codit
integrating your business

white paper

INTEGRATION OF THINGS

WHY AND HOW ORGANIZATIONS CAN REUSE INTEGRATION CONCEPTS AND PRACTICES TO BUILD INTERNET OF THINGS SOLUTIONS

TABLE OF CONTENTS

1. ABOUT CODIT

Large, international companies often struggle to easily exchange data with their subsidiaries, customers, suppliers and other business partners. Many also face challenges with the upcoming trends such as Cloud, SaaS apps, Mobile, Internet of Things, Big Data...
Companies not only have to be aware of them, yet have to adopt these technology changes strategically to gain competitive advantage. That is exactly what our expert teams do: we integrate business applications with the newest Microsoft technologies.

AUTHOR

Tom Kerkhove | Azure Consultant

REVIEWERS

I would like to thank Sam Vanhoutte, CTO at Codit and Glenn Colpaert, Azure/IoT Domain Lead at Codit for reviewing this white paper.

2. IN THIS WHITE PAPER

In 2015, Internet of Things (IoT) was one of the hottest topics in the industry. It was hyped by a lot of companies. However, Internet of Things is more than a hype and is here to stay - research reports envision that by 2020 almost everything will be connected to the internet. Doing so will make our lives easier and more enjoyable, automate production pipelines in factories, build smart houses that are aware of the environment, and so on.

Nonetheless, such solutions need to be scalable to meet the rising demand, secure to prevent tampering or harm, and last but not least they need to be remotely managed to avoid manual interventions. Building such solutions need to be carefully planned, designed, implemented and deployed with these challenges in mind.

As Codit is building integration solutions since its inception in 2000, we believe that a lot of the same concepts and patterns can be reused to build Internet of Things solutions. However, Internet of Things ecosystems also have a lot of differences compared to a traditional integration application. **This white paper will highlight why and how System Integrators can reuse existing concepts and practices to build Internet of Things solutions.** It is Codit's core business to integrate all things.

3. EXPLORING THE IOT VALUE CHAIN

The goal of integration solutions is to connect multiple systems or multiple parties, allowing them to collaborate and work together by using a **centralized middleware**. Since each connected system typically has its own set of supported protocols, the middleware platform is in charge of performing protocol translation so that they can communicate. Typically, integration platforms provide **adapters**, allowing flexible choice over the protocol(s) that will be used.

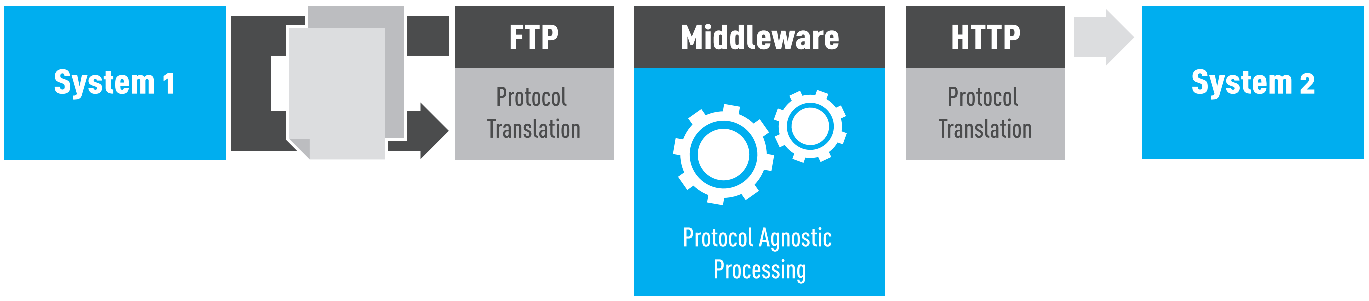


Figure: Simple integration with "connected systems"

The goal of Internet of Things is to connect 'things', which mostly acts as **event producers**, rather than systems. The Internet of Things platform will collect specific information and analyze it in the cloud to learn from the data. It is possible to interact with the devices in the field when a certain event happens, or proactively when we want to prevent something from malfunctioning. By doing so, businesses can improve their customer services, optimize their business processes, automate production pipelines, et cetera.

