and batch processing. Apache Storm and Apache Flink are other Hadoop ecosystem projects designed for distributed stream processing and are commonly deployed for IoT use cases. Storm can pull data from Kafka and process it in a near-real-time fashion, and so can Apache Flink. This space is rapidly evolving, and projects will continue to gain and lose popularity as they evolve.

Lambda Architecture

Ultimately the key elements of a data infrastructure to support many IoT use cases involves the collection, processing, and storage of data using multiple technologies. Querying both data in motion (streaming) and data at rest (batch processing) requires a combination of the Hadoop ecosystem projects discussed. One architecture that is currently being leveraged for this functionality is the Lambda Architecture. Lambda is a data management system that consists of two layers for ingesting data (Batch and Stream) and one layer for providing the combined data (Serving). These layers allow for the packages discussed previously, like Spark and MapReduce, to operate on the data independently, focusing on the key attributes for which they are designed and optimized. Data is taken from a message broker, commonly Kafka, and processed by each layer in parallel, and the resulting data is delivered to a data store where additional processing or queries can be run. Figure 7-11 shows this parallel data flow through the Lambda Architecture.

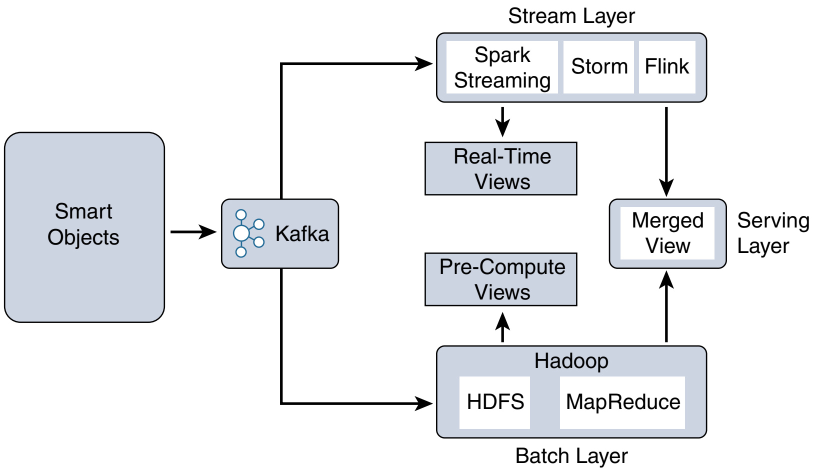


Figure 7-11 *Lambda Architecture*

The Lambda Architecture is not limited to the packages in the Hadoop ecosystem, but due to its breadth and flexibility, many of the packages in the ecosystem fill the requirements of each layer nicely:

- **Stream layer:** This layer is responsible for near-real-time processing of events. Technologies such as Spark Streaming, Storm, or Flink are used to quickly ingest, process, and analyze data on this layer. Alerting and automated actions can be triggered on events that require rapid response or could result in catastrophic outcomes if not handled immediately.
- **Batch layer:** The Batch layer consists of a batch-processing engine and data store. If an organization is using other parts of the Hadoop ecosystem for the other layers, MapReduce and HDFS can easily fit the bill. Other database technologies, such as MPPs, NoSQL, or data warehouses, can also provide what is needed by this layer.
- **Serving layer:** The Serving layer is a data store and mediator that decides which of the ingest layers to query based on the expected result or view into the data. If an aggregate or historical view is requested, it may invoke the Batch layer. If real-time analytics is needed, it may invoke the Stream layer. The Serving layer is often used by the data consumers to access both layers simultaneously.

The Lambda Architecture can provide a robust system for collecting and processing massive amounts of data and the flexibility of being able to analyze that data at different rates. One limitation of this type of architecture is its place in the network. Due to the processing and storage requirements of many of these pieces, the vast majority of these deployments are either in data centers or in the cloud. This could limit the effectiveness of the analytics to respond rapidly enough if the processing systems are milliseconds or seconds away from the device generating the data. When this is the case, a distributed edge-processing architecture may be needed to augment the central data center infrastructure.

Edge Streaming Analytics

A major area of evolution for IT in the past few years has been the transition to cloud services. Nearly every large technology company is now selling software and services from the cloud, and this includes data analytics systems, whether they are offered as a service from a public cloud operator or are built in massive private data center clouds. However, analyzing a massive volume of time-sensitive IoT data in a centralized cloud is often not ideal.

In the world of IoT, vast quantities of data are generated on the fly and often need to be analyzed and responded to immediately. Not only is the volume of data generated at the edge immense—meaning the bandwidth requirements to the cloud or data center need to be engineered to match—but the data may be so time sensitive that it needs immediate attention, and waiting for deep analysis in the cloud simply isn't possible.

One industry where data analytics is used extensively is the world of automobile racing. For example, in Formula One racing, each car has between 150 to 200 sensors that, combined, generate more than 1000 data points per second, resulting in hundreds of gigabytes of raw data per race. The sensor data is transmitted from the car and picked up by track-side wireless sensors. During a race, weather conditions may vary, tire conditions change, and accidents or other racing incidents almost always require an adaptable and flexible racing strategy. As the race develops, decisions such as when to pit, what tires to use, when to pass, and when to slow down all need to be made in seconds. Teams have found that enormous insights leading to better race results can be gained by analyzing data on the fly—and the data may come from many different sources, including trackside sensors, car telemetry, and weather reports.

Most teams use sophisticated data analytics systems to enhance racing strategy, but in many cases, this equipment resides back in the team's data center, far away from the track. For a team that has its analytics software in a data center in the UK, the latency to Australia (the most remote race) is several hundred milliseconds away. The time it takes to collect and analyze this data as a batch process in a distant part of the world is not only inefficient but can mean the difference between a successful race strategy that adapts to changing conditions and one that lacks the flexibility and agility to send meaningful instructions to the drivers. In short, it can mean the difference between winning and losing a race.

Comparing Big Data and Edge Analytics

When you hear the term *big data*, it is usually in reference to unstructured data that has been collected and stored in the cloud. The data is collected over time so that it can be analyzed through batch-processing tools, such as an RDBMS, Hadoop, or some other tool, at which point business insights are gained, and value is drawn from the data. Tools like Hadoop and MapReduce are great at tackling problems that require deep analytics on a large and complex quantity of unstructured data; however, due to their distance from the IoT endpoints and the bandwidth required to bring all the data back to the cloud, they are generally not well suited to real-time analysis of data as it is generated.

In applying data analytics to the car racing example discussed earlier, big data analytics is used to examine all the statistics of the racing team and players based on their performance in the data center or cloud. While big data can apply analytics in real-time (as discussed earlier), it is mainly focused on batch-job analytics on large volumes of data. Streaming analytics involves analyzing a race while it is happening and trying to figure out who is going to win based on the actual performance in real-time—and this analysis is typically performed as close to the edge as possible. Streaming analytics allows you to continually monitor and assess data in real-time so that you can adjust or fine-tune your predictions as the race progresses.

In the context of IoT, with streaming analytics performed at the edge (either at the sensors themselves or very close to them, in a fog node that is, for example, integrated into the gateway), it is possible to process and act on the data in real-time without waiting for the results from a future batch-processing job in the cloud. Does this mean that streaming analytics replaces big data analytics in the cloud? Not at all. They both have roles to play and both contribute to improved business insights and processes.

In one sense, if raw data is generated in the data center, it makes sense to analyze it there. But what if the majority of data is being generated in remote locations by sensors that are spread all over a wide area? To be truly effective at the moment it is created, the data needs to be analyzed and responded to as close to the edge as possible. Once it has been analyzed and reduced at the edge, the resultant data can be sent to the cloud and used to gain deeper insights over time. It is also important to remember that the edge isn't in just one place. The edge is highly distributed, which means analytics at the edge needs to be highly coordinated and structured. This also implies a communications system where edge/fog nodes are able to communicate with each other when necessary and report results to a big data system in the cloud.

From a business perspective, streaming analytics involves acting on data that is generated while it is still valuable, before it becomes stale. For example, roadway sensors combined with GPS wayfinding apps may tell a driver to avoid a certain highway due to traffic. This data is valuable for only a small window of time. Historically, it may be interesting to see how many traffic accidents or blockages have occurred on a certain segment of highway or to predict congestion based on past traffic data. However, for the driver in traffic receiving this information, if the data is not acted upon immediately, the data has little value.

From a security perspective, having instantaneous access to analyzed and preprocessed data at the edge also allows an organization to realize anomalies in its network so those anomalies can be quickly contained before spreading to the rest of the network.

To summarize, the key values of edge streaming analytics include the following:

- **Reducing data at the edge:** The aggregate data generated by IoT devices is generally in proportion to the number of devices. The scale of these devices is likely to be huge, and so is the quantity of data they generate. Passing all this data to the cloud is inefficient and is unnecessarily expensive in terms of bandwidth and network infrastructure.
- **Analysis and response at the edge:** Some data is useful only at the edge (such as a factory control feedback system). In cases such as this, the data is best analyzed and acted upon where it is generated.
- **Time sensitivity:** When timely response to data is required, passing data to the cloud for future processing results in unacceptable latency. Edge analytics allows immediate responses to changing conditions.

Edge Analytics Core Functions

To perform analytics at the edge, data needs to be viewed as real-time flows. Whereas big data analytics is focused on large quantities of data at rest, edge analytics continually processes streaming flows of data in motion. Streaming analytics at the edge can be broken down into three simple stages:

- **Raw input data:** This is the raw data coming from the sensors into the analytics processing unit.
- **Analytics processing unit (APU):** The APU filters and combines data streams (or separates the streams, as necessary), organizes them by time windows, and performs various analytical functions. It is at this point that the results may be acted on by micro services running in the APU.
- **Output streams:** The data that is output is organized into insightful streams and is used to influence the behavior of smart objects, and passed on for storage and further processing in the cloud. Communication with the cloud often happens through a standard publisher/subscriber messaging protocol, such as MQTT.

Figure 7-12 illustrates the stages of data processing in an edge APU.

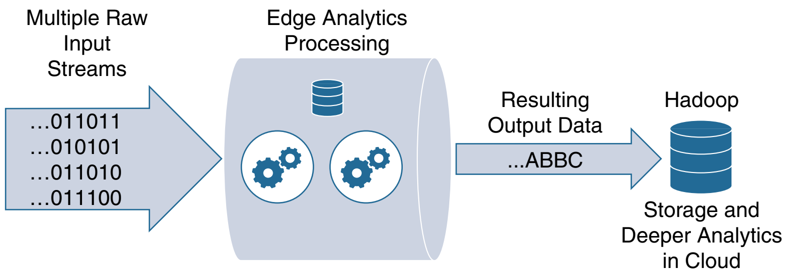
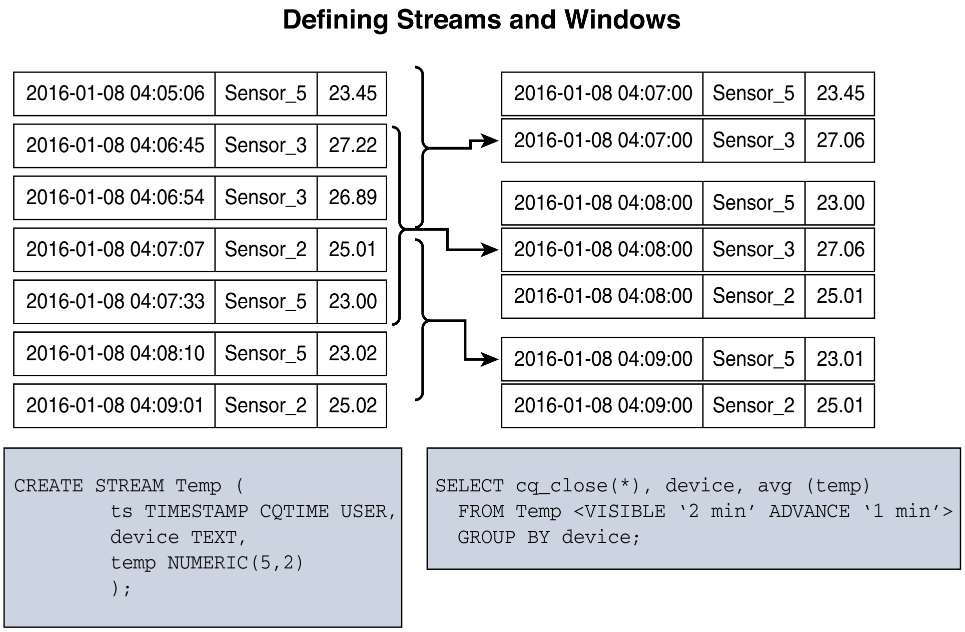

Multiple Raw Input Streams
...011011
...010101
...011010
...011100
Edge Analytics Processing
Resulting Output Data
...ABBC
Hadoop
Storage and Deeper Analytics in Cloud

Figure 7-12 *Edge Analytics Processing Unit*

In order to perform analysis in real-time, the APU needs to perform the following functions:

- **Filter:** The streaming data generated by IoT endpoints is likely to be very large, and most of it is irrelevant. For example, a sensor may simply poll on a regular basis to confirm that it is still reachable. This information is not really relevant and can be mostly ignored. The filtering function identifies the information that is considered important.
- **Transform:** In the data warehousing world, Extract, Transform, and Load (ETL) operations are used to manipulate the data structure into a form that can be used for other purposes. Analogous to data warehouse ETL operations, in streaming analytics, once the data is filtered, it needs to be formatted for processing.
- **Time:** As the real-time streaming data flows, a timing context needs to be established. This could be to correlated average temperature readings from sensors on a minute-by-minute basis. For example, Figure 7-13 shows an APU that takes input data from multiple sensors reporting temperature fluctuations. In this case, the APU is programmed to report the average temperature every minute from the sensors, based on an average of the past two minutes. (An example where this may be used is in real-time monitoring of food in a grocery store, where rolling averages of the temperature in open-air refrigeration units needs to be monitored to ensure the safety of

the food.) Note that on the left side is the cleaned stream data. This data is presented as streams to the analytics engine (note the syntax at the bottom right of the figure) that establishes the time window and calculates the average temperature over the past two minutes. The results are reported on a per-minute basis (on the right side of the figure).



