6

IOT SERVICE AS A PLATFORM

Unit Structure

6.0 OBJECTIVE

- IoT requires expertise in enabling smart sensors to observe, learn, and make decisions to produce limitless market opportunities.
- IoT development services in India leverage widespread interconnected devices to enhance products that retain connectivity, convert data, and supervise them constantly.

6.1 INTRODUCTION

HTTP is a stateless request & response protocol where clients request information from a server and the server responds to these requests accordingly. A request is made up of a method, a resource, some headers, and some optional content. A response is made up of a three-digit status code, some headers and some optional content.

6.2 IOT SECURITY: HTTP, UPNP, COAP, MQTT, XMPP.

6.2.1. Hypertext Transfer Protocol (HTTP): It is used in machine-to-machine (M2M) communication, automation, and Internet of Things, among other things.

6.2.2. Introduction

HTTP is a stateless request & response protocol where clients request information from a server and the server responds to these requests accordingly. A request is made up of a method, a resource, some headers,

and some optional content. A response is made up of a three-digit status code, some headers, and some optional content. A Uniform Resource Locator (URL) identifies each resource, originally thought to be a collection of Hypertext documents or HTML documents. Clients simply use the GET method to request a resource from the corresponding server. In the structure of the URL presented next, the path and the server identify the resource by the authority portions of the URL. The PUT and DELETE methods allow clients to upload and remove content from the server, while the POST method allows them to send data to a resource on the server.

HTTP is a cornerstone of service-oriented architecture (SOA), where methods for publishing services through HTTP are called web services. One important manner of publishing web services is called Simple Object Access Protocol (SOAP), where web methods, their arguments, return values, bindings, and so on, are encoded in a specific XML format. It is then documented using the Web Services Description Language (WSDL).

6.2.3. Adding HTTP support to the sensor

The following are the three strategies used when publishing the data using HTTP:

- In the first strategy the sensor is a client who publishes information to a server on the Internet. The server acts as a broker and informs the interested parties about sensor values. This pattern is called publish/subscribe.
- Second Strategy is to allow all entities in the network be both clients and servers, depending on what they need to do. The UPnP Protocol. This reduces latency in communication but requires all participants to be on the same side of any firewalls.

Third Strategy is to let the sensor become an HTTP server, and who is interested in knowing the status of the sensor become the clients. It also allows easy access to the sensor from the parties behind firewalls if the sensor is publicly available on the Internet.

6.2.4. Setting up an HTTP server on the sensor

To begin with,

1. add the namespace in the application.

   using System.Xml;

   using System.Text;

   using System.IO;

   using System.Drawing;

2. Then, add references to the following Clayster namespaces, which will help to work with HTTP and along with different content types mentioned.

```
using Clayster.Library.Internet;
using Clayster.Library.Internet.HTTP;
using Clayster.Library.Internet.HTML;
using Clayster.Library.Internet.MIME;
using Clayster.Library.Internet.JSON;
using Clayster.Library.Internet.Semantic.Turtle;
using Clayster.Library.Internet.Semantic.Rdf;
using Clayster.Library.IoT;
using Clayster.Library.IoT.SensorData;
using Clayster.Library.Math;
```

The Internet library helps us with communication and encoding, the IoT library deals with an interoperability, and the Math library deals with graphs.

3. Next step is application initialization, which is done using the following code:

```
HttpSocketClient.RegisterHttpProxyUse (false, false);
```

To instantiate an HTTP server, add the following code before application initialization ends and the main loop begins:

```
HttpServer HttpServer = new HttpServer (80, 10, true, true, 1);
Log.Information ("HTTP Server receiving requests on port" + HttpServer.Port.ToString ());
```

The HTTP server can process both synchronous and asynchronous web resources:

A synchronous web resource responds within the HTTP handler we register for each resource. These are executed within the context of a working thread.

- An asynchronous web resource handles processing outside the context of the actual request and is responsible for responding by itself. This is not executed within the context of a working thread.

4. Register web resources on the server: register the path of each resource and connect that path with an HTTP handler method, which will process each corresponding request.

```
HttpServer.Register ("/", HttpGetRoot, false);
HttpServer.Register ("/html", HttpGetHtml, false);
HttpServer.Register ("/historygraph", HttpGetHistoryGraph, false);
HttpServer.Register ("/xml", HttpGetXml, false);
HttpServer.Register ("/json", HttpGetJson, false);
HttpServer.Register ("/turtle", HttpGetTurtle, false);
HttpServer.Register ("/rdf", HttpGetRdf, false);
```


5. Disposing the server when the application ends

```
HttpServer.Dispose ();
```

Adding a root menu

Add a root menu which is accessible through the path /.

```
private static void HttpGetRoot (HttpServerResponse resp, HttpServerRequest req)
{
 networkLed.High ();
 try
{
 resp.ContentType = "text/html";
 resp.Encoding = System.Text.Encoding.UTF8;
resp.ReturnCode = HttpStatusCode.Successful_OK;
}
finally
{
networkLed.Low ();
}
}
```

Return the actual HTML page with the following code:

```
resp.Write ("<html><head><title>Sensor</title></head>");
resp.Write ("<body><h1>Welcome to Sensor</h1>");
resp.Write ("<p>Below, choose what you want to do.</p><ul>");
resp.Write ("<li>View Data</li><ul>");
resp.Write ("<li><a href='/xml?Momentary=1'>");
resp.Write ("View data as XML using REST</a></li>");
resp.Write ("<li><a href='/json?Momentary=1'>");
resp.Write ("View data as JSON using REST</a></li>");
resp.Write ("<li><a href='/turtle?Momentary=1'>");
resp.Write ("View data as TURTLE using REST</a></li>");
resp.Write ("<li><a href='/rdf?Momentary=1'>");
resp.Write ("View data as RDF using REST</a></li>");
resp.Write ("<li><a href='/html'>");
```

```
resp.Write ("Data in a HTML page with graphs</a></li></ul>");
resp.Write ("</body></html>");
```

Accessing WSDL

The SOAP web service interface is documented in what is called a Web Service Definition Language (WSDL) document. The web services engine automatically generates this document.

Adding HTTP support to the controller

Sensor and an actuator that speaks about HTTP also need to add HTTP to the controller. The controller will act as an HTTP client.

6.3 THE UPNP PROTOCOL

Universal Plug and Play (UPnP) is a protocol or an architecture that uses multiple protocols, helps devices in ad hoc IP networks to discover each other, detects services hosted by each device, and executes actions and reports events. Ad hoc networks are networks with no predefined topology or configuration; here, devices find themselves and adapt themselves to the surrounding environment. UPnP is largely used by consumer electronics in home or office environments. The standard body for UPnP is the UPnP Forum (upnp.org). UPnP is largely based on an HTTP application where both clients and servers are participants.

Discovery of devices in the network is performed using Simple Service Discovery Protocol (SSDP), which is based on HTTP over UDP, and event subscriptions and notifications are based on General Event Notification Architecture (GENA). Both SSDP and GENA introduce new HTTP methods to search, notify and subscribe to and unsubscribe from an event. Actions on services are called using SOAP web service calls.

UPnP defines an object hierarchy for UPnP-compliant devices. Each device consists of a root device. Each root device can publish zero or more services and embedded devices. Each embedded device can iteratively publish more services and embedded devices by itself. Each service in turn publishes a set of actions and state variables. Actions are methods that can be called on the service using SOAP web service method calls. Actions take a set of arguments. Each argument has a name, direction (if it is input or output), and a state variable reference. From this reference, the data type of the argument is deduced. State variables define the current state of a service, and each one has a name, data type, and variable value. Furthermore, state variables can be normal, evented, and/or multicast-evented. When evented state variables change their value, they are propagated to the network through event messages. Normally, evented state variables are sent only to subscribers who use normal HTTP. Multicast-evented state variables are propagated through multicast HTTPMU NOTIFY messages on the SSDP multicast addresses being used, but using a different port number.

Each UPnP-compatible device in the network is described in a Device Description Document (DDD), an XML document hosted by the device

itself. When the device makes its presence known to the network, it always includes a reference to the location of this document. Interested parties then download the document and any referenced material to learn what type of device this is and how to interact with it. The document includes some basic information understandable by machines, but it also includes information for human interfaces. Finally, the DDD includes references to embedded devices, if any, and references to any services published by the device.

Each service published by a device is described in a standalone Service Control Protocol Description (SCPD) document, each one an XML document also hosted by the device. Even though SOAP is used to call methods on each service, UPnP-compliant services are drastically reduced in functionality compared to normal SOAP web services. SOAP and WSDL simply give devices too many options, making interoperability a problem.

6.4 THE COAP PROTOCOL

CoAP reduces the set of methods that can be used; it allows you to have four methods: GET, POST, PUT, and DELETE. In addition, in CoAP, method calls can be made using confirmable and non-confirmable message services. When you receive a confirmable message, the receiver always returns an acknowledgement. The sender can, in turn, resend messages if an acknowledgement is not returned within the given time period. The number of response code has also been reduced to make implementation simpler. CoAP also broke away from the Internet Media Type scheme used in HTTP and other protocols and replaced this with a reduced set of Content-Formats, where a number instead of its corresponding Internet Media Type identifies each format.

CoAP supports multicasting, which is used to detect devices or communicate through firewalls; it also provides a set of useful extensions. One of these extensions provides a block transfer algorithm, which allows you to transfer larger amounts of data. CoAP also supports encryption in the unicast case with Datagram Transport Layer Security (DTLS).

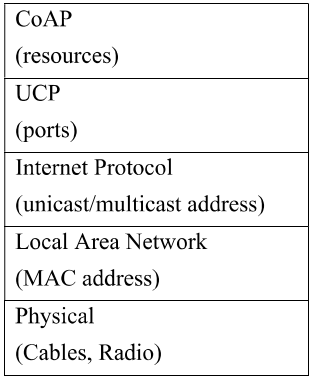


Fig: CoAP protocol stack diagram:

CoAP is relatively new; the availability of development tools for this protocol is not available. There exists an add-on to Firefox, which allows you to view and interact with CoAP resources.