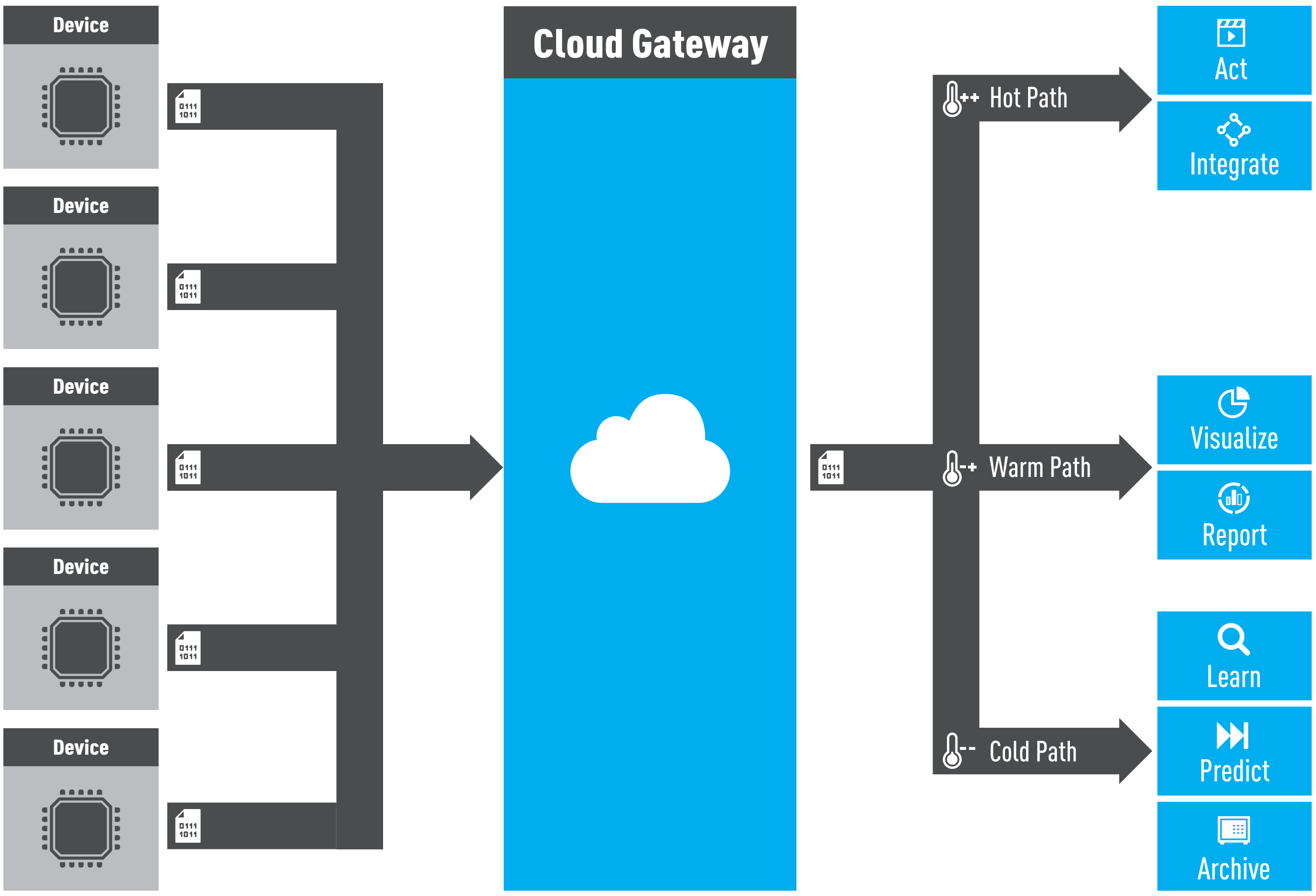


Figure: Basic figure of IoT, Multiple devices, Learn & integrate with processes, etc.

A typical scenario is where corporations install motion sensors in rooms to detect if people are in the room. If not, the IoT intelligence can send a shutdown command to the heating system in that room to optimize costs. You can also take it one step further and integrate the booking system for meeting rooms. If there is a scheduled meeting, the system will begin heating up the room 15 minutes in advance. And if no-one shows up, the system can stop the heating since this is no longer needed.

3.1. EVENT PRODUCERS & GATEWAYS

The 'things' are mostly physical units that are capable of measuring certain data and sending the info to a **cloud gateway** for further processing. They are the event producers in the IoT landscape. These event producers can range from very small dumb devices to intelligent on-board devices running a full OS like Windows 10 IoT or Linux.

The cloud gateway handles communication with all the devices in the field. Gateways are typically capable of performing **protocol translation** for several protocols. AMQP, MQTT and HTTPS are frequently used. Unfortunately, not all event producers are capable of speaking these protocols.

Field gateways are commonly used (especially in industrial IoT) and they can bridge the existing (local) protocols of devices to one of the protocols mentioned above. They often connect multiple devices over one single connection to the cloud gateway. Using field gateways also provides the capability to perform edge analytics.

For example, aggregation can be done before sending the data to the cloud gateway. A field gateway also takes away the complexity of buffering, security, protocol updates and so on away from the actual device firmware.

One of the biggest benefits of using field gateways is that they add an additional layer of security because they isolate event producers from the public. This also ensures that communication with the cloud can happen/is possible by using a protocol that is secure and doesn't harm the devices.

If we compare this with traditional integration, it is clear that both scenarios (integration and IoT) need to be able to use several protocols to connect to "external" entities, whether these are business partners, applications or devices.

3.2. INGESTION & TRANSFORMATION

On a high-level, event producers and gateways basically use event streaming technologies to create **data streams** into the cloud allowing them to send events at a high velocity. The advantage of using data streams is that you can interpret the information by using a cursor-based reading, allowing it to be played back again. This typically speeds up performance, compared to peek-lock patterns often found in messaging and integration systems.

Often, multiple data streams are being merged and cross-referenced so the data can be combined, correlated and transformed before we start the real analysis. These other streams can be data from other types of event producers, reference data such as weather information, or even output from our own intelligence services.

Connecting devices and processing the incoming events can take a lot of work. But it is just the start. Later on we'll see how we can use different types of data analytics to build our own intelligent platform. Once the data flows in the system via the cloud gateways we can transform it, for example by using aggregation. After that we can use adapters to output the data into a variety of data stores or other streams for further processing, reporting or reacting.

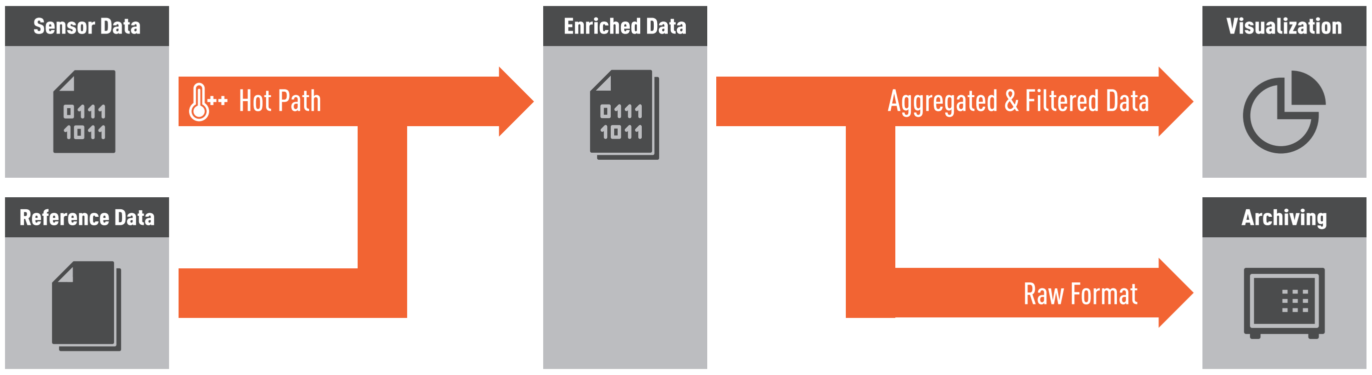


Figure: Multiple data streams, transforming data, data aggregation

With traditional integration, we use a centralized broker that is in charge of receiving and storing messages until they are being delivered. This can be a simple queue or a more advance pub/sub technology. However, with Internet of Things we've seen that a streaming approach is most commonly used to meet the scalability needs of thousands or millions of event producers.