2016-01-08 04:05:06	Sensor_5	23.45
2016-01-08 04:06:45	Sensor_3	27.22
2016-01-08 04:06:54	Sensor_3	26.89
2016-01-08 04:07:07	Sensor_2	25.01
2016-01-08 04:07:33	Sensor_5	23.00
2016-01-08 04:08:10	Sensor_5	23.02
2016-01-08 04:09:01	Sensor_2	25.02

2016-01-08 04:07:00	Sensor_5	23.45
2016-01-08 04:07:00	Sensor_3	27.06
2016-01-08 04:08:00	Sensor_5	23.00
2016-01-08 04:08:00	Sensor_3	27.06
2016-01-08 04:08:00	Sensor_2	25.01
2016-01-08 04:09:00	Sensor_5	23.01
2016-01-08 04:09:00	Sensor_2	25.01

```
CREATE STREAM Temp (
        ts TIMESTAMP CQTIME USER,
        device TEXT,
        temp NUMERIC(5,2)
        );
```

```
SELECT cq_close(*), device, avg (temp)
  FROM Temp <VISIBLE '2 min' ADVANCE '1 min'>
  GROUP BY device;
```

Figure 7-13 *Example: Establishing a Time Window for Analytics of Average Temperature from Sensors*

- **Correlate:** Streaming data analytics becomes most useful when multiple data streams are combined from different types of sensors. For example, in a hospital, several vital signs are measured for patients, including body temperature, blood pressure, heart rate, and respiratory rate. These different types of data come from different instruments, but when this data is combined and analyzed, it provides an invaluable picture of the health of the patient at any given time.[4] However, correlation goes beyond just combining real-time data streams. Another key aspect is combining and correlating real-time measurements with preexisting, or historical, data. For example, historical data may include the patient's past medical history, such as blood test results. Combining historical data gives the live streaming data a powerful context and promotes more insights into the current condition of the patient (see Figure 7-14).

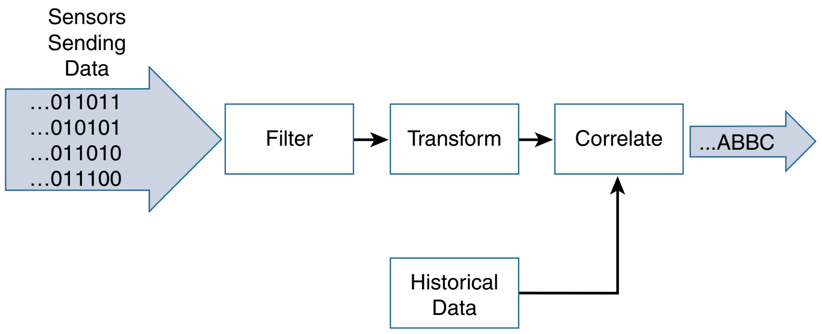


Figure 7-14 *Correlating Data Streams with Historical Data*

- **Match patterns:** Once the data streams are properly cleaned, transformed, and correlated with other live streams as well as historical data sets, pattern matching operations are used to gain deeper insights to the data. For example, say that the APU has been collecting the patient's vitals for some time and has gained an understanding of the expected patterns for each variable being monitored. If an unexpected event arises, such as a sudden change in heart rate or respiration, the pattern matching operator recognizes this as out of the ordinary and can take certain actions, such as generating an alarm to the nursing staff. The patterns can be simple relationships, or they may be complex, based on the criteria defined by the application. Machine learning may be leveraged to identify these patterns.
- **Improve business intelligence:** Ultimately, the value of edge analytics is in the improvements to business intelligence that were not previously available. For example, conducting edge analytics on patients in a hospital allows staff to respond more quickly to the patient's changing needs and also reduces the volume of unstructured (and not always useful) data sent to the cloud. Over time, the resulting changes in business logic can produce improvements in basic operations, bringing in higher levels of care as well as better efficiencies for the hospital.

Distributed Analytics Systems

Depending on the application and network architecture, analytics can happen at any point throughout the IoT system. Streaming analytics may be performed directly at the edge, in the fog, or in the cloud data center. There are no hard-and-fast rules dictating where analytics should be done, but there are a few guiding principles. We have already discussed the value of reducing the data at the edge, as well as the value of analyzing information so it can be responded to before it gets stale. There is also value in stepping back from the edge to gain a wider view with more data. It's hard to see the forest when you are standing in the middle of it staring at a tree. In other words, sometimes better insights can be gained and data responded to more intelligently when we step back from the edge and look at a wider data set.

This is the value of fog computing. (Fog computing is introduced in Chapter 2.) Fog analytics allows you to see beyond one device, giving you visibility into an aggregation of edge nodes and allowing you to correlate data from a wider set. Figure 7-15 shows an example of an oil drilling company that is measuring both pressure and temperature on an oil rig. While there may be some value in doing analytics directly on the edge, in this example, the sensors communicate via MQTT through a message broker to the fog analytics node, allowing a broader data set. (MQTT is discussed in depth in Chapter 6, "Application Protocols for IoT.") The fog node is located on the same oil rig and performs streaming analytics from several edge devices, giving it better insights due to the expanded data set. It may not be able to respond to an event as quickly as analytics performed directly on the edge device, but it is still close to responding in real-time as events occur. Once the fog node is finished with the data, it communicates the results to the cloud (again through a message broker via MQTT) for deeper historical analysis through big data analytics tools.

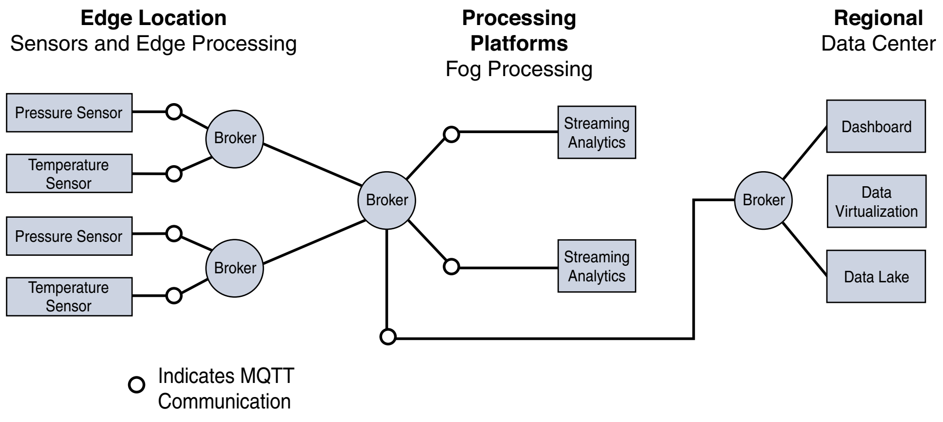


Figure 7-15 *Distributed Analytics Throughout the IoT System*

Network Analytics

Another form of analytics that is extremely important in managing IoT systems is network-based analytics. Unlike the data analytics systems previously discussed that are concerned with finding patterns in the data generated by endpoints, network analytics is concerned with discovering patterns in the communication flows from a network traffic perspective. Network analytics has the power to analyze details of communications patterns made by protocols and correlate this across the network. It allows you to understand what should be considered normal behavior in a network and to quickly identify anomalies that suggest network problems due to suboptimal paths, intrusive malware, or excessive congestion. Analysis of traffic patterns is one of the most powerful tools in an IoT network engineer's troubleshooting arsenal.

As discussed in Chapter 6, IoT endpoints, contrary to generic computing platforms, are designed to directly communicate with a very small number of specific application servers, such as an IoT message or data broker, or specific application servers and network management systems. Therefore, it could be said that IoT solutions and use cases tightly couple devices and applications. Figure 7-16 shows field area network (FAN) traffic analytics performed on the aggregation router in a smart grid.

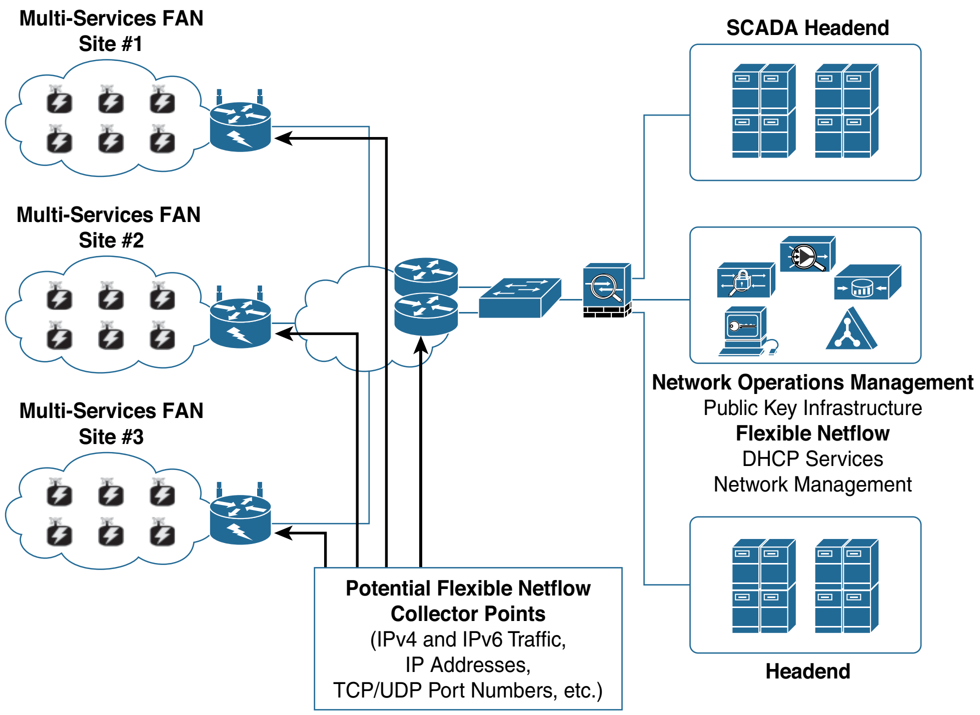

Multi-Services FAN Site #1
Multi-Services FAN Site #2
Multi-Services FAN Site #3
SCADA Headend
Network Operations Management
Public Key Infrastructure
Flexible Netflow
DHCP Services
Network Management
Headend
Potential Flexible Netflow Collector Points
(IPv4 and IPv6 Traffic,
IP Addresses,
TCP/UDP Port Numbers, etc.)

Figure 7-16 *Smart Grid FAN Analytics with NetFlow Example*

This behavior represents a key aspect that can be leveraged when performing network analytics: Network analytics offer capabilities to cope with capacity planning for scalable IoT deployment as well as security monitoring in order to detect abnormal traffic volume and patterns (such as an unusual traffic spike for a normally quiet protocol) for both centralized or distributed architectures, such as fog computing.

Consider that an IoT device sends its traffic to specific servers, either directly to an application or an IoT broker with the data payload encapsulated in a given protocol. This represents a pair of source and destination addresses, as well as application layer–dependent TCP or UDP port numbers, which can be used for network analytics.

One of the drivers of the adoption of an IP architectural framework for IoT is to leverage tools and processes largely known and deployed by Internet service providers (ISPs) as well as private corporate enterprise networks. To monitor network infrastructure, de facto industry standards and protocols allow pervasive characterization of IP traffic flows, including identification of source and/or destination addresses, data timing and volume, and application types within a network infrastructure. Flow statistics can be collected at different locations in the network. For example, centralized routers or switches that aggregate subnetworks as well as nodes that are highly distributed and connect the last mile of the infrastructure can be used to collect flow information. After data is collected in a known format, it can be sent to an external network analytics tools that delivers unique services to network managers, like security and performance monitoring and capacity planning.

In the context of IoT infrastructure deployments, for technologies discussed in Chapter 4, "Connecting Smart Objects," Chapter 5, "IP as the IoT Network Layer," and Chapter 6, the benefits of flow analytics, in addition to other network management services, are as follows:

- **Network traffic monitoring and profiling:** Flow collection from the network layer provides global and distributed near-real-time monitoring capabilities. IPv4 and IPv6 networkwide traffic volume and pattern analysis helps administrators proactively detect problems and quickly troubleshoot and resolve problems when they occur.
- **Application traffic monitoring and profiling:** Monitoring and profiling can be used to gain a detailed time-based view of IoT access services, such as the application-layer protocols, including MQTT, CoAP, and DNP3, as well as the associated applications that are being used over the network.
- **Capacity planning:** Flow analytics can be used to track and anticipate IoT traffic growth and help in the planning of upgrades when deploying new locations or services by analyzing captured data over a long period of time. This analysis affords the opportunity to track and anticipate IoT network growth on a continual basis.
- **Security analysis:** Because most IoT devices typically generate a low volume of traffic and always send their data to the same server(s), any change in network traffic behavior may indicate a cyber security event, such as a denial of service (DoS) attack. Security can be enforced by ensuring that no traffic is sent outside the scope of the IoT domain. For example, with a LoRaWAN gateway, there should be no reason to see traffic sent or received outside the LoRaWAN network server and network management system. Such traffic could indicate an attack of some sort.
- **Accounting:** In field area networks, routers or gateways are often physically isolated and leverage public cellular services and VPNs for backhaul. Deployments may have thousands of gateways connecting the last-mile IoT infrastructure over a cellular network. Flow monitoring can thus be leveraged to analyze and optimize the billing, in complement with other dedicated applications, such as Cisco Jasper, with a broader scope than just monitoring data flow.
- **Data warehousing and data mining:** Flow data (or derived information) can be warehoused for later retrieval and analysis in support of proactive analysis of multiservice IoT infrastructures and applications.