CoAP resources- The CoAP endpoint registers a resource by itself called .well-known/core. Here, it publishes a Link Format document called the Constrained RESTful Environments (CoRE) Link Format document. This document contains a list of resources published by the endpoint and some basic information about these documents. This document corresponds in some sense to WSDL documents for web services, even though the Link Format document is very lightweight. It consists of a sequence of resources and some corresponding attributes for each resource.

6.5 MQTT PROTOCOL

The MQTT protocol is based on the publish/subscribe pattern, as opposed to the request/response and the event subscription patterns. The publish/subscribe pattern has three types of actors:

a. Publisher: The role of the publisher is to connect to the message broker and publish content.

b. Subscriber: They connect to the same message broker and subscribe to content that they are interested in

c. Message broker: This makes sure that the published content is relayed to interested subscribers

Content is identified by topic. When publishing content, the publisher can choose whether the server should retain the content or not. If retained, each subscriber will receive the latest published value directly when subscribing. Furthermore, topics are ordered into a tree structure of topics, much like a file system. The forward slash character (/) is used as a delimiter when describing a topic path. When subscribing to content, a subscriber can subscribe to either a specific topic by providing its path, or an entire branch using the hash wildcard character (#). There is also a single-level wildcard character: the plus character (+).

<table>
<tr><td>MQTT</td><td>MQTT (SSL/TLS)</td></tr>
<tr><td>TCP
(port 1883)</td><td>TCP
(port 8883)</td></tr>
<tr><td colspan="2">Internet Protocol (IP)
(Unicast/multicast IP address)</td></tr>
<tr><td colspan="2">Local Area Network (LAN)</td></tr>
<tr><td colspan="2">Physical (Cables, Radio)</td></tr>
</table>

Fig : MQTT architecture

There are three Quality of Service levels in MQTT available while publishing content. The lowest level is an unacknowledged service. In this, the message is delivered at most once to each subscriber. The next level is an acknowledged service. Here, each recipient acknowledges the receipt of the published information. If no receipt is received, the information can be sent again. This makes sure the information is delivered at least once. The highest level is called the assured service. Here, information is not only acknowledged but sent in two steps. First, it is transmitted and then delivered. Each step is acknowledged. This makes it possible to make sure that the content is delivered exactly once to each subscriber.

6.6 XMPP PROTOCOL

Extensible Messaging and Presence Protocol (XMPP) protocol. The XMPP protocol also uses message brokers to bypass firewall barriers. Apart from the publish/subscribe pattern, it also supports other communication patterns, such as point-to-point request/response and asynchronous messaging, that allow you to have a richer communication

XMPP was originally designed for use in instant messaging applications (or chat). It is an open protocol, flexible and richness of communication patterns.

The XMPP architecture is built on scalability of the Simple Mail Transfer Protocol (SMTP). XMPP uses a network of XMPP servers as message brokers to allow clients behind separate firewalls to communicate with each other. Each server controls its own domain and authenticates users on that domain. Clients can communicate with clients on other domains using federation where the servers create connections between themselves in a secure manner to interchange messages between their domains. They only need to ensure that they maintain the connection with their respective servers, and through the servers, each of them will have the possibility to send messages to any other client in the federated network. XMPP is scalable and allows you to make billions of devices communicate with each other in the same federated network.

XMPP provides each client with an authenticated identity. When clients connect, the servers make sure the clients authenticate themselves by providing their corresponding client credentials, which would consist of a username and password. This authentication is done securely using an extensible architecture based on Simple Authentication and Security Layer (SASL). The connection can also be switched over to Transport Layer Security (TLS) through negotiation between the client and the server. The identity of the client is often called XMPP address or Jabber ID (JID).

The reason for using XMPP servers is to relay communication to assure the clients that only authorized communication will be relayed. This feature is used for small devices with limited decision-making capabilities. The server does so by ensuring that the full JID identifier instead of only the bare JID identifier is used to communicate with the application.

The reason is:

- First, multiple clients might use the same account at the same time. Then provide the resource part of the full JID for the XMPP Server to be able to determine which connection the corresponding message should be forwarded to. Only this connection will receive the message. This enables the actual clients to have direct communication between them.

- Second, only trusted parties (or friends) are given access to the resource part once the thing or application is connected. This means that, in turn, only friends can send messages between each other, as long as the resource parts are sufficiently long and random so they cannot be guessed, and the resource part is kept hidden and not published somewhere else. XMPP communication consists of bidirectional streams of XML fragments.

IoT Service as a Platform: Clayster, Thinger.io, SenseIoT, carriots and Node RED.

1. Clayster

There are many available platforms, they vary in functionality and development. To get the IoT platform go to http://postscapes.com/internet-of-things-platforms and review the registered platforms.

2. Clayster Platform

Download the Clayster platform by downloading from http://www.clayster.com/downloads. All the information about Clayster, including examples and tutorials, is available in a wiki. You can access this wiki at https://wiki.clayster.com/.

3. Libraries

Clayster.AppServer.Infrastructure: This library contains the application engine available in the platform. Apart from managing applications, it also provides report tools, cluster support, management support for operators and administrators; it manages backups, imports, exports, localization and various data sources used in IoT, and it also provides rendering support for different types of GUIs, among other things.

Clayster.Library.Abstract: This library contains a data abstraction layer, and is a crucial tool for the efficient management of objects in the system.

Clayster.Library.Installation: This library defines the concept of packages.

Clayster.Library.Meters: This library replaces the Clayster. Library. IoT library used in previous chapters. It contains an abstraction model for things such as sensors, actuators, controllers, meters, and so on.

Clayster: To facilitate the development of IoT applications, seven Clayster libraries are used for private and commercial applications.

Clayster Library	**Description**
Clayster.Library.Data	It provides the application with a powerful object database. Objects are persisted and can be searched directly in the code using the object's class definition. Data can be stored in the SQLite database provided in Raspberry Pi.
Clayster.Library.EventLog	This provides the application with an extensible event logging architecture that can be used to get an overview of what happens in a network of things.
Clayster.Library.Internet	It consist of classes that implements various internet protocol. This library can be used dynamically for communication via internet.
Clayster.Library.Language	It is used to create localizable applications that are simple to translate and that can work in an international setting.
Clayster.Library.Math	It has powerful mathematical scripting language that will be used in automation, scripting, graph plotting, and others.
Clayster.Library.IoT	It has classes that help the applications to become interoperable by providing data representation and parsing capabilities of data in IoT.
Clayster.Library.RaspberryPi	It has Hardware Abstraction Layer (HAL) used in Raspberry Pi. It provides object-oriented interfaces to interact with devices connected to the general-purpose Input/output (GPIO) pins available.

4. Service modules

The service modules available are:

Clayster.HomeApp.MomentaryValues: This is a simple service that displays momentary values using gauges. We will use this project to display gauges of our sensor values.

Clayster.Metering.Xmpp: This module contains an implementation of XMPP on top of the abstraction model defined in the Clayster.Library.Metersz namespace.

5. Clayster Management Tool (CMT)

It comes with a management tool that helps you to manage the server. This Clayster Management Tool (CMT) can also be downloaded from http://www.clayster.com/downloads. This includes data sources, objects in the object database, current activities, statistics and reports, and data in readable event logs

Browsing data sources

Most of the configurable data in Clayster is ordered into data sources. These can be either tree-structured, flat or singular data sources. Singular data sources contain only one object. Flat data sources contain a list (ordered or unordered) of objects. Tree structured data sources contain a tree structure of objects, where each object in the structure represents a node. The tree-structured data sources are the most common, and they are also often stored as XML files. Objects in such data sources can be edited directly in the corresponding XML file, or indirectly through the CMT, other applications or any of the other available APIs. When you open the CMT for the first time, make sure that you open the Topology data source. It is a tree-structured data source whose nodes represent IoT devices. The tree structure shows how they are connected to the system. The Root represents the server itself.

Interfacing the devices

XMPP is already implemented and supported through the Clayster. Metering. This module models each entity in XMPP as a separate node in the Topology data source. Connections with provisioning servers and thing registries are handled automatically through separate nodes dedicated to this task. Friendships are handled through simple child creation and removal operations. It can be done automatically through requests made by others or recommendations from the provisioning server, or manually by adding friends in the CMT. Provide specialized classes that override base class functionality and add specific features that are needed.

6. Thinger.io

What is Thinger.io

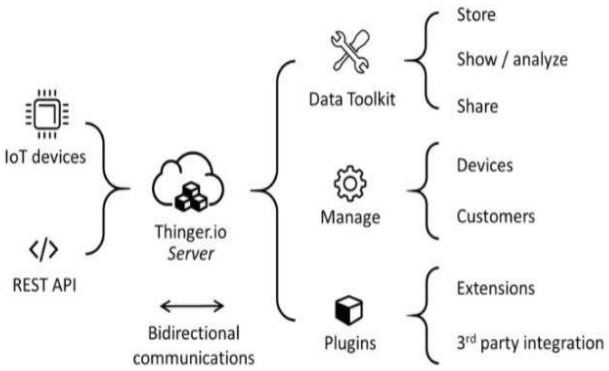
Thinger.io is a cloud IoT Platform that provides every needed tool to prototype, scale and manage connected products in a very simple way.

Features

- **Free IoT platform**: Thinger.io provides a lifetime freemium account with only few limitations to start learning and prototyping when your product becomes ready to scale, you can deploy a Premium Server with full capacities within minutes.
- **Simple but Powerful**: Just a couple code lines to connect a device and start retrieving data or controlling its functionalities with our web-based Console, able to connect and manage thousands of devices in a simple way.
- **Hardware agnostic:** Any device from any manufacturer can be easily integrated with Thinger.io's infrastructure.
- **Extremely scalable & efficient infrastructure:** thanks to our unique communication paradigm, in which the IoT server subscribes device resources to retrieve data only when it is necessary, a single Thinger.io instance is able to manage thousands of IoT devices with low computational load, bandwidth and latencies.
- **Open-Source**: most of the platform modules, libraries and APP source code are available in our Github repository to be downloaded and modified with MIT license.