On the other hand, the same approach is being used in traditional integration – both scenarios receive data, transform it and output it to a different system by using adapters.

Event streams are often using a log-based approach to leverage playback functionality. If you want to learn more about this, I recommend reading I Heart Logs by Jay Kreps, co-creator of Kafka.

3.3. REPORT, ACT & PREDICT

With all the telemetry data in the cloud, a variety of scenarios is possible. These range from visualizing and reporting to reacting and starting **business processes and the prediction of the future** – giving your business the added value that is promised.

Visualizing the collected telemetry data can be interesting to gain insight on how your landscape is performing, how your systems are working and how evolution happens over time. Prediction allows corporations to react on scenarios that haven't happened before. On the other hand, the telemetry data can be used to start automating business processes and be more cost efficient.

In traditional integration, the focus is mainly on exchanging information between different parties and not specifically on interpreting the data itself, this is typically done by another system like ETL. That said, there are integration scenarios that trigger data as well, to do forecasting or trigger processes such as handling shipments.

4. BUILDING AN ECOSYSTEM INSTEAD OF MANAGING ARCHITECTURES

With an **Integration Landscape**, one team is typically in charge of building and maintaining the central hub that connects the systems. These systems are either provided and managed by an external party or by an internal team that is dedicated to the system, such as a CRM system.

This means that the integration team is only in charge of connecting to these systems and making sure that they can communicate with each other.

However, in an IoT scenario it is more a matter of building an **IoT ecosystem** rather than managing a landscape. While there is still a centralized **Cloud Hub** in the ecosystem, there is also a need for a **Device Management Component**. This component is in charge of provisioning, monitoring and managing all the devices that are installed in the field. They can be either on a fixed location or roaming.

The **Device Management Component** leverages at least the following capabilities:

- **Authentication**: each device gets a dedicated authentication key that can be used to authenticate with device management and cloud hub
- **Identification**: the service generated a unique identifier for each device
- **Monitoring**: keeping track of device information like current firmware version, application version, battery life, connection status, etc.
- **Job scheduling and orchestration**: remotely pushing firmware updates, deploying new application versions or pushing new configurations to (parts of) the device eco system
- **Revoking Access**: the capability to revoke access for a specific device
- **Provisioning**: providing a provisioning flow to enable a new device to start communicating with the cloud hub

While the device management module is in charge of defining unique devices and performing operational tasks, it can be beneficial to maintain a **Device Catalog module**. This module will consolidate all business-specific information about the device such as customer information, address of the device, etc.

This leverages a clear distinction between storing operational information in the device management module and keeping business-related metadata in the device catalog.

4.1. A CLOSER LOOK AT DEVICE PROVISIONING

The Device Provisioning flow allows a new device to start sending telemetry to the cloud gateway once it is physically installed and authorized by the cloud gateway. To successfully achieve this flow, it is recommended to provide a **Provisioning API** as part of the device management module that is solely in charge of registering and activating new devices.

4.1.1. REGISTERING A NEW DEVICE

When a new device is manufactured, it must have an ID that can be used to uniquely identify that specific device. We will call this the **Device ID**.

The device can be provisioned via the provision API by specifying the Device ID, a specific **Activation Key** and business-specific information, if applicable. That activation key will be used when the device is physically installed and requesting permissions to start sending.

Behind the scenes, the provision API will register this new device in your device management module of the cloud gateway as well as in your device catalog, if you have one.

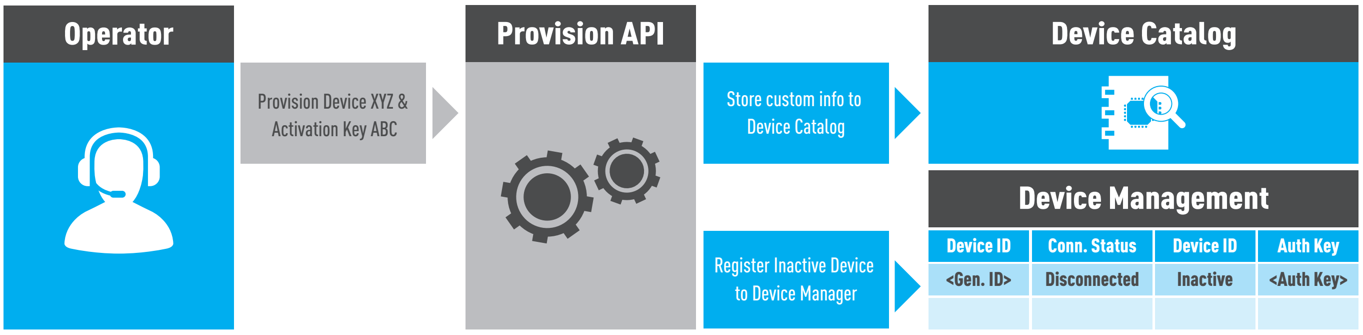


Figure: Operator registers a new device in Device Management via the Provision API

Once the provisioning is done, both the device management module and the device catalog module contain information about the new device - although it is not yet activated.

By using this activation flow, the device management module can hold off issuing a new authorization key until the device is deployed and actually has a need of authenticating itself with the cloud gateway. This also reduces the risk of leaking an authorization key or abuse by an intruder.

4.1.2. ACTIVATING A DEVICE

After a device has been deployed and configured, it can activate itself by using the provision API and specifying its unique device ID and activation ID.

The provision API will verify the specified information, if it matches it will activate the device in the cloud gateway and issue a new authentication key for that specific device ID.