Flexible NetFlow Architecture

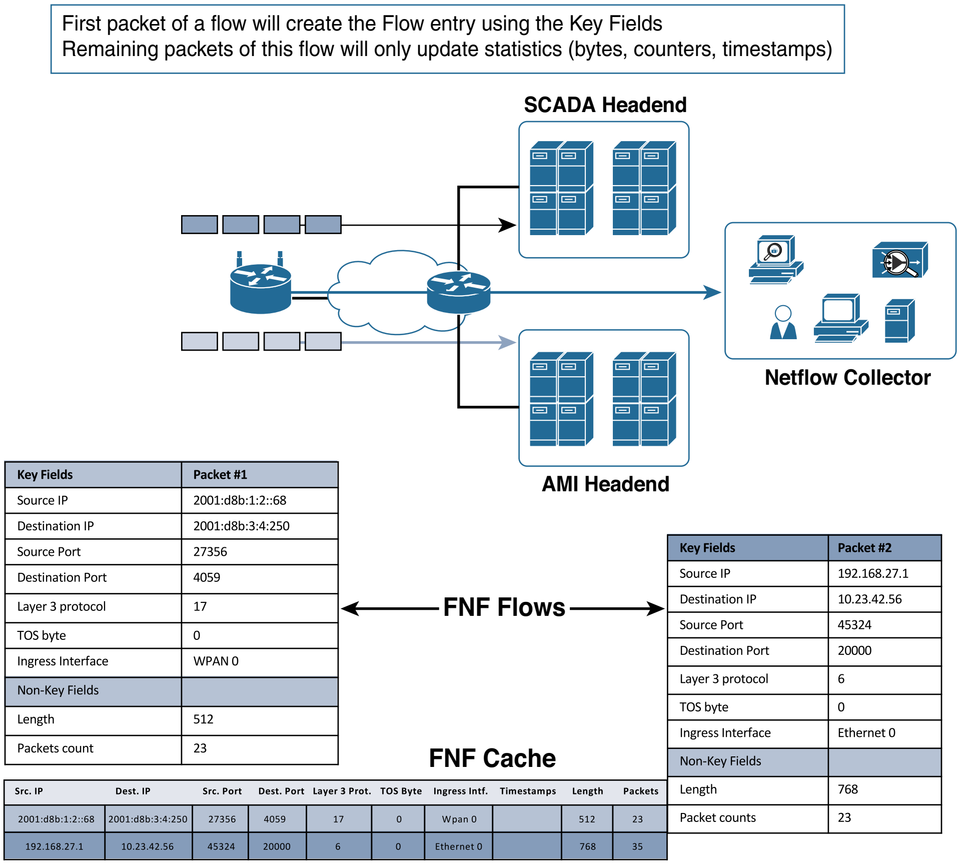
Flexible NetFlow (FNF) and IETF IPFIX (RFC 5101, RFC 5102) are examples of protocols that are widely used for networks. This section examines the fundamentals of FNF and how it may be used in an IoT deployment.

FNF is a flow technology developed by Cisco Systems that is widely deployed all over the world. Key advantages of FNF are as follows:

- Flexibility, scalability, and aggregation of flow data
- Ability to monitor a wide range of packet information and produce new information about network behavior
- Enhanced network anomaly and security detection
- User-configurable flow information for performing customized traffic identification and ability to focus and monitor specific network behavior
- Convergence of multiple accounting technologies into one accounting mechanism

FNF Components

FNF has the following main components, as shown in Figure 7-17:


First packet of a flow will create the Flow entry using the Key Fields
Remaining packets of this flow will only update statistics (bytes, counters, timestamps)
SCADA Headend
Netflow Collector
AMI Headend
FNF Flows
FNF Cache

Key Fields	Packet #1
Source IP	2001:d8b:1:2::68
Destination IP	2001:d8b:3:4:250
Source Port	27356
Destination Port	4059
Layer 3 protocol	17
TOS byte	0
Ingress Interface	WPAN 0
Non-Key Fields	
Length	512
Packets count	23

Key Fields	Packet #2
Source IP	192.168.27.1
Destination IP	10.23.42.56
Source Port	45324
Destination Port	20000
Layer 3 protocol	6
TOS byte	0
Ingress Interface	Ethernet 0
Non-Key Fields	
Length	768
Packet counts	23

Src. IP	Dest. IP	Src. Port	Dest. Port	Layer 3 Prot.	TOS Byte	Ingress Intf.	Timestamps	Length	Packets
2001:d8b:1:2::68	2001:d8b:3:4:250	27356	4059	17	0	Wpan 0		512	23
192.168.27.1	10.23.42.56	45324	20000	6	0	Ethernet 0		768	35

Figure 7-17 *Flexible NetFlow overview*

- **FNF Flow Monitor (NetFlow cache):** The FNF Flow Monitor describes the NetFlow cache or information stored in the cache. The Flow Monitor contains the flow record definitions with key fields (used to create a flow, unique per flow record: match statement) and non-key fields (collected with the flow as attributes or characteristics of a flow) within the cache. Also, part of the Flow Monitor is the Flow Exporter, which contains information about the export of NetFlow information, including the destination address of the NetFlow collector. The Flow Monitor includes various cache characteristics, including timers for exporting, the size of the cache, and, if required, the packet sampling rate.

Note Each packet that is forwarded within a router or switch is examined for a set of IP packet attributes. These attributes are the IP packet identity, or *key fields*, for the flow and determine whether the packet information is unique or similar to other packets. If packet key fields are unique, a new entry in the flow record is created. The first packet of a flow creates the flow entry, using the key fields. Remaining packets of this flow only update statistics (bytes, counters, timestamps). This methodology of flow characterization is scalable because a large amount of network information is condensed into a database of NetFlow information called the *NetFlow cache*.

Additional information (non-key fields) can be added to the Flow Record and exported. The non-key fields are not used to create or characterize the flows but are exported and just added to the flow. If a field is non-key, normally only the first packet of the flow is used for the value in this field. Examples include flow timestamps, next-hop IP addresses, subnet masks, and TCP flags.

- **FNF flow record:** A flow record is a set of key and non-key NetFlow field values used to characterize flows in the NetFlow cache. Flow records may be predefined for ease of use or customized and user defined. A typical predefined record aggregates flow data and allows users to target common applications for NetFlow. User-defined records allow selections of specific key or non-key fields in the flow record. The user-defined field is the key to Flexible NetFlow, allowing a wide range of information to be characterized and exported by NetFlow. It is expected that different network management applications will support specific user-defined and predefined flow records based on what they are monitoring (for example, security detection, traffic analysis, capacity planning).
- **FNF Exporter:** There are two primary methods for accessing NetFlow data: Using the **show** commands at the command-line interface (CLI), and using an application reporting tool. NetFlow Export, unlike SNMP polling, pushes information periodically to the NetFlow reporting collector. The Flexible NetFlow Exporter allows the user to define where the export can be sent, the type of transport for the export, and properties for the export. Multiple exporters can be configured per Flow Monitor.

- **Flow export timers:** Timers indicate how often flows should be exported to the collection and reporting server.
- **NetFlow export format:** This simply indicates the type of flow reporting format.
- **NetFlow server for collection and reporting:** This is the destination of the flow export. It is often done with an analytics tool that looks for anomalies in the traffic patterns.

Figure 7-18 illustrates the analysis reported from the FNF records on a smart grid FAN. In this example, the FNF collector is able to see the patterns of traffic for various applications as well as management traffic on the FAN.

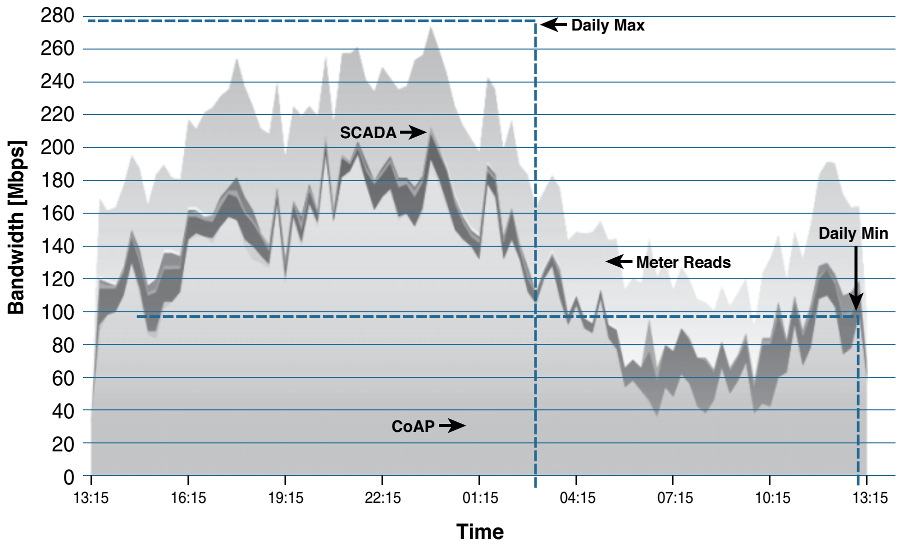


Figure 7-18 *FNF Report of Traffic on a Smart Grid FAN*

Flexible NetFlow in Multiservice IoT Networks

In the context of multiservice IoT networks, it is recommended that FNF be configured on the routers that aggregate connections from the last mile's routers. This gives a global view of all services flowing between the core network in the cloud and the IoT last-mile network (although not between IoT devices). FNF can also be configured on the last-mile gateway or fog nodes to provide more granular visibility. However, care must be taken in terms of how much northbound data is consumed through reporting.

However, flow analysis at the gateway is not possible with all IoT systems. For example, LoRaWAN gateways simply forward MAC-layer sensor traffic to the centralized LoRaWAN network server, which means flow analysis (based on Layer 3) is not possible at this point. A similar problem is encountered when using an MQTT server that sends data through an IoT broker. Some other challenges with deploying flow analytics tools in an IoT network include the following:

- The distributed nature of fog and edge computing may mean that traffic flows are processed in places that might not support flow analytics, and visibility is thus lost.
- IPv4 and IPv6 native interfaces sometimes need to inspect inside VPN tunnels, which may impact the router's performance.
- Additional network management traffic is generated by FNF reporting devices. The added cost of increasing bandwidth thus needs to be reviewed, especially if the backhaul network uses cellular or satellite communications.

In summary, existing network analytics protocols and tools may be leveraged to provide great value for IoT environments, helping to both automate and secure them.

Summary

IoT systems are producing vast volumes of data—far more than has ever been available in the past. This new paradigm of continual data generation from all forms of connected and networked instruments has created an opportunity to gain new insights and improve efficiencies like never before. The business value of IoT is not just in the ability to connect devices but comes from understanding the data these devices create. A new form of data management has therefore emerged: IoT data analytics.

Traditionally data management was performed by relational databases, which cared for well-structured data in tables where the relationships between tables and data structures were well understood and could be easily accessed through SQL. However, the majority of data generated by IoT devices is unstructured. As the IoT data is collected over time, it becomes big data and requires special handling in order to reveal the patterns within the lake of data.

To unlock the value of the data, special algorithms that perform machine learning are required to process the data and find patterns. Different types of machine learning can be used for specific purposes, including supervised, unsupervised, and neural networks.

Processing of aggregate IoT data happens in the cloud or data center and is performed by big data analytics systems, such as NoSQL, Hadoop, and MPP. These systems are specifically designed to deal with the vast volume, velocity, and variety of data generated by IoT systems.

Over time, streaming edge analytics systems have been developed to not only filter and reduce the data generated by IoT devices but also to allow near-real-time response to the IoT deices as close to the edge of the network as possible.

Finally, a different form of analytics, network analytics, is discussed in this chapter. Network analytics doesn't look at the content of the data but rather is used to discover patterns in the communications behavior of the network, helping identify and prevent security vulnerabilities, plan network evolution, and better understand the behavior of the various network elements.

In summary, network analytics comes in many shapes and forms. Each of them plays a key role in the world of IoT and helps define the true value that comes from connecting things.

References

1. Bhoopathi Rapolu, *Internet of Aircraft Things: An Industry Set to Be Transformed*, January 18, 2016, http://aviationweek.com/connected-aerospace/internet-aircraft-things-industry-set-be-transformed.
2. Goutam Chakraborty and Murali Krishna Pagolu, *Analysis of Unstructured Data: Applications of Text Analytics and Sentiment Mining*, https://support.sas.com/resources/papers/proceedings14/1288-2014.pdf
3. Bernard Marr, *That's Data Science: Airbus Puts 10,000 Sensors in Every Single Wing!* April 9, 2015, www.datasciencecentral.com/profiles/blogs/that-s-data-science-airbus-puts-10-000-sensors-in-every-single.
4. William Vorhies, *Stream Processing and Streaming Analytics—How It Works*, October 29, 2015, www.datasciencecentral.com/profiles/blogs/stream-processing-and-streaming-analytics-how-it-works.

This page intentionally left blank

Chapter 8

Securing IoT

It is often said that if World War III breaks out, it will be fought in cyberspace. As IoT brings more and more systems together under the umbrella of network connectivity, security has never been more important. From the electrical grid system that powers our world, to the lights that control the flow of traffic in a city, to the systems that keep airplanes flying in an organized and efficient way, security of the networks, devices, and the applications that use them is foundational and essential for all modern communications systems. Providing security in such a world is not easy. Security is among the very few, if not the only, technology disciplines that must operate with external forces continually working against desired outcomes. To further complicate matters, these external forces are able to leverage traditional technology as well as nontechnical methods (for example, physical security, operational processes, and so on) to meet their goals. With so many potential attack vectors, information and cybersecurity is a challenging, but engaging, topic that is of critical importance to technology vendors, enterprises, and service providers alike.