  Thinger.io platform is formed by two main products a Backend (which is the actual IoT server) and a web-based Frontend that simplifies working with all the features using any computer or smartphone.
- **Connect devices:** Fully compatible with every kind of device, no matter the processor, the network, or the manufacturer. Thinger.io allows to create **bidirectional communications** with Linux, Arduino, Raspberry Pi, or MQTT devices and even with edge technologies like Sigfox or LoRaWAN or other internet API data resources.
- **Store Device Data:** Just a couple clicks to create a Data Bucket a store IoT data in a scalable, efficient, and affordable way, that also allows real-time data aggregation.
- **Display Real-time** or **Stored Data** in multiple widgets such as time series, donut charts, gauges, or even custom-made representations to create awesome dashboards within minutes.
- **Trigger events and data values** using an embedded Node-RED rule engine
- **Extend with custom features** with multiple plugins to integrate IoT projects into your company's software or any other third-party Internet service.

- **Custom the appearance** thanks to our fully rebrand able frontend, that allows introducing your branding colours, logotypes and web domain. Refer: https://docs.thinger.io/



7. SenseIoT

The Sense IoT cloud server is a **logical server** built, hosted, and delivered through a cloud computing platform over the internet. Unlike normal physical servers, cloud servers can be accessed remotely at any time.

The term IoT stands for Internet of Things, and it is the most significant as well as promising technology nowadays. There are a billion devices relate to sensors like wearables, smartphones, etc. Currently, every sensor plays an essential role in the Internet of Things. These sensors are mainly used for detecting or monitoring the quality of air, health status, home security, etc. Similarly, these sensors are used in IoT for monitoring the process of production, so named as IoT sensor.

There are different types of sensors which is used for different applications like to collect the data from the environment. In an IoT ecosystem, there are two main things the internet & the physical devices such as actuators & sensors. The sensor and network connectivity in the IoT mainly located in the bottom layer. The main function of this is to collect the information. This bottom layer in the IoT is a very important part, and it includes connectivity of network to next layer like the gateway & network layer.

The main function of these sensors is to gather information from the surroundings. The connection of these to IoT can be done directly otherwise indirectly once the conversion of signal & processing is done. All the sensors are not similar because different IoT applications need different kinds of sensors.

Types of IoT Sensors

The different types of IoT sensors with its working as follows:

Temperature Sensor

The temperature sensor is used to detect the heat energy which is produced from an object or nearby area. The main role of these sensors in manufacturing is for temperature monitoring of machines. Similarly, in the agriculture field, these sensors are used to monitor the temperature of plants, soil, and water. The applications of temperature sensors mainly include refrigerators, ACs, etc.

Smoke Sensor

Smoke sensors have been using in various applications like homes, industries, etc. These sensors are very convenient as well as easy to use by the arrival of the Internet of Things. Also, by adding a wireless connection to smoke detectors, the additional features can be enabled to increase security & ease.

Motion Sensor

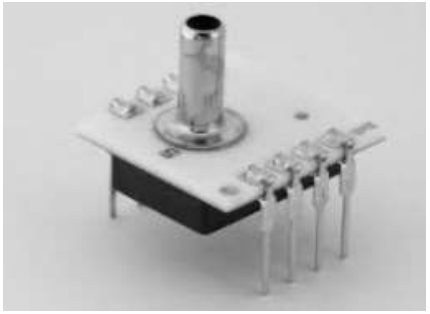
The motion sensor is used in hand dryers, energy management systems, automatic parking systems, automatic door controls, automated toilet flushers, automated sinks, etc.

Humidity Sensors

Humidity sensors are used to monitor the level of humidity in the amount of vapor of water within the air. The units for measurement humidity is RH (relative humidity) & PPM (parts per million).

Pressure Sensor

The pressure sensors are used in IoT for monitoring devices and systems which are determined by force signals. As the range of pressure is outside the threshold stage, then the device gives an alert to the user regarding the issues that must be fixed. The best example of a pressure sensor is BMP180, which can be used in mobile phones, GPS navigation devices, etc. These sensors are also applicable in aircraft and smart vehicles to decide altitude & force correspondingly. In a motor vehicle, TMPS (tire pressure monitoring system) can also be used for giving an alert to the driver while tire pressure is extremely less & it could make unsafe driving situations.



Gas Sensor

Gas sensors are mainly used to detect toxic gases. The most frequently used technologies are photoionization, semiconductor, and electrochemical.

IR Sensors

Infrared sensors are mainly used to measure the heat which is produced by objects. These sensors are used in the various applications of IoT like healthcare for monitoring the flow of blood, BP, etc. These sensors are used in smartphones for controlling, wearable devices for detecting the amount of light, detection of blind spot within vehicles, etc.

Accelerometer Sensor

Accelerometer sensors are utilized in aircrafts vehicles, smartphones. Similarly, these are used in different applications to identify the direction of an object, tilt, tap, shake, positioning, and motion, vibration, or shock. Types of accelerometers are like capacitive, Hall-effect & piezoelectric.

Image Sensor

Image sensors are applicable in medical imaging systems, thermal imaging devices, digital cameras, night-vision equipment, sonars, radars, & biometric systems. These sensors are used in the retail industry for monitoring the visiting count of the customers in the store with the help of network like IoT. The applications of image sensors mainly include offices, corporate buildings for monitoring the employees.

Proximity Sensors

Proximity sensors are used to detect the existence or nonexistence of a near object with no physical contact. These sensors are classified into different types like capacitive, inductive, ultrasonic, magnetic, and photoelectric. These sensors are frequently used for process monitoring, control, and object counters.

Source: IoT Sensor: Different Types, Working and Its Applications (elprocus.com)

7. Carriots

The Carriots IoT Platform is designed to help companies, businesses and institutions to improve their business and efficiency by controlling their IoT devices, and provides safe transfer of information among IoT devices, cloud and by bringing solutions to the problems by using AI and other technologies.

Carriots is a smart Platform as a Service (PaaS), functions in a machine-to-machine way and doesn't necessitate any human interruption after the system is set up. It is also established by Altair Engineering.

What Does Carriots Do?

Carriots has a four-staged working mechanism in helping a business to find solutions to its problems. First stage is the data analysis stage. The investors who want to develop their business by innovation gives the data to the Carriots IoT platform after it is set up to the system. After the data on business is given to the Carriots platform, the Carriots begins to analyse the data and try to detect the problems, obstacles and other latent issues in the business operation process. In the second stage, Carriots tries to find a solution to the problems that it detected in the first stage. It has its own mechanism to eliminate the methods which are not useful to the business. In the third stage, it approves the solution method which should be implemented on the business that decided in the second stage.

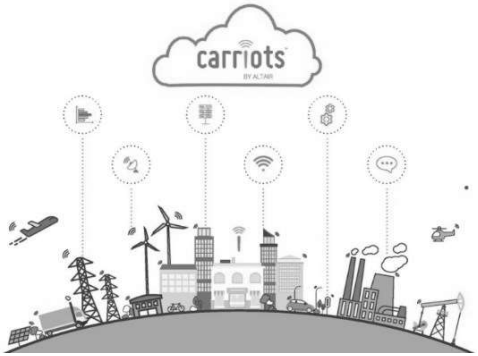
What are the Advantages of Carriots IoT Platform?

Features of Carriots that make it more advantageous:

- Carriots has an open architecture that enables it to work in collaboration with the third-party machines. This enables it to find the

best solution due to its wide data collection because of this feature and flexibility.

- The Carriots Platform can be controlled by a device that is connected to a remote controller. This feature enables to keep a watch what they are doing with your business. Also, Carriots is not bound by the check status and can change configurations.
- Rules: Carriots has rule to apply in complex business scripts.
- Carriots also has new triggers that push the data into the platforms and enables the usage of that data.
- It can integrate with other AI and IT systems
- System also has custom alarms when it faces a problematic situation during the innovation.



Source: Carriots IoT Platform | IoT5.net

9.Node RED.

Node-Red is based on graphical interface and has three components:

Flow -A project is called a flow and consist of data and functions linked together.

Message-It carries data from one node to another.

Nodes- These are the functions that generate, transfer or use messages.

The Node-RED GUI consists of three parts, from left to right:

- The left pane lists all the nodes, grouped by categories.
- The centre pane corresponds to the working area, where the flow is going to be designed.

- The right pane provides useful tools as documentation, a console for debugging, and the organisation for the dashboard.

Node-RED offers native support for other services. For example, there is a node for sending e-mails. Node-RED relies on MQTT, which requires a TCP/IP stack.

6.7 SUMMARY

- Innovative companies utilize information potential in Internet of Things as a service (IoTaaS) technology that senses information to create better products for that satisfies customers.
- The complex handling of IoT's fusion technologies leaves enterprises without in-house expertise or software to make IoT a profitable and worthwhile investment.

6.8 REFERENCES

- Learning Internet of Things by Peter Waher, Packt Publishing. (All notes are taken from this prescribed reference book).
- Notes also taken from the link below:
- https://github.com/Clayster/ Learning-IoT-CoAP.
- https://docs.thinger.io/
- IoT Sensor: Different Types, Working and Its Applications (elprocus.com)

6.9 UNIT END EXERCISE QUESTIONS

1. What is Clayster in IoT? Explain different libraries in it.
2. Explain the following IoT security: HTTP, UPnp, CoAP, MQTT, XMPP
3. What is meant by IoT service as a platform?
4. Explain different types of IoT Service as a Platform: Clayster, Thinger.io, SenseIoT, carriots and Node RED.
5. What is Thinger.io? What are the features of it.
6. Describe SenseIoT?
7. What are sensors? Explain different types of sensors used in IoT.
8. What are Carriots? What are its advantages

❄❄❄❄❄❄❄

7

IOT SECURITY AND INTEROPERABILITY

Unit Structure

7.1 OBJECTIVE

Hardware, Software, and other devices needs security to work efficiently. Without security in IoT devices & it's hardware will result in hacking & hackers will gain access to the controlling of the IoT devices and hence the IOT application will fail.