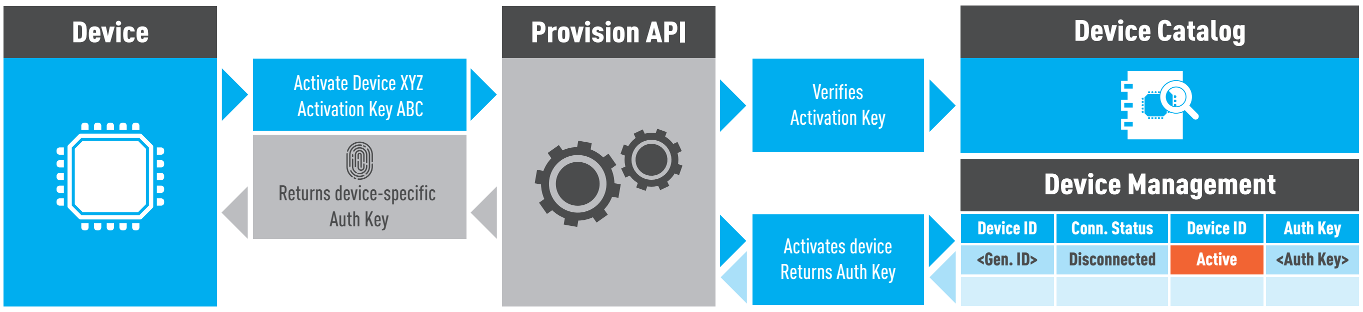


Figure: Device activates itself via Activation Key over Provision API

Optionally, the deployment engineer can manually activate the device via a similar flow but this all depends on what the business needs are.

4.1.3. SECURELY STORE KEYS

It goes without saying that the activation and authentication key should be securely stored on the device. This can be achieved by using a **Trusted Platform Module (TPM)** to store the secrets and/or by burning the activation key directly in the silicon during manufacturing.

4.2. BUILDING AN ECOSYSTEM IN MICROSOFT AZURE

When building an ecosystem on Microsoft Azure, Azure IoT Hub is the perfect fit to use as a cloud gateway. It leverages crucial features such as device management out-of-the-box, message routing, ability to schedule jobs such as device updates, etc.

Azure Event Hubs would be a good alternative for telemetry-only scenarios but Azure IoT Hub brings a lot more to the table. One thing is a unique identity per device that can be revoked when needed. That said, Azure Event Hubs is perfect for the scalability that is needed in IoT scenarios. It is often used down the pipeline to store real-time analytics results.

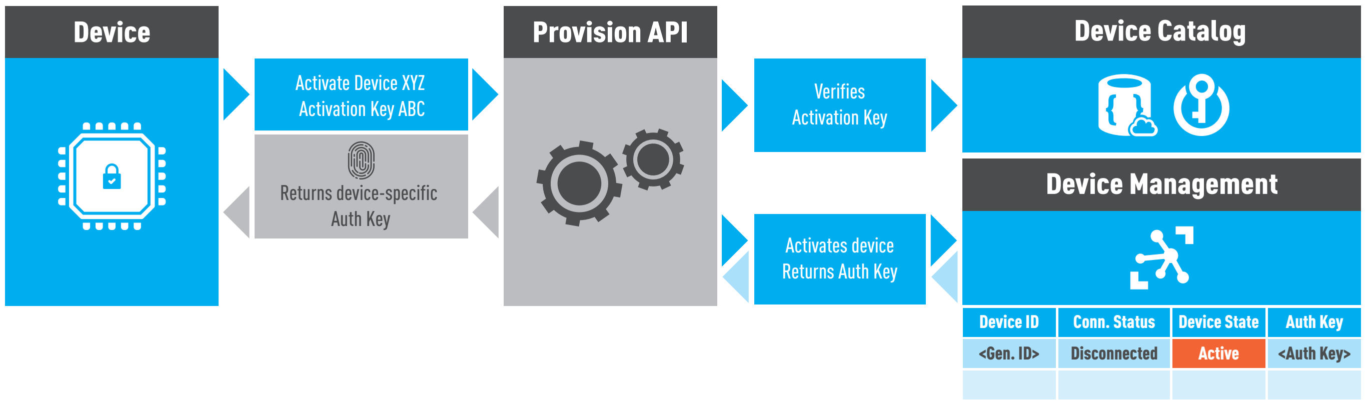
The **Provision API** is a great way to build a single endpoint for all operations related to the devices itself. It understands which modules need to be consulted for each operation, which makes the device flow a lot easier because it is abstracted away from them. Operations can be done by an engineer that provisions the new devices, the devices can activate themselves, or the metadata can be updated for a specific device.

The provision API can be hosted as an **Azure Web App** or **API App** that is exposed to devices directly, or by using Azure API Management to add an extra layer of security. This allows the API itself to use certificate authentication that is handled by **Azure API Management** where the devices authenticate with a subscription ID towards Azure API Management.

Unfortunately, using Azure IoT Hub as a cloud gateway doesn't always cover all requirements. It leverages custom properties per device. This is fine, but often it is better to have a dedicated device catalog module which consolidates all business-related metadata along with the corresponding device ID in IoT Hub.
The device catalog module can be any data store that is preferred. **Document DB** is a good choice because it allows you to store a JSON document for each device; which offers more flexibility in the structure that is required. Yet, this also means that you have to maintain the module and keep the data up to date.

The cloud gateway may need to store sensitive information, whether those are secrets for the platform or per device, they should be handled with care. If this is the case, Azure Key Vault is a perfect fit for handling your secrets, private keys or certificates as part of your platform. It goes without saying that the device catalog shouldn't store any secrets either but offload them to **Azure Key Vault**. Authentication keys for Azure IoT Hub should never be stored outside of IoT Hub itself.

Below, you'll find an example of how a device activates itself by using the provision API. Behind the scenes, the provision API goes through all the required steps to activate the device by using the device management & catalog modules.



5. STORING YOUR DATA

Both integration and IoT are working with data but each in a different way.

