happiest minds
The Mindful IT Company
Born Digital . Born Agile

IoT LENS

A WELL-ARCHITECTED FRAMEWORK FOR IOT WORKLOADS

Introduction

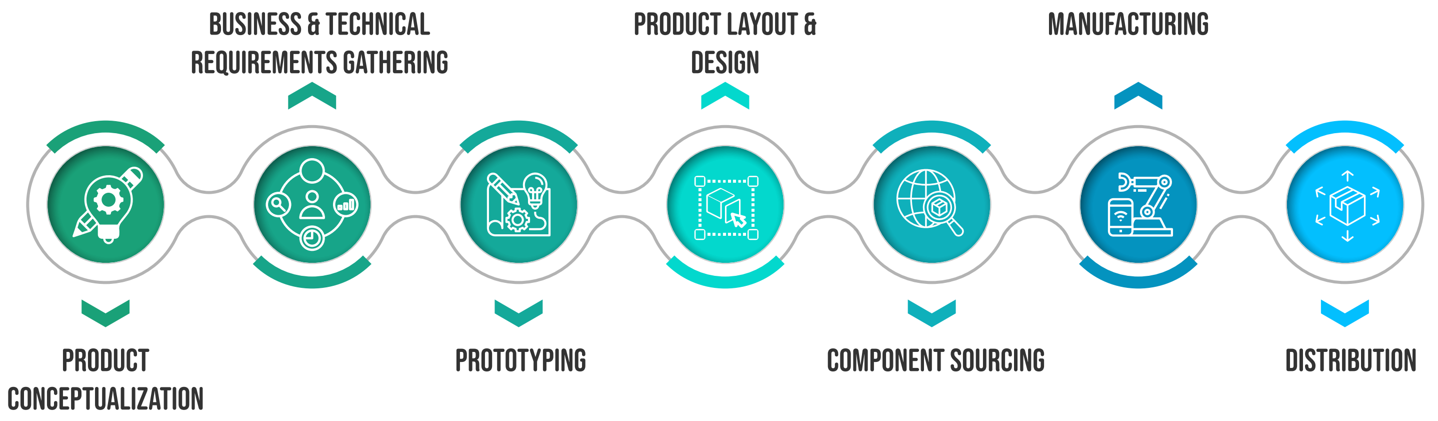
Amazon Web Services (AWS) has defined a Well-Architected Framework which focuses on 6 pillars - operational excellence, security, reliability, performance efficiency, cost optimization, and sustainability. For IoT (Internet of Things) workloads, to adhere to this quality, AWS provides different services. Following these principles ensures that we design robust architecture for IoT or IIoT (Industrial Internet of Things) applications.

Logical Layers of IoT workload

When building an IoT workload there are seven distinct logical layers that we need to consider. We will understand these layers through this article.

Design and Manufacturing Layer

The Design and Manufacturing layer includes the following points:



Decisions made in each layer affect the next layer in IoT workload. For example, the decision on whether each device is loaded with X.509 certificate during manufacturing or whether it is loaded with a common security credential or certificate, will have an impact on how the device will be provisioned in the IoT ecosystem. It will also influence how secure the devices will be.

Edge Layer

Edge layer consists of

- Physical hardware of devices
- Embedded OS on the device
- Device firmware

IIoT Edge layer consists of

- Plant-local Operational Technology
- Plant-local Information Technology
- Remote IT resources

AWS offers the following software and services for the edge layer: AWS IoT device SDKs, FreeRTOS, AWS IoT Greengrass, AWS IoT Sitewise Edge, AWS IoT Fleetwise Edge, AWS IoT RoboRunner Fleet Management System Gateway (FMSG) and AWS IoT ExpressLink.

FREERTOS

AWS IOT GREENGRASS

AWS IOT SITEWISE

AWS IOT EXPRESSLINK

AWS IOT FLEETWISE

AWS IOT ROBORUNNER

These services run at the edge or the device as OS/agents and connect with their respective cloud services.

Fleet Provisioning Layer

This layer provisions or onboards the devices to the cloud. AWS uses X.509 certificates issued by Certificate Authority (CA) to securely provision a device.

IOT CERTIFICATE

The following things are part of the provisioning layer.

- X.509 certificates, AWS IoT Device Registry, AWS Private Certificate Authority (CA), AWS IoT Just-In-Time Registration (JITR), Provisioning devices by claim and Provisioning devices by trusted user.

For example, AWS Private CA helps to automate the lifecycle of the private certificates of the IoT devices.