This chapter talks about the different types of security mechanisms, protocols, methods & techniques that should be deployed when the IoT devices & its application are ready to function.

Also discussed various types of virus, malwares & Trojan horse which has harmed the devices & made the resources less to perform.

7.2 INTRODUCTION

There are a lot of different technologies that can be used for Internet of Things (IoT), but security and interoperability issues come to any extent. We will discuss the topics, issues and that need to be addressed during the design of the overall architecture to avoid many of the unnecessary problems that might otherwise arise and minimize the risk.

7.3 IOT SECURITY AND INTEROPERABILITY

Risks with IoT, Modes of attacking a system and some counter measures, The importance of interoperability in IoT.

7.4 RISKS

There are many solutions and products under IoT that lack basic security architectures. It is very easy for a knowledgeable person to take control of devices for malicious purposes. Not only devices at home are at risk, but cars, trains, airports, stores, ships, logistics applications, building automation, utility metering applications, industrial automation applications, health services, and so on, are also at risk because of the lack of security measures in their underlying architecture.

5.MODES OF ATTACKS

a. Denial of Service

A Denial of Service (DoS) or Distributed Denial of Service (DDoS) attack is normally used to make a service on the Internet crash or become unresponsive, and in some cases, behave in a way that it can be exploited. The attack consists in making repetitive requests to a server until its resources gets exhausted. In a distributed version, the requests are made by many clients at the same time, which obviously increases the load on the target. It is often used for blackmailing or political purposes.

b. Guess the Username and Password

Getting it in the system is to try to impersonate by guessing the username and password of the authenticated clients. Guessing the client credentials is risky and it makes the attack less effective, in the communication. Always have the habit of changing the credentials frequently. The preset and fixed password helps the attacker to crack it easily.

c. Access to stored credentials

People reusing the credentials in different systems. There are various ways to avoid this risk. Credentials of the clients are not to be reused in different devices or across different services and applications. Another is to randomize the credentials, lessening the desire to reuse memorized credentials. Never store actual credentials centrally, even encrypted if possible, and instead store hashed values of these credentials. This is often possible since authentication methods use hash values of credentials in their computations. Even though some hashing functions are vulnerable in such a way that a new string can be found that generates the same hash value.

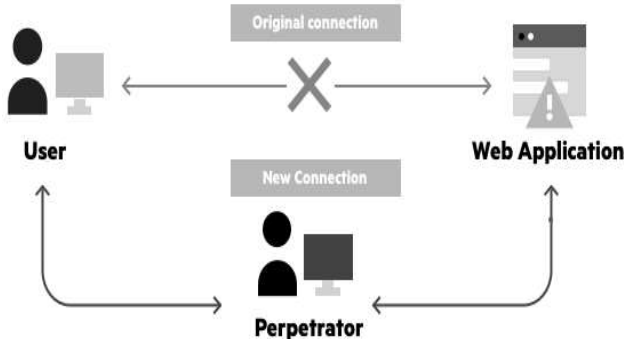
d. Man in the Middle attack (MITM)

Man-in-the-middle (MITM) attacks are a valid and extremely successful threat vector.

A man in the middle (MITM) attack is a general term for when a perpetrator positions himself in a conversation between a user and an

application—either to eavesdrop or to impersonate one of the parties, making it appear as if a normal exchange of information is underway.

The goal of an attack is to steal personal information, such as login credentials, account details and credit card numbers. Targets are typically the users of financial applications, businesses, e-commerce sites and other websites where logging is needed.



https://www.imperva.com/learn/application-security/man-in-the-middle-attack-mitm/

An MITM attack can take a few different forms. ARP poisoning is the most common, but DHCP, DNS, and ICMP poisoning are also effective, as well as the use of a malicious wireless access point (AP). Fake APs have become a common threat vector, exploiting the way clients automatically connect to known SSIDs. This enables an attacker to connect and intercept the victim's network traffic without the victim seeing any indication they are under attack. To hasten a connection, attacks against the legitimate AP can be made to help the malicious AP become the last AP standing.

It can lead to launch the different attacks:

IP spoofing involves an attacker disguising himself as an application by altering packet headers in an IP address. As a result, users attempting to access a URL connected to the application are sent to the attacker's website.

ARP spoofing is the process of linking an attacker's MAC address with the IP address of a legitimate user on a local area network using fake ARP messages. As a result, data sent by the user to the host IP address is instead transmitted to the attacker.

DNS spoofing, also known as DNS cache poisoning, involves infiltrating a DNS server and altering a website's address record. As a result, users attempting to access the site are sent by the altered DNS record to the attacker's site.

MITM Attack Prevention

- Avoiding Wi-Fi connections that aren't password protected.
- Paying attention to browser notifications reporting a website as being unsecured.
- Immediately logging out of a secure application when it's not in use.
- Not using public networks (e.g., coffee shops, hotels) when conducting sensitive transactions.

e. Sniffing Network Communications

Sniffing occurs when an unauthorized third party captures network packets destined for computers other than their own. Packet sniffing allows the attacker to look at transmitted content and may reveal passwords and confidential data.

Specialized packet driver software must be connected to the network segment they want to sniff, and must use sniffer software. By default, a network interface card (NIC) in a computer will usually drop any traffic not destined for it. By putting the NIC in promiscuous mode, it will read any packet going by it on the network wire.

Packet-sniffing attacks are more common in areas where many computer hosts share the same collision domain.

Sniffing and Spoofing-Computer's exchange messages with each other in the form of small groups of data called packets. Packets contains the actual data to be sent & addressing information. Attackers target these packets as they travel from source to destination. Thus, two types of attacks:

Packet Sniffing(snooping)/IP Sniffing – It is a type of passive attack where an attacker need not hijack a conversation but simply observe(sniff) packets as they pass through. To prevent sniffing, the information must be protected by encoding or the transmission link itself is encoded.

Packet Spoofing/IP Spoofing – The attacker sends packets with an incorrect source address. The receiver will receive these packets containing false address and replies back to this forged address (spoofed address) not to an attacker. It will lead to three possible cases:

- If the attacker is between the destination and the forged source, the attacker can see the reply & use that information for hijacking.
- If the attacker's intention was DOS attack, then the attacker need not bother about the reply.

- The attacker could simply be angry with the host, so it may put that host's address as the forged source address & send the packet to the destination. The attacker does not want a reply from the destination, as it wants the host with the forged address to receive it & get confused.

Pharming (DNS Spoofing)-

Domain Name Server maintains the mapping between domain names & the corresponding IP address. Eg- Merchant (Bob) whose sites domain name is **Error! Hyperlink reference not valid.** & The IP address is 100.10.10.20 in the DNS.

Attacker (Tom IP address- 100.20.20.20) manages to hack and replace the IP address of the Bob with its own IP address. Now Bob's IP address is 100.20.20.20 for **Error! Hyperlink reference not valid.**.

Alice will communicate with Bob's site; it will query the DNS & will get Bob's IP address - 100.20.20.20. Hence Alice is communicating with an attacker (Tom) assuming that she is communicating with the Bob.

f. Port Scanning and Web Crawling

It is technique of determining which port on the network is open and which could send and receive data, it is also a process for sending packets to specific ports on a host and analyzing responses to identify vulnerabilities. This scanning take place by first identifying a list of active hosts and mapping those hosts to their IP addresses. The goal behind port and network scanning is to identify the organization of IP addresses, hosts, and ports to properly determine open or vulnerable server locations and diagnose security levels. Port scanning can be used by both IT administrators and cybercriminals to verify or check the security policies of a network and identify vulnerabilities, also exploit any potential weak entry points.

Web Crawling Web crawling is the process of indexing data on web pages by using a program or automated script. These automated scripts or programs are known by multiple names, including web crawler, spider, spider bot, or a crawler.

Web crawlers copy pages for processing by a search engine, which indexes the downloaded pages so that users can search more efficiently. The goal of a crawler is to learn what webpages are about. This enables users to retrieve any information on one or more pages when it's needed.

g. Wildcards

Devices that use multicast communications, such as UPnP, CoAP, anybody can listen and see who sends the messages. Devices that use single-cast communication, such as those using HTTP or CoAP, port-

scanning techniques can be used. For devices that are protected by firewalls and use message brokers to protect against incoming attacks, such as those that use XMPP and MQTT, search features or wildcards can be used to find the identities of devices managed by the broker

HTTP and CoAP that support some level of local client authentication but lacks a good distributed identity and authentication mechanism.

XMPP only permits messages from specific requests that has come from another end. The device which accepts the request is a friend devise or not. This can be either configured by somebody else with access to the account or by using a provisioning server if the device cannot make such decisions by itself. The device does not need to worry about client authentication, as this is done by the brokers themselves, and the XMPP brokers always propagate the authenticated identities of everybody who send them messages.

In MQTT the only way to control who gets access to the data is by building a proprietary end-to-end encryption layer on top of the MQTT protocol, thereby limiting interoperability.

h. Breaking Ciphers

Cryptanalysis is the study of analyzing information systems in order to study the hidden aspects of the systems. Cryptanalysis is used to breach cryptographic security systems and gain access to the contents of encrypted messages, even if the cryptographic key is unknown.

In addition to mathematical analysis of cryptographic algorithms, cryptanalysis includes the study of side-channel attacks that do not target weaknesses in the cryptographic algorithms themselves, but instead exploit weaknesses in their implementation. Attacks can be classified based on what type of information the attacker has available.

Cryptography-An art of achieving security by encoding messages to make them non-readable.

Cryptanalysis- A technique of decoding messages from a non-readable format back to a readable format without knowing how they were initially converted from readable format to a non-readable format.

Plain Text – Message in a plain text can be understood by anybody & it is in human readable form.

Cipher Text –When a plain text message is coded using a suitable scheme, the resulting message is called cipher text.