Information technology (IT) environments have faced active attacks and information security threats for many decades, and the incidents and lessons learned are well-known and documented. By contrast, operational technology (OT) environments were traditionally kept in silos and had only limited connection to other networks. Thus, the history of cyber attacks on OT systems is much shorter and has far fewer incidents documented. Therefore, the learning opportunities and the body of cataloged incidents with their corresponding mitigations are not as rich as in the IT world. Security in the OT world also addresses a wider scope than in the IT world. For example, in OT, the word *security* is almost synonymous with *safety*. In fact, many of the industrial security standards that form the foundation for industrial IoT security also incorporate equipment and personnel safety recommendations.

It is for these reasons that this chapter focuses on the core principles of securing OT environments. IT security is a vast domain with many books dedicated to its various aspects. An exhaustive treatment of the subject is simply not possible in one chapter, so we instead focus on OT security and the elements of IT security that are fundamental to OT security. In addition, the industry-specific chapters in Part III, “IoT in Industry,” discuss the application of security to specific industry verticals.

This chapter provides a historical perspective of OT security, how it has evolved, and some of the common challenges it faces. It also details some of the key differences between securing IT and OT environments. Finally, this chapter explores a number of practical steps for creating a more secure industrial environment, including best practices in introducing modern IT network security into legacy industrial environments. It includes the following sections:

- **A Brief History of OT Security:** This section provides an overview of how OT environments have evolved and the impact that the evolution has had on securing operational networks.
- **Common Challenges in OT Security:** This section provides a synopsis of different security challenges in operational environments, including legacy systems and insecure protocols and assets.
- **How IT and OT Security Practices and Systems Vary:** This section provides a comparison between the security practices in enterprise IT environments and operational industrial environments.
- **Formal Risk Analysis Structures: OCTAVE and FAIR:** This section provides a holistic view of securing an operational environment and a risk assessment framework that includes the people, processes, and vendor ecosystem components that make up a control system.
- **The Phased Application of Security in an Operational Environment:** This section provides a description of a phased approach to introducing modern network security into largely preexisting legacy industrial networks.

A Brief History of OT Security

To better understand the current situation in industrial environments, it is important to differentiate between assumptions and realities. Few topics in information technology inspire more fear, uncertainty, or doubt than cybersecurity. This chapter is therefore limited to incidents and data sources from official sources rather than public media reports or uncorroborated third-party accounts.

More than in most other sectors, cybersecurity incidents in industrial environments can result in physical consequences that can cause threats to human lives as well as damage to equipment, infrastructure, and the environment. While there are certainly traditional IT-related security threats in industrial environments, it is the physical manifestations and impacts of the OT security incidents that capture media attention and elicit broad-based public concern.

One example of a reported incident where physical damage was caused by a cybersecurity attack is the Stuxnet malware that damaged uranium enrichment systems in Iran. Another example is an event that damaged a furnace in a German smelter. In both incidents, multiple steps led to the undesirable outcomes. Many of the security policies and mitigation procedures that were in place went unheeded; however, if properly implemented, they could have impeded or possibly stopped the attacks entirely. For example, Stuxnet is thought to have been deployed on USB memory sticks up to two years before it was finally identified and discovered.

In addition to physical damage, operational interruptions have occurred in OT environments due to cybersecurity incidents. For example, in 2000, the sewage control system of Maroochy Shire in Queensland, Australia, was accessed remotely, and it released 800,000 liters of sewage into the surrounding waterways. In 2015, the control systems of the Ukrainian power distribution operator Kyiv Oblenergo were remotely accessed by attackers, causing an outage that lasted several hours and resulted in days of degraded service for thousands of customers. In both cases, known mitigation techniques could have been applied to detect the attacks earlier or block the ability to hijack production systems and affect service.

Historically, attackers were skilled individuals with deep knowledge of technology and the systems they were attacking. However, as technology has advanced, tools have been created to make attacks much easier to carry out. To further complicate matters, these tools have become more broadly available and more easily obtainable. Compounding this problem, many of the legacy protocols used in IoT environments are many decades old, and there was no thought of security when they were first developed. This means that attackers with limited or no technical capabilities now have the potential to launch cyber attacks, greatly increasing the frequency of attacks and the overall threat to end operators. It is, however, a common misconception that attackers always have the advantage and that end operators lack effective defensive capabilities. An important advantage for operators is the fact that they are far more familiar with their environment and have a better understanding of their processes, and can thus leverage multiple technologies and capabilities to defend their networks against attack. This is critical as networks will continue to face ever-evolving and changing methods of attack that will be increasingly difficult to defend against and respond to.

Communication networks, both local and geographically dispersed, have been used in industrial environments for decades. For example, remote monitoring of substations in utilities and communications between semi-autonomous systems in manufacturing are long-standing examples of such OT networks. These OT-specific communication systems have typically been standalone and physically isolated from the traditional IT enterprise networks in the same companies. While it follows the traditional logic of “security through obscurity,” this form of network compartmentalization has led to the independent evolution of IT and OT networks, with interconnections between the environments strictly segregated and monitored.

The isolation between industrial networks and the traditional IT business networks has been referred to as an “air gap,” suggesting that there are no links between the two. While there are clearly examples of such extreme isolation in some industries, it is actually not an accurate description of most IoT networks today. Broadly speaking, there is a varying amount of interconnection between OT and IT network environments, and many interdependencies between the two influence the level of interconnection.

In addition to the policies, regulations, and governance imposed by the different industrial environments, there is also a certain amount of end-user preference and deployment-specific design that determines the degree of isolation between IT and OT environments. While some organizations continue to maintain strict separation, others are starting to allow certain elements of interconnection. One common example of this is the use of Ethernet and IP to transport control systems in industrial environments. As much as IT and OT networks are still operated and managed separately in a good portion of the world, the prevailing trend is to consolidate networks based on IT-centric technologies such as TCP/IP, Ethernet, and common APIs.

This evolution of ever-increasing IT technologies in the OT space comes with the benefits of increased accessibility and a larger base of skilled operators than with the nonstandard and proprietary communication methods in traditional industrial environments. The challenges associated with these well-known IT standards is that security vulnerabilities are more widely known, and abuse of those systems is often easier and occurs on a much larger scale. This accessibility and scale makes security a major concern, particularly because many systems and devices in the operational domain were never envisioned to run on a shared, open standards–based infrastructure, and they were not designed and developed with high levels of built-in security capabilities.

Projects in industrial environments are often capital intensive, with an expected life span that can be measured in decades. Unlike in IT-based enterprises, OT-deployed solutions commonly have no reason to change as they are designed to meet specific (and often single-use) functions, and have no requirements or incentives to be upgraded. A huge focus and priority in OT is system uptime and high availability, so changes are typically only made to fix faults or introduce new system capabilities in support of that goal. As a result, deployed OT systems often have slower development and upgrade cycles and can quickly become out of sync with traditional IT network environments. The outcome is that both OT technologies and the knowledge of those looking after those operational systems have progressed at a slower pace than their IT counterparts.

Most of the industrial control systems deployed today, their components, and the limited associated security elements were designed when adherence to published and open standards were rare. The proprietary nature of these systems meant that threats from the outside world were unlikely to occur and were rarely addressed. There has, however, been a growing trend whereby OT system vulnerabilities have been exposed and reported. This increase is depicted in Figure 8-1, which shows the history of vulnerability disclosures in industrial control systems (ICSs) since 2010. While the number of reports has been increasing over the past years, it is likely that there are still many others that are not reported or discovered.

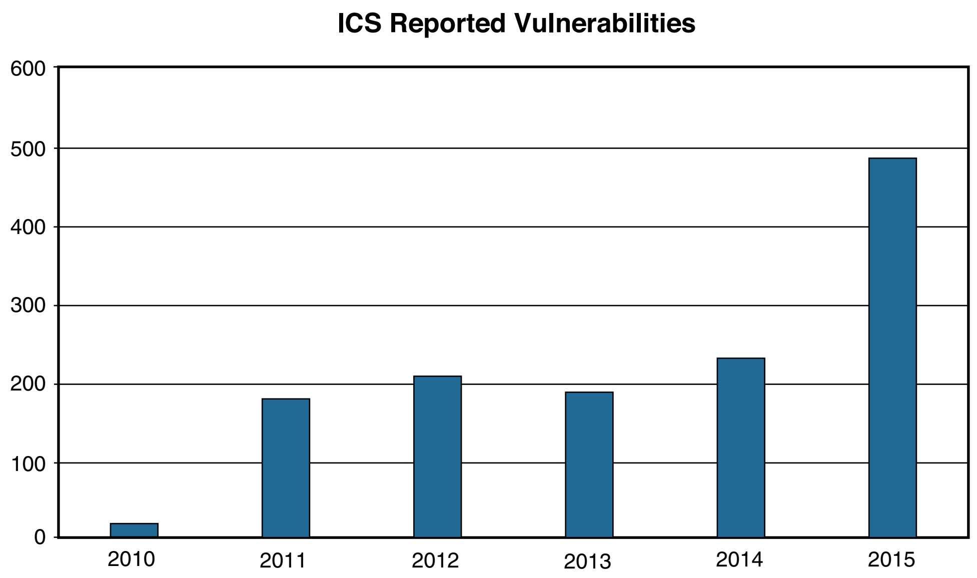


Figure 8-1 *History of Vulnerability Disclosures in Industrial Control Systems Since 2010 (US Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) https://ics-cert.us-cert.gov).*

Given the slow rate of change and extended upgrade cycles of most OT environments, the investment in security for industrial communication and compute technologies has historically lagged behind the investment in securing traditional IT enterprise environments.

Common Challenges in OT Security

The security challenges faced in IoT are by no means new and are not limited to specific industrial environments. The following sections discuss some of the common challenges faced in IoT.

Erosion of Network Architecture

Two of the major challenges in securing industrial environments have been initial design and ongoing maintenance. The initial design challenges arose from the concept that networks were safe due to physical separation from the enterprise with minimal or no connectivity to the outside world, and the assumption that attackers lacked sufficient knowledge to carry out security attacks. In many cases, the initial network design is sound and even follows well-defined industrial best practices and standards, such as the Purdue Model for Control Hierarchy that was introduced in Chapter 2, "IoT Network Architecture and Design." The challenge, and the biggest threat to network security, is standards and best practices either being misunderstood or the network being poorly maintained. In fact, from a security design perspective, it is better to know that communication paths are insecure than to not know the actual communication paths. It is more common that, over time, what may have been a solid design to begin with is eroded through ad hoc updates and individual changes to hardware and machinery without consideration for the broader network impact. This kind of organic growth has led to miscalculations of expanding networks and the introduction of wireless communication in a standalone fashion, without consideration of the impact to the original security design. These uncontrolled or poorly controlled OT network evolutions have, in many cases, over time led to weak or inadequate network and systems security.

There is a wide variety in secured network designs within and across different industries. For example, power utilities have a strong history of leveraging modern technologies for operational activities, and in North America there are regulatory requirements in place from regulatory authorities, such as North American Electric Reliability Corporation's (NERC's) Critical Infrastructure Protection (CIP), discussed in greater detail in Chapter 11, "Utilities"), to implement secure network connectivity and control with reasonably prescriptive actions. By contrast, in other industries, there are often no legislative requirements or compliance policies, which has resulted in widespread differences in security capabilities.

In many industries, the control systems consist of packages, skids, or components that are self-contained and may be integrated as semi-autonomous portions of the network. These packages may not be as fully or tightly integrated into the overall control system, network management tools, or security applications, resulting in potential risk.

Pervasive Legacy Systems

Due to the static nature and long lifecycles of equipment in industrial environments, many operational systems may be deemed legacy systems. For example, in a power utility environment, it is not uncommon to have racks of old mechanical equipment still operating alongside modern intelligent electronic devices (IEDs). In many cases, legacy components are not restricted to isolated network segments but have now been consolidated into the IT operational environment. From a security perspective, this is potentially dangerous as many devices may have historical vulnerabilities or weaknesses that have not been patched and updated, or it may be that patches are not even available due to the age of the equipment.

Beyond the endpoints, the communication infrastructure and shared centralized compute resources are often not built to comply with modern standards. In fact, their communication methods and protocols may be generations old and must be interoperable with the oldest operating entity in the communications path. This includes switches, routers, firewalls, wireless access points, servers, remote access systems, patch management, and network management tools. All of these may have exploitable vulnerabilities and must be protected.

Insecure Operational Protocols

Many industrial control protocols, particularly those that are serial based, were designed without inherent strong security requirements. Furthermore, their operation was often within an assumed secure network. In addition to any inherent weaknesses or vulnerabilities, their operational environment may not have been designed with secured access control in mind.

Industrial protocols, such as supervisory control and data acquisition (SCADA) (refer to Chapter 6, "Application Protocols for IoT"), particularly the older variants, suffer from common security issues. Three examples of this are a frequent lack of authentication between communication endpoints, no means of securing and protecting data at rest or in motion, and insufficient granularity of control to properly specify recipients or avoid default broadcast approaches. These may not be as critical in self-contained systems, but between zones or on longer network segments, such as a WAN (particularly a public WAN), they may be significant considerations.

The structure and operation of most of these protocols is often publicly available. While they may have been originated by a private firm, for the sake of interoperability, they are typically published for others to implement. Thus, it becomes a relatively simple matter to compromise the protocols themselves and introduce malicious actors that may use them to compromise control systems for either reconnaissance or attack purposes that could lead to undesirable impacts in normal system operation.

The following sections discuss some common industrial protocols and their respective security concerns. Note that many have serial, IP, or Ethernet-based versions, and the security challenges and vulnerabilities are different for the different variants.

Modbus

Modbus is commonly found in many industries, such as utilities and manufacturing environments, and has multiple variants (for example, serial, TCP/IP). It was created by the first programmable logic controller (PLC) vendor, Modicon, and has been in use since the 1970s. It is one of the most widely used protocols in industrial deployments, and its development is governed by the Modbus Organization. For more details on Modbus, refer to Chapter 6.