Fleet Provisioning Layer

Connectivity and message routing between the devices and the cloud are managed in this layer.
The following services are part of it:

- **AWS IoT Core:** provides a managed message broker that supports MQTT protocol.
- **AWS IoT Device Shadow Service:** is a data store to represent the current state of the device.
- **AWS IoT Core for LoRaWAN:** fully managed LoRaWAN Network Server (LNS) that allows you to connect wireless devices that uses LoRaWAN protocol to the AWS Cloud. Example use cases would be irrigation management, logistics and transportation.
- **Amazon API Gateway:** Helps to create API interfaces for systems such as dashboards for technicians, mobile apps, etc.

Ingestion Layer

This layer is responsible for collecting disparate data from various devices, decouple the flow of the data and transmit it to the IoT application in a secure and reliable manner.

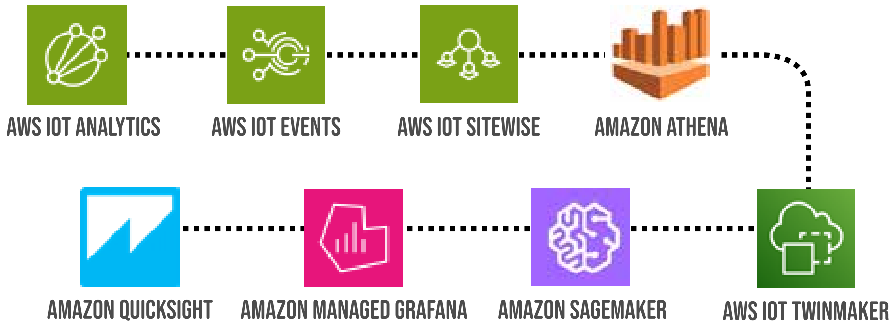
- **IoT Rules Engine:** routes the data to the AWS services according to the rule created. AWS IoT rules are analyzed and actions are triggered based on the topic a message is received on.
- **Basic Ingest:** to securely send device data to the AWS services supported by AWS IoT rule actions. This optimizes data flow and reduces cost by removing publish/subscribe message broker from the ingestion path.
- **AWS IoT Greengrass:** Since, it has an edge agent too, data transfer between edge agent and cloud as well as deployment to edge can be done seamlessly. It can send data to different AWS services like S3, Firehose, IoT Sitewise, IoT Analytics, etc.
- **AWS IoT Sitewise:** managed service that helps to collect, organize, and analyze industrial equipment data at scale. It can be used to monitor operations, compute performance metrics, and create applications that analyze industrial equipment data.
- **AWS IoT Fleetwise:** managed service that collects, organizes, and transfers vehicle data to the cloud.

  It helps you gain insights about the fleet of vehicles and use it for diagnostics, alerts, and take real-time actions.
- **AWS IoT RoboRunner:** provides centralized storage for storing data from different robot vendor systems. It can be used for example, to visualize robot location and status on a single map view.
- **Amazon Kinesis:** is a managed service for streaming data which helps to get insights from IoT devices, and which can be integrated with IoT Rules Engine. It allows seamless integration of devices to applications that support non-MQTT protocols. It also helps to decouple the communication layer from application layer.
- **Amazon Simple Queue Service (SQS):** provides an event-driven, scalable ingestion queue when the IoT application requires a queue which does not require a message order.

Analytics Layer

To gain deep insights from the data sent by the IoT devices to the cloud, this layer includes two types of services:

- Storage Services: to save structured or unstructured data to S3.
- Analytics and machine learning services:



These analytics and ML services give users the ability to run analytics on large datasets, detect and respond to IoT events, collect data from industry equipment, run queries on large datasets, build, train, and deploy ML models, build operational digital twins, create business intelligence dashboards, etc.

Application Layer

The application layer in AWS IoT Lens is user-facing applications. It includes management applications that can operate, inspect, secure, and manage IoT operations. The services that fall under this layer have management applications like AWS IoT Device Defender, AWS IoT Device Management, and Fleet Hub, user applications like Amazon Cognito and database services like Amazon DynamoDB, Amazon Aurora, and Amazon Timestream, and last of all, compute services like AWS Lambda. For example, AWS IoT Defender does audits on device fleets, and when it detects abnormal behavior, and it alerts you about the security issue.