In traditional integration scenarios, data is being exchanged between parties to interact with each other, automate business processes, etc. or to store it and run a small notion of analytics on top of it. IoT, however, is entirely based on the collection and analysis of data to gain insights on how your business is behaving and how to make it more intelligent.

If we have a closer look at the traditional integration scenarios, the usage of different data stores is very limited because the system is mainly in charge of integrating several systems with each other, each with their own data store. However, it also occurs that the heart of the system needs to interact with one or more data stores. This is more for operational purposes or to make information available for internal analysis by a dedicated team. These stores are typically relational databases, data warehouses or the file system itself.

The data used in the IoT ecosystemdeals with a variety data sets coming in different sizes, structures, types and needs across a variety of data stores and are being used for different reasons. Choosing the correct data store for the correct data structure and analysis is crucial.

5.1. DIFFERENT KINDS OF DATA

When we talk about different kinds of data it is more about what the data allows us to do.

Business data	Reference data	Operational data
Contains the real-business value and can be the raw device telemetry or the output of one or more analytics experiments. In conjunction with reference data, businesses use it for decision making.	Is used during analysis to enrich the business data by using other (external) data sets ranging from static flat files with geolocation information to external weather information.	Is used to keep our system up and running. This can either be configuration information, auditing information about the system itself, application telemetry, etc.

While in integration these kinds of data exist to a certain notion as well, in an IoT ecosystem it is the heart of the system. It is crucial to be aware of what the type of a specific data set is, in order to store it in the most optimal way.

5.2. HANDLING DIFFERENT TYPES OF DATA

Besides the differentiation of kinds of data, data sets also differ from each other in manner of velocity, granularity and lifetime in IoT scenarios.

We can distinguish them into the following categories:

Hot Path	Warm Path	Cold Path
In the Hot Path, data comes in at a high velocity representing device telemetry, typically in small payloads that are often aggregated by a field gateway. This data set only lives for a short period, seconds rather than minutes. It is frequently used for near-real-time analytics.	The Warm Path contains aggregates of the hot data and is thus less specific but it serves as a summary of the last minutes and is generally used for dashboarding.	The Cold Path contains archived data that is less frequently accessed and precise since it's aggregated but still represents the complete history of our ecosystem. It will typically be used for more deep learning such as batch processing.

In an integration scenario, you'll never see this distinction since the integration either doesn't perform any analysis or it is done by a dedicated team outside of the integration context. It will simply receive the data, optionally change the format by using specific mappings, and persist it in a specific data store.

5.3. CHOOSING THE RIGHT STORAGE MODEL

Choosing the correct data store model for a specific data set is hard, since every data store has its benefits and drawbacks.

As previously mentioned, both IoT & integration often make use of the same data stores:

- A simple **File Store** to store BLOBs on the file system that can be structured or unstructured. This can be used to store all kinds of information, certainly for operational & reference data.
- **Relational Databases** and **Data Warehouses** that enable you to store structured data that is pre-defined by a schema.
- Data needs to be cleaned first before being stored. This means that data attributes will be removed because they are not relevant and thus the value of the data set is being limited.
- While relational databases are being used for On-Line Transaction Processing (OLTP) on smaller data sets, data warehouses preferred for storing and performing heavy analytics with On-Line Analytical Processing (OLAP) on bigger data sets.
- Relational databases are used for rather hot to warm data while data warehouses are used for warm to cold data.

Since IoT fully relies on the analytics of the data, a couple of new data store types are used to tackle a few issues:

- **Stream-based Stores** are immutable data streams that are used to send data in a high velocity manner. In an IoT scenario, they are typically used to send telemetry information to the system and making it available for stream processing in near-real-time. This approach typically uses an append-log approach.
- **Document Stores** are NoSQL stores used to store schema-less documents. While these are not used for storing business data, they are perfect for storing operational data such as device metadata and reference data.
- **Data Lakes** are designed to store data sets without any limitations in size, structure or format leveraging a solid platform that is optimized for running analytics on top of it. This is used to store business data, reference data, etc. The biggest change is that, in contrast with a data warehouse, you don't need to define a schema for the data sets, thus storing all the data attributes. This enables data scientists to analyze and work on the full data set and avoids removing potential business value that is not relevant at the moment, but can be in the future. Since there is no structure at all in a data lake, it is crucial to use some sort of **Data Catalog** that keeps track of what each data set is, where it comes from, who owns it, etc. If this is not provided your data lake will turn into a Data Swamp.

  Martin Fowler wrote an article that explains this concept and how it differs from a data warehousing approach. I recommend reading it here: http://martinfowler.com/bliki/DataLake.html.

5.4. STORING DATA IN MICROSOFT AZURE

Microsoft Azure provides a rich set of services for storing data.

- **Azure SQL Database** is basically a managed relational database running in Azure that uses the same engine as Microsoft SQL Server. It also comes with a lot of additional features out of the box, such as Dynamic Data Masking and Transparent Data Encryption.
- In case you would want more control over your database, you can still run Microsoft SQL Server on an **Azure VM** but that means you need to manage it as well.
- **Azure SQL Warehouse** provides a high-scalable data warehouse in Azure to run your OLAP workloads on.
- **Azure Storage** offers a variety of storage capabilities ranging from a NoSQL table offering to BLOB-based storage.
- **Azure Event Hubs** gives you a hyper-scale immutable stream that can store your data to make it available for near-real-time analytics. Next to that Azure IoT Hub provides the same capabilities but with IoT-specific needs built on top of it. However, it can't be used as an intermediary storage to store your analytics results.
- **Azure Data Lake Store** enables you to build your own data lake and is optimized for analytics scenarios without any size limitations.
- **Azure Document DB** provides you the capability of storing JSON documents that are automatically indexed. It also enables you to replicate your documents to different regions and has protocol support for Mongo DB.

Prefer to use open-source technology? No problem! Microsoft also allows you to set up an **Azure HDInsight** cluster that is running your favorite open-source technology.