The security challenges that have existed with Modbus are not unusual. Authentication of communicating endpoints was not a default operation because it would allow an inappropriate source to send improper commands to the recipient. For example, for a message to reach its destination, nothing more than the proper Modbus address and function call (code) is necessary.

Some older and serial-based versions of Modbus communicate via broadcast. The ability to curb the broadcast function does not exist in some versions. There is potential for a recipient to act on a command that was not specifically targeting it. Furthermore, an attack could potentially impact unintended recipient devices, thus reducing the need to understand the details of the network topology.

Validation of the Modbus message content is also not performed by the initiating application. Instead, Modbus depends on the network stack to perform this function. This could open up the potential for protocol abuse in the system.

DNP3 (Distributed Network Protocol)

DNP3 is found in multiple deployment scenarios and industries. It is common in utilities and is also found in discrete and continuous process systems. Like many other ICS/SCADA protocols, it was intended for serial communication between controllers and simple IEDs. (For more detailed information on DNP3, refer to Chapter 6.)

There is an explicit "secure" version of DNP3, but there also remain many insecure implementations of DNP3 as well. DNP3 has placed great emphasis on the reliable delivery of messages. That emphasis, while normally highly desirable, has a specific weakness from a security perspective. In the case of DNP3, participants allow for unsolicited responses, which could trigger an undesired response. The missing security element here is the ability to establish trust in the system's state and thus the ability to trust the veracity of the information being presented. This is akin to the security flaws presented by Gratuitous ARP messages in Ethernet networks, which has been addressed by Dynamic ARP Inspection (DAI) in modern Ethernet switches.

ICCP (Inter-Control Center Communications Protocol)

ICCP is a common control protocol in utilities across North America that is frequently used to communicate between utilities. Given that it must traverse the boundaries between different networks, it holds an extra level of exposure and risk that could expose a utility to cyber attack.

Unlike other control protocols, ICCP was designed from inception to work across a WAN. Despite this role, initial versions of ICCP had several significant gaps in the area of security. One key vulnerability is that the system did not require authentication for communication. Second, encryption across the protocol was not enabled as a default condition, thus exposing connections to man-in-the-middle (MITM) and replay attacks.

OPC (OLE for Process Control)

OPC is based on the Microsoft interoperability methodology Object Linking and Embedding (OLE). This is an example where an IT standard used within the IT domain and personal computers has been leveraged for use as a control protocol across an industrial network.

In industrial control networks, OPC is limited to operation at the higher levels of the control space, with a dependence on Windows-based platforms. Concerns around OPC begin with the operating system on which it operates. Many of the Windows devices in the operational space are old, not fully patched, and at risk due to a plethora of well-known vulnerabilities. The dependence on OPC may reinforce that dependence. While newer versions of OPC have enhanced security capabilities, they have also opened up new communications modes, which have both positive and negative security potential.

Of particular concern with OPC is the dependence on the Remote Procedure Call (RPC) protocol, which creates two classes of exposure. The first requires you to clearly understand the many vulnerabilities associated with RPC, and the second requires you to identify the level of risk these vulnerabilities bring to a specific network.

International Electrotechnical Commission (IEC) Protocols

The IEC 61850 standard was created to allow vendor-agnostic engineering of power utility systems, which would, in turn, allow interoperability between vendors and standardized communication protocols. Three message types were initially defined: MMS (Manufacturing Message Specification), GOOSE (Generic Object Oriented Substation Event), and SV (Sampled Values). Web services was a fourth protocol that was added later. Here we provide a short summary of each, but for more information on IEC protocols, see Chapter 11:

- **MMS (61850-8.1):** MMS is a client/server protocol that leverages TCP/IP and operates at Layer 3. It provides the same functionality as other SCADA protocols, such as IEC 60870 and Modbus.
- **GOOSE (61850-8.1):** GOOSE is a Layer 2 protocol that operates via multicast over Ethernet. It allows IEDs to exchange data “horizontally,” between bays and between substations, especially for interlocking, measurement, and tripping signals.
- **SV (61850-9-2):** SV is a Layer 2 protocol that operates via multicast over Ethernet. It carries voltage and current samples, typically on the process bus, but it can also flow over the station bus.

Both GOOSE and SV operate via a publisher/subscriber model, with no reliability mechanism to ensure that data has been received.

IEC 61850 has several known security deficiencies that could be leveraged by skilled attackers to compromise a control system. Authentication is embedded in MMS, but it is based on clear-text passwords, and authentication is not available in GOOSE or SV. Firmware is typically not signed, which means there is no way to verify its authenticity or integrity. GOOSE and SV have limited message integrity, which makes it relatively easy to impersonate a publisher.

When the standard was first released, there was minimal security capability in these protocols, but this is being addressed by IEC 62351 with the introduction of well-known IT-based security measures, such as certificate exchange.

IEC 60870 is widely used for SCADA telecontrol in Europe, particularly in the power utility industry, and for widely geographically dispersed control systems. Part 5 of the standard outlines the communication profiles used between endpoints to exchange telecontrol messages. 60870-5-101 is the serial implementation profile, 60870-5-104 is the IP implementation profile, and 60870-5-103 is used for protection equipment. Again, in the early iterations of IEC 60870-5, security was lacking. This is now being addressed by IEC 62351, with the 60870-5-7 security extensions work, applicable to 60870-101 and 60870-104.

Other Protocols

At times, discussions about the security of industrial systems are decidedly focused on industrial control protocols as if they were the sum total of what would be observed or considered. This assumption is narrow-minded and problematic on many levels. In fact, it is highly recommended that a security practitioner passively identify all aspects of the traffic traversing the network prior to implementing any kind of controls or security measures therein. Of particular importance are proper accounting, handling, and understanding of the most basic protocols, transport mechanisms, and foundational elements of any network, including ARP, UDP, TCP, IP, and SNMP.

Some specialized environments may also have other background control protocols. For example, many IoT networks reach all the way to the individual sensors, so protocols such as Constrained Application Protocol (CoAP) (see Chapter 6) and Datagram Transport Layer Security (DTLS) are used, and have to be considered separately from a security perspective.

Device Insecurity

Beyond the communications protocols that are used and the installation base of legacy systems, control and communication elements themselves have a history of vulnerabilities. As mentioned earlier in this chapter (see Figure 8-1), prior to 2010, the security community paid little attention to industrial compute, and as a result, OT systems have not gone through the same "trial by fire" as IT systems. Figure 8-2 shows this graphically by simply overlaying the count of industrial security topics presented at the Black Hat security conference with the number of vulnerabilities reported for industrial control systems. The correlation between presentations on the subject of OT security at Black Hat and the number of vulnerabilities discovered is obvious, including the associated slowing of discoveries.

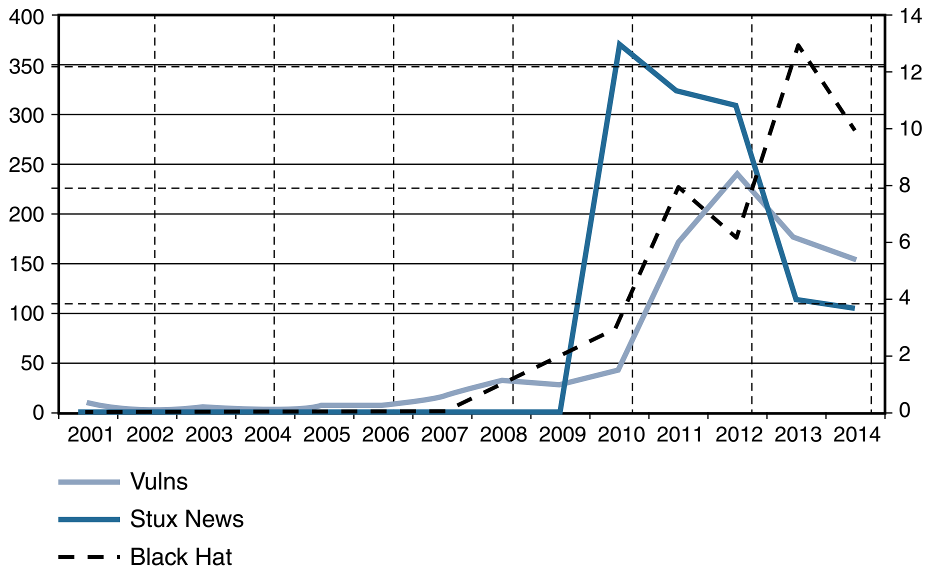


Figure 8-2 *Correlation of Industrial Black Hat Presentations with Discovered Industrial Vulnerabilities (US Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) https://ics-cert.us-cert.gov).*

To understand the nature of the device insecurity, it is important to review the history of what vulnerabilities were discovered and what types of devices were affected. A review of the time period 2000 to 2010 reveals that the bulk of discoveries were at the higher levels of the operational network, including control systems trusted to operate plants, transmission systems, oil pipelines, or whatever critical function is in use.

It is not difficult to understand why such systems are frequently found vulnerable. First, many of the systems utilize software packages that can be easily downloaded and worked against. Second, they operate on common hardware and standard operating systems, such as Microsoft Windows. Third, Windows and the components used within those applications are well known to traditionally IT-focused security researchers. There is little need to develop new tools or techniques when those that have long been in place are sufficiently adequate to breach the target's defenses. For example, Stuxnet, the most famous of the industrial compute-based attacks, was initially successful because it was able to exploit a previously unknown vulnerability in Windows.

The ICS vendor community is also lagging behind IT counterparts with regard to security capabilities and practices, as well as cooperation with third-party security researchers. That said, this situation is beginning to get significant industry focus and is improving through a number of recent initiatives designed to formally address security vulnerability and system testing in the industrial environment. While there are some formal standards, such as ISO/IEC 15408 (Common Criteria), ISO/IEC 19790, and a few others, there remain few formal security testing entities. Beyond formal testing, there is little regulatory enforcement of common criteria that address device security testing.

It was not too long ago that the security research community was viewed as a threat, rather than as a valued and often free service to expose potential dangers. While the situation has improved, operational efforts still significantly lag behind IT-based initiatives, such as bug bounty reward programs and advanced vulnerability preparation programs, along the lines of something like the Microsoft Active Protections Program (MAPP). To go a step further, in the industrial realm, there aren't even parallels to the laws that protect individuals' private data. While many states and countries require notification if an individual's personal and financial data is possibly exposed, outside the electrical utility industry, very few laws require the reporting of incidents that may have put lives at risk.

Dependence on External Vendors

While modern IT environments may be outsourcing business operations or relegating certain processing or storage functions to the cloud, it is less common for the original equipment manufacturers of the IT hardware assets to be required to operate the equipment. However, that level of vendor dependence is not uncommon in some industrial spaces.

Direct and on-demand access to critical systems on the plant floor or in the field are sometimes written directly into contracts or are required for valid product warranties. This has clear benefits in many industries as it allows vendors to remotely manage and monitor equipment and to proactively alert the customer if problems are beginning to creep in. While contracts may be written to describe equipment monitoring and management requirements with explicit statements of what type of access is required and under what conditions, they generally fail to address questions of shared liability for security breaches or processes to ensure communication security.

Such vendor dependence and control are not limited to remote access. Onsite management of non-employees that are to be granted compute and network access are also required, but again, control conditions and shared responsibility statements are yet to be observed.

Security Knowledge

In the industrial operations space, the technical investment is primarily in connectivity and compute. It has seen far less investment in security relative to its IT counterpart. According to the research firm Infonetics, the industrial firewall market in 2015 was only approximately 4% the size of the overall firewall market.

Another relevant challenge in terms of OT security expertise is the comparatively higher age of the industrial workforce. According to a study by the US Bureau of Labor, in North America the average age gap between manufacturing workers and other non-farm workers doubled between 2000 and 2012, and the trend shows no sign of reversing. Simultaneously, new connectivity technologies are being introduced in OT industrial environments that require up-to-date skills, such as TCP/IP, Ethernet, and wireless that are quickly replacing serial-based legacy technologies. The rapid expansion of extended communications networks and the need for an industrial controls-aware workforce creates an equally serious gap in security awareness.

This gap in OT security knowledge is actively being addressed. Education for industrial security environments has grown steadily, particularly in the electrical utility space, where regulations such as NERC CIP (CIP 004) and IEC 62351 (01) require ongoing training.

Due to the importance of security in the industrial space, all likely attack surfaces are treated as unsafe. Unfortunately, considering the potential massive public impact of breaching these systems, there remains a healthy paranoia concerning the connection of IT-centric technologies and external connections, despite the massive amount of investment in security in these areas. Bringing industrial networks up to the latest and most secure levels is a slow process due to deep historical cultural and philosophical differences between OT and IT environments.

How IT and OT Security Practices and Systems Vary

The differences between an enterprise IT environment and an industrial-focused OT deployment are important to understand because they have a direct impact on the security practice applied to them. Some of these areas are touched on briefly earlier in this chapter, and they are more explicitly discussed in the following sections.

The Purdue Model for Control Hierarchy

Regardless of where a security threat arises, it must be consistently and unequivocally treated. IT information is typically used to make business decisions, such as those in process optimization, whereas OT information is instead characteristically leveraged to make physical decisions, such as closing a valve, increasing pressure, and so on. Thus, the operational domain must also address physical safety and environmental factors as part of its security strategy—and this is not normally associated with the IT domain. Organizationally, IT and OT teams and tools have been historically separate, but this has begun to change, and they have started to converge, leading to more traditionally IT-centric solutions being introduced to support operational activities. For example, systems such as firewalls and intrusion prevention systems (IPS) are being used in IoT networks.

As the borders between traditionally separate OT and IT domains blur, they must align strategies and work more closely together to ensure end-to-end security. The types of devices that are found in industrial OT environments are typically much more highly optimized for tasks and industrial protocol-specific operation than their IT counterparts. Furthermore, their operational profile differs as well.