Encryption(encoding)- It transforms a plain text into a cipher text.

Decryption(decoding) – It transforms a cipher text into a plain text.

Two ways in which a plain text can be coded to obtain the corresponding cipher text - substitution & Transposition.

Eg :Substitution Techniques – The characters of a plain text messages are replaced by other characters, numbers or symbols.

a. **Caesar Cipher – Algorithm**

- Read each alphabet in the cipher text message, & search for it in the second row of the replacement table.
- When a match is found replace that alphabet in the cipher text message with the corresponding alphabet in the same column but the first row of the table
- Repeat the process for all alphabets in the cipher text message.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Eg- Plain Text – HELLO HOW ARE YOU

Cipher Text – KHOOR KRZ DUH BRX.

Eg- Cipher Text – I AM FINE

Plain Text - F XJ CFKB

I. Password Cracking

Password crackers either try to guess passwords or they use brute-force tools. *Brute-force* tools attempt to guess a password by trying all the character combinations listed in an accompanying *dictionary*. The dictionary may start off blindly guessing passwords using a simple incremental algorithm. (For example, trying aaaaa, aaaab, aaaac, and so on) or it may use passwords known to be common on the host (such as password, blank, michael, and so on).

If the attacked system locks out accounts after a certain number of invalid login attempts, some password attackers will gain enough access to copy down the password database, and then brute force it offline.

7.6 TOOLS FOR ACHIEVING SECURITY

There are a number of tools that architects and developers can use to protect against malicious use of the system.

a. Virtual Private Networks

A method that is often used to protect unsecured solutions on the Internet is to protect them using Virtual Private Networks (VPNs). Machine to Machine solutions work well in local intranets that needs to expand across the Internet. One way to achieve this is to create such VPNs that allow the devices to believe they are in a local intranet, even though communication is transported across the Internet. Telephone operators use the

Internet to transport long distance calls, it doesn't make it Voice over IP (VoIP). Using VPNs might protect the solution, but it eliminates the possibility to interoperate with others on the Internet.

A virtual private network, or VPN, is an encrypted connection over the Internet from a device to a network. The encrypted connection helps ensure that sensitive data is safely transmitted. It prevents unauthorized people from eavesdropping on the traffic and allows the user to conduct work remotely. VPN technology is widely used in corporate environments.

How does a virtual private network (VPN) work?

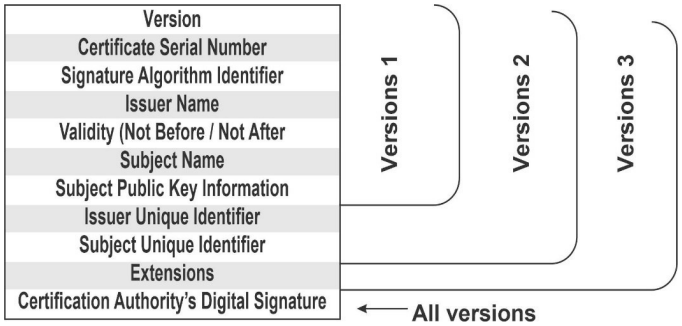
A VPN extends a corporate network through encrypted connections made over the Internet. Because the traffic is encrypted between the device and the network, traffic remains private as it travels. An employee can work outside the office and still securely connect to the corporate network. Even smartphones and tablets can connect through a VPN.

b. X.509 certificates and encryption

The use of certificates to validate the identity of high-value entities on the Internet. Certificates allow you to validate not only the identity, but also to check whether the certificate has been revoked or any of the issuers of the certificate have had their certificates revoked, which might be the case if a certificate has been compromised. Certificates also provide a Public Key Infrastructure (PKI) architecture that handles encryption. Each certificate has a public and private part. The public part of the certificate can be freely distributed and is used to encrypt data, whereas only the holder of the private part of the certificate can decrypt the data. Using certificates incurs a cost in the production or installation of a device or item. They also have a limited life span, so they need to be given either a long lifespan or updated remotely during the life span of the device. Certificates also require a scalable infrastructure for validating them. It is difficult to see that certificates will be used by other than high-value entities that are easy to administer in a network.

Digital Certificate

A standard called X.509 defines the structure of a digital certificate. Figure shows the structure of a X.509 V3 digital certificate.



Field	Description
Version	Identifies a particular version of the X.509 protocol, which is used for this digital certificate. Currently, this field can contain 1, 2 or 3.
Certificate Serial Number	Contains a unique integer number, which is generated by the CA.
Signature Algorithm Identifier	Identifies the algorithm used by the CA to sign this certificate.
Issuer Name	Identifies the Distinguished Name (DN) of the CA that created and signed this certificate.
Validity (Not Before/Not After)	Contains two date-time values (Not Before and Not After), which specify the time frame within which the certificate should be considered valid. These values generally specify the date and time up to seconds or milliseconds.
Subject Name	Identifies the Distinguished Name (DN) of the end entity (i.e. the user or the organization) to whom this certificate refers. This field must contain an entry unless an alternative name is defined in Version 3 extensions.
Subject Public Key Information	Contains the subject's public key and algorithms related to that key. This field can never be blank.

PKIX Services

a. Registration

It is the process where an end-entity (subject) makes itself known to a CA. Usually, this is via an RA.

b. Initialization

This deals with the basic problems, such as who the end-entity is sure that it is talking to the right CA?

c. Certification

In this step, the CA creates a digital certificate for the end-entity and returns it to the end-entity maintains a copy for its own records and copies it in public directories, if required.

d. Key pair recovery

Keys used for encryption may be required to be recovered later for decrypting some old documents. Key archival and recovery services can be provided by a CA or by an independent key recovery system.

e. Key generation

PKIX specifies that the end-entity should be able to generate private and public key pairs or the CA/RA should be able to do this for the end-entity (and then distribute these keys securely to the end-entity).

f. Key update

This allows a smooth transition from one expiring key pair to a fresh one, by the automatic renewal of digital certificates. However, there is a provision for manual digital certificates renewal request and response.

g. Cross-certification

Helps in establishing trust models, so that end-entities that are certified by different CAs can cross-verify each other.

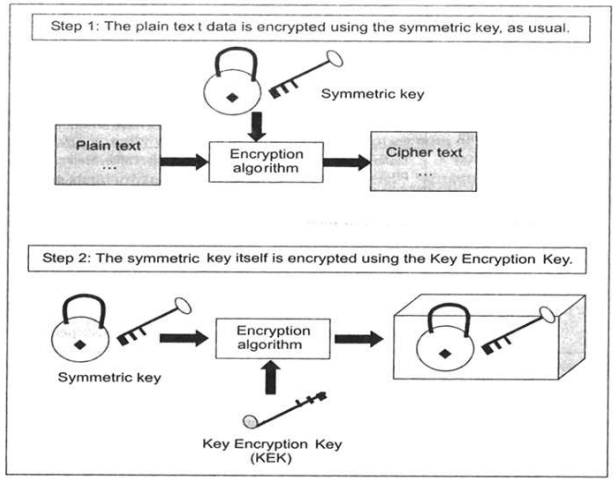
h. Revocation

PKIX provides support for the checking of the certificate status in two modes: online (using OCSP) or offline (using CRL).

Public Key Cryptography Standards (PKCS)

PKCS#5 ¾ Password Based Encryption (PBE) Standard

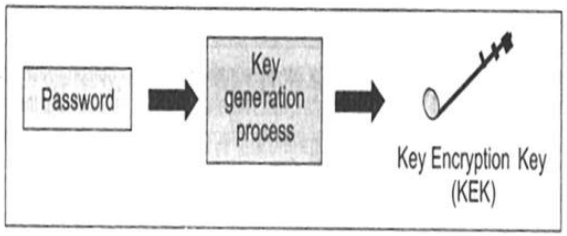
PBE is a solution for keeping the symmetric session keys safe. This technique ensures that the symmetric keys are protected from an unauthorized access. The PBE method uses a password-based technique for encrypting a session key.



first encrypt the plain-text message with the symmetric key, and then encrypt the symmetric key with a Key Encryption Key (KEK). This protects the symmetric key from an unauthorized access.

To protect KEK, never store it anywhere! This will ensure that no one will have access to it.

the approach used in PBE is to generate it on demand, use it for encrypting/decrypting the symmetric key, and then discard it immediately. The password is the input to a key-generation process (usually a message digest algorithm), the output of which is the KEK.



c. Authentication of identities

Authentication is the process of validating whether the identity provided is actually correct or not. Authenticating a server might be as simple as validating a domain certificate provided by the server, making sure it has not been revoked and that it corresponds to the domain name used to connect to the server. Authenticating a client might be more involved, as it has to authenticate the credentials provided by the client. Normally, this can be done in many different ways. It is vital for developers and architects to understand the available authentication methods and how they work to be able to

assess the level of security used by the systems they develop. Some protocols, such as HTTP and XMPP, use the standardized Simple Authentication and Security Layer (SASL) to publish an extensible set of authentication methods that the client can choose from. This is good since it allows for new authentication methods to be added. But it also provides a weakness: clients can be tricked into choosing an unsecure authentication mechanism, thus unwittingly revealing their user credentials to an impostor. MD5-DIGEST, and so on, even if they are the only options available.