6. LEARNING FROM YOUR DATA

While in some integration scenarios the ultimate goal is to provide communication between other systems so that they can perform business logic and analytics, the integration layer is just an orchestrator that handles the communication itself. The logic is done by the other systems and/or their team.

For IoT a few new analytics approaches were introduced. Because there are already a lot of books that discuss them in-depth we will briefly introduce the concepts.

6.1. OVERVIEW OF NEW ANALYTICS MECHANISMS

Stream Processing, also called near-real-time Analytics, is used to process hot data coming in via the Stream Stores containing either raw device telemetry or aggregated device telemetry by a field gateway. Since this processing is based on the raw telemetry it needs to be able to deal with a significant load for thousands/millions of devices.

Some use cases are to aggregate data sets and route them further down the pipeline or make fast business decisions i.e. a device that exceeds a specific threshold should trigger a business process.
For deeper learning **Batch Processing** is typically used where it gets a big data set to analyze and gain insights on new aspects. By using this kind of technology very large data sets get distributed across many nodes by which the processing is separated across N nodes.

Machine Learning enables you to train a model that will predict future outcomes by using algorithms on top of your historical data sets; the larger the data set, the better.

Once the model is trained you can ask what the predicted outcome will be for a specific case and take actions based on that. Sometimes this can even be done by exposing an API. This can be interesting to provide predictive services to customers but it is not something that is fully operational from day one, unless you already have a big data set available to work with.

6.2. MAKE YOUR DATA MORE INTELLIGENT WITH REFERENCE DATA

Not all intelligent decisions are made on the telemetry data from devices, they are made in conjunction with reference data. This can be from a public data source, a licensed data source or an existing enterprise data source from inside the company.

By using this reference data, data sets can improve their value by being mapping to weather information, building information, predictions from a ML model, exchange rates, etc. The challenge here is to incorporate this data into your own analytics pipeline and make it available in a processable format that is discoverable for everybody.
As mentioned before, using a data catalog can help improve the discoverability of reference data sets by providing a central hub that describes what each individual data set is and what it is not.

6.3. ANALYZING DATA IN MICROSOFT AZURE

Microsoft offers a variety of services in Azure that can be used to gain intelligence from your data.
The Cortana Intelligence Suite combines all the products that can be used for making your applications more intelligent and gaining insights in it.

Here are a few interesting products that are valuable for Internet of Things scenarios:

- **Stream Processing** can be done by using **Azure Stream Analytics**. This enables you to run near-real-time processing by using a SQL-like language to analyze data streams flowing in.
- **Batch Processing** can be performed by writing **U-SQL queries** (a mix of SQL and C#) allowing you to build scripts that fit your needs with extensibility in case you need it. By using **Azure Data Lake Analytics**, you can just submit your script and it will run on your data source so you don't need to worry about setting up things and managing clusters.
- Azure also offers **Azure Machine Learning** that can be used to build, train and consult machine learning models that fit your needs. Once your model is ready you can even expose it as a web service to integrate with other applications.

That said, Azure HDInsight also allows you to set up and use clusters with open-source technologies to analyze your data with.

7. SECURITY FROM THE GROUND UP

It goes without saying that security is one of the most important topics when designing an IoT ecosystem. While this topic is worth a white paper in its own right, we will highlight certain aspects that are crucial and how they differentiate from a traditional integration scenario.

Security is not something you simply add on top of the solution when it is done, it is something that needs to be designed for and be built from the ground up to protect your system, the company and the consumers.
As discussed earlier, instead of having an integration landscape we now have an IoT ecosystem which brings a new set of security challenges such as protecting our devices from malicious people to communication from device-to-cloud and cloud-to-device.

7.1. SECURING AN ECOSYSTEM INSTEAD OF A LANDSCAPE

With traditional integration scenarios, there is one central hub that connects to and integrates a variety of systems, these can be either from external partners or an internal system.

That means that we only have to secure our central hub since it is running in an isolated environment, in a local datacenter or in a cloud offering, and that the connected systems are secured by the third parties and internal teams themselves.

You can consider the central hub as one security zone with a trust boundary until we connect to the other systems.

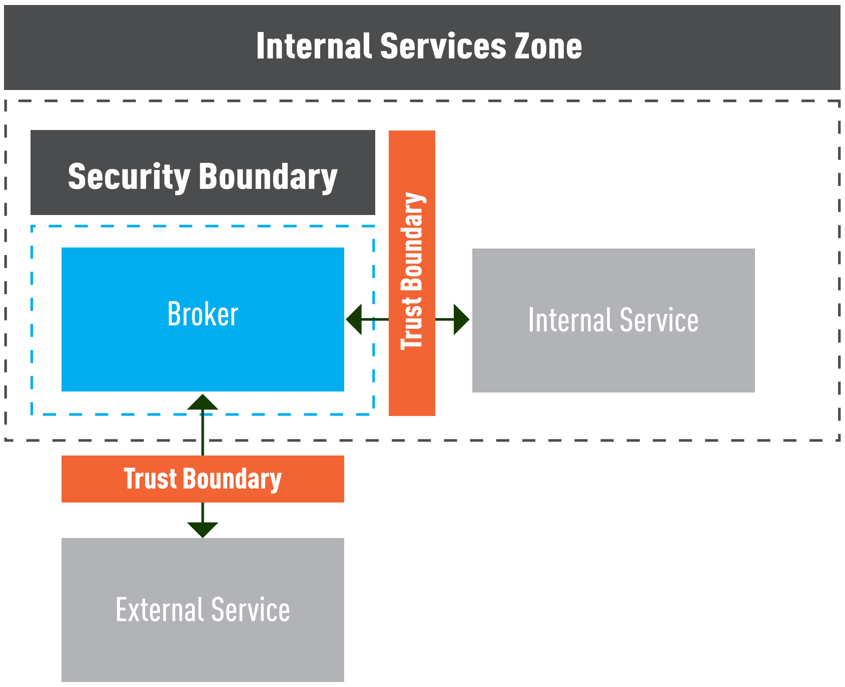


Figure: Integration scenario with zone & trust boundary

With IoT it is a totally different story – instead of only having one centralized engine in the cloud, like we have for our platform, we now also have thousands of individual devices scattered across the city, country or even globe.