Industrial environments consist of both operational and enterprise domains. To understand the security and networking requirements for a control system, the use of a logical framework to describe the basic composition and function is needed. The Purdue Model for Control Hierarchy, introduced in Chapter 2, is the most widely used framework across industrial environments globally and is used in manufacturing, oil and gas, and many other industries. It segments devices and equipment by hierarchical function levels and areas and has been incorporated into the ISA99/IEC 62443 security standard, as shown in Figure 8-3. For additional detail on how the Purdue Model for Control Hierarchy is applied to the manufacturing and oil and gas industries, see Chapter 9, "Manufacturing," and Chapter 10, "Oil and Gas."

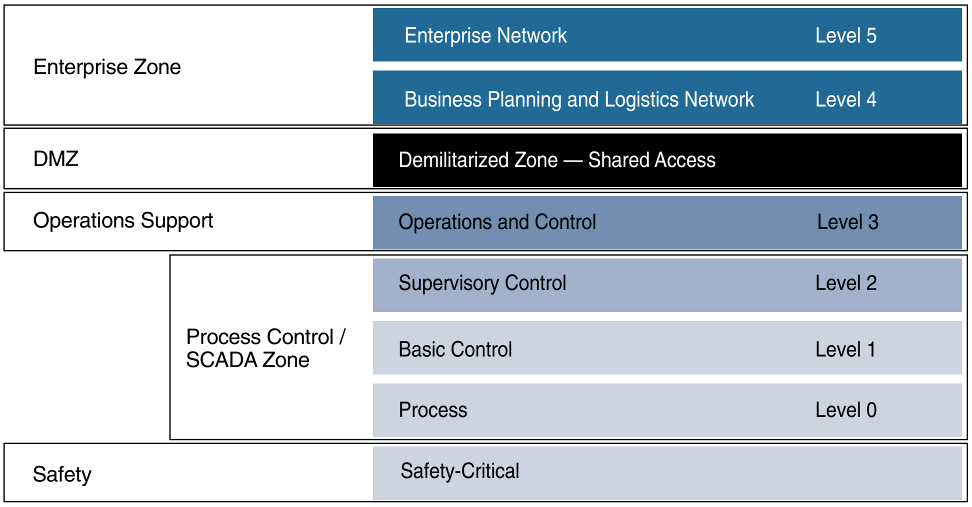


Figure 8-3 *The Logical Framework Based on the Purdue Model for Control Hierarchy*

This model identifies levels of operations and defines each level. The enterprise and operational domains are separated into different zones and kept in strict isolation via an industrial demilitarized zone (DMZ):

- Enterprise zone
  - **Level 5: Enterprise network:** Corporate-level applications such as Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), document management, and services such as Internet access and VPN entry from the outside world exist at this level.
  - **Level 4: Business planning and logistics network:** The IT services exist at this level and may include scheduling systems, material flow applications, optimization and planning systems, and local IT services such as phone, email, printing, and security monitoring.
- Industrial demilitarized zone
  - **DMZ:** The DMZ provides a buffer zone where services and data can be shared between the operational and enterprise zones. It also allows for easy segmentation of organizational control. By default, no traffic should traverse the DMZ; everything should originate from or terminate on this area.
- Operational zone
  - **Level 3: Operations and control:** This level includes the functions involved in managing the workflows to produce the desired end products and for monitoring and controlling the entire operational system. This could include production scheduling, reliability assurance, systemwide control optimization, security management, network management, and potentially other required IT services, such as DHCP, DNS, and timing.
  - **Level 2: Supervisory control:** This level includes zone control rooms, controller status, control system network/application administration, and other control-related applications, such as human-machine interface (HMI) and historian.
  - **Level 1: Basic control:** At this level, controllers and IEDs, dedicated HMIs, and other applications may talk to each other to run part or all of the control function.
  - **Level 0: Process:** This is where devices such as sensors and actuators and machines such as drives, motors, and robots communicate with controllers or IEDs.
- Safety zone
  - **Safety-critical:** This level includes devices, sensors, and other equipment used to manage the safety functions of the control system.

One of the key advantages of designing an industrial network in structured levels, as with the Purdue model, is that it allows security to be correctly applied at each level and between levels. For example, IT networks typically reside at Levels 4 and 5 and use security principles common to IT networks. The lower levels are where the industrial systems and

IoT networks reside. As shown in Figure 8-3, a DMZ resides between the IT and OT levels. Clearly, to protect the lower industrial layers, security technologies such as firewalls, proxy servers, and IPSs should be used to ensure that only authorized connections from trusted sources on expected ports are being used. At the DMZ, and, in fact, even between the lower levels, industrial firewalls that are capable of understanding the control protocols should be used to ensure the continuous operation of the OT network.

Although security vulnerabilities may potentially exist at each level of the model, it is clear that due to the amount of connectivity and sophistication of devices and systems, the higher levels have a greater chance of incursion due to the wider attack surface. This does not mean that lower levels are not as important from a security perspective; rather, it means that their attack surface is smaller, and if mitigation techniques are implemented properly, there is potentially less impact to the overall system. As shown in Figure 8-4, a review of published vulnerabilities associated with industrial security in 2011 shows that the assets at the higher levels of the framework had more detected vulnerabilities.

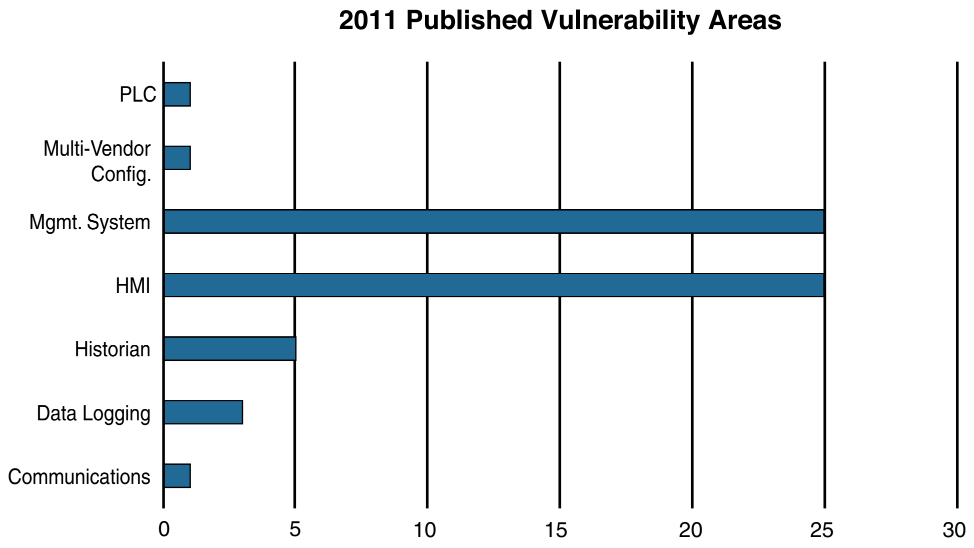


Figure 8-4 *2011 Industrial Security Report of Published Vulnerability Areas (US Industrial Control Systems Cyber Emergency Response Team (ICS-CERT) https://ics-cert.us-cert.gov).*

OT Network Characteristics Impacting Security

While IT and OT networks are beginning to converge, they still maintain many divergent characteristics in terms of how they operate and the traffic they handle. These differences influence how they are treated in the context of a security strategy. For example, compare the nature of how traffic flows across IT and OT networks:

- **IT networks:** In an IT environment, there are many diverse data flows. The communication data flows that emanate from a typical IT endpoint travel relatively far. They frequently traverse the network through layers of switches and eventually make their way to a set of local or remote servers, which they may connect to directly. Data in the form of email, file transfers, or print services will likely all make its way to the central data center, where it is responded to, or triggers actions in more local services, such as a printer. In the case of email or web browsing, the endpoint initiates actions that leave the confines of the enterprise network and potentially travel around the earth.

- **OT networks:** By comparison, in an OT environment (Levels 0–3), there are typically two types of operational traffic. The first is local traffic that may be contained within a specific package or area to provide local monitoring and closed-loop control. This is the traffic that is used for real-time (or near-real-time) processes and does not need to leave the process control levels. The second type of traffic is used for monitoring and control of areas or zones or the overall system. SCADA traffic is a good example of this, where information about remote devices or summary information from a function is shared at a system level so that operators can understand how the overall system, or parts of it, are operating. They can then implement appropriate control commands based on this information.

When IT endpoints communicate, it is typically short and frequent conversations with many connections. The nature of the communications is open, and almost anybody can speak with anybody else, such as with email or browsing. Although there are clearly access controls, most of those controls are at the application level rather than the network level.

In an OT environment, endpoint communication is typically point-to-point, such as a SCADA master to SCADA slave, or uses multicast or broadcast, leveraging a publisher/subscriber type of model. Communication could be TCP or UDP or neither (as in the case of PROFINET, discussed in Chapter 9, "Manufacturing").

Although network timing in the OT space typically mirrors that of the enterprise with NTP/SNTP used for device clocking against a master time source, a number of use cases require an extremely accurate clock source and extremely accurate time/synchronization distribution, as well as measurable and consistent latency/jitter. Some industrial applications require timing via IEEE 1588, PTP (Precision Time Protocol), so that information from source and destination can be accurately measured and compared at microsecond intervals with communication equipment introducing delays of no more than 50 nanoseconds. Jitter for the sending and receiving of information must also be minimized to ensure correct operation. By way of comparison, in the enterprise space, voice is often considered the highest-priority application, with a typical one-way delay of 150 milliseconds or more. In a number of operational environments for oil and gas, manufacturing, and power utilities, delay must be under 10 microseconds. Security attacks that cause delay, such as denial of service (DoS) attacks, can cause systems to malfunction purely by disrupting the timing mechanism.

IT networks are typically more mature and use up-to-date technologies. These mature modern networking practices are critical to meet the high degree of flexibility required in the IT environment. Virtual networking, virtual workspaces, and virtual servers are commonplace. It is likely that there are a wide variety of device types actively participating in any given network at any one time. Flexible interoperability is thus critical. To achieve interoperability, there is usually minimal proprietary communication activity, and the emphasis is typically on open standards. The movement to IPv6 continues to progress, and higher-order network services, such as quality of service (QoS), are normal as well. Endpoints are not just promiscuous in their communications, but they operate a wide number of applications from a large number of diverse vendors. The open nature of these compute systems means a wide range of protocols are traversing the OT network.

Industrial networks often still rely on serial communication technologies or have mixed serial and Ethernet. This means that not only do many devices lack IP capabilities, but it is not even possible to monitor and secure the serial traffic in the same way you do for IP or Ethernet. In some environments, the network remains very static, meaning a baseline of traffic patterns can be built up and monitored for changes. In static environments, the visibility of devices, protocols, and traffic flows can be managed and secured more easily. However, there is a continued growth of mobile devices and ad hoc connectivity, especially in industries such as transportation and smart cities, as well as a rise in mobile fleet assets across a plethora of other industries. These dynamic and variable networks are much more difficult to baseline, monitor, and secure.

Security Priorities: Integrity, Availability, and Confidentiality

Security priorities are driven by the nature of the assets in each environment. In an IT realm, the most critical element and the target of attacks has been information. In an OT realm, the critical assets are the process participants: workers and equipment. Security priorities diverge based on those differences.

In the IT business world, there are legal, regulatory, and commercial obligations to protect data, especially data of individuals who may or may not be employed by the organization. This emphasis on privacy focuses on the confidentiality, integrity, and availability of the data—not necessarily on a system or a physical asset. The impact of losing a compute device is considered minimal compared to the information that it could hold or provide access to. By way of comparison, in the OT world, losing a device due to a security vulnerability means production stops, and the company cannot perform its basic operation. Loss of information stored on these devices is a lower concern, but there are certainly confidential data sets in the operating environment that may have economic impacts, such as formulations and processes.

In an operational space, the safety and continuity of the process participants is considered the most critical concern. Thus, the goal is the continued uptime of devices and the safety of the people who operate them. The result is to emphasize availability, integrity, and confidentiality. The impact of loss here extends even to loss of life.

Security Focus

Security focus is frequently driven by the history of security impacts that an organization has experienced. In an IT environment, the most painful experiences have typically been intrusion campaigns in which critical data is extracted or corrupted. The result has been a significant investment in capital goods and humanpower to reduce these external threats and minimize potential internal malevolent actors.

In the OT space, the history of loss due to external actors has not been as long, even though the potential for harm on a human scale is clearly significantly higher. The result is that the security events that have been experienced have come more from human error than external attacks. Interest and investment in industrial security have primarily been in the standard access control layers. Where OT has diverged, to some degree, is to emphasize the application layer control between the higher-level controller layer and the receiving operating layer. Later in this chapter you will learn more about the value and risks associated with this approach.

Formal Risk Analysis Structures: OCTAVE and FAIR

Within the industrial environment, there are a number of standards, guidelines, and best practices available to help understand risk and how to mitigate it. IEC 62443 is the most commonly used standard globally across industrial verticals. It consists of a number of parts, including 62443-3-2 for risk assessments, and 62443-3-3 for foundational requirements used to secure the industrial environment from a networking and communications perspective. Also, ISO 27001 is widely used for organizational people, process, and information security management. In addition, the National Institute of Standards and Technology (NIST) provides a series of documents for critical infrastructure, such as the NIST Cybersecurity Framework (CSF). In the utilities domain, the North American Electric Reliability Corporation's (NERC's) Critical Infrastructure Protection (CIP) has legally binding guidelines for North American utilities, and IEC 62351 is the cybersecurity standard for power utilities.

The key for any industrial environment is that it needs to address security holistically and not just focus on technology. It must include people and processes, and it should include all the vendor ecosystem components that make up a control system.