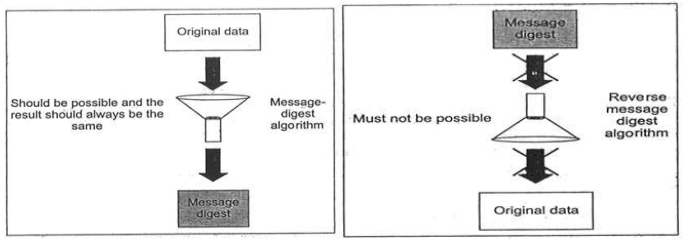
Message Digest

What is a message digest -It is a fingerprint or the summary of a message? It is similar to the concept of Longitudinal Redundancy Check (LRC) or Cyclic Redundancy Check (CRC) that is used to verify the integrity if the data. (i.e., to ensure that a message has not been tampered with after it leaves the sender before it reaches the receiver)

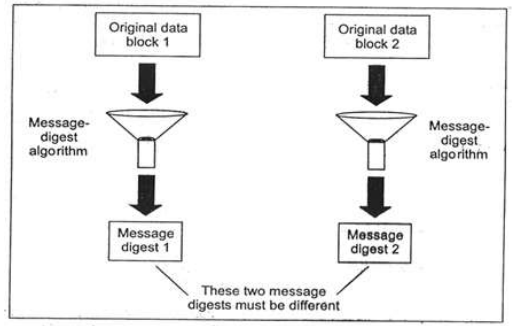
The requirements of the message digest concept, as follows:

(a) Given a message, it should be very easy to find its corresponding message digest. This is shown in Figure, Also, for a given message, the message digest must always be the same.

(b) Given a message digest, it should be very difficult to find the original message for which the digest was created.



(c) Given any two messages, if we calculate their message digests, the two message digests must be different.



MD5 hashing algorithm:

MD5 is quite fast and produces 128-bit message digest.

Secure Hash Algorithm (SHA).

SHA works with any input message that is less than 2 bits in length. The output of SHA is a message digest, which is 160 bits in length (32 bits more than the message digest produced by MD5).

Comparison between MD5 & SHA.

Points	MD5	SHA
Message-digest length in bits	128	160
Attack to try and find the original message given a message digest	Requires 2^{128} operations to break in	Requires 2^{160} operations to break in, therefore more secure
Attack to try and find two messages producing the same message digest	Requires 2^{64} operations to break in	Requires 2^{80} operations to break in
Successful attacks so for	There have been reported attempts to some extent.	No such claims so far.
Speed	Faster (64 iterations and 128 bit buffer.)	Slower (80 iterations, and 160 bit buffer)
Software implementation	Simple, does not need any large programs or complex tables.	Simple does not need any large programs or complex table.

d. Usernames and passwords

A common method to provide user credentials during authentication is by providing a simple username and password to the server. Some solutions use the concept of a pre-shared key (PSK) instead, as it is more applicable to machines. One way to generate secure, difficult-to-guess usernames and passwords is to randomly create them. In this way, they correspond more to pre-shared keys. Both the server and the client need to be aware of this information. The identity must also be distributed among the entities that are to communicate with the device. In the case of XMPP, this problem has been solved. The XMPP Protocol. Here, the device creates its own random identity and creates the corresponding account in the XMPP server in a secure

manner. It then reports its identity to a Thing Registry or provisioning server where the owner can claim it and learn the newly created identity. This method never compromises the credentials and does not affect the cost of production negatively. Furthermore, passwords should never be stored in clear text. Important on servers where many passwords are stored. Instead, hashes of the passwords should be stored. Most modern authentication algorithms support the use of password hashes. Storing hashes minimizes the risk of unwanted generation of original passwords for attempted reuse in other systems.

e. Using message brokers and provisioning servers

Using message brokers can greatly enhance security in an IoT application and lower the complexity of implementation when it comes to authentication, if message brokers provide authenticated identity information in messages it forwards. In XMPP, all the federated XMPP servers authenticate clients connected to them as well as the federated servers themselves when they intercommunicate to transport messages between domains. This relieves clients from the burden of having to authenticate each entity in trying to communicate with it since they all have been securely authenticated. It's sufficient to manage security on an identity level. Even this step can be relieved further using provisioning, as described in Chapter 6, The XMPP Protocol. Unfortunately, not all protocols using message brokers provide this added security since they do not provide information about the sender of packets. MQTT is an example of such a protocol.

f. Centralization versus decentralization

When designing IoT architecture -avoid storing data in a central position if possible. Only store the data centrally which is actually needed to bind things together. Distribute logic, data, and workload. Perform work out of network as far as possible. This makes the solution more scalable, and it utilizes existing resources better. Use the linked data to spread data across the Internet, and use standardized grid computation technologies to assemble distributed data (for example, SPARQL) to avoid the need to store and replicate data centrally. Use a federated set of small local brokers instead of trying to get all the devices on the same broker.

7.7 THE NEED FOR INTEROPERABILITY

Interoperability is the ability of two or more devices, systems, platforms or networks to work in conjunction. Interoperability enables communication between heterogeneous devices or system in order to achieve a common goal. The devices and systems are fragmented with respect to the communication technologies, protocols, and data formats. This makes it difficult for the devices and systems in the IoT network to communicate and share their data with one another.

a. Solves complexity

Those companies that believe they can control the entire value chain, from things to services, middleware, administration, operation, apps. What will happen if they fail to operate. Companies that built devices with protocols, middleware, and mobile phone applications, where human can control things. Imagine a future where you have a thousand different things in your apartment from a hundred manufacturers. Would you want to download a hundred smart phone apps to control them? Would you like five different applications just to control your lights at home, just because you have light bulbs from five different manufacturers? An alternative would be to have one app to rule them based on requirement & feedback. To make it interoperable, they should communicate using a commonly understood language.

b. Reduces cost

Interoperability does not only affect simplicity of installation and management; it also takes care of the price of installation and solution to it. Companies that promote products, where you're forced to use their system to control your devices, can force their clients to pay a high price for future devices and maintenance, or the large investment made originally might be lost. Interoperability provides competition, and competition drives down cost and increases functionality and quality.

c. Allows new kinds of services and reuse of devices

New applications and services will be built that will reuse existing devices, which were installed perhaps as part of other systems and services. These applications will deliver new value to the inhabitants of the city without the need of installing new duplicate devices but such multiple use of devices is only possible if the devices communicate in an open and interoperable way. However, care must be taken at the same time since installing devices in an open environment requires the communication infrastructure to be secure as well. To build smart cities, it is important to use technologies that allow to have both a secure communication infrastructure and an interoperable.

d. Combining security and interoperability

Depending on the communication infrastructure, we might have to use security measures that directly oppose the idea of an interoperable infrastructure, prohibiting third parties from accessing existing devices in a secure fashion. It is important during the architecture design phase, before implementation, to thoroughly investigate what communication technologies are available, and what they provide and what they do not provide. All such implementation is by its very nature proprietary, and therefore not interoperable.

7.8 SUMMARY

In this we talked about the risks involved in IoT, Security and interoperability. There always a risk associated in any kind of work whether it is technical or non-technical. IoT security is the protection of Internet of Things devices from attack. While many business owners are aware that they need to protect computers and phones with antivirus, the security risks related to IoT devices are less well known and their protection is often neglected. While consumer IoT devices allow lifestyle benefits, businesses are quickly adopting IoT devices due to high potential for savings. IoT devices can greatly increase productivity for businesses, they also come with risks. Since IoT devices are connected to the internet, they can be hacked just like any other internet-enabled device. The attack surface of a network consists of all the possible places where it can be attacked, and it expands with every new internet-connected device. Even if the chance of one device being accessed by a perpetrator is small, the large number of IoT devices being brought into businesses can create a significant security risk.

7.9 REFERENCES FOR FUTURE READING

Learning Internet of Things by Peter Waher, Packt Publishing. (All notes are taken from this prescribed reference book).

Notes are taken from the link below:

https://blog.avast.com/iot-security-business-risk.

https://www.cisco.com/c/en_in/products/security/vpn-endpoint-security-clients/what-is-vpn.html.

https://www.imperva.com/learn/application-security/man-in-the-middle-attack-mitm/

7.10 UNIT END EXERCISE

1. What is a risk in IoT. Elaborate with an example.
2. What are the different modes of attacks in the network.
3. What do you mean by interoperability in IoT?
4. What is the need of interoperability in IoT.
5. Explain Denial of Service attack.
6. What do you mean by Man-in-the-Middle attack?
7. Describe in detail port scanning and Web Crawling?
8. How wildcards and breaking cipher helps in attacks in the network.
9. What are the different tools available to achieve security?
10. Explain Virtual Private Network (VNP)? What are its features.

❄❄❄❄❄❄❄

8

INTRODUCTION TO IOT

Unit Structure

8.1 OBJECTIVE

The aim of the Internet of Things to creation network is:

To coordinate and help to increase and optimize the utilization of results and value creation in IoT.

To create innovation strategies for the development of enabling technologies (nano-electronics, embedded systems, communication technologies, software, and cloud computing, etc.) required for IoT applications.

The main purpose of IoT is to create an ecosystem that connects everything. An ecosystem where everything is connected to each other is known as the Internet of Everything.

8.2 INTRODUCTION

Nowadays information technology is developed to be as one part that covers different phases related to the spread of the Internet and the Web into the tangible infrastructure which is called as the Internet of Things. The Internet of Things (IoT) is considered important knowledge that adapted the human to live in a smart and speed life, through enabling things to communicate with other objects, such as machines. IoT describes a system that contains many things that are exiting in the real world, along with sensors attached to them or embedded in these things, and connected to the Internet through a networked structure both as wireless or wired media.

8.3 DEFINITION

The IoT is «an interconnected environment in which all kinds of objects have a digital presence and the ability to communicate with other objects and people. IoT device comes with some common set of features like connectivity, analytics, endpoint management, etc.

The IoT enables the entities to connect and control the IoT devices present in the network by-

1. The entity can use a remote (tablets, smartphones) for sending a command or request for information over an IoT device.
2. The device then performs the command or can even send the information back over the network which has to be analyzed
3. This storing and analyzing of data can be carried out in multiple locations which include- cloud, local database or sometimes the data himself.

8.4 FEATURES OF INTERNET OF THINGS

Any IoT device should have the following features associated with it.

1. Connectivity

IoT devices can be connected over Radio waves, Bluetooth, Wi-Fi, Li-Fi, etc. Various protocols of internet connectivity layers can be used to maximize efficiency and establish connectivity across IoT Industry. Without seamless communication among the interrelated components of the IoT ecosystems (i.e sensors, compute engines, data hubs, etc.) it is not possible to execute any business.