It is crucial to monitor and patch the device but also harden it against intruders trying to influence our ecosystem. Communication between each device and the cloud should be well protected as well to avoid eavesdropping and losing valuable business intelligence or to prevent data being manipulated.

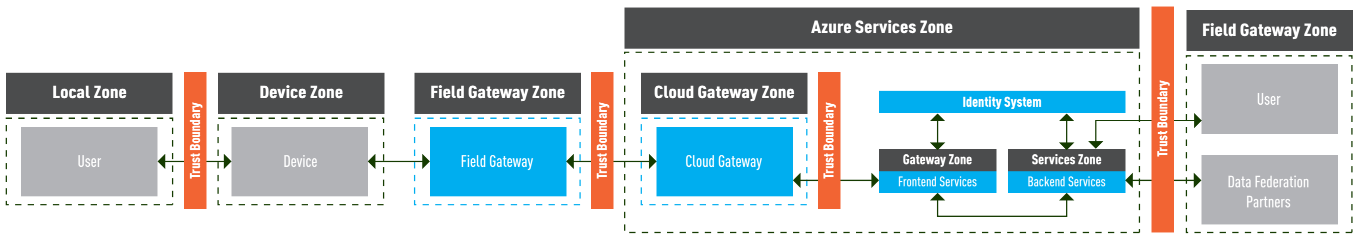


Figure: Internet of Things security architecture

7.2. SECURELY CONNECTING DEVICES TO THE ECOSYSTEM

Securely connecting devices to the ecosystem is a very challenging task. This contrasts with integration, where everything in the environment can be considered a trustworthy source, such as a fileshare. In IoT every component should be treated as a hostile source.

As discussed in chapter "4. Building an ecosystem instead of managing architectures", it is crucial to set up a device management flow where new devices can be provisioned, activated and de-activated. The latter is important in scenarios where it is no longer in service or when it has been compromised by an intruder.

Communication between every device and the cloud should follow the **Service Assisted Communication**-principle defined by Clemens Vasters. This mandates that a device should communicate in an outbound-fashion only and should block all unnecessary ports. The device itself should initiate the communication instead of other parties.

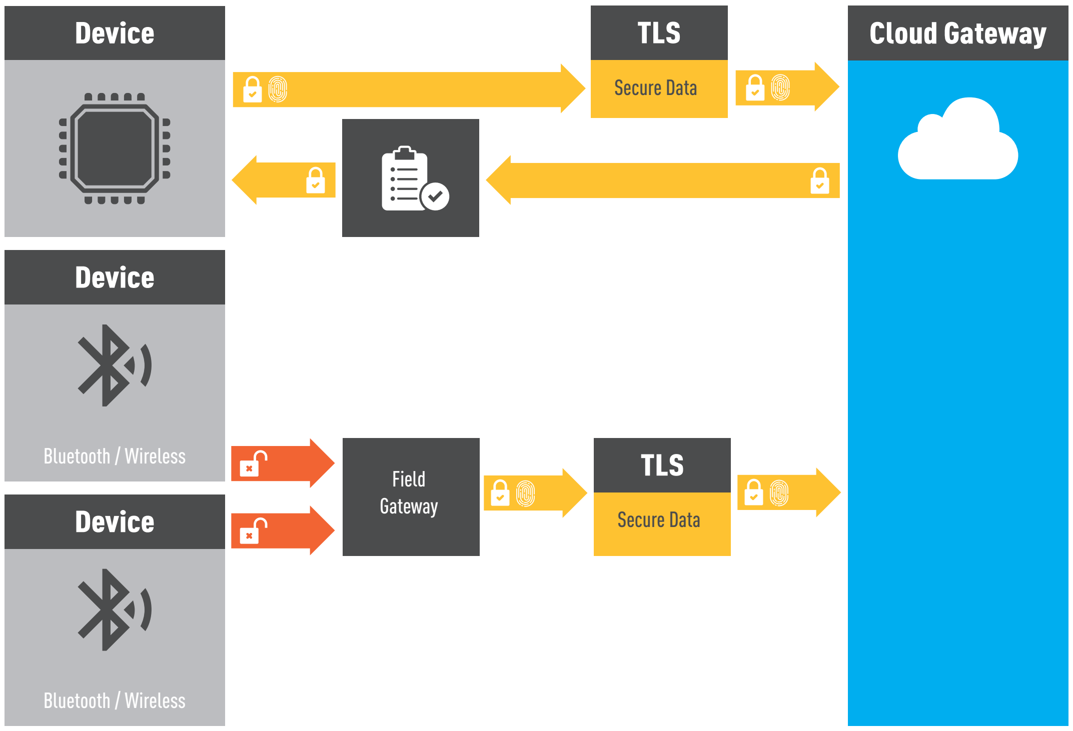


Figure: Devices with unsecure communication use a field gateway
Secure devices communicate directly with the cloud gateway

Having devices open in the field is a dangerous thing. You could say they are waiting to be abused. Websites like **shodan.io** and **censys.io** highlight how vulnerable and discoverable devices are.

Devices should also only communicate using a **secure protocol** such as HTTPS, if they aren't capable in doing so all communication should go through a local field gateway that acts as a proxy between the device itself and the ecosystem.

By using secure protocols, we can avoid attackers to eavesdrop and protect the information that is being exchanged. Taking things a step further, payloads can even be **signed** to guarantee that the sender is a trusted device instead of an intruder with leaked credentials.

Information that is being exchanged can also be **encrypted on the edge nodes** but keep in mind that this can have an impact on your analytics pipeline.

7.3. PROTECTING YOUR DEVICES FROM INTRUDERS

Integration scenarios are typically hosted in either a cloud infrastructure by a cloud provider that secures everything or self-hosted in a rented rack inside a datacenter or simply on-premises. Either way, malicious people don't have direct access unless they have the security clearance or break in.