In this section, we present a brief review of two such risk assessment frameworks:

- OCTAVE (Operationally Critical Threat, Asset and Vulnerability Evaluation) from the Software Engineering Institute at Carnegie Mellon University
- FAIR (Factor Analysis of Information Risk) from The Open Group

These two systems work toward establishing a more secure environment but with two different approaches and sets of priorities. Knowledge of the environment is key to determining security risks and plays a key role in driving priorities.

OCTAVE

OCTAVE has undergone multiple iterations. The version this section focuses on is OCTAVE Allegro, which is intended to be a lightweight and less burdensome process to implement. Allegro assumes that a robust security team is not on standby or immediately at the ready to initiate a comprehensive security review. This approach and the assumptions it makes are quite appropriate, given that many operational technology areas are similarly lacking in security-focused human assets. Figure 8-5 illustrates the OCTAVE Allegro steps and phases.

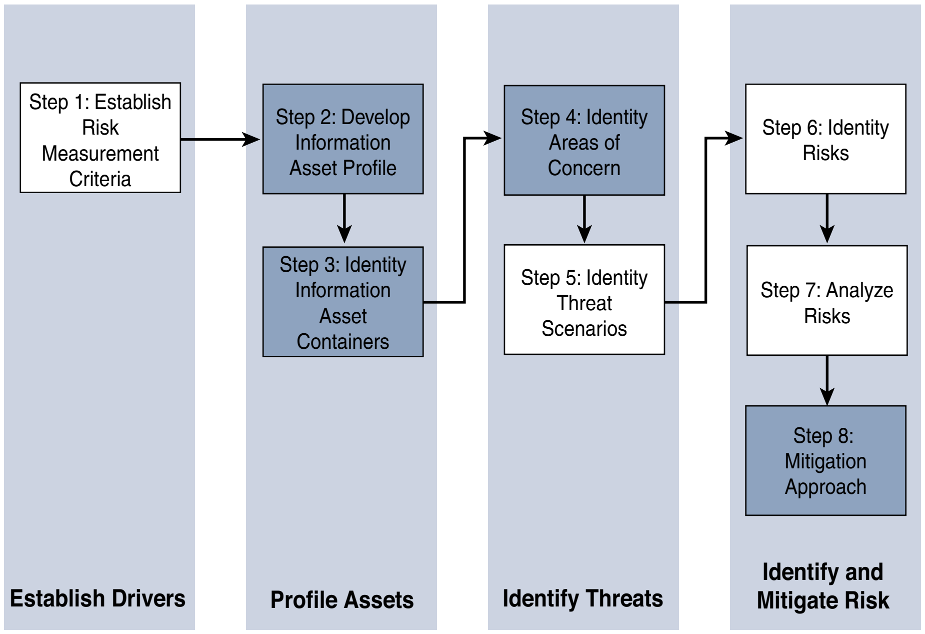

Step 1: Establish Risk Measurement Criteria
Step 2: Develop Information Asset Profile
Step 3: Identity Information Asset Containers
Step 4: Identity Areas of Concern
Step 5: Identity Threat Scenarios
Step 6: Identity Risks
Step 7: Analyze Risks
Step 8: Mitigation Approach
Establish Drivers
Profile Assets
Identify Threats
Identify and Mitigate Risk

Figure 8-5 *OCTAVE Allegro Steps and Phases (see https://blog.compass-security.com/2013/04/lean-risk-assessment-based-on-octave-allegro/).*

The first step of the OCTAVE Allegro methodology is to establish a risk measurement criterion. OCTAVE provides a fairly simple means of doing this with an emphasis on impact, value, and measurement. The point of having a risk measurement criterion is that at any point in the later stages, prioritization can take place against the reference model. (While OCTAVE has more details to contribute, we suggest using the FAIR model, described next, for risk assessment.)

The second step is to develop an information asset profile. This profile is populated with assets, a prioritization of assets, attributes associated with each asset, including owners, custodians, people, explicit security requirements, and technology assets. It is important to stress the importance of process. Certainly, the need to protect information does not disappear, but operational safety and continuity are more critical.

Within this asset profile, process are multiple substages that complete the definition of the assets. Some of these are simply survey and reporting activities, such as identifying the asset and attributes associated with it, such as its owners, custodians, human actors with which it interacts, and the composition of its technology assets. There are, however, judgment-based attributes such as prioritization. Rather than simply assigning an arbitrary ranking, the system calls for a justification of the prioritization. With an understanding of the asset attributes, particularly the technical components, appropriate threat mitigation methods can be applied. With the application of risk assessment, the level of security investment can be aligned with that individual asset.

The third step is to identify information asset containers. Roughly speaking, this is the range of transports and possible locations where the information might reside. This references the compute elements and the networks by which they communicate. However, it can also mean physical manifestations such as hard copy documents or even the people who know the information. Note that the operable target here is information, which includes data from which the information is derived.

In OCTAVE, the emphasis is on the container level rather than the asset level. The value is to reduce potential inhibitors within the container for information operation. In the OT world, the emphasis is on reducing potential inhibitors in the containerized operational space. If there is some attribute of the information that is endemic to it, then the entire container operates with that attribute because the information is the defining element. In some cases this may not be true, even in IT environments. Discrete atomic-level data may become actionable information only if it is seen in the context of the rest of the data. Similarly, operational data taken without knowledge of the rest of the elements may not be of particular value either.

The fourth step is to identify areas of concern. At this point, we depart from a data flow, touch, and attribute focus to one where judgments are made through a mapping of security-related attributes to more business-focused use cases. At this stage, the analyst looks to risk profiles and delves into the previously mentioned risk analysis. It is no longer just facts, but there is also an element of creativity that can factor into the evaluation. History both within and outside the organization can contribute. References to similar operational use cases and incidents of security failures are reasonable associations.

Closely related is the fifth step, where threat scenarios are identified. Threats are broadly (and properly) identified as potential undesirable events. This definition means that results from both malevolent and accidental causes are viable threats. In the context of operational focus, this is a valuable consideration. It is at this point that an explicit identification of actors, motives, and outcomes occurs. These scenarios are described in threat trees to trace the path to undesired outcomes, which, in turn, can be associated with risk metrics.

At the sixth step risks are identified. Within OCTAVE, risk is the possibility of an undesired outcome. This is extended to focus on how the organization is impacted. For more focused analysis, this can be localized, but the potential impact to the organization could extend outside the boundaries of the operation.

The seventh step is risk analysis, with the effort placed on qualitative evaluation of the impacts of the risk. Here the risk measurement criteria defined in the first step are explicitly brought into the process.

Finally, mitigation is applied at the eighth step. There are three outputs or decisions to be taken at this stage. One may be to accept a risk and do nothing, other than document the situation, potential outcomes, and reasons for accepting the risk. The second is to mitigate the risk with whatever control effort is required. By walking back through the threat scenarios to asset profiles, a pairing of compensating controls to mitigate those threat/risk pairings should be discoverable and then implemented. The final possible action is to defer a decision, meaning risk is neither accepted nor mitigated. This may imply further research or activity, but it is not required by the process.

OCTAVE is a balanced information-focused process. What it offers in terms of discipline and largely unconstrained breadth, however, is offset by its lack of security specificity. There is an assumption that beyond these steps are seemingly means of identifying specific mitigations that can be mapped to the threats and risks exposed during the analysis process.

FAIR

FAIR (Factor Analysis of Information Risk) is a technical standard for risk definition from The Open Group. While information security is the focus, much as it is for OCTAVE, FAIR has clear applications within operational technology. Like OCTAVE, it also allows for non-malicious actors as a potential cause for harm, but it goes to greater lengths to emphasize the point. For many operational groups, it is a welcome acknowledgement of existing contingency planning. Unlike with OCTAVE, there is a significant emphasis on naming, with risk taxonomy definition as a very specific target.

FAIR places emphasis on both unambiguous definitions and the idea that risk and associated attributes are measurable. Measurable, quantifiable metrics are a key area of emphasis, which should lend itself well to an operational world with a richness of operational data.

At its base, FAIR has a definition of risk as the probable frequency and probable magnitude of loss. With this definition, a clear hierarchy of sub-elements emerges, with one side of the taxonomy focused on frequency and the other on magnitude.

Loss even frequency is the result of a threat agent acting on an asset with a resulting loss to the organization. This happens with a given frequency called the threat event frequency (TEF), in which a specified time window becomes a probability. There are multiple sub-attributes that define frequency of events, all of which can be understood with some form of measurable metric. Threat event frequencies are applied to a vulnerability. *Vulnerability* here is not necessarily some compute asset weakness, but is more broadly defined as the probability that the targeted asset will fail as a result of the actions applied. There are further sub-attributes here as well.

The other side of the risk taxonomy is the probable loss magnitude (PLM), which begins to quantify the impacts, with the emphasis again being on measurable metrics. The FAIR specification makes it a point to emphasize how ephemeral some of these cost estimates can be, and this may indeed be the case when information security is the target of the discussion. Fortunately for the OT operator, a significant emphasis on operational efficiency and analysis makes understanding and quantifying costs much easier.

FAIR defines six forms of loss, four of them externally focused and two internally focused. Of particular value for operational teams are productivity and replacement loss. Response loss is also reasonably measured, with fines and judgments easy to measure but difficult to predict. Finally, competitive advantage and reputation are the least measurable.

Note The discussion of OCTAVE Allegro and FAIR is meant to give you a grounding in formal risk analysis processes. While there are others, both represent mechanics that can be applied in an OT environment.

The Phased Application of Security in an Operational Environment

It is a security practitioner's goal to safely secure the environment for which he or she is responsible. For an operational technologist, this process is different because the priorities and assets to be protected are highly differentiated from the better-known IT environment. The differences have been discussed at length in this chapter, but many of the processes used by IT security practitioners still have validity and can be used in an OT environment. If there is one key concept to grasp, it is that security for an IoT environment is an ongoing process in which steps forward can be taken, but there is no true finish line.

The following sections present a phased approach to introduce modern network security into largely preexisting legacy industrial networks.

Secured Network Infrastructure and Assets

Given that networks, compute, or operational elements in a typical IoT or industrial system have likely been in place for many years and given that the physical layout largely defines the operational process, this phased approach to introducing modern network security begins with very modest, non-intrusive steps.

As a first step, you need to analyze and secure the basic network design. Most automated process systems or even hierarchical energy distribution systems have a high degree of correlation between the network design and the operational design. It is a basic tenet of ISA99 and IEC 62443 that functions should be segmented into zones (cells) and that communication crossing the boundaries of those zones should be secured and controlled through the concept of conduits. In response to this, it is suggested that a security professional discover the state of his or her network and all communication channels.

Figure 8-6 illustrates inter-level security models and inter-zone conduits in the process control hierarchy.

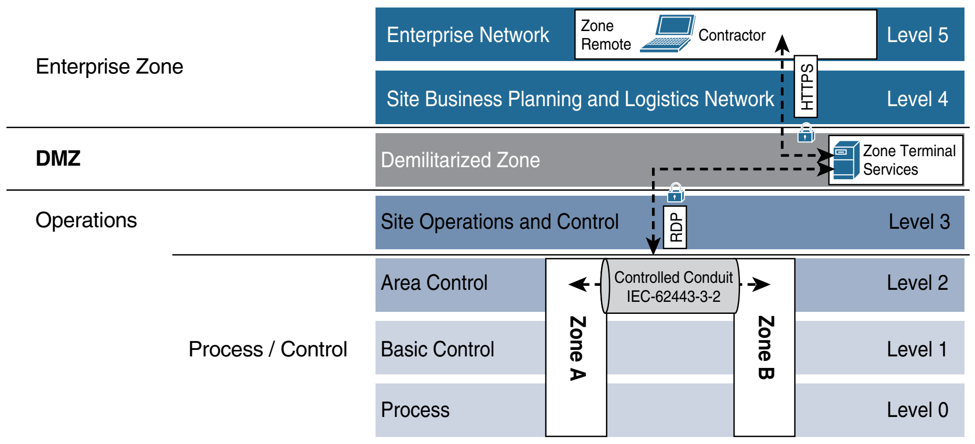


Figure 8-6 *Security Between Levels and Zones in the Process Control Hierarchy Model*

Normal network discovery processes can be highly problematic for older networking equipment. In fact, the discovery process in pursuit of improved safety, security, and operational state can result in degradation of all three. Given that condition, the network discovery process may require manual inspection of physical connections, starting from the highest accessible aggregation point and working all the way down to the last access layer. This discovery activity must include a search for wireless access points. For the sake of risk reduction, any on-wire network mapping should be done passively as much as possible.

It is fair to note that this prescribed process is much more likely to succeed in a smaller confined environment such as a plant floor. In geographically distributed environments, it may not be possible to trace the network, and in such cases, the long-haul connections may not be physical or may be carried by an outside communication provider. For those sections of the operational network, explicit partnering with other entities is required.

A side activity of this network tracing process is to note the connectivity state of the physical connections. This is not just an exercise to see what fiber or cables are in what ports but to observe the use or operational state of other physical connections, such as USB, SD card, alarm channel, serial, or other connections, at each network appliance. For more modern environments where updated networking devices and protocols are used, tools like NetFlow and IPFIX can also be used to discover the network communication paths.

As the network mapping reaches the aggregation point, it is worthwhile to continue to the connected asset level.

Normally, in an IT environment, the very first stage of discovery is focused on assets connected to the network. Assets remain critical, but from an efficiency and criticality perspective, it is generally recommended to find data paths into and between zones (cells) rather than the serial links between devices within a zone. One thing to continually be on the lookout for is the ever-dangerous, unsecured, and often undocumented convenience port.

Any physical port that is not physically locked down or doesn't have an enforceable protection policy is an uncontrolled threat vector.

Once the network is physically mapped, the next step is to perform a connectivity analysis through the switch and router ARP tables and DHCP requests within the network infrastructure. This should help further illuminate connectivity, good or bad, that has occurred. Firewall and network infrastructure data can contribute to understanding what devices are talking to other devices and the traffic paths over which this is done.