2. Sensing

In the case of IoT we need to read the analog signal, convert it in such a way that we can derive meaningful insights out of it. Use of electrochemical, gyroscope, pressure, light sensors, GPS, Electrochemical, pressure, RFID, etc. to gather data based on an analysis. For example, for automotive use cases, we use Light detection sensors along with pressure, velocity and imagery sensors.

3. Active Engagements

IoT device connects various products, technologies and services work together by establishing an active engagement between them. Cloud computing is used to establish active engagements among IoT components. In the case of Industry grade, IoT takes the raw data which need to be acquired, preprocessed, and rescale as per business capacity. While designing an IoT ecosystems carriers need to consider the future needs of manipulating such a huge scale of data to satisfy incremental business needs.

4. Scale

IoT devices should be designed in such a way that they can be scaled up or down easily on demand based on market or industry standards. In general, IoT is being used from smart home automation to automating large factories and workstations. A carrier should design their IoT infrastructure depending upon their current and future engagement scale.

5. Dynamic Nature

For any IoT use case, the first and important step is to collect and convert data in such a way that means business decisions can be made of it. In this whole process, various components of IoT need to change their state dynamically. For example, the input of a temperature sensor will vary continuously based on weather conditions, locations, etc. IoT devices should be designed this keeping in mind.

6. Intelligence

In IoT, the data is used to make important business insights and drive important business decisions based on the analysis. Various machine learning & deep learning algorithms are built to models on top of this massive data to obtain valuable insights. The analog signals are preprocessed and converted to a format on which machine-learning models are trained.

7. Energy

From end components to connectivity and analytics layers, the whole ecosystems demand a lot of energy. While designing an IoT ecosystem, one need to consider design methodology such that it should be eco-friendly and energy consumption is minimal moreover recycling can also be done.

8. Safety

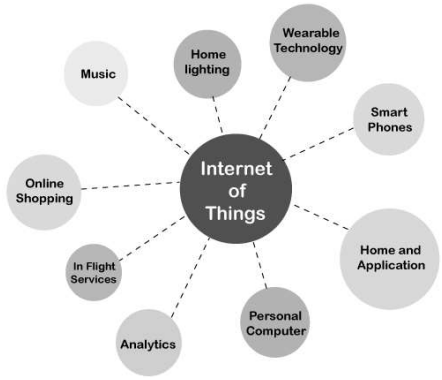
One of the main features of the IoT ecosystem is security. In the network of an IoT system, crucial and sensitive information travels from endpoints to the analytics layer via connectivity components. While designing an IoT system we need to adhere to proper safety, security measures, and firewalls to keep the data away from misuse and manipulations. Compromising any component of an IoT system can eventually lead to failure of the whole system.

9. Integration

IoT integrates various cross-domain models to enrich user experience. It also ensures proper trade-off between infrastructure and operational costs.

8.5 APPLICATIONS OF IOT

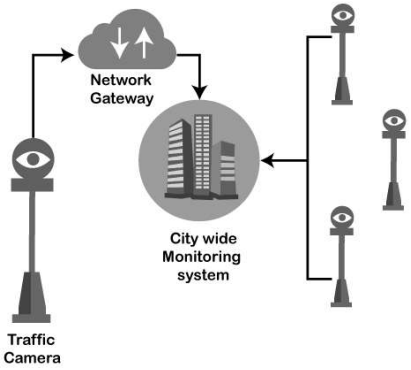
The Internet of Things (IoT) provides the ability to interconnect computing devices, mechanical machines, objects, animals or unique identifiers and people to transfer data across a network without the need for human-to-human or human-to-computer is a system of conversation. IoT applications bring a lot of value in our lives. The Internet of Things provides objects, computing devices, or unique identifiers and people's ability to transfer data across a network without the human-to-human or human-to-computer interaction.



Source : https://www.javatpoint.com/internet-of-things-applications

A traffic camera is an intelligent device. The camera monitors **traffic congestion, accidents** and **weather conditions** and can access it to a common entrance.

This gateway receives data from such cameras and transmits information to the city's **traffic monitoring system**.



Source: https://www.javatpoint.com/internet-of-things-applications

1. **Wearables**

Wearable technology is the hallmark of IoT applications and one of the earliest industries to deploy IoT. Devices are designed for heart rate monitors and smartwatches etc. nowadays.

Guardian glucose monitoring device has been developed to help people with diabetes. It detects glucose levels in our body, uses a small electrode called the glucose sensor under the skin, and relates it to a radiofrequency monitoring device.

2. **Smart Home Applications**

The smart home or smart city talks about the IoT application. AI home automation is employed, home automation system, where a string of musical notes uses in-house functions.

3. **Health care**

IoT applications can transform reactive medical-based systems into active wellness-based systems. Resources uses controlled environments, leftover data, and volunteers for clinical trials. The Internet of Things improves the device's power, precision, and availability.

4. **Smart Cities**

Smart city uses technology of IoT to provide services. The smart city includes improving transportation and social services, promoting stability etc.

Engineers use the Internet of Things to analyze the complex factors of town and each city. IoT applications help in water management, waste control and emergencies.

Eg: Smart city - Palo Alto.

Palo Alto, San Francisco, is the city to acquire the traffic approach. They realized that most cars roam around the same block on the streets in search of parking spots. It is the primary cause of traffic congestion in the city. Thus, the sensors were installed at all parking areas in the city. These sensors pass occupancy status to the cloud of each spot. And hence the vacant space for cars is identified and cars can be parked properly.

5. **Agriculture**

To feed such a large population, agriculture needs to collaborate with technology and get the best results out of it. Eg: Smart Greenhouse where farming techniques grow crops by environmental parameters. Traditional method of manual handling results in production losses, energy losses and labor costs, making it less effective and involves huge amount of loss due to weather conditions. The greenhouse makes it easy to monitor and enables to control the climate inside it.

6. **Industrial Automation**

In industry automation is must where the quality of products is an essential factor for a more significant investment return and good turnover. Anyone can design or re-construct products and their packaging to provide superior quality and performance in cost and customer experience with IoT applications. IoT will prove as a game-changer. In industrial automation, IoT is used in the areas such as:

- Product flow monitoring
- Factory digitization
- Inventory management
- Safety and security
- Logistics and Supply Chain Optimization
- Quality control
- Packaging customization

7. **Hacked Car**

A connected car is a technology-driven car with Internet access and a WAN network. The technology offers the user some benefits such as in-car infotainment, advanced navigation and fuel efficiency.

8. **Healthcare**

Healthcare does real-time monitoring with the help of smart devices. It gathers and transfers health data such as blood pressure, blood sugar levels, weight, oxygen, and ECG. The patient can contact the doctor by the smart mobile application in case of any emergency.

9. **Smart Retail**

IoT applications in retail give shoppers a new experience. Customers do not have to stand in long queues as the checkout system can read the tags of the products and deduct the total amount from the customer's payment app with IoT applications' help.

10. **Smart Supply Chain**

Customers automate the delivery and shipping with a smart supply chain. It also provides details of real-time conditions and supply networks.

11. **Smart Farming**

Farmers can minimize waste and increase productivity. The system allows the monitoring of fields with the help of sensors. Farmers can monitor the status of the area.

8.6 IOT EXAMPLES IN REAL LIFE

1. Sensors

IoT sensors consist of manual or digital sensors connected to circuit boards such as Arduino Uno or Raspberry Pi 2. The circuit boards can be programmed to measure a range of data collected from a sensor device such as carbon monoxide, temperature, humidity, pressure, vibration, and motion. IoT sensors gather data at different physical environments and sends data to the connected devices. They can be used by businesses for predictive maintenance, enhanced efficiency, and reduced costs.

2. Data Analysis

Businesses are increasingly using IoT data analytics to determine trends and patterns by analyzing big and small data. IoT data analytics apps can analyze structured, unstructured, and semi-structured data to extract meaningful insights and predict the result.

IoT can be applied to data analytics to investigate different types of data including motion data sets, geographical data, and health care data. It can be used by businesses for predictive and descriptive analysis to improve customer knowledge, enhance operational efficiency, and create business value.

3. Tracking and Monitoring Systems

A lot of industry such as Amazon, Flip Kart etc. are using IoT systems for asset tracking. IoT asset tracking devices use GPS or radio frequency (RF) to track and monitor the product. The smart devices can be used for long-range identification and verification of assets.

4. Smart Supply Chain Management

Supply chain managers can make improved predictions through smart routing and rerouting algorithms. IoT devices are tagged to packages that can provide instant location finding facility via GPS and RFID signals that can help to make informed supply chain decisions. IoT applications can help in mitigating uncertainty risks in supply chain management. Supply chain managers can make use of smart supply chain management programs for minimizing variance, reducing costs, and improving profitability. The programs can help in inventory management, vendor relationship, fleet management, and scheduled maintenance.

5. Smart Barcode Readers

IoT barcode readers can help in better inventory management for retailers. The readers support AI-based digital signal processing. These devices can optimize the operations of many sectors including retail, logistics, warehouse, and much more.

IoT based bar card readers feature cloud data connections to connect with other systems. Using the connected bar code reader will ease the process of managing inventory.

IoT barcode readers can be incorporated into shopping carts. The readers use AI-based sensor to detect products as they are dropped or removed from the cart. The reader can transfer data to the computer automatically, and that can save a lot of time in checkout resulting in an improved experience for the customers.

6. Smart Grids

The smart grid is another industrial application of IoT. The grid allows real-time monitoring of data regarding supply and demand of electricity. It involves the application of computer intelligence for the efficient management of resources.

Utility companies can use IoT smart grid technologies for more efficient outage management. They can use the technology to identify load distribution and improve reliability. The technology can also assist in fault detection and repairs.

With the smart grid, utilities can interconnect all their assets including meters and substations. Applying IoT technologies to the grid ecosystem allows utility companies to exercise greater control over the power infrastructure and resources. Moreover, they allow consumers with better quality access to energy.

7. Connected HealthCare System

IoT has numerous applications in the healthcare industry. The technology can be used to provide high-quality medical services using smart medical devices.