IoT on the other hand comes with an additional challenge – most of the devices are deployed in the field and are potentially visible or accessible for everybody. Think of devices in public buildings, cars, etc. This brings the risk of malicious people who can tamper with the devices. Therefore, the devices need to be physically secured.

Everything that is stored on the device on **connected storage should be encrypted** so that when it is compromised there is no information disclosure and the storage becomes useless.

Sensitive information such as secrets and keys should either be burned into the silicone or stored in a **Trusted Platform Module (TPM).**

The device should not have any physical connections that are open or connected peripherals that are easy to remove. This is to avoid that malicious people insert their own hardware such as external storage to run a specific application, take memory dumps or install viruses. Physically wrapping the device in a shield can help, if this is applicable to your scenario.

These are only a few of the considerations that should be taken into account, and there are more, but this is out of scope for this white paper since we want to provide a high-level overview.

Want to read more about security in IoT? Chapter "9. Further reading" includes links to interesting documentation by Microsoft on this.

8. CONCLUSION

Internet of Things is quite different from the traditional integration scenarios we are used to. This ranges from how data is secured and used to the scale of the ecosystem. Nevertheless, both scenarios also have a lot in common since they both connect a variety of systems and use a central platform to handle them accordingly.

One of the differences is the variety of data stores and types of data. Hot data is stored and processed different than cold data. Next to that, one of the biggest focuses on IoT is learning from the data to gain more insights in how your business is operating or the performance of your products in the field.

Data has become the new currency and IoT is a good example of it. Using techniques like machine learning can help you optimize and grow your business based on that data.

One of the biggest challenges within IoT is security, but also one of the cornerstones for a successful IoT ecosystem. Think about security from the start and build from there. Think from an intruder perspective and realize where your vulnerabilities are, use threat modelling, etc. Don't just build your own IoT ecosystem, build your own secure IoT ecosystem.

Microsoft Azure leverages great building blocks to build your IoT ecosystem on top of, with services like Azure IoT Hub as a cloud gateway, Azure Data Lake Store as your data lake and many more. By using these building blocks, there is no need to worry about securing and managing these components as this is done by Microsoft.

This is only the beginning of a new era, what are you waiting for? Connect, measure, learn and improve!

9. FURTHER READING

9.1. BOOKS

Here are some recommended books for further reading:

- **Mastering Azure Analytics** - Architecting in the Cloud with Azure Data Lake, HDInsight, and Spark,Zoiner Tejada, O'Reilly Media
- **I Heart Logs** - Event Data, Stream Processing, and Data Integration, Jay Kreps, O'Reilly Media
- **Field Guide to Hadoop** - An Introduction to Hadoop, Its Ecosystem, and Aligned Technologies, Kevin Sitto & Marshall Presser, O'Reilly Media

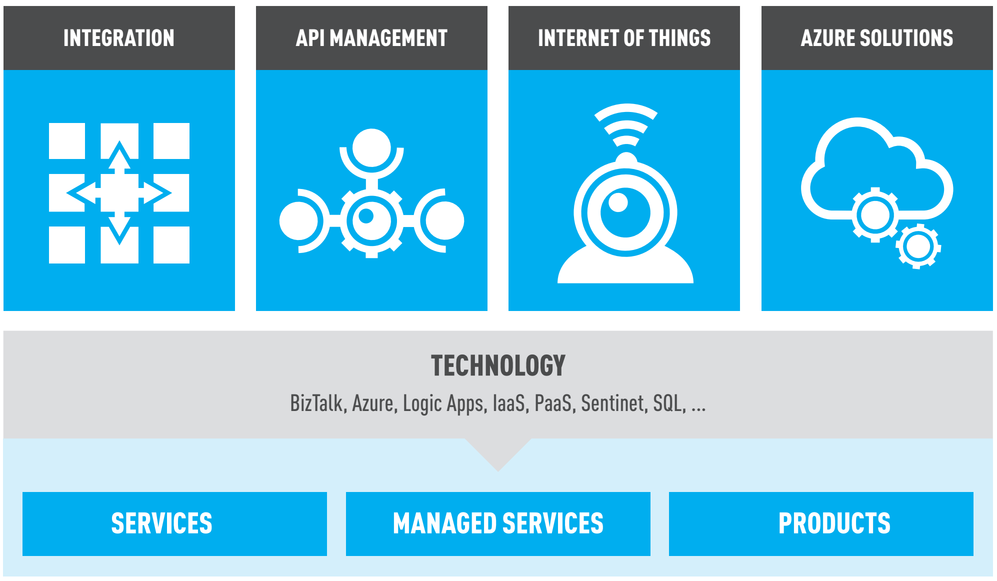
9.2. ARTICLES & REFERENCE MATERIAL

Here are some recommended articles & reference material for further reading:

- **"Data Lake"** by Martin Fowler:
  https://martinfowler.com/bliki/DataLake.html
- **"Service Assisted Communication"** for Connected Devices by Clemens Vasters:
  https://blogs.msdn.microsoft.com/clemensv/2014/02/09/service-assisted-communication-for-connected-devices/
- **Internet of Things (IoT) Security Architecture:**
  https://azure.microsoft.com/en-us/documentation/articles/iot-security-architecture
- **Securing your Internet of Things from the ground up:**
  https://azure.microsoft.com/en-us/documentation/articles/securing-iot-ground-up/
- **Internet of Things (IoT) Security Best Practices:**
  https://azure.microsoft.com/en-us/documentation/articles/iot-security-best-practices/
- **Azure IoT Reference Architecture** (30th of March 2016)
  Announcement & documentation:
  https://azure.microsoft.com/en-us/updates/microsoft-azure-iot-reference-architecture-available
- **MyDriving Reference Implementation** (1st of April 2016)
  Campaign website: https://azure.microsoft.com/en-us/campaigns/mydriving
  Reference Documentation: http://aka.ms/mydrivingdocs

codit

connecting (is) everything



codit.eu

linkedin.com/company/codit

twitter.com/CoditCompany

facebook.com/CoditCompany