At this stage, the network should be reasonably well understood and prepared for secure connectivity.

Modern networking equipment offers a rich set of access control and secured communications capabilities. Starting at the cell/zone level, it is important to ensure that there is a clear ingress/egress aggregation point for each zone. If your communications patterns are well identified, you can apply access control policies to manage who and what can enter those physical portions of the process. If you are not comfortable explicitly controlling the traffic, then begin with alert-only actions. With time, you should be confident enough in your knowledge to apply controls.

At upstream levels, consider traffic controls such as denial of service (DoS) protection, traffic normalization activities, and quality of service (QoS) controls (such as marking and black-holing or rate-limiting scavenger-class traffic). The goal here is to ensure that these aggregated traffic segments are carrying high-priority traffic without impediment.

Network infrastructure should also provide the ability to secure communications between zones via secured conduits (see Figure 8-6). The primary method is encrypted communications in the form of virtual private networks (VPNs). VPNs can come in multiple forms, such as site-to-site, which would be appropriate between a utility substation and a control center, or perhaps in cell-to-cell communications. Remote access controls can be established in more ad hoc situations and utilize the convenience of browser-based VPNs with Secure Sockets Layer (SSL)–based VPNs. If latency concerns are not particularly high, you can use Media Access Control Security (MACSec) hop-by-hop encryption to allow for potential controls and visibility at key junctions.

The next discovery phase should align with the software and configurations of the assets on the network. At this point, the rights and roles of the network administrator may be insufficient to access the required information. Certainly, the network infrastructure and its status are within the network admin's view, but the individual assets likely are not. At this point, organizational cooperation is required for success. For an experienced IT-based network practitioner, this is not an unusual situation. It is very common, especially in larger enterprises, to see a separation of responsibilities and controls between the communications transport and the assets to which they are connected. At the operations level, similar cooperation is required with those responsible for the maintenance of the OT assets.

There are reasonable sources of information describing the configuration state of OT assets. The control systems associated with the processes hold historical data describing what is connected and what those assets are doing. A review of historical data should provide an idea of what assets are present and what operations are being performed on them, and it should identify such things as firmware updates and health status. The volume of data to analyze may be challenging, but if it is organized correctly, it would be valuable for understanding asset operation.

With an initial asset inventory completed, you can initiate a risk analysis based on the network and assets, and determine an initial scope of security needs.

Deploying Dedicated Security Appliances

The next stage is to expand the security footprint with focused security functionality. The goal is to provide visibility, safety, and security for traffic within the network. Visibility provides an understanding of application and communication behavior. With visibility, you can set policy actions that reflect the desired behaviors for inter-zone and conduit security.

While network elements can provide simplified views with connection histories or some kind of flow data, you get a true understanding when you look within the packets on the network. This level of visibility is typically achieved with deep packet inspection (DPI) technologies such as intrusion detection/prevention systems (IDS/IPS). These technologies can be used to detect many kinds of traffic of interest, from simply identifying what applications are speaking, to whether communications are being obfuscated, to whether exploits are targeting vulnerabilities, to passively identifying assets on the network.

With the goal of identifying assets, an IDS/IPS can detect what kind of assets are present on the network. Passive OS identification programs can capture patterns that expose the base operating systems and other applications communicating on the network. The organizationally unique identifier (OUI) in a captured MAC address, which could have come from ARP table exploration, is yet another means of exposure. Coupled with the physical and historical data mentioned before, this is a valuable tool to expand on the asset inventory without having to dangerously or intrusively prod critical systems.

Application-specific protocols are also detectable by IDS/IPS systems. For more IT-like applications, user agents are of value, but traditionally, combinations of port numbers and other protocol differentiators can contribute to identification. Some applications have behaviors that are found only in certain software releases. Knowledge of those differences can help to determine the software version being run on a particular asset.

Within applications and industrial protocols are well-defined commands and, often, associated parameter values. Again, an IDS/IPS can be configured to identify those commands and values to learn what actions are being taken and what associated settings are being changed.

All these actions can be done from a non-intrusive deployment scenario. Modern DPI implementations can work out-of-band from a span or tap. Viewing copies of packets has no impact on traffic performance or latency. It is easily the safest means of getting deep insight into the activities happening on a network.

Visibility and an understanding of network connectivity uncover the information necessary to initiate access control activity. Access control is typically achieved with access control lists (ACLs), which are available on practically all modern network equipment. For improved scalability, however, a dedicated firewall would be preferred. Providing strong segmentation and zone access control is the first step. Access control, however, is not just limited to the typical address and protocol identifiers. Modern firewalls have the ability to discern attributes associated with the user accessing the network, allowing controls to be placed on the "who" element also. In addition, access control can be aligned with applications and application behaviors. Equipped with the right toolset, a modern OT practitioner can ensure that only those operators in a certain user class can initiate any external commands to that particular asset.

Safety is a particular benefit as application controls can be managed at the cell/zone edge through an IDS/IPS. The same technologies that observe the who and what can also manage the values being passed to the target asset. For example, in a manufacturing scenario where a robot operates, there may be an area frequented by workers who are within the potential range of the robot's operation. The range is unique to the physical layout of the cell, and parameter changes could cause physical harm to a plant worker. With an IDS/IPS, the system can detect that a parameter value exceeds the safety range and act accordingly to ensure worker safety.

Safety and security are closely related linguistically (for example, in German, the same word, Sicherheit, can be used for both), but for a security practitioner, security is more commonly associated with threats. Threat identification and protection is a key attribute of IPSs using DPI.

Mature IPSs have thousands of threat identifiers, which address the complete range of asset types where remotely exploitable vulnerabilities are known. In some cases, the nature of the threat identifier is generic enough that it addresses a common technique without having to be associated with a particular application instance of the vulnerability type.

Placement priorities for dedicated security devices vary according to the security practitioner's perception of risk. If visibility is incomplete and concern dictates that further knowledge is necessary prior to creating a proactive defense, the security device should be placed where that gap is perceived. It is important to note that the process of gaining visibility or addressing risk is dynamic. Networks change, and as knowledge is gained, new priorities (either in the form of visible threats or a reduction of gaps) creates new points of emphasis. Given this dynamism, consider the idea that placement of a dedicated security device can change as well. In other words, just because you start with a device in one location does not mean you can't move it later to address security gaps.

Inevitably a decision must be made. Here we discuss some of the relative merits of different placement locations. Placement at the operational cell is likely the most fine-grained deployment scenario. By *fine-grained* we mean that it is the lowest portion of a network that gives network-based access to the lowest level of operational assets. As discussed earlier, the nature of the deployment—out-of-band or in-line—depends on the organization's comfort level for in-line operation and desire to actually exert control. In either case, the industrial security appliance should be attached directly to the switch, which denotes the access point into the cell. This location gives the greatest level of control for safety controls, visibility, and threats. If network design has properly segmented to a single zone entry point, then this is an optimal deployment location. For safety considerations, application control can be exerted to ensure that application changes will not allow for dangerous settings. Threats can be mitigated as they traverse the device, and traffic entering and exiting the cell can be made visible.

A particularly valuable function is enabled if a security device can terminate VPNs in addition to performing deep packet inspection. Secured communication, potentially from a vendor representative outside the organization, can be terminated at the ingress to the device and then inspected. The time cost of the termination would be similar to what would be done on the switch, and then inspection of what that remote user accessing the network is doing is viable. Naturally, any potential threat traffic can be halted as well.

If the zone/cell houses critical infrastructure and remote operation is requisite, a redundant high-availability configuration for both the network and security infrastructure is advised.

For the purposes of pure visibility, hanging off a mirror or span port from the switch would be optimal. For control capabilities, one must be in-line to truly act on undesired traffic. In most cases, the preferred location is upstream of the zone/cell access switch between the aggregation layer and the zone switch. It may be viable to have the security device between the zone assets and the zone access switch as well.

For broader, less detailed levels of control, placement of dedicated security devices upstream of the aggregation switches is the preferred approach. If the network has multiple zones going through the aggregation switch with mostly redundant functionality but with no communication between them, this may be a more efficient point of deployment.

At some point, a functional layer above the lowest zone layer becomes connected to the network, and there should be a device located between those functions and their OT charges in the zones/cells. At that next layer up, there may be HMIs or other lower-level operational tools. For safety considerations, a control point between that layer and the cell is valuable.

At the higher level of the network are a good number of higher-function assets, such as standard network elements (for example, directory servers, network monitoring tools, remote access plus proxy servers, print servers, security control elements). More operationally focused functionality involves elements such as engineering workstations and operations control applications. Depending on the diversity and network topologies at play, these operational structures could be replicated within their own subzones (subnets) at the same level. There may be justification for using a dedicated security device between the subzones, depending on the need to control access, but for the most part, this is a zone that needs controls placed above and below.

Below is where industrial awareness and, potentially, hardware ruggedization is more likely to be needed. With some amount of industrial traffic traversing this layer, a dedicated and security-aware tool would be advisable.

Above this highest level, a dedicated security device with IT-centric threat controls is recommended. If the applications hosted here are similar in nature to those found in IT environments (for example, Windows- or Linux-based applications), this requires common networking infrastructure, web-based access, and so on for proper visibility, control, and protection. Applying such controls to all ingress points (above and below) is important. There should be no assumptions made that an IT-centric threat can only emanate from the IT/enterprise layer above the DMZ. Attackers would not limit themselves to such thinking.

There is evidence that end-of-life OS and software components exist in operational environments. An all-too-common and unfortunate attribute of such systems is that further patching for security vulnerabilities is likely unavailable. To protect those systems after their official end-of-support date, the concept of a "virtual patch" layer may be possible. The idea is that protections for vulnerabilities can be applied through the network path by which these systems communicate. While this is not a substitute for keeping abreast of patching, it may be a mitigation approach that fits your organization's risk acceptance policy.

At the logical edge of the operational space is the DMZ (demilitarized zone)—a security boundary between two diverged compute realms. Assets in this area are meant to bridge communications in a secure fashion between the enterprise's IT realm and the industrial OT realm. Security should be applied both above and below this layer.

Before we leave the second phase of operational security, it is important to reemphasize that security, in whatever location, is an ongoing process. The policies applied and the knowledge gained should never stagnate. Conditions will inevitably change, so security deployments and sometimes networks themselves must change to adapt. Where you place your security enforcement products and the policies they employ must be ready to change with them.

Higher-Order Policy Convergence and Network Monitoring

So far we have focused on very basic concepts that are common and easily implemented by network engineering groups. Finding network professionals with experience performing such functions or even training those without prior experience is not difficult.

Another security practice that adds value to a networked industrial space is convergence, which is the adoption and integration of security across operational boundaries. This means coordinating security on both the IT and OT sides of the organization. Convergence of the IT and OT spaces is merging, or at least there is active coordination across formerly distinct IT and OT boundaries. From a security perspective, the value follows the argument that most new networking and compute technologies coming to the operations space were previously found and established in the IT space. It is expected to also be true that the practices and tools associated with those new technologies are likely to be more mature in the IT space.

There are advanced enterprise-wide practices related to access control, threat detection, and many other security mechanisms that could benefit OT security. As stated earlier, the key is to adjust the approach to fit the target environment.

Several areas are more likely to require some kind of coordination across IT and OT environments. Two such areas are remote access and threat detection. For remote access, most large industrial organizations backhaul communication through the IT network. Some communications, such as email and web browsing, are obvious communication types that are likely to touch shared IT infrastructure. Often vendors or consultants who require some kind of remote access to OT assets also traverse the IT side of the network. Given this, it would be of significant value for an OT security practitioner to coordinate access control policies from the remote initiator across the Internet-facing security layers, through the core network, and to a handoff point at the industrial demarcation and deeper, toward the IoT assets. The use of common access controls and operational conditions eases and protects network assets to a greater degree than having divergent groups creating ad hoc methods. Using location information, participant device security stance, user identity, and access target attributes are all standard functions that modern access policy tools can make use of. Such sophistication is a relatively new practice in industrial environments, and so, if these functions are available, an OT security practitioner would benefit from coordination with his or her IT equivalents.

Network security monitoring (NSM) is a process of finding intruders in a network. It is achieved by collecting and analyzing indicators and warnings to prioritize and investigate incidents with the assumption that there is, in fact, an undesired presence.

The practice of NSM is not new, yet it is not implemented often or thoroughly enough even within reasonably mature and large organizations. There are many reasons for this underutilization, but lack of education and organizational patience are common reasons. To simplify the approach, there is a large amount of readily available data that, if reviewed, would expose the activities of an intruder.

It is important to note that NSM is inherently a process in which discovery occurs through the review of evidence and actions that have already happened. This is not meant to imply that it is a purely postmortem type of activity. If you recognize that intrusion activities are, much like security, an ongoing process, then you see that there is a similar set of stages that an attacker must go through. The tools deployed will slow that process and introduce opportunities to detect and thwart the attacker, but there is rarely a single event that represents an attack in its entirety. NSM is the discipline that will most likely discover the extent of the attack process and, in turn, define the scope for its remediation.

Summary

As industries modernize in pursuit of operational efficiencies, improved safety, and competitive agilities, they must do so securely. Modernization processes frequently initiate greater connectivity in the context of older and highly vulnerable OT assets and processes. Security is a process that must be applied throughout the lifecycle of that change and operation. To achieve security, an organization must be able to define risks and make informed choices about how best to address them.

Fortunately, much of what is available to minimize risks from threats is readily available. Network connectivity can be made secure with the right equipment and policies. Threats from unsafe practices, attacks, and remote access needs can be identified and controlled with dedicated industrial security appliances and practices. With time, there are opportunities to expand risk reduction through convergence and cooperation. Learning from the more extensive and mature security practices and tools in IT environments as well as coordinating layers of defense to protect critical industrial assets are key security enablers for operational environments.