IoT automates the workflow by allowing the provision of effective health care services to the patients.

IoT medical devices can help in real-time monitoring of patients remotely. The devices can report an emergency like an asthma attack, heart failure, etc., immediately to a physician. This can help in potentially saving the lives of many individuals.

IoT devices can collect health care data including blood pressure, sugar levels, oxygen, and weight. Data is stored online and can be accessed anytime by a physician.

8. Smart Farming

Farmers can use smart IoT farming applications for optimizing a lot of different activities such as determining the best time to harvest plants, creating fertilizer based on the chemicals in the soil, and sensing soil nutrients and moisture levels.

IoT technologies can help in precise farming which can result in optimized production. Some IoT devices and sensors can detect weather conditions and other environmental data. Applications of IoT technologies can help to boost both the quality and quantity of agriculture production.

8.7 SIMPLE IOT LED PROGRAM

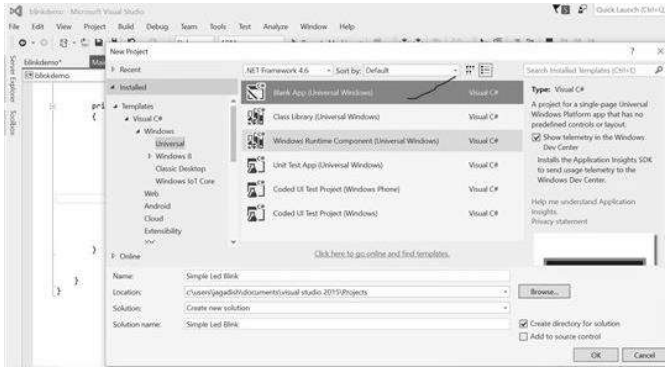
Simple LED Blink example

Aim: Simple experiment of connecting LED lights with Raspberry Pi device (device only with Windows 10 IoT Core OS)

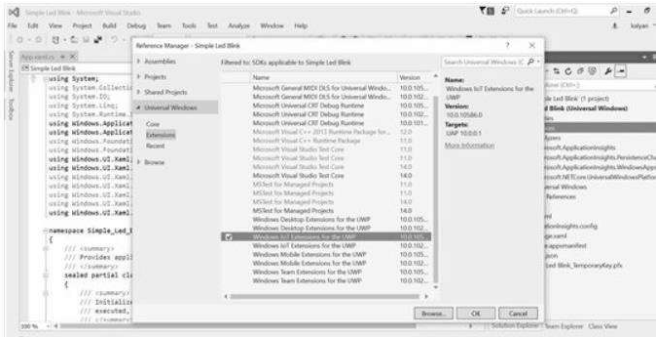
Requirements:

- Raspberry Pi 2/3 with Windows 10 IoT Core OS connected to internet
- Laptop or PC with Windows 10 OS connected to internet
- Visual Studio 2015 installed on your laptop or PC
- Bread Board
- LED Light
- Resistor

Step – 01: Select File -> Create new project -> Name it "Simple Led Blink"



Step – 02: Right click on the selection and add reference. Select Windows IoT Extension for the UWP and click ok.



Step – 03: Open MainPage.xaml and add the following for UI

```
<Grid Background="Wheat">

   <StackPanel HorizontalAlignment="Center"
VerticalAlignment="Center">

     <Ellipse x:Name="LED" Fill="LightGray" Stroke="White"
Width="100" Height="100" Margin="10"/>

     <TextBlock x:Name="DelayText" Text="500ms" Margin="10"
TextAlignment="Center" FontSize="26.667" />

     <TextBlock x:Name="GpioStatus" Text="Waiting to initialize
GPIO..." Margin="10,50,10,10" TextAlignment="Center"
FontSize="26.667" />

   </StackPanel>

 </Grid>
```

Step – 04: Open the MainPage.xaml.cs add the following in the cs file.

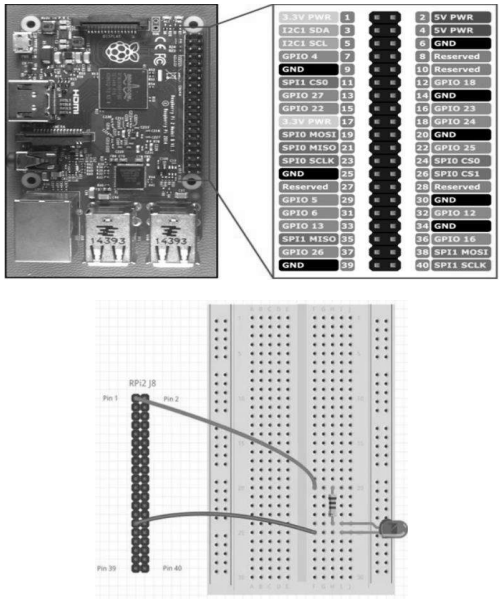
```
private const int LED_PIN =5;

 private GpioPin pin;

 private GpioPinValue pinValue;

 private DispatcherTimer timer;

 private SolidColorBrush redBrush = new
SolidColorBrush(Windows.UI.Colors.Red);

 private SolidColorBrush grayBrush = new
SolidColorBrush(Windows.UI.Colors.LightGray);

  public MainPage()

  {

   InitializeComponent();
```

```
  timer = new DispatcherTimer();
  timer.Interval = TimeSpan.FromMilliseconds(500);
  timer.Tick += Timer_Tick;
  InitGPIO();
  if (pin != null)
  {
    timer.Start();
  }
}
private void InitGPIO()
{
  var gpio = GpioController.GetDefault();

  // Show an error if there is no GPIO controller
  if (gpio == null)
  {
    pin = null;
    GpioStatus.Text = "There is no GPIO controller on this device.";
    return;
  }
  pin = gpio.OpenPin(LED_PIN);
  pinValue = GpioPinValue.High;
  pin.Write(pinValue);
  pin.SetDriveMode(GpioPinDriveMode.Output);
   GpioStatus.Text = "GPIO pin initialized correctly.";
}
private void Timer_Tick(object sender, object e)
{
  if (pinValue == GpioPinValue.High)
```

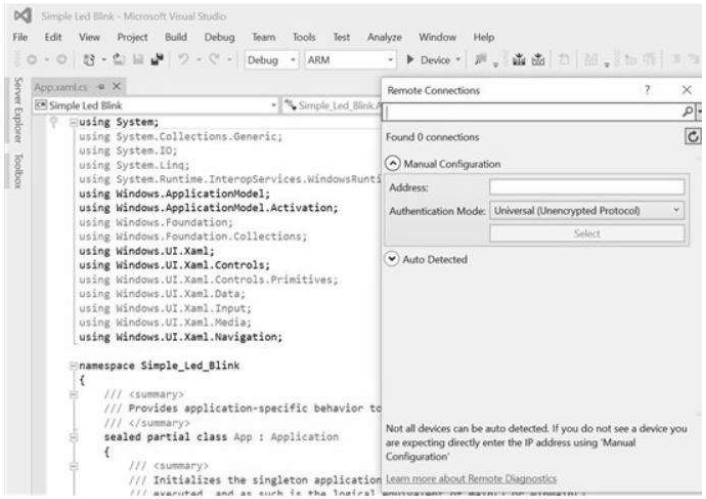


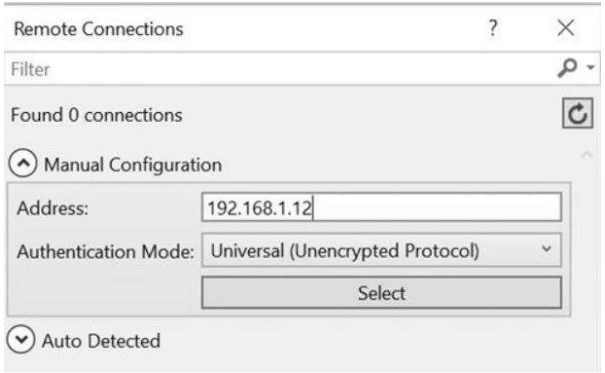
```
  {
    pinValue = GpioPinValue.Low;
    pin.Write(pinValue);
    LED.Fill = redBrush;
  }
  else
  {
    pinValue = GpioPinValue.High;
    pin.Write(pinValue);
    LED.Fill = grayBrush;
  }
 }

}
```

Use GPIO (General Purpose Input Output) 5 over here



Step – 05: Now build the solution and select ARM and select Remote Machine, In the address column enter the Ip address of the network connected with Raspberry Pi device and click ok.





Step – 06: Now once Remote Machine is selected build and run the application. Once the application is build & run operation is completed successfully in the sense you will find the LED blinking according to the timer set.

Source:
https://social.technet.microsoft.com/wiki/contents/articles/33948.iot-simple-led-blink-example.aspx

8.8.SUMMARY

The Internet of Things (IoT) provides the ability to interconnect computing devices, mechanical machines, objects, animals or unique identifiers and people to transfer data across a network without the need for human-to-human or human-to-computer is a system of conversation.

IoT applications bring a lot of value in our lives. The Internet of Things provides objects, computing devices, or unique identifiers and people's ability to transfer data across a network without the human-to-human or human-to-computer interaction.

IoT is used in real life applications such as –health monitoring systems, whether forecasting, tracking of the product, inventory management, smart grid and many more.

8.9.REFERENCES FOR FUTURE READING

Notes taken from the link below:

https://www.javatpoint.com/internet-of-things-applications

https://www.educba.com/iot-features/

https://www.softwaretestinghelp.com/best-iot-examples/#10_Best_Real-World_IoT_Examples

https://social.technet.microsoft.com/wiki/contents/articles/33948.iot-simple-led-blink-example.aspx

8.10 UNIT END EXERCISE

What is IoT?

What are the features of IoT?

Explain the applications of IoT

With an example explain the real-life application of IoT?

Elaborate the process of IoT in supply chain management system?

Explain how IoT is applicable to medical industry?

Write a note on “Improvement in traffic congestion relief using IoT?

❄❄❄❄❄❄❄