General Design Principles

The Well-Architected Framework expects the following design principles to be followed for a good architecture:

- Decouple ingestion from processing.
- Design for offline behavior.
- Design for lean data at the edge and enrich in the cloud.
- Handle personalization.
- Ensure that devices regularly send status checks.
- Use gateways for edge computing, network segmentation, security compliance and bridging administrative domain.
- Build security into your IoT solution and apply security at all layers.

Pillars of a Well-Architected Framework

As we have understood the layers of an IoT workload and the design principles to be followed, let us see the 6 pillars of the Well-Architected Framework.

- Decouple ingestion from processing.
- Design for offline behavior.
- Design for lean data at the edge and enrich in the cloud.
- Handle personalization.
- Ensure that devices regularly send status checks.
- Use gateways for edge computing, network segmentation, security compliance and bridging administrative domain.
- Build security into your IoT solution and apply security at all layers.

Operational Excellence Pillar

The operational excellence pillar of AWS IoT Lens includes procedures and practices for managing production workloads. It is built on four key pillars: prepare, operate, evolve, and achieve.

Design principles to be followed here are:

- Plan for device provisioning.
- Implement device bootstrapping.
- Document device communication patterns.
- Implement over-the-air (OTA) updates.
- Implement function testing on physical assets.
- Design and build for operations at scale.

Security Pillar

This pillar focuses on protecting systems, data, and assets while also delivering business value.
Design principles to be followed:

- Manage device security lifecycle holistically.
- Ensure least privilege permissions.
- Secure device credentials at rest.
- Implement device identity lifecycle management.
- Take a holistic view of data security.
- Preserve safety and reliability in critical IoT/IIoT environments.
- Implement zero trust principles.
- Establish secure connection with AWS via Site-to-Site VPN or Direct Connect from the industrial edge.
- Use VPC Endpoints whenever possible.
- Use HTTP over TLS proxy and a Firewall for services connecting to AWS online.
- Use secure protocols whenever possible and when using insecure protocols, convert insecure protocols into standardized and secure protocols as close to the source as possible.
- Use network segmentation and tighten trust boundaries.
- Securely manage and access edge computing resources.

Reliability Pillar

The reliability pillar focuses on the ability to prevent and quickly recover from failures to meet business and customer demand.

Three design principles for reliability are:

- Simulate device behavior at production scale.
- Buffer message delivery from the IoT rules engine with streams or queues.
- Design for failure and resiliency.

Performance Efficiency Pillar

The performance efficiency pillar includes the ability to use computing resources efficiently to meet system requirements and maintain that efficiency as demand changes and technologies evolve. The four best practices in this pillar are selection, review, monitoring, and tradeoffs.

Design principles under this pillar are:

- Use managed services.
- Decouple ingestion and processing.
- Use event-driven architecture.

Cost Optimization Pillar

This is a process that involves continually improving and refining a system throughout its lifecycle.

The goal is to help organizations design and implement systems that:`

- Closely align cost with demand
- Use the right AWS resources to achieve business goals cost effectively.
- Enable granular and deep cost attribution and analysis.

Three design principles under this pillar are:

- Manage manufacturing cost tradeoffs.
- Avoid unnecessary data access, storage, and transmission.
- Process data at the edge whenever possible.

Sustainability Pillar

This pillar is to design your IoT solutions to improve sustainability by lowering their carbon footprint while simultaneously reducing operational costs.

Some of the design principles under this pillar are:

- Right-size your hardware.
- Choose a processor to minimize the energy used by your workload.
- Choose storage that supports device longevity.
- Choose a power source with high efficiency.
- Choose a processor with advanced power management features.
- Dimension and manage batteries to maximize battery life.
- Choose a power efficient programming language.
- Reduce the amount of data transmitted.

Conclusion

Using AWS Well-Architected Framework to design and implement your IoT workloads helps to produce stable and efficient systems and helps you to focus on your functional requirements. The architectural best practices which can be seen across the six pillars helps to design reliable, secure, efficient, cost-effective, and sustainable IoT applications. A series of questions or checklists provided by the framework can be used to evaluate a current or prospective architecture and, if necessary, take corrective action.

Author

Soumya R Prabhu

Sr. Technical Lead, IoT Architect

Soumya is a Senior Technical Lead with 17 years of experience working in different domains like Media and Entertainment, Manufacturing, Automobile, Pharmaceutical, and Energy. Her major expertise is in the IoT space where she has worked for 11 years and created a lot of innovative work across different domains. She also loves to mentor and train people in this space and help them build their careers. In her free time, she likes to spend time with her kids, watch cartoons along with them, and do a little bit of herbal gardening.